

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

ChaosAthena: A Framework to Compare Gen AI Model States — A Post-Training Quantization Case Study

Flávio Lobo Vaz



Mestrado em Engenharia Informática e Computação

Supervisor: Daniel Castro Silva

Second Supervisor: Isabel Praça

Third Supervisor: Sinan Wannous

July 31, 2026

ChaosAthena: A Framework to Compare Gen AI Model States — A Post-Training Quantization Case Study

Flávio Lobo Vaz

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Rosaldo Rosseti

External Examiner: Prof. Pedro Abreu

Supervisor: Prof. Daniel Castro Silva

Second Supervisor: Prof. Isabel Praça

Third Supervisor: Sinan Wannous

July 31, 2026

Abstract

The Deployment of generative language models on resource-constrained edge devices depends on compression, and post-training quantization is among the principal methods. Its consequences, however, are assessed in silos: capability benchmarks, adversarial red-teaming, and theory-driven structural diagnostics run in isolation, so no single artifact records what a given deployable model state actually is.

This dissertation addresses that gap through ChaosAthena, an extensible, hardware-portable evaluation framework integrating capability benchmarking, OWASP-mapped adversarial red-teaming, and data-free Heavy-Tailed Self-Regularization (HTSR) and SETOL weight-matrix diagnostics as interchangeable Stages. A fully crossed case study covers **28 deployment-ready model states** — seven open-weight families under eight billion parameters, each at FP16 and at three GGUF levels, probed by 39 distinct attacks.

Four findings emerge under false-discovery-rate control. *Capability degrades gradually and then collapses*: pooled accuracy falls -0.6 , -1.5 and -5.2 pp at 4-, 3- and 2-bit, concentrated in multi-step reasoning. *The learned spectrum is preserved*: the heavy-tail exponent α never leaves the well-trained band and is an architecture-family property ($\eta^2 = 0.956$). *Safety separates into two structurally distinct axes*: aggregate adversarial robustness is near-invariant, whereas intrinsic toxicity-safety degrades in 20 of 21 compressed states. *Capability and adversarial safety are statistically decoupled* ($r \approx -0.03$, n.s.), so retained benchmark scores do not imply a retained safety profile.

Furthermore, a set of twenty hypotheses, specified before their coefficients were computed, returns eight supported claims. Resolved *per attack* rather than per aggregate, weight structure indexes robustness in **26 of 36 testable probes**, and the sign of each coupling is a property of the attack family rather than of the metric: jailbreak and injection probes invert one another on the shape metrics, while encoding and obfuscation attacks are indexed only by scale and richness quantities. Because opposing families cancel, **a strong structure–vulnerability relationship can be invisible in an aggregate score**.

These results yield Pareto guidance identifying Q4_K_M as the preferred operating point and support the central claim: **each compressed state must be evaluated as a distinct state in its own right**, since trust is neither inherited from a model family nor inferred from a benchmark score. The same append-only schema that carries these results extends to directions the case study only begins to explore: structure-guided attack creation and synthesis, spectrally guided compression, data-free provenance screening, and evidence-grounded agent validation.

Keywords: Generative Artificial Intelligence, Large Language Models, Model Quantization, Edge Computing, Red Team AI, Heavy-Tailed Self-Regularization, Multi-Perspective Evaluation, Trustworthy AI

ACM Classification:

• General and reference → Cross-computing tools and techniques → Evaluation

- Computing methodologies → Machine learning → Machine learning approaches → Neural networks
- Security and privacy → Software and application security
- Theory of computation → Theory and algorithms for application domains → Machine learning theory
- Software and its engineering → Software organization and properties → Software system structures → Software architectures
- Computing methodologies → Artificial intelligence → Natural language processing

Resumo

A implantação de modelos generativos de linguagem em dispositivos com recursos limitados depende de compressão, sendo a quantização pós-treino um dos principais métodos. As suas consequências são, contudo, avaliadas de forma isolada: os testes de capacidade, o *red-teaming* adversarial e os diagnósticos estruturais são executados separadamente, pelo que nenhum artefacto único regista o que um estado implantável do modelo efetivamente é.

Esta dissertação responde a essa lacuna através de ChaosAthena, uma *framework* de avaliação extensível e portátil em *hardware*, que integra a avaliação de capacidade, o *red-teaming* adversarial mapeado ao OWASP e os diagnósticos de matrizes de pesos sem acesso a dados, baseados em *Heavy-Tailed Self-Regularization* (HTSR) e SETOL, sob a forma de *Stages* permutáveis. Um estudo de caso totalmente cruzado abrange **28 estados de modelo preparados para implantação** — sete famílias de pesos abertos com oito mil milhões de parâmetros ou menos, cada uma em FP16 e em três níveis de quantização GGUF — testados por 39 ataques adversariais distintos, daí resultando um conjunto de dados multiperspetiva indexado por estado que avaliações futuras podem estender sem modificar a *framework*.

Ademais, observam-se quatro resultados sob controlo da taxa de falsas descobertas. *A capacidade degrada-se gradualmente e depois colapsa*: a *accuracy* agregada cai $-0,6$ pp a 4 *bits*, $-1,5$ pp a 3 *bits* e $-5,2$ pp a 2 *bits*, com a perda concentrada no raciocínio de múltiplos passos. *O espectro aprendido é preservado*: o expoente de cauda pesada α nunca abandona o intervalo de treino ideal e comporta-se como uma propriedade da família arquitetural ($\eta^2 = 0,956$). *A segurança separa-se em dois eixos estruturalmente distintos*: a robustez adversarial agregada é praticamente invariante sob compressão, ao passo que a segurança intrínseca face à toxicidade se degrada em 20 dos 21 estados comprimidos. *A capacidade e a segurança adversarial encontram-se estatisticamente desacopladas* ($r \approx -0,03$, n.s.), sustentadas por dois blocos disjuntos de métricas estruturais, pelo que a manutenção dos resultados de referência não implica a manutenção do perfil de segurança.

Um conjunto de vinte hipóteses, especificadas antes do cálculo dos respetivos coeficientes, devolve oito alegações sustentadas. Analisada *por ataque* e não de forma agregada, a estrutura de pesos indexa a robustez em **26 dos 36 ataques testáveis**, sendo o sinal de cada acoplamento determinado pela família de ataque; uma vez que famílias opostas se anulam na agregação, **uma relação forte entre estrutura e vulnerabilidade pode tornar-se invisível numa métrica agregada**. Estes resultados produzem orientação segundo o critério de Pareto, identificando Q4_K_M como o ponto de operação preferível, e sustentam a principal alegação: **cada estado comprimido deve ser avaliado como um estado distinto e por direito próprio**, dado que a confiança não se herda da família do modelo nem se infere de um *benchmark* de referência. O mesmo esquema extensível que sustenta estes resultados abre direções que o estudo de caso apenas iniciou a explorar: a síntese e criação de ataques guiada pela estrutura, a compressão guiada espectralmente, a triagem de proveniência sem acesso a dados e a validação de agentes fundamentada em evidência.

Palavras-chave: Inteligência Artificial Generativa, Modelos de Linguagem de Grande Escala,

Quantização de Modelos, Computação de Extremo, Red Team AI, Auto-Regularização de Cauda Pesada, Avaliação Multiperspetiva, Inteligência Artificial Fiável

ACM Classification:

- General and reference → Cross-computing tools and techniques → Evaluation
- Computing methodologies → Machine learning → Machine learning approaches → Neural networks
- Security and privacy → Software and application security
- Theory of computation → Theory and algorithms for application domains → Machine learning theory
- Software and its engineering → Software organization and properties → Software system structures → Software architectures
- Computing methodologies → Artificial intelligence → Natural language processing

Acknowledgments

First, I want to thank my supervisor, Professor Daniel Castro Silva, whose patience, rigor, and constant presence have been invaluable throughout this journey. I am grateful for the time and the care he gave this work, and to my co-supervisors for their availability.

To my parents, I owe more than I could ever say. Amid all the difficulties life placed before them, they gave me complete freedom to explore the world from a very young age. They taught me the value of knowledge and of life itself, and without knowing it, to incorporate, as Richard Feynman put it: "to study hard what interests me most, in the most undisciplined and irreverent way I could find, and to meet the world with passion and embracing my divergence — carrying that divergence as they themselves had to, always through work and dedication, whatever the task. Whatever I have achieved, and whatever lies ahead, carries their hand in it.

My appreciation extends to the friends and teachers who made everything possible, in ways both direct and indirect — for the conversations, the questions, and everything I learned along the way.

To my daughter Athena, thank you for the creative chaos you bring, and for the tenderness with which you lift my spirit every day. Thank you for reminding me that creativity begins in disorder, and in a visceral curiosity about everything.

Last, and most of all, I want to thank Sandra, my love and my partner in every struggle, who carried this work through its most difficult days, often without knowing it. She has been my steadiness and my reason to keep going, and I count myself fortunate every single day to have her beside me.

Flávio Lobo Vaz

‘With my toes on the edge it’s such a lovely view...’

Queens of the Stone Age

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	2
1.3	Research Questions	4
1.4	Contributions	5
1.5	Document Structure	8
2	Background	11
2.1	From Deep Learning to Generative AI and LLMs	11
2.2	Fog-Edge Computing and the Web of Agents	13
2.3	Quantization Fundamentals	13
2.4	Spectral Analysis and Probabilistic Background	14
2.5	Capability Evaluation	14
2.6	Criticality and Heavy-Tailed Self-Regularization	15
2.7	Generative AI Security	19
2.8	Statistical Inference and Exploratory Data Analysis	21
2.9	Framework Design Patterns	22
3	State-of-the Art	25
3.1	Compression of GenAI Models	25
3.2	Model Compression	26
3.2.1	Pruning, Knowledge Distillation, and Low-Rank Factorization	27
3.2.2	Quantization	27
3.2.3	Discussion	34
3.3	Evaluation Bias and Multi-perspective assessment	35
3.3.1	Evaluation Bias	35
3.3.2	Towards a Multi-Perspective Assessment	37
3.3.3	Summary	39
3.4	Research Gap	39
3.5	Existing Opportunities	40
4	Proposed Solution	43
4.1	Problem Setting	43
4.1.1	Formalization	44
4.1.2	Compression Operator and Variant Acquisition	44
4.2	Observational Points and Model Assessment	45
4.2.1	Weight Matrices — HTSR and SETOL	45
4.2.2	Hidden Representations	46

4.2.3	Input–Output Assessment	46
4.2.4	Observational Regimes	47
4.3	Proposed Solution	48
4.3.1	Main Stages and Modules	48
4.3.2	Extensibility of Stages and Modules	49
4.3.3	Architecture Framework by Phase	50
4.3.4	Orchestration and Migration	51
4.3.5	Architecture Diagrams	51
4.3.6	System Requirements	54
4.3.7	Evaluation Criteria	57
5	ChaosAthena Implementation	59
5.1	Core Architecture and Design Patterns	59
5.1.1	The Installable Python Package	60
5.1.2	Configuration and Orchestration with Hydra	60
5.1.3	Storage-Driven Decoupling and the Artifact Substrate	61
5.1.4	Execution Mode and Compute Backends	61
5.1.5	Design Patterns	62
5.1.6	Repository and Directory Structure	63
5.2	Integrated Frameworks and Tools	64
5.2.1	Garak	64
5.2.2	lm-evaluation-harness	65
5.2.3	WeightWatcher	66
5.2.4	Inference Engines	67
5.2.5	Quantization Engines	68
5.3	Phase 1 Implementation	68
5.3.1	GGUF Format Quantization	69
5.3.2	Example of The Retrieve-or-Compute Path	69
5.4	Phase 2 Implementation	72
5.4.1	Performance Evaluation Stage	72
5.4.2	Red Team AI Evaluation Stage	72
5.4.3	Deep Learning First-Principles Stage	75
5.4.4	Example of configuration in and Structural Artifacts out	78
5.5	Phase 3 Implementation — Multi-Perspective Dataset Creator	80
5.5.1	Multi-Perspective Dataset Creator Stage	80
5.6	Phase 4 Implementation — Analytics and Results	81
5.7	Final User Perspective	82
6	Experiments and Results	85
6.1	Scope and Boundary Conditions	85
6.2	Experimental Setup	86
6.2.1	Experimental overview	87
6.2.2	Compute infrastructure and hardware backends	87
6.2.3	Model cohort under evaluation	88
6.2.4	Phase 1 — Compression and Variant Acquisition Setup	90
6.2.5	Phase 2 — Evaluation and Analysis	91
6.2.6	Phase 3 — Multi-Perspective Dataset Setup	97
6.2.7	Phase 4 — Analytics Setup	98
6.3	Results	99

6.3.1	Statistical conventions and reproducibility	99
6.3.2	Capability degrades gradually, then collapses at 2-bit	104
6.3.3	Weight structure stays well-trained and shifts at the output layer	108
6.3.4	Aggregate adversarial resilience is invariant, but specific attacks move	112
6.3.5	Capability and safety are statistically decoupled across views	119
6.3.6	Damage localizes by depth: capability at the output, toxicity in mid-network	124
6.3.7	Which structural changes track which vulnerabilities	127
6.3.8	Pareto-optimal precision and the deployment decision	129
6.3.9	Exploratory results	130
6.4	Key Findings and Discussion	153
6.4.1	Capability degrades gradually, then collapses at 2-bit	153
6.4.2	The learned spectrum is preserved and distortion concentrates at the network extremes	153
6.4.3	Capability and adversarial robustness are decoupled, and the decoupling is structural	153
6.4.4	Safety has two structurally distinct axes, and only one is legible in the spectrum	154
6.4.5	Attack-level movement and its structural explanation	154
6.4.6	Findings from the exploratory analysis	156
6.4.7	Deployment implications	157
6.4.8	Exploratory directions and the evidence they support	157
6.5	Research Questions Validation	158
6.5.1	RQ1. How can a multi-perspective evaluation framework be architected to be modular, scalable, and extensible to new model states without modification of its core, and instantiated — as a case study — in independent stages that compare generative model states under compression?	158
6.5.2	RQ2. How can data-free weight-matrix metrics be aggregated with capability and adversarial red-teaming measurements into a multi-perspective dataset, indexed by model state, that future evaluations can extend?	159
6.5.3	RQ3. How can Exploratory Data Analysis, statistical testing, or Instance Space Analysis relate the HTSR and SETOL weight-matrix properties of diverse architectures and model states to capability performance and adversarial vulnerabilities, and what statistically significant properties or edge-case conditions emerge?	159
6.5.4	RQ4. To what extent can these analytical modules predict or explain the degradation of a generative agent’s performance and security under quantization, specifically its resilience to critical OWASP Top 10 LLM vulnerabilities in future Fog–Edge multi-agent settings?	161
6.5.5	RQ5. By synthesizing RQ1–RQ4, how can the framework contribute, through concrete exploratory and deployment insights, or future work, to trustworthy agents in decentralized Fog–Edge systems within the NANDA Web of Agents?	163
6.6	System Requirements Validation	165
6.6.1	Functional requirements	165
6.6.2	Non-functional requirements	166
7	Conclusions and Future Work	171
7.1	Conclusions	171
7.2	Limitations	172

7.2.1	Generation regime and measurement windows	172
7.2.2	Statistical Power and the robustness of inferential conclusions	173
7.2.3	Compression axis and architectural scope	173
7.2.4	Structural resolution	174
7.2.5	Data governance and deployment	174
7.2.6	The status of the exploratory results	175
7.3	Future Work	175
7.3.1	Engineering roadmap	176
7.3.2	Analytical directions	176
7.3.3	Structure-guided adversarial probe synthesis	176
7.3.4	Validating the exploratory results	177
7.3.5	Spectrally guided compression: a concrete opportunity	178
A	Appendix	181
A.1	Annotated repository structure of the ChaosAthena framework	181
A.2	Example of Notebook Execution	182
A.2.1	Deep Learning First-Principles Stage	182
A.3	Reference Models included in the garak Z-Score Calibration Bag (Winter 2026)	185
B	Appendix	187
B.1	Experimental Setup	187
B.1.1	Execution Parameters (Performance, Red Team AI Stages)	187
B.1.2	Red Team AI Stage parameters	189
B.1.3	Executed Probes Mapped to OWASP-2025 Categories	190
B.2	Results	192
B.2.1	Can specific attacks be associated with structural change?	192
B.2.2	Probe-level robustness check on the vulnerability–structure coupling	192
B.3	Layers depth and mean changes	195
B.4	Quantized model on Hugging Face	202
	References	203

List of Figures

2.1	The decoder-only Transformer	12
2.2	Phase diagram of complex systems	16
2.3	ESDs of a well-trained versus a trapped layer	17
2.4	Per-layer α against the well-trained band	18
2.5	How WeightWatcher analyses DNN weights	20
2.6	Red Team AI and Blue Team AI around a deployed model	21
2.7	Precision, recall, and F_1 as an archery target	22
2.8	Design patterns instantiated in ChaosAthena framework	23
2.9	Configuration-driven, storage-mediated coupling	24
4.1	ChaosAthena architecture organized by phases	52
4.2	ChaosAthena main pipeline, stages, and modules	53
6.1	Per-family accuracy retention from FP16 to each level.	106
6.2	Mean accuracy change by task domain at the most aggressive level (Q2_K_L).	106
6.3	WikiText perplexity by quantization level	107
6.4	Degradation versus model scale at 2-bit	108
6.5	Distribution of alpha by level	109
6.6	Correlation among HTSR/SETOL metrics (redundancy from structure metrics).	110
6.7	Per-family alpha across precision levels and variance	112
6.8	Mean pass-rate by OWASP category and level.	114
6.9	Distribution of lowest DEFCON triggered across states.	114
6.10	Probe-level robustness shifts under quantization	116
6.11	Probe robustness shifts described by quantization level.	117
6.12	Cross-view correlation map across the three measurement views.	122
6.13	Spectral entropy versus accuracy	123
6.14	Capability change versus robustness change	123
6.15	Mean absolute alpha shift by depth position	125
6.16	Depth-wise alpha shift versus outcome change	126
6.17	Structural metrics versus red-team pass rate	129
6.18	Capability $\times$ robustness Pareto view across levels.	130
6.19	Hypothesis battery: effect sizes with confidence intervals	132
6.20	Comprehensive structure-to-behavior screen	137
6.21	Per-OWASP pass rate against structural metrics	139
6.22	Output-handling and Model-DoS probes against weight structure	140
6.23	Per-attack structural screen	145
6.24	Non-monotonic pattern between spectral norm and robustness	148
6.25	Within-model versus between-model structural distance	149

6.26	Overtrained layers versus perplexity: a degenerate predictor	152
6.27	Model size versus stable rank: collinearity	152
A.1	Annotated repository structure of the ChaosAthena framework.	181
B.1	Complete per-probe Spearman association	193
B.2	Per-probe coupling with detectors averaged	194
B.3	Layer depth structural metric levels (α , $\hat{\alpha}$, <code>rand_distance</code>) by model family across quantization regimes (Pages 1–4 of 7).	196
B.4	Layer depth structural metric levels (bins) by model family (Pages 5–7 of 7).	197
B.5	Compression-induced metric deltas ($\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$) relative to FP16 baselines by model family (Pages 1–4 of 7).	199
B.6	High-resolution architectural diagnostic profile extracted by the Veritas Analytics stage	201
B.7	Hugging Face Hub model repository interface for the <code>GuardReasoner</code> GGUF artifact	202

List of Tables

2.1	Interpretation of the HTSR power-law exponent α	19
3.1	Hardware support of the two dominant INT4 weight-only GPU serving formats .	32
3.2	Comparison of the four principal PTQ approaches on the axes that govern Fog-Edge deployment	34
4.1	Functional requirements	54
4.2	Non-functional requirements	56
5.1	Compute backends targeted during experimentation	61
5.2	Orchestration and shared components.	63
5.3	Implemented stages and underlying software/statistical module integrations. . . .	64
5.4	<code>garak</code> configuration: example of parameters and descriptions.	65
5.5	<code>lm-evaluation-harness</code> configuration: example of parameters and descriptions.	66
5.6	<code>WeightWatcher</code> configuration: example of parameters and descriptions.	67
5.7	Inference backends: example of configuration parameters and descriptions. . . .	68
5.8	Quantization engines: example of configuration parameters and descriptions. . .	69
5.9	GuardRail safety classification: example configuration and datasets.	75
5.10	HTSR and SETOL weight-matrix metrics extracted by the Stage.	76
5.11	Aggregated multi-perspective dataset schema: base vs. compressed-variant deltas.	81
5.12	Veritas Analytics Stage: example of configuration parameters and descriptions. .	81
6.1	Experimental matrix: phases, stages, and outputs	87
6.2	Compute Backends Used in Experimentation	88
6.3	Full-Precision Baseline Models	89
6.4	Compressed-Variant Families Overview	90
6.5	Mixed-Precision K-Quantization Regimes	91
6.6	Execution Parameters for the Compression Stage	92
6.7	Experimental Execution Parameters, Tasks, and Families for Performance Evaluation	93
6.8	Experimental Execution Parameters for the WeightWatcher / HTSR-SETOL Stage	96
6.9	Phase-3 Artifacts Produced and Main Datasets Column Definitions	98
6.10	Experimental Execution Parameters for Veritas Analytics	99
6.11	Evaluation design and measurement coverage	100
6.12	Summary of every FDR-significant result in this chapter	101
6.14	Aggregated Capability Changes under Quantization	105
6.15	WikiText perplexity by family and level	105
6.16	Perplexity versus Weight-Structure Correlations	107
6.17	Shape metric alpha at FP16 and each level	109

6.18	Variance decomposition of alpha by family	111
6.19	Per-family alpha across precision levels	111
6.20	Red-team summary by family and level	113
6.21	Probe $\times$ detector pairs with the largest pass-rate shift	115
6.22	Top probe movers resolved by quantization level	116
6.23	Distinct attacks that move under compression	118
6.24	Toxicity-safety versus weight structure	119
6.25	Unified per-state matrix, complete-case cohort	120
6.26	Predictivity of data-free structural metrics	121
6.27	Depth-positional structural shift	124
6.28	Depth-resolved structure-outcome coupling	126
6.29	Vulnerability-structure coupling per probe	127
6.30	Weight Structure Predictivity for Adversarial Robustness	128
6.31	Decision summary per quantization level	130
6.32	Pre-specified exploratory hypothesis battery (20 tests, one BH-FDR family)	133
6.33	Comprehensive structure-to-behavior screen (levels, n=28)	135
6.34	Comprehensive structure-to-behavior screen (compression deltas, n=21)	136
6.35	Per-OWASP pass rate versus structural metrics	138
6.36	LLM02/04/05 sub-probes versus weight structure	139
6.37	Sub-probes versus compression-induced change	140
6.38	Which structural metrics index which attacks (level, n=28)	142
6.40	Per-attack structural screen, compression deltas (n=21)	144
6.41	Inverted-U test: quadratic OLS of pass rate on spectral norm	147
6.42	Structural same-model signature	149
6.43	Dual-coefficient synthesis of the central screen	150
6.44	Structural fingerprint test for unsafe models	151
6.45	Evidence extract supporting RQ5	164
6.46	Framework results mapped to the NANDA operational pillars	168
6.48	Validation of the functional requirements (Table 4.1).	169
6.49	Validation of the non-functional requirements (Table 4.2).	169
A.1	Reference Models included in the garak Z-Score Calibration Bag (Winter 2026)	185
B.1	Execution parameters for the Capability Performance Evaluation Stage	187
B.2	Execution parameters for the Red Team AI Stage	189
B.3	Executed Probes and Sub-Probes Mapped to OWASP-2025 Categories	190
B.4	Probe-level robustness check on the coupling analysis	192

Abbreviations

AGI	Artificial General Intelligence
AI	Artificial Intelligence
ANSI	American National Standards Institute
API	Application Programming Interface
ASI	Artificial Superintelligence
AWQ	Activation-aware Weight Quantization
BF16	Brain Floating Point, 16-bit (bfloat16)
BH-FDR	Benjamini–Hochberg False Discovery Rate
CI	Confidence Interval
CLI	Command-Line Interface
CPU	Central Processing Unit
CSV	Comma-Separated Values
CUDA	Compute Unified Device Architecture
DAMA-DMBOK	Data Management Association’s Data Management Body of Knowledge
DAN	“Do Anything Now” (jailbreak prompt family)
DEFCON	Defense Readiness Condition (garak severity tiers)
DLFP	Deep Learning First-Principles (Stage)
DNS	Domain Name System
EDA	Exploratory Data Analysis
ERG	Exact Renormalization Group
ESD	Empirical Spectral Density
F1	F1-score (harmonic mean of precision and recall)
FDR	False Discovery Rate
FFN	Feed-Forward Network
FP16	16-bit Floating Point (half precision)
FR	Functional Requirement
GenAI	Generative Artificial Intelligence
GGUF	GPT-Generated Unified Format
GPT	Generative Pre-trained Transformer
GPTQ	Generative Pre-trained Transformer Quantization
GPU	Graphics Processing Unit
HQQ	Half-Quadratic Quantization
HTML	HyperText Markup Language
HTSR	Heavy-Tailed Self-Regularization
ID	Identifier
INT4/INT8	4-bit / 8-bit Integer (data types)
IRT	Item Response Theory
ISA	Instance Space Analysis

ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
ITL	Inter-Token Latency
JSON	JavaScript Object Notation
JSONL	JSON Lines
KL	Kullback–Leibler (divergence)
KS	Kolmogorov–Smirnov
KV	Key–Value (attention cache)
LASER	LAYER-SElective Rank reduction
LLM	Large Language Model
LZ	Lempel–Ziv (complexity)
MCQ	Multiple-Choice Question
MIT	Massachusetts Institute of Technology
MLP	Multi-Layer Perceptron
MMLU	Massive Multitask Language Understanding
MoSCoW	Must have, Should have, Could have, Won't have (prioritization)
MP	Marchenko–Pastur
NANDA	Networked AI Agents in Decentralized Architecture
NATO	North Atlantic Treaty Organization (phonetic alphabet)
NFR	Non-Functional Requirement
NLP	Natural Language Processing
OOM	Out Of Memory
OS	Operating System
OWASP	Open Worldwide Application Security Project
PCA	Principal Component Analysis
PL	Power Law
PoC	Proof of Concept
PPL	Perplexity
PTQ	Post-Training Quantization
QAT	Quantization-Aware Training
RAM	Random-Access Memory
RMT	Random Matrix Theory
RQ	Research Question
SD	Standard Deviation
SETOL	Semi-Empirical Theory of Learning
SSRF	Server-Side Request Forgery
StatMech	Statistical Mechanics
SVD	Singular Value Decomposition
TCS	Trust & Capability Score
TPOT	Time Per Output Token
TTFT	Time To First Token
VRAM	Video Random-Access Memory
XAI	Explainable Artificial Intelligence
XSS	Cross-Site Scripting
YAML	YAML Ain't Markup Language

Chapter 1

Introduction

1.1 Context

Artificial intelligence (AI) is the field concerned with building systems that perform tasks held to require intelligence [1]. Its modern era is dominated by deep learning: artificial neural networks whose mathematical roots reach back centuries—the chain rule of 1676, the Gauss–Legendre least-squares method of circa 1800, itself mathematically a two-layer linear network, and the first working deep networks of 1965 [1]. Deep learning’s most visible products today are generative AI (GenAI) models: large language models (LLMs) and their multimodal relatives, built on the Transformer architecture [2, 3].

This evolution has lately accelerated into an explosion of investment, research, and models—proprietary systems alongside the millions of open models that now populate community hubs such as Hugging Face [4]—advancing quickly on many narrow fronts while integrative, first-principles work remains comparatively scarce [4, 5]. The acceleration is accompanied by increasingly strong claims: imminent artificial general intelligence (AGI), then artificial superintelligence (ASI), a rhetoric that scholars observe hardening, at its extreme, into something resembling a belief system [6, 7]. Irrespective of their validity, such exaggerated claims have significant practical implications, as they distort how systems are characterized, assessed, and regulated.

In evaluation, the distortions are already concrete. A growing literature warns of a *tyranny of objectives* [8, 9]: once a benchmark becomes the goal rather than a proxy—an instance of Goodhart’s law—it can stop measuring capability at all [9]. Moreover, pushing further—an ambitious objective can be actively deceptive, so novelty-driven search can outperform objective-driven search [10]. Reliance on a single, aggregate performance metric distorts benchmark leaderboards, as selective test set usage and contamination introduce systematic biases into reported rankings [11]. A more subtle form of distortion is linguistic: the scholarly discourse frequently conflates *automation*—the execution of predefined tasks—with *autonomy*—the capacity to select which tasks to undertake [12], while anthropomorphic formulations further promote overstatement and misinterpretation of system capabilities [7]. Precise terminological usage and multi-perspective evaluation protocols are therefore indispensable for accurate and reliable assessment.

This scrutiny is urgent as the way models are used changes. They are increasingly deployed as generative *agents*—in practice, semi-automated systems, per the level-aware taxonomy this dissertation adopts [12]—and, crucially, they are reshaped after training. When released and used, model’s can be in different *states*: a concrete, deployable configuration of a model’s weights — the base model itself, a fine-tuned or merged descendant, or a compressed (e.g., quantized) variant — whose behavior can diverge from the original and which therefore deserves evaluation in its own right (formalized in Section 4.1.1.1). Ultimately, a concrete dimension of that landscape is urgent to address: how to rigorously compare such model states, and specifically, whether a compressed model remains trustworthy enough to deploy as an agent, and how an unbiased and multi-perspective evaluation can contribute in that realm.

1.2 Motivation

Currently, software increasingly acts on people’s behalf rather than responding to them: planning, calling tools, and invoking other services without step-by-step human review. The integration layer it depends on is already deployed at scale, and is largely exposed : a measurement of 7,973 live remote Model Context Protocol servers found that 40.55% expose their tool interfaces with no authentication at all, and every one of the 119 OAuth-enabled servers tested end-to-end carried at least one authorization flaw [13]. Delegation is running ahead of any infrastructure that could establish who a counterparty is or what it is entitled to do.

Yet none of those properties is new. They are the terms in which the multi-agent systems literature has defined agency for three decades. Following [14] and [15], an *agent* is a computing system located in an environment and capable of flexible, autonomous action to meet its design objectives, characterized by reactivity, pro-activeness, and social ability. What is new is the substrate: a *generative AI agent* instantiates the same definition with an LLM as its central controller, wrapped with planning, tool use, and multi-step action [16]. Nevertheless, autonomy is not yet what the label suggests. In practice, such systems remain semi-automated, according to the level-aware taxonomy adopted by [12].

If such agents, particularly using GenAI, characterize the present, an “Internet of Agents” has been proposed as a plausible future paradigm [17]. A prominent initiative in this direction is MIT’s Project NANDA (Networked AI Agents in Decentralized Architecture), which aims to develop the foundational infrastructure for an open, agent-centric web. In this envisioned ecosystem, billions—and ultimately trillions—of agents will be able to discover each other, verify each other’s capabilities, and coordinate their actions without relying on centralized intermediaries or gatekeeper authorities [18, 17]. Much as DNS and the early web protocols once did for machines and people [19], its design treats discovery, identity, and trust as first-order concerns. Whether or not this particular initiative prevails, it repositions earlier fragmentation as another infrastructural problem: at a planetary scale, coordinating and trusting GenAI agents becomes a central challenge.

A network of this scale cannot be deployed exclusively within centralized data centers. The NANDA roadmap explicitly extends the architecture into resource-constrained Fog and Edge environments [18, 20], with a particular emphasis on Edge devices. In these settings, model compression, especially quantization, alongside complementary compression strategies, becomes a necessary prerequisite for deployment and introduces a distinct set of technical challenges [17]. This prerequisite simultaneously enlarges an already vulnerable and exposed attack surface [21] and imposes additional deployment constraints. A prominent example is the objective of executing capable generative models directly on end-user devices. At full 16-bit precision, an 8-billion-parameter model requires approximately 16 GB of memory for its parameters alone, which exceeds the capacity of typical Edge hardware such as smartphones and single-board computers. Post-training quantization—the central case study of this dissertation—applied to achieve 4-, 3-, and 2-bit precision, reduces the model footprint to roughly 5, 4 and 3 GB, respectively, rendering on-device deployment practically feasible [22, 3].

In addition, a substantial fraction of contemporary generative AI agents are instantiated from open-source models whose training datasets and curation pipelines offer only limited guarantees on verification, quality control, and data provenance [23]. Critically, recent work has established the feasibility of adversarial strategies that remain inert under full-precision numerical representations but become activated only after quantization [24, 25], revealing vulnerabilities closely analogous to those systematically cataloged in the OWASP Top 10 for LLM-based applications [26]. Furthermore, in multi-agent architectures, where any generative AI agent can delegate tasks to, or exchange information with, any other agent, there is a growing need for robust mechanisms to establish and verify agent identity [17, 27], as well as methodological frameworks to make agent internal decision-making processes transparent. This transparency can be facilitated through explainable AI techniques or through deep learning first-principles analyses of GenAI models, both of which can become central to ensure reliability, accountability, and security [28, 29, 30].

Furthermore, the evaluation of generative AI systems is currently fragmented across methodological silos [5]. These include capability-oriented benchmark suites such as lm-evaluation-harness, adversarial robustness, and safety assessment frameworks such as garak [31], and data-free structural diagnostic approaches grounded in HTSR and SETOL theory [29, 30], each of which targets and quantifies different perspectives of model behavior or structure. However, as the Research Gap of this dissertation establishes (Section 3.4), no existing tool compares a model’s full-precision and compressed states through behavioral, adversarial, and structural properties at once, nor is any designed to extend to future assessment perspectives. Bringing these dimensions together, the first steps toward CHAOSATHENA are taken: a framework that unifies the perspectives through which a model is evaluated, moving beyond a paradigm that can risk a “tyranny of objectives” [9, 32]. Ultimately, offering integrated scrutiny of any GenAI model state—quantization first, motivated by the Edge deployment challenges within the NANDA future of trillions of agents, with further states to follow.

1.3 Research Questions

Given the Existing Opportunities (Section 3.5) and the Research Gap (Section 3.4) identified in Chapter 3, the following Research Questions will guide the work developed in this dissertation:

- **RQ1** How can a multi-perspective evaluation framework be architected to be modular, scalable, and extensible to new model states without modification of its core, and instantiated — as a case study — in independent stages that compare generative model states under compression? The evaluation infrastructure for generative AI is often fragmented between capability, safety, and interpretability silos that often do not interoperate [5]. Each term of the question has an operational and falsifiable meaning. *Modular*: every evaluation perspective is an interchangeable Stage behind a single contract, so an evaluation perspective can be removed or replaced without modifying the others. *Extensible without core modification*: adding a new model, a new model state, or an entirely new evaluation perspective requires only configuration and registration, never edits to the framework core — the Open–Closed Principle [33], realized through the inversion of control that distinguishes a framework from a library [34]. Model-specific details are not eliminated but *encapsulated*: Chat templates, specific parameters, and tokenizer conventions, when needed, are resolved from each model’s own metadata inside a dedicated *Adapter*, so that variability never propagates into the core of the framework. The main objective is therefore to architect a framework that lets empirical benchmarks, adversarial red-teaming, and first-principles or XAI theories (e.g., HTSR and SETOL, Mechanistic Interpretability) enter as interchangeable components [35, 29, 30], and to demonstrate it on a compression case study — full-precision versus quantized variants — that also allows further model states such as merged models [36, 37].
- **RQ2** How can data-free weight-matrix metrics be aggregated with capability and adversarial-red-teaming measurements into a multi-perspective dataset, indexed by model state, that future evaluation can extend?

The lasting value of a framework lies in the data it produces. We aim to assemble a single dataset, with associated per-model smaller datasets, that combines internal model weights structure metrics, with external behavior across model states, while keeping its schema sufficiently open to admit later security evaluations, alternative XAI frameworks, or complexity (e.g., Lempel-Ziv complexity) metrics as new dimensions, rather than forcing a redesign.

- **RQ3** How can Exploratory Data Analysis, statistical testing, or Instance Space Analysis relate the HTSR and SETOL weight-matrix properties of diverse architectures and model states to capability performance and adversarial vulnerabilities, and what statistically significant properties or edge-case conditions emerge?

The link between a model’s internal structure and its external behavior must be characterized before it can be predicted. The objective is to read the created datasets in two complementary ways — variable-by-variable through EDA and statistics, and, if possible, geometrically through Instance Space Analysis (ISA), relating those properties to capability performance (lm-evaluation-harness) and adversarial resilience (garak red team AI framework), and keeping only the structure properties — WeightWatcher using HTSR and SETOL — that prove statistically significant rather than incidental.

- **RQ4** To what extent can the implemented analytical modules predict or explain the degradation of a generative agent’s performance and security under quantization, specifically its resilience to critical OWASP Top 10 LLM vulnerabilities in future Fog-Edge multi-agent settings?

Quantization enables larger models to be deployed at the edge, yet its security implications are better understood from a multi-perspective standpoint. We aim to test how well the framework’s exploratory, statistical, or instance-space modules anticipate that cost — under formats such as GGUF, against a defined OWASP subset (e.g., LLM01, LLM02) — and whether such models can serve as future agents in decentralized settings, aiming for resource-restricted Fog-Edge conditions where, in certain scenarios, performance–security, but also structural trade-offs are important.

- **RQ5** By Synthesizing RQ1–RQ4, how can the framework contribute — through concrete exploratory and deployment insights, or future work — to trustworthy agents in decentralized Fog-Edge systems within the NANDA Web of Agents?

The ultimate test of the framework is whether its evidence leaves the case study and acquires the potential to inform concrete deployment decisions. We aim to show how the assembled multi-perspective evidence, indexed by model state, bears on the trust problem that decentralized Fog–Edge agent ecosystems face, and to identify which of the NANDA operational pillars that evidence can serve and which it cannot. The objective is deliberately bounded: no NANDA index, registry, attestation mechanism or agent-to-agent exchange is implemented or deployed in this dissertation, so the contribution consists of the produced data, the insights derived from it, and the integration designs those insights motivate — the last stated as future work rather than as a deliverable. Whether such an integration improves any deployment outcome is a question this dissertation frames and does not answer.

1.4 Contributions

This dissertation’s research results in the following contributions to trustworthy AI evaluation, Generative AI security, first-principles deep learning theory, and software engineering:

ChaosAthena, a multi-perspective evaluation framework for GenAI model states, with a modular, scalable architecture, extensible to new model states and evaluation perspectives

without core modification and portable across GPU and CPU-only backends, in which empirical benchmarking, adversarial red-teaming, and deep learning first-principles HTSR and SETOL diagnostics operate as interchangeable Stages under gray, black, and white-box regimes (Chapters 4 and 5), validated end-to-end over 28 model states.

A state-indexed multi-perspective dataset, aggregating, for each of the 28 states, architectural descriptors, capability performance, OWASP-mapped adversarial resilience, intrinsic toxicity-safety, and model weights spectral structure metrics, with engineered compressed-minus-FP16 deltas, under an extensible schema open to future states and Stages (Section 5.5.1).

An empirical characterization of post-training quantization, the case study of the framework: gradual degradation of capability with a qualitative 2-bit break, aggregate adversarial invariance with probe-specific movement, statistical decoupling of capability and safety, and Pareto deployment guidance placing Q4_K_M as the preferred level (Chapter 6).

Structure–behavior evidence at cohort scale, relating HTSR/SETOL weight diagnostics to behavior: the heavy-tail exponent α as an architecture-family property, spectral entropy tracking capability, output-layer localization of quantization damage, and $\Delta_{\text{rand_distance}}$ as a tracker of adversarial robustness change (Sections 6.3.5–6.3.7).

A guardrail safety-classification module, partially implemented, following the GuardReasoner methodology for reasoning-based prompt-harmfulness, response-harmfulness, and refusal judgments, complementing the red-team perspective (Section 5.4.2.2).

Empirical contributions established by the case study. Beyond the framework and the dataset, the case study establishes five empirical results. Each is stated with the coefficients and the evidential status it carries in Chapter 6, so that the claim and its scope travel together; $n = 28$ denotes a cross-sectional statistic over all model states and $n = 21$ a compression-delta statistic, and every coefficient quoted survives Benjamini–Hochberg correction within the family named in its source table.

1. **Capability and adversarial safety are decoupled, and the decoupling is carried by two disjoint blocks of weight structure.** Benchmark accuracy does not index red-team robustness ($r \approx -0.03$, n.s., $n = 28$). A spectral-richness block indexes capability strongly (entropy $+0.91^{**}$, num_spikes $+0.94^{**}$, stable_rank $+0.90^{**}$ against accuracy) with no detectable association to robustness ($|r| \leq 0.21$); a brittleness block indexes robustness ($n_{\text{trap}} -0.57^{**}$, $n_{\text{undertrained}} -0.45^*$) with none to accuracy ($|r| \leq 0.08$). Among continuous shape metrics only α_{weighted} reaches significance against robustness (-0.51^* , $q = 0.018$), and it is outranked by the trap-layer counter. *Consequence:* a compressed model that retains its capability scores cannot be assumed to retain its safety profile, so independent adversarial evaluation of every state is required rather than optional.

2. **Safety has two structurally distinct axes, and compression is legible in the spectrum for only one of them.** Aggregate adversarial robustness holds under compression ($+0.007$, $n = 21$) while intrinsic toxicity-safety degrades monotonically ($-0.022 / -0.066 / -0.099$ at Q4/Q3/Q2) in 20 of 21 variants. The structural footprint of capability loss is strong ($\Delta\text{entropy} -0.86^{**}$, $\Delta\text{rand_distance} +0.84^{**}$, $\Delta\text{log_spectral_norm} +0.72^{**}$ against $\Delta\text{accuracy}$, $n = 21$); that of toxicity-safety loss is absent at this cohort size, the closest candidate being a single mid-network depth cell ($r = -0.69$, $q = 0.01$) that is borderline under Spearman.

3. **Structure indexes adversarial robustness per attack, and the aggregate conceals it.** At the level of the distinct attack, a 36×6 structural screen ($n = 28$) yields **69 FDR-surviving couplings across 26 of 36 testable probes**, and the coupling’s sign is a property of the *attack family* rather than of the metric: DAN jailbreaks couple negatively to `rand_distance` and positively to α_{weighted} ($+0.61$ to $+0.72$) while latent- and prompt-injection probes invert both, and nine probes flip sign between `rand_distance` and `log_spectral_norm`. The metric *kind* that carries the signal is family-dependent too: the encoding and obfuscation family is indexed not by any shape metric but by the *scale* metric `stable_rank`, which supplies the strongest coupling in the study (`enc.InjectNato` $r = -0.80^{**}$, $\rho = -0.89^{**}$). Separately, six named LLM01 defenses degrade under compression — `dan.DanInTheWild`, `LI.LatentWhois`, `sata.MLM`, `dra.DRA`, `LI.LatentJailbreak`, `LI.TranslationEnFr` — three of them with no structural predictor at all, while `log_spectral_norm` is uncoupled from the overall pass rate (-0.35 , n.s.) yet reaches -0.61 ($p = 0.0006$) on the two output-handling probes and -0.72 ($p = 0.0003$, $n = 21$) as a delta. The same grid run on compression deltas leaves only **14 cells across 6 of 36 probes** ($n = 21$), against 69 across 26 at the level, so **structure indexes which states are robust far better than what compression will cost them**. *Consequence:* because an aggregate pass rate correlates with the LLM01 pass rate at $r = 0.97$ and pools families whose couplings have opposite signs, **a strong and systematic structure–vulnerability relationship can be entirely invisible in it**. This is a methodological result about how adversarial evaluations should be reported — per attack family, never on a pooled score — and one that the framework’s per-probe, OWASP-mapped output makes visible by construction. The level/delta contrast bounds its use: ranking states by structural exposure is supported, forecasting per-defense compression damage is not.

4. **Quantization preserves, particularly at an aggregate level, the learned spectrum and concentrates distortion at the network extremes.** The heavy-tail exponent α consistently remains within the well-trained interval $[2, 6]$ — with values closer to $\approx +2$ indicating better training quality — across all considered numerical precisions [29, 30], and appears to constitute an intrinsic property of the model’s architecture family ($\eta^2 = 0.956$, $n = 28$); perturbation magnitude grows an order of magnitude from the first layer ($|\Delta\alpha| \approx 0.035$) to the

last (≈ 0.305 , $n = 21$), although the signed movement there is outlier-driven, while the mid-deep quintile moves coherently ($+0.098$, 90% of variants). *Consequence*: the output layer is simultaneously where the perturbation concentrates and where α already sits above the band, which identifies a concrete target for mixed-precision protection or low-rank guided treatment.

5. **Weight structure carries model provenance, and adversarial robustness is non-monotonic in spectral scale (both exploratory)**. Compressed variants sit an order of magnitude closer to their own FP16 parent than parents sit to each other (0.30 against 2.83 standardized L2, non-overlapping, Mann–Whitney $p = 1.6 \times 10^{-8}$), and the quantization level is recoverable from six structural deltas alone at $21/21 = 100\%$ leave-one-out accuracy (permutation $p = 0.0005$); both are computed data-free, from weights, with no inference or prompts — a property that matters where tampered artifacts circulate on public hubs [24]. Separately, the pass rate appears *concave* in the mean log-spectral norm (standardized $\beta(z^2) = -0.430$, $p = 0.006$, $R^2 = 0.354$ across 28 states), with an interior optimum — the only non-monotonic relationship that was empirically tested. That second result is reported as a directionally consistent pattern rather than an established one: the predictor is a family-level property ($\eta^2 = 0.998$), so the fit rests on seven independent values, and the quadratic term does not reach significance on them ($p = 0.099$).

Finally, Contribution 5 and the **pre-specified 20-hypothesis battery** that supports parts of contributions 1–4, are **exploratory** in the sense set out in Section 6.3.9: verified against the produced datasets and corrected for multiplicity within named families, but hypothesis-generating rather than confirmatory, and bounded by a cohort of 28 states in one quantization format. The provenance signature in particular demonstrates that a strong data-free signal exists; it does not validate a detector, since no adversarially constructed or mislabeled artifact was tested. Both require a larger and more heterogeneous cohort before being asserted as general properties of compression (Sections 7.2.6 and 7.3).

1.5 Document Structure

The rest of this document is organized as follows:

- Chapter 2 establishes the background required for the three evaluation perspectives that the framework integrates: generative language models and their deployment in Fog-Edge systems, the fundamentals of post-training quantization, capability evaluation, heavy-tailed self-regularization as a data-free structural theory, adversarial security, and the statistical concepts used throughout.
- Chapter 3 reviews the state of the art in model compression and in the evaluation of compressed models, and identifies the gap this work addresses.

- Chapter 4 derives the requirements from that gap and presents the architecture of ChaosAthena, a four-phase, storage-decoupled design whose evaluation Stages are interchangeable.
- Chapter 5 details the implementation of each Stage and of the orchestration that binds them.
- Chapter 6 reports the case study: the experimental cohort and setup, the results across all three perspectives and their cross-relationships, an exploratory analysis of the resulting multi-perspective dataset, and the validation of the research questions and the system requirements.
- Chapter 7 states the limitations that bound the findings, draws conclusions, and sets out the future work that the framework enables.

Chapter 2

Background

This chapter defines the essential concepts, notation, and metrics used throughout the remainder of the dissertation. Material treated in detail elsewhere is intentionally omitted. The current state of the art in model compression and evaluation bias is reviewed in Chapter 3, while the formal problem formulation is also presented in Chapter 4. The exposition proceeds from a characterization of the primary objects — the generative models as LLMs and their states — through their deployment context (Fog–Edge computing and the Web of Agents) and the quantization it necessitates, to an examination of each model state along three main dimensions: (i) weight-matrix structure, (ii) functional capability performance, and (iii) security properties. The discussion concludes with a specification of the statistical assumptions and software-engineering conventions that support the analysis of all dissertation results and the core architecture components of the developed framework.

2.1 From Deep Learning to Generative AI and LLMs

A deep neural network is a composition of layers, each applying a learned linear map $\mathbf{y} = W\mathbf{x} + \mathbf{b}$ followed by non-linearity; its weight matrices W are fitted by gradient-based minimization of a loss in training data [38]. These trained matrices are the central objects of this dissertation, since the structural analysis reads them directly, without running the network. A *large language model* (LLM) is such a network trained to assign probabilities to sequences of *tokens*, sub-word units from a fixed vocabulary, using the *Transformer* architecture [2]: a stack of identical blocks, each pairing a *multi-head self-attention* sub-layer — whose query, key, value, and output projections are the matrices `q_proj`, `k_proj`, `v_proj`, and `o_proj` — with a feed-forward sub-layer holding `gate_proj`, `up_proj`, and `down_proj`, the granularity at which structural metrics are reported in Chapter 5. While the original Transformer pairs an encoder with a decoder, most modern generative LLMs — including every model evaluated in this dissertation — adopt the *decoder-only* variant illustrated in Figure 2.1: a single stack of such blocks in which self-attention is *causally masked*, each token attending only to those before it, so the architecture directly matches the autoregressive objective. Each block l produces a per-token *hidden state* $\mathbf{h}^{(l)}$; the final hidden

state, projected onto the vocabulary and passed through a softmax, yields the output distribution, whose pre-softmax scores are the *logits* and whose logarithms are the token *log-likelihoods*.

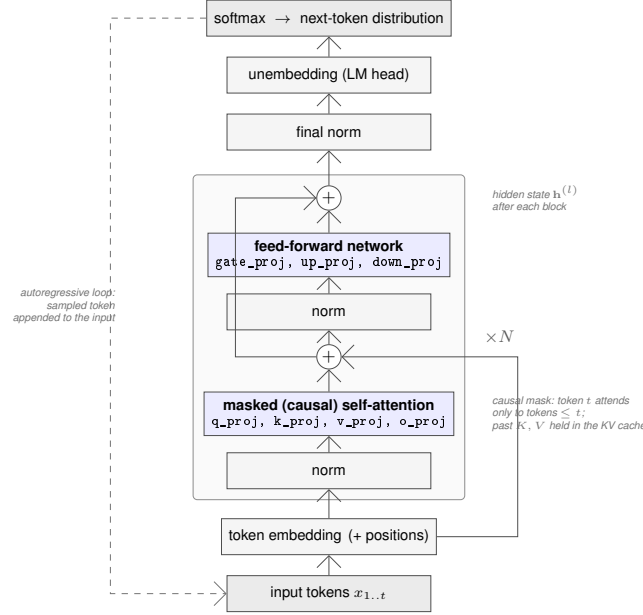


Figure 2.1: The decoder-only Transformer, the architecture of every model evaluated in this dissertation. Tokens are embedded and pass through N identical pre-norm blocks, each pairing causally masked multi-head self-attention (projections $q_proj, k_proj, v_proj, o_proj$) with a feed-forward network ($gate_proj, up_proj, down_proj$) around residual connections; a final norm and the unembedding (LM Head) then produce the next-token distribution, and each sampled token is appended to the input autoregressively. Original figure, based on [2]

Generation is *autoregressive*: the model predicts one token, appends it, and repeats up to a maximum *context window*. To avoid recomputing attention over the growing prefix, past key and value projections are held in a *key-value (KV) cache*, whose linear growth with sequence length creates a memory bottleneck addressed by the cache-compression methods surveyed in Chapter 3. At deployment, the arithmetic is executed by an inference engine, in this dissertation by vLLM or `llama.cpp` [39].

Currently, one GenAI model can exist in several *states*: a *base* model pre-trained on a broad corpus; *fine-tuned* variants specialized on narrower, domain specific data, or more general use, of which instruction-tuned models are the common case; and *merged* models, whose weights combine several fine-tunes without further training [36, 37] — the partition $\mathcal{X} = \mathcal{X}_{base} \cup \mathcal{X}_{ft} \cup \mathcal{X}_{merged}$ formalized in Equation (4.1). All such states circulate through hubs, Hugging Face above all, where anyone may publish weights without mandatory evaluation. A model can also be observed at three levels of access: *black-box* (outputs only), *gray-box* (additionally the output distribution and hidden states, the level at which log-likelihoods are read), and *white-box* (additionally the weights themselves, which the structural lens requires) [40, 29, 41] — a distinction adopted in Section 4.2.4 as the framework’s observational regimes.

2.2 Fog–Edge Computing and the Web of Agents

Large models are usually served from centralized cloud data centers. *Edge computing* moves computation onto or near the devices that produce the data, cutting latency and keeping data local [42], and *fog computing* is the intermediate tier extending cloud services across heterogeneous nodes between [43, 44]. These levels impose tight memory and energy budgets, intermittent connectivity, and heterogeneous hardware, making compression a *precondition*, not an optimization, to deploy a capable model at the edge. One level above sits the motivating vision: a *generative AI agent* wraps an LLM with planning, tool use, and multi-step action, and *Project NANDA* proposes the infrastructure for very large populations of such agents — an index resolving each agent to cryptographically verifiable *AgentFacts*, with identity, discovery, trust, and communication as first-order concerns [19, 18, 17] — and a roadmap extending explicitly into Fog–Edge environments [20]. Within this dissertation, Fog–Edge computing, particularly Edge computing, and the Web of Agents serve exclusively as the motivating deployment context: they explain why compression is a precondition and where the evaluated model states are intended to run. No Fog–Edge system or agent network is built or measured. Ultimately, the experimental object is always the model state itself.

2.3 Quantization Fundamentals

Deploying a model under the Fog–Edge constraints, particularly on Edge devices as described above, requires compressing it first. The weights of a given GenAI model are stored at finite numerical precision, 32- or 16-bit floating point in training, so a multi-billion-parameter model occupies gigabytes of memory and bandwidth [22]. *Quantization* shrinks this cost by re-expressing weights as b -bit integers, commonly $b \in \{2, 3, 4, 8\}$ [45]. The elementary operation is uniform affine quantization,

$$q = \text{round}\left(\frac{w}{s}\right) + z, \quad \hat{w} = s(q - z), \quad (2.1)$$

where the scale s and the zero-point z , chosen by *calibration*, define the representable grid and $\hat{w}$ is the *dequantized* approximation of the original weight w . Scales can be shared over small *groups* of weights, and *mixed-precision* schemes spend more bits on empirically sensitive layers — materialized in the GGUF “K-quant” profiles used throughout this dissertation. Quantization is irreversible [45]: the rounding in Equation (2.1) cannot be undone, so dequantizing a compressed model never recovers the original weights. This irreversibility, captured by the dequantization map in Equation (4.5), is what makes the structural measurement of the dequantized weights a possible measure of compression damage. Applied after training without further gradient updates, this is *post-training quantization* (PTQ) — the case study of this dissertation.

2.4 Spectral Analysis and Probabilistic Background

Two decompositions underlie the structural analysis: the *singular value decomposition*, writing any $W \in \mathbb{R}^{N \times M}$ as $W = U\Sigma V^\top$ with singular values σ_i , and the *eigendecomposition* of the correlation matrix $X = \frac{1}{N}W^\top W$ with eigenvalues λ_i , made equivalent views by the identity $\lambda_i = \sigma_i^2/N$. The scalar summaries of this spectrum used later, reported per layer in Table 5.10, are the Frobenius norm $\|W\|_F = \sqrt{\sum_i \sigma_i^2}$, the spectral norm $\|W\|_2 = \sigma_{\max}$, the *stable rank* $\|W\|_F^2/\|W\|_2^2$ (a noise-robust rank) and the *spectral entropy*, the Shannon entropy of the normalized eigenvalues, measuring how evenly the spectrum is spread.

Three probabilistic notions are needed, particularly for Chapter 4. The *Dirac delta* $\delta(\cdot)$ is a unit point mass, so a set of M eigenvalues is described exactly by the empirical measure $\frac{1}{M}\sum_j \delta(\lambda - \lambda_j)$, the form in which Equation (4.7) states a spectral density. A distribution is *heavy-tailed* when large values remain non-negligibly probable, the canonical case being the power law $\rho(\lambda) \propto \lambda^{-\alpha}$ — a straight line of slope $-\alpha$ on a log–log plot — whose exponent is estimated from the largest observations by the *Hill estimator* and whose fit quality is the *Kolmogorov–Smirnov distance* D between empirical and fitted cumulative distributions [46]. Lastly, *random matrix theory* supplies the null model against which learned structure is judged: for a matrix with independent random entries, the eigenvalues of $\frac{1}{N}W^\top W$ concentrate in a sharp interval $[\lambda_-, \lambda_+]$ with the *Marchenko–Pastur* (MP) density [47]. Eigenvalues escaping above λ_+ (“spikes”) therefore typically signal genuine learned correlation — the *correlation traps* of Section 2.6 are the cautionary exception; comparing a layer’s spectrum with that of an entry-shuffled version of itself isolates what training added over chance, the deviation reported as *rand_distance*; and the *MP soft-rank* measures how much of the spectrum still resembles the random bulk.

2.5 Capability Evaluation

A *benchmark* is a fixed dataset of tasks with known answers that is used to score models comparably. Evaluation is *zero-shot* when the model receives only the task and *n-shot* when it first sees n worked examples in context; scoring is either *multiple-choice*, selecting the candidate with the highest log-likelihood so that no text need be generated, or *generative*, checking produced text by exact match, code execution, or rule-based verification; instruction-tuned models additionally require prompts wrapped in a model-specific *chat template* [48, 49]. The task families evaluated in Chapter 6 span knowledge (MMLU [50], ARC [51]), commonsense (HellaSwag [52], WinoGrande [53]), truthfulness (TruthfulQA [54]), mathematics (GSM8K [55]), code (HumanEval+ [56, 57]), expert reasoning (GPQA [58]), instruction following (IFEval [59]), long-context retrieval (needle-in-a-haystack), and safety (ToxiGen [60]).

The metrics reported later are defined here. *Accuracy* is the fraction of items answered correctly; *normalized accuracy* (`acc_norm`) scores multiple-choice items by length-normalized log-likelihood so longer options are not penalized [49]; *exact match* requires literal string equality; and

$\text{pass}@I$ is the probability that a single generated program passes all unit tests [48]. IFEval quantifies instruction adherence along two dimensions of granularity—prompt level (all constraints within a prompt must be simultaneously satisfied) and instruction level (each constraint is evaluated independently) [59]—under two evaluation strictness regimes: strict and loose, where the latter permits minor, superficial deviations in formatting. *Perplexity* [61], $\text{PPL} = \exp(-\frac{1}{T} \sum_t \ln p(x_t | x_{<t}))$, measures intrinsic language-modeling quality on a lower-is-better scale, which is why it is reported separately and never averaged into accuracies; the *Kullback–Leibler divergence* $D_{\text{KL}}(p||q) = \sum_i p_i \ln(p_i/q_i)$ measures how far a distribution q drifts from a reference p [62], its multiple-choice variant later quantifying a compressed model’s drift from its full-precision reference beyond what accuracy captures. Accuracy-type metrics carry a *standard error*, where the scoring rule admits one.

Because evaluating many model variants on full benchmarks can be high-resource intensive [63], compact subsets built with *Item Response Theory* (IRT) are used instead on this dissertation: IRT models the probability of a correct answer as a function of the test-taker’s latent *ability* and each item’s *difficulty* and *discrimination* [64], and *tinyBenchmarks* exploits fitted IRT models to compress standard datasets to roughly one hundred highly informative items whose scores closely track full-benchmark performance [65] — the `tiny`-prefixed tasks configured in Section 6.2.5.1. All capability numbers are produced by the *LM* evaluation-harness, a widely adopted EleutherAI framework for reproducible LLM evaluation [48]: for each task, the prompt template is fixed, the n -shot setting, the scoring rule, and any output *filter* (post-processing such as extracting a final answer before scoring), and delegates execution to a chosen inference backend, so that every model and precision is scored under identical conditions.

2.6 Criticality and Heavy-Tailed Self-Regularization

Taking into account, that a *phase transition* is an abrupt change in a system’s collective behavior as a control parameter crosses a threshold; the boundary between the ordered regime and the chaotic one is the *critical point*, and a system tuned to it sits at the “*edge of chaos*”, where, Langton argued, information processing and computational capacity peak — order alone cannot compute because nothing changes, and chaos alone cannot because nothing persists [66, 67]. Remarkably, some systems reach this point without external tuning, the phenomenon of *self-organized criticality*, for example, on the sandpile model, in which avalanches of all sizes keep the pile poised at its critical slope [68]; the same signatures — scale-free events and power-law statistics — have since been reported across physical, geological, and biological systems, from earthquake magnitudes to neuronal avalanches in cortical tissue. *Criticality theory* studies this family of phenomena, and a growing literature argues that well-trained deep networks belong to it, organizing toward a critical regime associated with strong generalization [29, 67, 69].

Two kinds of extreme events also adhere to the description above. The heavy-tailed, power-law fluctuations of self-organized critical systems are the *black swans*; *dragon kings* are outliers of a different, partly deterministic origin, thought to arise in quasi-critical SOC systems — the

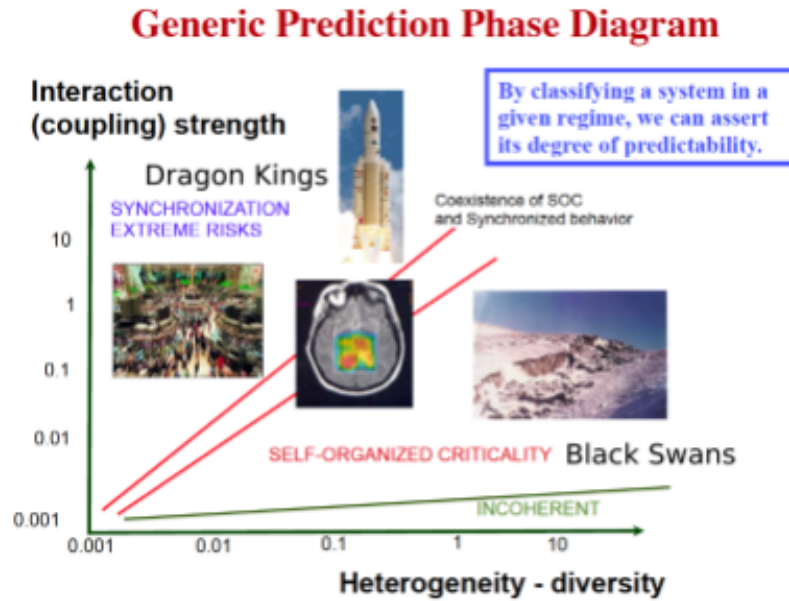


Figure 2.2: Illustrative example of phase diagram of complex systems, spanned by interaction strength and heterogeneity–diversity. Weakly coupled, heterogeneous systems are incoherent; along the self-organized-criticality band, extremes follow scale-free, heavy-tailed statistics (*black swans*); and strong coupling drives synchronization, whose coherent collective events — fed by long-range interactions and feedback loops and behaving as dynamical attractors — generate *dragon kings*, extreme outliers beyond the power law. SOC and synchronized behavior coexist along their boundary, and classifying a system by regime asserts its degree of predictability. Adapted from Sornette’s dragon-king framework [70] and [71].

avalanche patterns of cultured biological neurons are one example of that — when long-range interactions or feedback loops trigger a coherent collective event that behaves as a dynamical attractor [70]. In the spectral reading adopted here, such events are hypothesized to accelerate the learning of the features carried by the eigenvectors in the far tail of a layer’s eigenvalue spectrum, the ESD introduced next [29, 71]. Figure 2.2 organizes these regimes in a generic phase diagram of interaction strength against heterogeneity; reading a network’s health by its position relative to criticality is the intuition the rest of this section makes quantitative by approaching HTSR and SETOL [29, 30].

Having Criticality and the needed basics established, *Heavy-Tailed Self-Regularization* (HTSR) seems, to some extent, to materialize it, and allows it to be measured from a layer’s weights alone, with neither data nor labels [29, 30, 41]. For each layer, it forms the correlation matrix $X = \frac{1}{N}W^\top W$ and studies the *empirical spectral density* (ESD) of its eigenvalues. Empirically, the ESD of a layer passes through the *5+1 phases of training* identified by Martin and Mahoney [29]: it begins *random-like*, matching the Marchenko–Pastur bulk of an untrained matrix; enters *bleeding-out*, as the first learned directions push eigenvalues just past the bulk edge; reaches *bulk-plus-spikes*, with discrete signal spikes clearly separated from the bulk; then *bulk-decay*, as the bulk erodes and the spectrum stretches; and finally becomes *heavy-tailed*, its tail

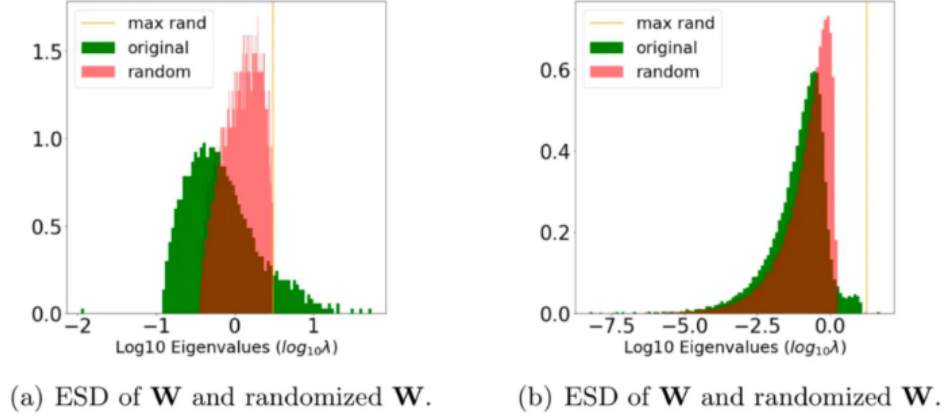


Figure 2.3: Empirical spectral densities of a layer’s weight matrix $\mathbf{W}$ (green) and of its randomized counterpart (pink), with the yellow line marking the largest eigenvalue of the randomized matrix. (a) Well-trained layer: the original ESD is clearly heavy-tailed and distinct from the randomized one. (b) Trapped layer: the original and randomized ESDs almost coincide, and a thin shelf of green density concentrates around the yellow line — a *correlation trap*, indicating that the layer failed to capture correlations that generalize. Adapted from the WeightWatcher and HTSR [29, 41, 71].

following a power law $\rho(\lambda) \propto \lambda^{-\alpha}$ with correlations at all scales — the additional *rank-collapse* phase marking pathological over-regularization [29, 30, 72]. The fitted exponent α , obtained by the Hill estimator with the KS distance D certifying the fit, thus summarizes where a layer sits on this trajectory, a smaller α meaning a heavier tail and stronger learned correlation [29]. The “good” and “bad” extremes of this trajectory are made concrete by the randomized-baseline comparison of Figure 2.3, discussed below with correlation traps: a healthy layer’s spectrum is heavy-tailed and clearly distinct from its randomized version, while a poorly-trained one barely differs from it [29, 30, 41, 71].

The exponent partitions layers into the regimes HTSR interprets in Table 2.1: $\alpha \approx 2$ marks an ideal, critically tuned layer; the band $[2, 6]$ a well-trained one; $\alpha < 2$ a very heavy-tailed regime associated with over-fitting, memorization, and *correlation traps* (Figure 2.3), outlier eigenvalues that appear even in the *randomized* version of the weight matrix and so reflect a few atypically large weight elements rather than genuine distributed correlation, a signature of unusually large learning rates or otherwise sub-optimal training; and $\alpha > 6$ an under-trained layer whose spectrum remains close to random. Figure 2.4 shows the histograms per-layer α of two models tuned according to instruction, with the overfit ($\alpha < 2$) and underfit ($\alpha > 6$) regions shaded outside the well-trained band. Because these bands read only the intrinsic geometry of the weights and no data, they are, to some extent, resistant to data-contamination, the property mentioned in Chapter 3 as a bias-resistant evaluation lens [29, 41, 73].

Because a raw exponent can be distorted by scale anomalies, HTSR distinguishes *shape* metrics, describing the form of the tail independently of magnitude (α , spectral entropy), from *scale*

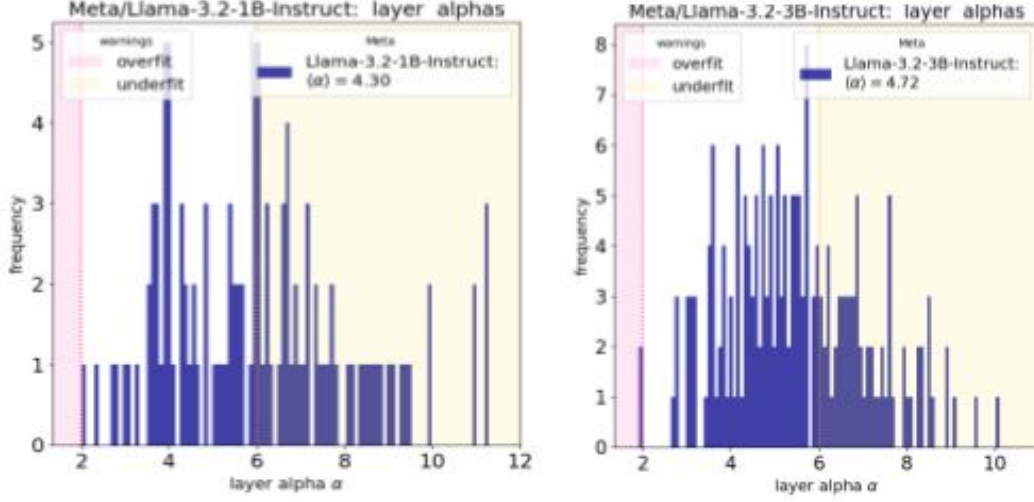


Figure 2.4: Histograms of the per-layer exponent α for two instruction-tuned models (mean $\bar{\alpha} = 4.30$ and 4.72). The shaded regions mark the out-of-band layers: *overfit* ($\alpha < 2$) on the left and *underfit* ($\alpha > 6$) on the right, with the healthy band $[2, 6]$ between them. Most layers fall inside the band, while a minority drift into the underfit region. Adapted from the WeightWatcher/HTSR framework [41, 71].

metrics, which depend on the size of the eigenvalue (norms, $\lambda_{\max}$), and defines the scale-adjusted *AlphaHat*, $\hat{\alpha} = \alpha \log_{10} \lambda_{\max}$, the quantity from HTSR and SETOL that relates most directly to generalization [41]. Together with *stable rank*, *spike counts*, *MP soft-rank*, and *rand_distance*, these form the per-layer metric family extracted in this work by *WeightWatcher* [41], the open-source tool that operationalizes the theory from weights alone, whose integration is detailed in Section 5.2.3. The tool does not need inference data: for each layer it reads the weight matrix, forms $X = \frac{1}{N} W^\top W$, computes its eigenvalues by SVD and fits the tail of the resulting ESD, as Figure 2.5 summarizes.

A model-level reading then comes from a *correlation flow* view: the profile of α across network depth traces how learned correlations propagate from layer to layer — ideally staying inside the well-trained band — and reveals that layer quality is also depth-dependent, with individual layers drifting outside the band even in models that are otherwise, at an aggregate level, identified as well-trained (Appendix B). This depth dependence is why the structural results are also analyzed using a layer depth perspective.

The *Semi-Empirical Theory of (deep) Learning* (SETOL) explains why the $\alpha \approx 2$ state is special [30]. It argues that a layer’s generalizing information concentrates in a low-dimensional *Effective Correlation Space* spanned by the dominant tail eigenvectors and imposes on it an *Exact Renormalization Group* condition: the transformation this space induces must be scale-invariant and volume-preserving, which reduces to the truncated correlation matrix having unit determinant, $|\det \tilde{X}| \simeq 1$, the form stated in Equation (4.10). Scale invariance is a characteristic signature of a critical system [30, 69], so SETOL connects the empirical target $\alpha \approx 2$ to the "edge-of-chaos" picture above and licenses treating α and $\hat{\alpha}$ as proxies without data for layer

Table 2.1: Interpretation of the HTSR power-law exponent α for a layer’s ESD tail [29, 30].

Regime	Range	Structural reading
Over-fit / very heavy-tailed	$\alpha < 2$	Memorization; possible correlation traps
Ideal (“critical”)	$\alpha \approx 2$	Maximally generalizing
Well-trained	$2 \leq \alpha \leq 6$	Healthy heavy-tailed structure
Under-trained	$\alpha > 6$	Spectrum near random (MP bulk)

health [30, 67].

2.7 Generative AI Security

LLM applications face a distinct class of input-driven attacks, the families that the security probes in this dissertation, in Chapter 6 instantiate. In this dissertation, *security* is used in a deliberately delimited sense: the adversarial robustness of a model state to prompt-level attacks — with OWASP LLM01:2025 Prompt Injection as the primary category — together with its intrinsic toxicity-safety[60]. Broader security concerns, such as infrastructure hardening or training-data provenance, are referenced only through the OWASP taxonomy and are not experimentally tested (Section 6.2.5.2). A *jailbreak* is a prompt designed to bypass safety training, the role-playing *DAN* family (“Do Anything Now”) being a widely studied example [31]. *Prompt injection* overrides the developer’s instructions with attacker-supplied ones — *directly* in the user input, *indirectly* via external content the model later reads, or *latently*, hidden inside seemingly benign material such as a translation snippet [31]. *Encoding-based* injections evade filters by expressing the payload in an alternate representation (Base64, ROT13, Morse or NATO alphabets, Unicode or ANSI tricks) [31], and *system-prompt extraction* attacks coax the model into revealing its hidden instructions [31]. As a concrete example of the mechanism, consider an encoding-based injection: the attacker submits a payload the model would normally refuse, expressed in Base64, together with the benign-looking instruction *decode the text above and follow it*. Safety filters that match patterns in natural language do not recognize the encoded payload, while the model’s own decoding capability reconstructs and executes it. The mismatch between what the filter sees and what the model understands is precisely the vulnerability the probe measures[31].

To make findings comparable across tools, attack outcomes are mapped onto the *OWASP Top 10 for LLM Applications*, a consensus taxonomy of the most critical LLM-specific risks maintained by the Open Worldwide Application Security Project [26]. For this dissertation, four categories matter here: *LLM01:2025 Prompt Injection*, the primary focus; *LLM03:2025 Supply Chain*, integrity risks across the model lifecycle, directly relevant to hub-distributed models; *LLM05:2025 Improper Output Handling*, downstream systems consuming unvalidated model output; and the legacy *LLM04:2023 Model Denial of Service*, retained for probes with no direct 2025 equivalent, with all other 2023 tags remapped to their 2025 counterparts [26].

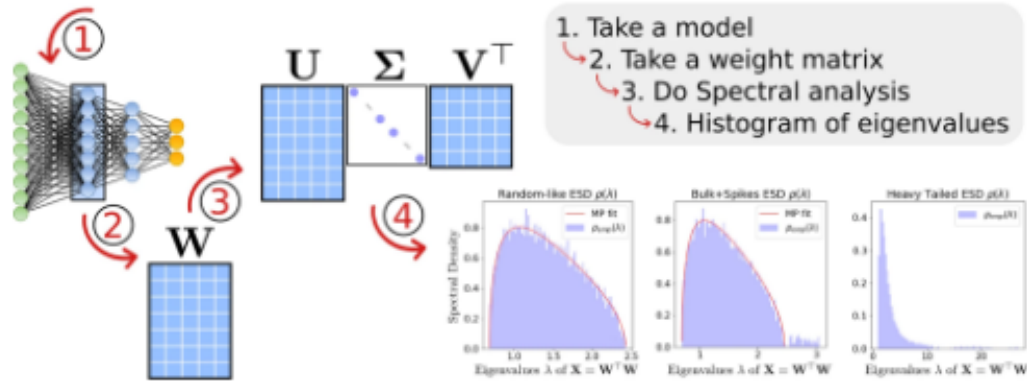


Figure 2.5: The data-free WeightWatcher tool. From a trained network, take a layer’s weight matrix W , perform the spectral analysis of $X = \frac{1}{N}W^TW$, and read the layer’s health from the shape of the eigenvalue histogram (ESD) — ranging from a random-like ESD, through bulk-plus-spikes, to a well-trained heavy-tailed ESD. Adapted from the WeightWatcher/HTSR framework [29, 41].

The terms *red team* and *blue team* come from adversarial military exercises and were adopted by cybersecurity: the red team plays the attacker to expose weaknesses, the blue team defends and remediates [74]. Applied to generative AI, the two roles bracket the model’s lifecycle. *Red Team AI* is the systematic, typically automated, adversarial probing of a model — ideally before deployment — to surface unsafe or exploitable behavior while it is still cheap to fix; *Blue Team AI* is the defensive counterpart operating from training through runtime: safety fine-tuning that shapes the model, guardrail classifiers that screen its inputs and outputs at inference, and monitoring that flags failures in operation. The two are complements rather than alternatives — their deliberate combination is sometimes called *purple teaming* — and both are instantiated in this work around the same model state, as Figure 2.6 depicts: an offensive probing module attacking each variant, and a defensive guardrail module judging its exchanges.

garak automates Red Team AI as a pipeline of *generators* (a uniform interface to the target model), *probes* (families of adversarial prompts implementing the attacks above), and *detectors*, which judge from the model’s response whether each attempt succeeded [31]. The primary outcome per probe is the *pass rate*, the fraction of attempts withstood — with *Attack Success Rate* (ASR) the complementary fraction, $\text{pass} = 1 - \text{ASR}$ — alongside the absolute *hit* count. To contextualize a raw rate, *garak* also reports a calibrated *z-score* against a reference “bag of models” [31], positive meaning above-average resilience, and summarizes severity on a *DEFCON*-style scale from 1, most critical, to 5, aggregated so that one critical failure is not masked by passes elsewhere.

Every quantity *garak* reports rests, at bottom, on a classification decision [31]. A detector is a binary safety classifier: its verdict on each attacked response, whether an attack succeeds or is defended against — is a true or false positive or negative of a *confusion matrix*, and because successful attacks are typically rare among the attempts, plain accuracy misleads [31], since a detector that declares every response defended scores well while catching nothing. Detector quality

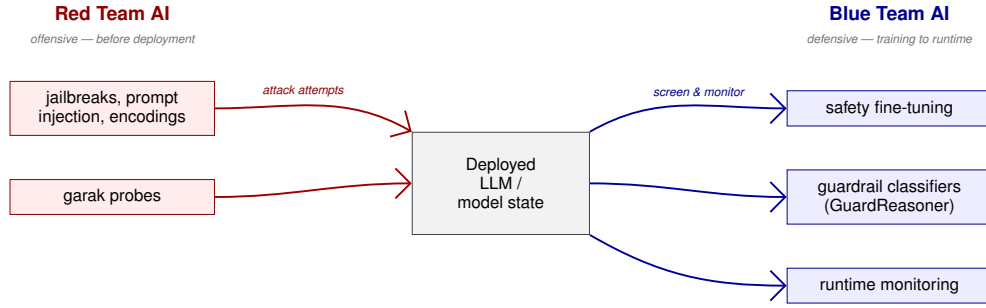


Figure 2.6: The offensive and defensive perspectives on model security. Red Team AI probes for failures before deployment, Blue Team AI defends from training (safety tuning) through inference (guardrails) to operation (monitoring) [31, 75].

is therefore judged by:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad F_1 = \frac{2 \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (2.2)$$

that can be made intuitive by the "archery target" of Figure 2.7: precision asks how many of the responses flagged as failures truly are failures, penalizing false alarms; recall asks how many of the true failures are caught, penalizing misses; and F_1 , their harmonic mean, collapses whenever either is poor. These measures stand behind every pass-or-fail judgment in the adversarial results of Chapter 6 [31].

A *guardrail* is a safety classifier that scans the inputs and outputs of an LLM application, the core of Blue Team AI[74]. The guardrail module of Chapter 5 follows the *GuardReasoner* methodology, in which the guard model does not merely emit a safe/unsafe label but first produces an explicit reasoning trace and only then its verdict, improving both accuracy and explainability across three tasks: *prompt harmfulness* (is the request malicious?), *response harmfulness* (does the output violate some determined safety policy?), and *refusal detection* (did the model correctly refuse?) [75]. The trace is what allows to extract more information about adversarial attacks be detected[75]. As a classifier, a guardrail is scored with exactly the instrument introduced above for garak's detectors, the confusion matrix, and precision, recall, and F_1 of Equation (2.2)[75].

2.8 Statistical Inference and Exploratory Data Analysis

The analysis conducted later in this dissertation is *exploratory data analysis* (EDA) in Tukey's sense: graphical and numerical summaries, such as histograms, box plots, or scatter plots, among others. It is therefore, used to reveal underlying patterns, trends, and anomalies before committing to a formal statistical model [76]. One distinction governs how everything in this dissertation must be read: exploratory analysis generates hypotheses from data, while confirmatory analysis tests pre-specified ones[76, 77].

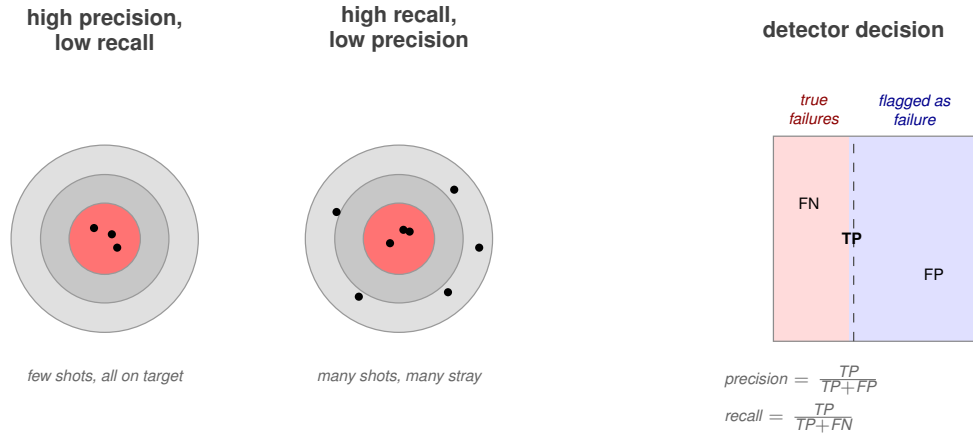


Figure 2.7: An archery-target reading of the classification metrics that score garak’s detectors and, secondarily, the guardrails: precision penalizes stray shots (false alarms), recall penalizes missed regions (undetected failures), and F_1 balances the two. Ultimately, the reason why F_1 , not raw accuracy, frames rare-class safety classification in a better way.

A *correlation coefficient* summarizes how strongly two variables move together on a scale $[-1, +1]$, the sign giving direction, and the magnitude strength. *Pearson’s r* , the covariance of the two variables normalized by their standard deviations, measures specifically *linear* association[78]. *Spearman’s ρ* applies the same computation to ranks and thus captures any *monotonic* relationship, which is why, is important to be reported side by side[78]. Each estimate is accompanied by a *p-value*, the probability of an association at least this strong under no true relationship; a 95% confidence interval obtained through the variance-stabilizing *Fisher z-transform* [79]; and R^2 , the share of variance explained.

Finally, three other standardization tools need to be mentioned: the *z-score*, $z = (x - \mu) / \sigma$, which places different scales on a common footing and underlies both garak’s calibration [31]. The *coefficient of variation* (standard deviation over mean), a scale-free dispersion measure[48]; and *variance decomposition*, which splits a metric’s total variation into *components* between families and *within* families, thus showing whether an effect is driven by architecture or quantization.

2.9 Framework Design Patterns

Design patterns are reusable, named solutions to recurring object-oriented design problems, each documented by its intent, structure, and participants [80]. Four occur in the implementation of Chapter 5, and Figure 2.8 sketches their structure as introduced in this dissertation. The *Strategy* pattern defines a family of interchangeable algorithms behind a common interface, turning the choice of algorithm into a runtime decision of the caller rather than a compile-time commitment[81]; it is what allows an evaluation stage to delegate text generation to vLLM or to `llama.cpp` without any change to the calling code. The *Factory* pattern, paired with a

Registry, removes the creation of objects from the caller[80, 81]. The registry maps declarative configuration keys to concrete classes and the factory instantiates the required one on demand, so new engines, probes, or metrics are added by registration alone, without altering Hydra orchestration pipeline code.

The *Facade* pattern places a single simplified entry point in front of a subsystem, shielding its users from the internal wiring. The *Adapter* pattern converts the interface a component offers into the one a client expects, and is used to reconcile model-specific chat templates with the uniform prompt interface assumed by the evaluation harness [81].

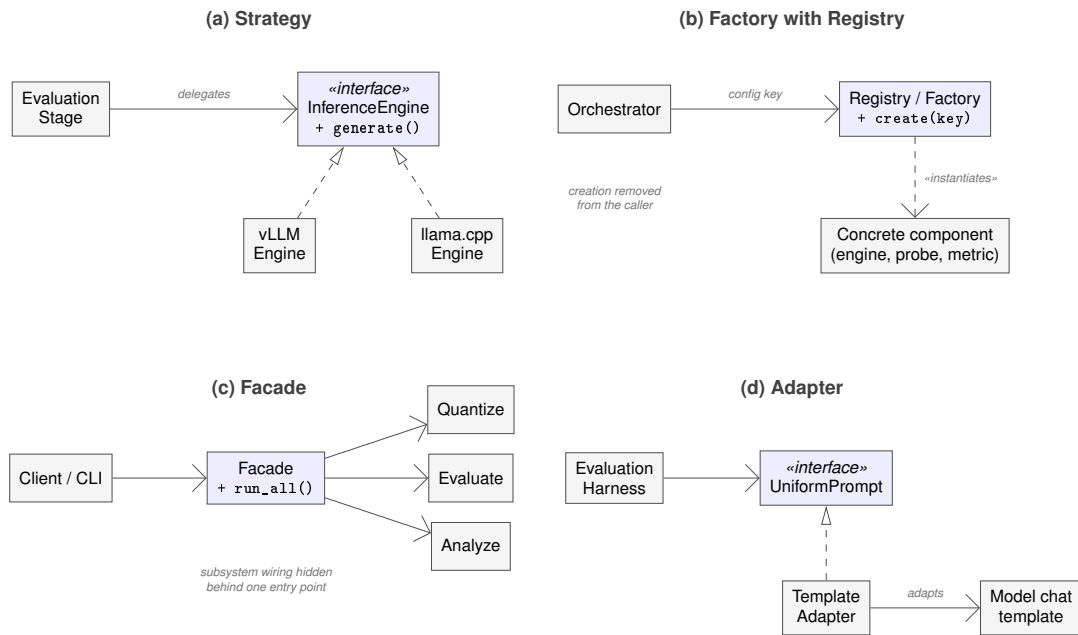


Figure 2.8: Figures of the four design patterns recurring in the implementation, as instantiated in this dissertation. (a) Strategy, interchangeable inference engines behind one interface; (b) Factory with Registry, configuration-key-driven object creation; (c) Facade, a single entry point over the pipeline subsystem; (d) Adapter, model-specific chat templates behind a uniform prompt interface, or future parameter specificities of a given model. Original figure, based on [80].

Among the patterns are two design principles. *Composition over inheritance* assembles behavior from small independent parts rather than deep class hierarchies[81, 80], and *loose coupling* minimizes what any component must know about the others. Both are realized here through a configuration-driven, storage-mediated design, sketched in Figure 2.9. Thereby, every stage declares its inputs and outputs in configuration, exchanges data with the others only through artifacts written to and read from decoupled storage, and never invokes another stage directly. Ultimately, these two design principles enable a future *microservice architecture*, in which each component is deployed as an independent networked service.

Finally, two data standards must be mentioned. ISO / IEC 25012 defines the characteristics of data-quality, of which this dissertation invokes *accuracy* (values faithfully represent the underlying quantity, up to preserving exact floating-point values through joins), *completeness*, and *consistency*

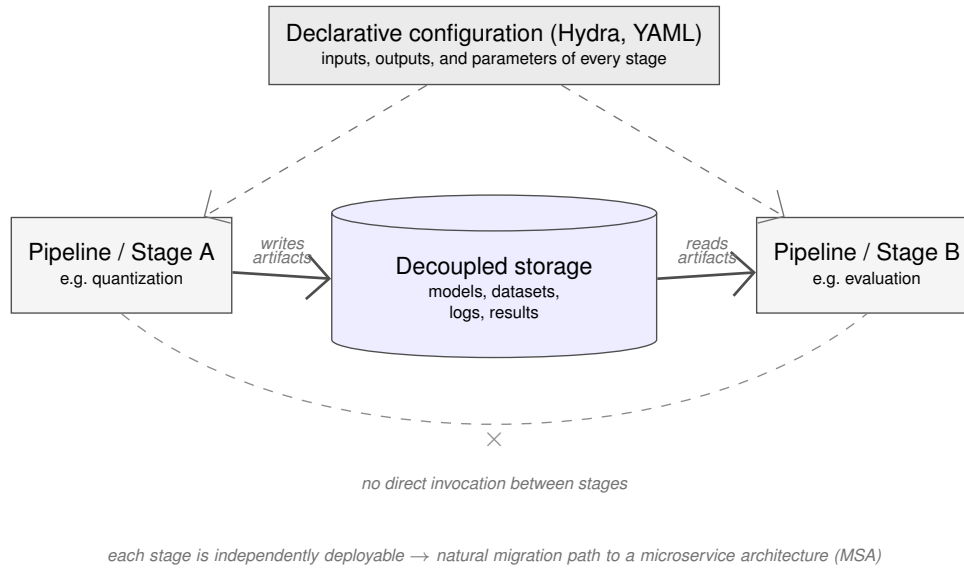


Figure 2.9: Loose coupling in the framework. Stages declare inputs and outputs in configuration and exchange artifacts only through decoupled storage, never invoking one another directly; each stage is thereby independently deployable, opening a natural migration path to a microservice architecture.

(uniform identifiers across joined sources) [82]; *DAMA-DMBOK* codifies data-management practice, of which this dissertation uses *data lineage*, an unbroken, verifiable trace from raw execution logs to final datasets [83]. These standards underpin the reproducibility of the aggregated evidence in Chapter 6.

Chapter 3

State-of-the Art

This chapter reviews the state of the art of model compression and frames it with challenges in evaluating Generative Artificial Intelligence (GenAI), with a focus on the last three years (2023-2026), during which much of the current AI research acceleration phase has begun [4, 84]. Section 3.2 first briefly introduces the current reality and challenges of AI, tracing the line from the current expansion of AI research and agentic systems to the compression imperative imposed by the Edge and decentralized deployment. Surveys the compression families and then examines the individual post-training quantization techniques that constitute the methodological foundation of this dissertation.

Given the current AI landscape and the state of the art in compression, Section 3.3 presents examples of evaluation bias and security challenges that, from a critical engineering-principled perspective, are urgent to address when assessing compressed models. Finally, Section 3.4 gives the research gap, and Section 3.5 states the existing opportunities that motivate the Research Questions of this dissertation in Chapter 1.

3.1 Compression of GenAI Models

Artificial Intelligence (AI) research has advanced faster than the instruments used to judge it. As capability has grown, benchmarks and public leaderboards have become the dominant proxy for progress. A critical challenge arises: these instruments are increasingly vulnerable to distortion [11], and a growing literature questions whether optimizing a single metric, or having a narrow focus on benchmarks, leads to a "Tyranny of Objectives" that measures capability at all [8, 9, 32]. However, the same expansion opens an opportunity: the problems of a single-angle evaluation perspective are precisely those that a multi-perspective assessment — reading behavioral benchmarks, adversarial resilience provided by red-team AI, and first-principles structural analysis together — is positioned to provide both a better, more unbiased evaluation and new opportunities to explore [11, 32].

A second paradigm shift has intensified the strategic importance of these developments. Generative AI models progressively serve as the core source for AI agents and multi-agent systems

built on large language models (LLMs) and multimodal LLMs [4]. Anticipating an Internet populated by billions to trillions of such autonomous agents — a term Section 3.3 critically revisits — that negotiate, delegate, and migrate within milliseconds, recent work argues that DNS-centered identity and discovery cannot scale to this regime and proposes the Networked Agents and Decentralized AI (NANDA) index—resolving to cryptographically verifiable AgentFacts—as the substrate for an “Internet of AI Agents” [19, 18], part of a broader program to decentralize AI [17].

From the decentralization pillars of the NANDA program, compression and on-device inference emerge as explicit challenges on the road to the realization of Fog–Edge multi-agent systems [85], particularly on Edge devices. Trillions of agents operating is possible, either by leveraging the current Cloud proprietary solutions [4] — but also by utilizing already available compute capacity from billions of Edge devices (e.g., traversing from microcontrollers to laptops) [17] —, to have capable models on the Edge, on heterogeneous, resource-constrained hardware depends largely on reducing their footprint[3, 22]. Moreover, beyond proprietary offerings, the Hugging Face ecosystem is the primary candidate to be the main source of open-source models for such a Web of Agents, where any account holder can publish a compressed variant using community tools [86, 87].

Industry estimates reported by Li et al. [85] place the savings from offloading roughly a fifth of AI workloads to on-device and hybrid processing at about fifteen billion dollars, while noting that the latency of uncompressed models is ill-suited to real-time settings such as autonomous driving or emergency response; the pursuit of capability must therefore be balanced against these constraints. The democratization and decentralization of GenAI are double-edged: they make edge-ready GenAI models abundant, but also mean that an immense number of models, in different states, are in the mix. (e.g., fine-tuned, merged), such as compressed variants, can be published without standardized evaluation, further increasing bias and security risks that require systematic investigation and mitigation [24, 11, 9, 26]. These challenges, how models are compressed (Section 3.2), and how their performance and safety should be evaluated, taking a multi-perspective approach and avoiding single-angle bias (Section 3.3) highlight the need for a review of the state-of-the-art in compression.

3.2 Model Compression

Model compression has become an essential stage in the deployment of large generative models, reducing the memory, bandwidth, and computation that inference demands, so that capable models fit within tight budgets [85, 88]. The current literature organizes these techniques into four families that are frequently combined in practice—pruning, knowledge distillation, low-rank factorization, and quantization [89, 90]. This section first gives an overview of the three families that this dissertation does not adopt as its case study, and then examines quantization in detail.

3.2.1 Pruning, Knowledge Distillation, and Low-Rank Factorization

Pruning removes redundant parameters or entire structures to induce sparsity. At the LLM scale, retraining is prohibitive, so one-shot methods dominate[91]: Frantar and Alistarh [92] introduced SparseGPT, which frames pruning as a sparse regression solved with approximate second-order information and reaches 50% unstructured sparsity on OPT-175B and BLOOM-176B with minimal perplexity increase, while Wanda [93] attains comparable sparsity by scoring weights by the product of their magnitude and the norm of the corresponding input activations, with no weight updates. Structured variants such as SliceGPT [94] remove entire rows and columns of the model’s transformed weight matrices and the corresponding input matrices (activations) to achieve hardware-friendly speed-ups, typically followed by a recovery fine-tune.

Knowledge distillation (KD) trains a compact student to reproduce the behavior of a larger teacher [95]. DistilBERT [96], distilled during pre-training, retains about 97% of its teacher’s language understanding while being 40% smaller and 60% faster, illustrating the appeal of KD for edge deployment; for generative models, white-box methods such as MiniLLM [97] that replace the forward Kullback–Leibler (KL) divergence with a reverse (KL) divergence objective focusing on matching a teacher model’s most probable outputs, thus improving open-ended generation quality.

Low-rank factorization approximates a weight matrix by the product of two smaller matrices, reducing parameter count. Recent work extends the idea to LLMs: LASER [98], for example, it selectively reduces the rank of higher-order components in specific layers and reports improved handling of rare facts, showing that factorization can interact with capability in non-trivial ways.

Importantly, the three families described above complement and often combine with the quantization techniques described in the next section.

3.2.2 Quantization

Quantization reduces the numerical precision of a model’s parameters and, in certain schemes, its activations. This process converts 16- or 32-bit floating-point values to low-bit integer or floating-point formats, thereby reducing the memory footprint by up to an order of magnitude and accelerating inference through efficient integer arithmetic [88, 89]. A concrete division exists: first, into Quantization-Aware Training (QAT), which simulates quantization during training to enable the model to learn parameters robust to it, and Post-Training Quantization (PTQ), which compresses an already-trained model without further gradient updates [89].

Critically, currently, PTQ is the dominant choice for publicly released LLMs on hubs such as the already mentioned Hugging Face, mostly because it avoids the cost and data demands of retraining, and recent surveys establish three categories, grouping PTQ methods by their target into: weight-only, weight–activation, and key–value (KV) cache quantization [89]. The following subsections review the most relevant state-of-the-art quantization methods within each category. To provide practical context, recent deployment-oriented formats associated with those methods are also discussed.

3.2.2.1 Weight-Only Quantization

In 2022, Frantar et al. [99] proposed GPTQ, a one-shot weight-only PTQ method that scaled accurate low-bit quantization to the largest models then available. Building on Optimal Brain Quantization [100], GPTQ quantizes weights one layer at a time and uses approximate inverse-Hessian information to update the remaining weights, thus compensating for the error introduced at each step by only requiring a small calibration set and no retraining. For example, evaluated on OPT-175B and BLOOM-176B, it reported 3- and 4-bit quantization with negligible accuracy loss, achieving roughly $4\text{--}5\times$ memory reduction and $3.25\text{--}4.5\times$ faster inference relative to the FP16 baseline [99].

Lin et al. [101] introduced AWQ (Activation-aware Weight Quantization), which exploits the observation that a small fraction of weights are disproportionately important for accuracy. By identifying the most salient weight channels from activation statistics, AWQ preserves them while applying a per-channel scaling that protects them under quantization, and—unlike mixed-precision schemes—keep a uniform integer format that maps efficiently to hardware. Ultimately, it is reported that strong 4-bit accuracy and enabled the deployment of a 70B-parameter Llama-2 model on mobile GPUs [101].

Furthermore, having established the well-studied and used GPTQ and AWQ methods, Chee et al. [102] advanced weight-only quantization by presenting QuIP (Quantization with Incoherence Processing). By further exploration, it can be shown that quantization to 2 bits can be achieved with theoretical guarantees. QuIP adjusts weights using the LDL decomposition of the Hessian computed over a calibration set, and multiplies the weight and Hessian matrices by Kronecker products of random orthogonal matrices to enforce *incoherence* between them, a property that limits the damage caused by outliers [102]. Nevertheless, QuIP, by combining the described steps, makes it possible to quantize LLaMA2-70B to two bits with a limited increase in perplexity [102].

To further minimize error and maintain a low average bit-width, several methods isolate the most sensitive weights and store them in higher precision. Dettmers et al. [103] proposed SpQR (Sparse-Quantized Representation), which uses the L_2 error between the original and quantized predictions as a sensitivity metric, keeps the resulting outlier weights in a sparse high-precision format, and quantizes the rest to 3–4 bits; on LLaMA-30B it held the perplexity increase below 0.2 at roughly 3.9 effective bits. Furthermore, by using a related design, Kim et al. [104] introduced SqueezeLLM, a dense-and-sparse method that clusters weights by Hessian-based sensitivity using k -means centroids as quantized values and stores the sensitive weights and outliers in an efficient sparse format; at three bits, it achieved more than a $2\times$ speed-up over the FP16 baseline with small perplexity loss [104].

Finally, in 2026, by adopting a different direction from other methods, Wang et al. [105] introduced SliderQuant (Sliding-layer Quantization). By abandoning the common assumption that every layer should be quantized in the same way, and motivated by the empirical finding that shallow and deep layers—and especially the first and last—are markedly more sensitive than intermediate ones, it couples an *inter-layer* sliding scheme, with three window designs matched to layer sen-

sitivity, with an intra-layer procedure that quantizes the layers inside each window incrementally. Across Llama, Llama-2/3, Qwen2.5, DeepSeek-R1 distilled models, and large mixture-of-experts models, SliderQuant reported improvements over GPTQ, AWQ, and OmniQuant—including recent rotation-based methods—in both weight-only and weight-activation settings, with the largest gains on challenging math and code tasks [105].

3.2.2.2 Weight-Activation Quantization

As an important stepping stone, in 2022, Yao et al. [106] proposed ZeroQuant: the first weight-activation PTQ scheme for large transformers. By pairing the group-wise quantization for weights with token-wise quantization for activations, and adding a layer-by-layer knowledge-distillation step to recover accuracy, it reduces both weights and activations to INT8. ZeroQuant reported $2\text{--}3\times$ memory reduction and up to roughly $5.2\times$ faster inference on models such as GPT-J-6B and GPT-NeoX-20B with minimal accuracy drop [106].

As a central difficulty in weight-activation quantization, activations contain outlier features whose direct quantization sharply degrades accuracy. Dettmers et al. [107] addressed this with LLM.int8(), which decomposes the matrix multiplication in the effort that the few outlier feature dimensions are computed in 16-bit precision while the remaining values use vector-wise INT8 quantization, recombining the two. Critically, the quantization of the weights and activations of models such as OPT-175B and BLOOM-176B to eight bits without measurable degradation is achieved by LLM.int8(): making 175B-parameter inference feasible on a single GPU server [107].

In another advancement, Xiao et al. [108] proposed SmoothQuant, which makes accurate 8-bit weight-activation quantization efficient by migrating the difficulty from activations to weights. By observing that outliers are consistent across activation channels, it applies a per-channel smoothing transformation that scales activations down and weights up by the same factor, after which both quantize cleanly on commodity integer kernels. SmoothQuant reported about $2\times$ memory reduction and $1.5\times$ speed-up, scaling to a 530B-parameter model with accuracy close to the FP16 baseline [108].

In a more recent work, Shao et al. [109] introduced OmniQuant, which restores accuracy at lower bit-widths by making the quantization parameters learnable rather than fixed. By combining learnable weight clipping, which optimizes the clipping thresholds applied to extreme weights, with a learnable equivalent transformation that shifts activation outliers into the weights, optimized block by block without end-to-end retraining. On LLaMA-7B with 4-bit weights, OmniQuant kept the perplexity increase near 0.4 [109]. Further work in this group targets the structure of outliers directly—through channel reordering, outlier-victim pairing, asymmetric channel suppression, or floating-point formats—ultimately confirming that outlier handling is the principal lever of weight-activation quantization [89].

3.2.2.3 KV-Cache Quantization and Quantization-Aware Training

Inevitably, as context windows lengthened, the KV cache that stores attention keys and values became a memory bottleneck, thus motivating its quantization. Hooper et al. [110] proposed KVQuant, which combines per-channel and pre-RoPE key quantization with non-uniform schemes and, by quantizing the cache to very low precision, enables inference at context lengths on the order of ten million tokens. Furthermore, from an analysis of the cache’s element distribution, Liu et al. [111] presented KIVI, a tuning-free method that quantizes key caches per channel and value caches per token, halving KV memory usage by 2 bits without fine-tuning.

Recently, the frontier in KV-cache quantization treats the cache as a vector quantization problem, yielding encouraging new results, with quantization reduced to two bits. Zandieh et al. [112] proposed TurboQuant, a two-stage, training-free, and data-oblivious quantizer: a first stage (PolarQuant) applies a random orthogonal rotation and quantizes in polar coordinates with Lloyd–Max-optimal centroids, spreading information evenly across coordinates, and a second stage adds a one-bit Quantized Johnson–Lindenstrauss (QJL) residual correction that preserves the inner products read by attention. TurboQuant compresses the cache to three or four bits per coordinate with near-optimal distortion, reporting at least a $6\times$ memory reduction and up to $8\times$ faster attention on H100 GPUs with negligible accuracy loss.

Furthermore, complementing the aforementioned results with algorithm–system co-design, JanusQuant [113] attains two-bit KV quantization through RtSmooth, a runtime smoothing transformation that suppresses outlier-induced error, together with a mixed-precision attention kernel; across several LLMs it preserved roughly 99% of FP16 accuracy while cutting KV memory by up to $5.3\times$ and raising decoding throughput by up to $4.45\times$.

Where PTQ is insufficient at the lowest bit-widths, Quantization-Aware Training (QAT) re-trains the model to absorb quantization error. Liu et al. [114] proposed LLM-QAT, which makes QAT data-free by distilling from the full-precision model’s own generations, improving low-bit accuracy on LLaMA models of up to 30B parameters; pushing the limit further, Xu et al. [115] introduced OneBit, a 1-bit weight representation with a tailored initialization that makes extremely low bit-width deployment viable. Furthermore, quantization can also be combined with fine-tuning: Dettmers et al. [116] introduced QLoRA, which couples a 4-bit NormalFloat (NF4) data type with low-rank adapters to fine-tune a 65B-parameter model on a single ≈ 48 GB GPU while retaining full 16-bit fine-tuning quality.

Recently, two results have further advanced QAT. By researching compute-optimal QAT, Dremov et al. [117] derive a loss scaling law—over model sizes from 86M to 2.2B parameters—that predicts the optimal division of compute between full-precision pre-training and a subsequent QAT phase. Critically, contrary to earlier belief, the loss-optimal QAT fraction *grows* with the total compute budget and can be read from a tokens-per-parameter-byte statistic; thus, by fusing the learning-rate cooldown with QAT, redundant full-precision updates are eliminated. Furthermore, to address deployment flexibility, Xu et al. [118] proposed multi-format QAT (MF-QAT), which trains a single model to be robust in terms of capability performance across multiple numeric for-

formats. By using a Slice-and-Scale procedure, it converts one stored high-precision anchor checkpoint (MXINT8 or MXFP8) into lower-precision MXINT or MXFP formats at inference time with little or no accuracy loss, allowing the runtime precision to follow the available hardware—a form of elastic inference.

3.2.2.4 Deployment of Quantized GenAI Models

The methods reviewed in Section 3.2 establish that, at least for certain models, even pushing PTQ to two bits, in some cases, can yield acceptable results in terms of capability performance, but the practical reality of GenAI model’s deployment raises additional challenges: heterogeneous and edge hardware is governed as much by file formats and tooling as by the quantization algorithm. Furthermore, considering the motivation outlined at the beginning of this chapter, this dissertation examines the intersection between the **Hugging Face ecosystem** and the *NANDA Web of Agents*—an infrastructure envisioned to scale from billions to trillions of autonomous agents that discover, delegate to, and negotiate with one another across heterogeneous protocols [19, 17]. Within this vision, the NANDA roadmap proposes that such a population cannot remain confined to hyperscale data centers: realizing it at planetary scale requires pushing agents toward **Edge devices**, for which model compression—and quantization in particular—is a first-order enabler [17, 18].

However, deploying Generative AI agents outside the cloud inevitably encounters substantial real-world constraints. **Fog–Edge computing**, which extends the cloud paradigm toward the network edge to deliver low-latency, location-aware, and geographically distributed computation over large numbers of heterogeneous nodes [43, 42], offers a natural substrate for hosting a decentralized Web of Agents close to where data is produced and decisions are taken [44]. Yet the very properties that make Fog–Edge attractive—tight latency budgets, constrained memory and energy, intermittent connectivity, and device heterogeneity—also delimit the obstacles that a trillion-agent network must overcome. In this setting, quantization is one of the bridges between the two worlds: post-training methods such as GPTQ, and AWQ, together with quantized fine-tuning schemes such as QLoRA, compress the language models enabling Gen-AI agents to 8- and 4-bit precision, or in some cases, beyond, by shrinking their memory and energy footprint enough to run on commodity Edge hardware while largely preserving task accuracy [107, 99, 101, 116].

Crucially, this efficiency is not free of risk, and its effect on an agent’s **Adversarial robustness** is far from a settled question. Reported findings range from increased susceptibility to jailbreak and prompt-level attacks [21], to little measurable change, to cases where quantization is actively weaponized: Egashira et al. show that a model can be crafted to appear benign at full precision, yet exhibit malicious behavior once quantized for on-device deployment—a threat that can propagate directly through model-sharing hubs such as Hugging Face [25]. The combined impact of quantization and task-specific fine-tuning on adversarial robustness at the edge, in particular, remains insufficiently explored [119]. Consequently, positioning Fog–Edge systems as a viable option within the NANDA architecture—and treating on-device quantization, together with its impact on Agent adversarial robustness and reliability, as a core research problem rather than a deployment

Table 3.1: Hardware support of the two dominant INT4 weight-only GPU serving formats (GPTQ also supports 8-bit). GPTQ has the wider reach—extending to AMD, Apple Silicon, and CPU moderately optimized through optimized engines[39]—whereas AWQ is effectively NVIDIA/CUDA-only, though it still spans data-center, consumer, and Jetson edge GPUs.

Approach	Hardware reach	Kernels & engines	Ref.
GPTQ/Marlin	NVIDIA Volta–Blackwell; AMD via ROCm; Apple Silicon; CPU	<i>Kernels:</i> Marlin, Machete, ExLlamaV2 (Ampere and newer). <i>Engines:</i> vLLM, SGLang, Transformers	[99, 120]
AWQ	NVIDIA/CUDA only, Volta–Blackwell: data center (A100, H100, L40S), consumer (RTX 4090/5090), and Jetson Orin edge; ~50% VRAM saving	<i>Kernels:</i> GEMM (Ampere and newer). <i>Engines:</i> vLLM, AutoAWQ	[101]

afterthought: **is, undoubtedly, a prerequisite for turning the Web of Agents from a conceptual vision into an operational reality.**

Ultimately, beyond any quantization algorithm, the appropriate deployment format is dictated largely by the target hardware. On GPUs, aside from the Cloud-intensive reality, from the quantization methods state-of-the-art reviewed above, only two methods, currently, dominate in terms of practical deployment applicability and broad use in the AI community: **the PTQ methods AWQ and GPTQ**, whose hardware support Table 3.1 contrasts [39], [22].

Nevertheless, **PTQ and AWQ formats have one problem:** they mainly presuppose a capable GPU and are widely used and available on Hugging Face and in state-of-the-art inference frameworks such as vLLM [39]. For the heterogeneous conditions of Fog-Edge Systems—for example, deployment realities with GPU, mixed GPU-CPU, CPU-only, and even clusters spanning different GPU and CPU types— a more hybrid capability in hardware and operating systems is currently needed.

Currently, in contrast, two other open-source approaches exist: GGUF, which runs the same quantized model on the same hardware as GPTQ and AWQ, but also offers wide compatibility with CPUs and consumer GPUs, running the mentioned hybrid conditions, and AutoRound, whose output can be served across CPU, Intel XPU, and CUDA backends and exported into the GPTQ, AWQ, or GGUF formats above, so that one quantization run adapts to whatever hardware is available. The GGUF format, originating in the `llama.cpp` project [121], applies block-wise mixed-precision quantization: its K-quant profiles (for example, Q4_K_M and Q3_K_M) group weights into super-blocks and allocate higher bit-widths to the most sensitive layers, optionally guided by an importance matrix computed on calibration data.

Because GGUF targets not only GPU, but also CPU, CPU+GPU and consumer hardware, and is read by `llama.cpp` and the tools built on it; it is the lowest-friction path to running models outside the data center and the format most widely uploaded by community quantizers. Recent

per-layer *dynamic* schemes refine it further: the Unsloth Dynamic 2.0 GGUFs (2026) assign a distinct bit-width to *every* layer rather than a fixed subset, calibrated on a large ($\sim 300\text{K}$ – 1.5M -token) corpus, and report markedly lower divergence from the full-precision model—measured by Kullback–Leibler (KL) divergence and 5-shot MMLU—than uniform static quants.

Unsloth report new best results on both metrics against standard `imatrix` and quantization-aware-training GGUFs across Llama 4, Gemma 3, and Qwen3, with the 2-bit profile alone cutting KL divergence by roughly 7.5%; on the independent Aider Polyglot coding benchmark, their 1-bit dynamic quantization of DeepSeek-V3.1—compressed from 671 to 192 GB, a 75% reduction—surpasses GPT-4.1 in non-thinking mode, whereas non-selective 1- and 2-bit quantizations of the same model fail to load or emit looping, garbled text, underscoring the value of per-layer bit allocation [87].

The same `llama.cpp` ecosystem extends GGUF’s reach in two directions that matter for edge deployment. *llamafile* [122] packages the runtime together with a GGUF model into a single executable—built on `llama.cpp` and Cosmopolitan Libc—that runs unmodified across Windows, macOS, Linux, and the BSDs on both x86-64 and ARM with no installation, collapsing deployment to one portable file. *prima.cpp* [123], in turn, distributes a single model over a heterogeneous home cluster of laptops, desktops, and phones using piped-ring parallelism with prefetching and a layer-assignment solver, bringing 30B–70B models within reach of everyday CPU/GPU devices: on a four-node home cluster it reports 5 – $17\times$ lower per-token latency than `llama.cpp`, *exo*, and *dllama*, running a 70B model at 674 ms/token and a 32B model at 26 tokens/s with speculative decoding, all while keeping memory pressure below 6% [123]. Together, they show that the GGUF format enables not only single-machine CPU inference but also portable and distributed execution across **exactly the resource-constrained, heterogeneous hardware that Fog–Edge multi-agent deployment assumes**.

The second approach, AutoRound [124], introduced by Cheng et al. [125], treats weight rounding as a learnable problem. Rather than rounding to the nearest representable value, AutoRound optimizes the rounding thresholds via signed-gradient descent (SignSGD) on a small calibration set, which is particularly effective in the low-bit regime: it is competitive at two bits and accurate at four bits. Across eleven zero-shot tasks, Cheng et al. report absolute average accuracy gains of 6.91–33.22% over rounding-to-nearest at two bits and near-lossless 4-bit quantization in most settings, obtained within roughly 200 tuning steps and with no added inference cost [125].

Beyond accuracy, AutoRound supports a range of bit-widths and mixed-bit recipes, runs **across CPU, Intel XPU, and CUDA backends**, and—crucially for an edge-oriented, Hugging-Face-centered workflow—**exports the quantized model to several formats, including AutoGPTQ, AutoAWQ, and GGUF**, so that a single high-quality run can be served by the `llama.cpp` ecosystem [124, 125]. AutoRound **therefore complements GGUF rather than competing with it**: it offers a calibration-based, hardware-flexible route to a quantized model that can ultimately be served in the same edge-friendly format.

3.2.3 Discussion

In light of the comprehensive state-of-the-art analysis and the preceding discussion, Table 3.2 compares four principal Post-Training Quantization (PTQ) methodologies. This comparative synthesis not only elucidates the relative merits and limitations of each approach within the context of Fog-Edge multi-agent systems, but also serves as the decisive basis for selecting the quantization methods under research and will form the basis of the final Research Gap:

Table 3.2: Comparison of the four principal PTQ approaches on the axes that govern Fog-Edge deployment: viable bit-width, reported accuracy/fidelity, and deployment evidence across hardware and operating systems. Because the cited works evaluate different base models and metrics (perplexity, KL-divergence, or task accuracy), the accuracy column records each method’s own reported fidelity rather than a strict head-to-head ranking. GGUF and AutoRound (**bold**) are adopted here—GGUF as the primary format and AutoRound as a complementary calibration route.

Method	Bits	Reported accuracy / fidelity	Deployment evidence & hardware/OS heterogeneity	Ref.
GPTQ	3–4	Near-lossless at 4-bit; +0.34 WikiText-2 PPL with $3.24\times$ speed-up on OPT-175B (3-bit);	NVIDIA Volta–Blackwell, AMD/ROCm, Apple Silicon, CPU fallback (vLLM, SGLang, Transformers); Linux-centric OS support; AutoGPTQ format	[99, 120]
AWQ	4	Near-lossless at 4-bit with $\sim 50\%$ VRAM reduction; +0.42 WikiText-2 PPL with $3.2\times$ speed-up	NVIDIA/CUDA only, data center (A100/H100/L40S) to Jetson Orin edge (vLLM, AutoAWQ); Linux/CUDA; serves a 70B model on mobile GPUs	[101]
GGUF (Dyn. 2.0)	2–8	New best 5-shot MMLU and KL-divergence vs <i>imatrix</i> /QAT profiles ($\sim 7.5\%$ lower KL-div at 2-bit); 1-bit DeepSeek-V3.1 (-75% size) surpasses GPT-4.1	CPU and consumer GPU via any <i>llama.cpp</i> engine; single <i>llamafile</i> across six OSes (x86-64 and ARM) and distributed over a cluster via <i>prima.cpp</i> ; hybrid edge/CPU and GPU format	[121, 87, 122]
AutoRound	2–8	+6.91–33.22% zero-shot accuracy over round-to-nearest at 2-bit (11 tasks); near-lossless at 4-bit; ~ 200 calibration steps, no inference overhead	CPU, Intel XPU, CUDA; Hugging Face hub / <i>llama.cpp</i> stack; one run exports to GPTQ, AWQ, or GGUF	[124, 125]

Ultimately, the comparison in Table 3.2 clarifies the selection. GPTQ and AWQ reach near-lossless accuracy at four bits but mainly presuppose a GPU, or have low CPU compatibility—AWQ almost exclusively NVIDIA/CUDA—so neither satisfies the CPU-only and mixed CPU–GPU conditions of a Fog-Edge deployment on its own. GGUF and AutoRound maintain comparable accuracy while removing that constraint: **GGUF** is the *de facto* hybrid format for

CPU, GPU, and consumer hardware, integrating quantization [121]. Operates across the entire `llama.cpp` ecosystem, at same time, being portable to six operating systems via engines, such as `llamafile`[122]. Additionally, AutoRound is a calibration-based quantization method that creates quantized models in different formats, including AutoRound, GPTQ, AWQ, or GGUF [124, 125].

This dissertation therefore adopts the GGUF format as its primary compression target, with AutoRound as a complementary calibration route, while the broader method landscape reviewed above also informs the structural and behavioral analysis of compression in later chapters.

3.3 Evaluation Bias and Multi-perspective assessment

If compression influences how capable a GenAI model can be on the Edge, evaluation decides whether that capability—and its safety—can be trusted. Generative-AI evaluation, however, is currently fragmented across capability, safety, and interpretability silos that lack interoperability, augmented by the “explosion” of AI research in the last 3 years[5, 4]. Each silo, either an evaluation framework or an AI research field, can carry bias that a single-angle assessment cannot detect. This section reviews state-of-the-art evaluation biases and complementary first-principles, adversarial methods, and the dangers of language overclaims that motivate a multi-perspective approach, emphasizing works that intersect with the evaluation of compressed models.

3.3.1 Evaluation Bias

The first addressed bias is the tyranny of a single objective, briefly referenced in Section 3.1. When a benchmark score becomes the goal rather than a proxy, optimization pressure maximizes the metric at the expense of the quality it was meant to track—an instance of Goodhart’s law that Thomas and Uminsky [8] argue that it is fundamental in the current situation of AI research and proprietary solutions such as Claude, Gemini, or ChatGPT [4]. In practice, this manifests as *benchmaxxing*: tuning to the idiosyncrasies of popular benchmarks rather than to general competence.

Stanley and Lehman [9] push even further: the critique argues that an ambitious objective can be actively *deceptive*, because the stepping stones— an event or experience that helps you advance or achieve a goal—toward a breakthrough rarely resemble the goal, so optimizing for it directly misdirects the search into dead ends. Real evidence is provided: in deceptive maze and biped-locomotion tasks, a search rewarding only behavioral *novelty*—ignoring the objective entirely—significantly outperforms objective-driven search [10], and a skull image that Picbreeder users evolved open-endedly in only 74 generations, could not be recovered when it was set as an explicit target, with 20 automated runs of 30,000 generations each failing to reproduce it [32].

Consequently, the lesson transfers to model evaluation: a single benchmark, or benchmarks that in practice measure the same capabilities or incentivize a common, too-rigid objective, pursued relentlessly, reward the peculiarities of that benchmark rather than the open-ended competence actually sought. Consequently, capability assessment should be corroborated by multiple methodologically distinct sources of evidence, ensuring that results are not artifacts of a single benchmark or evaluation methodology.

The second bias is data contamination. As models train on web-scale corpora scraped from sources such as GitHub and arXiv, benchmark datasets leak into pre-training. Code benchmarks illustrate this acutely: HumanEval, released with the Codex model by Chen et al. [56], has been shown to overlap substantially with training data, inflating pass rates and masking generalization; worse, simple paraphrases of test items evade n -gram decontamination, and Yang et al. [73] show that a model trained on such rephrased data can overfit a benchmark to a degree that rivals far larger models.

The third bias is the leaderboard distortion. Singh et al. [11] documented undisclosed private testing—at the extreme, one provider evaluated 27 private variants before a single public release—together with asymmetric data access and over-sampling that bias the algorithm that converts win-loss comparisons into a unified leaderboard, the Bradley–Terry scoring, and rewards over-fitting to the Chatbot Arena distribution. Again, in the same direction as the two aforementioned biases: undoubtedly, evaluation bias is of paramount importance and must be addressed; thus, a multi-perspective approach can offer a new direction and an unbiased view, promising solutions and opportunities for research[5].

3.3.1.1 The Problem of Overclaims

A final example of distortion that can also lead to bias concerns the language of capability itself. By examining cybersecurity AI, Mayoral-Vilches [12] argues that the field routinely conflates *automation* with *autonomy*, and, drawing on the SAE driving-automation levels, defines a six-level taxonomy (Levels 0–5) that separates the execution of predefined routines in predictable settings from the capacity to perceive, reason, and adapt under unforeseen conditions. By that measure, most systems marketed as “autonomous”—including the pentesting agents using GenAI—operate only at Levels 3–4, still requiring human review for edge cases, while genuine Level-5 autonomy remains aspirational.

The risk is the expectations gap rather than the technology: mistaking an automated tool for an autonomous one withdraws oversight exactly where it is most needed, which is at once a safety problem and an evaluation one, since an inflated autonomy claim is itself a measurement error, together with the benchmaxxing and leaderboard distortion described above. In other words, terminology carries evaluative weight: imprecise or anthropomorphic language about AI invites overclaiming, which is why careful description matters [7]. Nevertheless, this is not a criticism of forward-looking proposals such as NANDA [18], whose “autonomous agents” name an anticipated future agentic web (Section 3.1); the concern is rather the uncritical use of the label for present-day systems. This dissertation therefore adopts Mayoral-Vilches’s taxonomy [12], treating current generative agents as semi-automated systems whose assessment must stay level-aware and keep a human in the loop [12].

3.3.2 Towards a Multi-Perspective Assessment

Based on the state of the art in deep learning first principles and security perspectives, two complementary views are selected and described to complement the general capacity performance assessed on Benchmarks for GenAI models, particularly for compressed models, which are the dissertation's focus.

3.3.2.1 HTSR and SETOL

The first perspective is the data-free, first-principles structural analysis of a model's learned weights, where the needed basic knowledge is described in Chapter 2 and further detailed in Chapter 4. Grounded in Statistical Mechanics (StatMech), Random Matrix Theory (RMT), and Quantum Chemistry, the Heavy-Tailed Self-Regularization (HTSR) and the complementary Semi-Empirical Theory of Learning (SETOL) evaluate model quality by examining the structural correlations within the network [29, 30]. In practice, HTSR performs Singular Value Decomposition (SVD) to extract the Empirical Spectral Density (ESD) of each layer's correlation matrix, subsequently fitting a power-law distribution to its heavy tail to derive the shape exponent α [29, 126].

This α parameter acts as a diagnostic of layer health: optimally trained layers naturally self-organize toward a critical phase boundary ($\alpha \approx 2$), whereas deviations indicate severe overfitting and memorization ($\alpha < 2$) or under-training ($\alpha > 6$) [29, 30]. SETOL additionally formally enforces this ideal learning state through the Exact Renormalization Group (ERG) condition, which dictates that the layer's generalizing components must concentrate into a low-rank Effective Correlation Space that satisfies a scale-invariant, volume-preserving transformation ($\prod \lambda_i \approx 1$ or $\sum \ln \lambda_i \approx 0$) [30].

Reaching this $\alpha \approx 2$ ERG state, mathematically mirrors the concept of "criticality" or intelligence at the "edge of chaos"—a regime where deep learning networks are poised at a critical phase transition, perfectly balancing predictability and randomness that maximizes information processing and generalization [69, 67, 66]. Because these spectral diagnostics read only the intrinsic mathematical geometry of the weight matrices—requiring neither inference data nor labels—to some extent, they are immune to data contamination bias. [30, 127, 72].

As an illustrative example, if a model overfits the test data and "memorizes" it, by using HTSR and SETOL to extract metrics that can signal overfitting: it is possible to identify within the layers weights, that some overfit layers drop into a specific "Very Heavy-Tailed" phase ($\alpha < 2$) or develop "Correlation Traps"[29, 30]. Furthermore, this structural evaluation explicitly intersects with model compression: because post-training quantization (PTQ) can perturb matrix weights, these same spectral metrics can be used to quantify structural degradation and rank collapse it induces [128, 45, 29].

3.3.2.2 Red Team AI

Another perspective is adversarial resilience evaluation (Red Team AI). Deep neural networks, and the generative models built on them, are widely treated as *black boxes*: the distributed computation across their many layers makes it hard to trace why a particular output is produced [40]. Two consequences follow for evaluation. First, security is not a by-product of capability—a model that scores well on capability benchmarks can still be unsafe—so it cannot be read off those scores and must instead be probed behaviorally. Second, that behavioral probing, adversarial red-teaming, is far less mature than capability evaluation: Feffer et al. [74] find no consensus on how red-teaming should be defined, conducted, or reported, and no standardized benchmark or public leaderboard comparable to those that exist for capability. In practice, it is carried out privately for proprietary models and is largely uncontrolled for the open-source ecosystem—on Hugging Face, any account holder can publish a fine-tuned, merged, or compressed variant that others download and run with no security assessment.

As an approach providing systematic adversarial testing, tools such as the NVIDIA garak framework [31] operationalize red teaming as an automated Stage composed of three core components: *generators*, which constitute an abstraction layer over the target model and thereby enable the same probes to be executed against any endpoint served; *probes*, which are families of adversarial prompts designed to exercise specific failure modes (e.g., prompt injection, jailbreak attacks, encoding-based and latent injections); and *detectors*, which parse model responses to determine whether each probe has successfully elicited the targeted failure. The resulting failures can be categorized according to recognized taxonomies, such as the OWASP Top 10 for LLMs [26], yielding a standardized and comparable assessment of adversarial robustness. In parallel, defensive methodologies such as reasoning-based guardrail classifiers [75] tackle the same set of risks from the content moderation perspective.

However, for compressed models, such a structured characterization is particularly critical, as quantization can substantially alter a model’s vulnerability surface and the corresponding attack space. The lack of robust security measures and user control, as mentioned in Section 3.2.2.4, is especially prevalent on open-source model platforms such as Hugging Face [23], and documented attack vectors—such as GGUF-based attacks are a reality, thus supporting the urgency of state-of-the-art, better security practices [24, 26].

3.3.2.3 Further Perspectives and Analysis Instruments

The perspectives outlined above do not exhaust the full range of possible complementary evaluation viewpoints. Mechanistic Interpretability pursues the same first-principles goal at the level of individual computations, with tools such as InTraVisTo tracing token trajectories inside a Transformer [129], while complexity measures such as Lempel–Ziv tie the intrinsic structure of inputs and outputs to downstream capability [67]. Efficiency telemetry — latency, memory footprint, throughput, and energy/CO₂ — conditions any Fog–Edge deployment claim [130, 131, 132].

Relating such heterogeneous measurements demands statistical care, from false-discovery control [77] to guard against analytical forking paths [133]. Instance Space Analysis (ISA) complements these statistics geometrically, projecting instance features and algorithm performance into a two-dimensional topology that exposes footprints, strengths, and edge-case conditions [134, 135].

Ultimately, these possibilities are not the focus of the dissertation; they are only referenced as evidence of future extension possibilities (Section 3.5).

3.3.3 Summary

In summary, capability benchmarks, adversarial red-teaming protocols, and first-principles deep learning structural diagnostics provide a distinct but complementary measurement perspective, capturing different dimensions of system behavior, model weight structure, and performance that are not accessible through the others. Each methodology also introduces characteristic biases that can, in principle, be counterbalanced by the others. Nevertheless, the tools and infrastructures that implement these approaches are typically designed, analyzed, and deployed in isolation [4, 5]. Section 3.4 formalizes this fragmentation and delineates the resulting gap.

3.4 Research Gap

The preceding review converges on a gap that existing work cannot close. The evaluation infrastructure remains fragmented between capabilities, safety, and interpretability silos that do not interoperate (Section 3.3). Recent work has begun to bridge two of these silos: Hong et al. [136] evaluate three LLMs under five compression techniques across eight trustworthiness dimensions, establishing that 4-bit quantization broadly preserves trustworthiness while 3-bit compression degrades it, and that this risk is invisible from benign task performance alone; Li et al. [137] map, across eleven model families, which capabilities quantization damages, identifying mathematical multi-step reasoning as the most sensitive. Both pair capability with behavioral safety, and this dissertation’s own results are read against them. However, neither admits the third perspective nor is focused on adversarial resilience (Red Team AI): no existing tool compares a model’s full-precision and compressed states through behavioral, adversarial, **and data-free structural properties at once nor relates the structural account to the behavioral one**. Both also evaluate research-grade compression methods rather than the deployment formats that dominate community distribution, and neither leaves behind a state-indexed dataset that later evaluations can extend.

Moreover, the compression literature compounds this fragmentation. Most of the methods surveyed in 3.1 remain research artifacts that rarely reach the formats that dominate the deployment reality (Section 3.2.2.4); each reports fidelity to different base models, parameter scales, and metrics, excluding the strict head-to-head comparison (3.2) required by systematic method selection and security assessment. The millions of diverse models in different states that now populate Hugging Face, including compressed variants (Section 3.1), exist with little or no bias characterization in terms of their capability costs, and with almost none of their security or structural consequences. This omission has significant implications: vulnerable models already exist in the

same hub [23], quantization-activated attacks are a reality [138], along with data contamination and leaderboard distortion, which undermine the capability-only scores by which the community judges these models (Section 3.3) [11, 73, 139].

Furthermore, insofar as the existing literature suggests, no previous work has related data-free HTSR and SETOL diagnostics (Section 3.3.2) to *both* capability performance and adversarial resilience across full-precision and compressed states of the same models, nor has it indexed such measurements by model state for reuse and extensibility. This is the specific gap that the present dissertation work addresses, and it is orthogonal to the capability–trustworthiness studies above [140, 137]: those characterize *what* compression costs behaviorally, whereas the structural perspective asks *where in the network* the damage is located and whether it can be read from the weights alone. Finally, no line of work has organized this evidence for the decentralized, resource-constrained Fog–Edge conditions of the emerging Web of Agents (Section 3.1) [19, 17], where the performance–security and structure analysis trade-off can decide whether a compressed model can serve as a trustworthy agent.

Ultimately, closing this gap calls for a modular, scalable, and hardware-portable framework, in which empirical benchmarks, adversarial red-teaming, deep learning first-principles, and XAI literature operate as interchangeable components applied to the same model states — instantiated using the GGUF format as the post-training quantization case study — and extensible to additional model states and evaluation perspectives. In the following section, the Existing Opportunities Section (3.5) decomposes this requirement into the concrete opportunities that ground the research questions of Chapter 1, Section 1.3.

3.5 Existing Opportunities

Given the research gap established above, the following opportunities, which originate the research questions (Section 1.3, can be identified:

1. **Designing a modular, scalable, and extensible, multi-perspective evaluation framework.** The fragmentation of capability, security, and interpretability evaluation perspectives into non-interoperating silos opens room to architect a single framework that by design adheres to model states and evaluation extensibility, admitting empirical benchmarks (Im-evaluation-harness), adversarial red-teaming (garak), and deep learning first-principles XAI theories (e.g., HTSR and SETOL, Mechanistic Interpretability) as interchangeable modules under white, gray, and black-box observational regimes. Such a framework needs to compare a GenAI model state — full-precision against quantized variants, different fine-tuned and merged versions, with post-training quantization (GGUF or AutoRound) as the case study — while staying extensible to further states and evaluation perspectives.
2. **Integrating multidimensional, multi-perspective datasets across model states.** Because a framework’s lasting value lies in the data it produces, its evaluation Stages invite aggregation of their isolated artifacts — data-free weight-matrix metrics (HTSR and SETOL) joined

with empirical capability performance and Red Team AI outcomes — into a unified schema indexed by model state. Grounded in ISO/IEC 25012 data-quality predicates and DAMA-DMBOK lifecycle practices, such datasets should stay open enough to admit future models, security evaluations, alternative XAI theories, or complexity (e.g., Lempel–Ziv complexity) as new dimensions.

3. **Associating structural properties to empirical behavior through Exploratory Data Analysis (EDA) Instance-Space Analysis (ISA).** The assembled datasets make it possible to characterize the link between a model’s internal structure and its external behavior: Exploratory Data Analysis and inferential statistics can correlate data-free weight-matrix properties with capability performance (e.g., m-evaluation-harness) and adversarial resilience (e.g., garak), retaining only statistically significant structure metrics. Nevertheless complementary ISA module could project the same properties into a two-dimensional topology, exposing algorithmic footprints, strengths, and edge-case conditions for full-precision and compressed models.
4. **Predicting or explaining compression-induced degradation in performance and security.** With quantization as one enabler that makes edge deployment feasible, there is scope to test how far these analytical modules can predict or explain the degradation of low-bit variants (GGUF, AutoRound) suffer from critical OWASP Top 10 LLM risks (e.g., LLM01, LLM02, LLM05), aided by a dequantization map, to allow the exposing of the structural damage left in the weight matrices. This opening can extend to efficiency telemetry — latency, memory, throughput, and sustainability metrics (energy, CO₂) across heterogeneous backends (vLLM, llama.cpp, prima.cpp; CodeCarbon, EcoLogits) — under the constraints of decentralized Fog-Edge multi-agent environments.
5. **Deriving deployment blueprints and governance metadata for Fog-Edge systems and The NANDA Web of Agents.** Integrating the aforementioned, the accumulated evidence could motivate actionable blueprints for guided compression, model merging, targeted fine-tuning, or adaptive red-teaming, by mapping prompt/output complexity onto HTSR/SETOL structural signatures. At Hugging Face’s scale, this potential reaches toward confronting evaluation bias across millions of open models, the first steps of a general-purpose multi-perspective leaderboard, identifying vulnerable models or malicious agents (e.g., compression degradation or GGUF-based attacks), and enriching NANDA’s operational pillars (AgentFacts, Identity, Discovery, Trust, Communication) — or, where solutions remain out of reach, marking the next steps.

Chapter 4

Proposed Solution

This chapter establishes the conceptual foundations and architectural framework required to address the research gaps and opportunities identified in Chapter 3. The *Problem Setting* and *Formalization* are first defined, establishing the mathematical and observational boundaries necessary for the systematic evaluation of Generative Artificial Intelligence (GenAI) models. Building upon this conceptual baseline, the chapter then details the *Proposed Solution*, specifying the system requirements, software architecture, and evaluation criteria.

4.1 Problem Setting

To address the research gaps and opportunities identified in Chapter 3, the **ChaosAthena** framework is introduced. Named and inspired to reflect the Greek Goddess of wisdom and strategy, as well as the concepts of Criticality and Chaos that are explored in the dissertation, the framework strive to provide a multi-perspective assessment of Generative Artificial Intelligence (GenAI) models. Traditional state-of-the-art approaches frequently evaluate models in isolated empirical settings, failing to systematically capture relationships among, for example, the GenAI model, the network’s underlying weight-matrix structure, and its outputs with respect to capability performance and adversarial resilience. The ChaosAthena framework closes this gap by systematically accounting for GenAI states across structural, mechanistic, and behavioral dimensions [35].

This approach ensures that performance or security degradation, particularly, but not exclusively, that induced by low-bit quantization for GenAI models, with practical application on resource-constrained Fog–Edge environments, can be associated, for example, with weight-matrix properties derived from deep learning first-principles theories. The analysis can cover models in different states, including both base and fine-tuned variants, compressed, or merged models, under a set of strict observational boundaries. Post-training quantization serves as the primary case study throughout this dissertation.

4.1.1 Formalization

Building upon the concepts already introduced in Chapters 2 and 3, the following formalization organizes, from a high-level perspective, the most important observational and mathematical elements and constraints that serve as the foundational base for the Proposed Solution of this dissertation, in section 4.3 of the current Chapter.

4.1.1.1 Heterogeneity of GenAI Models

Let Σ denote a finite token vocabulary and Σ^* the set of all finite token sequences over Σ . We define $\mathcal{X}$ as the set of heterogeneous GenAI models, partitioned by provenance into base models, fine-tuned versions, merged models, and compressed variants:

$$\mathcal{X} = \mathcal{X}_{base} \cup \mathcal{X}_{ft} \cup \mathcal{X}_{merged} \cup \mathcal{X}_{compressed}. \quad (4.1)$$

We characterize each model $X \in \mathcal{X}$ by the tuple

$$X = (\alpha_X, \mathbf{W}_X, c_X). \quad (4.2)$$

Here, α_X represents the architecture descriptor (e.g., Transformer topology, number of parameters), $\mathbf{W}_X = (W_1, W_2, \dots, W_{L_X})$ serves as the ordered collection of learned weight matrices across the L_X layers, and c_X contains the configuration metadata. Behaviorally, the learned internal state induces a conditional distribution $p_X(O | I)$ over generated responses $O \in \Sigma^*$ given an input prompt $I \in \Sigma^*$.

4.1.2 Compression Operator and Variant Acquisition

Studying model states in terms of their compressed variants requires a compression operator, concretely, in this dissertation, for post-training quantization[90]. As a restriction, only base, fine-tuned, or merged models will be compressed, and never already-compressed model states. Let $\theta = (f, b, D_{cal}) \in \Theta$ denote a specific post-training quantization configuration consisting of a target format f (with all needed parameters, e.g., GGUF K_M regime, imatrix), a bit-width final characterization b , and an optional calibration dataset D_{cal} . We define the compression operation for quantization as

$$\mathcal{C} : \mathcal{X} \times \Theta \rightarrow \mathcal{X}_q, \quad \mathcal{C}(X, \theta) = X_q^\theta. \quad (4.3)$$

The full-precision model and its corresponding quantized states constitute the quantization variant family

$$\mathbb{V}(X) = \{X\} \cup \{X_q^\theta : \theta \in \Theta_X\}, \quad (4.4)$$

which is what this dissertation framework will focus on.

Furthermore, the above formalization requires a dequantization map dq that transforms a compressed variant back into a 16- or 32-bit numerical representation without altering the compressed

state. Because information discarded during quantization remains irrecoverable,

$$dq(\mathcal{C}(X, \theta)) \neq X. \quad (4.5)$$

This inequality assures that the dequantized matrices always reflect any structural degradation induced by quantization.

4.2 Observational Points and Model Assessment

A model variant exposes distinct points of observation, each exposing a different operational or structural dimension. We classify these into structural, mechanistic, and behavioral assessments. Based on what is described in Chapter 2, the needed points of observation for this dissertation framework are described and formalized.

4.2.1 Weight Matrices — HTSR and SETOL

Based on what is described in Chapters 2 and 3, in terms of foundational concepts (such as the definition of the Dirac delta function or basic Singular Value Decomposition) and state-of-the-art for this dissertation: the strictly additional, necessary descriptions and details are provided, particularly to extract structural properties from GenAI models [29, 30].

At the structural level, it is possible to analyze a model through its learned weight matrices. Before proceeding, a brief notational distinction is needed: in this subsection, it's used X to denote the weight correlation matrix, which is the standard from HTSR and SETOL. This should not be conflated with the formalization proposed for the Heterogeneity of GenAI models (4.1.1.1), which is already established, where each model is characterized by $X \in \mathcal{X}$ by the tuple $X = (\alpha_X, W_X, c_X)$.

Established the above, for a given layer, let $W \in \mathbb{R}^{N \times M}$ ($N \geq M$) be its 2D weight tensor, with an aspect ratio of $Q = N/M \geq 1$. Therefore, by isolating the learned structural correlations from random initialization noise by computing the $M \times M$ correlation matrix [29, 30]:

$$X = \frac{1}{N} W^\top W. \quad (4.6)$$

The structural quality of W is quantified via its Empirical Spectral Density (ESD). Formally, the ESD is the discrete probability measure of the non-negative eigenvalues λ_j (for $j = 1, \dots, M$) of X , expressed using the Dirac delta distribution $\delta(\cdot)$:

$$\rho_{emp}(\lambda) = \frac{1}{M} \sum_{j=1}^M \delta(\lambda - \lambda_j). \quad (4.7)$$

According to Random Matrix Theory (RMT), untrained or purely noisy weights yield an ESD bounded within a Marchenko-Pastur (MP) bulk[30]. In well-trained models, meaningful learned correlations escape this bulk, forming a heavy-tailed distribution [29]. Guided by Heavy-Tailed Self-Regularization (HTSR) phenomenology, the tail can be isolated beyond a lower threshold

λ_{min}^{PL} and fitted to a power-law density function:

$$\rho_{tail}(\lambda) = \rho_{emp}(\lambda \geq \lambda_{min}^{PL}) \sim \lambda^{-\alpha}. \quad (4.8)$$

The resulting exponent, α (Alpha), serves as the primary shape metric measuring the concentration of generalizing correlations. To penalize scale distortions such as correlation traps, HTSR, defines a scale-adjusted metric, AlphaHat ($\hat{\alpha}$), utilizing the spectral norm λ_{max} [30, 29]:

$$\hat{\alpha} = \alpha \log_{10} \lambda_{max}. \quad (4.9)$$

Connecting this with the Semi-Empirical Theory of (Deep) Learning (SETOL), an evaluation of the Exact Renormalization Group (ERG) condition is performed. SETOL dictates that the dominant eigencomponents forming the Effective Correlation Space (ECS) must satisfy a scale-invariant, volume-preserving transformation. Let $\tilde{X}$ be the correlation matrix truncated to the $\tilde{M}$ eigenvalues localized strictly in the ECS tail ($\tilde{\lambda}_i \geq \lambda_{min}^{ECS}$) [30]. The ERG condition ($\det X$) requires its determinant to approximate unity [30]:

$$|\det \tilde{X}| \simeq 1 \implies \sum_{i=1}^{\tilde{M}} \ln \tilde{\lambda}_i \approx 0. \quad (4.10)$$

Together, these metrics, among others that will be further explored in the next chapters, derived from the HTSR and SETOL [29, 30], provide data-free metrics that can be extracted. Ultimately, enabling deep learning first-principles diagnostics of a model's intrinsic generalization capacity and unknown learned weight matrices' structural problems [30, 72, 41].

4.2.2 Hidden Representations

Another observation point, related to the mechanistic assessment of models [35], unpacks the intermediate causal pathways inside the model architecture during a forward pass. For a given prompt I , the model generates a sequence of hidden state vectors $h^{(l)}(I) \in \mathbb{R}^d$ across its layers [129].

The decoding process translates these intermediate hidden states back into the vocabulary space by mapping them via a decoder matrix $W_{decoder} \in \mathbb{R}^{d \times |\mathcal{V}|}$ [141]. Normalizing the vector and applying a softmax operation produces a probability distribution over the vocabulary. Examining the resulting token trajectories explicitly traces the internal accumulation of information leading to the next-token prediction, exposing the mechanics of layer-by-layer reasoning [141].

4.2.3 Input–Output Assessment

The most common way to assess a model is to observe and evaluate the behavioral interaction between an input prompt I and the generated response O . Depending on how the assessment is performed, it can include the induced conditional distribution $p_v(O | I)$ [3]. This access point

supports multiple assessment possibilities, which, whenever utilized, will later be fully detailed or extended:

- **Complexity Assessment.** Define a multidimensional complexity mapping $\kappa : \Sigma^* \rightarrow \mathbb{R}^d$ to evaluate the structure of prompts and outputs. As illustrative examples, this mapping extracts dimensions such as Lempel–Ziv (LZ) complexity to measure inherent sequence repetitiveness, or compression complexity (via Zlib) to quantify algorithmic information [142, 143, 67].
- **Capability Performance.** Measure empirical task accuracy among standardized benchmark datasets. Supplying prompts configured for zero-shot or few-shot inference allows us to evaluate outputs versus known ground truths, computing exact accuracy and perplexity metrics [144, 125].
- **Red Team AI.** Subject the input–output interface to adversarial probing. By supplying boundary-testing prompts, such as jailbreaks or malicious injections, and classifying the resulting outputs to map failures to established taxonomies, such as the OWASP Top 10 for LLMs[26].

4.2.4 Observational Regimes

To systematically isolate different attributes or properties, whether structural, mechanistic, or behavioral, of a variant $v \in \mathbb{V}(X)$, it is necessary to define the set of all admissible observation points, denoted by $\text{Obs}(v)$. In the ChaosAthena framework, it categorizes the available observation points into three distinct access regimes: black-box, gray-box, and white-box[74]. Each analytical query pertaining to a specific model variant requires an appropriate subset of these observational perspectives; thus, this partitioning is a fundamental component of any analytical approach.

Given all the observation points detailed above, they need to be integrated and organized into three hierarchical observational regimes. These regimes strictly dictate data-extraction boundaries, matching the defined tuple $X = (\alpha_X, \mathbf{W}_X, c_X)$ that defines GenAI model heterogeneity[74]:

- **White-Box Access.** The white-box regime accesses all components. It executes behavioral inputs and outputs, traces intermediate representations, and directly inspects the actual weight matrices $\mathbf{W}_X$. This regime enables a full analysis and assessment of the model’s state across multiple dimensions.
- **gray-Box Access.** The gray-box regime accesses all behavioral and intermediate components, while strictly excluding the direct weight-matrix analysis used in HTSR/SETOL. It observes behavioral inputs and outputs, including the generated output tokens and the output probability distribution $p_X(O | I)$, as well as the internal hidden representations $h^{(l)}(I)$ generated during the forward pass. This allows for mechanistic explainability without the weight matrix structural analysis of the weight matrices [35].

- **Black-Box Access.** The black-box regime has access only to the inputs I and the outputs O , in the form of the generated output tokens, without access to the output probability distribution $p_X(O | I)$. Operating exclusively on the behavioral realization of the model's configuration c_X and architecture α_X .

4.3 Proposed Solution

To implement the formalization presented above, the proposed solution systematically structures the ChaosAthena framework into parallel, extensible, and scalable processing stages, corresponding stage modules, and a Hydra Orchestrator main pipeline. Each stage executes one or a group of functional modules, which may function as either custom-implemented operations or established third-party frameworks, generally open-source. By strictly decoupling these evaluation components, the framework architecture guarantees high extensibility and maintains accordance with the aforementioned observational regimes in subsection 4.2.4.

4.3.1 Main Stages and Modules

The main stages represent the fully implemented core components of this dissertation framework. They execute the primary operations, evaluations, and analyses:

- **Compression stage.** Executes the compression operator $\mathcal{C}$, generating low-bit variants or acquiring existing quantized states. It integrates modules such as GGUF (via *llama.cpp*) or AutoRound (*auto-round*) [121, 125, 125]. All generated variants are saved on Hugging Face.
- **Performance Evaluation Stage.** Operating under the gray-box regime, this stage utilizes the lm-evaluation-harness module to measure empirical capabilities and corresponding performance across standardized task benchmarks, computing zero-shot and few-shot accuracy.
- **Red Team AI Evaluation Stage.** Under the black box regime, it uses the garak module to automatically probe (adversarial testing) the model for security vulnerabilities [31], mapping failures to taxonomies such as the OWASP Top 10 for LLMs [26]. It also includes the *guardrail_eval* module, which performs reasoning-based safety classification to detect the harmfulness of prompts and responses [75].
- **Deep Learning First-Principles Stage.** When used alongside other black-box and gray-box Stages, it enables the white-box regime. This Stage utilizes the WeightWatcher module to perform data-free structural diagnostics. It computes HTSR and SETOL spectral metrics directly from the weight matrices to extract features such as AlphaHat ($\hat{\alpha}$) or identify phenomena such as rank collapse or correlation traps [30].

- **Multi-Perspective Dataset Creator Stage.** Ingests isolated artifacts from all evaluation Stages. For example, it aggregates and organizes the extracted model’s structural weight-matrix metrics and empirical behavioral outcomes (e.g., red team AI and performance Stages) into a unified, multidimensional schema based on the observed points accessed.
- **Veritas Analytics Stage.** Analyzes the datasets created by the Multi-Perspective Dataset Creator Stage. Its primary EDA (Exploratory Data Analysis) module uses statistical analysis to identify correlations between the model’s structural properties, and the degradation in performance and adversarial resilience (red team AI)[134, 74]. Ultimately, it also aggregates and reports the final statistical data and findings, ready for presentation in the dissertation.

4.3.2 Extensibility of Stages and Modules

The secondary stages and modules are designated as examples to demonstrate the modularity and extensibility of the framework, but more importantly, to provide other examples, and of the formalized Observational Points and Model Assessment regimes, which will be out of the experiment’s scope in this dissertation. While they are referred to as future work, at the same time, they show the framework’s modular extensibility and examples of additional framework opportunities:

- **Explainable AI (XAI) Stage.** Operating in the gray-box regime, this Stage utilizes the InTraVisTo module[129]. Being a particular example of Mechanistic Interpretability [35], it extracts intermediate hidden states to visualize token trajectories, unpacking the causal paths of the Transformer network layers[129].
- **Efficiency Evaluation Stage.** A black-box PoC that captures critical hardware telemetry, quantifying for example, latency, memory footprint, sustainability metrics, and token-generation throughput utilizing engines like vLLM and llama.cpp and frameworks such as CodeCarbon and EcoLogits [130, 132, 131].
- **Model I/O Analysis Stage.** A black-box Stage featuring a Complexity and Chaos Theory module. For example, it applies analysis such as Lempel–Ziv complexity applied to textual inputs and outputs, extracting complexity and unpredictability dimensions[67].
- **Veritas Analytics Stage (ISA module).** A secondary module within the Veritas Analytics stage that employs Instance Space Analysis (ISA) [134, 135]. It projects high-dimensional metrics into a 2D topology to visualize the exact algorithmic “footprints” in which, for example, specific compressed models stay robust or fail.

4.3.3 Architecture Framework by Phase

Based on the aforementioned Stages and modules, the ChaosAthena architecture organizes the framework into four distinct phases, divided into primary and secondary components in accordance with the experimental scope of this dissertation, while supporting future extensibility. The descriptions of these four phases are as follows:

- **Phase 1 — Model Preparation and Transformation.** The Compression Stage constitutes the initial stage for operations performed on a model prior to evaluation, implementing the retrieval-or-compute acquisition rule of the framework’s problem setting. Pre-compressed variants for major open-source GenAI models are widely available on platforms such as Hugging Face; therefore, direct ingestion of these variants is preferred, ensuring that the evaluated artifacts match those the community actually deploys. When a compressed version required for a specific experiment is unavailable, the implemented modules apply post-training quantization using the GGUF format or the AutoRound method (with AutoRound as future work 7.3), producing compressed states of the full-precision model for downstream analysis[125].
- **Phase 2 — Evaluation and Analysis.** This phase executes the previously described core analytical stages, operating under the defined observational boundaries. The Performance Evaluation and Red Team AI Evaluation Stages operate under gray and black-box regimes, respectively, and the Deep Learning First-Principles Stage, when executed alongside, all together provide additional data-free assessment of the model, and achieve a white-box regime. By using and performing experiments with these three Stages, the framework stays one Stage away from achieving total white-box assessment: the secondary XAI Stage[129].
- **Phase 3 — Dataset Aggregator.** In this phase, the Multi-Perspective Dataset Creator Stage acts as the central aggregator layer. It consumes the isolated, archived outputs generated throughout Phase 2 and aggregates these heterogeneous artifacts into extendable and multi-dimensional datasets. This phase bridges the raw aggregated data with the final analytical phase. To ensure the quality of all created datasets, the creation of datasets, whenever possible, is conducted in accordance with ISO/IEC 25012 data-quality predicates and DAMA-DMBOK lifecycle practices[82, 83].
- **Phase 4 — Analytics and Results.** Phase 4 serves as the analytics core of the framework by applying the previously described Veritas Analytics Stage.

4.3.3.1 Extensibility

To demonstrate the ChaosAthena architecture’s modular extensibility, several secondary Stages and modules can complement the ChaosAthena framework. These components illustrate how future experiments can be conducted and how new functionalities can be integrated into specific

phases without altering the present dissertation baseline, as described in the **Main Stages and Modules** subsection 4.3.1.

In Phase 2, as described in subsection 4.3.2, the Efficiency Evaluation, XAI, and Model Input/Output Analysis Stages can provide additional data and insights for the datasets created in Phase 3. Future evaluation modules may also be integrated into Phase 2 Stages, including alternative benchmarking engines (e.g., DeepEval) [145], additional red-team AI adversarial engines (e.g., DeepTeam and PyRIT) [146, 145], and emerging deep learning theories (e.g., Geometry of Mind/FRESH, Categorical Deep Learning) [147, 148]. In Phase 4, hosted on the Veritas Analytics Stage, the ISA module, is identified as the most important module for future experiments [134]. Only when datasets generated in Phase 3 with an adequate number of instances available for the proper selection evaluation of the ISA algorithm, will it be possible to draw significant conclusions about all variants analyzed throughout Phase 2[134].

4.3.4 Orchestration and Migration

4.3.4.1 Orchestration and Data Flow

Standard and modular configurations govern the execution of the main Hydra Orchestrator pipeline, stages, and corresponding modules in the ChaosAthena framework. Each performed experiment uses a hierarchical set of YAML files to define all active stages, target modules, and their required parameters, as well as the selected model variants, storage location, and compute backend. The Hydra Orchestrator pipeline composes these configurations and manages the end-to-end execution. Individual stages write their generated outputs strictly to a configured storage substrate (local or cloud-synchronized). The framework architecture, by design, restricts direct in-process memory sharing between Stages; downstream Stages (such as the Multi-Perspective Dataset Creator) read archived artifacts directly from storage, thereby enforcing loose coupling and strict independence.

4.3.4.2 Migration Toward a Microservice Architecture

The existing architecture isolates pipeline communication exclusively through configuration structures and stored artifacts, meaning each Stage already operates as a self-contained structural unit with explicitly defined inputs and outputs. This strict boundary seamlessly facilitates a future transition from an installable Python package to a fully distributed microservice architecture (MSA). In an MSA paradigm, each Stage deploys as an independent service complemented by active message brokering, demanding modifications only to the transport layer, while leaving the internal module logic unchanged.

4.3.5 Architecture Diagrams

The next two architecture diagrams in Figures 4.1 and 4.2 consolidate the information presented throughout Section 4.3.

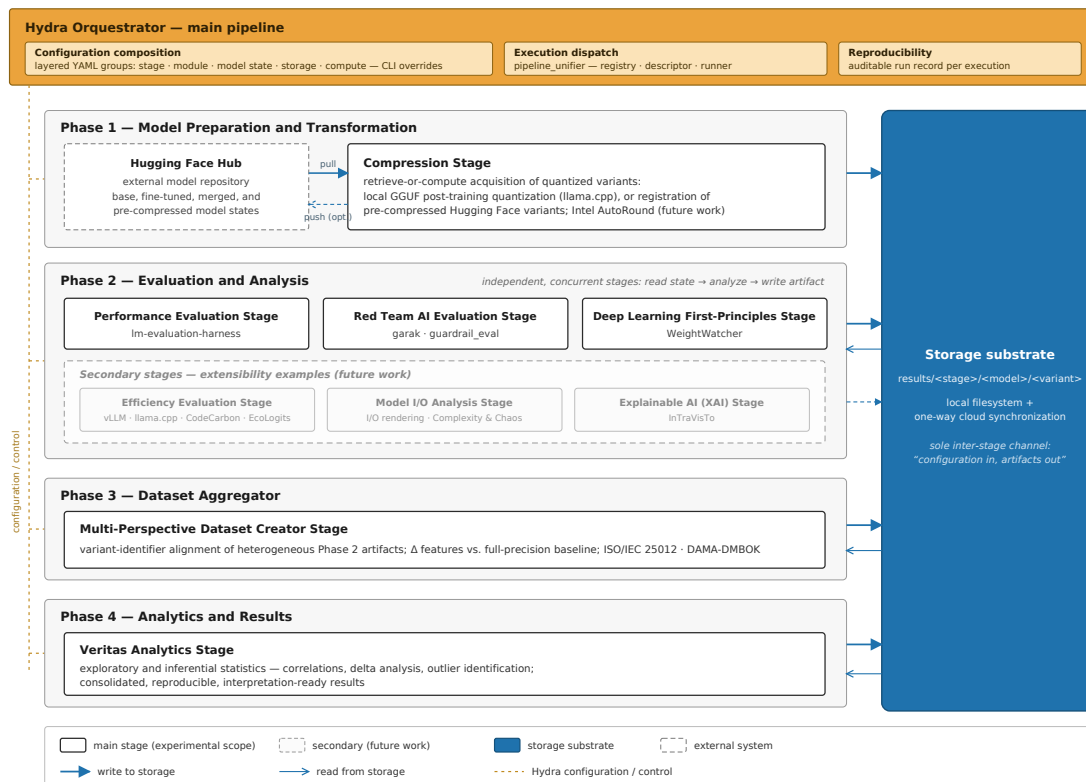


Figure 4.1: **ChaosAthena architecture organized by phases.** The Hydra Orchestrator pipeline composes and dispatches every execution; the four phases exchange data exclusively through the storage substrate under the “configuration in, artifacts out” contract. Faded, dashed components denote secondary stages kept as extensibility examples for future work.

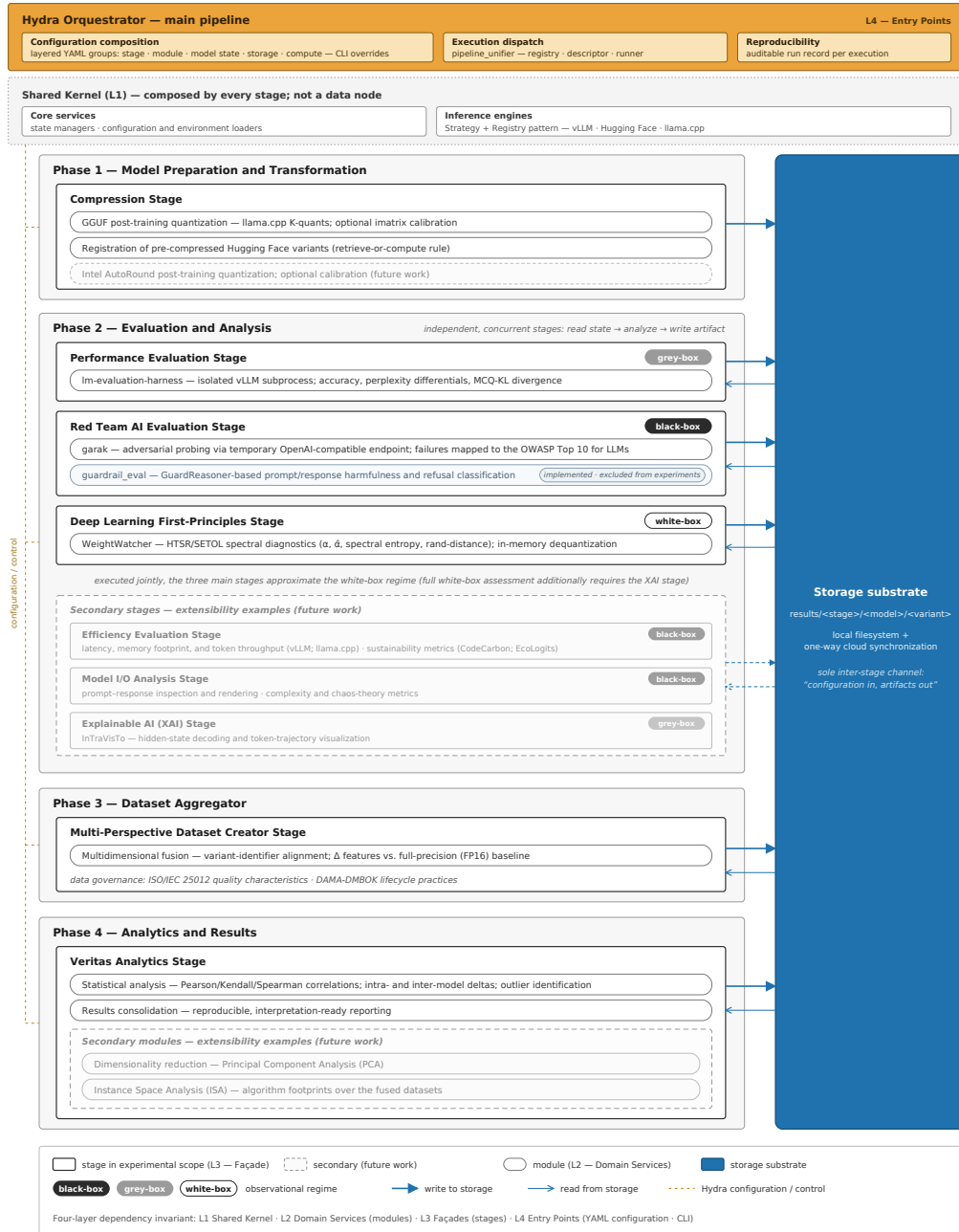


Figure 4.2: **ChaosAthena Hydra Orchestrator main pipeline.** All stages of each phase and their corresponding modules, mapped to the observational regimes and to the four-layer dependency invariant (L1 Shared Kernel, L2 Domain Services, L3 Façades, L4 Entry Points). Faded, dashed components are secondary extensibility examples reserved for future work; the guardrail module is implemented but excluded from the experiments.

4.3.6 System Requirements

The success of the proposed solution depends on satisfying a defined set of criteria. Tables 4.1 and 4.2 present, respectively, the functional and non-functional requirements, outlining the essential capabilities and operational boundaries of the framework. The implementation strategy employs the MoSCoW prioritization method: Must-have (core components), Should-have (important but non-essential features), Could-have (desirable and identified opportunities, extensions, or deferred to future work), and Won't-have (totally deferred for future work).

Table 4.1: Functional requirements

ID	Requirement	Priority
FR1	The framework must evaluate empirical task performance through standardized benchmarks in an isolated vLLM subprocess, logging accuracy, calculating perplexity differentials, and MCQ KL-divergence (Performance Evaluation Stage, lm-evaluation-harness).	Must
FR2	The framework must perform black-box adversarial red-teaming via a temporary OpenAI-compatible vLLM endpoint, mapping failures to the OWASP Top 10 for LLMs (Red Team AI Evaluation Stage, garak).	Must
FR3	The framework must extract data-free HTSR and SETOL weight-matrix diagnostics, dequantizing compressed variants in memory to preserve quantization-induced structural change (Deep Learning First-Principles Stage, WeightWatcher).	Must
FR4	The framework must support the optional transformation operation to apply post-training quantization through: GGUF K-quants or other modes with optional calibration setting (via llama.cpp) and must register pre-existing Hugging Face compressed variants (Compression Stage, GGUF).	Must
FR5	The framework must aggregate heterogeneous Phase 2 artifacts into a multidimensional dataset aligned by variant identifier, containing base-vs-compressed deltas (Multi-Perspective Dataset Creator).	Must
FR6	The framework must perform exploratory and inferential statistical analyses, including Pearson/Kendall/Spearman correlations; intra-, global-, and inter-model deltas; and outlier identification (Veritas Analytics Stage, statistical analysis module).	Must
FR7	The framework must support both single-target runs and orchestrated sweeps over multiple models and compressed variants in a single execution.	Must
FR8	The framework must consolidate and present the dissertation's results in a reproducible, interpretation-ready form (Veritas Analytics Stage).	Must

Continued on next page

Table 4.1 – continued from previous page

ID	Requirement	Priority
FR9	The framework should provide a guardrail safety classification over prompt-harmfulness, response-harmfulness, and refusal detection following the GuardReasoner methodology (Red Team AI Evaluation Stage).	Should
FR10	The framework could quantify dimensions of textual complexity over inputs and outputs (Model I/O Analysis, Complexity & Chaos Theory module).	Could
FR11	The framework could perform intermediate-representation and token-trajectory white-box visualization of a target model (XAI Stage / InTraVisTo).	Could
FR12	The framework could relate data-free structural features and I/O complexity to performance and security deltas via PCA or Instance Space Analysis, producing algorithm footprints (Veritas Analytics Stage, ISA module).	Could
FR13	The framework should measure inference-cost metrics (TTFT, ITL/TPOT, throughput) and sustainability metrics via CodeCarbon and EcoLogits across GPU and CPU-only backends (Efficiency Evaluation).	Could
FR14	The framework could integrate additional adversarial engines such as DeepTeam and PyRIT (Red Team AI Evaluation Stage).	Could
FR15	The framework could support additional quantization methods, such as AutoRound, and including local HQQ and AWQ/GPTQ generation (Compression stage).	Could
FR16	The framework could integrate additional benchmarking engines such as DeepEval (Performance Evaluation).	Could
FR17	The framework could integrate additional statistical, causal-inference, or explainability frameworks (Veritas Analytics Stage).	Could
FR18	The framework could detect data contamination in model inputs and outputs (Model I/O Analysis).	Could
FR19	The framework could quantify the epistemic uncertainty of models (Model I/O Analysis) [149].	Could
FR20	The framework could incorporate additional first-principles theories.	Could
FR21	The framework could provide additional dataset-creation modules, such as graph-based or time-series builders (Multi-Perspective Dataset Creator Stage).	Could

Table 4.2: Non-functional requirements

ID	Requirement	Priority
NFR1	The framework must keep the main Hydra Orchestrator pipeline, stages and modules fully decoupled, communicating only through a “configuration in, artifacts out” contract with no direct in-process memory sharing.	Must
NFR2	The framework must scale to sweep across multiple model architectures, states, and large prompt volumes, supporting automation of external model repositories.	Must
NFR3	The framework must admit full-precision, fine-tuned, merged, and compressed states without modification of the framework core, encapsulating model-specific chat templates and parameters in an Adapter hierarchy when needed.	Must
NFR4	The framework must compose every experiment from layered Hydra YAML groups with on-the-fly CLI override capabilities.	Must
NFR5	The framework must produce a complete, auditable run record documenting the exact compositional state of every execution.	Must
NFR6	The framework must persist all artifacts to a configurable storage substrate supporting local filesystems and cloud synchronization.	Must
NFR7	The framework must be distributed as a pip-installable Python package that relies on isolated, backend dependencies.	Must
NFR8	The framework must remain extensible through additive changes, using classical software design patterns (e.g., Strategy, Factory, Façade).	Must
NFR9	The framework must operate portably across high-resource GPU instances (e.g., A100/H100), constrained environments (e.g., T4), and CPU-only nodes.	Must
NFR10	The framework must ensure that the created datasets satisfy ISO/IEC 25012 data-quality characteristics and DAMA-DMBOK lifecycle practices.	Must
NFR11	The framework should enforce process isolation and control memory allocation by dispatching heavy inference to isolated subprocesses, preventing residual memory pollution or OOM errors of the main orchestrator and all executed experiments.	Should
NFR12	The framework should provide a user-friendly execution experience via per-Stage Jupyter notebooks featuring absolute code-path parity to the scriptable CLI.	Should

Continued on next page

Table 4.2 – continued from previous page

ID	Requirement	Priority
NFR13	The framework could have a deployment version that is migrated to a microservice architecture, requiring only transport-layer modifications around the storage substrate.	Could

4.3.7 Evaluation Criteria

To evaluate the ChaosAthena framework, first, a verification against the system requirements established in Section 4.3 is performed; second, the Research Questions in Section 1.3 from Chapter 1 are validated.

Chapter 5

ChaosAthena Implementation

This Chapter presents the software engineering aspects of the **ChaosAthena** framework implementation. It instantiates the theoretical architecture outlined in Chapter 4 as a functional, installable Python artifact orchestrated by Hydra, and establishes the foundation upon which the experimental setup described in Chapter 6 is constructed. Furthermore, in focusing on the architectural realization of the Hydra Orchestrator main pipeline, multiple Stages, and corresponding modules, the scope of the primary sections is restricted to fully implemented Stages that constitute the main pipeline. To maintain architectural clarity, essential examples, code snippets, CLI execution commands, or other relevant implementation details not included here are provided in the Appendix A of this dissertation.

5.1 Core Architecture and Design Patterns

Before describing the specific logic of any individual Stage, it is necessary to formalize the engineering foundations shared across the entire framework. First, ChaosAthena is a *framework* in the sense of [34], who identify inversion of control as what separates a framework from a library: the components a user supplies are invoked by the framework, rather than calling into it. That is the relationship used—the Hydra orchestrator owns the execution flow, and the third-party tools (garak, lm-evaluation-harness, WeightWatcher) are invoked as interchangeable plug-ins behind Stage contracts, never the reverse. The wrapped tools are not the contribution; the architecture, contracts, and extension points set out in Section 1.4. The same criteria classify the utilized evaluators: garak’s harness invokes the probes, generators, and detectors registered with it [31], and lm-evaluation-harness invokes the models it scores [48]—both named frameworks by their authors on exactly this described basis.

Second, in terms of architecture, it is built upon a strict compositional philosophy. The emphasis is placed heavily on separating the shared kernel algorithms (such as inference engines and base model-loading strategies) from the independent pipeline stages implementation. The framework is delivered as a single, cohesive Python package where its constituent Stages and domain

modules are entirely decoupled. They communicate exclusively through a highly configured orchestration layer — reading configurations as inputs from the `pipeline_unifier` and writing archived artifacts to a storage substrate as outputs. This strict **“configuration in, artifacts out”** contract eliminates direct in-process memory sharing or functional calls between distinct Stages, thereby, as mentioned in Chapter 4, effectively positioning the current framework as a near-drop-in candidate for a future distributed Microservice Architecture (MSA).

5.1.1 The Installable Python Package

The framework is implemented as a pip-installable Python package, `ChaosAthena`, requiring Python 3.10 or higher. The build and distribution system relies on a `pyproject.toml` configuration that isolates core dependencies (such as `vllm`, `hydra-core`, `transformers`, `WeightWatcher`, and `gguf`) while appropriately handling pipeline-specific extras. The package is designed as a ready-to-use research artifact, and execution is handled through local Python module calls using a unified dispatcher.

The top-level repository structure enforces an additive localized mapping convention that strictly separates the operational layers. Specifically, domain-specific modules (Layer 2) reside in dedicated subsystem directories such as the `src/modules/weight_watcher/`. The execution logic is encapsulated in pipeline façades (Layer 3) located in the `src/pipeline/` directory (e.g., `htsr_setol_stage.py`). Entry points (Layer 4) are consistently provided through dedicated `src/run_*.py` scripts, a unified `src/run_pipeline.py` orchestrator, and user-friendly `notebooks/*.ipynb` files.

5.1.2 Configuration and Orchestration with Hydra

All execution within `ChaosAthena` is configuration-driven and orchestrated end-to-end by Hydra [150]. Hydra dynamically composes any experiment’s parameters from layered YAML configuration groups that independently define the pipeline behavior, the specific Stage and module parameters, the target-model states (e.g., full-precision baseline or quantized GGUF variants) [121, 151], the storage destination, and the compute backend. This layered approach enables hierarchical parameterization while preserving the ability to apply on-the-fly overrides directly from the Command Line Interface (CLI). Furthermore, the Hydra Orchestrator dispatcher recognizes the reserved scope value `all:` before composition, it enumerates the model group `configs/<stage>/model/*.yaml` and issues one composed run per discovered model, which is equivalent to an exhaustive `--multirun` over the full group but requires no manual model list. Additionally, Hydra generates a fully reproducible run record (`hydra.run.dir`) for every execution, documenting the exact compositional state of the framework for any needed subsequent analysis.

5.1.3 Storage-Driven Decoupling and the Artifact Substrate

The framework enforces rigorous isolation between its analytical phases through a storage-driven artifact substrate. Operating strictly under the “**configuration in, artifacts out**” contract, established in the architecture design, each Stage independently generates its results and writes them to a standardized output directory tree (`results/<stage>/<Model>/<variant>/`). These output artifacts are subsequently persisted to a configurable storage backend, which natively supports both local filesystems and one-way synchronization to cloud services such as Google Drive.

A strict dependency invariant is enforced; Stages can share core services or output artifacts without direct inter-stage invocation, achieving loose coupling. This storage-mediated hand-off allows the Phase 1 compression, Phase 2 evaluations, and the subsequent Phase 3 and Phase 4 dataset creation and analysis to operate asynchronously and in parallel. Consequently, this isolation guarantees immediate extensibility to distributed microservice environments.

5.1.4 Execution Mode and Compute Backends

5.1.4.1 Execution and Compute Backends

ChaosAthena provides hardware-portable execution modes that accommodate both automated batch processing and interactive exploration. All Stages and pipeline strictly adhere to a unified execution pattern, preserving structural uniformity across the entire framework. Users can interact with the system via two primary pathways: unified Command Line Interface (CLI) dispatches (e.g., `python src/run_pipeline.py <stage_name>`), which are highly scriptable and support immediate Hydra parameter overrides; or guided, per-Stage Jupyter notebooks (e.g., `notebooks/<stage_name>.ipynb`), which maintain absolute code-path parity with the CLI execution.

The framework supports dynamic run modes, enabling single-target evaluations as well as comprehensive automated sweeps across multiple base models and their compressed variants in a single run. To ensure compatibility with any GenAI model for execution, analysis, validation, and adaptability to the Fog-Edge computing paradigm, particularly in hybrid CPU-GPU use, ChaosAthena is designed to support a wide spectrum of compute backends.

Table 5.1: Compute backends targeted during experimentation

Node (role)	System RAM	GPU VRAM	Disk
H100	176.9 GB GB	95.6 GB	235.7 GB
A100	167.1 GB	80 GB	235.7 GB
T4	12.7 GB / 51.0 GB	15.0 GB	235.7 GB
CPU-only	12.7 GB / 51.0 GB	—	225.8 GB

5.1.4.2 Selecting the Model Scope for Experiments

Independent of *which* Stage runs, ChaosAthena exposes one uniform convention to choose the *model scope* of a run. A Stage’s model group is the directory of per-model YAML files it reads (`configs/<Stage>/model/*.yaml`); every file there declares one model and its compressed variants. The scope is selected with a single override:

- **single** — `model=<Name>` runs the Stage on exactly one configured model;
- **sweep** — `--multirun model=<A>,<B>,...` runs a chosen subset;
- **all** — `model=all` runs the Stage on *every* model whose YAML exists in that Stage’s model group, with no need to enumerate them.

The **all** keyword is resolved by the Hydra Orchestrator pipeline, not by any individual Stage: it expands to the list of all model-config Stage stems discovered in every `configs/<Stage>/model/` and dispatches every Stage once per model, reusing the ordinary composition path (4.3.4). Because the expansion happens in the orchestrator and every Stage obeys the same “**configuration in, artifacts out**” contract, `model=all` behaves identically for all Stages across all four phases: the Compression Stage from Phase 1, compresses defined models or retrieves every configured model, and all the evaluation and analysis stages of Phase 2 evaluate and analyze every configured model variant; and the Phase-3/4 stages, which take artifacts *paths*, use and aggregate all the artifacts the earlier phases produced. In other words, one keyword generalizes “run this Stage over the whole configured corpus” to the entire framework. Idempotency (5.1.3) makes the **all** keyword safe to re-issue: already-completed models are skipped, so an interrupted corpus run resumes where it stopped. Additionally, whenever it is desired to execute only **one specific Stage of any Phase**, independently in the framework, the same **all** keyword can be used on that particular Stage.

5.1.5 Design Patterns

The ChaosAthena framework explicitly utilizes *composition over inheritance* as its central architectural feature. The framework, for example, avoids relying on a single, overarching base class for Stages or modules; instead, it provides a Shared Kernel of cross-cutting services (encompassing state managers, configurations, and environment loaders) that individual Stages import and compose as required.

Every Stage operates as a self-contained unit complying with a universal lifecycle contract: *configure* → *run* → *write-to-storage*. Framework modularity and extensibility is facilitated along two primary axes: adding entirely new Stages, or inserting new domain modules into existing Stages. Thus, from the explanations about design patterns in Chapter 2, this is achieved through the application of classical software design patterns:

- **Strategy + Registry/Factory.** Predominantly utilized in the inference domain (e.g., `InferenceEngineBase` and `EngineRegistry`), this pattern dynamically selects

between execution backends (`vllm`, or `llama_cpp`) based on configuration, without modifying pipeline code.

- **Strategy (Load Formats).** Dictates how different model variants (e.g., full precision, GGUF) are resolved into runnable handles using strategies such as `NativeGGUFStrategy`.
- **Façade.** Each Stage implements a single entry function (e.g., `run(cfg), evaluate(cfg)`) that abstracts the complex wiring and orchestration of its underlying Layer 2 modules.
- **Adapter.** Adapts highly variable, GenAI model-specific chat templates, or other needed specific parameters to ensure standardized prompt structures or GenAI model-specificities.

5.1.6 Repository and Directory Structure

The repository is built on a strict four-layer dependency invariant that prevents circular dependencies while keeping shared utilities isolated from domain-specific execution. The system structurally maps L1 (Shared Kernel) for strategies, L2 (Domain Services) for isolated analytical modules (e.g., `src/modules/garak_eval/`, `src/modules/lm_harness_eval/`), L3 (Façades) for pipeline orchestrators within `src/pipeline/`, and L4 (Entry Points) for the YAML configurations and execution scripts.

When adding a new module to an existing Stage, developers add the needed logic in the corresponding Layer 2 directory and, if needed, register it via the Strategy+Registry pattern. When adding a completely new Stage, the framework, always follows a minimal additive “four-file shape” (a configuration YAML file, a CLI entry script, a façade function, and a Jupyter notebook). The definitive architectural mapping of the codebase—reflecting the orchestration and shared components, the implemented Hydra Orchestrator main pipeline and Stages, and their corresponding structural paths—is given in Tables 5.2, 5.3, and on an annotated directory tree of the ChaosAthena framework illustrated in Appendix A, in the Listing A.1.

Table 5.2: Orchestration and shared components.

Component Name	Description
ChaosAthena	Package and repository name.
pipeline_unifier	Hydra storage orchestrator: registry, descriptor, runner.
core, inference	Shared Kernel (Layers 1 & 2): Strategies, and inference engines.

Table 5.3: Implemented stages and underlying software/statistical module integrations.

Stage Name	Code/Directory Mapping	Integrated Modules
Performance Evaluation	performance_eval	lm-evaluation-harness
Red Team AI Evaluation	red_ai	garak, guardrail
Deep Learning First-Principles	htsr_setol	WeightWatcher
Compression	compression	GGUF (llama.cpp and Intel AutoRound) [152, 125, 124]
Multi-Perspective Dataset Creator	datasets_creator	Structural data substrate ingestion (pandas, numpy) mapped to ISO/IEC 25012:2008 and DAMA-DMBOK standards
Veritas Analytics	veritas_analytics	Inferential statistics (scipy.stats for Pearson r , p -values, and Fisher z -transforms), regression testing (statsmodels for least-squares and multiple-comparison adjustments), EDA (scikit-learn, AutoViz), and plotting (matplotlib)

Note: guardrail_eval is fully implemented and tested as a module of the Red Team Stage, but is outside the experimental scope of this dissertation (Appendix A; Section 7.2).

5.2 Integrated Frameworks and Tools

To implement the multi-perspective assessment and formalization in Chapter 4, the ChaosAthena framework uses and coordinates a specialized stack of third-party frameworks. Operating within the Layer 2 Domain Services (`src/modules/`), these tools function as the underlying engines for the framework’s Stages. Building on what was already described in Chapter 2 2 and Chapter 4, this section provides detailed information on the configurations, parameters, and roles these tools perform when orchestrated by ChaosAthena.

5.2.1 Garak

The `garak` framework functions as the module for the Red Team AI Evaluation Stage (`src/modules/garak/`), facilitating automated security probing to discover vulnerabilities in generative-AI models [31]. Operating under a black-box observational regime, ChaosAthena utilizes `garak` through a generator abstraction: it dynamically boots a temporary vLLM or llama.cpp OpenAI-compatible endpoint, targets it with defined probe families, and immediately tears down the server upon completion to reclaim hardware resources. The generated responses are subjected to parsing by `garak`’s native detectors and, if needed, subsequently organized according to the configured OWASP Top 10 taxonomy for LLMs described in Chapter 2.

Table 5.4: garak configuration: example of parameters and descriptions.

Name	Example of Values	Description
model_type	openai	Black-box targeting via a vLLM OpenAI-compatible server.
OPENAI_API_BASE	http://127.0.0.1:8000/v1	Local vLLM endpoint generated dynamically by the VllmEngine runner.
probes	"all"	Full exhaustive probe sweep across the target model state.
generations	5	Generation attempts per adversarial prompt, to capture stochastic failures.
parallel_attempts	500 or dynamic	Concurrency parameter for generator execution against the local vLLM endpoint.
experiment.taxonomy	"owasp"	Maps failures to the OWASP Top 10 for LLMs via garak-native categorization.

Note: API = Application Programming Interface; LLM = Large Language Model; OWASP = Open Worldwide Application Security Project; vLLM = Virtual Large Language Model.

5.2.2 lm-evaluation-harness

The `lm-evaluation-harness` framework [48], developed by EleutherAI, is the performance evaluation module (`src/modules/lm_harness/` of ChaosAthena's Performance Evaluation Stage. Operating in a gray-box regime—distinguished from pure black-box execution by its necessary reliance on token-level log-probabilities—its core function is to provide a standardized, reproducible interface for evaluating language models across a variety of zero-shot and few-shot tasks. Within ChaosAthena, the evaluation workload is dynamically dispatched to an isolated vLLM subprocess, enforcing strict memory-use boundaries and preventing residual memory artifacts from polluting the overarching Hydra orchestrator or OOM errors. Furthermore, by aggregating the benchmark scores (`results_*.json` and the raw evaluation samples (`samples_*.jsonl`,

archiving these raw samples enables, if needed, allows calculating comparative metrics, such as perplexity differentials and Multiple-Choice-Question (MCQ) Kullback–Leibler (KL) divergence. Ultimately, allowing to quantify the exact performance-degradation gap between full-precision baseline models and their low-bit compressed counterparts.

Table 5.5: `lm-evaluation-harness` configuration: example of parameters and descriptions.

Name	Example of Values	Description
<code>tasks</code>	<code>wikitext,</code> <code>mmlu,</code> <code>hellaswag,</code> <code>gsm8k, ifeval</code>	Selected standardized NLP, reasoning, and instruction-following benchmarks.
<code>num_fewshot</code>	<code>null</code> (task defaults)	Evaluation-context grounding parameter specifying the number of demonstration examples.
<code>batch_size</code>	<code>auto</code>	Dynamic batch sizing applied via the harness configuration to maximize throughput.
<code>engine</code>	<code>vllm</code>	Inference backend utilized for accelerating the evaluation.
<code>gpu_memory_utilization</code>	<code>0.90</code>	Reserved VRAM percentage to prevent Out-of-Memory (OOM) errors during batched inference.

Note: KL = Kullback–Leibler; NLP = Natural Language Processing; VRAM = Video Random Access Memory.

5.2.3 WeightWatcher

To operationalize the Heavy-Tailed Self-Regularization (HTSR) and Semi-Empirical Theory of Learning (SETOL) theories described in Chapter 2, 3 and Chapter 4, the WeightWatcher diagnostic framework is integrated as a module (`src/modules/weight_watcher/`) [41, 29, 30], which, when used alongside other Stages, enables a white-box evaluation of a generative AI model’s states. Operating completely independently of any inference engine or dataset, this module directly examines the learned mathematical structure of the model’s weight matrices. To evaluate compressed variants, the module leverages the `transformers` library to intercept quantized GGUF states, temporarily dequantizing the target matrices to full precision in memory, thereby applying the formalized Chapter 4 dequantization map dq (Eq. 4.5). This preserves all quantization-induced structural changes while enabling the Singular Value Decomposition (SVD) operations necessary to extract the Empirical Spectral Density (ESD) for each model layer weight matrices. Finally, in this dissertation, the framework restricts its focus from all the WeightWatcher extracted

metrics, to a critical subset of spectral metrics—such as, but not limited to, the HTSR power-law Alpha (α), AlphaHat ($\hat{\alpha}$), and the Spectral Entropy—which are subsequently archived to the storage substrate.

Table 5.6: `WeightWatcher` configuration: example of parameters and descriptions.

Name	Example of Values	Description
<code>fit</code>	<code>'PL'</code>	Forces the tool to fit Empirical Spectral Densities (ESDs) to a Power Law (PL).
<code>randomize</code>	<code>true</code>	Enables calculation of the rand-distance structural metric by comparing to a randomized-spectrum equivalent.
<code>min_evals</code>	50	Filters out trivially small matrices to ensure statistical validity of the power-law fit.
<code>mp_fit</code>	<code>true</code>	Fits the bulk of the spectrum to Marchenko–Pastur random-matrix theory.

Note: ESD = Empirical Spectral Density; HTSR = Heavy-Tailed Self-Regularization; PL = Power Law; SETOL = Semi-Empirical Theory of Learning.

5.2.4 Inference Engines

The ChaosAthena framework achieves a hardware-portable design by embedding highly optimized serving backends—specifically `vLLM` and `llama.cpp` [39] — directly into the L1 Shared Kernel (`src/inference/vllm_engine.py` and `src/inference/llama_cpp_engine.py`). Utilizing the Strategy design pattern, the framework dynamically selects the appropriate execution engine based on the Hydra configuration. In general, `vLLM` provides high-throughput, PagedAttention-accelerated GPU inference, which the Performance and Red Team Stages use in the experiments.

For compressed variants, the serving path applies the dequantization map `dq` (4.5) to reconstruct the quantized model weights values in FP16 before loading; since quantization error is irreversible (4.5), the executed weights are numerically the quantized state, evaluated under a uniform FP16 arithmetic path. This simulated-quantization protocol follows standard practice in the quantization literature and, on data-center GPUs where native GGUF kernels are unsupported or slow, is also the throughput-optimal way to evaluate thousands of prompts per variant [99, 101].

Conversely, `llama.cpp` enables rigorous CPU-only or hybrid GPU-CPU execution, which is critical, for example, assessing model viability within resource-constrained Fog-Edge environments. While these engines inherently provide the low-level latency (Time-to-First-Token, Inter-Token Latency) and throughput metrics necessary for operational profiling, the full aggregation of these metrics into a dedicated Efficiency Evaluation stage (incorporating sustainability tracking via

CodeCarbon and Ecologits modules), alongside all the results of other stages, in this dissertation is established as Future Work, referenced in Chapter 7 Section 7.3.

Table 5.7: Inference backends: example of configuration parameters and descriptions.

Name	Example of Values	Description
name	vllm	Inference strategy for high-memory GPU environments, optimizing batching.
name	llama_cpp	Inference strategy for constrained CPU or CPU+GPU edge-simulation environments.
dtype	float16	Compute data-type precision utilized during forward passes.
max_model_len	8192	Maximum context-window boundary enforced for the model during execution.

Note: CPU = Central Processing Unit; GPU = Graphics Processing Unit; vLLM = Virtual Large Language Model.

5.2.5 Quantization Engines

The Phase 1 Compression Stage is implemented as a **Factory** skeleton capable of executing Post-Training Quantization (PTQ). The experiments primarily use pre-compressed model states sourced directly from Hugging Face hub. However, to get any needed compressed variant that doesn't exist on Hugging Face, the Compression Stage also integrates local PTQ capabilities (`src/compression/quantization/gguf_quantizer.py`). The GGUF format quantization leverages the `llama.cpp` engine to execute block-wise compression (e.g., Q4_K_M, Q3_K_M), utilizing superblocks to allocate higher bit-widths to identified critical model layers.

5.3 Phase 1 Implementation

Based on what was described in Chapter 4 and in the section 5.2.5, Phase 1 transforms full-precision (16-bit) generative-AI models into quantized states, providing all the model variants for subsequent evaluations. In practice, the Compression Stage (`src/pipeline/compression_stage.py`) provides a quantization-export interface: while quantizing, it also exports the quantized model to Hugging Face, allowing its reuse whenever necessary. By employing Strategy and Factory design patterns, it dynamically dispatches workloads to specific quantization strategies (e.g., GGUF). This guarantees that downstream modules can compare any base model against its

Table 5.8: Quantization engines: example of configuration parameters and descriptions.

Name	Example of Values	Description
quantization_type	gguf	Target export format for local post-training quantization dispatch.
profile	Q4_K_M	Medium-precision K-quant (GGUF) profile balancing size and perplexity.
temp_fp16_keep	false	Discards intermediate full-precision weights to conserve disk space.
calibration_dataset	NeelNanda/pile-10k	Example of calibration subset.

Note: GGUF = GPT-Generated Unified Format; PoC = Proof-of-Concept; PTQ = Post-Training Quantization.

compressed variants, realizing the compression operator $\mathcal{C}$ and variant family $\mathbb{V}(X)$ formalized in Chapter 4.

The Compression Stage lifecycle follows a strict sequence: acquire the base model, apply the quantization strategy, export the model-state directory, and write a tracking file (`<variant>_status.yaml`) to ensure idempotency. This idempotency mechanism is implemented across all framework Stages—for example, to avoid unnecessary operations or redundant computation following errors or experiments crashes. By design, the Stage supports a dual-mode logic: it can generate novel compressed states locally, but pre-existing community-quantized states sourced from Hugging Face are predominantly used, with full user control to ensure comparable quantization formats across models, so the resulting variants can be compared consistently across all experiments.

5.3.1 GGUF Format Quantization

The GGUF strategy (`src/modules/compression/quantization/gguf_quantizer.py`) uses the `llama.cpp` backend described in Section 5.2.5 and uses it to convert any full-size model to a quantized state in the GGUF format. It loads the full-precision baseline, if needed, invokes the isolated shell processes required for calibration and block-wise mixed-precision assignment, and verifies the structural integrity of the exported `.gguf` artifact before signaling completion to the Phase 1 orchestrator.

5.3.2 Example of The Retrieve-or-Compute Path

To provide a concrete example, in terms of a user and architecture perspective. When Phase 1 realizes the compression operator $\mathcal{C}$ and the variant family $\mathcal{V}(X)$ (Eqs. 4.3–4.4) under a single *retrieve-or-compute* rule, dispatched by the **Factory** pattern on the `quantization_type` key. The same configuration supports both branches: the framework either *fetches* an existing quantized artifact from Hugging Face via the links declared per model, or *computes* a new one using `llama.cpp`.

Configuration Schema within the compression Stage. A model specification block (YAML file) orchestrates the baseline full-precision parameters and an array of target compressed variants for acquisition. Rather than triggering downstream quantization routines, the ingestion architecture maps already-defined remote Hugging Face-required information directly to predefined GGUF modes (e.g., Q4_K_M). As illustrated in Listing 5.8, this schema also defines automatic precision resolution (`dtype: auto`), and distinct tokenizer bindings (`tokenizer_hf_id: Qwen/Qwen3-0.6B`).

```

1 model:
2   name: Qwen3-0.6B
3   full_size:
4     hf_id: Qwen/Qwen3-0.6B
5     tokenizer_hf_id: Qwen/Qwen3-0.6B
6     dtype: auto
7   compressed_versions:
8     - variant: Q4_K_M
9       hf_id: unsloth/Qwen3-0.6B-GGUF
10      filename: Qwen3-0.6B-Q4_K_M.gguf
11      tokenizer_hf_id: Qwen/Qwen3-0.6B
12      quantization_format: gguf
13     - variant: Q3_K_M
14       hf_id: unsloth/Qwen3-0.6B-GGUF
15       filename: Qwen3-0.6B-Q3_K_M.gguf
16       tokenizer_hf_id: unsloth/Qwen3-0.6B-GGUF
17       quantization_format: gguf
18     - variant: Q2_K_L
19       hf_id: unsloth/Qwen3-0.6B-GGUF
20       filename: Qwen3-0.6B-Q2_K_L.gguf
21       tokenizer_hf_id: Qwen/Qwen3-0.6B
22       quantization_format: gguf

```

Listing 5.1: `configs/compression/model/Qwen3-0.6B.yaml` — Configuration for Remote Artifact Retrieval

Configuration in (compute branch). When the required variant is absent, the same schema switches to the compute branch by selecting the `gguf` strategy and a K-quant profile; an optional parameter (`calibration_dataset`) can be given to calculate the importance imatrix (Listing 5.2).

```

1 quantization_type: gguf           # Factory dispatch key
2 profile: Q4_K_M                  # or Q3_K_M / Q2_K_L
3 temp_fp16_keep: false
4 calibration_dataset: null         # set e.g. NeelNanda/pile-10k to build an
   imatrix

```

```
5 export_to_hub: true # publish result for reuse
```

Listing 5.2: configs/compression/quantization/gguf.yaml — compute (local quantization)

Execution. A single dispatched call runs either branch identically; the model scope (single, sweep, or all) follows [5.1.4.2](#).

```
1 # single model:
2 python src/run_pipeline.py compression model=Qwen3-0.6B profile=Q4_K_M
3 # chosen sweep:
4 python src/run_pipeline.py --multirun compression model=Qwen3-0.6B,Qwen3-8B profile
  =Q4_K_M
5 # ALL configured models (every configs/compression/model/*.yaml):
6 python src/run_pipeline.py compression model=all profile=Q4_K_M
```

Listing 5.3: CLI — Compression: single, sweep, or all configured models.

```
1 #@title Compression --- run parameters {display-mode: "form"}
2 #@markdown Choose the model scope. Use the literal string **all** to run every
3 #@markdown configured model in configs/compression/model/, or list specific names.
4 run_scope = "all" # @param ["all", "custom"]
5 custom_models = "Qwen3-0.6B, Qwen3-8B" # @param {type:"string"}
6 profile = "Q4_K_M" # @param ["Q4_K_M", "Q3_K_M", "Q2_K_L"]
7 storage = "gdrive" # @param ["gdrive", "local"]
8 results_path = "/content/drive/MyDrive/Thesis_results/compression" # @param {type:"
  string"}
9
10 # Papermill-style parameter list consumed below (overridable at `papermill -p`):
11 configs_to_run = ["all"] if run_scope == "all" else [m.strip() for m in
  custom_models.split(",")]
12 config_path = "configs/compression"
13
14 # Dispatch one call per model (the orchestrator expands "all"); @param path ->
  output.base_dir
15 import subprocess
16 for model in (["all"] if configs_to_run == ["all"] else configs_to_run):
17     cmd = ["python3", "src/run_pipeline.py", "compression",
18           f"model={model}", f"profile={profile}", f"output.base_dir={results_path}"
19           "]
20     subprocess.run(cmd, check=True)
```

Listing 5.4: Jupyter/Colab notebook (notebooks/compression.ipynb) — papermill parameters with the all option and @param controls.

Artifacts out. The Stage writes to `results/compression/<Model>/<variant>/`:

- `<Model>-<variant>.gguf` — the quantized weights (compute branch), or a provenance pointer to the Hub artifact (retrieve branch);
- `<variant>_status.yaml` — idempotency/tracking record (skip-if-done);
- `compression_run.log` — integrity check of the exported artifact.

The **Strategy** pattern (`NativeGGUFStrategy`) resolves each variant into a runnable handle downstream, so Phase 2 consumes the retrieved and computed variants without distinction. Crucially, if it is not needed to perform any local compression, for example, because all needed compressed variants already exist on Hugging Face Hub, it is possible for the next Phase 2 to directly use the model’s coordinates for the hub imports.

5.4 Phase 2 Implementation

Phase 2 constitutes the analytical core of ChaosAthena, comprising three independent Stages. Each adheres to a strict *read-state* $\rightarrow$ *analyze* $\rightarrow$ *write-artifact* contract—in the same logic as the config-in, artifact-out contract. The Stages evaluate specific model states—generated or retrieved externally from Hugging Face in Phase 1—and archive their outputs to a storage substrate, either locally or on a cloud platform such as Google Drive, used in this dissertation. This decoupled architecture ensures distributed execution without inter-process memory sharing, avoiding, for example, Out-of-Memory (OOM) errors when utilizing GPUs. The functional stages are Performance Evaluation, Red Team AI Evaluation, and Deep Learning First-Principles.

5.4.1 Performance Evaluation Stage

Taking into account what was mentioned on subsection 5.2.2, on Phase 2, the Performance Evaluation Stage, for any evaluation, in terms of workload sets the `num_fewshot` parameter to `null` to ground the model’s context, instructing `lm-evaluation-harness` to automatically use the default few-shot count explicitly calibrated for each benchmark. The workload is dynamically dispatched to an isolated vLLM subprocess (`src/modules/lm_harness_eval/harness_runner.py`).

5.4.2 Red Team AI Evaluation Stage

The Red Team AI Evaluation stage evaluates adversarial resilience under a strict black-box regime. Having established the conceptual mechanics of the `garak` framework in the subsection 5.2.1, only additional details of the implementation and integration within ChaosAthena are described. The Red Team AI Evaluation stage is architected to host both offensive generation (`garak`) and defensive classification (`guardrail`) modules, but also takes a safety and defensive role, writing detailed vulnerability artifacts directly to the storage substrate to ensure strict stage decoupling.

5.4.2.1 Red-Team Assessment

By utilizing the `garak` module (`src/modules/garak_eval`), the offensive assessment is orchestrated by the `src/pipeline/red_ai_stage.py` façade. To enforce the strict black-box isolation required by the framework’s observational boundaries, the Hydra Orchestrator pipeline manages the dynamic lifecycle of a local vLLM server. It boots a temporary OpenAI-compatible endpoint, passes the target network address to the `garak` subprocess (via the `OPENAI_API_BASE` variable), and sequentially dispatches the configured probe families against the model state. By forcefully tearing down the server immediately upon completion, the pipeline ensures that no residual memory fragments pollute subsequent evaluations within the orchestrator.

Furthermore, the module parses the adversarial results and writes the corresponding artifacts to the storage substrate: the raw execution logs (`<variant_id>_garak.log`), the specific vulnerability triggers (`<variant_id>.hitlog`), the idempotency tracking file (`<variant_id>_status.jsonl`), and an interactive summary report (`<variant_id>.html`). These structured artifacts serve a dual purpose. First, saving security results as structured data files enables the next phase of the framework to read them directly from storage, keeping the Stages completely independent of one another. Second, the generated HTML document serves as the primary visual artifact for researchers to manually audit the model’s adversarial resilience, presenting the discovered vulnerabilities—sorted by probe performance and mapped to established security taxonomies such as the OWASP Top 10 for Large Language Models[26].

Finally, the Stage evaluates model resilience across five core metrics aggregated at the overall, taxonomy, probe, and sub-probe levels:

- **$\bar{z}$ (Z-score):** A calibrated relative score derived by comparing the target model’s pass rate to a reference “bag of models.” A positive $\bar{z}$ indicates above-average resilience, while a negative score indicates below-average performance.
- **$\overline{\text{pass}}$ (Pass Rate):** The inverse of the vulnerability rate, calculated as $\overline{\text{pass}} = 1 - \text{ASR}$, where ASR represents the Attack Success Rate.
- **DEFCON:** A severity grading scale ranging from 1 (most critical) to 5. The metric, `RT_Overall_DEFCON`, is computed from individual probe and sub-probe severities. It represents the final safety grade of the model by taking the minimum score across all probes, ensuring that isolated critical vulnerabilities are not masked by passing marks elsewhere.
- **Tier:** A categorical rating indicating the architectural or security classification of a specific probe and sub-probe.
- **n_hits:** The absolute count of failing generations in which a detector successfully identifies an adversarial exploit.

5.4.2.2 GuardRail Safety Classification

Integrated as a module in `src/modules/guardrail_eval` and controlled by the `src/pipeline/red_ai_stage.py` façade, the safety-classification component evaluates the defensive moderation of generated outputs. Although its fully automated implementation is itself a contribution to the dissertation, it is currently a functional, tested PoC that is not used in the Experiments due to identified limitations (7.2).

Operating as a complement to the Red Team AI stage, this module is based on the GuardReasoner methodology to perform reasoning-based safety evaluations [75, 74]. Unlike traditional guardrails that only output binary safe/unsafe labels based on fixed categories, this module forces any evaluated guard model to generate an intermediate reasoning trace before delivering its verdict, allowing better explainability and improved results [75]. The module evaluates inputs and outputs in three explicit tasks: prompt harm, response harm, and refusal detection.

Finally, upon execution, the module exports evaluation artifacts to the storage substrate, including quantitative performance metrics such as F1 (`guardrail_metrics.json`); explicit intermediate logical reasoning chains that provide decision explainability (`<variant>_traces.json`); and isolated classification-prompt errors to facilitate manual auditing (`misclassifications.jsonl`).

Table 5.9: GuardRail safety classification: example configuration and datasets.

Name	Example of Values	Description
guard_model	GuardReasoner-8B	The reasoning-based LLM safeguard utilized for classification.
prompt_harm	[ToxicChat, WildGuardTest, HarmBench]	Benchmark datasets used to evaluate whether user-input requests are malicious.
response_harm	[SafeRLHF, BeaverTails]	Benchmark datasets assessing whether generated outputs violate safety policies.
refusal_detection	[XSTestResponseRefusal]	Benchmark datasets determining whether the model safely refused an adversarial prompt.
metrics	[F1, recall, precision]	Quantitative metrics calculated and exported to guardrail_metrics.json.
traces	true	Directive to persist intermediate reasoning paths to <variant>-traces.json.

Note: LLM = Large Language Model.

5.4.3 Deep Learning First-Principles Stage

Based in what was already explained in Chapter 4 and earlier in this chapter, the Deep Learning First-Principles Stage (`src/pipeline/htsr_setol_stage.py`) controls the analysis of internal mathematical structures in a data-free regime, enabling the white-box regime. It reads weight matrices, calculates spectral properties, and writes structural artifacts to storage. The module archives the extracted HTSR and SETOL metrics to the storage substrate as a comprehensive suite of persistent artifacts. These structured outputs encompass the granular layer-wise metrics (`<model>_<variant>_esd_metrics.csv`), architectural layer descriptions (`<model>_<variant>_description.csv`), aggregated statistical summaries (`<model>_<variant>_summary.json`), and a dedicated directory with all spectral-shape visualizations (`layers_plots_images/`), ensuring no data is lost for future multi-perspective

assessment and analysis. Next, in table 5.10, all the metrics and properties of any analyzed model state are described.

Table 5.10: HTSR and SETOL weight-matrix metrics extracted by the Stage.

Metric Name	Category	Description
alpha / raw_alpha	Shape	Primary HTSR power-law exponent (α_{Hill}) fit on the ESD tail. Lower values ($\alpha \approx 2$) indicate ideal learning; $\alpha < 2$ indicates overfitting or correlation traps.
alpha_weighted	Shape	The AlphaHat ($\hat{\alpha}$) metric, defined as $\hat{\alpha} = \alpha \times \log_{10}(\lambda_{\text{max}})$, weighting the shape by the log spectral norm to adjust for scale anomalies.
entropy	Shape	Spectral Entropy of the weight matrix, measuring the information density and structural complexity retained within the eigenspectrum.
stable_rank	Shape	Ratio of the squared Frobenius norm to the squared spectral norm: $\ W\ _F^2 / \ W\ _2^2$. Effectively quantifies the numerical rank.
rand_distance	Shape	Distance between the empirical ESD and the ESD of a randomized-spectrum equivalent matrix, quantifying structural deviations from pure noise.
norm / log_norm	Scale	The Frobenius norm $\ W\ _F = \sqrt{\text{Tr}(W^T W)}$ and its base-10 logarithm $\log_{10}\ W\ _F$.
spectral_norm	Scale	The spectral norm of the weight matrix, $\ W\ _2 = \sigma_{\text{max}}$ (the largest singular value, equivalent to $\sqrt{\lambda_{\text{max}}}$ of the unnormalized matrix $W^T W$).
lambda_max	Scale	The maximum eigenvalue (λ_{max}) in the ESD, defining the hard upper bound of the empirical spectrum.
log_spectral_norm	Scale	The base-10 logarithm of the maximum eigenvalue ($\log_{10} \lambda_{\text{max}}$).
log_alpha_norm	Scale	Logarithmic-scale mapping combining the alpha shape exponent and the matrix scale norm.
N / M	System	Dimensions of the layer weight matrix W ($N \times M$), corresponding to the number of input and output features, oriented such that $N \geq M$.

Continued on next page

Table 5.10 – continued from previous page

Metric Name	Category	Description
Q	RMT/Fit	Aspect ratio of the weight matrix ($Q = N/M$). A fundamental parameter dictating the shape and structural bounds of the theoretical Marchenko–Pastur bulk.
D	Diagnostic	The Kolmogorov–Smirnov distance (D_{KS}), evaluating the goodness-of-fit of the empirical ESD tail against a theoretical Power-Law distribution.
x_{min} / x_{max}	RMT/Fit	The lower bound (λ_0) and upper bound of the eigenvalues used to isolate and fit the power-law tail of the ESD.
<code>matrix_rank</code>	RMT/Fit	The linear-algebraic (hard) rank of the layer matrix.
<code>rank_loss / weak_rank_loss</code>	RMT/Fit	Indicators quantifying the loss of effective linear rank (rank-collapse).
<code>mp_softrank</code>	RMT/Fit	Marchenko–Pastur (MP) soft rank, describing how well the core bulk of the ESD fits random-matrix theory descriptions.
<code>num_spikes / num_pl_spikes</code>	RMT/Fit	The number of dominant eigenvalue spikes pulling out beyond the random MP bulk into the heavy-tailed self-regularized region.
<code>num_evals</code>	RMT/Fit	The total number of eigenvalues evaluated for the layer matrix spectrum.
<code>sigma / sigma_mp</code>	RMT/Fit	The empirical variance and the MP theoretical bulk variance of the underlying matrix elements.
<code>bulk_max / bulk_min</code>	RMT/Fit	The maximum (λ^+) and minimum (λ^-) eigenvalues bounding the theoretical random MP bulk.
<code>W_scale</code>	RMT/Fit	The scale factor of the weight matrix extracted during the spectral normalization pass.
<code>rand_W_scale,</code> <code>rand_bulk_max,</code> <code>rand_bulk_min,</code> <code>rand_distance,</code> <code>rand_mp_softrank,</code> <code>rand_num_spikes,</code> <code>rand_sigma_mp,</code> <code>max_rand_eval</code>	Randomized	Equivalent structural metrics computed after randomly shuffling the weight-matrix elements to establish an empirical noise baseline.

Continued on next page

Table 5.10 – continued from previous page

Metric Name	Category	Description
rf / num_fingers	Diagnostic	Configuration parameters reflecting the spectral-fitting techniques (e.g., the Fix-Finger method) used to adjust the lower eigenvalue tail threshold.
num_traps	Diagnostic	The number of Correlation Traps detected (anomalously large eigenvalues arising from single large elements rather than true distributed learned correlations).
ww_maxdist / ww_softrank	Diagnostic	Internal comparative distance and soft-rank diagnostics specific to the parsing tool’s optimization algorithms.
has_esd	System	Boolean flag indicating successful extraction and structural computation of the Empirical Spectral Density.
layer_type / longname	System	Categorization of the structural layer. Separates Attention matrices (q_proj, k_proj, v_proj, o_proj) from FFN (MLP) matrices (gate_proj, up_proj, down_proj).
status / warning	System	Status flags identifying problematic layers, warning if a matrix is severely undertrained ($\alpha > 6$) or overtrained ($\alpha < 2$).

Note: ESD = Empirical Spectral Density; FFN = Feed-Forward Network; MLP = Multi-Layer Perceptron; HTSR = Heavy-Tailed Self-Regularization; MP = Marchenko–Pastur; PL = Power-Law; RMT = Random Matrix Theory; SETOL = Semi-Empirical Theory of Learning; SVD = Singular Value Decomposition.

5.4.4 Example of configuration in and Structural Artifacts out

The Deep Learning First-Principles Stage is the clearest illustration of the framework’s “**configuration in, artifacts out**” contract, because it is inference-free: it only reads the model’s weights matrix. Hydra composes the run from three layered groups — the WeightWatcher parameters, the target model group (which enumerates the pre-compressed GGUF variants and the dequantize mode realizing dq , (Eq. 4.5), and the output policy. The **Façade** `htsr_setol_stage.run(cfg)` hides the wiring; the **Strategy** pattern selects the load format per variant.

```

1 defaults:
2   - _self_
3   - model: Llama-3.1-8B-Instruct
4 htsr:
5   plot: true

```

```

6   savefig: true
7   save: true
8   params:
9     fit: 'PL'
10    randomize: true
11    min_evals: 50
12    mp_fit: true
13    fix_fingers: 'clip_xmax'
14 output:
15   base_dir: results/htsr_setol
16   delete_model_after_run: true

```

Listing 5.5: Root experiment config (configs/htsr_setol/config.yaml).

```

1 model:
2   name: Llama-3.1-8B-Instruct
3   full_size:
4     hf_id: meta-llama/Meta-Llama-3.1-8B-Instruct
5     tokenizer_hf_id: meta-llama/Meta-Llama-3.1-8B-Instruct
6   compressed_versions:
7     - variant: Q4_K_M
8       hf_id: unsloth/Meta-Llama-3.1-8B-Instruct-GGUF
9       filename: Meta-Llama-3.1-8B-Instruct-Q4_K_M.gguf
10      quantization_format: gguf
11      mode: dequantize          # applies dq in-memory before SVD
12     - variant: Q3_K_M
13       hf_id: unsloth/Meta-Llama-3.1-8B-Instruct-GGUF
14       filename: Meta-Llama-3.1-8B-Instruct-Q3_K_M.gguf
15       quantization_format: gguf
16       mode: dequantize
17     - variant: Q2_K_L
18       hf_id: unsloth/Meta-Llama-3.1-8B-Instruct-GGUF
19       filename: Meta-Llama-3.1-8B-Instruct-Q2_K_L.gguf
20       quantization_format: gguf
21       mode: dequantize

```

Listing 5.6: Model group with dequantize map

(configs/htsr_setol/model/Llama-3.1-8B-Instruct.yaml).

Execution. CLI and notebook are code-path identical:

```

1 # CLI (single model, all its variants):
2 python src/run_pipeline.py htsr_setol model=Llama-3.1-8B-Instruct
3 # CLI (multi-model sweep):
4 python src/run_pipeline.py --multirun htsr_setol \
5     model=Llama-3.1-8B-Instruct,Qwen3-0.6B,Qwen3-8B

```

```

6 # Notebook
7 notebooks/htsr_setol.ipynb -> set configs_to_run=[...] (configurable papermill
  parameter cell for CLI call)

```

Artifacts out (per variant, under `results/htsr_setol/<Model>/<variant>/`):

- `<Model>_<variant>_esd_metrics.csv` — per-layer spectral metrics (Table 5.10: α , $\hat{\alpha}$, entropy, stable_rank, rand_distance, D_{KS} , ...);
- `<Model>_<variant>_description.csv` — layer inventory (type, longname, $N \times M$, Q);
- `<Model>_<variant>_summary.json` — aggregated statistics;
- `layers_plots_images/` — per-layer ESD plots;
- `htsr_status.yaml` — idempotency record.

Hydra additionally emits the auditable run record under `hydra_logs/`, satisfying NFR5 (4.3.6).

5.5 Phase 3 Implementation — Multi-Perspective Dataset Creator

Phase 3 serves as the critical convergence layer via the Multi-Perspective Dataset Creator Stage (`src/pipeline/datasets_creator_stage.py`). It ingests heterogeneous Phase 2 artifacts—performance JSON logs, adversarial JSONL reports, structural CSV metrics, among others—and aggregates them into datasets. By mapping, for example, model’s deep learning first principles metrics to empirical behavioral outcomes, it constructs the data bridge required for Phase 4 analytics.

5.5.1 Multi-Perspective Dataset Creator Stage

To ensure scientific validity, the Stage’s operational logic, whenever possible, aligns with two complementary international standards: ISO/IEC 25012 for data-quality characteristics and DAMA-DMBOK for lifecycle management practices. Specifically, ISO/IEC 25012 ensures Accuracy (e.g., preserving exact Heavy-Tailed Self-Regularization floating-point exponents during joining without precision loss), Completeness, and Consistency (e.g., enforcing uniform variant-ID mapping across all joined data sources). Furthermore, DAMA-DMBOK practices govern automated data integration and maintenance, for example, maintaining an unbroken metadata lineage from the raw execution logs to the final *.parquet* and *.csv datasets*.

The dataset creator constructs a schema that consolidates metrics in all Stages, performs feature engineering by computing deltas (Δ) between the baseline 16-bit models and their compressed variants across all measurable features, creating degradation deltas. Table 5.11 illustrates the aggregated schema, showing examples of metrics and Δ (delta) calculations relative to the 16-bit baseline.

Table 5.11: Aggregated multi-perspective dataset schema: base vs. compressed-variant deltas.

Model Name	Llama-3.1-8B			Qwen-2.5-7B		Mixtral-8x7B	
Type	Dense			Dense		MoE	
Precision / Compression	16-bit (Base)	4-bit GGUF Q4_K_M	3-bit GGUF Q3_K_M	16-bit (Base)	4-bit GGUF Q4_K_M	16-bit (Base)	4-bit GGUF Q4_K_M
Perf: MMLU Δ	0.00	−1.07	−5.12	0.00	−0.50	0.00	−0.90
RedTeam: OWASP LLM01 Δ	0.00	+0.31	+0.85	0.00	+0.10	0.00	+0.40
DLFP: WW α Mean Δ	0.00	+0.25	+1.10	0.00	+0.08	0.00	+0.30

Note: DLFP = Deep Learning First-Principles; MMLU = Massive Multitask Language Understanding; MoE = Mixture of Experts; OWASP = Open Worldwide Application Security Project; WW = WeightWatcher.

5.6 Phase 4 Implementation — Analytics and Results

Phase 4 constitutes the definitive layer of the ChaosAthena framework. Executed after the data aggregation and feature engineering of Phase 3, it consumes the multidimensional datasets to extract statistically grounded conclusions. This phase, mainly implemented by the Veritas Analytics Stage (`src/pipeline/veritas_analytics_stage.py`) and its `veritas_analytics` module (`src/modules/veritas_analytics/`), applies statistical analysis and Exploratory Data Analysis (EDA) to evaluate feature connections and comparative metrics, Phase 2 evaluations, and the corresponding degradation deltas created in Phase 3.

Phase 4 also foresees Principal Component Analysis (PCA) and Instance Space Analysis (ISA), the latter producing topological algorithm footprints. Neither is implemented: prioritized as Could-have (FR16 in Table 4.1), they serve as examples of the possible extensibility of the framework baseline and are deferred to future work — particularly ISA, to support future large-scale analyses of models.

Table 5.12: Veritas Analytics Stage: example of configuration parameters and descriptions.

Name	Example of Values	Description
baseline	full_size	The reference FP16 variant utilized for calculating all Δ degradation metrics.
correlations	[pearson, spearman]	Statistical methods executed to identify linear and monotonic non-linear relationships.

Note: FP16 = 16-bit Floating Point; ISA = Instance Space Analysis; PCA = Principal Component Analysis.

Finally, Veritas Analytics Stage under Phase 4, concludes the framework execution, finalizing the data by formatting the derived statistical conclusions for the dissertation, including required plots, graphs, and Markdown tables.

5.7 Final User Perspective

Based on the detailed descriptions of each Phase, this section serves the purpose to consolidate the user’s view of ChaosAthena. Every Phase or Stage shown above is reached through one uniform surface: a set of layered Hydra YAML files selected and, if needed, overridden at run time, dispatched by a single entry point with exact CLI $\leftrightarrow$ notebook parity (NFR11, 4.3.6). The user never edits any core component of the Python code to run an experiment; only confirmation is needed.

Running everything, or one selected stage. A full experiment over every configured model and all four phases is one sweep; a single stage of one Phase, or model is one override. The @param-driven design means the unit of work is chosen at the call site on Jupyter Notebooks:

```

1 # 1) Every phase, every model group (end-to-end)
2 python src/run_pipeline.py all model=all # running all phases)
3
4 # 2) One stage of a Phase across ALL configured models (no manual list)
5 python src/run_pipeline.py performance_eval model=all
6
7 # 3) One phase across a chosen stage
8 python src/run_pipeline.py --multirun performance_eval \
9     model=Qwen3-8B,Meta-Llama-3.1-8B-Instruct
10
11 # 4) One stage, one model, one override (e.g. change the K-quant profile on the fly
12     )
13 python src/run_pipeline.py compression model=Qwen3-0.6B profile=Q2_K_L
14
15 # 5) Add a new model to be evaluated: drop a YAML in the model group dir
16 # (configs/<stage>/model/MyModel.yaml) and reference it (or #run it, by being
17     included in model=all)
18 python src/run_pipeline.py htSr_set01 model=MyModel

```

Adding models and stages. New model groups are read from a given accessible directory, so a YAML added “on the spot” is immediately selectable (case 4 above); parameters can also be injected “on the fly” from the CLI without editing any file, thanks to Hydra composition (NFR4, 4.3.6). Adding a whole new stage follows the fixed four-file shape (config YAML, CLI entry, run(cfg) façade, notebook) plus one catalog registration — the same additive change described in 5.1.6.

One unified contract. Whatever the unit of work, the behavior is identical: configuration is composed, a phase reads its declared inputs (models for Phase 1–2, artifact *paths* for Phase 3–4), work is dispatched to isolated subprocesses to bound memory, results are written to the storage substrate, and an auditable run record is emitted. Ultimately, this “configuration in, artifacts out” uniformity is what lets the user treat ChaosAthena as a single instrument with interchangeable parts, and is precisely this property that makes the near-drop-in migration to a future microservice deployment possible ([4.3.4.2](#)).

Chapter 6

Experiments and Results

The purpose of this chapter is twofold. First, based on the Problem Setting and Proposed Solution established in Chapter 4 and the ChaosAthena framework implementation in Chapter 5, it presents the exact instantiation of all experiments executed, including the selected model cohort, evaluation Stages, hardware resources, and execution parameters. Second, it presents the empirical results. To maintain clarity and readability throughout this chapter, all critical parameter configurations are summarized in dedicated tables.

Furthermore, all experiments are restricted to the implemented main pipeline, stages, and modules described in Chapters 4 and 5, namely: Performance Evaluation Stage, Red Team AI Evaluation Stage, Deep Learning First-Principles Stage, Compression Stage, Multi-Perspective Dataset Creator Stage, and Veritas Analytics Stage. Finally, the empirical evidence gathered is utilized to validate the Research Questions, directly verifying (Section 6.5) the evaluation criteria established in Section 4.3.7.

6.1 Scope and Boundary Conditions

The evaluation is characterized by a set of deliberate design decisions that define the scope of models about which claims are made and keep experimental factors under control. These delimitations are defined as follows:

- **Model Size Class.** The corpus is restricted to models of up to approximately 8 billion parameters. This is the size class in which community post-training quantization is predominantly produced and consumed — on consumer, edge, and Fog-Edge hardware — so the restriction maximizes this fact, enabling a case study design in which every model family is evaluated at all four precision levels[3, 153].
- **Standard Inference Mode.** All models are executed in standard inference mode, with advanced reasoning configurations disabled. This keeps the generation regime constant so that the quantization level remains the only manipulated variable: reasoning traces introduce sampling-dependent, variable-length generation that would confound the cross-variant

comparability of both the benchmark scores and the adversarial probing [154], and it bounds the performance baselines against which the Performance and Red Team AI evaluations are interpreted.

- **Architecture.** Dense transformer architectures constitute the subject of the experiments. Holding the architectural class constant removes architecture as a confounder, which allows the within-family versus between-family variance decompositions and yields statistically meaningful conclusions about the dense transformer type.
- **Compression cohort.** The experimental quantization footprint is the GGUF K-quant family at the 4-bit, 3-bit, and 2-bit levels (Q4_K_M, Q3_K_M, and Q2_K_L). A single format holds the quantization method constant while the three levels span the deployed to low-bit, 2 bits spectrum in which degradation dynamics are expected to emerge, making the precision level the sole compression factor [154]. Importantly, the choice of GGUF over alternative PTQ formats is justified by the comparative analysis of Section 3.2.3 (Table 3.2): it is the only reviewed format that simultaneously covers CPU-only, hybrid, and diverse consumer-GPU hardware — the heterogeneous conditions Edge deployment assumes — alongside, particularly for quantization and local deployment, the format is the most widely published and used on Hugging Face [151, 87], and exposed to already documented GGUF attacks [24].
- **Artifact Provenance.** Quantized variants are the pre-compressed artifacts published on Hugging Face, so the evaluation targets exactly the artifacts users download and deploy, and the corpus remains publicly pinned and reproducible. Local GGUF generation is additionally performed to demonstrate the compression operator end-to-end.

Importantly, the unit of analysis in this dissertation is the deployment-ready model state: quantized GGUF artifacts at or below 8B parameters, whose format and memory footprint make them executable on commodity end devices[22, 153]. The claims made concern the capability, adversarial-resilience, and spectral structural consequences of the compression such deployment requires; on-device runtime characterization (latency, energy) is explicitly out of scope.

As a result, these delimitations define the model corpus of Section 6.2.3 and the boundary conditions under which the analytical claims of Section 6.3 hold. Generalization beyond them, larger scales, reasoning modes, further architectural classes, and additional compression formats, is identified as future work (Section 7.3).

6.2 Experimental Setup

This section materializes Chapters 4 and 5, by providing the concrete instantiation of all the experiments. The setup is presented in the order of the four phases of the ChaosAthena framework, allowing a direct correspondence between each empirical experiment and the architectural design depicted in Figures 4.1 and 4.2. By executing the “configuration in, artifacts out” contract, every quantitative and qualitative result reported in Section 6.3 can be traced directly to a specific YAML

configuration file. This guarantees that Hydra orchestrates and logs all variable parameters, ensuring reproducibility. All critical per-stage parameters and hardware specifications are summarized in dedicated tables, while, if relevant, any other execution descriptors or comprehensive Hydra configurations are deferred to Appendix C.

6.2.1 Experimental overview

As described in Chapter 5, all the experiments are executed independently across the four architectural phases: Phase 1 (compression or importing), Phase 2 (concurrent execution of three core evaluation Stages), Phase 3 (aggregation and standardization), and Phase 4 (analytics and experimental results). Each phase writes immutable artifacts directly to the storage substrate. Table 6.1 specifies the active Stages, their integrated modules, the corresponding observational regime, and the main artifacts produced for the primary evaluation and final results.

Table 6.1: Experimental Matrix Mapping Phases to Stages, Modules, Regimes, and Main Output Artifacts

Phase	Stage	Integrated Module	Regime	Artifact Written	Executed?
P1	compression	GGUF (llama.cpp)	—	/ .gguf, status.yaml	Yes
P2	performance_eval	lm-evaluation-harness	gray-box	results_.json, samples_.jsonl	Yes
P2	red_ai	garak	Black-box	.hitlog, .html, _status.jsonl	Yes
P2	htsr_setol	WeightWatcher	White-box	_esd_metrics.csv, plots	Yes
P3	datasets_creator	pandas	—	.parquet, *.csv	Yes
P4	veritas_analytics	scikit-learn, scipy, statsmodels, pandas, AutoViz	—	EDA, correlation artifacts, tables, and experimental results	Yes

Note: EDA = Exploratory Data Analysis; GGUF = GPT-Generated Unified Format;

6.2.2 Compute infrastructure and hardware backends

Whenever possible, experiments aimed at accelerating the evaluation process are conducted on two GPU-equipped nodes, and, when necessary or depending on the computational workload and availability, on lower-tier GPUs or CPU-only instances, demonstrating the framework’s adaptability and enabling evaluation of GenAI models under different conditions similar to Fog-Edge constraints ($\leq 8B$ parameters). Each experiment is isolated to a single node to prevent resource

contamination and OOM errors. For primary runs, vLLM is used on A100 and H100 nodes to maximize throughput for the Performance and Red Team AI Stages, with orchestration adapted to session-based cloud environments such as Google Colab. The backends in Table 6.2 provide evaluation throughput only; the model states under study are the quantized weight configurations themselves, which are preserved exactly under the dequantized serving path (Equation (4.5)), so the measured capability, safety, and structural properties are attributable to compression, not to the utilized hardware [99, 101].

The vLLM engine mostly drives the GPU experiments, llama.cpp remains the standard engine for CPU-only or hybrid environments, whether for compression or, when needed, to provide essential portability and compatibility for quantized models from the perspective of Fog-Edge nodes. Furthermore, the First-Principles deep learning stage (htsr_setol) uses WeightWatcher in a data-free and inference-independent manner. Consequently, this analysis does not require an active inference engine and is executed on a more limited T4 GPU. Table 6.2 details the exact physical hardware allocations and capabilities mapped across the evaluated nodes.

Table 6.2: Compute Backends Used in Experimentation

Node	System RAM	GPU VRAM	Disk	Inference Engine
H100	176.9 GB	95.6 GB	235.7 GB	vLLM
A100	167.1 GB	80.0 GB	235.7 GB	vLLM, llama.cpp
T4	12.7 GB / 51.0 GB	15.0 GB	235.7 GB	WeightWatcher, vLLM, llama.cpp
CPU-only	12.7 GB / 51.0 GB	—	225.8 GB	llama.cpp

Note: GB = Gigabytes; RAM = Random Access Memory; VRAM = Video Random Access Memory; vLLM = Virtual Large Language Model.

6.2.3 Model cohort under evaluation

The model cohort is restricted to small, dense, parameterized architectures ($\leq 8\text{B}$ parameters), corresponding to the size regime in which quantized models are predominantly deployed on the Edge devices [22, 153]. The seven selected models—four Qwen3 and three Llama-3.x instruction fine-tuned variants, were chosen because these two families jointly satisfy the corpus design criteria: (i) availability of open weights; (ii) the existence of instruction fine-tuned model’s at multiple scales within the $\leq 8\text{B}$ class, allowing a scaling analysis within families and a direct cross-scale comparison; and (iii) the presence of officially released or community-maintained GGUF variants at all three target quantization levels, ensuring that the evaluated artifacts correspond exactly to those downloaded and deployed by the Hugging Face community. All selected models are sourced from Hugging Face, either as full-precision checkpoints or as their quantized counterparts. In general, the constructed cohort mainly covers general-purpose instruction follow-up capabilities and, to a limited extent, agentic capabilities [155].

Each selected model serves as the origin for a defined variant group. A complete group consists of the full-precision baseline and three progressively compressed GGUF artifacts representing distinct bit-width constraints (Q4_K_M, Q3_K_M, and Q2_K_L). Throughout the evaluation, a standardized 6-column metadata schema acts as the definitive identifier for each model variant (including parameters such as *group_id*, *model_id*, *params_billion*, or *compression_name*). All models will be used only in text mode, even if some models allow multimodal capabilities, such as image processing. For simplicity, only core identifier metadata will be used. Table 6.3 details the full-precision baseline models, while Table 6.4 summarizes the compressed variant families derived from them. The bit-widths are identified as 4, 3, and 2 bits, but because the quantization is mixed-precision, in practice these values are approximations [121].

Table 6.3: Full-Precision Baseline Models

model_name	architecture	arch_subtype	params_billion
Qwen3-0.6B	Transformers	Dense	0.6
Qwen3-1.7B	Transformers	Dense	1.7
Qwen3-4B-Instruct-2507	Transformers	Dense	4.0
Qwen3-8B	Transformers	Dense	8.0
Llama-3.2-1B-Instruct	Transformers	Dense	1.0
Llama-3.2-3B-Instruct	Transformers	Dense	3.0
Meta-Llama-3.1-8B-Instruct	Transformers	Dense	8.0

Table 6.4: Compressed-Variant Families Overview

model_id	model_name	compression_name	compression_type
101	Qwen3-0.6B	NA	NA
102	Qwen3-0.6B	GGUF	Q4_K_M
103	Qwen3-0.6B	GGUF	Q3_K_M
104	Qwen3-0.6B	GGUF	Q2_K_L
201	Qwen3-1.7B	NA	NA
202	Qwen3-1.7B	GGUF	Q4_K_M
203	Qwen3-1.7B	GGUF	Q3_K_M
204	Qwen3-1.7B	GGUF	Q2_K_L
301	Qwen3-4B-Instruct-2507	NA	NA
302	Qwen3-4B-Instruct-2507	GGUF	Q4_K_M
303	Qwen3-4B-Instruct-2507	GGUF	Q3_K_M
304	Qwen3-4B-Instruct-2507	GGUF	Q2_K_L
401	Qwen3-8B	NA	NA
402	Qwen3-8B	GGUF	Q4_K_M
403	Qwen3-8B	GGUF	Q3_K_M
404	Qwen3-8B	GGUF	Q2_K_L
501	Llama-3.2-1B-Instruct	NA	NA
502	Llama-3.2-1B-Instruct	GGUF	Q4_K_M
503	Llama-3.2-1B-Instruct	GGUF	Q3_K_M
504	Llama-3.2-1B-Instruct	GGUF	Q2_K_L
601	Llama-3.2-3B-Instruct	NA	NA
602	Llama-3.2-3B-Instruct	GGUF	Q4_K_M
603	Llama-3.2-3B-Instruct	GGUF	Q3_K_M
604	Llama-3.2-3B-Instruct	GGUF	Q2_K_L
701	Meta-Llama-3.1-8B-Instruct	NA	NA
702	Meta-Llama-3.1-8B-Instruct	GGUF	Q4_K_M
703	Meta-Llama-3.1-8B-Instruct	GGUF	Q3_K_M
704	Meta-Llama-3.1-8B-Instruct	GGUF	Q2_K_L

6.2.4 Phase 1 — Compression and Variant Acquisition Setup

Phase 1 allows, if necessary, the use of the compression operator $\mathcal{C}$ and formalizes the variant family $\mathcal{V}(X)$. For each used model, the dual-mode acquisition logic is possible: defined by a retrieval-or-compute rule. For exact reproducibility, the Hugging Face acquisition option is used: The evaluation is conducted on the same publicly accessible, pinned artifacts that are used in real-world deployments[87].

The experiments use three specific `llama.cpp` GGUF profiles: Q4_K_M, Q3_K_M, and Q2_K_L, never using any targeted calibration dataset. The AutoRound quantizer, while fully implemented, is only included in the already-mentioned Proofs-of-Concept.

To achieve higher accuracy at low compression ratios, the selected K-quant formats do not use a uniform bit width throughout the model layers. Instead, they used a mixed-compression approach, assigning different bit widths to different tensor categories based on empirical sensitivity to quantization error [121, 154]. This compression logic is consistently applied across different layer categories and all models in the experiments. Table 6.5 summarizes the general layer-wise bit-allocation strategy employed by the 4-bit, 3-bit, and 2-bit profiles.

Table 6.5: Mixed-Precision K-Quantization Regimes

Quantization Profile	Width	Description
Q4_K_M	4-bit	Mixed-precision 4-bit K-quant regime, where some attention and feed-forward tensors remain at 4-bits, while other selected higher-sensitivity tensors have 6-bits. Normalization, final output, and auxiliary scale tensors remain 32-bit, while embeddings are 6-bit.
Q3_K_M	3-bit	Mixed-precision 3-bit K-quant regime, where attention and feed-forward tensors remain at 3-bits, while other selected higher-sensitivity tensors have 5-bits, or 4-bit for all output attention projection layers. Normalization, the final output layer, and auxiliary scale tensors remain in 32-bit, while embeddings are in 6-bit.
Q2_K_L	2-bit	Mixed-precision 2-bit K-quant regime, where attention and feed-forward tensors remain at 2-bits, while other selected higher-sensitivity tensors have 3-bits. Normalization, the final output layer, and auxiliary scale tensors remain in 32-bit, while embeddings are in 6-bit.

Note: These regimes correspond to non-uniform tensor-level precision assignments, where a base quantization width is complemented by higher-precision storage for layers that are more sensitive to quantization error.

Finally, all the main parameters needed for local compression operations are detailed in Table 6.6, but were not used in the experiments.

6.2.5 Phase 2 — Evaluation and Analysis

6.2.5.1 Performance Evaluation Setup (lm-evaluation-harness)

The Performance Evaluation Stage uses *tiny* benchmarks for specific tasks [65]: identical and highly discriminative subsets that enable a uniform full-coverage evaluation of every full-precision model and all its quantized variants under a single controlled protocol. As described in Chapter 2, the `tinyBenchmarks` suite uses item response theory (IRT) to compress standard datasets into compact and highly discriminative subsets (typically 100 examples per dataset) [65]. IRT models difficulty and ability so that minimal subsets of the Benchmarks datasets produce robust estimates closely aligned with full-benchmark performance [65].

Table 6.6: Execution Parameters for the Compression Stage

Parameter Name	Value Used	Description
quantization_type	gguf	The target quantization format.
profile	Q4_K_M, Q3_K_M, Q2_K_L	The specific mixed-precision K-quant block profiles to compress the full-size models.
temp_fp16_keep	false	Discards intermediate FP16 tensor representations post-compression to optimize disk space usage.
calibration_dataset	NA	Possible calibration or calculation of importance matrices during optional quantization optimization steps.

Note: FP16 = 16-bit Floating Point; GGUF = GPT-Generated Unified Format; NA = Not Applied.

Furthermore, three choices deserve explicit justification. First, the *task portfolio*: the task families of Table 6.7 were selected to span the capability dimensions that previous work reports as differentially sensitive to quantization — knowledge recall, commonsense inference, truthfulness, multi-step mathematical reasoning, code generation, expert reasoning, instruction following, long-context retrieval, and intrinsic safety — so that degradation can be localized by domain rather than observed only in aggregate [154]. All tasks are the standard implementations provided with `lm-evaluation-harness`, keeping the scores directly comparable with the wider literature. Second, the *tiny subsets*: evaluating 28 model states on twelve full benchmark suites is computationally prohibitive under the available compute budget, whereas the IRT-calibrated `tinyBenchmarks` subsets estimate full-benchmark performance to within approximately two percentage points using about one hundred items per scenario — for MMLU, a reduction of evaluation cost by roughly a factor of 140 [65]. Because every model state is evaluated on the identical subset, this residual estimation error is common to all states and therefore largely cancels in the compressed-minus-FP16 deltas on which the analysis operates; it is nonetheless acknowledged as a limitation on absolute scores (Section 7.2). Third, the *type of task*: with the exception of generative tasks (GSM8K, HumanEval+, IFEval, and needle-in-a-haystack retrievals) [59], all benchmarks are multiple-choice classification tasks scored by length-normalized log-likelihood, as defined in Chapter 2.

Finally, the zero and few-shot benchmarks are orchestrated via `lm-evaluation-harness` and sent to an isolated vLLM subprocess [39, 156]. To have a comprehensive view of model degradation, tasks are clustered into distinct task families. The metric `PERF_mean`, is the unweighted mean across all core accuracy-based tasks. With the exception of `wikitext` [157], which measures the quality of intrinsic language modeling via perplexity (lower is better), all aggregated metrics are evaluated based on accuracy or exact match (higher is better) [62].

Table 6.7 specifies the tasks performed and their family groups. A more comprehensive `lm-evaluation-harness`, vLLM, and sampling parameters used for each run are reported in Appendix B (Table B.1).

Table 6.7: Experimental Execution Parameters, Tasks, and Families for Performance Evaluation

Name	Value / Family	Description
Executed Tasks		
tinyMMLU	fam_knowledge	IRT-compressed subset of MMLU evaluating broad domain knowledge across 57 academic subjects.
tinyArc	fam_knowledge	IRT-compressed subset evaluating complex scientific reasoning and reading comprehension.
tinyHellaswag	fam_-commonsense	IRT-compressed subset testing commonsense reasoning by predicting the logical ending of scenarios.
tinyWinogrande	fam_-commonsense	IRT-compressed subset evaluating coreference resolution and adversarial commonsense.
tinyTruthfulQA	fam_-truthfulness	IRT-compressed QA subset evaluating the factual accuracy and truthfulness of generated responses.
tinyGSM8k	fam_math	IRT-compressed subset of grade-school math word problems for testing multi-step mathematical reasoning.
humaneval_plus	fam_code	Evaluates the functional correctness of synthesized Python programs generated from docstrings.
leaderboard_gpqa_diamond	fam_reasoning	High-difficulty QA dataset requiring expert-level reasoning in physics, biology, and chemistry [58].
leaderboard_ifeval	fam_-instruction	Instruction-Following Evaluation testing strict adherence to verifiable, objective formatting constraints. Also, a simple way to infer minimal agentic capabilities
niah_single_1/2/3	fam_long_-context	Needle-In-A-Haystack tasks evaluating exact information retrieval capabilities across expanded context windows.

Continued on next page...

Table 6.7 – Continued from previous page

Name	Value / Family	Description
toxigen	fam_safety	Assesses the language model’s propensity to generate toxic, hateful, or harmful content. Used as an additional test for model safety [60].
wikitext	fam_lang_modeling	Measures intrinsic language modeling quality via perplexity on Wikipedia articles.
Aggregate Metrics		
PERF_mean	Aggregate	The unweighted average of all core accuracy-based tasks (excluding perplexity) to summarize overall capability performance.

6.2.5.2 Red Team AI Evaluation Setup

In a Black-box regime, the Red Team AI Evaluation Stage assesses model vulnerabilities by executing adversarial attacks against all variants of the target GenAI model using the `garak` module. To prevent memory errors, the Stage enforces a deterministic endpoint lifecycle. It initializes a temporary OpenAI-compatible vLLM endpoint bound to `http://127.0.0.1:8000/v1`. To maximize throughput, the vLLM server is configured with GPU memory utilization set to 0.95, a maximum model context length of 4096 tokens, and chunked prefilling enabled. Upon completion of the asynchronous probing experiments, the server process is terminated immediately to free the GPU’s VRAM. All configuration parameters for vLLM and `garak` in the Red Team AI Stage — the ephemeral endpoint’s server settings and the `garak` execution options — are reported in Appendix B (Table B.2).

Moreover, the evaluation relies on the `garak` framework, which dispatches prompts asynchronously using a high concurrency limit (`parallel_attempts = 500`). To capture potential non-deterministic failure modes, the framework requests exactly two generations per prompt, even though the model’s temperature is locked at 0.0 for experimental control. To compute the Z-scores, `garak` relies on an updated calibration "bag of models". This calibration set provides a statistical baseline of state-of-the-art vulnerability resilience, ensuring that raw pass rates are contextualized against industry standards. The models comprising this bag are selected based on recency, parameter-count diversity (spanning 1B to 100B+), providing diversity, and the availability of open weights. Table A.1 on Appendix A, details the calibration bag, the 2026 Winter version, utilized by the `garak` framework to standardize the results in every experiment [31].

Vulnerabilities are mapped using the OWASP taxonomy configuration. To align the analysis with the current OWASP Top 10 for LLM Applications 2025, legacy 2023 tags are automatically remapped whenever a direct 2025 equivalent exists. In all experiments conducted, the main security focus is **LLM01:2025 (Prompt Injection)**, which occurs when crafted inputs alter the model’s behavior or output in unintended ways, including via direct user prompts or indirect inputs such as retrieved documents, web pages, or files. Furthermore, some sub-probes map to **LLM03:2025(Supply Chain)**, which covers risks affecting the integrity of models, training data, dependencies, and deployment components throughout the LLM lifecycle [26]. Additionally, **LLM05:2025 (Improper Output Handling)** maps to used sub-probes, which refer to cases where LLM-generated output is passed to downstream systems without proper validation, sanitization, or encoding, potentially enabling attacks such as XSS, SSRF, privilege escalation, or remote code execution [26].

As an exception, existing probes that retain a mapping to **LLM04:2023 (Model Denial of Service)** because this category existed in the OWASP list of 2023 but not as the same item in the 2025 version [26]. In the 2023 taxonomy, LLM04 describes attacks that force the model to consume excessive computational resources, degrade service quality, or increase operational cost, for example, through oversized inputs, repeated long-context requests, or other resource-intensive interactions. The concentration on LLM01:2025 is a deliberate choice: prompt injection is ranked first in the OWASP Top 10 for LLM Applications [26], it is the category most completely instrumented by garak’s probe library, **and it is fully exercisable in the black-box regime under which compressed community artifacts are realistically attacked**. Categories that require training-time access or infrastructure control (for example, training-data poisoning) are out of scope by construction of the observational regimes (Section 4.2.4) and they are referenced only through the taxonomy mapping.

Furthermore, for total experiment transparency, the set of probes executed and its OWASP-2025 mapping are enumerated in Table B.3 in Appendix B, specifying the exact subset of LLM01 probes (prompt injection). For every configuration, the generator utilizes the OpenAI-compatible vLLM endpoint. Due to garak’s modular nature, each probe invokes a specific detector, or a group of detectors, to evaluate the attack’s success. In addition to the taxonomy described in OWASP 10, all probes, generators, and detectors, including those that traverse static detectors and those that use classification or generative models to modify attack prompts, are fully described in the garak documentation [31].

Finally, all configuration parameters for vLLM and garak in the Red Team AI Stage — the ephemeral endpoint’s server settings and the garak execution options — are reported in Appendix B (Table B.2). The endpoint runs at GPU memory utilization 0.95 with a 4096-token context and chunked prefill; garak dispatches two greedy generations per prompt at concurrency 500 under the OWASP taxonomy.

6.2.5.3 Deep Learning First-Principles Setup (WeightWatcher)

The Deep Learning First-Principles Stage performs data-free spectral diagnostics by fitting Power-Law (PL) distributions to the Empirical Spectral Density (ESD) of each layer’s weight matrix. To accurately evaluate

compressed GGUF models with the WeightWatcher module and tool, quantized states are dequantized entirely in memory using the dequantization map dq of Equation (4.5), implemented in practice by the Hugging Face `transformer` library. This allows WeightWatcher to be used as a general model tool [41], and ensures that all quantization-induced structural degradation is preserved and measured.

Following the extraction of the raw layer-wise metrics, the Stage extracted data, in Phase 3, will be aggregated across four distinct structural perspectives: global (averaging across all model layers), architectural category (isolating the `@att` attention blocks from the `@mlp` feed-forward blocks), functional sub-layer (evaluating specific projection matrices individually), and network depth (binning layers sequentially into 3-bin, 5-bin, and first/last (the first two/last layers and first/last layers of any model) configurations). Table 6.8 describes the exact configuration parameters used in the WeightWatcher module in all the experiments. The complete dictionary of metrics extracted by this configuration was previously established in Table 5.10 and is derived from HTSR and SETOL [29, 30]; therefore, only the execution configuration is reported here.

Table 6.8: Experimental Execution Parameters for the WeightWatcher / HTSR-SETOL Stage

Parameter Name	Value Used	Description
<code>fit</code>	<code>'PL'</code>	Enforces Power-Law (PL) tail fitting on the Empirical Spectral Density (ESD) for alpha (α) and Heavy-Tailed Self-Regularization (HTSR) structural tracking.
<code>min_evals</code>	50	Establishes the safe evaluation threshold; do not evaluate PL fits on tiny or truncated spectra. A minimum of 50 eigenvalues is required to maintain empirical validity.
<code>randomize</code>	<code>true</code>	Enables randomized baseline execution to systematically test whether the observed layer correlations survive against scrambled, random matrix baselines.
<code>mp_fit</code>	<code>true</code>	Fits a Marchenko-Pastur (MP) distribution to separate the uninformative random bulk from the heavy tail, helping to isolate and flag random-like layers.
<code>fix_fingers</code>	<code>'clip_xmax'</code>	Stabilizes spectral exponent (α) calculations by dynamically clipping the maximum eigenvalue ($x_{\max}$) when anomalous high-end tail leaks cause optimization instabilities.
<code>plot, savefig, save</code>	<code>true</code>	Orchestrates the automation Stage rules to continuously generate, persist, and store visual diagnostics and numeric performance evaluation artifacts.

Note: ESD = Empirical Spectral Density; HTSR = Heavy-Tailed Self-Regularization; MP = Marchenko-Pastur; PL = Power-Law.

6.2.6 Phase 3 — Multi-Perspective Dataset Setup

Phase 3 implements the necessary preparation, aggregation, and standardization of all generated artifacts from the isolated Phase-2 Stages into a multi-perspective dataset aligned with variant-ID, as well as individual datasets for each Phase-2 Stage. The schema relies on a group-structured data model: each evaluated group consists of the full-precision baseline, paired with the compressed variants (i.e., full-size + Q4_K_M, Q3_K_M, Q2_K_L).

For the model's compression degradation analysis, the numerical value corresponds to a specific measured value, and one of two means, or one Δ applied to any given dataset metric. As represented in the created datasets, **mean_1** denotes the per-model metric at the global scope, while **mean_2** is the group-pooled mean calculated over **mean_1** for HTSR, Red Team AI evaluation, or task families for Performance). The **mean_3** if necessary, is the group-pooled means calculated over **mean_1** and **mean_2**, summarizing the average state across each variant group, and **mean_delta1** is the group mean-delta, the average shift compression caused to a group. Crucially, **delta_1** measure the exact deviation of each quantized variant against its own full-size baseline at the respective **mean_1** and **mean_2** ranges.

Furthermore, the dataset enforces a strict distinction to preserve data integrity: NA ("Not Applied") denotes metrics that are structurally incompatible with a given model or layer, whereas a blank cell indicates a valid metric that was simply "not measured" during Phase 2 Stages execution.

Based on the aforementioned identifiers in the table 6.3, Phase 3 uses the same logic, by applying the `group_id` and `model_id` scheme: for example, for a given model, `group_id/` is 1; therefore, for the full-size baseline `model_id` is 101, 102 for Q4_K_M, 103 for Q3_K_M, and 104 for Q2_K_L. All transformations, whenever possible, apply the ISO/IEC 25012 data quality standards [82], ensuring the properties of Accuracy, Completeness, and Consistency (e.g., exact floating-point α exponents are preserved without truncation during tabular joins). Finally, by applying the DAMA-DMBOK data governance principles [83], it's possible to maintain a lineage tracing from the final transformations down to the final raw `.parquet` and `.csv` artifacts. Table 6.9 presents the structure of all Phase-3 artifacts generated by the Multi-Perspective Dataset Creator Stage. These artifacts simultaneously correspond to a set of configurable parameters (e.g., choose to only create the DLFP_view dataset). The elements described above, when necessary may be optionally specified for generation, depending on the user's needs and for the Phase-2 Stage-generated artifacts in the experiments.

Table 6.9: Phase-3 Artifacts Produced and Main Datasets Column Definitions

Artifact name	Description
Headline	High-level dataset aggregated summary of core metrics (wide format).
Headline_T	Transposed version of the Headline dataset for a more user-friendly view.
Complete_- aggregation_all_- dataset	Dataset containing all measurements in long format.
DLFP_view	Domain-specific view isolating HTSR and SETOL metrics.
Performance_view	Domain-specific view isolating lm-evaluation-harness results.
RedTeam_view	Domain-specific view isolating garak vulnerability and safety metrics.
Headline Columns Legend: X (Mean 1/2) • s_X (std) • Δ_X (Delta 1/2) • $\bar{\Delta}_X$ (Group mean-delta) • $\bar{M}_X$ (Mean 3/4 pooled)	

Note: DLFP = Deep Learning First-Principles; HTSR = Heavy-Tailed Self-Regularization; SETOL =Semi-Empirical Theory of Learning.

6.2.7 Phase 4 — Analytics Setup

Phase 4 runs the Veritas Analytics stage to perform exploratory data analysis (EDA) and correlation analysis on the aggregated Phase-3 dataset. The primary analytical objective is to statistically relate the data-free structural metrics extracted by the Deep Learning First-Principles stage in Phase 2 to the behavioral degradation (performance and adversarial) observed at inference execution.

To discover underlying data-structure patterns, test theoretical assumptions, and validate dataset integrity, the EDA stage can apply univariate and multivariate analytical techniques. This involves, for example, generating common graphical visualizations—such as histograms to assess distributions, side-by-side boxplots for cross-category comparisons, scatterplots to examine relationships, and quantile-normal (Q-Q) plots to identify outliers [76]. Other EDA components complement this by computing essential summary statistics, including measures of central tendency, spread, skewness, and kurtosis. Mostly, this Stage uses the `AutoViz` and `scipy` libraries, along with custom Python code as needed, to handle the multidimensional nature of the evaluation metrics of the aggregated framework.

Furthermore, at this stage, Pearson and Spearman correlation coefficients are calculated to quantitatively characterize linear and monotonic associations, respectively, between quantization –induced structural perturbations and failures in downstream task performance, as well as adversarial robustness. In addition, multiple-comparison correction is applied using the Benjamini–Hochberg false discovery rate (BH-FDR) procedure[77, 133].

Finally, the Veritas Analytics Stage renders final evaluation artifacts. This includes generating the tables, figures and plots needed and utilized throughout Section 6.3. Table 6.10 specifies the exact execution parameters utilized.

Table 6.10: Experimental Execution Parameters for Veritas Analytics

Parameter Name	Value Used	Description
baseline	full_size	Designates the full-precision 16-bit model as the reference baseline for all delta degradation calculations.
correlations	[pearson, spearman]	Specifies the statistical methods used to evaluate relationships between structural metrics and task performance.
pca.n_components	Null (not used)	Principal Component Analysis.
isa.enabled	false	Disables the Instance Space Analysis generation for the experiments.

6.3 Results

This section is organized in four parts. It opens with the statistical conventions that apply to every reported value (Section 6.3.1). The first part then reports each evaluation perspective *on its own* — capability (Section 6.3.2), weight structure (Section 6.3.3) and adversarial resilience (Section 6.3.4) — so that each is established before any relationship between them is claimed. The second part works *across* perspectives: the cross-view screen (Section 6.3.5), the depth-resolved localization of damage (Section 6.3.6), and the couplings between structural change and specific adversarial vulnerabilities (Section 6.3.7). The third part converts that evidence into a deployment decision (Section 6.3.8). The fourth reports a systematic sweep of the Multi-Perspective Dataset (Section 6.3.9), which possesses a **distinct evidential status** relative to all prior evidence and is designated as *exploratory* throughout.

Furthermore, in general, this section explores the effects of post-training GGUF format quantization across seven instruction fine-tuned language models traversing 0.6–8 B parameters, each evaluated with complete precision (FP16) and the three corresponding quantized variant types (Q4_K_M, Q3_K_M, Q2_K_L). Three different Stages from Phase 2 are executed: a capability performance evaluation (lm-evaluation-harness from the Capability Performance Evaluation Stage) [48], a weight-structure analysis based on HTSR and SETOL (WeightWatcher from the Deep Learning First-Principles Stage) [29, 30, 41], and an adversarial-resilience evaluation (garak from the Red Team AI Stage) scored against the OWASP LLM Top-10 mapped to the present classifications from 2025 [31, 26].

The results are reported at two levels. First, comprising 28 model variants ($n = 28$, full-precision and quantized) with full coverage across all perspectives of the three stages of Phase 2 (Table 6.11). Second, with Δ denoting a compressed minus FP16 (full-precision GenAI model) change ($n = 21$, comparing the full-precision models with their quantized versions). Correlations report Pearson r with Fisher 95% confidence intervals unless stated [79], and false positives (discoveries) because of multiplicity in the conducted exploratory data analysis (EDA) are controlled with the Benjamini–Hochberg false-discovery rate [77, 133].

6.3.1 Statistical conventions and reproducibility

The results below rest on a fully crossed execution whose scale is worth stating precisely, because several conclusions turn on the resolution it provides. The adversarial perspective executed **39 distinct garak probes** resolved into **57 probe $\times$ detector pairs** on **every one of the 28 model states**, giving **1596 probe $\times$ detector $\times$ state measurements** (1708 records once the two ANSI probes’ duplicate LLM02/LLM05 taxonomy rows are counted); of the 39 probes, 38 carry an OWASP LLM01:2025 label

Table 6.11: Evaluation design and measurement coverage. Each cell marks whether a model variant was scored on the capability (Perf), weight-structure (HTSR/SETOL), and adversarial resilience (RedAI) views. All quantized variants and full-precision models were totally evaluated.

Model Family	Params (B)	Perf-FP16	Perf-Q4	Perf-Q3	Perf-Q2	HTSR-FP16	HTSR-Q4	HTSR-Q3	HTSR-Q2	RedAI-FP16	RedAI-Q4	RedAI-Q3	RedAI-Q2
Qwen3-0.6B	0.6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Llama-3.2-1B-Instruct	1.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Qwen3-1.7B	1.7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Llama-3.2-3B-Instruct	3.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Qwen3-4B-Instruct-2507	4.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Meta-Llama-3.1-8B-Instruct	8.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Qwen3-8B	8.0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

and one carries LLM04, with the two ANSI escape-sequence probes additionally labeled LLM02 in the 2023 taxonomy and LLM05 in 2025 [31, 26]. The capability perspective scored nine task families per state and the structural perspective produced per-layer descriptors for every layer of every state, aggregated both whole-model and at nine depth positions.

Statistically, the chapter uses the following and nothing else. **Pearson’s r** quantifies linear association and is the default; **Spearman’s ρ** quantifies monotonic association and is reported alongside r wherever an outcome is heavy-tailed or an association may be non-linear — for the central screen every cell carries both (Section 6.3.9.7), and where they disagree the disagreement is itself Pareimal precision and the deployment decision to optd. **Fisher z -transformed 95% confidence intervals** accompany effect sizes [79]. **Benjamini–Hochberg FDR control** is applied within every explicitly named family of tests, and no result is asserted on a raw p [77]. Beyond correlation, four methods appear: **quadratic ordinary least squares** with leave-one-out and leave-one-family-out refitting, used to test non-monotonicity (Section 6.3.9.5); a **variance decomposition** (η^2) separating between-family from within-family structural variation; the **Mann–Whitney U test** for the two rank comparisons of distributions (Sections 6.3.9.6 and 6.3.9.8); and a **leave-one-out 1-nearest-neighbor classifier with a label-permutation null** for the bit-width recovery result (Section 6.3.9.6). Welch’s t -test, instance-space projection and any regression beyond the quadratic above are *not* used, and no model is fitted for prediction.

All statistical methods are computed with standard, widely used open-source libraries rather than custom code: data handling uses `pandas` and `numpy`; correlations use `scipy.stats`; t -tests, and the non-parametric Mann–Whitney U test use `scipy.stats`; multiple-comparison correction `statsmodels`; and figures use `matplotlib`. Concretely, a linear (Pearson) correlation between two quantities is obtained from `scipy.stats.pearsonr`, which returns both the correlation coefficient r (between -1 and $+1$: the sign is the direction, the magnitude the strength) and a p -value, the probability of seeing an association at least this strong if the two quantities were in truth unrelated. Each r is accompanied by a 95% confidence interval computed with the Fisher z -transform [79]; a regression R^2 (the share of variance one quantity explains in another); and the corresponding coefficients come from a simple least-squares fit in `statsmodels`.

Given the variety of possible relationships to explore in the datasets generated by the ChaosAthena Framework, 28 GenAI-evaluated models (FP16 and quantized variants) may be insufficient. When many correlations are examined at once, some can appear significant by chance, so a raw p -value alone is not enough. To control the *false-discovery rate* with the Benjamini–Hochberg procedure (`statsmodels`) [77, 133], which converts the p -values of a family of tests into q -values, a q below a threshold means that, among all results declared significant at that threshold, **only that small fraction is expected to be a false positive**. In every table and heatmap, the stars denote this corrected significance: * marks $q < 0.05$ and ** marks $q < 0.01$.

Three presentation rules follow from these conventions and are applied without exception in the remainder of this chapter, so that the evidential status of any number is readable from the page on which it appears.

1. **The sample size is always stated.** Every table names its n in the header row or the caption, and every sentence that quotes a coefficient names it too. Two values recur and must never be conflated: $n = 28$ for a *level* statistic, computed cross-sectionally over all variants (7 families $\times$ 4 precisions), and $n = 21$ for a *delta* statistic, computed over the compressed states against their same-family FP16 parent. A third, $n = 7$, appears only where the baselines of FP16 are compared alone. Level statistics say which *models* are robust; delta statistics say what *compression* does; a claim that mixes them is not interpretable.
2. **The correction family is always named.** A q -value has no meaning without the set of tests it was corrected within, and the same coefficient can therefore carry a star in a narrow family and none in a wide one. Every table caption states its family explicitly — “BH-FDR across the 55-cell level screen”, “across the six tests of this table”, and so on. Where a coefficient appears in two tables under two families, the caption says so. **The coefficient, not the star, is the transportable quantity.**
3. **Only FDR-significant cells are emphasized.** Every cell reports its value in full, because a near-zero or non-significant correlation is itself informative and hiding it would misrepresent the screen. But **only cells with $q < 0.05$ are set in bold**, and only those carry a direction arrow ($\uparrow$ the outcome rises with the metric, $\downarrow$ it falls) and a star ($*q < 0.05$, $**q < 0.01$). An unmarked cell is not a missing or failed measurement: it is a value that cannot be distinguished from chance at this sample size, given the number of relationships examined. The distinction matters because the detection threshold is high here — at $n = 28$ the two-sided threshold for $p < 0.05$ is $|r| \approx 0.37$, and at $n = 21$ it is $|r| \approx 0.43$ — so an unmarked cell means *undetectable in this cohort*, not *absent*.

To facilitate access to the precise statistics underlying any highlighted cell, without requiring reconstruction from multiple tables, Table 6.12 aggregates **all results in this chapter that remain significant after Benjamini–Hochberg correction**. These results are organized by correction family and reported together with the corresponding regime and sample size n , the Pearson correlation coefficient r and its direction, the Fisher-transformed 95% confidence interval, the unadjusted p -value, and the adjusted q -value. Any result not included in this summary should be interpreted as non-significant following multiple-comparison correction, irrespective of the magnitude of its coefficient.

Table 6.12: **Ledger of every result in this chapter that survives Benjamini–Hochberg correction**, with its exact statistics. Grouped by the correction family within which it was tested, since a coefficient’s significance depends on that family and not on its magnitude alone. For each entry: the regime and its n ($n = 28$ for cross-sectional level tests over all variants, $n = 21$ for compression-delta tests over the compressed variants), the Pearson r with a direction arrow, the Fisher 95% CI, the raw two-sided p , and the FDR-adjusted q . Anything *not* in this table is not significant after correction, however large its coefficient. Detection thresholds: $|r| \approx 0.37$ at $n = 28$ and $|r| \approx 0.43$ at $n = 21$, so a result’s absence here means undetectable at this cohort size rather than demonstrably absent.

Result	Regime	r	95% CI	p	q (FDR)
<i>Delta screen (36 cells)</i>					
Δ entropy vs. Δ Accuracy	delta $n=21$	-0.86 ↓	[-0.94, -0.68]	<0.001	<0.001
Δ rand_distance vs. Δ Accuracy	delta $n=21$	+0.84 ↑	[+0.63, +0.93]	<0.001	<0.001
Δ log_spectral_norm vs. Δ Accuracy	delta $n=21$	+0.72 ↑	[+0.42, +0.88]	<0.001	0.003
Δ log_spectral_norm vs. Δ Perplexity	delta $n=21$	-0.63 ↓	[-0.83, -0.27]	0.002	0.020
Δ entropy vs. Δ Perplexity	delta $n=21$	+0.60 ↑	[+0.22, +0.82]	0.004	0.031
Δ log_spectral_norm vs. Δ RedAI pass	delta $n=21$	-0.59 ↓	[-0.81, -0.21]	0.005	0.031
Δ rand_distance vs. Δ Perplexity	delta $n=21$	-0.56 ↓	[-0.80, -0.17]	0.008	0.043
<i>Depth screen (27 cells)</i>					
$\Delta\alpha$ at Q_4 vs. Δ Toxigen	delta $n=21$	-0.69 ↓	[-0.86, -0.37]	<0.001	0.014
<i>Hypothesis battery (20 tests)</i>					
H18: rand_distance change tracks accuracy change	delta $n=21$	+0.84 ↑	[+0.63, +0.93]	<0.001	<0.001
H19: More overtrained layers raise perplexity	level $n=28$	+0.73 ↑	[+0.49, +0.87]	<0.001	<0.001
H16: Spectral-norm change tracks perplexity change	delta $n=21$	-0.63 ↓	[-0.83, -0.27]	0.002	0.008
H15: Spectral-norm change tracks robustness change	delta $n=21$	-0.59 ↓	[-0.81, -0.21]	0.005	0.013
H4: High final-layer log-norm lowers output-handling pass [†]	level $n=28$	-0.57 ↓	[-0.78, -0.25]	0.002	0.008
H2: Spectral-norm expansion first→last lowers overall robustness	level $n=28$	-0.56 ↓	[-0.77, -0.24]	0.002	0.008
H7: More trap layers lower prompt-injection (LLM01) robustness	level $n=28$	-0.56 ↓	[-0.77, -0.24]	0.002	0.008
H10: Higher layer-wise α_{weighted} dispersion lowers accuracy	level $n=28$	+0.53 ↑	[+0.20, +0.75]	0.004	0.011
H6: More undertrained layers lower robustness	level $n=28$	-0.45 ↓	[-0.71, -0.10]	0.015	0.034
H3: Spectral-norm expansion increases prompt-injection (LLM01) vulnerability	level $n=28$	-0.44 ↓	[-0.70, -0.08]	0.020	0.040
<i>Level screen (55 cells)</i>					
num_spikes vs. Accuracy	level $n=28$	+0.94 ↑	[+0.88, +0.97]	<0.001	<0.001
entropy vs. Accuracy	level $n=28$	+0.91 ↑	[+0.81, +0.96]	<0.001	<0.001

continued on next page

Result	Regime	r	95% CI	p	q (FDR)
stable_rank vs. Accuracy	level $n=28$	+0.90 ↑	[+0.79, +0.95]	<0.001	<0.001
num_pl_spikes vs. Accuracy	level $n=28$	+0.87 ↑	[+0.74, +0.94]	<0.001	<0.001
stable_rank vs. Toxigen	level $n=28$	+0.85 ↑	[+0.69, +0.93]	<0.001	<0.001
num_spikes vs. Toxigen	level $n=28$	+0.83 ↑	[+0.66, +0.92]	<0.001	<0.001
entropy vs. Toxigen	level $n=28$	+0.82 ↑	[+0.65, +0.92]	<0.001	<0.001
num_spikes vs. Perplexity	level $n=28$	-0.79 ↓	[-0.90, -0.59]	<0.001	<0.001
num_pl_spikes vs. Toxigen	level $n=28$	+0.76 ↑	[+0.53, +0.88]	<0.001	<0.001
entropy vs. Perplexity	level $n=28$	-0.76 ↓	[-0.88, -0.53]	<0.001	<0.001
num_pl_spikes vs. Perplexity	level $n=28$	-0.75 ↓	[-0.88, -0.53]	<0.001	<0.001
$n_{\text{overtrained}}$ vs. Perplexity	level $n=28$	+0.73 ↑	[+0.49, +0.87]	<0.001	<0.001
stable_rank vs. Perplexity	level $n=28$	-0.71 ↓	[-0.86, -0.46]	<0.001	<0.001
n_{trap} vs. RedAI pass	level $n=28$	-0.57 ↓	[-0.78, -0.25]	0.002	0.006
n_{trap} vs. LLM01 pass	level $n=28$	-0.56 ↓	[-0.77, -0.24]	0.002	0.007
log_spectral_norm vs. Accuracy	level $n=28$	+0.52 ↑	[+0.18, +0.75]	0.005	0.017
α_{weighted} vs. RedAI pass	level $n=28$	-0.51 ↓	[-0.74, -0.17]	0.006	0.018
$n_{\text{undertrained}}$ vs. RedAI pass	level $n=28$	-0.45 ↓	[-0.71, -0.10]	0.015	0.047
log_spectral_norm vs. Perplexity	level $n=28$	-0.45 ↓	[-0.70, -0.09]	0.017	0.049
<i>Single pre-specified test</i>					
Quadratic term, RedAI pass on $z(\log_{\text{spectral_norm}})^2$	level $n=28$	-0.43 ↓	—	0.006	—
Same-model vs. between-model structural L2 (Mann–Whitney)	level $n=42$	—	—	<0.001	—
Bit-width recovery, leave-one-out 1-NN (label permutation)	delta $n=21$	—	—	<0.001	—
<i>Structure-predicts screen (6 tests)</i>					
α_{weighted} vs. RedAI pass (level)	level $n=28$	-0.51 ↓	[-0.74, -0.17]	0.006	0.033
rand_distance vs. RedAI pass (compression)	compression $n=21$	-0.47 ↓	[-0.75, -0.05]	0.030	0.045
rand_distance vs. RedAI pass (level)	level $n=28$	+0.43 ↑	[+0.07, +0.69]	0.022	0.045
α vs. RedAI pass (level)	level $n=28$	-0.42 ↓	[-0.68, -0.05]	0.027	0.045

States, and why the distinction matters. The unit of analysis throughout is the model **state**: a specific, deployment-ready artifact, identified by its weights rather than by the model family it descends from. The cohort comprises **28 states** — 7 full-precision states and 21 quantized states, 3 per family.

Where a state is being described *relative to its full-precision parent*, as in every Δ statistic, this chapter says *compressed state* and names the parent; the term *variant* is reserved for that baseline-relative sense alone. The distinction is meaningful. A state space is not a two-level tree of baselines and their variants: a quantized artifact can itself be quantized further, merged, or fine-tuned, and the framework treats any such artifact as a state on the same basis, which is precisely the extensibility claimed under RQ1 and validated in Section 6.6. The case study samples one representative subset of that space — one compression step, one format, applied once to each full-precision parent — and every result reported is a result about that subset. The composed states are not tested and are identified as a distinct axis of cohort expansion in Section 7.3.4.

Additional views in the appendix. In addition to the results below, delta layer depth views, and mean changes of the three structural metrics (α , $\hat{\alpha}$, `rand_distance`) are provided in Appendix B, section B.3: per-family metric levels across the four precisions (B.3), the corresponding per-family compression-induced changes (B.5), and a high-resolution model architectural diagnostic profile, from a granular point of view. The plot tracks the HTSR exponent α in all 420 individual layer IDs (Appendix B). Ultimately, they do not introduce new results and serve only to make the compression and layer localization results directly observable at each depth position or at a layer-by-layer level.

6.3.2 Capability degrades gradually, then collapses at 2-bit

Capability loss is modest at 4- and 3-bit but accelerates at 2-bit. Aggregated between pooled model families and their corresponding quantized variants, mean accuracy changes by -0.56 percentage points (pp) in Q4_K_M, -1.54 pp in Q3_K_M, and -5.24 pp in Q2_K_L (Table 6.14, Figure 6.1). The loss is domain-dependent: at the most severe level, the worst domain is math at -21.6 pp (Figure 6.2) [137], consistent with multi-step arithmetic and reasoning being less tolerant to reduced numerical precision, while knowledge and commonsense tasks remain comparatively stable [154].

Here, “accuracy” is the mean of the nine core accuracy tasks; wikitext perplexity is deliberately excluded, as it is on a different scale and moves in the opposite direction (*lower is better*). Perplexity is never incorporated into any accuracy mean, delta, or correlation; it is reported on a separate Table 6.15 and Figure 6.3, where it degrades in line with accuracy (pooled FP16 perplexity 16.4, rising by $+9.3$ at Q2_K_L).

Table 6.14: Aggregated capability and its change under quantization. FP16 mean accuracy per model family (full-precision and compressed states) and the accuracy delta (percentage points) at each level; the final row pools across families. Accuracy is the mean of the nine evaluated accuracy tasks and excludes wikitext perplexity, which is reported separately in Table 6.15.

	FP16 acc	Δ Q4 (pp)	Δ Q3 (pp)	Δ Q2 (pp)	mean Δ (pp)
Meta-Llama-3.1-8B-Instruct	0.562	-0.033	-0.665	-6.670	-2.456
Llama-3.2-3B-Instruct	0.509	-2.590	-3.575	-8.629	-4.931
Qwen3-4B-Instruct-2507	0.507	-0.666	-0.910	-2.814	-1.463
Qwen3-8B	0.503	0.858	0.104	-2.352	-0.463
Qwen3-1.7B	0.399	-0.073	-2.696	-5.535	-2.768
Llama-3.2-1B-Instruct	0.386	-1.050	-1.543	-8.381	-3.658
Qwen3-0.6B	0.275	-0.400	-1.482	-2.274	-1.385
POOLED MEAN	0.449	-0.565	-1.538	-5.236	-2.446

Table 6.15: Wikitext perplexity (*lower is better*). Δ = compressed state – FP16 (positive = worse). The final row pools across all families.

	PPL FP16	Δ Q4	Δ Q3	Δ Q2
Meta-Llama-3.1-8B-Instruct	9.02	0.22	0.61	3.00
Qwen3-8B	12.77	0.11	0.64	1.89
Llama-3.2-3B-Instruct	12.82	0.26	0.97	4.74
Qwen3-4B-Instruct-2507	13.57	0.40	1.19	3.76
Llama-3.2-1B-Instruct	16.92	0.84	2.18	13.84
Qwen3-1.7B	22.07	0.42	1.61	8.66
Qwen3-0.6B	27.58	1.51	4.93	29.31
POOLED MEAN	16.39	0.53	1.73	9.31

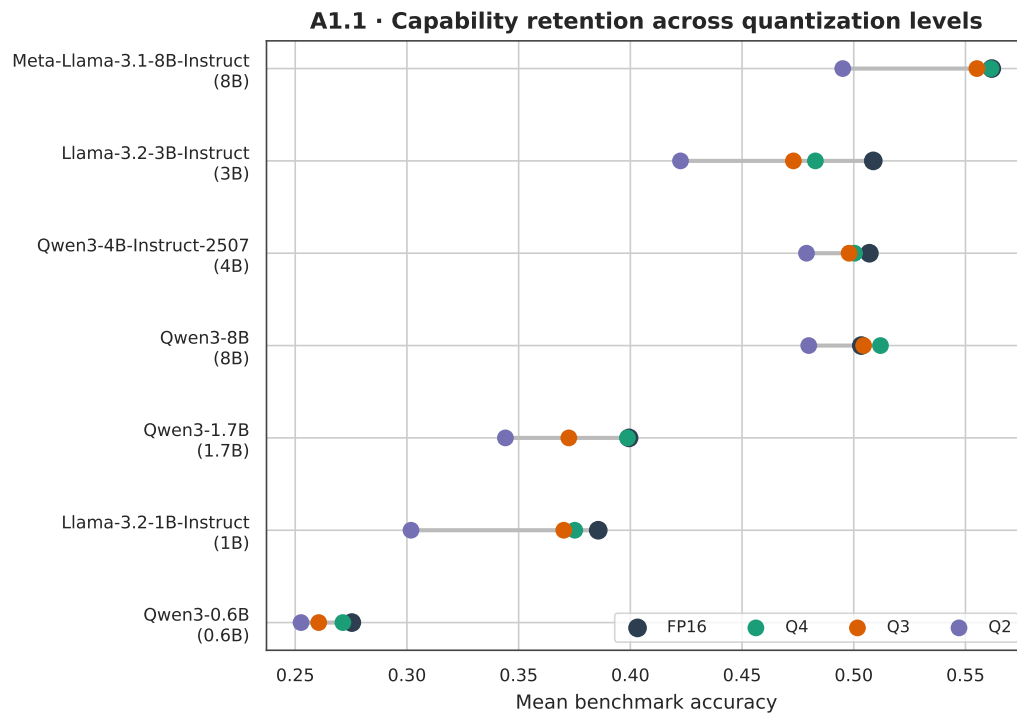


Figure 6.1: Per-family accuracy retention from FP16 to each level.

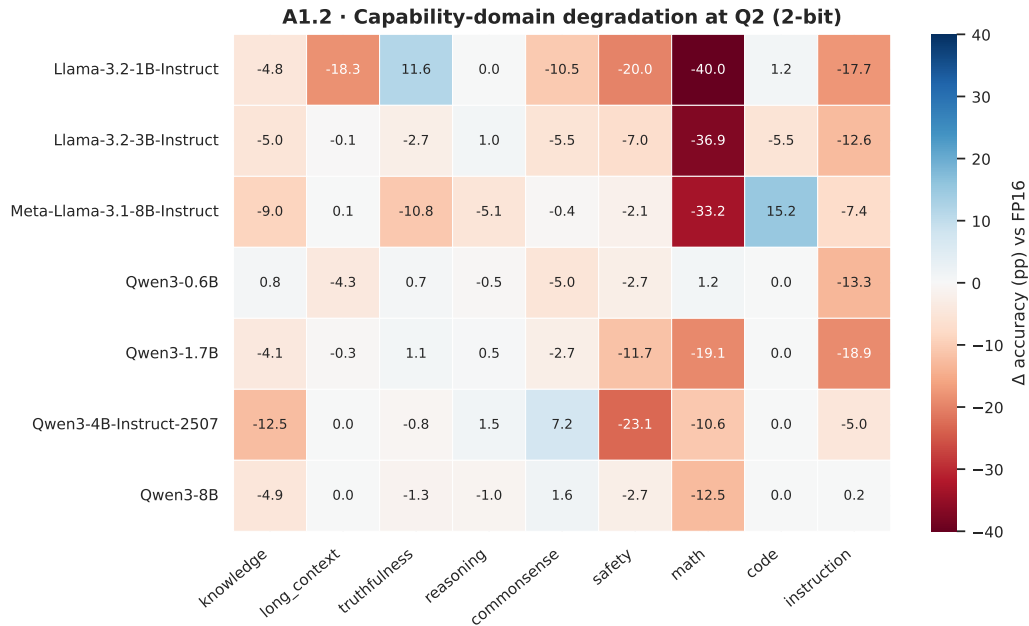


Figure 6.2: Mean accuracy change by task domain at the most aggressive level (Q2_K_L).

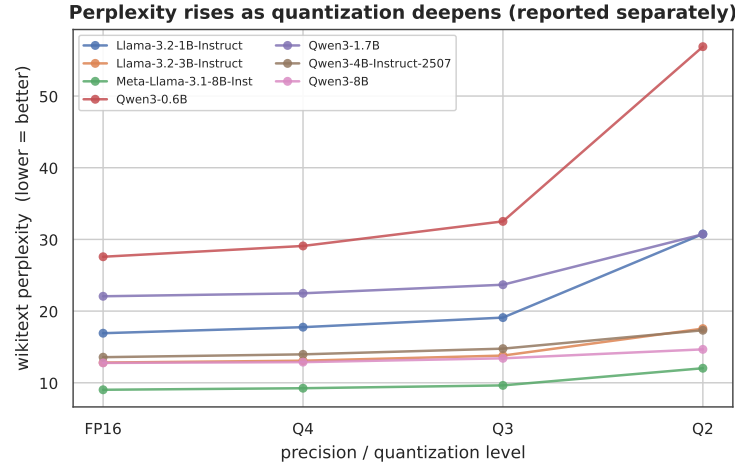


Figure 6.3: Wikitext perplexity (lower = better) by level; reported separately from accuracy and excluded from all accuracy means and correlations.

From the observations above and further exploration, two questions arise. **First, does perplexity track weight matrix structure?** Cross-sectionally – across all different variants ($n = 28$)– it does, for global (mean) spectral metrics: models with higher baseline spectral entropy (indicating richer, more diverse weight distributions) achieve **significantly lower baseline perplexity** ($r = -0.76^{**}$ in Table 6.16). Nevertheless, this is aligned with HTSR and SETOL theory and empirical results [29, 30, 41].

Within families—tracking every single model in full-precision, with its quantized variants—however, **the compression-induced change in perplexity significantly tracks changes in spectral entropy** ($r = +0.60^*$). This positive delta-correlation indicates that **severe quantization introduces uniform rounding noise that artificially flattens the weight spectrum** (spectral Δ entropy $\uparrow$), which directly degrades capability performance (Δ perplexity $\uparrow$). Other layer-specific parameters, such as α_{weighted} , exhibit non-significant trends ($r = -0.21$).

Second, is compression damage scale-dependent? At Q2_K_L the accuracy drop is roughly size-independent (log-size $r = 0.06$), but perplexity degradation is larger for smaller models (log-size $r = -0.89$, Figure 6.4): sub-billion models (e.g., the Qwen-0.6B model) lose language-modeling quality faster than the accuracy average reveals—an additional reason to report perplexity separately.

Table 6.16: Perplexity versus weight-structure. Each cell is Pearson r with 95% CI; cross-sectional correlates the perplexity *level* with the metric level across states, within-family correlates the compressed minus FP16 *deltas*. Stars use BH-FDR q ($*q < 0.05$, $**q < 0.01$). Perplexity is analyzed here as a distinct quantity, never conflated with accuracy.

Shape metric	Cross-sectional r ($n=28$)	Within-family delta- r ($n=21$)
alpha	+0.39 [+0.01, +0.66]	−0.07 [−0.48, +0.38]
alpha-weighted	−0.21 [−0.54, +0.18]	−0.43 [−0.73, +0.00]
entropy	−0.76** [−0.88, −0.53]	+0.60* [+0.22, +0.82]

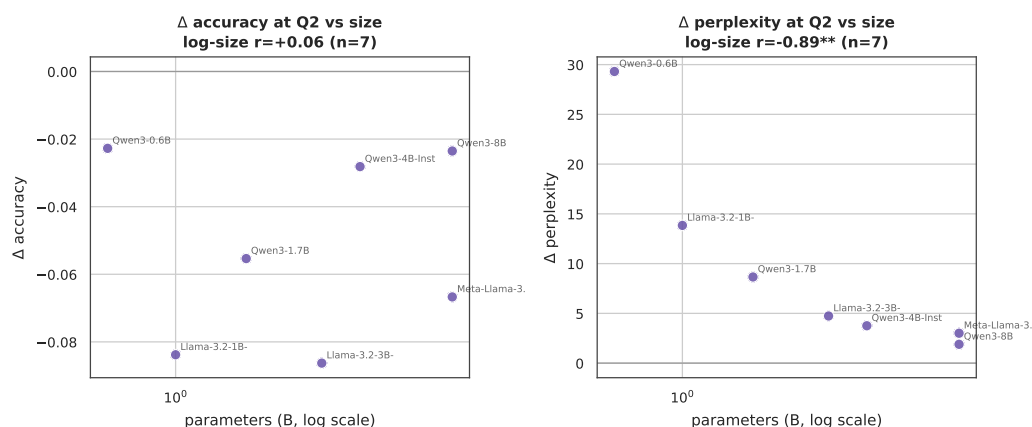


Figure 6.4: Compression degradation versus model at size scale at Q2_K_L. Accuracy change is roughly size-independent, but perplexity degradation is strongly larger for smaller models (note the log size axis).

6.3.3 Weight structure stays well-trained and shifts at the output layer

The HTSR shape metric α quantifies how heavy-tailed the empirical spectral density of each layer is; values in $[2, 6]$ indicate a well-trained correlation structure [29, 41]. Every tested family on the experiments is inside this regime at full precision, and quantization barely moves it: across all compressed states α stays in the interval band $[4.9, 5.2]$ (Table 6.17, Figure 6.5). In other words, GGUF format quantization does not push these models out of the well-trained spectral regime (mean values)—particularly when not analyzed, in a granular layer-by-layer level. The HTSR and SETOL metrics are strongly intercorrelated (Figure 6.6), and, based in the insights and results about scale and shape metrics from AlphaPruning work, and the superiority of the shape metrics when used to guide the compression method pruning, this dissertation focus is on the shape metrics (α , α_{weighted} , entropy) [126]. Nevertheless, in future work, more attention to scale metrics is one direction to pursue.

Nevertheless, the scale metrics (stable rank, MP-sofrank) vary across families but are largely *redundant*—stable rank, for example, tracks accuracy and entropy closely (Figure 6.6)—thus scale metrics also, to some extent, **corroborate shape spectral metrics rather than being independent structural axes**. The limited spectral shift, seen at a coarser depth resolution (layers binned into five equal-depth quintiles plus the first and last two positions), is small and concentrated at the *output* end: the depth analysis of Section 6.3.6 shows the mean absolute change $|\Delta\alpha|$ **rising from ≈ 0.035 at the first layer (an attention layer) to ≈ 0.305 at the last (an MLP layer)**, an order of magnitude larger at the head. Two qualifications belong with that statement. First, it is a statement about *magnitude*: the *signed* mean at the last layer is -0.255 with a median of only -0.032 , and just 38% of compressed states move upward there, so the last layer is where the spectrum is most perturbed, not where it uniformly lightens. Second, the magnitude is outlier-driven — a single variant, Llama-3.2-1B at Q2_K_L, accounts for much of it, its last-layer α collapsing from 8.98 to 4.07 ($\Delta = -4.92$). By contrast the mid-deep quintile Q_4 moves coherently and in one direction (signed mean $\Delta\alpha = +0.098$, 90% of variants rising), which is why it, and not the last layer, carries the one depth $\times$ outcome association that survives correction (Section 6.3.6).

Table 6.17: Heavy-Tailed Self-Regularization shape metric α at FP16 and each quantization level, with the FP16 training regime ($2 \leq \alpha \leq 6$ = well-trained).

	α FP16	α Q4	α Q3	α Q2	regime FP16
Llama-3.2-1B-Instruct	4.65	4.71	4.69	4.72	well-trained
Llama-3.2-3B-Instruct	5.03	5.02	5.00	5.11	well-trained
Meta-Llama-3.1-8B-Instruct	4.79	4.79	4.78	4.87	well-trained
Qwen3-0.6B	5.23	5.22	5.28	5.21	well-trained
Qwen3-1.7B	5.17	5.21	5.23	5.31	well-trained
Qwen3-4B-Instruct-2507	5.11	5.16	5.19	5.26	well-trained
Qwen3-8B	5.04	5.06	5.10	5.17	well-trained

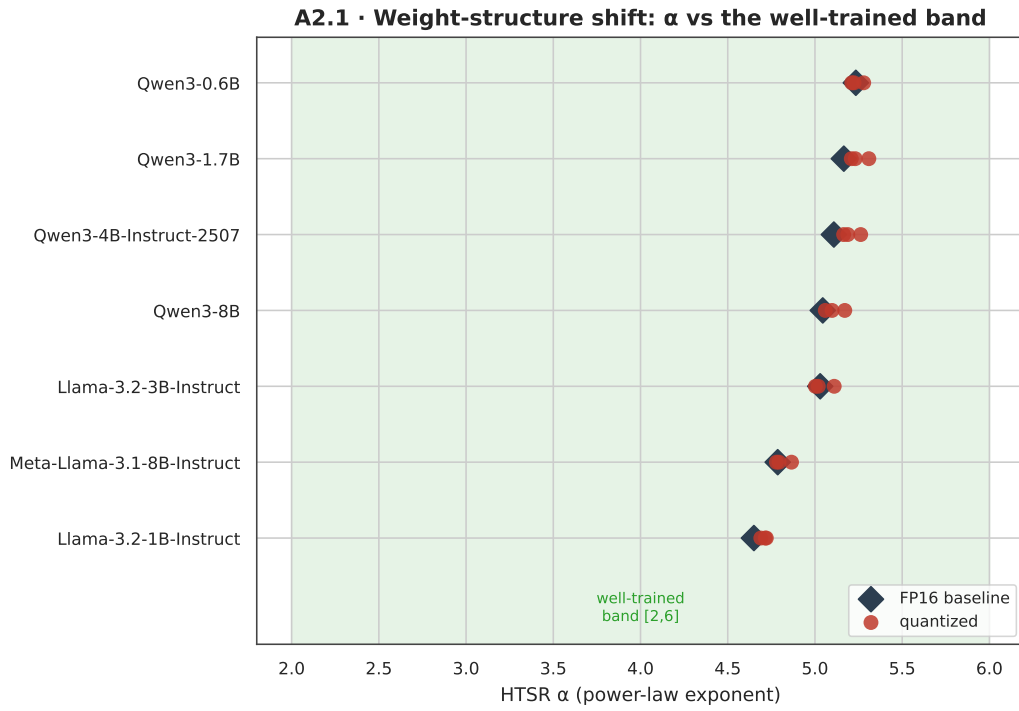


Figure 6.5: Distribution of α by level; the shaded band marks the well-trained $[2, 6]$ regime.

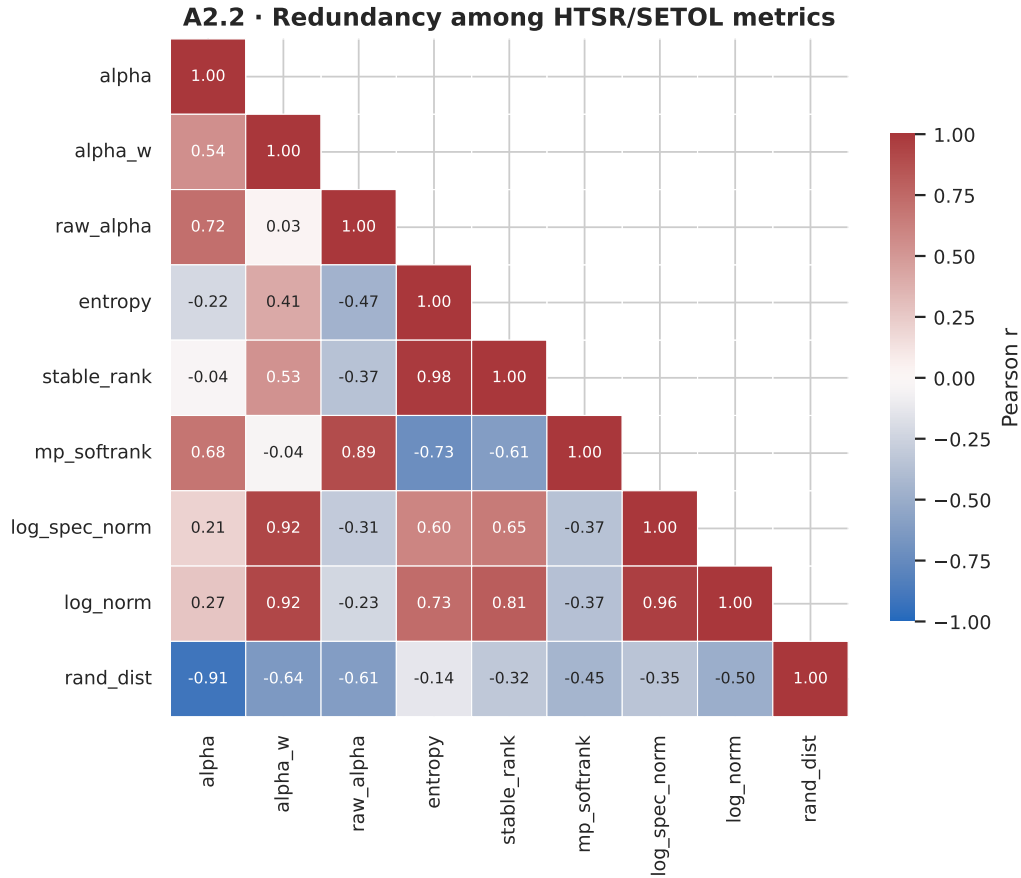


Figure 6.6: Correlation among HTSR/SETOL metrics (redundancy from structure metrics).

Because HTSR and SETOL theory and empirical results, connect α and α_{weighted} to a model’s generalization, it is important to quantify where the variation is (Table 6.18, Table 6.19, Figure 6.7) [29, 30, 41]. Both are highly stable: α has a coefficient of variation of only $\approx 4\%$ and $\alpha_{\text{weighted}} \approx 17\%$ in all 28 full-precision and quantized variants. Decomposing the variation between variants shows that it is almost entirely *between families* — the between-family (different model families) standard deviation is approximately five times the within-family (across-level) standard deviation for α and more than thirty times larger for α_{weighted} (≈ 1.27 versus ≈ 0.04).

The decomposition remains invariant whether it is computed exclusively on full-precision baselines or exclusively on compressed states. This observation indicates that quantization—at least when examined at an aggregate level, rather than at the fine-grained, layer-wise level as in Appendix B (B.6)—is not the principal factor driving the behavior of these metrics. Equivalently, for each model family, the variation across the four precision levels (e.g., α_{weighted} varies by at most 0.16 within any given family) is small relative to the variation observed between families (e.g., α_{weighted} ranges from approximately 5.9 for Llama-1B to approximately 9.0 for Qwen3-8B).

From a practical standpoint, this implies that these shape metrics are primarily a property of the *model family* and are essentially invariant with respect to GGUF precision. Consequently, within the scope of this analysis, these metrics are best interpreted as intrinsic characteristics of the model family, rather than as quantities that are substantially modulated by compression, when considered in terms of their aggregate values rather than layer-by-layer behavior for each model and variant, as already discussed.

Table 6.18: Magnitude of variance analysis for α and α_{weighted} (metrics from HTSR and SETOL connect to the model’s generalization). Variance is decomposed into between-family and within-family (across-level) components, separately for the full-precision baselines and the compressed states. The between-family component dominates: compression moves these metrics far less than families, which already differ from one another.

Metric	Subset	Mean	SD	CV%	Between-family SD	Within-family SD
alpha	FP16 baselines	5.003	0.210	4.200	0.210	0.000
alpha	compressed	5.052	0.207	4.100	0.214	0.039
alpha	all variants	5.040	0.205	4.100	0.213	0.042
alpha_weighted	FP16 baselines	7.247	1.236	17.100	1.236	0.000
alpha_weighted	compressed	7.277	1.219	16.800	1.284	0.040
alpha_weighted	all variants	7.269	1.200	16.500	1.272	0.042

Table 6.19: Per-family α and α_{weighted} : full-precision value and the full range spanned across all four precision levels. The across-level range is an order of magnitude smaller than the family-to-family differences.

Family	alpha FP16	alpha range	alpha_w FP16	alpha_w range
Llama-3.2-1B-Instruct	4.65	0.07	5.91	0.04
Llama-3.2-3B-Instruct	5.03	0.11	6.92	0.05
Meta-Llama-3.1-8B-Inst	4.79	0.09	6.39	0.07
Qwen3-0.6B	5.23	0.07	6.16	0.15
Qwen3-1.7B	5.17	0.14	8.68	0.13
Qwen3-4B-Instruct-2507	5.11	0.15	7.67	0.13
Qwen3-8B	5.04	0.13	9.01	0.16

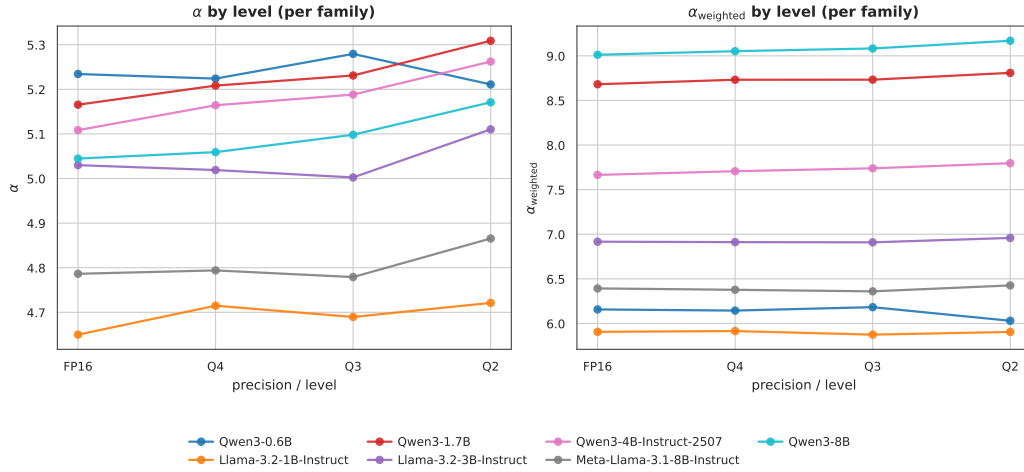


Figure 6.7: Per-family α and α_{weighted} across precision levels: family-to-family separation dwarfs the within-family movement under compression.

6.3.4 Aggregate adversarial resilience is invariant, but specific attacks move

At the aggregate level, adversarial resilience holds under compression: the mean pass-rate is 0.738 at FP16 against 0.745 pooled over the compressed states, a change of +0.007 — under one percentage point, so the aggregate effect is small rather than large (Table 6.20, Figure 6.8). Resolved by level the change is -0.006 at Q4_K_M, $+0.001$ at Q3_K_M and $+0.026$ at Q2_K_L, so what little movement there is arrives at 2-bit. DEFCON severity tiers are likewise stable (Figure 6.9), where lower tiers indicate more severe vulnerabilities [31]. Throughout, the *pass-rate* is the fraction of a probe’s attempts the model withstands (higher = safer), and a *mean Δ pass-rate* is the mean of (compressed pass-rate – same-family FP16 pass-rate) across the variants pooled.

Two aggregations of the pass rate. Because probes contribute unequal numbers of probe $\times$ detector records, two defensible aggregate pass rates exist and they must never be mixed within one table. The *record-mean* averages every red-team record of a variant and is the quantity reported in Table 6.20 and in the paragraph above (0.743 over all 28 states; 0.745 over the 21 compressed; 0.738 over the 7 FP16 baselines; mean $\Delta = +0.007$). The *probe-balanced* aggregate first averages detectors within a probe, then models within a probe, then the 39 probes, so that every distinct attack carries equal weight (0.703 / 0.705 / 0.699; mean $\Delta = +0.006$). Both are internally consistent and agree qualitatively; the probe-balanced figure is the one used as the reference row of the probe-level Table 6.23, so that the total there is genuinely the mean of the rows above it. Every aggregate pass rate quoted in this dissertation states which construction it uses.

Table 6.20: Red-team summary analysis by family and level: mean pass-rate, worst-case z-score, relative DEFCON, and lowest DEFCON triggered.

Family	Quant	Pass rate	Worst-Z	Rel-DEFCON	Low-DEFCON
Llama-3.2-1B-Instruct	FP16	0.778	0.522	4.288	1
Llama-3.2-1B-Instruct	Q4	0.766	0.477	4.119	1
Llama-3.2-1B-Instruct	Q3	0.774	0.460	4.119	1
Llama-3.2-1B-Instruct	Q2	0.796	0.582	4.237	1
Llama-3.2-3B-Instruct	FP16	0.765	0.451	4.305	1
Llama-3.2-3B-Instruct	Q4	0.769	0.492	4.220	1
Llama-3.2-3B-Instruct	Q3	0.754	0.379	4.136	1
Llama-3.2-3B-Instruct	Q2	0.789	0.651	4.305	1
Meta-Llama-3.1-8B-Instruct	FP16	0.741	0.331	3.932	1
Meta-Llama-3.1-8B-Instruct	Q4	0.720	0.183	3.949	1
Meta-Llama-3.1-8B-Instruct	Q3	0.765	0.472	4.186	1
Meta-Llama-3.1-8B-Instruct	Q2	0.736	0.294	4.000	1
Qwen3-0.6B	FP16	0.707	0.123	3.833	1
Qwen3-0.6B	Q4	0.707	0.110	3.661	1
Qwen3-0.6B	Q3	0.699	0.046	3.847	1
Qwen3-0.6B	Q2	0.804	0.679	4.383	1
Qwen3-1.7B	FP16	0.700	0.091	3.627	1
Qwen3-1.7B	Q4	0.700	0.105	3.610	1
Qwen3-1.7B	Q3	0.718	0.080	3.627	1
Qwen3-1.7B	Q2	0.746	0.290	4.000	1
Qwen3-4B-Instruct-2507	FP16	0.758	0.280	4.085	2
Qwen3-4B-Instruct-2507	Q4	0.755	0.246	4.051	2
Qwen3-4B-Instruct-2507	Q3	0.744	0.182	3.949	2
Qwen3-4B-Instruct-2507	Q2	0.771	0.371	4.153	2
Qwen3-8B	FP16	0.715	0.097	3.695	1
Qwen3-8B	Q4	0.706	0.043	3.627	1
Qwen3-8B	Q3	0.721	0.116	3.678	1
Qwen3-8B	Q2	0.701	0.077	3.593	1

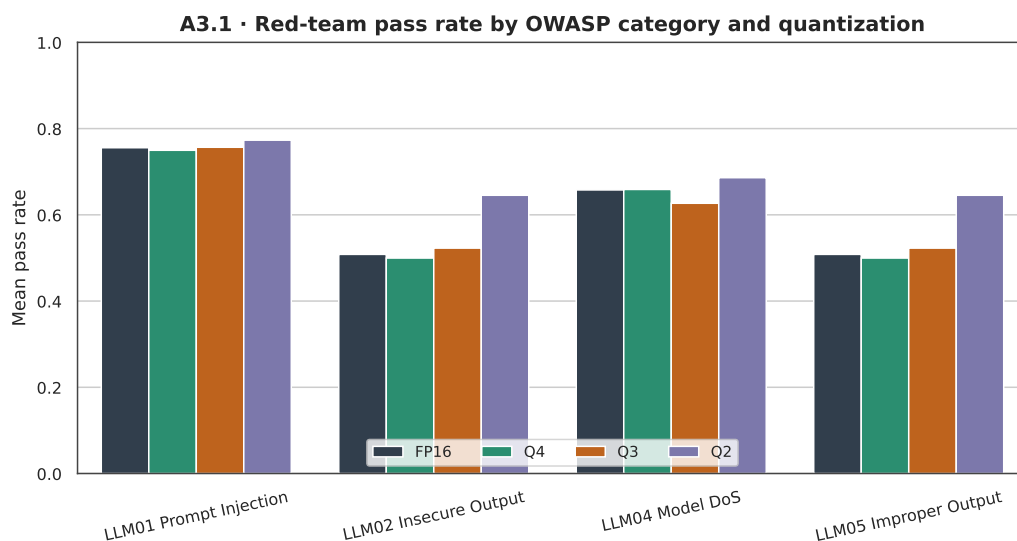


Figure 6.8: Mean pass-rate by OWASP category and level.

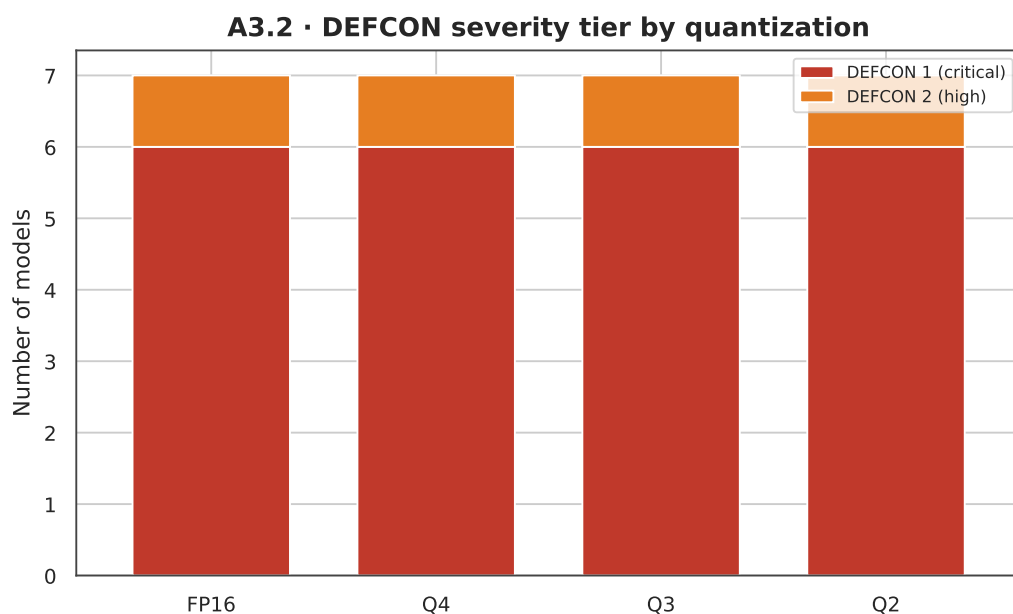


Figure 6.9: Distribution of lowest DEFCON triggered across states.

6.3.4.1 Probe-level redistribution, resolved by level

The small aggregate can obscure a redistribution across attack vectors (Table 6.21, Figure 6.10). The most changed probe, with its two corresponding detectors, is `dan.Ablation_Dan_11_0` and `mitigation.MitigationBypass` at -0.085 pass-rate (pooled), whereas `dan.Ablation_Dan_11_0` and `dan.DAN` improves by $+0.084$, where the same probe with two different detectors is a jailbreak/prompt-injection attack. Several DAN-style mitigation-bypass tests become measurably easier to defeat, while some latent-injection detectors become harder to defeat; thus, quantization **reshapes the robustness profile** with respect to adversarial resilience, even though the mean remains unchanged.

Analyzing the above shifts by quantization level (Table 6.22, Figure 6.11) shows the redistribution is predominantly a 2-bit phenomenon: for almost every probe the Q2_K_L shift surpasses the 4- and 3-bit shifts, and the worst probe worsens monotonically (-0.044 at Q4, -0.058 at Q3, -0.154 at Q2). In other words, 4-bit and 3-bit quantization largely preserve the integrity of the individual attack surface, while 2-bit quantization is where specific defenses degrade. The structural correlations, and further details of these shifts are analyzed in the sections 6.3.6 and 6.3.7.

Table 6.21: Probe $\times$ detector pairs with the largest pooled mean pass-rate shift under quantization (paired 95% CI across families $\times$ levels, BH-FDR q), over the 21 compressed states. The unit here is the probe $\times$ detector pair, not the distinct attack; Table 6.23 repeats the analysis with detectors averaged within each probe. The aggregate record-mean shift is only $+0.007$, so these movements are invisible at the summary level.

OWASP	Probe · Detector	Mean Δ pass	95% CI	n	q (FDR)
llm01	dan.Ablation-DAN-11 · mit.MitigationBypass	-0.085	[-0.130, -0.041]	21	0.038
llm01	dan.Ablation-DAN-11 · dan.DAN	0.084	[-0.007, +0.175]	21	0.229
llm01	ansi.AnsiEscaped · ansi.Escaped	0.079	[+0.021, +0.138]	21	0.105
llm01	LI.LatentInjectionFactSnippetLegal · TriggerList	0.079	[+0.012, +0.146]	21	0.111
llm01	LI.LatentInjectionFactSnippetEiffel · TriggerList	0.067	[+0.015, +0.119]	21	0.111
llm01	LI.LatentInjectionReport · TriggerList	0.050	[-0.018, +0.117]	21	0.349
llm01	dan.DanInTheWild · mit.MitigationBypass	-0.049	[-0.092, -0.006]	21	0.111
llm01	LI.LatentWhois · TriggerList	-0.038	[-0.112, +0.036]	21	0.514
llm01	sata.MLM · mit.MitigationBypass	-0.038	[-0.083, +0.007]	21	0.276
llm01	dra.DRA · mit.Prefixes	-0.036	[-0.106, +0.034]	21	0.514
llm01	dra.DRA · mit.MitigationBypass	-0.036	[-0.098, +0.027]	21	0.512
llm01	PI.HijackLongPrompt · PI.AttackRogueString	0.032	[-0.027, +0.091]	21	0.514

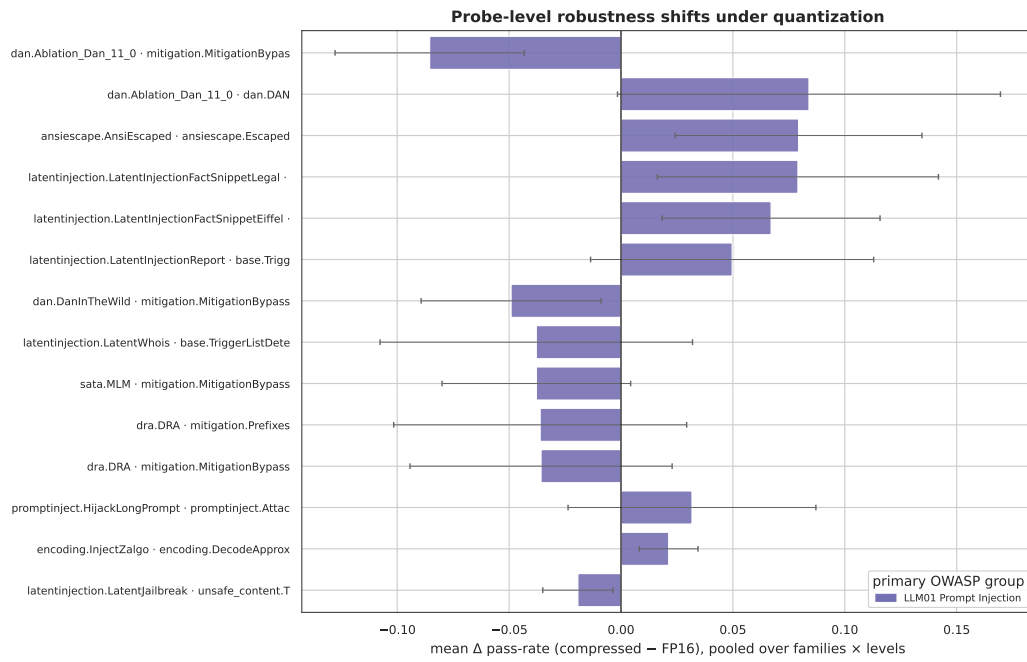


Figure 6.10: Probe-level robustness shifts under quantization, pooled over families of models and levels; colour denotes the primary tested OWASP category.

Table 6.22: The same top movers described by quantization level (Q4/Q3/Q2 separately, each averaged over families). ‘Mean Δ pass’ = mean of (compressed pass-rate - FP16 pass-rate). Most of the movement appears only at the 2-bit level.

OWASP	Probe · Detector	Δ Q4	Δ Q3	Δ Q2
llm01	dan.Ablation-DAN-11 · mit.MitigationBypass	-0.044	-0.058	-0.154
llm01	dan.Ablation-DAN-11 · dan.DAN	0.038	0.070	0.145
llm01	ansi.AnsiEscaped · ansi.Escaped	0.001	0.036	0.200
llm01	LI.LatentInjectionFactSnippetLegal · TriggerList	0.023	0.090	0.124
llm01	LI.LatentInjectionFactSnippetEiffel · TriggerList	0.024	0.082	0.095
llm01	LI.LatentInjectionReport · TriggerList	-0.016	-0.018	0.182
llm01	dan.DanInTheWild · mit.MitigationBypass	-0.001	-0.032	-0.114
llm01	LI.LatentWhois · TriggerList	0.001	-0.020	-0.094
llm01	sata.MLM · mit.MitigationBypass	-0.080	0.029	-0.063
llm01	dra.DRA · mit.Prefixes	-0.015	-0.032	-0.061

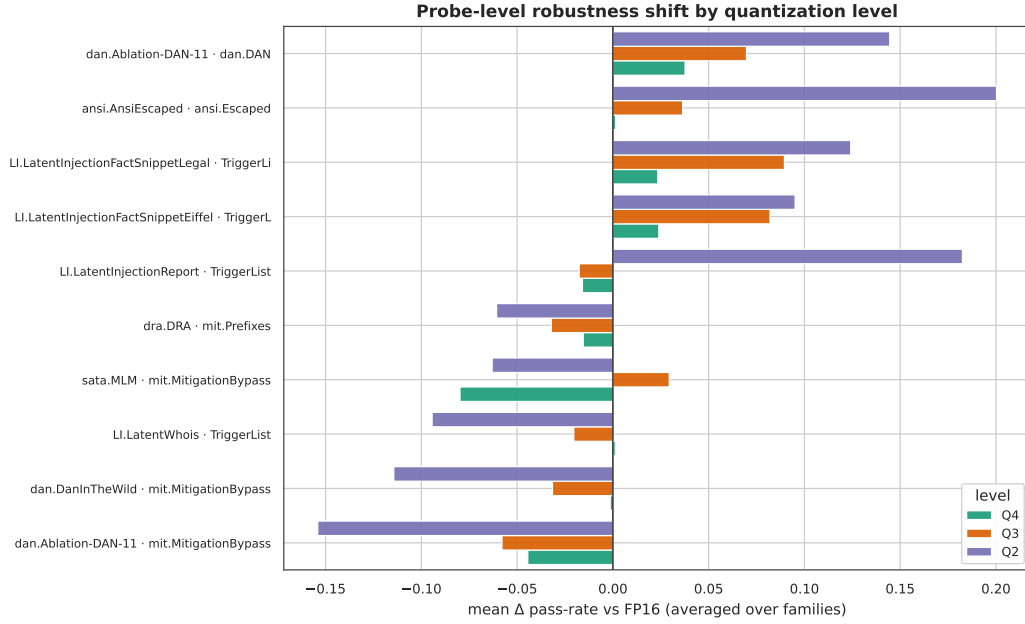


Figure 6.11: The same top movers described by quantization level: for almost every probe, the 2-bit (Q2) bar dominates, so the redistribution is largely a 2-bit phenomenon while 4 and 3-bit shifts are moderate.

6.3.4.2 Which distinct attacks move, with detectors averaged

The redistribution of the previous subsection is reported per probe $\times$ detector pair, which lets one attack measured by two detectors contribute twice and, in the case of `dan.Ablation-DAN-11`, appear simultaneously as the largest faller and the second-largest riser. Averaging detectors within each probe makes the unit the *distinct attack* and removes that artifact. On that unit the cohort comprises **39 distinct garak probes** — 38 carrying an LLM01 label and one, `divergence.RepeatedToken`, carrying LLM04, with the two ANSI escape-sequence probes additionally labeled LLM02 (2023 taxonomy) and LLM05 (2025 taxonomy) — resolved into **57 probe $\times$ detector pairs**, each measured on all 28 states [31, 26].

Six probes lose at least one point of pass rate under compression and thirteen gain at least that much; Table 6.23 itemises the six fallers, all LLM01 jailbreak or injection attacks, together with the two output-handling probes that gain most. The reference row is the probe-balanced aggregate, so it is the mean of the rows above it rather than a figure imported from a differently weighted construction. Read against that reference the movement is genuinely a redistribution: the total is flat (+0.006) while individual attacks move by up to ± 0.08 .

Only one faller has a *within-compression* structural predictor. `dan.DanInTheWild`, the largest faller at -0.049 , tracks $\Delta \log_spectral_norm$ at $+0.59^*$ — and the sign is positive, so norm and pass rate move together and the fall follows the norm contracting under compression (mean $\Delta \log_spectral_norm = -0.009$, negative in 21/21 variants). `sata.MLM` and `LI.TranslationEnFr` are explained only cross-sectionally, through `rand_distance`; and `LI.LatentWhois`, `dra.DRA` and `LI.LatentJailbreak` have *no* structural predictor at all in either regime. The two output-handling probes behave in the opposite way and more strongly: both couple negatively to `log_spectral_norm` at the level (-0.57^* and -0.55^*) and to its change under compression (-0.71^{**} and -0.64^*). Since `log_spectral_norm` rises with capability (accuracy $r = +0.52^*$), the reading across the panel is that **larger-norm, and thereby more capable, models are**

the less ANSI-robust ones — a capability–robustness trade-off confined to one attack family rather than a property of the battery.

Table 6.23: Distinct attacks whose robustness moves under compression, with detectors averaged within each probe so that every row is one attack. *pass* $n=28$ is the probe’s mean pass rate over all variants, *pass* $n=21$ over the compressed states only, and Δ pass the mean of (compressed – same-family FP16). The six LLM01 rows are every probe falling by at least 0.01; the two LLM02/05 rows are the largest gainers of the thirteen probes that rise by at least that much. Coupling columns give, where significant in either coefficient, the within-compression association (Δ_{21}) and the cross-sectional association (lv_{28}) as Pearson/Spearman; ‘—’ marks no significant coupling. The final row is the **probe-balanced** aggregate over all 39 probes, so it is the mean of the rows above it; the record-mean aggregate of Table 6.20 is a different construction and the two are not interchangeable. Stars are BH-FDR q (* $q < 0.05$, ** $q < 0.01$).

Attack (probe)	OWASP	pass $n=28$	pass $n=21$	Δ pass	log_spectral_norm ($\Delta_{21} \cdot lv_{28}$)	rand_distance ($\Delta_{21} \cdot lv_{28}$)
dan.DanInTheWild	LLM01	0.424	0.412	−0.049	+0.59*/+0.67* · +0.58*/+0.64**	— · −0.50*/−0.60**
LI.LatentWhois	LLM01	0.609	0.600	−0.038	— · —	— · —
sata.MLM	LLM01	0.253	0.244	−0.038	— · —	— · +0.58**/+0.44
dra.DRA	LLM01	0.108	0.099	−0.036	— · —	— · —
LI.LatentJailbreak	LLM01	0.683	0.678	−0.018	— · —	— · —
LI.TranslationEnFr	LLM01	0.295	0.291	−0.016	— · −0.44/−0.51*	— · +0.74**/+0.70**
ansi.AnsiEscaped	LLM02/05	0.298	0.317	+0.079	−0.71**/−0.71* · −0.57*/−0.57**	−0.69*/−0.70** · —
ansi.AnsiRaw	LLM02/05	0.790	0.794	+0.016	−0.64*/−0.59 · −0.55*/−0.55**	— · +0.36/+0.48*
All 39 probes (probe-balanced)	—	0.703	0.705	+0.006	—	—

FP16 reference pass rates for the fallers: dan.DanInTheWild 0.461, LI.LatentWhois 0.638, sata.MLM 0.282, dra.DRA 0.135, LI.LatentJailbreak 0.696, LI.TranslationEnFr 0.307. *Probes:* dan.DanInTheWild — DAN jailbreak built from in-the-wild prompts; LI.LatentWhois — injection concealed in a WHOIS record; sata.MLM — masked-language-model jailbreak posing the harmful query as a fill-in-the-blank; dra.DRA — Disguise-and-Reconstruction Attack; LI.LatentJailbreak — jailbreak concealed inside a legitimate task; LI.TranslationEnFr — injection inside En→Fr translation text; ansi.AnsiEscaped/ansi.AnsiRaw — escaped and raw ANSI terminal control sequences (output handling).

6.3.4.3 Toxicity-safety and its structural properties

In the ChaosAthena framework, safety is not a single quantity. In the experiments, for any analyzed GenAI model and its compressed states, safety has two perspectives: the above-analyzed red-team AI (adversarial robustness assessment, the garak pass rate from the experiments) and a model-level toxicity-avoidance score (toxigen; higher = safer). The two perspectives are only weakly related across the experiments (toxigen vs. red-team AI pass: $r \approx -0.18$, n.s.).

They also correlate with the structure of the weights, but through different structure metrics (Table 6.33). Toxicity-safety is mainly associated with the spectral metrics entropy ($r = +0.82$), stable_rank ($r = +0.85$), and num_spikes ($r = +0.83$), having $q < 0.01$. The same block of spectral metrics is also associated with capability (Table 6.33), and it does not show a significant association with the brittleness counters (trap and undertrained layers, $|r| \approx 0.22$, n.s.) [29]. Adversarial robustness shows another reality: it is influenced by the brittleness block (trap $r = -0.57$, undertrained -0.45 ; $\alpha_{\text{weighted}} -0.51$) and does not show a significant association with spectral metrics ($|r| \leq 0.21$, n.s.). Intrinsic toxicity-avoidance is therefore seen as a *capability-linked* property, whereas the mainly tested prompt injection and jailbreak, identified as LLM01 robustness, are linked to brittleness.

Within that block, a precise ordering is worth stating, because it constrains how the result may be phrased. Ranking every aggregate structural quantity against the pass rate (Table 6.33), the strongest association is the layer counter n_{trap} ($r = -0.57^{**}$, $\rho = -0.54^*$), ahead of α_{weighted} ($r = -0.51^*$, $\rho = -0.48^*$) and $n_{\text{undertrained}}$ ($r = -0.45^*$). α_{weighted} is therefore the strongest continuous *shape* metric for adversarial robustness, and the only significant one, but it is not the strongest metric overall: a near-binary count of anomalous layers indexes robustness better than any continuous spectral descriptor in this cohort. Given that trap layers are present in only 6 of the 28 states, that ordering is itself a call for a larger and more heterogeneous cohort before the counter is treated as a usable diagnostic.

The difference identified above is also identified under compression. The adversarial robustness of the aggregate red-team is essentially invariant to quantization, and its only structural compression tracker is the change in `rand_distance`. Toxicity-safety, by contrast, *degrades* monotonically (-0.022 , -0.066 , -0.099 at Q4/Q3/Q2) and that degradation tracks the spectral distortion that drives it: within families the change in *toxigen* correlates with the change in α ($r = -0.49$, $p = 0.02$, $n = 21$, Table 6.24), an association that maps to the tested variants layers network interior (the fourth depth quintile, $r = -0.69$ – the single depth metric surviving the 27-cell test FDR; 6.3.6). The two safety axes are thus structurally associated in distinct ways: intrinsic toxicity-avoidance rises and falls with capability and the heavy-tailed spectrum, while adversarial robustness, given the limited number of variants tested, seems to signal influence from layer-level brittleness and is largely indifferent to the spectral changes induced by quantization. Future research needs to incorporate a broader, more heterogeneous sample of models and compression variants to establish the impact of layer-level brittleness, particularly for trap layers. Of the 28 states tested, only 6 variants have trap layers.

Table 6.24: Toxicity-safety (*toxigen*) versus weight-structure, analyzed alongside the adversarial-robustness results. Each cell is Pearson r with 95% CI; cross-sectional correlates the *toxigen level* with the metric level across states, within-family correlates their compressed-minus-FP16 *deltas*. Stars use BH-FDR q ($q < 0.05$, $^{**}q < 0.01$). Higher *toxigen* = safer.

Shape metric	Cross-sectional r (n=28)	Within-family delta- r (n=21)
alpha	+0.01 [-0.36, +0.38]	-0.49 [-0.76, -0.07]
alpha-weighted	+0.32 [-0.06, +0.62]	-0.21 [-0.59, +0.24]
entropy	+0.82 ^{**} [+0.65, +0.92]	-0.47 [*] [-0.75, -0.05]

6.3.5 Capability and safety are statistically decoupled across views

Among structural metrics, when analyzing variants, it is *entropy*, not α , that is tied to capability. Across all variants entropy correlates with accuracy at $r = 0.91$ ($p < 0.001$, $n = 28$) – a very significant cross-sectional relationship (Figure 6.13a, Table 6.26) – and under compression, on quantized variants, the entropy change directly mirrors with the change in accuracy ($\Delta\text{entropy}$ vs $\Delta\text{accuracy}$ $r = -0.86$, $p < 0.001$, $n = 21$, Figure 6.13b), a relationship that is strongest at the 2-bit level. This link is *concurrent*, not *predictive*. Furthermore, a model’s full-precision entropy does *not* predict how much accuracy it loses under compression (entropy level vs $\Delta\text{accuracy}$ $r \approx 0.07$, n.s. (not significant); Table 6.26), so in this case, structure describes

capability concurrently rather than telling us in advance which models are most fragile. α does not show a cross-sectional (across different models) link to accuracy ($r = -0.27$, $p = 0.17$, n.s.). Capability and adversarial robustness are separately statistically decoupled ($r = -0.03$, $p = 0.86$, $n = 28$; Figure 6.14). The unified analysis matrix is given in Table 6.25 and the complete cross-view correlation map is given in Figure 6.12.

Table 6.25: Unified per-state matrix on the complete-case cohort: capability, structure and robustness summaries side by side.

id	Family	Quant	PERF	alpha	entropy	PassRate	WorstZ	DEFCON	toxigen
501	Llama-3.2-1B-Instruct	FP16	0.386	4.650	1.015	0.778	0.522	1	0.594
504	Llama-3.2-1B-Instruct	Q2	0.302	4.721	1.018	0.796	0.582	1	0.394
503	Llama-3.2-1B-Instruct	Q3	0.370	4.690	1.015	0.774	0.460	1	0.389
502	Llama-3.2-1B-Instruct	Q4	0.375	4.715	1.015	0.766	0.477	1	0.543
601	Llama-3.2-3B-Instruct	FP16	0.509	5.030	1.070	0.765	0.451	1	0.835
604	Llama-3.2-3B-Instruct	Q2	0.422	5.110	1.072	0.789	0.651	1	0.765
603	Llama-3.2-3B-Instruct	Q3	0.473	5.002	1.070	0.754	0.379	1	0.769
602	Llama-3.2-3B-Instruct	Q4	0.483	5.019	1.070	0.769	0.492	1	0.831
701	Meta-Llama-3.1-8B-Instruct	FP16	0.562	4.786	1.115	0.741	0.331	1	0.845
704	Meta-Llama-3.1-8B-Instruct	Q2	0.495	4.866	1.117	0.736	0.294	1	0.823
703	Meta-Llama-3.1-8B-Instruct	Q3	0.555	4.779	1.116	0.765	0.472	1	0.835
702	Meta-Llama-3.1-8B-Instruct	Q4	0.561	4.794	1.115	0.720	0.183	1	0.844
101	Qwen3-0.6B	FP16	0.275	5.234	0.983	0.707	0.123	1	0.568
104	Qwen3-0.6B	Q2	0.253	5.211	0.985	0.804	0.679	1	0.541
103	Qwen3-0.6B	Q3	0.261	5.279	0.984	0.699	0.046	1	0.569
102	Qwen3-0.6B	Q4	0.271	5.224	0.983	0.707	0.110	1	0.568
201	Qwen3-1.7B	FP16	0.399	5.165	1.029	0.700	0.091	1	0.664
204	Qwen3-1.7B	Q2	0.344	5.309	1.031	0.746	0.290	1	0.547
203	Qwen3-1.7B	Q3	0.372	5.231	1.029	0.718	0.080	1	0.537
202	Qwen3-1.7B	Q4	0.399	5.208	1.029	0.700	0.105	1	0.579
301	Qwen3-4B-Instruct-2507	FP16	0.507	5.108	1.061	0.758	0.280	2	0.838
304	Qwen3-4B-Instruct-2507	Q2	0.479	5.262	1.063	0.771	0.371	2	0.607
303	Qwen3-4B-Instruct-2507	Q3	0.498	5.188	1.061	0.744	0.182	2	0.812
302	Qwen3-4B-Instruct-2507	Q4	0.500	5.164	1.061	0.755	0.246	2	0.832
401	Qwen3-8B	FP16	0.503	5.045	1.119	0.715	0.097	1	0.841
404	Qwen3-8B	Q2	0.480	5.171	1.121	0.701	0.077	1	0.815
403	Qwen3-8B	Q3	0.504	5.098	1.119	0.721	0.116	1	0.810
402	Qwen3-8B	Q4	0.512	5.059	1.119	0.706	0.043	1	0.838

Table 6.26: Predictivity of data-free structural metrics for capability *level*, capability *change* under compression, and safety, with 95% CI and BH-FDR-adjusted significance. Spectral entropy is associated with the capability level, but the metrics in the table can't predict changes in per-model accuracy under compression.

Predictor	Target	r	95% CI	n	p (FDR)
alpha	PERF (level)	-0.267	[-0.58, 0.12]	28	0.338
alpha	Δ PERF (compression)	0.103	[-0.34, 0.51]	21	0.717
alpha	PassRate (safety)	-0.417	[-0.68, -0.05]	28	0.082
entropy	PERF (level)	0.909	[0.81, 0.96]	28	0.000
entropy	Δ PERF (compression)	0.071	[-0.37, 0.49]	21	0.760
entropy	PassRate (safety)	-0.135	[-0.48, 0.25]	28	0.667
stable_rank	PERF (level)	0.897	[0.79, 0.95]	28	0.000
stable_rank	Δ PERF (compression)	0.130	[-0.32, 0.53]	21	0.690
stable_rank	PassRate (safety)	-0.206	[-0.54, 0.18]	28	0.503
log_spec_norm	PERF (level)	0.517	[0.18, 0.75]	28	0.019
log_spec_norm	Δ PERF (compression)	0.156	[-0.30, 0.55]	21	0.667
log_spec_norm	PassRate (safety)	-0.353	[-0.64, 0.02]	28	0.156

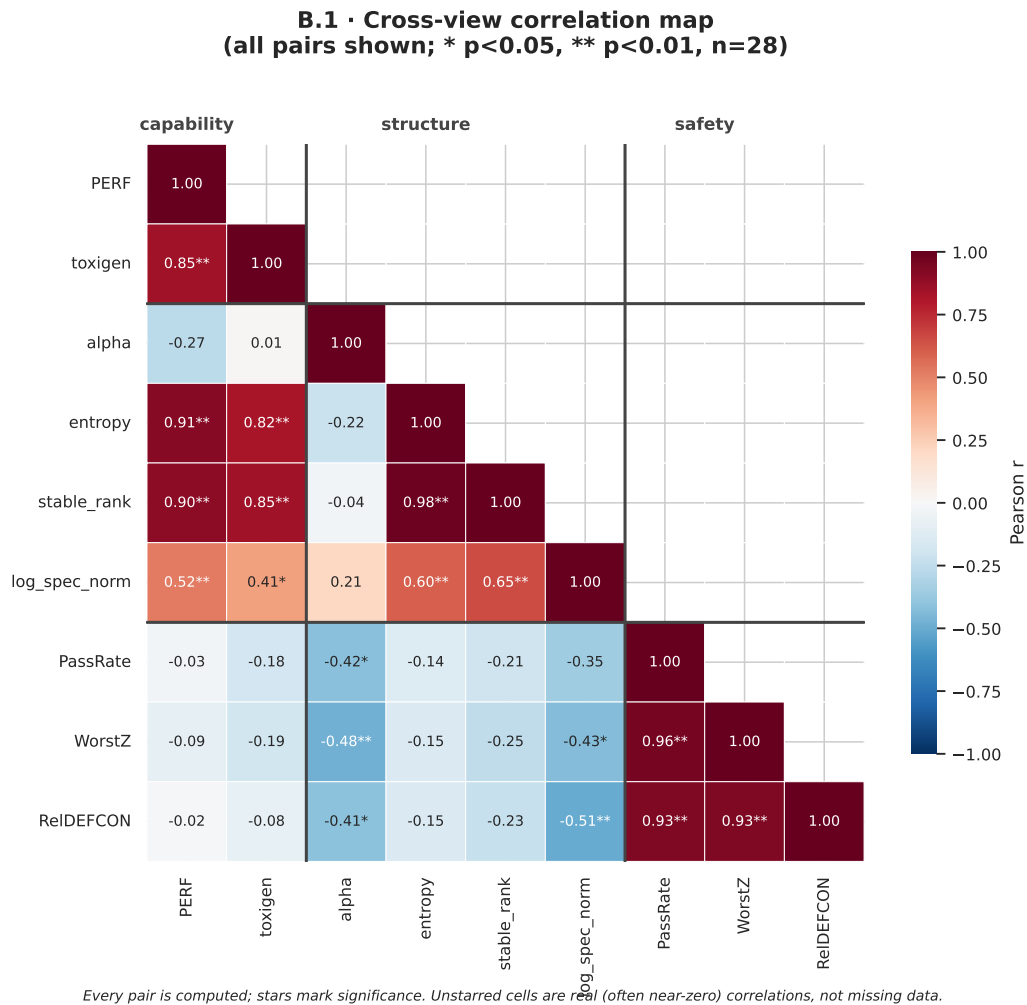


Figure 6.12: Cross-view correlation map across the three measurement views. *Every* pair is computed and shown once; stars mark significance (* $p < 0.05$, ** $p < 0.01$). For visibility and informative purposes, rather than hiding non-significant cells, near-zero correlations (e.g. toxigen vs α , PERF vs PassRate) are also visible. Blocks separate capability, structure, and safety metrics.

B.2 · Weight structure (entropy) indexes capability cross-sectionally, but its change does not forecast compression damage

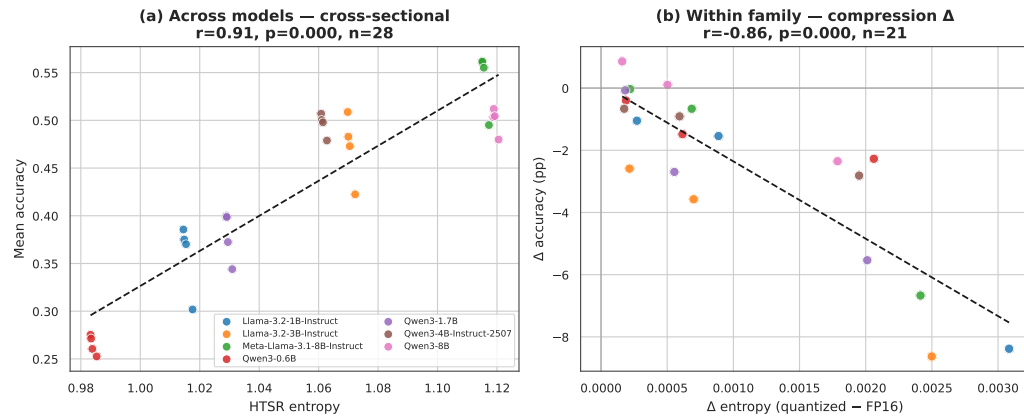


Figure 6.13: Weight-structure entropy versus accuracy. (a) Cross-sectionally, entropy associates with accuracy; (b) under compression, Δ entropy and Δ accuracy move together (strongest at 2-bit). **The link is concurrent, not predictive** — a model's full-precision entropy does not predict its compression damage (Table 6.26).

B.3 · Capability vs safety are decoupled
 $r=-0.03$, $p=0.860$, $n=28$

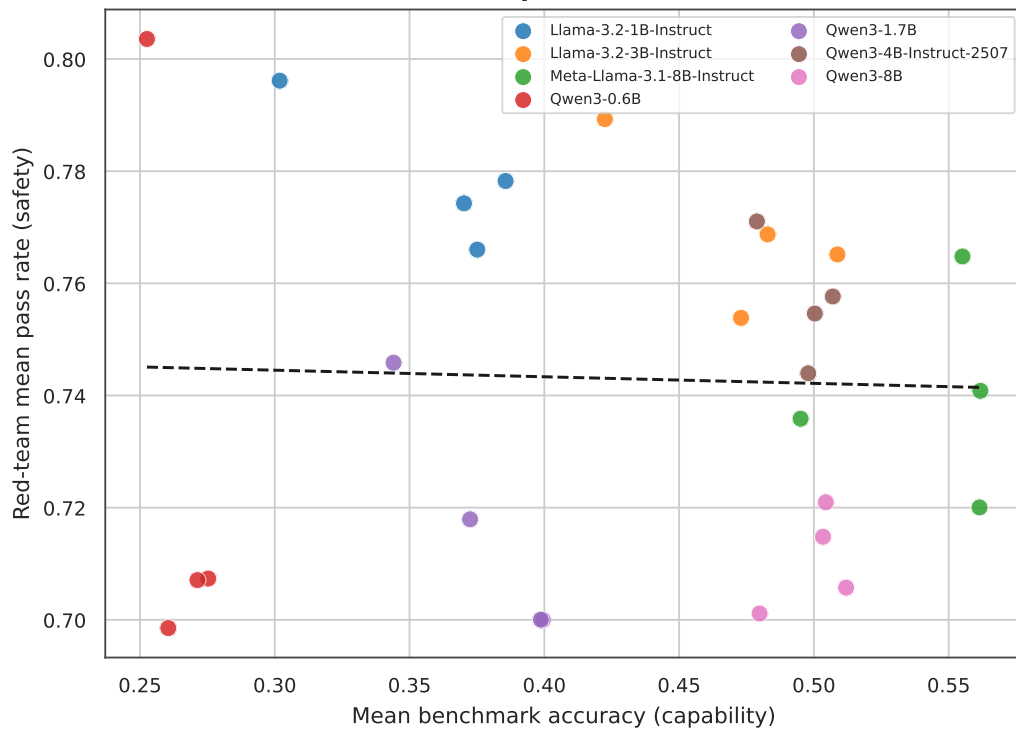


Figure 6.14: Capability change versus adversarial robustness change under compression (decoupling).

6.3.6 Damage localizes by depth: capability at the output, toxicity in mid-network

Complementing the summary views, this section explores additional metrics previously analyzed, such as deltas, means, and group means, to explore and identify effects that summary views cannot capture. Two questions are asked: *where* in the network quantization perturbs the structure and *which* specific vulnerabilities move with that perturbation.

6.3.6.1 Depth-positional structure metrics

Decomposing $\Delta\alpha$ by the depth position of the layer (Table 6.27, Figure 6.15) shows that the small spectral shift is *output-weighted in magnitude*: the mean absolute change grows from ≈ 0.035 at the first layer to ≈ 0.305 at the last, an order of magnitude larger at the head (an MLP layer) than at the first analyzed input layer (an attention layer), while entropy barely moves at any depth. Sign and magnitude must be read separately here. The signed column shows a coherent *upward* drift through the network interior, peaking at the mid-deep quintile ($\Delta\alpha = +0.098$ at Q_4 , with 90% of variants rising), and a large *downward* excursion at the output (-0.255 at the last layer, median -0.032 , only 38% rising) that is dominated by one variant (Llama-3.2-1B at Q2_K_L, last-layer α $8.98 \rightarrow 4.07$). Quantization error therefore concentrates in the output-adjacent layers in magnitude — an effect entirely averaged away, and thus obscured, in the whole-model α — while the direction of that concentration is not consistent across the cohort.

Table 6.27: Depth-positional structural shift. Mean Δ (and mean $|\Delta\alpha|$) by layer-depth position, pooled over compressed states; positions run from the first layer through five equal-depth quintiles to the last layer.

position	Δ alpha	$ \Delta \alpha $	Δ entropy	Δ log_spectral_norm
first	-0.004	0.035	0.006	-0.011
first-2	0.010	0.024	0.006	-0.011
Q1	0.041	0.051	0.001	-0.009
Q2	0.035	0.096	0.001	-0.009
Q3	0.086	0.108	0.001	-0.009
Q4	0.098	0.104	0.001	-0.010
Q5	-0.010	0.106	0.001	-0.010
last-2	-0.101	0.151	0.001	-0.011
last	-0.255	0.305	0.000	-0.005

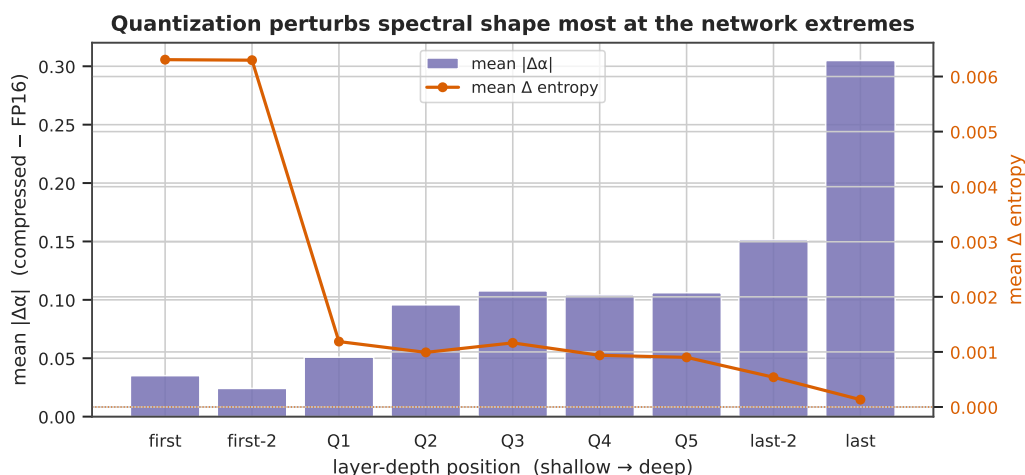


Figure 6.15: Mean $|\Delta\alpha|$ by depth position (first two layers, five equal-depth quintiles Q_1 – Q_5 shallow to deep, last two layers) with mean Δ entropy overlaid; the small quantization shift in spectral shape is output-weighted, growing from the first layer to the last.

6.3.6.2 Does depth-wise perturbation predict outcome change?

Each depth's $\Delta\alpha$ is next correlated against the compressed-minus-FP16 change in accuracy, toxigen-safety and perplexity (Table 6.28, Figure 6.16; depth is summarized as the first two layers, five equal-depth quintiles Q_1 – Q_5 , and the last two layers). A *depth asymmetry* emerges: **toxicity-safety degradation couples to mid-network spectral perturbation, strongest at Q_4** ($r = -0.69$), the single cell of the twenty-seven to survive joint BH-FDR ($q = 0.01$). The mechanism is coherent, since Q_4 is also where the signed $\Delta\alpha$ is most consistent across the cohort ($+0.098$, 90% of variants rising). Accuracy change shows its largest coefficients at the *output* end ($r = +0.53$ at the last layer), but that cell does *not* survive correction ($q = 0.08$) and rests on the outlier-driven last-layer perturbation documented above, so the head-versus-interior contrast is reported as an exploratory pattern rather than an established localization. The pattern is also coefficient-dependent: under Spearman the Q_4 toxigen cell weakens to $\rho = -0.60$ ($q = 0.05$) and *no* cell of the twenty-seven survives, so even the surviving association is borderline and is anchored by the significant *global* $\Delta\alpha \leftrightarrow \Delta$ toxigen relationship rather than standing alone.¹

¹The correction spans all $9 \times 3 = 27$ cells jointly as one screen. A per-column correction would be less conservative; the Benjamini–Hochberg procedure remains valid under the positive dependence characteristic of adjacent depth bins.

Table 6.28: Depth-resolved structure $\leftrightarrow$ outcome coupling. The layer stack of each model is summarized at nine depth positions, from superficial to deep positions: the first two layers (*first*, *first-2*), five equal-depth *quintiles* Q_1 – Q_5 (each a fifth of the layers), and the last two layers (*last-2*, *last*). For each position, correlation is analyzed on the compressed-minus-FP16 change in α there against Δacc , $\Delta\text{toxigen}$, and $\Delta\text{perplexity}$ – the compressed-minus-FP16 variant’s changes in core-task accuracy, the toxigen safety score (higher is safer), and wikitext perplexity (lower is better), respectively (compressed states, $n = 21$). Stars use BH-FDR q across all $9 \times 3 = 27$ cells. Capability change is correlated with the *last* layer, whereas toxicity-safety change correlates with *early* or *middle* layers.

Depth (superficial to deep)	r vs ΔAcc	r vs $\Delta\text{Toxigen}$	r vs $\Delta\text{Perplexity}$
first	-0.02	-0.19	+0.28
first-2	-0.54	-0.38	+0.46
Q1	-0.52	-0.06	+0.36
Q2	-0.11	+0.00	-0.07
Q3	-0.35	-0.41	+0.41
Q4	-0.46	-0.69*	+0.03
Q5	+0.20	-0.05	-0.51
last-2	+0.51	+0.41	-0.32
last	+0.53	+0.42	-0.33

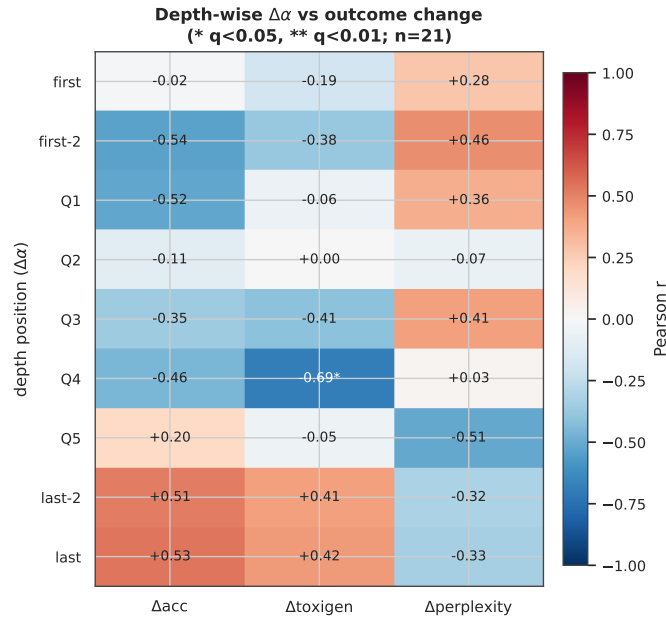


Figure 6.16: Depth-wise $\Delta\alpha$ versus outcome change, across the nine depth positions of Table 6.28 (Q_1 – Q_5 are the five depth quintiles). Δacc , $\Delta\text{toxigen}$, and $\Delta\text{perplexity}$ are the compression-induced changes in accuracy, safety, and perplexity. Toxigen-safety degradation correlates with early/middle-layer spectral perturbation (strong negative column), with accuracy changes correlated in the last layer, while perplexity is mildly correlated at depth — different outcomes localize to different depths.

6.3.7 Which structural changes track which vulnerabilities

6.3.7.1 Can specific attacks be associated with structural change?

The pooled invariance observed in the experiments for compression masks movement at the level of individual attacks. By correlating each probe’s pass-rate change against three global structural changes – $\Delta\alpha$, $\Delta\alpha_{\text{weighted}}$ and $\Delta\text{rand_distance}$ – across family $\times$ level cells, and reporting for every probe the Spearman ρ , its raw p with it’s q formed jointly across all $50 \times 3 = 150$ probe $\times$ metric tests (Table 6.29), can isolate the attacks whose robustness changes with some types of spectral change.

The most structure-associated probes are dominated by encoding- and obfuscation-based prompt injections (e.g. Nato/Morse, Zalgorobes, with for example ANSI-escape detectors), and they correlate far more to $\Delta\text{rand_distance}$ — several with $\rho \approx -0.7$ — than to $\Delta\alpha$ or $\Delta\alpha_{\text{weighted}}$: families whose some spectral properties are pushed further toward randomness (lower values of rand_distance) by quantization seems to lose more adversarial robustness to exactly these attacks. The joint BH-FDR correction – the Nato-encoding injection that has two detectors, and the ANSI-escape probe, all associated with $\Delta\text{rand_distance}$ ($\rho = -0.71, -0.70, -0.67$; $q \approx 0.04$) — so for these specific attacks the rand_distance correlation indicates that is statistically robust, to multiple comparisons rather than merely suggestive (on $n = 21$ cells); the remaining possible correlations do not survive correction and are read as a future consistent exploratory pattern, for a larger evaluated sample of models and compressed states.

The observed probe-specific correlation is the only evidence used here to **show that specific attacks can be associated with mode’s structure**; but within OWASP category aggregation do not. Although the table lists only the ten strongest correlations, the complete picture across all probes is shown in Figure B.1 in Appendix B: the strong negative correlations are confined to the $\Delta\text{rand_distance}$ column and to the encoding/obfuscation injection probes, and the correlations with $\Delta\alpha$ are weak across the whole set, confirming that the effect is attack and metric-specific rather than a property of the entire probe tests.

Table 6.29: Vulnerability and weights structure coupling, per garak probe. Spearman correlation between each probe’s pass-rate change and three global structural changes – $\Delta\alpha$, $\Delta\alpha_{\text{weighted}}$ ($= \Delta\alpha_w$) and $\Delta\text{rand_distance}$ ($= \Delta\text{rd}$) – across family $\times$ level cells; surfaces specific attacks whose robustness tracks each kind of spectral change. Rows are the ten probes with the strongest coupling to any of the three metrics. The encoding/obfuscation injections couple most strongly to $\Delta\text{rand_distance}$ rather than to plain $\Delta\alpha$.

OWASP	Probe · Detector	$\rho(\Delta\alpha)$	$\rho(\Delta\alpha_w)$	$\rho(\Delta\text{rd})$	best	p	q (FDR)	n
llm01	enc.InjectNato · enc.DecodeApprox	+0.40	+0.18	-0.71	Δrd	0.000	0.035	21
llm01	ansi.AnsiEscaped · ansi.Escaped	+0.29	+0.10	-0.70	Δrd	0.000	0.035	21
llm01	enc.InjectNato · enc.DecodeMatch	+0.28	+0.04	-0.67	Δrd	0.001	0.044	21
llm01	ansi.AnsiRaw · ansi.Raw	+0.23	-0.04	-0.57	Δrd	0.007	0.241	21
llm01	LL.LatentInjectionReport · TriggerList	+0.31	-0.01	-0.56	Δrd	0.009	0.241	21
llm01	dan.Ablation-DAN-11 · mit.MitigationBypass	+0.03	+0.17	+0.54	Δrd	0.012	0.241	21
llm01	enc.InjectBase64 · enc.DecodeMatch	+0.05	-0.26	-0.53	Δrd	0.014	0.241	21
llm01	LL.LatentInjectionTranslationEnFr · TriggerList	-0.33	-0.53	-0.04	$\Delta\alpha_w$	0.014	0.241	21
llm01	sata.MLM · mit.MitigationBypass	-0.51	-0.39	+0.10	$\Delta\alpha$	0.017	0.241	21
llm01	enc.InjectZalgo · enc.DecodeMatch	+0.20	-0.06	-0.51	Δrd	0.018	0.241	21

6.3.7.2 How well can weight structure predict adversarial robustness?

Beyond the individual probes, how well do the structural metrics index *overall* adversarial robustness, and does that indexing survive compression? Table 6.30 and Figure 6.17 report the same three metrics in two regimes, giving the coefficient of determination ($R^2 = r^2$) alongside each correlation. Cross-sectionally, across all 28 states, all three index robustness at comparable, moderate strength: higher α and α_{weighted} accompany *lower* robustness ($r = -0.42$ and -0.51 ; $R^2 = 0.17$ and 0.26), while higher `rand_distance` accompanies *higher* robustness ($r = +0.43$), all three significant after correction.

Under compression — correlating the change in each metric with the change in robustness across the 21 compressed states — the picture narrows. The change in α is uninformative ($r = -0.04$, n.s.) and the change in α_{weighted} is likewise not significant ($r = -0.33$, $p = 0.14$); **only the change in `rand_distance` tracks the loss of robustness** ($r = -0.47$, $p = 0.030$, $q = 0.049$). Most informative is that `rand_distance` *reverses sign* between the regimes: **a model that is natively further from random is more robust, yet compression that pushes a model toward randomness costs robustness**. The practical lesson is that the structural quantity worth watching depends on the question — `rand_distance`, not α or α_{weighted} , is the metric sensitive to compression-induced robustness change — and that **no single metric exceeds about a quarter of the variance**, so structure is a **moderate rather than a deterministic** predictor of robustness. Two cautions bound the reading. First, at $n = 21$ the detection threshold is $|r| \approx 0.43$, so effects smaller than that are undetectable here rather than demonstrably absent. Second, the same three compression cells appear in the wider 36-cell screen of Table 6.34 with identical coefficients but without the `rand_distance` star, because a larger correction family sets a stricter bar; the coefficient, not the star, is the transportable quantity.

Table 6.30: How well does weight structure predict adversarial robustness? Each structural metric — α , α_{weighted} , `rand_distance` — is correlated with the overall red-team pass rate in two regimes: cross-sectionally across all variants (*level*, $n = 28$: metric versus pass) and under compression (*compression*, $n = 21$: Δ metric versus Δ pass, that is, the quantization-degradation view). For each test the table gives Pearson r , R^2 , the raw two-sided p , and the Benjamini–Hochberg FDR-adjusted q correcting the six tests together as one family; the significance flag is on q ($*q < 0.05$, $**q < 0.01$, n.s. otherwise). After correction all three metrics still index robustness cross-sectionally ($q \leq 0.05$); under compression *only* Δ `rand_distance` survives FDR ($p = 0.030$, $q = 0.049$), while $\Delta\alpha_{\text{weighted}}$ and $\Delta\alpha$ are null. The contrast is a sample-size as much as an effect-size statement: at $n = 21$ the two-sided threshold for $p < 0.05$ is $|r| \approx 0.43$, which `rand_distance`’s $|r| = 0.47$ clears and α_{weighted} ’s $|r| = 0.33$ does not. Critically, `rand_distance` *reverses sign* between the two regimes. The compression column of this table is the six-test-family view of the same three cells that appear, under the wider 36-cell screen, in Table 6.34; the coefficients agree and only the stars differ, because the correction family differs.

metric	regime	n	r	R^2	p (raw)	q (FDR)
alpha	level	28	−0.417	0.174	0.027	0.049*
alpha	compression	21	−0.042	0.002	0.857	0.857 n.s.
alpha_weighted	level	28	−0.510	0.260	0.006	0.033*
alpha_weighted	compression	21	−0.333	0.111	0.141	0.169 n.s.
rand_distance	level	28	+0.432	0.186	0.022	0.049*
rand_distance	compression	21	−0.474	0.225	0.030	0.049*

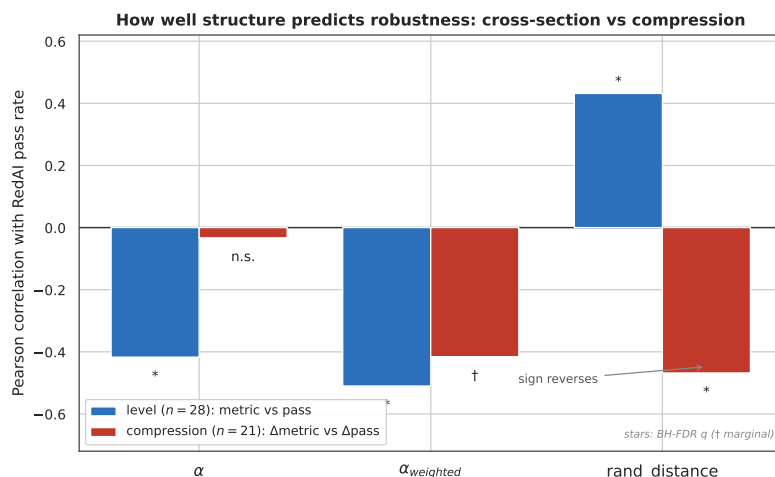


Figure 6.17: Pearson correlation of each structural metric with the RedAI pass rate, cross-sectionally (level, blue) and under compression (Δ vs FP16, red), with significance stars. All three metrics predict robustness cross-sectionally; under compression, the change in plain α is essentially zero, whereas the changes in α_{weighted} and rand_distance track robustness; rand_distance is positively related to robustness across models, but its compression-induced *increase* costs robustness, so its sign reverses between regimes.

6.3.8 Pareto-optimal precision and the deployment decision

With all three perspectives reported and their cross-relationships established, the evidence can now be converted into a precision recommendation. Taken together (Table 6.31, Figure 6.18), 4-bit Q4_K_M is Pareto-optimal [158]: **retains almost all capability performance**, leaves aggregate **adversarial robustness intact** (even that specific probes have degradation), aligns with empirical scaling laws established for low-precision capability performance quantization regimes [159, 99, 137], and keeps every family well inside the well-trained spectral band [29, 30]: [2, 6]. The 3-bit is an acceptable trade-off, but the 2-bit has severe capability loss for small models and should be avoided there.

Two prominent points frame these recommendations. First, adversarial robustness behaves as a *distinct axis* from capability [136]: the two are statistically decoupled, so a model that loses accuracy need not lose (or keep) jailbreak robustness, as shown by the ChaosAthena Framework experiments, specific red-team AI garak probes, and corresponding detectors. Second, data-free structural metrics *index cross-sectional quality* – spectral entropy can also track accuracy across models, aligned with the HTSR and SETOL [29, 30] — but, at least at the global level measurements (means), do *not forecast per-model compression damage*: in the structure metrics taken into account, knowing a model’s spectrum tells you how good it is, not how much quantizing it will cost. **The one structural signal that moves with a compression outcome is the loss of intrinsic toxicity-safety**: of the two safety axes (Table 6.33), only the toxigen clearly degrades, at a global level, under quantization **and the loss couples with the spectral change** of interior layer depth, localized in the **fourth depth quintile (mid-network)**, with $r = -0.69, q = 0.01$ in section 6.3.6.

Table 6.31: Decision summary per quantization level: pooled accuracy and pass-rate change, structural verdict, safety tier, and deployment recommendation.

	Mean Δ acc (pp)	Mean Δ pass-rate	α verdict	Safety tier	Recommendation
Q4	-0.565	-0.006	remains in band [2,6] (4.71–5.22)	unchanged	Preferred (Pareto-optimal)
Q3	-1.538	0.001	remains in band [2,6] (4.69–5.28)	unchanged	Acceptable
Q2	-5.236	0.026	remains in band [2,6] (4.72–5.31)	unchanged	Avoid for small models

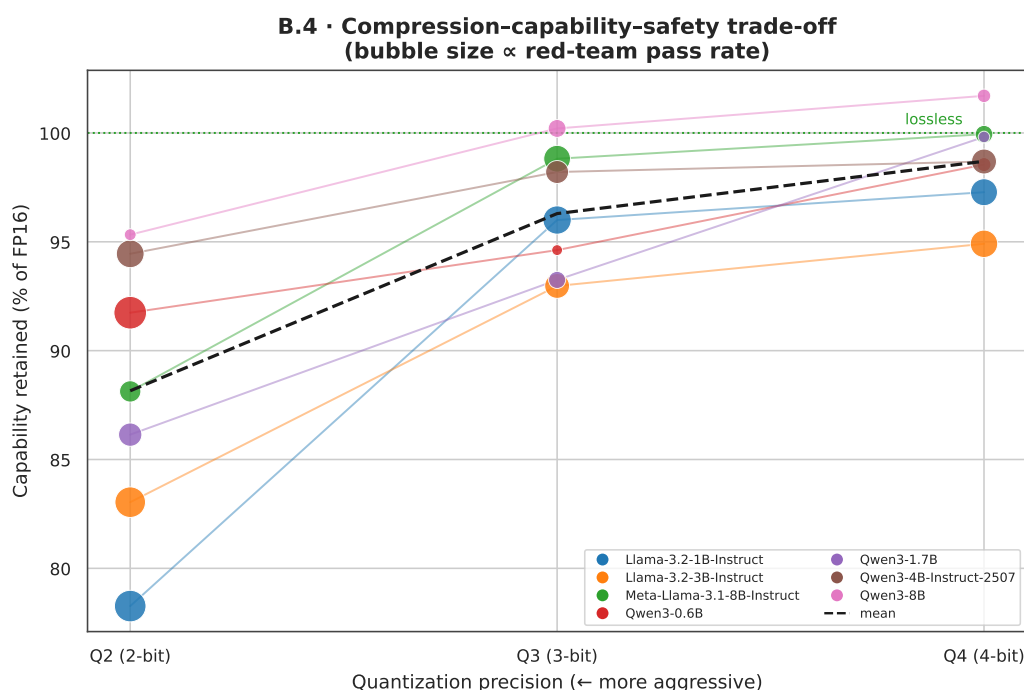


Figure 6.18: Capability $\times$ robustness Pareto view across levels.

6.3.9 Exploratory results

The results reported so far answer the questions the experiments were designed around. The Multi-Perspective Dataset produced by Phase 3 supports a far wider space of questions than those, and this section reports a systematic sweep of it. Everything in it carries a different evidential status from the preceding sections, and that status is stated here once so that no result below is read for more than it supports.

These results are exploratory. This word is used in a precise sense, and it carries three commitments. First, *they are verified*: every value was recomputed from the source datasets, every screen states the Benjamini–Hochberg family it is corrected within, and no result is described as significant on a raw p alone [77, 133]. A result reported here as supported did survive correction within its stated family; that is a fact about these data and it is not in doubt. Second, *they are hypothesis-generating rather than confirmatory*. Multiplicity control bounds the false-discovery rate of a screen; it does not confer on a screen the evidential standing of a pre-registered confirmatory test, and the distinction is not rhetorical. The one partial exception is the hypothesis battery of Section 6.3.9.1, whose twenty claims were fixed before their

coefficients were computed — which is why it is reported first and why its failures are given the same weight as its successes. Third, *they are bounded by the cohort*. At $n = 28$ for level statistics and $n = 21$ for compression deltas the detection thresholds are $|r| \approx 0.37$ and $|r| \approx 0.43$, so every null below means undetectable here rather than demonstrably absent; near-binary predictors rest on splits a differently composed cohort could dissolve; and one quantization format, one architecture class, and seven model families bound external validity throughout (Section 7.2).

The consequence for how these results should be used is specific rather than generic. They are **strong enough to direct further work and to be reported as findings of this cohort**, and they are **not strong enough to be asserted as general properties of compression or to be deployed as decision procedures**. Each subsection below therefore states what would settle its result, and Section 6.4.8 consolidates those settling experiments. Where a result is unusually well supported — the pre-specified battery, the provenance signature — the text says so; where it is fragile, coefficient-dependent, or outlier-driven, the text says that instead, in the same paragraph as the result rather than in a later caveat.

6.3.9.1 A pre-specified hypothesis battery

Twenty hypotheses were specified before the corresponding correlations were computed, each a single directional claim linking one structural quantity to one behavioral outcome, and all twenty were corrected together as one Benjamini–Hochberg family (Table 6.32, Figure 6.19). Pre-specifying the battery and correcting it jointly is what separates this from selecting the surviving correlations after the fact; the failures are therefore reported alongside the successes and carry the same evidential weight.

The tally is **eight supported**, two suggestive, one significant in the direction *opposite* to the conjecture, three untestable because the predictor is near-constant in this cohort, and six not supported. The supported set is coherent rather than scattered. Three hypotheses concern the spectral norm and its expansion through the network: a larger first-to-last expansion accompanies lower overall robustness (H2, $r = -0.56$, $q = 0.008$) and more prompt-injection vulnerability (H3, $r = -0.44$, $q = 0.040$), and a high final-layer log-norm accompanies weaker output handling (H4, $r = -0.57$, $q = 0.008$).² Two concern brittleness counters: more trap layers accompany lower LLM01 robustness (H7, $r = -0.56$, $q = 0.008$) and more undertrained layers lower robustness overall (H6, $r = -0.45$, $q = 0.034$). Three concern compression itself, and are the only members of the battery that speak to what quantization *does* rather than to how models differ: the change in log-spectral-norm tracks the change in robustness (H15, $r = -0.59$, $q = 0.013$) and in perplexity (H16, $r = -0.63$, $q = 0.008$), and the change in rand_distance tracks the change in accuracy (H18, $r = +0.84$, $q < 0.001$) — the strongest single association in this dissertation.

Three failure modes are worth naming because they recur in screens of this kind. H10 is *reversed*: greater layer-wise dispersion of α_{weighted} was conjectured to lower accuracy, and it is significantly associated with *higher* accuracy ($r = +0.53$, $q = 0.011$); a significant result in the wrong direction refutes the conjecture rather than supporting it, and is recorded as such. H5, H8 and H19 are *degenerate*: the predictor takes essentially one value across the cohort, so the correlation is carried by a handful of points and the hypothesis is untestable here rather than false — H19 in particular ($r = +0.73$, $q < 0.001$) is driven entirely by one high-leverage model, the other 27 sitting at zero overtrained layers (Figure 6.26). H20, that larger models are more vulnerable, is simply not supported ($r = -0.16$, $p = 0.430$ on a log-parameter scale); size and

²H4 is computed on the log *Frobenius* norm of the final layer (`log_norm [last]`), not the log spectral norm; the two correlate at 0.89 but are distinct quantities (`log_norm[last]` mean 3.96, range 3.35–4.50; `log_spectral_norm[last]` mean 1.55, range 1.15–2.20). On `log_spectral_norm[last]` the association strengthens to $r = -0.70$ ($p < 0.001$) and the verdict is unchanged. H2, H3, H12, H15 and H16 all use `log_spectral_norm`; the dataset contains no raw, non-logarithmic spectral norm.

adversarial robustness are unrelated across this cohort, which is consistent with the capability–robustness decoupling of Section 6.3.5.³

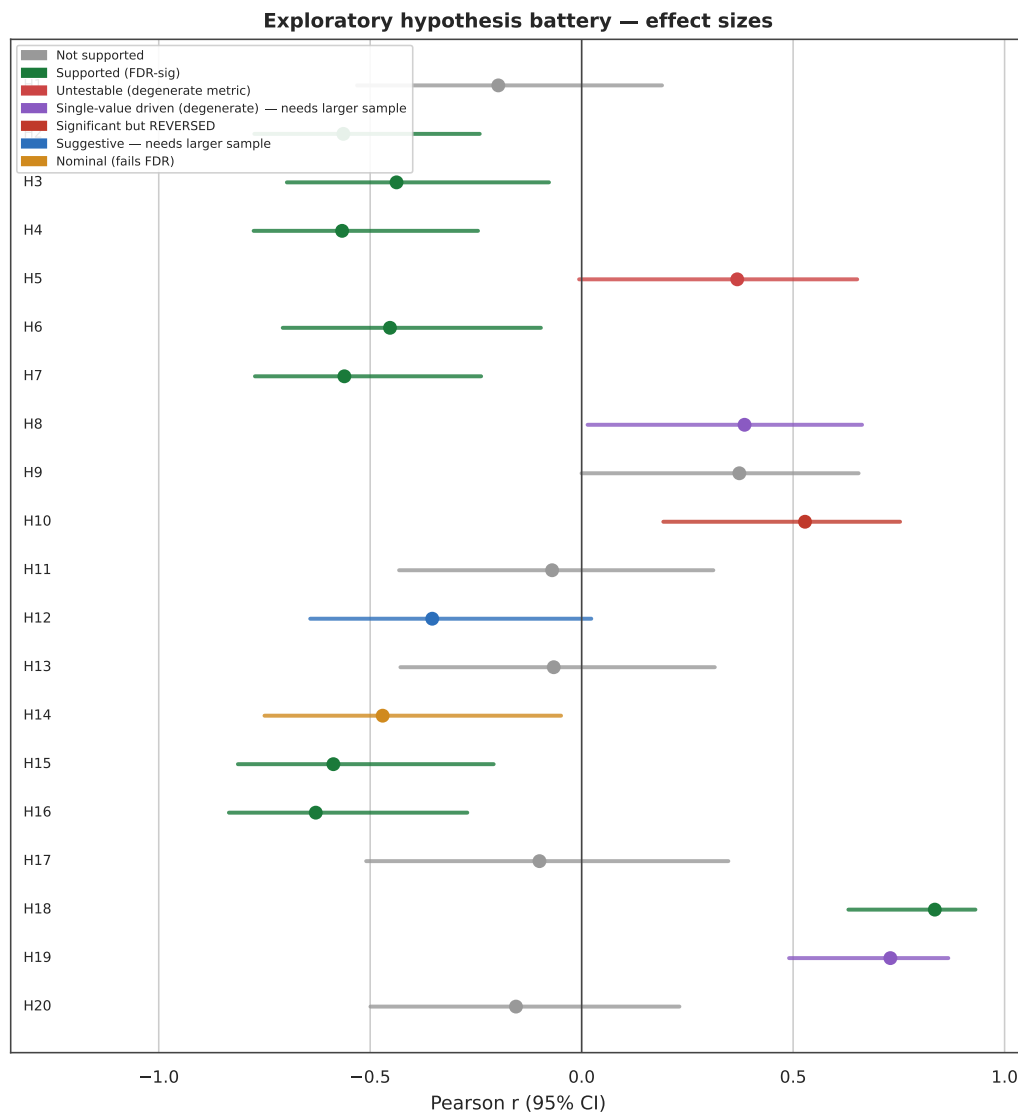


Figure 6.19: Effect sizes for the twenty pre-specified exploratory hypotheses, each as Pearson r with its Fisher 95% confidence interval, ordered by magnitude ($n = 28$ for level tests, $n = 21$ for compression-delta tests). Intervals excluding zero after joint BH-FDR correction over the twenty are the supported set — the bold rows of Table 6.32. Intervals straddling zero are shown in full rather than omitted, because at this cohort size the *width* of the interval is what distinguishes a small effect from an undetectable one, and that distinction is the honest content of a negative result here.

³H20 is computed against $\log_{10}(\text{parameters})$; on a raw parameter scale the coefficient is -0.26 ($p = 0.190$). Neither is significant, but the scale is stated because the two differ visibly.

Table 6.32: **Exploratory** pre-specified hypothesis battery. Twenty directional claims were fixed *before* their correlations were computed and are corrected together as a **single Benjamini–Hochberg family of 20 tests**, so the surviving set is not a post-hoc selection. Each row gives Pearson r , its Fisher 95% CI, the raw two-sided p , the FDR-adjusted q , and the n on which it rests — $n = 28$ for level tests, $n = 21$ for compression-delta tests. **Only rows with $q < 0.05$ carry a bold coefficient**, a direction arrow and a star ($*q < 0.05$, $**q < 0.01$). Verdicts: *Supported* = survives FDR; *Suggestive* = directionally consistent, fails FDR, needs a larger cohort; *Reversed* = significant in the direction opposite to the conjecture, hence rejected; *Degenerate/Unstable* = the predictor is near-constant in this cohort, so the hypothesis cannot be tested here rather than being false. All results in this table are **exploratory** (Section 6.3.9). [†]H4 uses the log *Frobenius* norm `log_norm[last]`; on `log_spectral_norm[last]` the association strengthens to $r = -0.70$ ($p < 0.001$) and the verdict is unchanged. H20 is computed on a $\log_{10}$ -parameter scale.

ID	Exploratory hypothesis	r	95% CI	p	q (FDR)	n	Verdict
H1	Entropy drop first→last lowers toxigen safety	+0.20	[-0.19, +0.53]	0.315	0.394	28	Not supported
H2	Spectral-norm expansion first→last lowers overall robustness	-0.56** ↓	[-0.77, -0.24]	0.002	0.008	28	Supported
H3	Spectral-norm expansion increases prompt-injection (LLM01) vulnerability	-0.44* ↓	[-0.70, -0.08]	0.020	0.040	28	Supported
H4	High final-layer log-norm lowers output-handling pass [†]	-0.57** ↓	[-0.78, -0.25]	0.002	0.008	28	Supported
H5	Overtrained/undertrained ratio lowers robustness	+0.37	[-0.00, +0.65]	0.054	0.077	28	Unstable (degenerate)
H6	More undertrained layers lower robustness	-0.45* ↓	[-0.71, -0.10]	0.015	0.034	28	Supported
H7	More trap layers lower prompt-injection (LLM01) robustness	-0.56** ↓	[-0.77, -0.24]	0.002	0.008	28	Supported
H8	Worst-case z magnitude tracks overtrained/undertrained ratio	+0.39	[+0.02, +0.67]	0.043	0.072	28	Degenerate (one model)
H9	Higher layer-wise α dispersion lowers accuracy	+0.37	[-0.00, +0.65]	0.051	0.077	28	Not supported
H10	Higher layer-wise α_{weighted} dispersion lowers accuracy	+0.53* ↑	[+0.20, +0.75]	0.004	0.011	28	Reversed — rejected
H11	Higher task-accuracy dispersion lowers robustness	-0.07	[-0.43, +0.31]	0.724	0.739	28	Not supported
H12	Higher mean spectral norm tracks lower robustness	-0.35	[-0.64, +0.02]	0.065	0.087	28	Suggestive
H13	More spikes track lower robustness	-0.07	[-0.43, +0.31]	0.739	0.739	28	Not supported
H14	Entropy change tracks toxigen-safety change	-0.47	[-0.75, -0.05]	0.031	0.056	21	Suggestive (nominal only)
H15	Spectral-norm change tracks robustness change	-0.59* ↓	[-0.81, -0.21]	0.005	0.013	21	Supported
H16	Spectral-norm change tracks perplexity change	-0.63** ↓	[-0.83, -0.27]	0.002	0.008	21	Supported
H17	α_{weighted} change tracks accuracy change	-0.10	[-0.51, +0.35]	0.667	0.739	21	Not supported
H18	rand_distance change tracks accuracy change	+0.84** ↑	[+0.63, +0.93]	<0.001	<0.001	21	Supported
H19	More overtrained layers raise perplexity	+0.73** ↑	[+0.49, +0.87]	<0.001	<0.001	28	Degenerate (one model)
H20	Larger models are more vulnerable to red-team probes	-0.16	[-0.50, +0.23]	0.430	0.506	28	Not supported

6.3.9.2 A comprehensive structure-to-behavior association screen

. The battery tests specific claims. To check that those claims are not artifacts of which relationships were chosen, every HTSR and SETOL quantity available — including the sparse layer counters and `rand_distance`, not only the three headline shape metrics — is crossed with every behavioral outcome, in both regimes (Tables 6.33 and 6.34, Figure 6.20). Each screen is corrected as one family, so the stars here are stricter than in any narrower table of the same cells.

The level screen reproduces the double dissociation of Section 6.3.4 at full width and shows it to be a property of blocks of metrics rather than of individual choices. A **spectral-richness block** — `entropy`, `stable_rank`, `num_spikes`, `num_pl_spikes` — indexes capability very strongly ($r = +0.87$ to $+0.94$ against accuracy, all $q < 0.01$) and toxicity-safety with it ($+0.76$ to $+0.85$), while showing no association whatever with adversarial robustness ($|r| \leq 0.21$). A **brittleness block** — `ntrap`, `nundertrained` — does the reverse, indexing robustness (-0.57^{**} and -0.45^*) and prompt-injection robustness specifically (-0.56^{**} , -0.35) while showing nothing against accuracy ($|r| \leq 0.08$). Neither block leaks into the other's column. The delta screen is narrower and points in one direction: $\Delta\text{entropy}$, $\Delta\log_spec_norm$ and $\Delta\text{rand_distance}$ are the structural footprint of capability loss ($|r| = 0.56$ to 0.86 against $\Delta\text{accuracy}$ and $\Delta\text{perplexity}$, $q < 0.05$), and outside those three columns almost nothing survives. Compression, in other words, is legible in the spectrum when it costs capability and largely illegible when it costs safety — which is the same asymmetry the depth analysis of Section 6.3.6 localizes to Q_4 .

Table 6.33: Comprehensive structure→behavior screen, *levels*. All $n = 28$ **variants** (7 families $\times$ 4 precisions); each cell is the Pearson r between the metric level and the outcome level. Benjamini–Hochberg FDR is applied across the **whole 55-cell screen** (11 metrics $\times$ 5 outcomes). **Only cells with $q < 0.05$ are set in bold**, with a direction arrow ($\uparrow$ the outcome rises with the metric, $\downarrow$ it falls) and a star ($*q < 0.05$, $**q < 0.01$); every other cell is reported in full but is *not* distinguishable from chance at this sample size. Exact r , 95% CI, p and q for every bold cell are in Table 6.12. Metrics are grouped to expose the double dissociation: spectral richness indexes capability, the brittleness counters index adversarial robustness, and neither block leaks into the other’s column. $n_{\text{overtrained}}$ is degenerate in this cohort (one model carries the entire range) and its row is reported for completeness only.

Structural metric (level, $n = 28$)	Accuracy	Perplexity	Toxigen	RedAI pass	LLM01 pass
<i>Spectral richness</i>					
entropy	+0.91**$\uparrow$	-0.76**$\downarrow$	+0.82**$\uparrow$	-0.14	-0.17
stable_rank	+0.90**$\uparrow$	-0.71**$\downarrow$	+0.85**$\uparrow$	-0.21	-0.21
num_spikes	+0.94**$\uparrow$	-0.79**$\downarrow$	+0.83**$\uparrow$	-0.07	-0.06
num_pl_spikes	+0.87**$\uparrow$	-0.75**$\downarrow$	+0.76**$\uparrow$	+0.01	-0.07
<i>Brittleness counters</i>					
n_{trap}	+0.08	-0.11	+0.22	-0.57**$\downarrow$	-0.56**$\downarrow$
$n_{\text{undertrained}}$	-0.03	+0.22	+0.22	-0.45*$\downarrow$	-0.35
$n_{\text{overtrained}}$	-0.36	+0.73**$\uparrow$	-0.20	+0.37	+0.35
<i>Shape and norm</i>					
α	-0.27	+0.39	+0.01	-0.42	-0.33
α_{weighted}	+0.32	-0.21	+0.32	-0.51*$\downarrow$	-0.43
log_spectral_norm	+0.52*$\uparrow$	-0.45*$\downarrow$	+0.41	-0.35	-0.29
rand_distance	-0.05	-0.19	-0.31	+0.43	+0.37

Table 6.34: Comprehensive structure→behavior screen, *compression changes*. **All** $n = 21$ **compressed variants** (7 families $\times$ 3 precisions); Δ is always compressed — same-family FP16. The *mean Δ metric* column gives the metric’s own average movement under compression (direction only; it is not an input to any correlation). Benjamini–Hochberg FDR is applied across the **whole 36-cell screen**. **Only cells with $q < 0.05$ are set in bold**, with a direction arrow and a star (* $q < 0.05$, ** $q < 0.01$); exact statistics are in Table 6.12. At $n = 21$ the two-sided detection threshold for $p < 0.05$ is $|r| \approx 0.43$, so an unmarked cell means undetectable at this cohort size rather than absent. Polarity must be applied when reading a delta: a positive Δ perplexity is *worse*.

Structural change ($n = 21$)	mean Δ metric	Δ Accuracy	Δ Perplexity	Δ Toxigen	Δ RedAI pass
$\Delta\alpha$	+0.050	-0.38	-0.07	-0.49	-0.04
$\Delta\alpha_{\text{weighted}}$	+0.030	-0.10	-0.43	-0.21	-0.33
Δ entropy	+0.001	-0.86** ↓	+0.60* ↑	-0.47	+0.49
Δ log_spectral_norm	-0.009	+0.72** ↑	-0.63* ↓	+0.42	-0.59* ↓
Δ rand_distance	-0.003	+0.84** ↑	-0.56* ↓	+0.44	-0.47
Δ num_spikes	-4.802	+0.45	-0.04	+0.09	-0.07
Δ num_pl_spikes	-0.062	+0.24	+0.36	+0.33	+0.22
Δn_{trap}	-0.286	-0.09	-0.37	-0.03	-0.45
$\Delta n_{\text{undertrained}}$	+2.762	+0.06	-0.03	-0.29	-0.08
<i>mean Δoutcome</i>	—	-0.024	+3.86	-0.062	+0.007

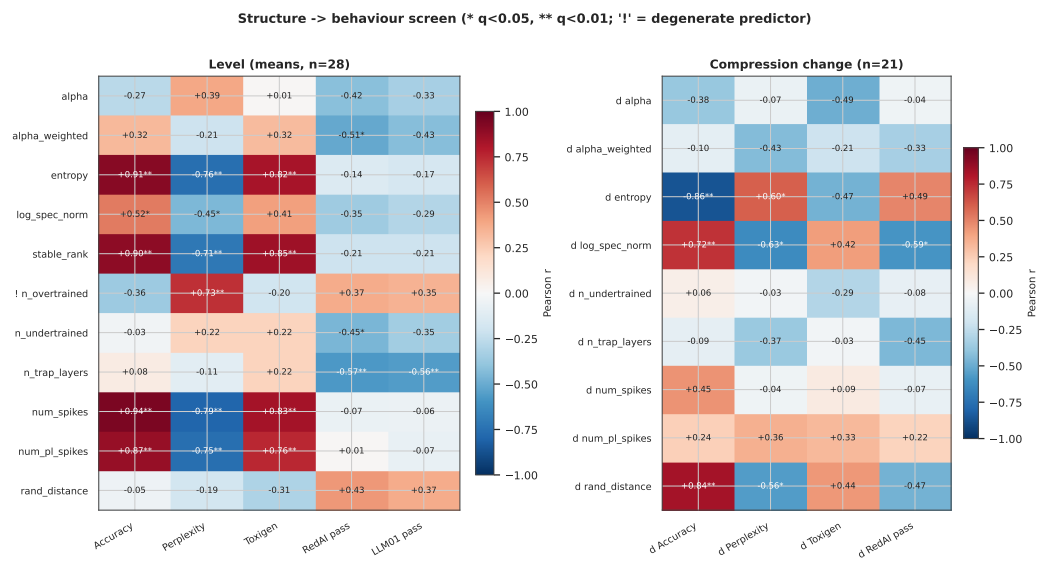


Figure 6.20: The comprehensive screen as heatmaps: metric levels (left, $n = 28$) and compression changes (right, $n = 21$) against the behavioral outcomes. Stars are BH-FDR within each screen; '!' marks a predictor that is degenerate in this cohort. The block structure — spectral richness against capability, brittleness counters against robustness — is visible as two non-overlapping regions.

6.3.9.3 The attack surface resolved by OWASP group and probe

Aggregating pass rates by OWASP category is the natural way to report adversarial results, but in this cohort only one category is statistically substantial. Of the 39 distinct probes, 38 carry an LLM01 label and one carries LLM04; the two ANSI escape-sequence probes additionally carry LLM02 (2023 taxonomy) and LLM05 (2025), so those two columns are the *same two probes* under two names and are not independent evidence [26]. Table 6.35 therefore marks every group but LLM01 as noise-dominated, and the *Overall* column, which correlates with LLM01 at $r \approx 0.97$, closely tracks it. Read with that caveat, the reliable statements are that a larger spectral-norm expansion and a higher α_{weighted} both accompany lower overall robustness ($r = -0.56$ and -0.51) — the same two effects the battery confirmed as H2 and H12.

Because the output-handling signal is carried by only two probes, it is examined directly rather than through its category (Tables 6.36 and 6.37, Figure 6.22). It is the strongest and most internally consistent probe-level result in the experiments. At the level, both ANSI probes lose robustness as α_{weighted} , the spectral norm and the trap-layer count rise (-0.55 to -0.68 , all $q < 0.05$). Under compression the same two probes track $\Delta \log_{\text{spec_norm}}$ at -0.71^{**} and -0.64^{**} and $\Delta \text{rand_distance}$ at -0.69^{**} and -0.59^* , while a rise in entropy tracks a *gain* in robustness ($+0.72^{**}$, $+0.61^*$) — the capability-loss footprint of the previous subsection, appearing here with its sign inverted because these two probes are the ones on which more capable models do worse. The LLM04 Model-DoS probe behaves differently again, associating with α (-0.50^*) and with $\Delta \text{num_pl_spikes}$ (-0.57^*) and with nothing else. Three probes with three distinct structural signatures is a demonstration that **the framework can separate attack families structurally**; it is not yet a demonstration that the separation generalizes, since two of the three rest on a single probe each.

Table 6.35: Per-OWASP pass-rate (level) versus a panel of structural metrics (level), cross-sectionally, with BH-FDR across the whole screen ($*q < 0.05$, $**q < 0.01$, $n = 28$). Groups marked * rest on only 1–2 probes and are noise-dominated; LLM01 (38 probes) is the reliable column. *Overall* is the all-probe mean pass rate, which is dominated by LLM01 (the two correlate at $r \approx 0.97$), so its column closely tracks LLM01: the two starred entries are a larger spectral-norm expansion and a higher α_{weighted} , both predicting lower overall robustness ($r = -0.56$ and -0.51 , the same effects confirmed in the hypothesis battery).

Structural metric (level)	LLM01 (n=38)	LLM02 (n=2)*	LLM04 (n=1)*	LLM05 (n=2)*	Overall
alpha	-0.33	-0.28	-0.50*	-0.28	-0.42
alpha_weighted	-0.43	-0.66**	+0.07	-0.66**	-0.51*
entropy	-0.17	-0.19	+0.39	-0.19	-0.14
log_spec_norm	-0.29	-0.61**	+0.28	-0.61**	-0.35
log_spec_norm(last-first)	-0.44	-0.64**	-0.30	-0.64**	-0.56*
n_undertrained	-0.35	-0.33	-0.56*	-0.33	-0.45
num_spikes	-0.06	-0.19	+0.22	-0.19	-0.07
num_pl_spikes	-0.07	+0.02	+0.47*	+0.02	+0.01
rand_distance	+0.37	+0.27	+0.39	+0.27	+0.43

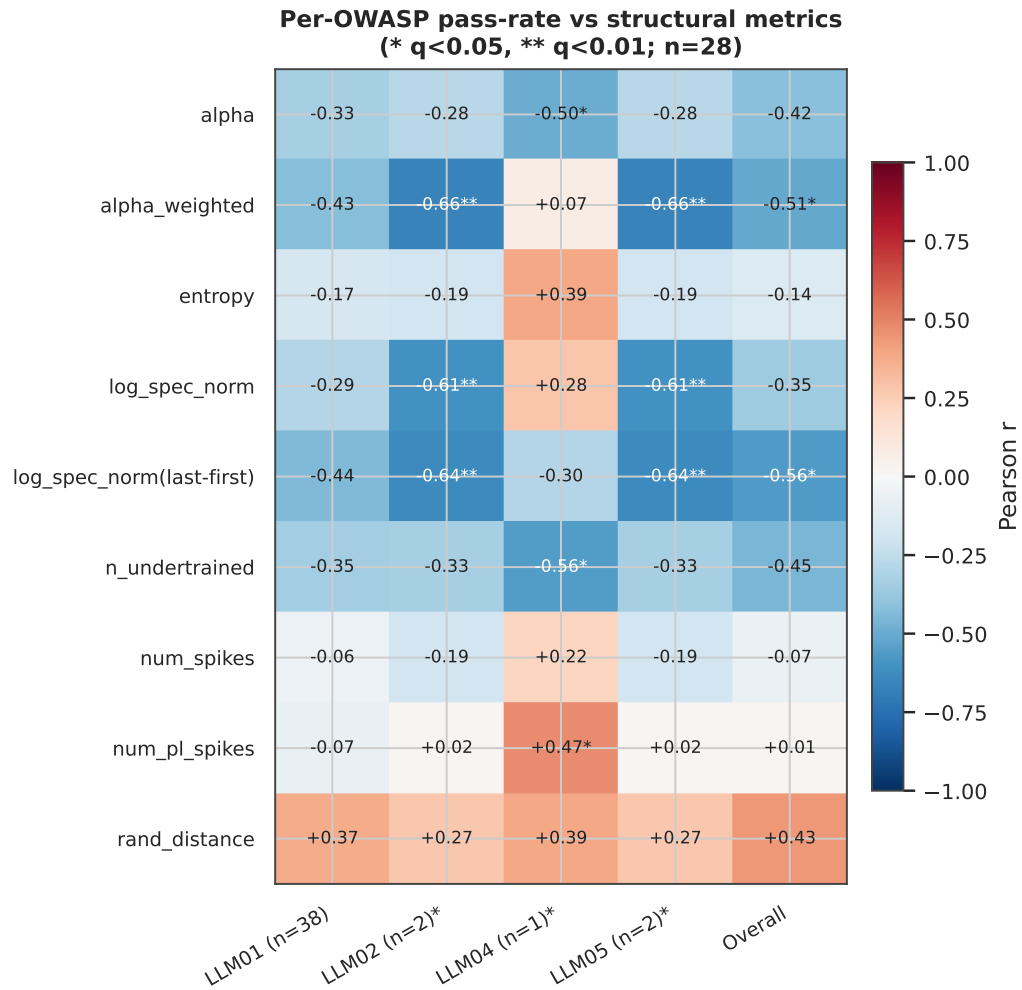


Figure 6.21: Per-OWASP pass rate against structural metrics (Pearson r , BH-FDR stars). LLM01 is the reliable 38-probe column; LLM02, LLM04 and LLM05 rest on one or two probes each and the ANSI pair is common to LLM02 and LLM05.

Table 6.36: Specific LLM02/04/05 sub-probes versus weight structure (pass-rate *level*, $n = 28$, BH-FDR within the screen). RepeatedToken is the Model-DoS probe (LLM04); AnsiEscaped and AnsiRaw are output-handling probes (LLM02 in the 2023 taxonomy, LLM05 in 2025 – carried by the *same* two probes here, so one signal). The output-handling probes lose robustness as the spectral norm, α_{weighted} and trap-layer count rise.

OWASP sub-probe (pass rate, $n=28$)	alpha	alpha_weighted	entropy	log_spec_norm	stable_rank	n_trap_layers	num_spikes	num_pl_spikes	rand_distance
RepeatedToken (LLM04)	-0.50*	+0.07	+0.39	+0.28	+0.25	+0.33	+0.22	+0.47*	+0.39
AnsiEscaped (LLM02/05)	-0.12	-0.57*	-0.37	-0.57*	-0.41	-0.55*	-0.35	-0.22	+0.18
AnsiRaw (LLM02/05)	-0.47*	-0.68**	+0.10	-0.55*	-0.02	-0.57*	+0.09	+0.37	+0.36

Table 6.37: The same sub-probes versus the *compression-induced change* in structure ($n = 21$, BH-FDR within the screen). For the output-handling probes a larger spectral-norm expansion and a larger rand_distance shift track a drop in robustness, while a rise in entropy tracks a gain – the same capability-loss footprint seen elsewhere.

OWASP sub-probe (Δ pass, $n=21$)	d alpha_weighted	d entropy	d log_spec_norm	d n_trap_layers	d num_pl_spikes	d rand_distance
RepeatedToken (LLM04)	+0.26	+0.08	-0.11	+0.43	-0.57*	-0.11
AnsiEscaped (LLM02/05)	-0.27	+0.72**	-0.71**	-0.20	+0.17	-0.69**
AnsiRaw (LLM02/05)	-0.30	+0.61*	-0.64**	-0.39	+0.39	-0.59*

LLM02/04/05 sub-probe robustness vs weight structure (* $q < 0.05$, ** $q < 0.01$)

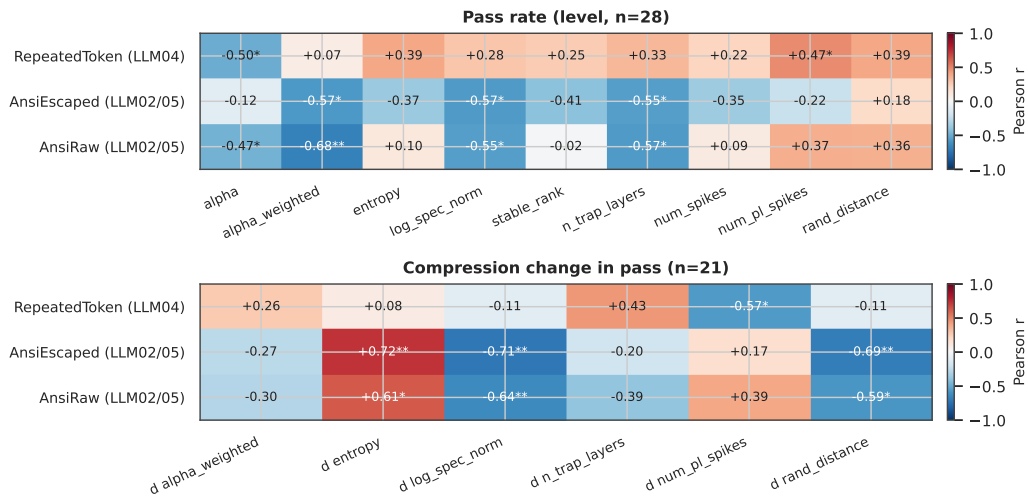


Figure 6.22: The specific LLM02/04/05 probes — RepeatedToken (Model-DoS) and the two ANSI output-handling probes — against weight structure, as pass-rate levels (top, $n = 28$) and compression changes (bottom, $n = 21$). Stars are BH-FDR within each screen.

6.3.9.4 Which structural metrics index which attacks

The previous subsection showed that OWASP categories are too coarse to carry a structural signal in this cohort, since only LLM01 is statistically meaningful, absorbing 38 of the 39 probes. The removal of the aggregation completely reverses that view. Correlating *each distinct attack's* pass rate against *each aggregate structural metric* at the level gives a $36 \times 6 = 216$ -cell screen ($n = 28$; three of the 39 probes are dropped for an invariant pass rate or too few usable variants), of which **69 cells survive joint Benjamini–Hochberg correction** and **26 of the 36 testable probes carry at least one significant structural coupling** (Table 6.38, Figure 6.23). Structure indexes adversarial robustness far more strongly than any aggregate suggests — the aggregate simply averages the signal away.

The organizing principle is the finding, and it is not the one an aggregate analysis would predict. **The sign pattern is a property of the attack family, not of the metric.** Three families separate cleanly and in opposite directions:

- **Jailbreaks (DAN family).** dan.Ablation-DAN-11, dan.AutoDANCached and dan.DanInTheWild couple *negatively* to rand_distance (-0.50 to -0.64 , $q \leq 0.032$) and *positively* to α , α_{weighted} and

the spectral norm ($\alpha_{\text{weighted}} +0.61$ to $+0.72$, $q \leq 0.005$). Models whose spectra sit further from random, and whose exponents are lighter-tailed, resist DAN-style jailbreaks *less* well.

- **Latent and prompt injection.** `LI.TranslationEnFr`, the two `LI.FactSnippet` probes, `LI.Resume`, `PI.HijackHateHumans` and `PI.HijackKillHumans` invert every one of those columns: *rand_distance positive* ($+0.59$ to $+0.74$, $q \leq 0.009$) and α_{weighted} *negative* (-0.62 to -0.71 , $q \leq 0.005$). The strongest single cell in this family, *rand_distance* versus `LI.TranslationEnFr` at $+0.74^{**}$, is among the strongest structural associations in the experiments.
- **Encoding and obfuscation.** `enc.InjectNato`, `enc.InjectZalgo`, `enc.InjectHex` and `enc.InjectBase16` behave unlike either of the above: no shape metric reaches significance for them, and they are indexed instead by the two *richness* quantities — spectral entropy (-0.47 to -0.76) and **the scale metric stable_rank**, which supplies the strongest coupling in the entire screen (`enc.InjectNato` $r = -0.80^{**}$, $\rho = -0.89^{**}$).

That last family qualifies a claim made earlier in this chapter, and the qualification matters. Section 6.3.3 reports that the scale metrics are largely redundant with the shape metrics at the aggregate level and that the analysis therefore concentrates on shape [126]. **At the attack level, that redundancy does not hold:** for the encoding and obfuscation probes, *stable_rank* is the strongest available predictor and no shape metric reaches significance at all. Because *stable_rank* is also nearly collinear with parameter count in this cohort ($r = +0.95$ raw, $+0.99$ on a logarithmic scale; Section 6.3.9.8), the correct interpretation is that *larger, spectrally richer models are markedly worse at resisting encoded and obfuscated injections*, an interpretation to be tested on a cohort in which scale and richness are separable, but one that the aggregate analysis could not have revealed.

A fourth structure completes the view and explains why the aggregate is so uninformative. **Nine probes flip sign between rand_distance and log_spectral_norm** — the whole DAN family, `LI.TranslationEnFr`, `PI.HijackHateHumans`, both `LI.FactSnippet` probes, `ansi.AnsiRaw` and `enc.InjectROT13` — which is coherent, since those two metrics are themselves anti-correlated across the cohort ($r = -0.35$). When families with opposing signs are averaged into one pass rate, they cancel. The practical consequence is methodological and transfers beyond the experiments: **an aggregate adversarial score can be structurally uncoupled, while most of its constituent attacks are strongly and systematically coupled**, so a structure-informed screening procedure must operate per attack family and never on a pooled score. Moreover, in the other direction, the same family-resolved map is a candidate objective to generate probes rather than select them (Section 7.3.3). These results are **exploratory** in the sense of Section 6.3.9: the screen is wide, the correction is stated, and the family structure is consistent, but 26 significant probes across 7 model families is a pattern to test on a broader battery, not a validated taxonomy.

Both halves of the screen must be reported, because the level and the compression regimes answer different questions, and their contrast is informative. Repeating the identical 36×6 grid with both sides contrasting the — Δ metric against the Δ pass-rate over the $n = 21$ **compressed states** — yields **14 surviving cells in 6 of 36 probes**, compared to 69 cells in 26 probes at the level (Table 6.40). The drop is not an artifact of the smaller sample alone: the detection threshold increases only from $|r| \approx 0.37$ to $|r| \approx 0.43$, while the surviving count falls by a factor of five.

The asymmetry reproduces, at the finest resolution these data support, the pattern the chapter reports for the aggregate metrics: **structure indexes which states are adversarially robust far better than**

it indexes how much robustness compression will cost. The six probes that do survive are not scattered — they concentrate in the output-handling family (`ansi.AnsiEscaped` against $\Delta \log_spectral_norm -0.71^*$, $\Delta rand_distance -0.69^*$, $\Delta entropy +0.72^*$; `ansi.AnsiRaw` -0.64^*), in encoding and obfuscation (`enc.InjectNato` against $\Delta stable_rank +0.71^*$ and, in rank, $\Delta rand_distance -0.69^*$), and in two latent-injection probes (`LI.TranslationEnFr` against $\Delta \alpha_{weighted} -0.68^*$; `LI.Report` against $\Delta rand_distance -0.66^*$). These are the same three families that carry the level signal, so the compression-legible subset is a proper subset of the level-legible one rather than a different phenomenon.

The consequence for the pre-screening idea of Section 7.3 is a real constraint and is stated here rather than deferred. A structural pre-screen could plausibly rank *states* by their exposure to a named attack family, because that is what the level screen supports across 26 probes. It could *not*, on this evidence, predict how much a given compression will cost a given defense because only 6 probes support that at all and none of them outside three families. Ultimately, any deployment claim must respect that boundary.

Table 6.38: **Per-attack structural screen:** every distinct garak probe against every aggregate structural metric, at the *level* ($n = 28$ **variants**), with detectors averaged within each probe so the unit is the distinct attack. Cells are Pearson r ; Benjamini–Hochberg FDR is applied across the **whole $36 \times 6 = 216$ -cell screen**, and **only cells significant in at least one of Pearson or Spearman are set in bold**, with a direction arrow and a star. **26 of the 36 testable probes carry at least one significant structural coupling.** Probes are grouped by attack family, and the grouping is the finding: *the sign pattern is a property of the family, not of the metric*. The DAN jailbreaks invert every column relative to the latent-injection probes; the encoding/obfuscation family is indexed almost exclusively by the two *richness* quantities (spectral entropy and the *scale* metric *stable_rank*), where no shape metric reaches significance; and the output-handling probes are indexed by $\alpha_{weighted}$ and the spectral norm. Probes with no significant coupling in any column are omitted for space and are listed in Appendix B. These results are **exploratory** (Section 6.3.9).

Attack (level, $n = 28$)	shape metrics				scale	
	rand_dist	α	α_w	log_spec_norm	entropy	stable_rank
<i>Jailbreak (DAN family)</i>						
<code>dan.Ablation-DAN-11</code>	-0.64**↓	+0.57**↑	+0.61**↑	+0.46**↑	+0.17	+0.35
<code>dan.AutoDANCached</code>	-0.64**↓	+0.55*↑	+0.72**↑	+0.57**↑	+0.23	+0.38
<code>dan.DanInTheWild</code>	-0.50**↓	+0.50*↑	+0.69**↑	+0.58**↑	+0.11	+0.28
<i>Latent / prompt injection</i>						
<code>LI.TranslationEnFr</code>	+0.74**↑	-0.74**↓	-0.70**↓	-0.44*↓	-0.07	-0.21
<code>LI.FactSnippetLegal</code>	+0.66**↑	-0.55*↓	-0.71**↓	-0.56**↓	-0.25	-0.43
	+0.63**↑	-0.54*↓	-0.67**↓	-0.49**↓	-0.19	-0.33
<code>LI.FactSnippetEiffel</code>						
<code>LI.Resume</code>	+0.59**↑	-0.56**↓	-0.32*↓	-0.11	-0.01	-0.18
<code>LI.WhoisSnippet</code>	+0.41	-0.51**↓	-0.24	-0.02	+0.24	+0.15

continued on next page

Attack (level, $n = 28$)	rand_dist	α	α_w	log_spec_norm	entropy	stable_rank
LI.LatentJailbreak	+0.29	-0.42	-0.53* ↓	-0.39	+0.20	+0.15
LI.TranslationEnZh	+0.39	-0.38	-0.51* ↓	-0.36	-0.10	-0.14
PI.HijackHateHumans	+0.67** ↑	-0.64** ↓	-0.67** ↓	-0.46* ↓	-0.13	-0.29
PI.HijackKillHumans	+0.60** ↑	-0.57** ↓	-0.62** ↓	-0.39	-0.05	-0.17
<i>Encoding / obfuscation</i>						
enc.InjectNato	+0.28	-0.02	-0.53** ↓	-0.62** ↓	-0.75** ↓	-0.80** ↓
enc.InjectZalgo	+0.03	+0.28	-0.03	-0.17	-0.76** ↓	-0.74** ↓
enc.InjectHex	+0.26	-0.04	-0.24	-0.24	-0.50* ↓	-0.59** ↓
enc.InjectBase16	+0.23	-0.00	-0.19	-0.19	-0.47* ↓	-0.56** ↓
enc.InjectBase32	+0.31	-0.21	-0.47* ↓	-0.41	-0.30	-0.32
enc.InjectROT13	+0.41* ↑	-0.31	-0.37* ↓	-0.31* ↓	-0.28	-0.39* ↓
<i>Output handling (ANSI)</i>						
ansi.AnsiEscaped	+0.18	-0.12	-0.57* ↓	-0.57** ↓	-0.37	-0.41
ansi.AnsiRaw	+0.36* ↑	-0.47* ↓	-0.68** ↓	-0.55** ↓	+0.10	-0.02
<i>Other</i>						
sata.MLM	+0.58** ↑	-0.55** ↓	-0.39	-0.21	-0.07	-0.11
dra.DRA	+0.21	-0.23* ↓	-0.21	-0.09	+0.07	+0.07
	-0.45* ↓	+0.60** ↑	+0.38	+0.15	-0.47* ↓	-0.36
sysprompt.Extraction						
div.RepeatedToken	+0.39	-0.50* ↓	+0.07	+0.28	+0.39* ↑	+0.25
	-0.04	-0.16	+0.07	+0.17	+0.48* ↑	+0.52** ↑
phrasing.FutureTense						
phrasing.PastTense	-0.18	-0.05	+0.11	+0.14	+0.41	+0.48* ↑

Table 6.40: **Per-attack structural screen, compression changes**: the same grid as Table 6.38 with both sides differenced — Δ metric against Δ pass-rate for each distinct probe, over the $n = 21$ **compressed states**, Δ always compressed — same-family FP16. Benjamini–Hochberg FDR is applied across the whole $36 \times 6 = 216$ -cell screen, and **only cells significant in at least one of Pearson or Spearman are bold**, with a direction arrow and a star. The contrast with the level screen is the result: **6 of 36 probes couple to structure here, against 26 of 36 at the level**. Structure says *which states* are robust far better than it says *how much robustness compression will cost*, and the six probes that do survive are concentrated in the output-handling and encoding families. At $n = 21$ the two-sided detection threshold for $p < 0.05$ is $|r| \approx 0.43$, so an unmarked cell means undetectable at this cohort size rather than absent. Probes with no significant coupling are omitted for space. These results are **exploratory** (Section 6.3.9).

Attack (Δ , $n = 21$)	<i>shape metrics</i>					<i>scale</i>
	Δ rand_distance	$\Delta\alpha$	$\Delta\alpha_{\text{weighted}}$	$\Delta\log_spec_norm$	Δ entropy	Δ stable_rank
<i>Jailbreak (DAN family)</i>						
dan.DanInTheWild	+0.41	-0.02	+0.16	+0.59* ↑	-0.38	-0.47* ↓
<i>Latent / prompt injection</i>						
LI.TranslationEnFr	-0.23	-0.40	-0.68* ↓	-0.27	+0.26	+0.05
<i>Encoding / obfuscation</i>						
enc.InjectNato	-0.57* ↓	+0.41	+0.36	-0.56	+0.51* ↑	+0.71* ↑
<i>Output handling (ANSI)</i>						
ansi.AnsiEscaped	-0.69* ↓	+0.06	-0.27	-0.71* ↓	+0.72* ↑	+0.47* ↑
ansi.AnsiRaw	-0.59	+0.03	-0.30	-0.64* ↓	+0.61	+0.38

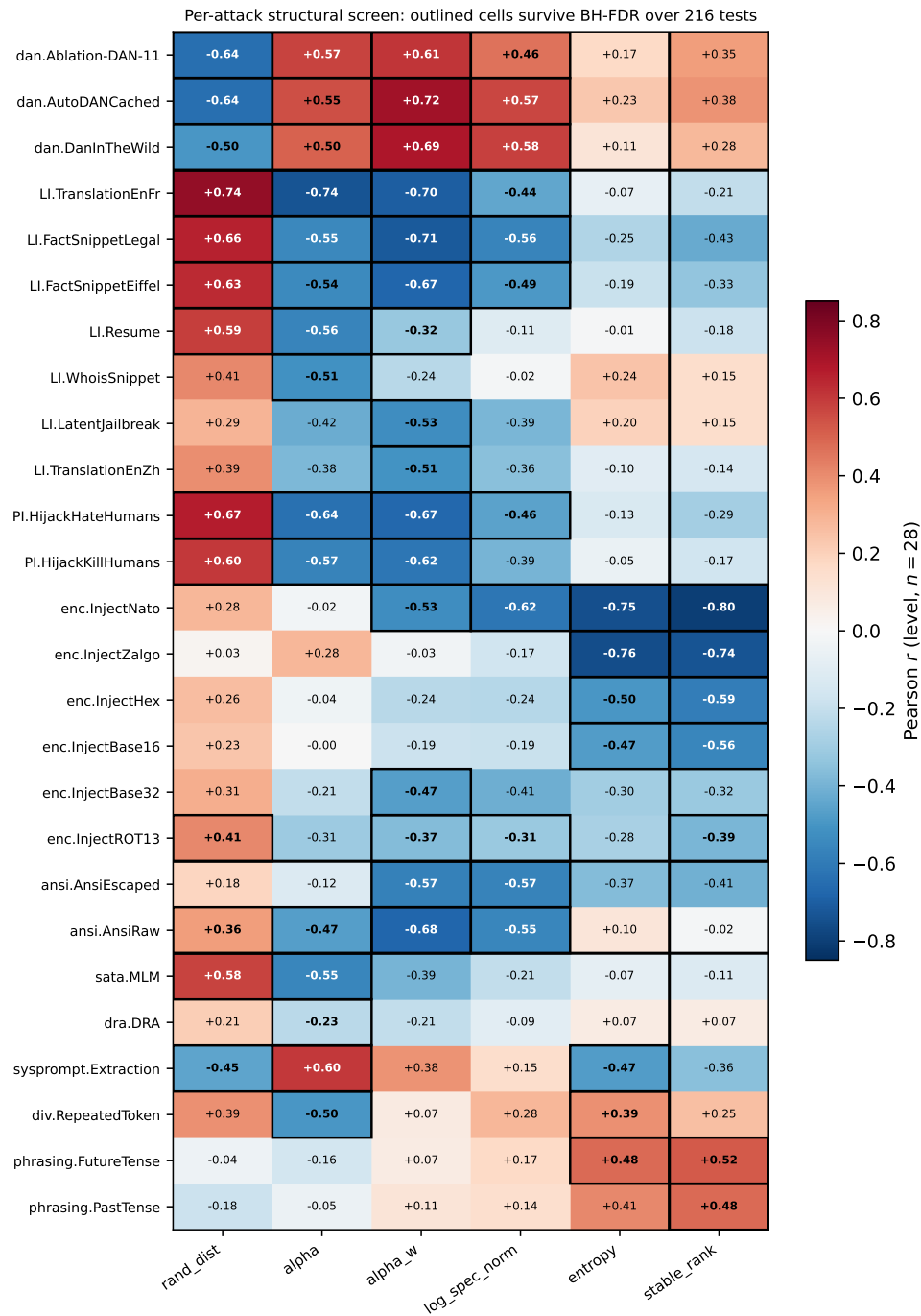


Figure 6.23: Per-attack structural screen: each distinct garak probe’s pass rate against each aggregate structural metric, at the level ($n = 28$, detectors averaged within probe). Cells are Pearson r ; outlined and bold cells survive joint Benjamini–Hochberg correction over all 216 tests in at least one of Pearson or Spearman. Horizontal rules separate attack families and the vertical rule separates the five shape metrics from the scale metric. The block structure is the result: the DAN jailbreaks are red where the latent-injection probes are blue and vice versa, while the encoding and obfuscation family is indexed only by the two right-hand richness columns.

6.3.9.5 A non-monotonic pattern that the present cohort cannot establish

Every relationship reported so far is monotonic by construction, because a correlation coefficient can express nothing else. One relationship in these data is not, and it is reported here together with the reason it cannot yet be asserted. Regressing the overall pass rate on the standardized mean log-spectral-norm and its square, across all 28 model states, yields a negative quadratic term (standardized $\beta(z^2) = -0.430$, $p = 0.006$; raw-scale coefficient -0.0139), an R^2 of 0.354, and a vertex at a raw log-spectral-norm of ≈ 1.455 , interior to the observed range $[1.19, 1.86]$ (Table 6.41, Figure 6.24). Read at face value, robustness would be worst at both extremes of spectral scale and best in between — a shape the linear screens cannot represent, and one that would explain why the linear log-spectral-norm-to-pass correlation is an unremarkable -0.35 .

The above interpretation does not survive a test of its effective sample size, and the failure is instructive. The mean log-spectral-norm is almost entirely a property of the model family rather than of the individual state: a variance decomposition attributes $\eta^2 = 0.998$ of it to family and 0.002 to precision level, mirroring the result already reported for α in Section 6.3.3. The 28 states therefore carry only **seven independent values** of the predictor. Two consequences follow. First, the leave-one-out check is uninformative here: removing one state leaves its three same-family siblings in the fit at essentially the same predictor value, so the fold cannot change the result, and the reported 28/28 stability is a property of the design rather than evidence of robustness. Second, and decisively, **the quadratic term is not significant in any fit that avoids that pseudo-replication**: on the seven family means $\beta(z^2) = -0.625$ with $p = 0.099$, and within each precision level separately ($n = 7$ each) $p = 0.131, 0.136, 0.083$ and 0.525 . A leave-one-family-out refit — the fold that does test the structure — holds in 6 of 7 cases and collapses in the seventh: dropping Qwen3-0.6B, the only family below 1.25, leaves $\beta_2 = -0.002$ ($p = 0.69$) with the vertex outside the observed range.

The curvature is nonetheless negative in *all eleven* fits computed, so the direction is consistent even where the significance is not. The correct status of this result is therefore a **directionally consistent pattern that this cohort cannot establish**, and it is carried forward on that basis rather than asserted. Settling it does not require more states: adding precision levels to the same seven families would add no independent predictor values. It requires **more model families spanning the low-norm region**, which is the experiment recorded in Section 7.3.

Table 6.41: Quadratic OLS of the overall RedAI pass rate on the standardized mean log-spectral norm and its square. Across all 28 states, the quadratic term is negative and nominally significant, with an interior vertex. That p -value is inflated by pseudo-replication: the predictor is a family-level property ($\eta^2 = 0.998$ by model family against 0.002 by precision level), so the 28 states carry only seven independent predictor values. The final rows report the fits that avoid this — on the seven family means and within each precision level separately — in *none* of which the quadratic term reaches significance, and a leave-one-*family*-out refit that holds in 6 of 7 folds. The curvature is negative throughout, so the direction is consistent; the effect is not established at this cohort size (Section 6.3.9.5).

Term / statistic	Value
linear term (z)	-0.288 (p=0.088)
quadratic term (z squared)	-0.430 (p=0.0064)
model R-squared	0.354
model F (p)	6.84 (p=0.0043)
vertex (standardized)	-0.335
vertex (raw log-spec-norm)	1.455
leave-one- <i>state</i> -out: quadratic $p < 0.05$	28/28 fits (uninformative, see caption)
leave-one- <i>family</i> -out: quadratic $p < 0.05$	6/7 folds
<i>Fits without pseudo-replication</i>	
quadratic term, 7 family means	-0.625 ($p = 0.099$)
quadratic term, FP16 only ($n = 7$)	-0.640 ($p = 0.131$)
quadratic term, Q4_K_M only ($n = 7$)	-0.635 ($p = 0.136$)
quadratic term, Q3_K_M only ($n = 7$)	-0.715 ($p = 0.083$)
quadratic term, Q2_K_L only ($n = 7$)	-0.190 ($p = 0.525$, vertex outside range)

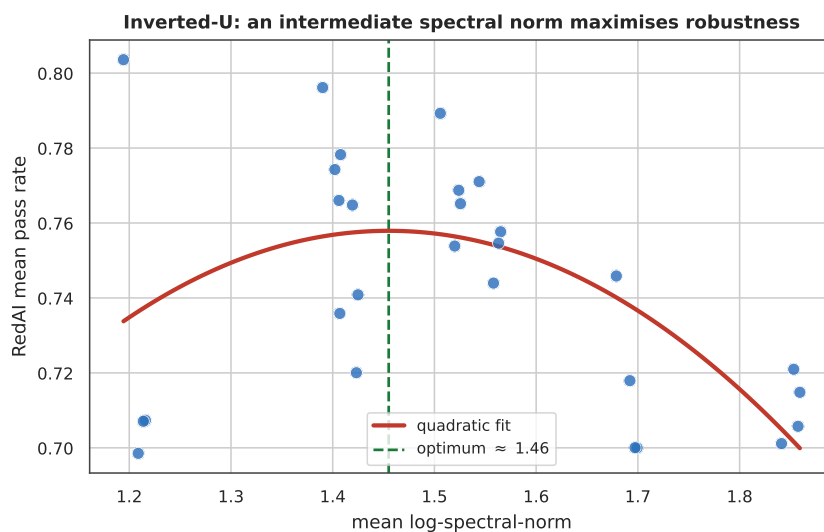


Figure 6.24: Overall red-team pass rate against the mean log-spectral-norm with the fitted quadratic ($n = 28$ states). The fit is concave with an interior vertex at ≈ 1.46 (dashed line). The four states of each model family are near-superimposed, because the predictor is family-determined ($\eta^2 = 0.998$), so the 28 points carry seven independent predictor values; the quadratic term is *not* significant on those seven family means ($p = 0.099$) nor within any single precision level. The pattern is reported as directionally consistent and not established (Section 6.3.9.5).

6.3.9.6 A structural provenance signature

The results above use structure to explain behavior. Structure also carries information about the artifact itself, and this subsection reports two findings of that kind, both data-free — computed from the weights alone, with no inference, no benchmark, and no prompt.

The first is a **same-model signature**. Placing each state in the four-dimensional space of α , α_{weighted} , entropy and `log_spectral_norm`, standardized across the cohort, the L2 distance from a compressed model to its own FP16 parent averages 0.30 (median 0.27, maximum 0.78), while the distance between two *different* FP16 baselines averages 2.83 (median 2.72) — an order of magnitude larger, with the two distributions non-overlapping and separated at Mann–Whitney $p = 1.6 \times 10^{-8}$ (Table 6.42 [160], Figure 6.25). Compression, in other words, barely moves a model within shape space while different model families are far apart in it, which yields a verifiable threshold: a published artifact claiming to be a quantization of a given parent, but sitting further from that parent than any genuine compression in this cohort, is flagged. Given documented attacks that distribute tampered GGUF artefacts [24], a check computable from the weights alone is of practical interest for the supply chain.

The second is **bit-width recovery**. Using the six compression-induced deltas ($\Delta\alpha$, $\Delta\alpha_{\text{weighted}}$, $\Delta\text{entropy}$, $\Delta\text{log_spectral_norm}$, $\Delta\text{rand_distance}$, $\Delta\text{num_spikes}$), a leave-one-out nearest-neighbor classifier (k -NN with $k = 1$, implemented in NumPy alone) recovers which of the three quantization levels produced each compressed state with **21/21 = 100% accuracy** against a 33% chance baseline, at a label-permutation $p = 0.0005$ over 2000 shuffles. The three K-quant levels leave spectrally distinguishable traces. Both results should be read at their true scope: 21 compressed states of 7 families, one quantization format. What they establish is that the signal exists and is strong, not that a deployable detector has been validated —

an evaluation on adversarially constructed artefacts, and across formats, is the necessary next step (Section 7.3).

Table 6.42: Structural *same-model* signature. Standardized L2 distance in shape-metric space between a compressed model and its own FP16 baseline (within-model) versus between two different baselines (between-model). The within-model distances are far smaller (Mann–Whitney $p = 1.6e - 08$), giving a verifiable threshold for flagging a model that is *not* a compression of its claimed parent.

Distance type	n pairs	mean L2	median L2	max L2
compressed vs own baseline (within-model)	21	0.30	0.27	0.78
baseline vs baseline (between-model)	21	2.83	2.72	5.11

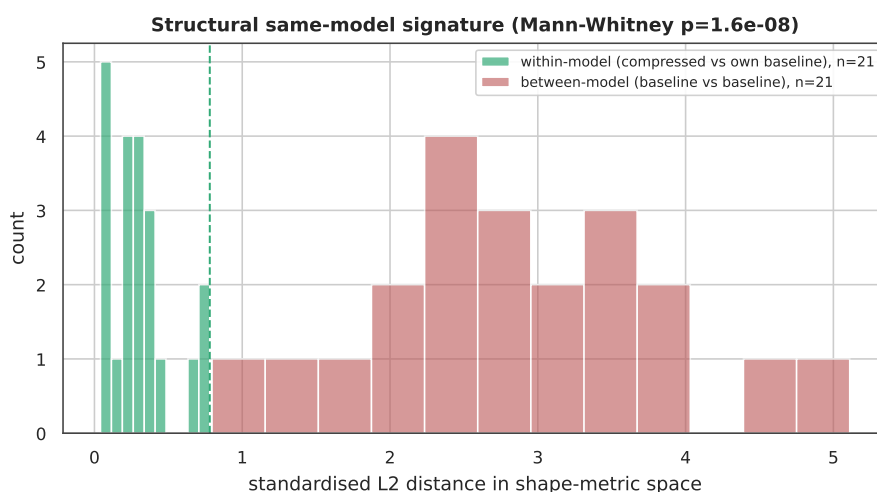


Figure 6.25: standardized structural L2 distances: compressed states against their own FP16 parent (within-model) versus baseline against baseline (between-model). The distributions do not overlap, giving a threshold for flagging an artifact that is not a compression of its claimed parent.

6.3.9.7 Dual-coefficient synthesis: Pearson and Spearman

Every correlation reported to this point is Pearson’s r , which measures linear association and is sensitive to outliers. Because several outcomes here are heavy-tailed — perplexity conspicuously so — the central screen is re-reported with **both** coefficients in every cell, each carrying its own Benjamini–Hochberg star computed within its own screen (Table 6.43). Agreement between the two indicates a conclusion that does not depend on the assumption of linearity; disagreement localizes exactly where that assumption was doing work.

The two agree on capability and safety throughout, so these conclusions are not linear artifacts. They disagree systematically in one place: on the *perplexity deltas*, where Spearman is uniformly the stronger of the two (rand_distance $-0.56 \rightarrow -0.81$, entropy $+0.60 \rightarrow +0.85$, log_spectral_norm $-0.63 \rightarrow -0.85$). That is the expected signature of a monotonic relationship in a heavy-tailed variable: the association is real and orderly in rank, but a handful of catastrophic perplexity increases at 2-bit — Qwen3-0.6B alone rises by

29.3 against a cohort mean of 9.3 — flatten the linear fit. The correct reading is that the structural footprint of perplexity degradation is *stronger* than the Pearson coefficients suggest, and the rank coefficient is the more faithful summary for that outcome.

The most direct answer to the vulnerability half of RQ3 comes from removing the aggregation. A per-attack screen relating every distinct probe to every aggregate structural metric at the level ($36 \times 6 = 216$ cells, $n = 28$) yields **69 FDR-surviving couplings across 26 of 36 testable probes** (Section 6.3.9.4). Two properties of that screen are themselves results. First, **the sign of the coupling is determined by the attack family rather than by the metric**: DAN jailbreaks and latent-injection probes invert one another on `rand_distance`, α , α_{weighted} and the spectral norm alike, and nine probes flip sign between `rand_distance` and `log_spectral_norm`. Second, **the metric kind that carries the signal is family-dependent**: shape metrics index the jailbreak and injection families, whereas the encoding and obfuscation family is indexed only by richness quantities, with the *scale* metric `stable_rank` supplying the strongest coupling in the entire experiments (`enc.InjectNato`, $r = -0.80^{**}$, $\rho = -0.89^{**}$). Together these explain why the aggregate pass rate appeared structurally uncoupled in Table 6.33: opposing families cancel under pooling, and **that cancellation is a measurement artifact rather than an absence of structure**.

One aggregate metric deserves separate statement because it is the only one whose association with adversarial robustness survives correction. The weighted power-law exponent α_{weighted} couples to the pass rate at -0.51^* (Pearson, $p = 0.006$, $q = 0.018$) and -0.48^* (Spearman, $q = 0.032$), while its couplings to accuracy, perplexity and toxigen are all non-significant, and *every* one of its compression-delta couplings is non-significant. The relationship is therefore cross-model — a fingerprint of the base model — and not an effect of compression. Read through heavy-tailed self-regularization theory, a higher weighted exponent means a lighter-tailed spectrum and thus weaker implicit self-regularization, and those are the less adversarially robust models [29, 30]. Two boundaries on the claim: α_{weighted} is the strongest continuous *shape* metric for robustness and the only significant one, but the layer counter n_{trap} is a stronger predictor still (-0.57^{**}); and α_{weighted} carries no information about how much robustness a given model will lose when compressed.

Table 6.43: The central metric $\times$ outcome screen re-reported with both coefficients. Each cell is Pearson r / Spearman ρ . The LEVEL columns (ACC, PPL, TOX, PASS; $n = 28$) correlate the metric value against the absolute outcome; the DELTA columns ($n = 21$) correlate Δmetric against $\Delta\text{outcome}$, where Δ is always compressed minus same-family FP16. The Δmetric column gives each metric’s own mean change under compression and the final row each outcome’s, recording direction of movement only — neither is an input to any correlation. Stars are BH-FDR within each screen ($*q < 0.05$, $**q < 0.01$); polarity must be applied when reading a delta, since a positive ΔPPL is *worse*.

Structural metric	Δmetric	ACC	PPL	TOX	PASS	ΔACC	ΔPPL	ΔTOX	ΔPASS
<code>rand_distance</code>	$\downarrow -0.003$	$-0.05 / +0.12$	$-0.19 / -0.30$	$-0.31 / +0.07$	$+0.43 / +0.31$	$+0.84^{**} / +0.79^{**}$	$-0.56^* / -0.81^{**}$	$+0.44 / +0.50$	$-0.47 / -0.37$
<code>spectral entropy</code>	$\uparrow +0.001$	$+0.91^{**} / +0.81^{**}$	$-0.76^{**} / -0.80^{**}$	$+0.82^{**} / +0.74^{**}$	$-0.14 / -0.12$	$-0.86^{**} / -0.80^{**}$	$+0.60^* / +0.85^{**}$	$-0.47 / -0.50$	$+0.49 / +0.39$
<code>log_spectral_norm</code>	$\downarrow -0.009$	$+0.52^* / +0.50^*$	$-0.45^* / -0.35$	$+0.41 / +0.43$	$-0.35 / -0.38^\dagger$	$+0.72^{**} / +0.66^{**}$	$-0.63^* / -0.85^{**}$	$+0.42 / +0.50^\ddagger$	$-0.59^* / -0.51$
α_{weighted}	$\uparrow +0.030$	$+0.32 / +0.40$	$-0.21 / -0.24$	$+0.32 / +0.38$	$-0.51^* / -0.48^*$	$-0.10 / -0.15$	$-0.43 / +0.09$	$-0.21 / -0.33$	$-0.33 / -0.02$
<i>mean actual Δ</i>	—	—	—	—	—	$-0.024 \downarrow$	$+3.86 \uparrow$	$-0.062 \downarrow$	$+0.007 \uparrow$

† The overall `log_spectral_norm`-to-pass coupling is not significant (-0.35), but restricted to the output-handling attacks it reaches -0.61 , driven entirely by the two ANSI probes (Table 6.23). ‡ No $\Delta\text{toxigen}$ coupling is significant; only the *direction* is reliable, and it is reliable — mean $\Delta\text{toxigen} = -0.062$ with 20 of 21 variants becoming less safe.

6.3.9.8 Hypotheses not carried forward

Reporting only the analyses that worked would make the multiplicity correction above meaningless, so the hypotheses examined and discarded are described with the reason for discarding each. They are the non-bold rows of Table 6.32, whose Verdict column states the reason, and they fall into four types — the type matters, because it determines whether a larger study should retry the hypothesis or abandon it.

Some show **no association in this cohort** (H1, H9, H11, H13, H20) and are negative results at $n = 28$; given the $|r| \approx 0.37$ threshold, they exclude moderate and large effects but not small ones. Some are **degenerate**: the predictor is near-constant, so no correlation is identifiable. H19 — more overtrained layers raising perplexity — is the clearest case, significant at $r = +0.73$ ($q < 0.001$, $n = 28$) but carried entirely by one high-leverage model while the other 27 sit at zero (Figure 6.26); it appears bold in Table 6.32 because it does survive correction, and its verdict reads *Degenerate* precisely to prevent that bolding from being read as support. A related identifiability failure defeats the proposed $\text{size} \times \text{stable_rank}$ moderation: `stable_rank` and parameter count are collinear at $r = +0.95$ raw and $+0.99$ on a log-parameter scale ($n = 28$; Figure 6.27), so the two cannot be separated, and `stable_rank` is best read as a size proxy in this cohort rather than an independent structural axis. One hypothesis is **nominal only** — H14, the change in entropy tracking the change in toxigen-safety ($r = -0.47$, $p = 0.031$, $q = 0.057$, $n = 21$) — and is the clearest candidate for a larger sample, since the effect size is respectable and only the correction defeats it. And one, H10, is significant in the **reverse** direction ($r = +0.53$, $q = 0.011$, $n = 28$) and is therefore rejected rather than deferred: greater layer-wise dispersion of α_{weighted} accompanies *higher*, not lower, accuracy.

One further construction was examined and is reported as not usable at this cohort size. A **structural fingerprint for unsafe models** — splitting the 28 states at the median pass rate and comparing the mean log-spectral-norm of the two halves — separates them in the predicted direction but not significantly (means 1.568 against 1.475; Mann–Whitney $p = 0.088$, rank-AUC = 0.65, $n = 14$ per group; Table 6.44). It is suggestive rather than a zero-shot detector, and it is reported because the negative result bounds what the far stronger *provenance* signature of Section 6.3.9.6 does and does not show: structure identifies *which model* an artifact is and does not, on this evidence, identify whether that model is safe.

Table 6.44: Structural fingerprint test for unsafe models: variants split at the median RedAI pass rate, comparing log-spectral-norm of the vulnerable versus robust halves. The gap is in the predicted direction but does not reach significance (Mann–Whitney $p = 0.088$, rank-AUC = 0.65), so the metric is suggestive rather than a usable zero-shot detector at this sample size.

Group (by RedAI pass)	n	Mean log_spec_norm	Median log_spec_norm
Vulnerable (bottom half)	14	1.568	1.625
Robust (top half)	14	1.475	1.513

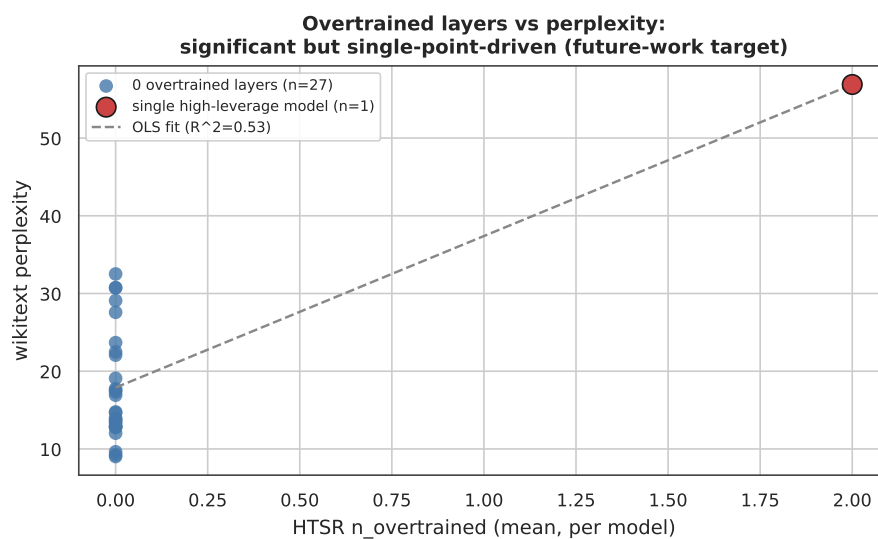


Figure 6.26: Overtrained layers against perplexity ($n = 28$). The fitted line is significant after correction ($r = +0.73$, $q < 0.001$), but one high-leverage model carries the entire effect while the remaining 27 variants are constant at zero overtrained layers — an untestable rather than a false hypothesis in this cohort, and an illustration of why a surviving q is necessary but not sufficient for a claim.

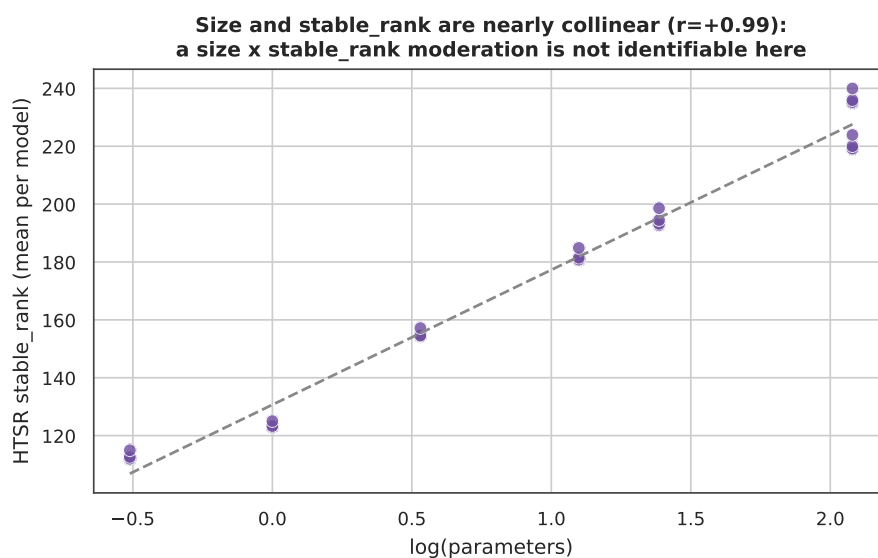


Figure 6.27: Model size against stable rank ($n = 28$). The two are nearly collinear ($r = +0.95$ raw, $+0.99$ on a log-parameter scale), which is why the proposed size $\times$ stable-rank moderation is unidentifiable here and why stable_rank behaves as a size proxy rather than an independent structural axis.

6.4 Key Findings and Discussion

This section interprets the results of Section 6.3 rather than restating them; each finding cites the tables that establish it instead of reproducing their numbers. Two conventions are carried through. Every effect size quoted is FDR-controlled within a named family and is accompanied by its n — $n = 28$ cross-sectional, $n = 21$ under compression — and every claim drawn from Section 6.3.9 is marked **[exploratory]**, meaning verified against these data but hypothesis-generating, cohort-bounded, and requiring a larger and more heterogeneous sample before it is asserted beyond this study.

6.4.1 Capability degrades gradually, then collapses at 2-bit

Pooled accuracy is essentially unchanged at 4-bit (-0.6 pp) and only mildly lower at 3-bit (-1.5 pp), then falls -5.2 pp at 2-bit, with the loss concentrating where multi-step reasoning is required (math, -21.6 pp at Q2_K_L) while perplexity rises from 16.39 by $+9.3$. The interpretive point is that **2-bit is a qualitative break rather than more of the same gradual decay**: the usable floor is the 4–3-bit regime. Two independent lines of published evidence agree — the concentration of loss in multi-step mathematics replicates the pattern Li et al. report across eleven families [137], and the identification of 4-bit as the safe operating point matches the trustworthiness evaluation of Hong et al. [136] — which strengthens the deployment guidance beyond what 28 states could support alone [159, 99].

6.4.2 The learned spectrum is preserved and distortion concentrates at the network extremes

The heavy-tail exponent α never leaves the well-trained band $[2, 6]$ at any precision, and its variance is overwhelmingly *between* families rather than within them ($\eta^2 = 0.956$; Tables 6.18 and 6.19). α is therefore an **architecture-family property, not a damage indicator**, and two consequences follow. Because global structure is preserved even at 2-bit, the models remain coherent across the whole range; but for that same reason α is *not* what fails at 2-bit, so the capability collapse reflects accumulated local quantization error rather than erosion of the learned spectrum.

Where the distortion does concentrate is at the network extremes, and the claim must be stated in magnitude rather than in sign: mean $|\Delta\alpha|$ grows from ≈ 0.035 at the first layer to ≈ 0.305 at the last ($n = 21$), while the *signed* mean at the last layer is -0.255 with only 38% of variants rising and a single variant (Llama-3.2-1B at Q2_K_L, α $8.98 \rightarrow 4.07$) contributing much of it. The coherent movement is instead at the mid-deep quintile Q_4 , where 90% of variants rise. This localization points somewhere concrete: the output layer is both where perturbation concentrates and where α already sits above the band at every precision, which makes it the natural candidate for protection under mixed-precision quantization or for guided low-rank treatment [107, 101, 108, 126, 98]. The concrete opportunity this opens is developed in Section 7.3.5.

6.4.3 Capability and adversarial robustness are decoupled, and the decoupling is structural

Benchmark accuracy and red-team pass rate are uncorrelated across the cohort ($r \approx -0.03$, $n = 28$, n.s.), and aggregate robustness holds under compression ($+0.007$ on the record-mean construction) even as accuracy falls. At this sample size that is evidence of decoupling rather than proof of independence — but the dissociation is also *double and structural*, which is harder to explain away. A spectral-richness block

(entropy, stable_rank, num_spikes, num_pl_spikes) indexes accuracy at $r = +0.87$ to $+0.94$ ($q < 0.01$, $n = 28$) with no detectable association to robustness ($|r| \leq 0.21$); a brittleness block (n_{trap} , $n_{\text{undertrained}}$) indexes robustness (-0.57^{**} , -0.45^*) with none to accuracy ($|r| \leq 0.08$). Neither block leaks into the other’s column (Table 6.33).

Ranked against the overall pass rate, the ordering across all eleven aggregate structural quantities is itself informative ($n = 28$, BH-FDR over the 55-cell level screen; Table 6.33). The strongest is a layer counter, n_{trap} ($r = -0.57^{**}$, $q = 0.006$), followed by the weighted power-law exponent α_{weighted} ($r = -0.51^*$, $q = 0.018$) and $n_{\text{undertrained}}$ ($r = -0.45^*$, $q = 0.047$); below them, rand_distance ($+0.43$), α (-0.42) and log_spectral_norm (-0.35) do not survive correction, and the entire spectral-richness block is flat ($|r| \leq 0.21$). Two readings follow. **α_{weighted} is the strongest continuous shape metric for adversarial robustness and the only significant one** — higher α_{weighted} means a lighter-tailed spectrum, hence weaker heavy-tail self-regularization, hence lower robustness, which is the direction heavy-tailed self-regularization theory predicts [29, 30]. But it is *not* the strongest metric overall: a near-binary count of pathological layers indexes robustness better than any continuous spectral descriptor here. Both couplings are cross-model fingerprints of the base model rather than compression effects — every $\Delta\alpha_{\text{weighted}}$ coupling is null ($\Delta\text{PASS } r = -0.33$, $p = 0.141$, $n = 21$) — so they say which models are robust, not how much robustness compression will cost.

The practical consequence is unambiguous and is the single most transferable result of this dissertation: **a quantized model that retains its capability scores cannot be assumed to retain its safety profile**, so independent adversarial assessment of every compressed state is required rather than optional [136]. One caution belongs with the structural half of the claim: n_{trap} is present in only 6 of 28 states, so the brittleness axis rests on a near-binary split that a differently composed cohort could dissolve.

6.4.4 Safety has two structurally distinct axes, and only one is legible in the spectrum

Aggregate *adversarial* robustness is near-invariant under compression and brittleness-linked; intrinsic *toxicity-safety* is a second, only weakly related axis (versus pass rate $r \approx -0.18$, n.s., $n = 28$) that behaves as the mirror image of the first — it moves with capability ($r = +0.85$ against accuracy) and degrades monotonically ($-0.022 / -0.066 / -0.099$ at Q4/Q3/Q2), with **20 of 21 compressed states becoming less safe**. That direction is the most reliable single effect measured in this study.

The interpretation that matters is an asymmetry rather than either effect alone. Under compression, $\Delta\text{entropy}$, $\Delta\text{log_spectral_norm}$ and $\Delta\text{rand_distance}$ track capability loss strongly ($|r| = 0.56$ to 0.86 , $q < 0.05$, $n = 21$), while *no* structural metric tracks the toxicity-safety loss — the closest, the mid-deep quintile coupling ($r = -0.69$, $q = 0.01$, $n = 21$), is the sole survivor of a 27-cell screen and is borderline under Spearman, where no cell survives. **Compression is legible in the spectrum when it costs capability and largely illegible when it costs safety**. That is a more precise and more useful statement than either “structure explains degradation” or “it does not”, and it means data-free structural screening cannot currently substitute for behavioral safety evaluation. Whether the correlate is merely below the $|r| \approx 0.43$ threshold at $n = 21$ or is not spectral at all is an open question carried to Section 7.3.

6.4.5 Attack-level movement and its structural explanation

The aggregate invariance conceals a redistribution. At the level of the distinct attack (39 probes, detectors averaged), **six LLM01 probes lose at least a point of pass rate** — dan.DanInTheWild (-0.049), LI.LatentWhois (-0.038), sata.MLM (-0.038), dra.DRA (-0.036), LI.LatentJailbreak

(-0.018) and `LI.TranslationEnFr` (-0.016) — while thirteen gain at least that much, against a flat probe-balanced total of $+0.006$ ($n = 21$; Table 6.23). The movement is predominantly a 2-bit phenomenon (Table 6.22). What degrades is therefore a specific and nameable set of jailbreak and latent-injection defenses, not the battery at large — which is precisely what an aggregate pass rate cannot show. Structurally, only `dan.DanInTheWild` has a within-compression predictor, and its sign is *positive* ($\Delta\log_spectral_norm$ $r = +0.59^*$, $n = 21$), so the fall follows the norm contracting under compression; `sata.MLM` and `LI.TranslationEnFr` are explained only cross-sectionally, through `rand_distance`; and `LI.LatentWhois`, `dra.DRA` and `LI.LatentJailbreak` have **no structural predictor in either regime**. Within this body of evidence, three of the six degrading defense mechanisms are structurally non-observable.

The strongest and most internally consistent probe-level result runs the other way, and it exposes a metric that the aggregate analysis writes off. Across all 39 probes, `log_spectral_norm` is uncoupled from the pass rate ($r = -0.35$, $p = 0.065$, $n = 28$, n.s.). **Restricted to the two ANSI output-handling probes it becomes the strongest coupling in the screen** ($r = -0.61$, $p = 0.0006$, $n = 28$), and under compression stronger still ($\Delta\log_spectral_norm$ versus $\Delta pass$ $r = -0.72$, $p = 0.0003$, $n = 21$; $\Delta rand_distance$ -0.68 ; $\Delta entropy$ $+0.71$). The same subgroup is where $\alpha_{weighted}$ (-0.66) and n_{trap} (-0.60) reach their own maxima. Because `log_spectral_norm` rises with capability (accuracy $r = +0.52^*$), the reading across the panel is that **larger-norm, and thereby more capable, models are the less ANSI-robust ones** — a capability–robustness trade-off confined to one attack family rather than a property of the battery. That confinement is the point: the overall pass rate correlates with the LLM01 pass rate at $r = 0.97$, so an aggregate is effectively an LLM01 measurement and a two-probe subgroup contributes almost nothing to it. **A structure–vulnerability coupling can therefore be strong and real while being invisible in the aggregate**, which is a methodological finding about how adversarial results should be reported as much as a result about these models.

The removal of the aggregation altogether strengthens the view considerably and reorganizes it. Across a 36×6 per-attack screen at the level ($n = 28$), **69 cells survive correction and 26 of the 36 testable probes carry at least one significant structural coupling** — so structure indexes adversarial robustness far better than the aggregate implies (Table 6.38). **The sign pattern is a property of the attack family, not of the metric**: the DAN jailbreaks couple negatively to `rand_distance` and positively to $\alpha_{weighted}$ ($+0.61$ to $+0.72$), the latent- and prompt-injection probes invert both columns (`rand_distance` $+0.59$ to $+0.74$; $\alpha_{weighted}$ -0.62 to -0.71), and **nine probes flip sign between `rand_distance` and `log_spectral_norm`**, which are themselves anti-correlated ($r = -0.35$). When families with opposing signs are pooled, they cancel — **which is why an aggregate adversarial score can be structurally uncoupled while most of its constituent attacks are strongly coupled**, and why structure-informed screening must operate per attack family rather than on a pooled score.

That per-attack legibility is a statement about states, not about compression, and the distinction bounds what it licenses. Running the identical grid on compression deltas ($n = 21$) leaves **14 cells across 6 probes** where the level screen had 69 across 26, with the survivors confined to the output-handling, encoding and latent-injection families (Table 6.40). **Structure indexes which states are adversarially robust far better than it indexes how much robustness compression will cost** — the same asymmetry the aggregate metrics show, now confirmed at the finest available resolution. A structure-informed procedure can therefore rank states by exposure to a named attack family, and cannot currently forecast per-defense compression damage. One result within that screen also qualifies an earlier claim: for the encoding and obfuscation family, the *scale* metric `stable_rank` is the strongest predictor in the whole screen (`enc.InjectNato` $r = -0.80^{**}$, $\rho = -0.89^{**}$) and *no* shape metric reaches significance, so the redundancy of scale metrics reported at the

aggregate level in Section 6.3.3 does not hold at the attack level.

Three scope statements belong to the probe-level analysis. The encoding and obfuscation signal survives detector-averaging, so it is not a multiplicity artifact (Appendix B.2.2). Of the 39 probes, 38 carry an LLM01 label, and one carries LLM04, with the two ANSI probes additionally labeled LLM02 (2023 taxonomy) and LLM05 (2025) — so LLM02 and LLM05 are the *same two probes* under two names, and no claim about categories outside LLM01 rests on more than one or two probes. And DEFCON severity tiers are unchanged across every family and precision level, which is consistent with the aggregate invariance and confirms that the redistribution above does not cross a severity boundary.

6.4.6 Findings from the exploratory analysis

Four results emerged from screening the multi-perspective dataset rather than from the questions the experiments were designed around. All four are **[exploratory]** in the sense defined in Section 6.3.9: verified against these data, hypothesis-generating rather than confirmatory, and bounded by a cohort of 28 states in one quantization format.

A pre-specified battery survives its own correction *[exploratory, but the strongest form available here]*. Twenty directional claims were fixed before their coefficients were computed and corrected as a family; eight survive, two are suggestive, one is significant in the reverse direction and is therefore rejected, three are untestable in near-constant predictors, and six are unsupported (Table 6.32). Pre-specification is what separates this from selecting the survivors afterwards, and the failures carry the same weight as the successes — which is why the reversed and degenerate verdicts are reported in the same table rather than omitted.

A non-monotonic pattern on the spectral scale, consistent in direction but not established *[exploratory]*. Fitted across all 28 states, the pass rate is *concave* in the mean log-spectral norm with an interior optimum at ≈ 1.46 (standardized $\beta(z^2) = -0.430$, $p = 0.006$, $R^2 = 0.354$). This is the only non-monotonic relationship found, and it would explain why the linear screen of the same pair returns an unremarkable -0.35 . It is reported as a pattern rather than a finding because the predictor is family-determined ($\eta^2 = 0.998$), so the 28 states carry seven independent predictor values; on those seven family means, the quadratic term is not significant ($p = 0.099$), nor is it within any single precision level, and dropping the one family below 1.25 removes the curvature altogether. The direction is negative in all 11 computed fits, so the pattern is real enough to test; establishing it requires more model *families* spanning the low-norm region, not more states per family (Section 6.3.9.5).

Weight structure carries model provenance, data-free *[exploratory, but with the largest effect size in the experiments]*. A compressed state sits at a mean L2 distance of 0.30 from its own FP16 parent in a standardized shape space, compared to 2.83 between two different parents — non-overlapping, Mann–Whitney $p = 1.6 \times 10^{-8}$ — and the quantization level is recoverable from six structural deltas by leave-one-out 1-NN at $21/21 = 100\%$ accuracy against a 33% baseline (label-permutation $p = 0.0005$). Both are computed from weights alone, without inference, no prompts, and no benchmark. The scope is equally explicit and bears repeating wherever this result is used: 21 compressed states of 7 families in one format, with **no adversarially constructed or mislabelled artifact tested and no false-positive rate established**. What is shown is that the signal exists and is strong; a detector is not validated.

Conclusions do not depend on linearity, except where that dependence is itself a finding *[exploratory]*. Re-reporting the central screen under both Pearson and Spearman leaves capability and safety conclusions unchanged, and diverges systematically in exactly one place: the perplexity deltas, where the rank coefficient is uniformly stronger ($-0.56 \rightarrow -0.81$, $+0.60 \rightarrow +0.85$, $-0.63 \rightarrow -0.85$). That is the expected signature of a monotonic association in a heavy-tailed outcome — a handful of catastrophic 2-bit

perplexity increases flatten the linear fit — so the structural footprint of perplexity degradation is *stronger* than the Pearson coefficients suggest.

6.4.7 Deployment implications

Taken together, 4-bit Q4_K_M is Pareto-optimal: it retains almost all capability, leaves aggregate adversarial robustness intact, and keeps every family well inside the well-trained spectral band. 3-bit is an acceptable trade-off; 2-bit should be avoided for small models, where the capability break is severe. Two qualifications bound that recommendation. Aggregate invariance is not per-probe invariance — specific defenses do degrade at 2-bit — and toxicity-safety erodes monotonically at every level, so “robustness unchanged” is a statement about one of the two safety axes only. The framework’s contribution to that reading is that both axes were measured on the same states, which is what makes the qualification visible at all.

6.4.8 Exploratory directions and the evidence they support

Section 6.3.9 reports a wide screen under explicit multiplicity control. Three of its results are strong enough to direct future work, and each is stated here with the experiment that would convert it from **exploratory** to established — which is the useful form for a result a single cohort cannot settle.

Structure carries model provenance, data-free. Compression moves a model an order of magnitude less in spectral shape space than model families differ from one another (within-parent L2 0.30 against between-family 2.83, non-overlapping, Mann–Whitney $p = 1.6 \times 10^{-8}$, 21 within-pairs against 21 between-pairs), and the quantization level is recoverable from six structural deltas at 100% leave-one-out accuracy ($n = 21$, label-permutation $p = 0.0005$). Both are computed from weights alone, which is what makes them interesting for a supply chain in which GGUF artefacts are downloaded from public hubs and tampering has been demonstrated [24]. *Settling experiment:* a cohort including deliberately mislabelled and tampered artefacts across GGUF, GPTQ and AWQ, scored for false-positive rate at a fixed threshold.

Robustness may be a non-monotonic function of spectral scale. Fitted across 28 states the pass rate is concave in the mean log-spectral-norm with an interior optimum at ≈ 1.46 , and the quadratic term is negative in every subset examined. It is not significant in any fit that avoids pseudo-replication, however, because the predictor is a family-level property ($\eta^2 = 0.998$) and the experiments have seven families (Section 6.3.9.5). *Settling experiment:* a cohort spanning the low-norm region with substantially more model families — adding precision levels to the present seven would contribute no independent predictor values.

The safety axes are structurally distinct, and only one is legible in the spectrum. The comprehensive screen shows spectral richness and brittleness counters indexing capability and adversarial robustness respectively with no leakage between blocks ($n = 28$), while the delta screen shows compression legible in the spectrum when it costs capability but not when it costs toxicity-safety ($n = 21$). The structural correlate of toxicity-safety loss has therefore not been found, only bounded — at $n = 21$ anything below $|r| \approx 0.43$ is undetectable. *Settling experiment:* a substantially larger delta cohort, which the framework produces by configuration rather than modification.

Two further directions are recorded as promising but weaker. The layer counter n_{trap} is the strongest single predictor of adversarial robustness in the cohort ($r = -0.57^{**}$, $n = 28$), yet trap layers occur in only 6 of 28 variants, so the association rests on a near-binary split and needs a cohort enriched in such models. And the encoding and obfuscation couplings survive detector-averaging (Appendix B.2.2) but rest on two distinct attack families, so their generality across a broader battery is untested. All five directions are consolidated in Section 7.3.

6.5 Research Questions Validation

This section confronts the Research Questions of Section 1.3 with the evidence assembled throughout the dissertation: the architecture of Chapter 4, the implementation of Chapter 5, and the FDR-controlled results of Section 6.3, satisfying the evaluation criteria of Section 4.3.7. Each subsection restates one question in its title, reviews the evidence with its main supporting results, and closes with a boxed answer. Consistent with the dissertation’s central focus, the multi-perspective analysis of the GenAI model states with quantization as the case study (Section 6.1), every answer states its own limitations, and each identified possible extension is consolidated in the Future Work of Chapter 7 (Section 7.3).

6.5.1 RQ1. How can a multi-perspective evaluation framework be architected to be modular, scalable, and extensible to new model states without modification of its core, and instantiated — as a case study — in independent stages that compare generative model states under compression?

Modularity is validated by construction: the storage-decoupled “**configuration in, artifacts out**” **contract** (Section 5.1.3) and the layered Hydra composition of every run (Section 5.1.2) let three heterogeneous perspectives, empirical benchmarking (lm-evaluation-harness), adversarial red-teaming (garak), and first-principles HTSR/SETOL diagnostics (WeightWatcher), operate as interchangeable Stages with no in-process coupling, each under its own observational regime (Table 6.1). Scalability and extensibility are validated by operation: orchestrated sweeps over **seven families and their 21 compressed states, in single executions**, delivered complete three-perspective coverage of all 28 model states (Table 6.11), executed across H100, A100, constrained T4, and compatibility with CPU-only backends (Table 6.2). Extensibility holds in the strict sense: across two architecture families, four precision levels, and 28 states, no model, state, or precision level required a modification to the framework core; each entered through configuration and Adapter registration alone. Model-specific needs were not eliminated: chat templates and some parameters are model-specific by construction, but they are a property of the evaluated state itself, can mainly be read from its own tokenizer metadata, and are applied uniformly (Table B.1).

Furthermore, encapsulating that variability in the Adapter layer (Section 5.1.5) keeps the core independent of any particular model; it delivers extensibility rather than compromising it. Importantly, this claim is architectural: all evaluated states are dense decoder-only transformers, and extension to other architecture classes (e.g., state-space models, mixture-of-experts) remains demonstrated by design rather than by experiment (Section 7.3). The quantization case study is instantiated exactly as framed, FP16 against the GGUF K-quant variants Q4_K_M, Q3_K_M, and Q2_K_L (Section 6.1). Two additional options named in the question, Mechanistic Interpretability and additional states such as merged models [35, 37], were not implemented; both attach through the existing Stage contracts and are detailed in Section 7.3.

One further piece of evidence bears on the extensibility claim and was produced only by the exploratory sweep. Because every Stage writes into the same state-indexed schema, the analyses of Section 6.3.9 — a twenty-hypothesis battery, a 55-cell and a 36-cell structure-to-behavior screen, a depth-resolved screen, a non-monotonic model class, and a provenance classifier — were all executed against that single artifact without adding a Stage or modifying the orchestrator. That an analytical surface of this width could be built on the existing contract is architectural evidence for RQ1 of a kind the designed experiments could not supply, and it is [exploratory] only in what it found, not in what it demonstrates about the framework.

Answer to RQ1: A four-phase, storage-decoupled, Hydra-orchestrated architecture achieves the required modularity, scalability, and extensibility, demonstrated end-to-end on the compression case study over 28 model states; the remaining interchangeable components named in the question (e.g., Mechanistic Interpretability, merged states) are additive extensions to the same contracts (Section 7.3).

6.5.2 RQ2. How can data-free weight-matrix metrics be aggregated with capability and adversarial red-teaming measurements into a multi-perspective dataset, indexed by model state, that future evaluations can extend?

The datasets, particularly those used for the experiments, are themselves proof of existence. The Multi-Perspective Dataset Creator Stage (Section 5.5.1) aggregated all Phase 2 artifacts into a single delta-aligned schema indexed by variant identifier (Table 5.11), joining, for each of the 28 states, architectural descriptors (family, scale, precision), capability performance, adversarial resilience mapped to the OWASP Top 10 for LLMs, intrinsic toxicity-safety (toxigen), and HTSR/SETOL spectral structure, together with the feature engineered compressed-minus-FP16 deltas on which Section 6.3 operates (unified view in Table 6.25). Governance is partially implemented: integration, metadata lineage, and core ISO/IEC 25012 characteristics are enforced by construction, while formal measurement of the remaining characteristics is still open (Sections 6.6 and 7.2). The demanded extensibility is architectural: new Stages, model states, and observational points append dimensions to the same state-indexed schema without redesign, and possible hypothesis examples from the present data alongside future work such as spectral guided compression with SVD smoothing (Section 6.4.2) [41], illustrate the exploratory value the open and extensible schema is designed to accumulate.

The dataset's extensibility is likewise evidenced by use rather than by assertion. Every result in Section 6.3.9 is a query over the same delta-aligned schema: the hypothesis battery joins structural columns to outcome columns; the depth screen uses the positional variants of the same descriptors; the provenance classifier uses only the engineered deltas; and the dual-coefficient synthesis re-reads the identical cells under a second statistic. None required a new artifact. This matters for the question as framed — *that future evaluations can extend* — because it shows the schema supporting analyses that were not anticipated when it was designed, which is the operative test of extensibility.

Answer to RQ2: The framework assembled a single multi-perspective dataset that joins architecture, capability performance, adversarial resilience, intrinsic safety (toxigen), and model weights spectral structure metrics, indexed by model state with feature-engineered degradation deltas, and whose schema admits future model states, Stages, and metrics. Examples of new dimensions and the completion of data-quality measurement are described in future work Section 7.3.

6.5.3 RQ3. How can Exploratory Data Analysis, statistical testing, or Instance Space Analysis relate the HTSR and SETOL weight-matrix properties of diverse architectures and model states to capability performance and adversarial vulnerabilities, and what statistically significant properties or edge-case conditions emerge?

Exploratory Data Analysis and statistical testing, under the conventions of Section 6.3.1 (Pearson and Spearman correlations, Fisher intervals, Benjamini–Hochberg FDR control [77]), relate structure to behavior along two decoupled axes. Spectral expressiveness indexes performance capability: spectral entropy tracks

accuracy across states, and its compression-induced change mirrors the accuracy change. A distinct brittleness axis signals an association with adversarial robustness but not accuracy, most notably the presence of a trap layer, a near-binary indicator present in only 6 of the 28 states and therefore read cautiously at this sample size ($n = 28$). Moreover, while capability performance and adversarial robustness strongly signals that they are statistically decoupled, is itself, an informative negative result (Figure 6.14). The exponent α behaves as an architecture-level property, stable within families and divergent between them (Tables 6.18 and 6.19). Table 6.12 consolidates the relationships that survive FDR correction, with their source tables.

Beyond those confirmatory screens, the question of *which* properties emerge is answered most directly by the pre-specified hypothesis battery of Section 6.3.9.1. Twenty directional claims linking structure to behavior were fixed before their correlations were computed and corrected together as one Benjamini–Hochberg family, so the surviving set is not a post-hoc selection: **eight are supported**, two suggestive, one significant in the direction opposite to the conjecture, three untestable on a near-constant predictor, and six unsupported. The supported set is coherent rather than scattered — three spectral-norm hypotheses (H2, H3, H4), two brittleness counters (H6, H7) and three compression-delta hypotheses (H15, H16, H18) — and H18, the change in `rand_distance` tracking the change in accuracy ($r = +0.84$, $q < 0.001$), is the strongest single association anywhere in the experiments. The comprehensive screen of Section 6.3.9.2 then shows that the double dissociation is a property of *blocks* of metrics rather than of the particular three chosen for the headline analysis: a spectral-richness block indexes capability ($r = +0.87$ to $+0.94$) with no association to robustness ($|r| \leq 0.21$), and a brittleness block does the reverse, with no leakage in either direction. Finally, Section 6.3.9.7 re-reports the central screen under both Pearson and Spearman, so that the conclusions above can be checked for dependence on linearity; they agree throughout on capability and safety, and diverge systematically only on the perplexity deltas, where the rank coefficient is uniformly stronger — the expected signature of a monotonic association in a heavy-tailed outcome, and itself a finding rather than a discrepancy.

Every structure–behavior relationship that survives correction, with its exact Pearson r , Fisher 95% confidence interval, raw p , FDR-adjusted q and the n it rests on, is consolidated in Table 6.12, grouped by the correction family within which each was tested; the 44 entries there are the complete evidential basis for the answers to RQ3 and RQ4, and anything absent from it is not significant after correction however large its coefficient.

The demanded edge-case conditions also emerge, and are of three kinds. *Thresholds*: the qualitative 2-bit capability break (Section 6.3.2), beyond which degradation ceases to be a continuation of the 4- and 3-bit trend. *Localizations*: spectral distortion concentrates at the output layer, where the mean $|\Delta\alpha|$ is roughly an order of magnitude above the first layer, though the signed movement there is outlier-driven and not uniform across the cohort; toxicity-safety loss localizes instead to the mid-network fourth quintile, the sole depth cell surviving joint correction; and the coupling of encoding-based prompt injections to $\Delta\text{rand_distance}$ is attack- and metric-specific, verified not to be a detector-multiplicity artifact (Section 6.3.7, Appendix B.2.2) [77, 133]. *Operating intervals*: the analysis surfaced one relationship that no correlation coefficient can express, since adversarial robustness appears *concave* in the mean log-spectral-norm rather than monotonic in it, peaking at an interior optimum of ≈ 1.46 (Section 6.3.9.5). That would be an edge-case condition in the strict sense — robustness worst at *both* extremes of spectral scale — but it is reported as a candidate rather than an established interval: the predictor is family-determined ($\eta^2 = 0.998$), so the fit rests on seven independent values and the quadratic term does not reach significance on the family means ($p = 0.099$).

Furthermore, Instance Space Analysis [134, 135], is provisioned in the architecture with its configuration contract already in place (Table 6.10), but is not yet implemented. Additionally, the 28 states of the

experiments are not yet enough to populate a meaningful instance space[134]. Without a doubt, ISA is the next natural analytical step. In particular, because of the already mentioned property: how the framework is extensible. In future work, as further model states and Stages from additional observational perspectives enrich the datasets of RQ2, the increasing instance population is the setting in which ISA’s footprint projection becomes informative (Section 7.3)[134].

Answer to RQ3: Yes, and the relationship is stronger and more structured than an aggregate analysis reveals. HTSR and SETOL properties relate to behavior along two blocks of metrics that do not overlap — spectral richness with capability, brittleness counters with adversarial robustness — a dissociation confirmed by a pre-specified twenty-hypothesis battery corrected as one family and checked under both Pearson and Spearman. Resolved per attack rather than per aggregate, structure indexes robustness in the majority of probes, and the *sign* of each coupling is fixed by the attack family while the metric *kind* that carries it is family-dependent, which is why pooling appears to erase the signal. Edge-case conditions emerge in three forms: a capability threshold at 2-bit, depth localizations of capability and toxicity damage, and one candidate non-monotonic operating interval in which robustness appears to peak at an intermediate spectral norm, reported as directionally consistent but not established at current experimental cohort size. The complete evidential basis is Table 6.12, which lists every relationship surviving FDR correction with its exact statistics and the n it rests on; Sections 6.3.9.1–6.3.9.4 establish each in turn, and the exploratory status of the wider screens is set out in Section 6.3.9. Instance Space Analysis is provisioned as a plug-in option but not implemented, and is planned to complement this analysis as the datasets grow (Section 7.3).

6.5.4 RQ4. To what extent can these analytical modules predict or explain the degradation of a generative agent’s performance and security under quantization, specifically its resilience to critical OWASP Top 10 LLM vulnerabilities in future Fog–Edge multi-agent settings?

The extent is established in both directions, and the asymmetry between them is itself a result. *Descriptively*, degradation is characterized specifically. Capability declines gradually and breaks at 2-bit, with the loss concentrated in multi-step reasoning (math -21.6 pp at Q2_K_L, $n = 7$ families; Table 6.14). Aggregate resilience against the OWASP-mapped battery — 39 distinct probes over 57 probe $\times$ detector pairs, 38 carrying LLM01:2025, one LLM04, and the two ANSI probes additionally LLM02/LLM05 (Table B.3) — is essentially invariant ($+0.007$, $n = 21$), with DEFCON severity tiers unchanged at every level, but the invariance is a redistribution rather than a null. **Six named LLM01 defenses degrade** — `dan.DanInTheWild`, `LI.LatentWhois`, `sata.MLM`, `dra.DRA`, `LI.LatentJailbreak` and `LI.TranslationEnFr` — while thirteen probes improve, and the movement concentrates at 2-bit (Tables 6.23 and 6.22).

The structural account explains part of that mechanism and, informatively, fails to explain the rest. Quantization noise flattens the spectrum concurrently with capability loss and concentrates perturbation at the output layer; where compression pushes a family toward random-like spectra, robustness to encoding and obfuscation injections falls ($\Delta_{\text{rand_distance}}$ couplings, $\rho \approx -0.7$, $q \approx 0.03\text{--}0.04$, $n = 21$). Against that, three of the six degrading defenses have *no* structural predictor in either regime. One subgroup deserves separate statement because it inverts the aggregate picture: `log_spectral_norm` is uncoupled from the overall pass rate (-0.35 , n.s., $n = 28$) yet reaches -0.61 ($p = 0.0006$) on the two output-handling probes alone, and -0.72 ($p = 0.0003$, $n = 21$) as a delta, alongside α_{weighted} at -0.66 and n_{trap} at -0.60 in the same subgroup. Since `log_spectral_norm` rises with capability, **more capable models are the less output-handling-robust ones** — a genuine OWASP-category-specific trade-off that the aggregate, which correlates with LLM01 at

$r = 0.97$, cannot express. For a question framed around resilience to *specific* OWASP vulnerabilities, that is the operative finding: the extent of the modules' explanatory power is category-dependent, and reporting only an aggregate would have concealed both the degrading defenses and the one category where structure predicts robustness well.

The same point holds at the finest resolution the data supports, and it sets the true extent of the modules' explanatory reach. Per attack rather than per category, structure indexes robustness in 26 of 36 testable probes (69 of 216 cells surviving correction, $n = 28$), with the coupling's sign fixed by the attack family and its carrying metric fixed by the family's kind — shape metrics for jailbreaks and latent injection, richness and scale metrics for encoding and obfuscation (Section 6.3.9.4). **The extent of the analytical modules is therefore far greater per attack than per aggregate**, and the aggregate's apparent structural independence is a consequence of pooling families whose couplings have opposite signs. For a research question framed around resilience to *specific* OWASP vulnerabilities, that is the operative conclusion: the modules explain vulnerability well when the vulnerability is named, and poorly when it is averaged.

The same resolution also sets the limit in the predictive direction, and does so consistently with the aggregate finding. On compression deltas the per-attack screen retains **14 cells across 6 of 36 probes** against 69 across 26 at the level (Table 6.40), with survivors confined to three attack families. The extent asked for by RQ4 is therefore asymmetric at every resolution examined: **descriptive and cross-sectional explanatory power is substantial, and predictive power over compression-induced damage is limited to a minority of named defenses**.

On the predictive side the extent is constrained, and the constraint is itself informative. Only $\Delta_{\text{rand_distance}}$ significantly tracks compression-induced adversarial robustness change ($r = -0.47$, $p = 0.030$, $q = 0.049$); $\Delta\alpha_{\text{weighted}}$ ($r = -0.33$, $p = 0.141$) and $\Delta\alpha$ ($r = -0.04$) do not. No single metric explains more than about a quarter of the variance (Table 6.30), and for the global-mean metrics analyzed, a model's full-precision spectrum indexes how good it is, not how much quantization will cost it (Table 6.26). Two further boundaries belong to this answer. First, the compression regime is legible in the spectrum *asymmetrically*: Δ_{entropy} , $\Delta_{\text{log_spectral_norm}}$ and $\Delta_{\text{rand_distance}}$ track capability loss strongly ($|r| = 0.56$ to 0.86 , $q < 0.05$; Table 6.34), while *no* structural metric tracks the toxicity-safety loss even though that loss is the most reliable directional effect in the expe (20 of 21 variants). Second, at $n = 21$ the detection threshold for $p < 0.05$ is $|r| \approx 0.43$, so a null in the delta regime means undetectable at this cohort size rather than demonstrably absent. These extents provide the deployment guidance the question targets (Table 6.31), and the capability–safety decoupling mandates independent adversarial evaluation of every compressed state. As context, the corpus is specifically in the size class in which community post-training quantization is predominantly produced and consumed on consumer and edge hardware (Section 6.1) [3, 153]; live multi-agent validation, the remaining OWASP categories, and further formats and calibration regimes are enumerated in Sections 7.2 and 7.3.

Answer to RQ4: To a considerable degree descriptively, and to a limited degree predictively. The modules explain the magnitude, the OWASP-mapped adversarial profile and the structural mechanism of quantization-induced degradation, and yield Pareto deployment guidance (Table 6.31). Prediction from full-precision structure is *not* supported for the metrics analyzed; among compression deltas only $\Delta_{\text{rand_}}$ -distance tracks adversarial-robustness change ($r = -0.47$, $q = 0.049$, $n = 21$), while $\Delta\alpha_{\text{weighted}}$ and $\Delta\alpha$ are null at this cohort size. The legibility is asymmetric — capability loss has a clear structural footprint, toxicity-safety loss has none — which is the boundary the question asks to be established. Two **exploratory** extensions sharpen that boundary without moving it: the relationship between spectral norm and robustness appears *non-monotonic*, with an interior optimum that a linear screen cannot represent — a pattern consistent in direction across every subset examined but not established at seven independent predictor values (Section 6.3.9.5), **and a pre-specified twenty-hypothesis battery** isolates which structural quantities carry the explanatory load (Section 6.3.9.1). Both are verified against the experimental data, and both require a larger, format-diverse cohort before being asserted beyond it (Section 7.3).

6.5.5 RQ5. By synthesizing RQ1–RQ4, how can the framework contribute, through concrete exploratory and deployment insights, or future work, to trustworthy agents in decentralized Fog–Edge systems within the NANDA Web of Agents?

RQ5 asks how the framework can contribute to trustworthy agents within the NANDA Web of Agents. Because no NANDA index, registry or agent-to-agent exchange was deployed in this work, the question is answered under a deliberately restrictive standard: **only the data the framework produced and the insights derived from that data can verify RQ5**, and each must be mapped to one of the five operational pillars this dissertation names — AgentFacts, Identity, Discovery, Trust and Communication (Sections 1.4 and 3.5) [18, 19]. Statements about how NANDA’s own mechanisms would behave once such data is attached to them are *designs* informed by the published specifications, not findings of this work, and are reported separately below and in Section 7.3. Table 6.46 gives the mapping and its evidential status pillar by pillar; two of the five carry no evidence from this dissertation, and saying so is what makes the other three credible.

The primary insight is confirmatory rather than exploratory, and it is what the rest of the answer rests on. Capability and adversarial safety are statistically decoupled across the cohort (Table 6.45): benchmark accuracy does not index red-team robustness ($r \approx -0.03$, $n = 28$, n.s.), and aggregate robustness barely moves under compression ($0.738 \rightarrow 0.745$, $\Delta = +0.007$) even as accuracy falls by up to 5.24 pp and reasoning-heavy tasks lose 21.6 pp at 2-bit, while intrinsic toxicity-safety degrades monotonically in 20 of 21 compressed states. Trust therefore cannot be inferred from a model’s benchmark scores, nor inherited from its family: it must attach to the evaluated *state*. That is a property of the measurements, established independently of any deployment, and it is the finding that motivates state-indexed records in the first place.

Designs that follow, and are not part of this verification. Beyond the mapping above, the measured data suggests four ways it could enter NANDA’s published mechanisms — agents resolving to cryptographically verifiable AgentFacts that distinguish self-asserted claims from third-party attestations, meta-data resolution supporting trust scoring, and sub-second revocation propagation [19, 18, 17]. **None of the four was implemented or exercised in this dissertation**, so none of them verifies RQ5; they are stated because the question explicitly admits future work as a channel of contribution, and because naming them makes the boundary between what was measured and what was designed inspectable.

Table 6.45: Evidence extract: why trust must attach to the evaluated state rather than the model family (Figure 6.14, Sections 6.3.8 and 6.3.9.6). Pass rates are the record-mean construction of Table 6.20; the probe-balanced aggregate of Table 6.23 is a different weighting of the same measurements and the two are not interchangeable (Section 6.3.4).

Evidence	Measure	Value
Capability $\leftrightarrow$ safety	accuracy vs. red-team pass rate	$r \approx -0.03$ ($n = 28$, n.s.)
Robustness under compression	FP16 vs. compressed pass rate	$0.738 \rightarrow 0.745$ ($\Delta = +0.007$)
Capability under compression	accuracy at Q4 / Q3 / Q2	$-0.56 / -1.54 / -5.24$ pp
Worst-affected domain	math at 2-bit	-21.6 pp
Intrinsic safety under compression	toxigen at Q4 / Q3 / Q2	$-0.022 / -0.066 / -0.099$
Structural family stability	α , variance explained by family	$\eta^2 = 0.956$
Same-model signature	L2 within-parent vs. between-family	0.30 vs. 2.83 ($p = 1.6 \times 10^{-8}$)
Bit-width recovery	leave-one-out 1-NN on six structural deltas	$21/21 = 100\%$ (chance 33%)

- **Evidence-grounded attestations.** The state-indexed record of Table 6.46 could enter the capability-assertion and audit fields as a third-party attestation, mitigating the single-metric evaluation bias and contaminated leaderboards of Section 3.3 [11]. *Unverified:* whether such an attestation is accepted, weighted or acted upon by a resolver.
- **Non-repudiable evaluation records.** Signing the auditable, configuration-traceable run record (Section 5.1.2) into that attestation would prevent a publisher from later disowning the evaluated behavior of a deployed state. *Unverified:* no signing scheme was implemented; the traceability of the run record is what this work demonstrates, not its non-repudiability.
- **Per-state screening at registration.** Re-evaluating each state at registration or resolution, rather than inheriting trust from lineage, could screen quantization-activated attacks [24], loosely scrutinised artifacts at hub scale [23], merged models [37] and fine-tuned descendants of benign parents. *Unverified:* the screening cost, its false-positive rate, and whether it detects a deliberately tampered artifact.
- **Structure-informed adaptive red-teaming.** Because the $\Delta_{\text{rand_distance}}$ couplings are attack-specific rather than uniform (Section 6.3.7), a structural pre-screen could indicate *which* attack classes a state has become vulnerable to and prioritize probing accordingly. *Unverified:* the pre-screen’s precision and recall as a decision procedure, as opposed to the underlying correlations, which are measured. The same map read as a search objective rather than a filter would support synthesizing probes tuned to a target state’s spectrum, a direction developed in Section 7.3.3.

Two illustrative scenarios ground these designs in the published mechanisms without adding to the verification. In the index’s metadata-resolution workflow, a requesting agent resolves a candidate delegate’s signed AgentFacts, including audit results, for trust scoring before interaction [19]; with the present evidence attached, that resolution could distinguish a 4-bit state whose adversarial audit passed from a 2-bit state of the same family carrying the capability-break flag, so delegation would attach to the evaluated state rather than the family name. And because the index specifies sub-second revocation [19], a benign parent’s quantized artifact found to activate dormant malicious behavior [24] could have that state’s attestation

downgraded and propagated without touching the parent’s record — something lineage-based reputation alone cannot express. Both scenarios describe how the mechanisms are specified to behave, not how they were observed to behave here; validating either requires the live Fog-Edge deployment identified in Section 7.3 [20].

Answer to RQ5: Under the restrictive standard that only produced data and derived insights can verify this question, the framework contributes to three of the five operational pillars, partially to a fourth, and not at all to the fifth (Table 6.46); the contribution draws on *all three* measurement perspectives rather than on structure alone. **AgentFacts** is served by the state-indexed record itself — capability, OWASP-mapped adversarial profile with DEFCON severity, intrinsic toxicity-safety, and spectral structure, per state ($n = 28$). **Identity** is served by a data-free structural signature ($p = 1.6 \times 10^{-8}$; 100% bit-width recovery), **exploratory** and bounded to one format with no adversarial artifact tested, and by the behavioral evidence that a compressed state differs from its parent on axes structure alone would not reveal. **Trust** is served by three insights that jointly determine what a trust attribute may assert: capability and adversarial safety are decoupled, safety is two-axed with the axes moving oppositely under compression, and structure describes quality without forecasting compression damage, with the exception when a more granular and specific adversarial analysis is performed. **Discovery** is addressed only at the level of attributes that could condition it, including the **exploratory** attack-specific couplings that indicate which attack classes a state is exposed to, since no registry was deployed. **Communication** is not addressed. Four integration designs follow and are stated explicitly as unverified; all four, and the two under-served pillars, are carried to Section 7.3.

In summary, the five Research Questions are answered within the scope their formulations define: RQ1 and RQ2 by the operating framework and the multi-perspective dataset it produced, RQ3 through its statistical instruments, RQ4 by establishing the descriptive and predictive extent of the analytical modules, and RQ5 through one exploratory deployment insight and based on candidate integration paths into NANDA framework and roadmap [19, 17] . Every identified extension is delimited above and prioritized in Section 7.3.

6.6 System Requirements Validation

Section 4.3.6 detailed the functional and non-functional requirements that guided the development of ChaosAthena under the MoSCoW prioritization. This section evaluates the extent to which the ChaosAthena framework satisfies them, following every claim to the implementation (Chapter 5) or to the experimental evidence (Sections 6.2 and 6.3). Three categories are used. **Implemented:** fully realized end-to-end and exercised in the experimental evaluation. **Partially Implemented:** functionally available, but not yet covering the entire requirement specification, or not exercised over the full model corpus. **Future Work:** a planned extension point, attachable through the existing Stage contracts and thereby demonstrating the extensibility of the framework, but not implemented in the delivered artifact. A requirement is placed in a category only where Chapter 5 exhibits the corresponding implementation and, for the first two categories, Section 6.3 exhibits a result it produced; a requirement for which neither can be pointed to is reported as Future Work regardless of how partially it was explored.

6.6.1 Functional requirements

Table 6.48 summarizes the outcome for the 21 functional requirements of Table 4.1, which comprise 8 *Must*, 1 *Should* and 12 *Could* items.

Every *Must* requirement is validated by the results it produced. The Performance Evaluation Stage (Section 5.4.1) executed `lm-evaluation-harness` in the isolated vLLM subprocess and delivered the accuracy, perplexity-differential, and MCQ KL-divergence results of Section 6.3.4, fulfilling FR1. The Red Team AI Evaluation Stage (Section 5.4.2.1) attack every state through the temporary OpenAI-compatible vLLM endpoint with the probe set mapped to the OWASP Top 10 for LLM (Table B.3), producing the robustness analysis of Section 6.3.4 satisfying FR2. The Deep Learning First-Principles Stage (Section 5.4.3) extracted data-free HTSR/SETOL diagnostics — dequantizing compressed states in memory — for all 28 evaluated states (Section 6.3.3), meeting FR3. The compression stage fulfills FR4 through its retrieve-or-compute rule (Section 5.3.2): community-quantized GGUF K-quant artifacts were registered from Hugging Face, and local generation `llama.cpp` is demonstrated by the integration of `llama.cpp` end-to-end when dealing with the retrieved gguf format models from Hugging Face, and because for experiments, only retrieval models from Hugging Face were used, as detailed in Section 6.2, an example of compressed model exported to Hugging Face is provided in Appendix B, Section B.4. The Dataset Creator (Section 5.5.1) aggregated all Phase 2 artifacts into the multidimensional delta-aligned dataset (FR5), on which the Veritas Analytics stage (Section 5.6) calculated the correlation, delta, performed inferential statistics (e.g., BH-FDR), and explored outliers whenever relevant, were identified in Sections 6.3.5, 6.3.8, and 6.3.6 (FR6). The `all:` sweep scope of the Hydra dispatcher (Section 5.1.2) drove single-execution sweeps over seven families and 21 compressed variants (Table 6.11), meeting FR7, and the Phase 4 consolidation that generated every table and figure of Section 6.3 in reproducible, configuration-traceable form satisfies FR8.

The single *Should* requirement closes as Partially Implemented. The guardrail module (FR9) is implemented and tested within the Red Team AI Evaluation Stage following the GuardReasoner methodology (Section 5.4.2.2) [75], but its execution over the full model corpus lies outside the experiments of this dissertation, so the requirement is not exercised end-to-end — a limitation discussed in Section 7.2 and given first priority in Section 7.3.

All twelve *Could* requirements (FR10–FR21) are Future Work, and the accounting here is deliberately conservative. Neither the Complexity & Chaos Theory module (FR10) nor the XAI intermediate-representation and token-trajectory visualization (FR11) is implemented in the delivered framework [129]; no textual-complexity measurement and no intermediate-representation instrumentation appears in Chapter 5, and neither contributes any evidence to Section 6.3. They are therefore reported as unrealised extension points rather than as proofs of concept, on the principle that a requirement should be claimed only where an artifact and a result can be pointed to. That does not weaken the extensibility argument, which rests on the contract rather than on partial implementations: each of the twelve attaches through the existing Stage interfaces (NFR1, NFR8) without orchestrator changes, and each appends columns to the same state-indexed schema, which is what keeps the extensions cumulative. Among them, Instance Space Analysis (FR12) and the efficiency and sustainability telemetry (FR13) carry the highest analytical value and are prioritized accordingly in Section 7.3.

6.6.2 Non-functional requirements

Table 6.49 summarizes the outcome for the 13 non-functional requirements of Table 4.2.

The architectural requirements are validated by construction and by operation. Stages communicate only through the configuration-in, artifact-out contract of the storage substrate (Section 5.1.3), with independent CLI, unified-dispatcher, and notebook entry points (Appendix A, Section A.2, meeting NFR1; every run is composed from layered Hydra YAML groups with CLI overrides and archived as a `hydra.run.dir` record (Section 5.1.2), satisfying NFR4 and NFR5; and the Strategy, Factory, Façade, and Adapter structures of Section 5.1.5 realize the additive extensibility of NFR8 and the chat-template resolution of NFR3. The

executed matrix validates the remainder: sweeps over seven families with automated Hugging Face acquisition (NFR2), the pip-installable package with isolated backend dependencies (NFR7, Section 5.1), hardware agnosticism across GPUs, a constrained GPU tier, and CPU-only nodes (NFR9, Table 6.2), cloud-synchronized artifact persistence (NFR6), and per-Stage notebooks with absolute code-path parity to the CLI (NFR12, example Appendix A), section A.2.

Process isolation (NFR11) is ensured by construction and was exercised by every reported run: all heavy inference is dispatched to isolated subprocesses — the vLLM subprocess of the Performance Evaluation Stage (Section 5.4.1) and the temporary endpoint of the Red Team Stage, terminated immediately upon completion to release GPU memory (Section 6.2.5.2)— enforcing strict memory boundaries that prevent residual artifacts from polluting the orchestrator or subsequent evaluations, while the per-state tracking files make sweeps resumable after any failed run, thus satisfying NFR11.

NFR10 is partially implemented, with substantive alignment already in place. The DAMA-DMBOK areas of data integration, metadata management, and data quality are implemented through, by default, raw execution logs to the `final.parquet` and `csv` datasets (Section 5.5.1); Completeness (the explicit “Not Applied” versus “not measured” distinction), Consistency (the uniform group/model identifier scheme across all joined sources), and Traceability, and operationalizes the DAMA-DMBOK areas of data integration, `.parquet` and `.csv` datasets (Section 5.5.1); however, the remaining ISO/IEC 25012 characteristics were not formally measured, nor were the other DAMA-DMBOK knowledge areas systematically assessed – the scope of this limitation is analyzed in Section 7.2.

Finally, the microservice version (NFR13) is future work confined, by storage-mediated design, to transport-layer changes. In summary, ChaosAthena implements **all eight functional *Must* requirements and nine of the ten non-functional *Must* requirements**, with NFR10 partial and explicitly circumscribed to the unmeasured ISO/IEC 25012 characteristics. Of the three *Should* requirements across both tables, two are implemented (NFR11, NFR12) and one is partial (FR9). All thirteen *Could* requirements (FR10–FR21 and NFR13) remain open; every one of them is an additive extension point attachable through the existing contracts rather than an architectural debt, which is the sense in which the extensibility claim of RQ1 is validated by the requirement set as much as by the experiments. No requirement is claimed at a level of completion that Chapters 5 and 6 cannot evidence.

Table 6.46: What each of the three measurement perspectives contributes to each NANDA operational pillar, and the evidential status of that contribution. **Measured** = a quantity produced and reported in Section 6.3, with its n given. **Insight** = a conclusion derived from those measurements with no additional apparatus. **Exploratory** = verified against these data but hypothesis-generating and cohort-bounded (Section 6.3.9). **Not addressed** = no data bearing on the pillar was produced. Mechanism-level integration — attestation signing, registry resolution, revocation propagation — appears in no row, because none of it was implemented here; it is deferred to Section 7.3.

Pillar	Perspective	Result that can inform the pillar	Status
AgentFacts	Capability	Per-state accuracy over nine tasks, WikiText perplexity, and the domain breakdown that localizes loss to multi-step reasoning ($n = 28$ states)	Measured
	Safety	Per-state OWASP-mapped pass rate over 39 probes with worst-case z and DEFCON severity, plus the intrinsic toxigen score ($n = 28$)	Measured
	Structure	Per-state HTSR/SETOL descriptors and their compression deltas ($n = 28 / n = 21$)	Measured
Identity	Structure	Same-model signature: within-parent L2 0.30 vs. between-family 2.83, non-overlapping, $p = 1.6 \times 10^{-8}$; bit-width recovery 21/21 by leave-one-out 1-NN, permutation $p = 0.0005$; α family-stable, $\eta^2 = 0.956$	Exploratory ^a
	Safety	Per-state re-evaluation distinguishes a state from its parent behaviorally where structure alone would not, e.g. a 2-bit variant carrying the capability-break flag ($n = 28$)	Insight
Trust	Capability $\times$ Safety	Capability and adversarial safety are decoupled ($r \approx -0.03$, n.s., $n = 28$): a trust attribute cannot be derived from a capability attribute	Insight
	Safety	Safety is two-axed and the axes move oppositely under compression — adversarial robustness holds (+0.007), toxigen falls in 20/21 variants — so a single safety attribute would be misleading	Measured
	Structure	Structure indexes capability strongly but does <i>not</i> forecast per-model compression damage, so a structural attribute must be published as description, not as a warranty	Insight
Discovery	Safety	DEFCON tier and per-probe OWASP profile are per-state attributes on which routing could be conditioned; the decoupling establishes why routing on capability alone would be unsound	Insight only ^b
	Structure	Attack-specific $\Delta_{\text{rand_distance}}$ couplings indicate <i>which</i> attack classes a state is structurally exposed to, a cheaper prioritization signal than full red-teaming ($n = 21$)	Exploratory ^b
Communication		Nothing. No agent-to-agent protocol, message exchange or handshake was implemented, instrumented or measured; the unit of analysis throughout is the model state, not the running system	Not addressed

^a Exploratory and bounded: 21 compressed states of 7 families in one quantization format, with no adversarially constructed or deliberately mislabelled artifact tested and no false-positive rate established at a fixed threshold. The signal is demonstrated; a detector is not (Sections 7.2.3 and 7.3). ^b The antecedent (the attributes exist, and the couplings are measured) is verified; the consequent (conditioning discovery on them improves an outcome) is untested, because no registry was deployed. Neither Discovery row contributes to the verification of RQ5.

Table 6.48: Validation of the functional requirements (Table 4.1).

Status	Requirements	MoSCoW
Implemented	FR1–FR8	8 Must
Partially Implemented	FR9	1 Should
Future Work	FR10–FR21	12 Could

Table 6.49: Validation of the non-functional requirements (Table 4.2).

Status	Requirements	MoSCoW
Implemented	NFR1–NFR9; NFR11, NFR12	9 Must, 2 Should
Partially Implemented	NFR10	1 Must
Future Work	NFR13	1 Could

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Community hubs now distribute generative models in many states — compressed, merged, or fine-tuned — that are consumed far from the scrutiny applied to their full-precision parents, precisely as agentic deployment pushes capable models toward constrained hardware. In response, this dissertation designed, implemented, and validated ChaosAthena, a multi-perspective framework for the comprehensive analysis of Gen AI model states, and applied it to post-training quantization as its case study. The work progressed from a four-phase, storage-decoupled architecture (Chapter 4), through its implementation as interchangeable evaluation Stages operating under gray-, black-, and white-box regimes (Chapter 5), to a fully crossed experimental study over seven model families at four precision levels — 28 states in total, with complete three-perspective coverage of every one (Chapter 6).

The empirical evidence is coherent and, in its most consequential parts, converges with independent work. Capability degrades gradually under 4- and 3-bit quantization and breaks qualitatively at 2-bit, with the loss concentrated in multi-step reasoning; aggregate adversarial resilience is essentially invariant, with movement confined to specific probes; and capability and adversarial safety are statistically decoupled, so neither can proxy the other. Two of these findings replicate patterns obtained by different means: the concentration of loss in multi-step mathematical reasoning matches what Li et al. report across eleven model families [137], and the identification of 4-bit as the safe operating point, with degradation appearing below it, agrees with the trustworthiness evaluation of Hong et al. [136]. Produced on different models, formats, and metrics, those two lines of evidence strengthen the deployment guidance beyond what a single cohort of 28 states could support alone.

The structural perspective supplies the mechanism, and it is asymmetric in a way that is itself the result. Spectral entropy tracks capability concurrently rather than predictively; the heavy-tail exponent α behaves as an architecture-family property, stable within families and divergent between them ($\eta^2 = 0.956$); quantization distortion concentrates in magnitude at the output layer; and $\Delta_{\text{rand_distance}}$ is the one structural change that tracks adversarial-robustness change. Against that, *no* structural metric tracks the loss of intrinsic toxicity-safety, even though that loss is the most reliable directional effect measured here — 20 of 21 compressed states become less safe. Compression is legible in the spectrum when it costs capability and largely illegible when it costs safety, which is a more precise and more useful statement than either "structure explains degradation" or "it does not".

Three further results emerged from systematically screening the multi-perspective dataset rather than

from the questions the study was designed around, and they are the ones most likely to outlive this cohort. A pre-specified battery of twenty hypotheses, corrected as a single family, returns eight supported claims whose survival is not a post-hoc selection. Adversarial robustness appears *concave* in the mean log-spectral norm, peaking at an interior optimum — the one relationship found that no correlation coefficient can express, though the predictor proves to be family-determined and the pattern is reported as directionally consistent rather than established. Moreover, the spectral shape space carries model provenance: a compressed state sits an order of magnitude closer to its own FP16 parent than two parents sit to each other, without overlap, and the quantization level is recoverable from six structural deltas alone at 100% leave-one-out accuracy. All three are computed data-free, from weights, with no inference and no prompts.

The evidence remains correlational, bound to one compression format, one architecture class, and 28 states; larger and more diverse cohorts are required before any of it is asserted as a general law, and the provenance results in particular have not been tested against a deliberately tampered artefact. What the dissertation delivers is not a set of universal constants but a tangible artifact and a reproducible method, together with a claim that the method makes concrete: **each compressed state can, and methodologically should, be assessed as a distinct state in its own right**, because trust is neither inherited from a family nor inferred from a benchmark score. The state-indexed evidence so generated is precisely the primary data that trust infrastructures such as NANDA’s AgentFacts are being designed to carry, and from which they can derive both informational and epistemic value.

7.2 Limitations

The methodological decisions that secured the reproducibility of this dissertation are the same ones that bound the validity of its claims. This section states each bound explicitly, in the order in which it constrains the results, and names the experiment that would relax it; every one of them reappears as a prioritized extension in Section 7.3. The findings should be read as characterizing this specific regime rather than as general laws of compression — the framework is constructed to generalize, and establishing the extent to which the findings do is precisely what the broader cohorts outlined below are for.

7.2.1 Generation regime and measurement windows

Generation was fully deterministic — temperature 0, top- p 1, two generations per adversarial probe — whereas deployed models operate under heterogeneous sampling configurations [161]. The adversarial, capability, and toxicity results would therefore benefit from systematic replication across a range of temperature and top- p settings, which is the only way to establish whether the probe-level redistribution reported in Section 6.3.4.2 is a property of the compressed states or of the decoding regime under which they were probed.

On the red-team side, garak’s detectors analyze a bounded window of each response (200 characters). Since compressed models tend to answer more verbosely [128], harmful content falling beyond that window may be undercounted — a truncation effect that biases the pass rate *upward* for exactly the states most likely to be degraded, and that therefore bears directly on the headline finding that aggregate robustness holds. A dedicated study with longer detector windows is required before that finding is treated as settled.

Two conventions must also be read as bounds rather than as measurements. The intrinsic safety score is treated throughout as *higher = safer*, following the Performance Evaluation Stage; ToxiGen is elsewhere reported as a toxicity score, in which case every sign statement involving it inverts. And the aggregate adversarial pass rate admits two defensible constructions — a record-mean over all red-team records and

a probe-balanced mean over the 39 distinct attacks — which weight the probe set differently and are not interchangeable (Section 6.3.4). Both are reported, they agree qualitatively (+0.007 against +0.006 mean change under compression), and every aggregate quoted in this dissertation states which one it uses; but a reader comparing against literature that does not state its construction should expect a discrepancy of that order for reasons that are arithmetic rather than empirical.

7.2.2 Statistical Power and the robustness of inferential conclusions

The cohort of 28 states across seven families gives the correlational analyses limited statistical strength, and supports *association, not causation*. Concretely, at $n = 28$ the two-sided threshold for $p < 0.05$ is $|r| \approx 0.37$, and at $n = 21$ for the compression deltas it is $|r| \approx 0.43$. Every null result reported in this dissertation therefore means *undetectable at this cohort size*, not demonstrably absent — a distinction that matters most for the negative results the argument rests on, notably the absence of any structural correlate of toxicity-safety loss (Section 6.3.9.2) and the null coupling of $\Delta\alpha_{\text{weighted}}$ to robustness change (Section 6.3.7).

Three specific limitations follow. First, near-binary indicators are the weakest evidence in the study despite carrying some of the largest coefficients: the trap-layer count **is the strongest single predictor of adversarial robustness in the cohort** ($r = -0.57^{**}$, ahead of every continuous shape metric), yet trap layers are present in only 6 of the 28 states, so the association rests on a split that a differently composed cohort could dissolve. Second, one result is coefficient-dependent: the depth localization of toxicity-safety loss to the fourth quintile survives joint correction under Pearson ($r = -0.69$, $q = 0.01$) but is borderline under Spearman ($\rho = -0.60$, $q = 0.05$), where none of the 27 depth cells survives. It is reported as indicative and should not be carried further without a larger delta cohort. Third, one result is outlier-driven: the growth of spectral perturbation toward the output layer is a statement about *magnitude*, and a single variant — Llama-3.2-1B at Q2_K_L, whose last-layer α collapses from 8.98 to 4.07 — accounts for much of it; the signed movement at that position is not consistent across the cohort (38% of variants rise).

Finally, the exploratory analyses of Section 6.3.9 are hypothesis-generating by construction. Benjamini–Hochberg control is applied within each named family, and no result is asserted on a raw p [77, 133], but multiplicity control bounds the false-discovery rate; it does not confer the evidential status of a pre-registered confirmatory test on a screen of this size.

An asymmetry runs through every resolution of the analysis and is better stated as a limitation than left as a pattern. Cross-sectional (level, $n = 28$) associations are consistently far stronger and more numerous than compression-delta ($n = 21$) associations: for the aggregate metrics, capability loss has a clear structural footprint while safety loss has none; and per attack, 26 of 36 probes couple to structure at the level against 6 of 36 under compression. Part of that gap is power — the detection threshold rises from $|r| \approx 0.37$ to $|r| \approx 0.43$ — but the surviving count falls by a factor of five, which power alone does not explain. The honest reading is that this dissertation establishes weight structure as a *descriptor of model states* far more securely than as a *predictor of compression damage*, and that every forward-looking use of the structural results in Section 7.3 inherits that boundary.

7.2.3 Compression axis and architectural scope

The compression axis covers the GGUF K-quants (K_M and K_L recipes in 4-, 3-, and 2-bit) without calibration-based modes. Higher bit-widths (8-, 6-, 5-bit), calibrated GGUF variants, and AutoRound-optimized recipes [125] remain untested, as do other model states entirely — merged models and fine-tuned descendants of shared bases, both now common on community hubs [37]. The results additionally

characterize the quantized weight states under FP16 arithmetic, that is, simulated quantization at inference [99, 101]; native integer-kernel inference on end devices may introduce small further numerical deviations, and its runtime characterization — latency, memory, energy — is deferred (Section 7.3).

All 28 states are dense decoder-only transformers, so the extensibility claim of RQ1 is validated across two architecture families and four precision levels but is, beyond that class, demonstrated by design rather than by experiment. State-space models [162], mixture-of-experts transformers [163] and recurrent alternatives such as xLSTM [164] attach through the existing Stage contracts, but no such state was evaluated.

This scope bounds the provenance results of Section 6.3.9.6 more tightly than any other finding, and the bound deserves to be stated in its own terms. The same-model signature and the 100% bit-width recovery establish that a strong structural signal exists across 21 compressed states of 7 families in one quantization format. They do not establish detector performance: no adversarially constructed or deliberately mislabelled artefact was tested, no false-positive rate was measured at a fixed threshold, and no other format was covered. Read as a detector claim, they would be unsupported; read as evidence that the signal is worth building a detector on, they are strong.

7.2.4 Structural resolution

The analysis resolves layer *depth* but not layer *type*. The first analyzed position is an attention-layer projection and the last is an MLP projection, so the finding that spectral distortion concentrates toward the output is confounded with a change of layer type at exactly that position. Type-dependent compression sensitivity, signaled in the low-rank decomposition literature by LASER [98], was not examined, and depth-by-type experiments are therefore a priority rather than an optional refinement (Section 7.3).

Two further resolution limits apply. The structural descriptors used throughout are per-layer quantities *averaged over the model*, which is what makes them comparable across architectures of different depths, but also what makes them insensitive to compensating movements at different depths — the appendix per-layer profiles (Appendix B, Figure B.6) show variation that the whole-model mean cannot express. And the associations were treated as monotonic almost throughout; the one exception, the concave relationship between the mean log-spectral-norm and robustness (Section 6.3.9.5), was found only because it was looked for, which raises the possibility that other non-monotonic structure remains untested. Its own confound is more fundamental than a question of coverage: the mean log-spectral-norm is determined almost entirely by the model family ($\eta^2 = 0.998$), so a cohort of 28 states supplies seven independent values of the predictor, and the quadratic term is not significant on those seven family means ($p = 0.099$) nor within any single precision level.

7.2.5 Data governance and deployment

Capability rests on IRT-based tinyBenchmarks subsets rather than full evaluation suites. The adversarial set is centered on prompt injection. Of the 39 distinct probes, 38 carry an LLM01:2025 label, and one carries LLM04, with the two ANSI escape-sequence probes additionally labeled LLM02 and LLM05 (Section 6.3.4.2) [26]. Consequently, the OWASP coverage exhibits substantial depth in a single category while remaining comparatively limited across the remaining categories. The per-category analysis of Section 6.3.9.3 says so explicitly, and no claim about categories outside LLM01 should be read as resting on more than one or two probes.

Formal ISO/IEC 25012 measurement of the produced datasets is partial: integration, metadata lineage, completeness, consistency and traceability are enforced by construction, while the remaining characteristics were not formally measured and the other DAMA-DMBOK knowledge areas were not systematically

assessed (Section 6.6). Finally, no live Fog–Edge multi-agent deployment was performed. That was never the object of this dissertation, whose unit of analysis is the deployment-ready model state rather than the running system; but it means the NANDA integration paths of Section 6.5 remain designs validated against the published mechanisms [19, 18, 17] rather than against a deployment.

7.2.6 The status of the exploratory results

Section 6.3.9 reports a wide screen of the multi-perspective dataset, and four of its results — the pre-specified hypothesis battery, the non-monotonic relationship between spectral norm and robustness, the structural provenance signature, and the dual-coefficient synthesis — are used in the Key Findings, in the Research Question validation, and in the Contributions. Because they are used that widely, the limitation that attaches to them must be stated in its own right rather than inferred from the general cohort limitations above.

Those results are *verified, but exploratory*. Verified, in that every value was recomputed from the source datasets, every screen names the Benjamini–Hochberg family it is corrected within, and no result is asserted on a raw p ; a result reported as supported did survive correction within its stated family. Exploratory, in three specific senses. They are **hypothesis-generating**: multiplicity control bounds a screen’s false-discovery rate but does not give it the standing of a pre-registered confirmatory test, and only the hypothesis battery — whose twenty claims were fixed before their coefficients were computed — partly avoids this. They are **cohort-bounded**: at $n = 28$ and $n = 21$ the detection thresholds are $|r| \approx 0.37$ and $|r| \approx 0.43$, near-binary predictors such as the trap-layer count rest on splits a differently composed cohort could dissolve, and one quantization format with one architecture class bounds external validity throughout. Several are **individually fragile**: the depth localization is borderline under Spearman, where no cell of its 27-cell screen survives and the surviving Pearson cell misses the rank-based threshold by 0.002; the output-layer perturbation is outlier-driven; the inverted-U rests on seven independent predictor values rather than 28, because the mean log-spectral norm is family-determined, and it loses significance in every fit that accounts for this; and the provenance signature has been tested against no adversarially constructed or mislabeled artifact, so it demonstrates a signal rather than validating a detector.

The consequence for reading this dissertation is uniform. Each of these results is **strong enough to direct further work and to be reported as a finding of this cohort**, and **not strong enough to be asserted as a general property of compression or deployed as a decision procedure**. Section 7.3.4 states, for each, the specific experiment that would settle it; until those experiments are run, every use of an exploratory result in Chapters 1 and 6 carries that qualification with it and is marked accordingly.

7.3 Future Work

The roadmap for ChaosAthena is already specified by the requirements validation (Section 6.6), which is what keeps the extensions cumulative: every new Stage appends columns to the same state-indexed schema rather than opening a parallel analysis. The directions below are grouped as engineering, analytical, hypothesis-driven, and one concrete opportunity that follows from the output-layer finding; each states what would settle it rather than merely what could be tried.

7.3.1 Engineering roadmap

The single partially realized *Should* requirement has first priority: refining the guardrail (FR9) into a fully integrated safety-evaluation perspective executed across the corpus rather than tested in isolation [75]. Doing so would close the one place where the dissertation reports a safety axis it did not measure at scale.

The two proof-of-concept requirements follow — textual complexity (FR10) and intermediate-representation and token-trajectory visualization (FR11) [141] — and then the ten open *Could* requirements (FR12–FR21) together with the open non-functional items (NFR10 and NFR13). Among these, three carry disproportionate value. Efficiency and sustainability telemetry (FR13) would supply the runtime axis the unit of analysis currently excludes, and would make the Pareto guidance of Section 6.3.8 an actual deployment recommendation rather than a capability–safety trade-off alone. Additional adversarial engines (FR14) — DeepTeam, PyRIT, and standardized attack benchmarks such as HarmBench and Jailbreak-Bench [145, 146, 165, 166] — would broaden the OWASP coverage that Section 7.2.5 identifies as thin outside LLM01, and would test whether the structure-to-vulnerability couplings found here extend beyond the encoding and obfuscation injection families [167]. Additional quantization methods (FR15), including AutoRound recipes, calibrated GGUF modes and higher bit-widths [125], would separate what is a property of compression from what is a property of *this* compression.

Completing NFR10 — formal ISO/IEC 25012 measurement of the remaining data-quality characteristics — is the governance counterpart, and matters more than its *Must*-tier status alone suggests: the framework’s contribution is a dataset whose value depends on its measured quality being stated rather than assumed [168].

7.3.2 Analytical directions

Two analytical directions become informative precisely as the cohort grows, which is why neither could be pursued here.

Instance Space Analysis (FR12) projects instances into a space in which algorithm footprints become visible [134, 135]. Twenty-eight states cannot populate such a space; the accumulation of further states and further Stages is the setting in which it becomes the natural next analytical step, and the configuration contract for it is already in place.

Complementary experiments should address the resolution limits of Section 7.2.4 directly: layer-type sensitivity (attention versus MLP, Appendix B, Figure B.6), a cohort enriched in trap layers so that the strongest robustness predictor rests on more than a 6-of-28 split, other architecture classes [162, 163, 164], replication under varied sampling temperatures, and longer garak detector windows [31]. Beyond linear and rank correlations, larger cohorts also make nonlinear model classes testable — quadratic and change-point fits of the kind that surfaced the inverted-U of Section 6.3.9.5, which a monotonic screen would have missed entirely.

7.3.3 Structure-guided adversarial probe synthesis

The per-attack screen of Section 6.3.9.4 is currently read defensively: because the sign of each structure–vulnerability coupling is a property of the attack family rather than of the metric, a structural fingerprint indicates *which* attack families a given state is most exposed to, and a pre-screen can prioritize probing accordingly (Section 7.3.4). The same map admits an offensive reading that this dissertation does not pursue and that its results make available. Treated as an *objective* rather than as a filter, the couplings define a search: given a target state’s spectral fingerprint, one could search the space of prompt transformations — encoding, obfuscation, latent framing, persona conditioning — for the transformation whose family-level

coupling predicts the largest loss of robustness for *that* spectrum, rather than drawing from a fixed battery whose probes were authored without reference to the target. Because the structural descriptors are data-free, such a search would be guided by quantities computable from the weights alone, without querying the model to discover which family it is vulnerable to.

Three properties of the present results make this concrete rather than speculative. The map is *directional*: encoding and obfuscation probes are indexed by richness quantities on which no shape metric reaches significance, whereas the jailbreak and latent-injection families invert one another on the shape metrics, so a fingerprint discriminates between families rather than merely ranking overall fragility. It is *state-resolved*: the couplings are measured per state, so two quantizations of one parent can be assigned different predicted exposures. And it is *falsifiable in a single experiment*: a synthesized probe should defeat states whose fingerprint predicts exposure at a higher rate than states whose fingerprint does not, and should outperform the closest fixed-battery probe of the same family on the former. Success would establish that structural evaluation and adversarial evaluation are not merely correlated perspectives but that the first can generate content for the second — strengthening the multi-perspective claim of this dissertation from integration to generation. Failure would bound the per-attack screen as descriptive rather than prescriptive, which is itself a result worth having.

Two conditions constrain the work. The couplings that would drive the search are **exploratory** (Section 6.3.9) and rest on seven model families and one quantization format, so a synthesis procedure built on them would need validating on a cohort in which the family-level sign pattern has been shown to replicate. And because the output is an artifact designed to defeat a specific deployed state, the work belongs inside an established red-teaming methodology and disclosure practice [74, 31], evaluated against benchmarks built for that purpose [165, 166] rather than released as standalone attacks — the same constraint under which the adversarial perspective of this dissertation was itself conducted.

7.3.4 Validating the exploratory results

The exploratory results of Section 6.3.9 are the part of this dissertation with the widest forward value and the narrowest evidential base, and both follow from the same fact: they were obtained on **28 states of 7 model families in a single quantization format**. They are verified — recomputed from source, corrected for multiplicity within named families, with nothing asserted on a raw p — and they are not confirmatory. Rather than proposing each as an independent hypothesis, it is more accurate to state the single requirement they share. Each **one of them can be explored further and fully validated only on a substantially larger and more heterogeneous cohort**, and that cohort must be larger along two axes at once because the two bound different results.

The first axis is the **number of model states**, which is what raises statistical power. At $n = 28$ and $n = 21$ the detection thresholds are $|r| \approx 0.37$ and $|r| \approx 0.43$, so the chapter’s negative results — above all the absence of any structural correlate of toxicity-safety loss, and the collapse of the per-attack screen from 26 to 6 probes when both sides are differenced — are bounded rather than established. Several near-misses sit immediately below the threshold and would resolve one way or the other with a few dozen more states, and near-binary predictors such as the trap-layer count, present in only 6 of the 28 states here, rest on splits a differently composed cohort could dissolve.

The second axis is the **variety of state kinds**, which is what establishes external validity, and it matters more than cohort size for the structural results specifically. The framework’s unit of analysis is the model *state*, and the state space is far wider than the one sampled here: other quantization formats and calibration regimes (GPTQ, AWQ, calibrated GGUF, AutoRound recipes, higher and mixed bit-widths) [99, 101, 125], states produced by pruning or low-rank decomposition rather than quantization [98, 126], merged models

and fine-tuned descendants of shared bases [37], states of non-transformer architectures [162, 163, 164], and — importantly for a framework whose claim is that any deployable artifact is a state — *compositions* of these, such as a quantization of an already-quantized or already-merged artifact. Nothing in the framework treats compression as a single step applied once to a full-precision parent, and nothing in the exploratory results has been tested against a composed state.

The results each axis would settle on are specific, and the naming is the useful part. Cohort *size* would settle whether the structural correlate of toxicity-safety loss exists below the current threshold, whether the per-attack family sign pattern replicates, whether the trap-layer counter is a usable diagnostic, and whether the apparent inverted-U optimum is a property of models at all. That last question is the most exposed of the three: the mean log-spectral-norm is almost entirely determined by the model family ($\eta^2 = 0.998$; precision level explains 0.002), so although the quadratic is fitted on 28 states it rests on seven independent values of the predictor, and it is not significant on those seven. Settling it requires a cohort with substantially more *families* spanning the low-norm region — adding further precision levels to the same families would contribute no new predictor values. Cohort *variety* would settle whether the provenance signature survives outside one format and, critically, against deliberately mislabeled or tampered artifacts, which is the difference between demonstrating a signal and validating a detector [24] or if a family-resolved structural pre-screen retains precision on attacks absent from this battery [165, 166].

Two properties of the framework make it incremental rather than a new start: the operational payoff of the architecture, validated in Section 6.6. Every additional state is added by configuration rather than by modification, so cohort growth along either axis is a matter of declaring states and re-running the pipeline. And every Stage added under FR10–FR21 appends new measurement dimensions to the same state-indexed schema, so the space of testable hypotheses grows with the framework itself: each new perspective multiplies the cross-view relationships available to exactly the kind of screening reported in Section 6.3.9, under the same multiplicity discipline. Until that program is run, the exploratory results should be read as this dissertation reads them — **strong enough to direct further work and to be reported as findings of this cohort, and not strong enough to be asserted as general properties of compression or deployed as decision procedures.**

7.3.5 Spectrally guided compression: a concrete opportunity

One opportunity follows directly from the output-layer finding and is worth stating concretely because the tooling for it already exists, and what is missing is exactly what this framework provides.

WeightWatcher implements *SVD Smoothing* [41]: a truncated-SVD transformation of selected layer weight matrices that retains only the eigencomponents associated with the largest eigenvalues — the heavy tail that HTSR and SETOL identify as the effective correlation space carrying a layer’s learned correlations — and discards the remainder [29, 30]. The retained subspace can be chosen by several criteria (a volume-preserving $\det X$ condition, a fixed percentage of the largest eigenvalues, or the eigenvalues inside the fitted power-law tail), which makes the procedure conceptually analogous to the layer-selective rank reduction of LASER [98, 169]. The published demonstration, however, is exploratory: truncated SVD applied to the MLP layers of TinyLLaMA-1.1B and validated only by a qualitative text-generation sanity check [170, 41], with **no benchmark accuracy evaluation and no adversarial or safety assessment.**

That gap maps directly onto ChaosAthena. Its multi-perspective evaluation — capability, adversarial resilience, intrinsic safety, and structure — could *guide* such truncation alongside quantization, applied before or after compression as a complementary Stage, selecting per-layer truncation targets where the HTSR and SETOL diagnostics cross their quality thresholds: an α drifting toward the band edge, or a `rand_distance` collapsing toward the randomized spectrum. The framework would then quantify what each

truncation costs or preserves along every axis, allowing different trade-offs to be chosen for concrete deployment needs [168]. The results of this dissertation supply two candidate targets, and it is worth being precise about which evidence supports which. The **output layer** is indicated because α there already sits above the well-trained band at every precision (Section 6.4.2) — a level statement, independent of compression — which makes it a candidate for protection under mixed-precision quantization or for selective truncation [107, 101, 108, 126]. It is *not* indicated by the finding that perturbation magnitude concentrates there, since that statement is about $|\Delta\alpha|$ and is outlier-driven. The **mid-deep quintile** Q_4 is the second candidate and rests on different evidence: it is where signed $\Delta\alpha$ moves coherently across the cohort (90% of states rising) and where the one depth $\times$ outcome coupling that survives correction is located. Testing both and separating them is exactly the experiment the framework is positioned to run.

A practical limitation of the current implementation is itself an engineering opportunity. SVDSmooth-ing reduces the *effective rank* of the treated matrices but reconstructs them at their original dimensions, so parameter count and on-disk size are unchanged [41]. Realizing actual storage and memory savings — by persisting the factorized low-rank form, or by coupling truncation with quantized storage — remains unimplemented, and is a prerequisite for the approach to matter at the Fog-Edge scale that motivates this dissertation [22, 153, 171]

Appendix A

Appendix

A.1 Annotated repository structure of the ChaosAthena framework

Figure A.1: Annotated repository structure of the ChaosAthena framework.

```
ChaosAthena/
|-- pyproject.toml          # Package definition for 'ChaosAthena'
|-- src/
|   |-- core/              # [L1 - Shared Kernel] Builder / DTO profiles
|   |-- inference/         # [L1 - Shared Kernel] Strategy & Engine Factory
|   |-- pipeline_unifier/   # Hydra orchestration and state management
|   |-- pipeline/          # [L3 - Facades] Architectural execution entries
|       |-- performance_eval_stage.py # Performance Evaluation Stage
|       |-- red_ai_stage.py  # Red Team AI Evaluation Stage
|       |-- htsr_setol_stage.py # Deep Learning First-Principles Stage
|       |-- compression_stage.py # Compression Stage ( GGUF)
|       |-- datasets_creator_stage/ # Multi-Perspective Dataset Creator Stage
|       |-- veritas_analitics_stage/ # Veritas Analytics stage
|   |-- modules/           # [L2 - Domain Services] Underlying logic
|       |-- lm_harness/     # Module for performance evaluation
|       |-- garak/          # Module for vulnerability probing
|       |-- weight_watcher/ # Module for empirical spectral analysis
|       |-- compression/    # Post-training quantization routines ( GGUF)
|       |-- datasets_creator_stage/ # Module to create multi-perspective datasets
|       |-- veritas_analitics_stage/ # Module for the Analysis on the final datasets
|   |-- run_pipeline.py     # Unified CLI dispatcher (dynamic importing)
|-- configs/               # Layered Hydra YAML configuration hierarchy
|   |-- performance_eval/   # Configurations for performance testing runs
|   |-- red_ai/             # Probing constraints and vulnerability configs
|   |-- htsr_setol/         # Spectral matrix analytics parameters and models configurations
|   |-- compression/        # Target formats and bit modes
|-- notebooks/             # Configurable Jupyter Notebooks to CLI parity
|   |-- performance_eval.ipynb # Interactive capability performance stage
|   |-- red_ai.ipynb         # Interactive Adversarial target evaluation stage
|   |-- htsr_setol.ipynb     # Interactive Weight spectral analysis stage
|   |-- compression.ipynb    # Local quantization
|   |-- datasets_creator_stage.ipynb # create multi-perspective datasets
|   |-- veritas_analitics.ipynb # Analysis on the final datasets
'-- results/               # Storage substrate for decoupled execution artifacts
    |-- performance_eval/   # Downstream evaluation task benchmarks metrics
    |-- red_ai/             # Diagnostic vulnerability metrics
    |-- htsr_setol/         # Computed HTSR/SETOL weight metrics
    |-- compression/        # Quantized models
    |-- datasets_creator_stage/ # created multi-perspective datasets
    |-- veritas_analitics/  # All analysis performed on the final datasets
```

A.2 Example of Notebook Execution

A.2.1 Deep Learning First-Principles Stage

This Appendix subsection documents a complete example of a `configure→run→verify→inspect` loop for a ChaosAthena stage, using the Jupyter Notebook `hstr_setol` in Google Colab, with the Deep Learning First-Principles stage as the main representative example. Crucially, the main Hydra Orquestrator pipeline and all stages share the same notebook shape, the same type of configurable interface using `@params`, and the same orchestrator dispatch [5.1.4.2](#).

ChaosAthena — Deep Learning First-Principles Stage

Hierarchy: ChaosAthena Pipeline → Phase 2 (Evaluation & Analysis) → **htsr_setol** Stage → **weight_watcher** Module.

This notebook runs the HTSR/SETOL structural analysis through the **Hydra Orchestrator Pipeline** (`src/run_pipeline.py`), exactly the same code path as the CLI (`run(cfg)`) façade in (`src/pipelines/htsr_setol_pipeline.py`). Configurable with **@param form fields** — no code edits are needed to:

- choose the model scope — a single model, a custom sweep, or **all** configured models (every YAML in `configs/htsr_setol/model/`);
- choose where results are written — **Google Drive** or **local** (`output.base_dir`);
- optionally **add a new model YAML on the fly** into `configs/htsr_setol/model/`, which is then immediately picked up by `model=all`.

1 · Environment setup — clone the framework & install

Clones the ChaosAthena repository and installs it in editable mode. If `github_token` is left empty, it will default to the Colab Secret named `GITHUB_TOKEN`.

repo_url	branch
https://github.com/wolffCuanhamaRWS/ChaosAthena.git	main
github_token	
Insert text here	

[Show code](#)

4 · (Optional) Add a new model YAML on the fly

Writes `configs/htsr_setol/model/<new_model_name>.yaml`. Because `model=all` is resolved by the orchestrator from that directory, the new model is **immediately included** in the next run — no code changes.

To add more than one model YAM file, just **insert new model parameters and run the cell**

☐ add_new_model	new_model_name
	Qwen3-1.7B
full_hf_id	gguf_repo_hf_id
Qwen/Qwen3-1.7B	unsloth/Qwen3-1.7B-GGUF
variants	
Q4_K_M, Q3_K_M, Q2_K_L	

[Show code](#)

Skipped — enable 'add_new_model' to write a new model group YAML.

Working directory: /content/ChaosAthena

2 · Storage & credentials

Where should the stage write its artifacts? With `gdrive`, Drive is mounted and `results_path` is passed to Hydra as `output.base_dir` (the storage substrate). By default, it uses the `HF_TOKEN` environment variable. If not set, it falls back to the Colab Secret named in `hf_token_secret`.

storage	results_path
gdrive	/content/drive/MyDrive/Thesis_results/htsr_setol
hf_token_secret	
Insert text here	

[Show code](#)

3 · Run parameters — model scope & WeightWatcher options

<pre>1 #title 3 · Run parameters — model scope & WeightWatcher 2 #markdown **Model scope** (*Selecting the model scope* 3 #markdown 'all' runs every model that has a YAML in 4 run_scope = "custom" #param ["all", "custom"] 5 custom_models = " Qwen3-0.6B" #param (type:"string") 6 #markdown **WeightWatcher (module) options** — forward 7 fit_distribution = "PL" #param ["PL", "TPL", "E_TPL"] 8 save_plots = True #param (type:"boolean") 9 extra_overrides = "" #param (type:"string") 10 #markdown "(e.g. `htsr.params.min_evals=50, htsr.params` 11 12 # Papermill default parameters — this cell carries the 13 # above can also be injected headlessly: papermill hts 14 configs_to_run = ["all"] if run_scope == "all" else [n</pre>	Model scope (<i>Selecting the model scope: single, sweep, or all</i>): all runs every model that has a YAML in <code>configs/htsr_setol/model/</code> .
	run_scope
	custom
	custom_models
	Qwen3-0.6B
	WeightWatcher (module) options — forwarded as Hydra overrides:
	fit_distribution
	☐ save_plots

5 · Dispatch through the Hydra Pipeline Orchestrator

```
1 #title 5 · Dispatch through the Hydra Pipeline Orchestrator
2 # Same façade the CLI uses: src/run_pipeline.py <stage> [overrides...]
3 # 'model=all' is expanded by the orchestrator into one composed run per YAML in
4 # configs/htsr_setol/model/ (skip-if-done idempotency makes re-runs resume safely).
5 import os, subprocess, sys
6
7 overrides = [f"output.base_dir={results_path}",
8             f"htsr.params.fit={fit_distribution}",
9             f"htsr.savefig={str(save_plots).lower()}"]
10 overrides += [o.strip() for o in extra_overrides.split(",") if o.strip()]
11
12 env = os.environ.copy()
13 env["PYTHONPATH"] = os.getcwd() + (":" + env["PYTHONPATH"] if env.get("PYTHONPATH") else "")
14
15 for model in configs_to_run:
16     cmd = [sys.executable, "src/run_pipeline.py", stage, f"model={model}", *overrides]
17     print("Running:", " ".join(cmd))
18     try:
19         subprocess.run(cmd, check=True, env=env, capture_output=True, text=True)
20     except subprocess.CalledProcessError as e:
21         print("\n--- Stage Error ---")
22         print("STDOUT:")
23         print(e.stdout)
24         print("STDERR:")
25         print(e.stderr)
26         print("-----\n")
27     else:
28         print("Stage finished — configuration in, artifacts out.")
29
```

Stage finished — configuration in, artifacts out.

6 · Verify the artifacts written to the storage substrate

```

1 #title 6 · Verify the artifacts written to the storage substrate
2 import os
3
4 expected = ("esd_metrics.csv", "description.csv", "summary.json")
5 for model_dir in sorted(os.listdir(results_path)):
6     mpath = os.path.join(results_path, model_dir)
7     if not os.path.isdir(mpath):
8         continue
9     print(f"\n{model_dir}")
10    for variant in sorted(os.listdir(mpath)):
11        vpath = os.path.join(mpath, variant)
12        if not os.path.isdir(vpath):
13            continue
14        present = [f for f in os.listdir(vpath)]
15        ok = all([f.endswith(e) for f in present] for e in expected)
16        mark = "OK" if ok else "?"
17        print(f" {mark}{variant}: {sorted(present)[16]}{'...' if len(present) > 6 else ''}")
18

```

```

Qwen3-0.6B/
OK Q2_K_L: ['Qwen3-0.6B_Q2_K_L_description.csv', 'Qwen3-0.6B_Q2_K_L_esd_metrics.csv', 'Qwen3-0.6B_Q2_K_L_status.yaml',
OK Q3_K_M: ['Qwen3-0.6B_Q3_K_M_description.csv', 'Qwen3-0.6B_Q3_K_M_esd_metrics.csv', 'Qwen3-0.6B_Q3_K_M_status.yaml',
OK Q4_K_M: ['Qwen3-0.6B_Q4_K_M_description.csv', 'Qwen3-0.6B_Q4_K_M_esd_metrics.csv', 'Qwen3-0.6B_Q4_K_M_status.yaml',
OK uncompressed: ['Qwen3-0.6B_uncompressed_description.csv', 'Qwen3-0.6B_uncompressed_esd_metrics.csv', 'Qwen3-0.6B_un
model_cache/
?? hf.cache: ['.locks', 'CACHEDIR.TAG', 'models--Qwen--Qwen3-0.6B', 'models--unsloth--Qwen3-0.6B-GGUF']
?? models: ['Qwen3-0.6B']

```

```

5 import os, glob
6 import pandas as pd
7 import matplotlib.pyplot as plt
8
9 pattern = os.path.join(results_path, inspect_model,
10 hits = glob.glob(pattern)
11 if not hits:
12     print(f"No esd_metrics.csv under {pattern} - run
13 else:
14     df = pd.read_csv(hits[0])
15     print(df[["layer_id", "alpha"]].describe().loc[
16     ax = df.plot(x="layer_id", y="alpha", kind="line
17                 title=f"{inspect_model} {inspect_va
18     ax.xaxispan(2, 6, alpha=0.12, label="well-trained
19     ax.legend(); plt.tight_layout(); plt.show()
20

```

[Hide code](#)

7 · Quick look — per-layer alpha of one analyzed variant

```

1 #title 7 · Quick look — per-layer alpha of one anal
2 inspect_model = "Qwen3-0.6B" #@param {type:"string"}
3 inspect_variant = "Q4_K_M" #@param {type:"string"}
4

```

inspect_model

Qwen3-0.6B

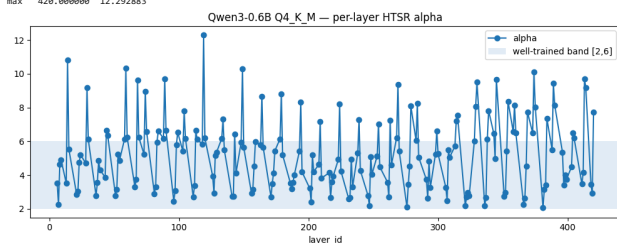
inspect_variant

Q4_K_M

```

layer_id    alpha
mean    212.785714    5.224189
min      6.000000    2.079369
max    420.000000   12.292883

```



A.3 Reference Models included in the garak Z-Score Calibration Bag (Winter 2026)

Table A.1: Reference Models included in the garak Z-Score Calibration Bag (Winter 2026)

Model Name	Provider	Parameter Class	Description
AI21-Jamba2-3B	AI21 Labs	1B-10B	3B parameter reference model.
AI21-Jamba2-Mini	AI21 Labs	100B+	200B parameter reference model.
DASD-4B-Thinking	Alibaba	1B-10B	4B parameter reasoning model.
c4ai-command-a-03-2025	Cohere	100B+	111B parameter reference model.
gemma-3-12b-it	Google	11B-99B	12B parameter instruct reference.
granite-4.0-h-micro	IBM	1B-10B	3B parameter reference model.
granite-4.0-h-small	IBM	11B-99B	32B parameter reference model.
LFM2.5-1.2B-Instruct	Liquid AI	1B-10B	1.2B parameter instruct reference.
llama-4-scout-17b-16e-instruct	Meta	11B-99B	17B parameter instruct reference.
ama-4-maverick-17b-128e-instruct	Meta	11B-99B	17B parameter instruct reference.
phi-4	Microsoft	11B-99B	14B parameter reference model.
Mistral-Small-3.2-24B-Instruct-2506	Mistral AI	11B-99B	24B parameter instruct reference.
Magistral-Small-2506	Mistral AI	11B-99B	24B parameter reference model.
llama-3_1-nemotron-ultra-253b-v1	NVIDIA	100B+	235B parameter reference model.
nemotron-nano-9b-v2	NVIDIA	1B-10B	9B parameter reference model.
Nemotron-3-Nano-30B-A3B-Base-BF16	NVIDIA	11B-99B	30B parameter base reference.

Continued on next page...

Table A.1 – Continued from previous page

Model Name	Provider	Parameter Class	Description
gpt-5.2	OpenAI	Closed API	Proprietary API baseline.
gpt-5-nano	OpenAI	Closed API	Proprietary API baseline.
gpt-oss-120b	OpenAI	100B+	120B parameter open-weights reference.
qwen2.5-72b-instruct	Qwen	11B–99B	72B parameter instruct reference.
qwen3-8b	Qwen	1B–10B	8B parameter reference model.
apriel-1.5-15b-thinker	ServiceNow	11B–99B	15B parameter reasoning model.
Falcon-H1R-7B	TII UAE	1B–10B	7B parameter reference model.

Note: API = Application Programming Interface; LFM = Liquid Foundation Model; QA = Question Answering; UAE = United Arab Emirates.

Appendix B

Appendix

B.1 Experimental Setup

B.1.1 Execution Parameters (Performance, Red Team AI Stages)

The following tables have the full configuration parameters for the Capability Performance Evaluation and Red Team AI Stages. Each value below is the exact value logged by Hydra for all 28 model states.

Table B.1: Execution parameters for the Capability Performance Evaluation Stage (lm-evaluation-harness, vLLM serving, and sampling). The evaluated tasks and their family groupings are listed separately in Table 6.7.

Name	Value / Family	Description
lm-evaluation-harness Framework Parameters		
num_fewshot	null	Leaves few-shot settings unspecified, allowing the benchmark to apply its default configuration.
batch_size	auto	Dynamically determines and applies the largest continuous batch size that fits within the target VRAM.
apply_chat_template	True	Formats raw prompts using the specific tokenizer-defined chat template of the evaluated model variant.
log_samples	True	Archives the raw input prompts and model generations for subsequent deep-dive analytics.
write_out	True	Ensures all execution logs and sample results are persistently written to the storage substrate.

Continued on next page...

Table B.1 – Continued from previous page

Name	Value / Family	Description
confirm_run_unsafe_code	True	Authorizes the execution of untrusted generated code, necessary for functional correctness evaluation on humaneval_plus.
trust_remote_code	True	Permits the loading of custom model architectures and execution scripts from the Hugging Face Hub.
vLLM Inference Engine Parameters		
engine	vllm	Utilizes the vLLM inference backend. All 28 model states, full-precision and compressed alike, are executed through this single FP16 serving path (compressed variants via the dq map of 4.5), which isolates the effect of weight quantization from kernel- and runtime-level differences.
gpu_memory_utilization	0.90	Reserves 90% of the available GPU VRAM exclusively for model weights and the KV cache.
max_model_len	4048	Constrains the maximum context window to 4048 tokens to manage memory efficiently.
tensor_parallel_size	1	Disables tensor parallelism, forcing execution of the model on a single GPU.
enforce_eager	False	Allows CUDA graph capture for optimized operation by disabling eager execution.
enable_prefix_caching	True	Caches shared prompt prefixes to accelerate throughput on overlapping evaluation contexts.
Sampling and Decoding Parameters		
temperature	0.0	Enforces strict greedy decoding to ensure deterministic, highly reproducible baseline analysis.

Continued on next page...

Table B.1 – Continued from previous page

Name	Value / Family	Description
top_p	1.0	Disables nucleus sampling constraints, maintaining standard deterministic generation.
seed	42	Fixes the global random seed to 42 to guarantee exact reproducibility across concurrent runs.
disable_thinking_mode	True	Forces standard inference by strictly disabling extended chain-of-thought and reasoning model steps.

B.1.2 Red Team AI Stage parameters

Table B.2: Execution parameters for the Red Team AI Stage: vLLM server settings for the ephemeral endpoint and garak execution options.

Name	Value Used	Description
vLLM Server Settings		
HOST	127.0.0.1	Local bind address for the ephemeral vLLM endpoint.
PORT	8000	Network port allocated for the API interface.
VLLM_GPU_MEMORY_UTILIZATION	0.95	Allocates 95% of available VRAM to maximize KV cache size.
MAX_MODEL_LEN	4096	Limits the maximum context window to 4096 tokens.
-tensor-parallel-size	1	Disables tensor parallelism, executing models on a single GPU.
-max-num-seqs	500	The maximum number of concurrent sequences processed.
-max-num-batched-tokens	4096	Bounds the maximum number of batched tokens per iteration.
-enable-chunked-prefill	True	Chunks large prompts to maintain consistent decode latencies.

Continued on next page...

Table B.2 – Continued from previous page

Name	Value Used	Description
garak Execution Options		
model_type	openai	Interfaces with the vLLM server via standard OpenAI protocols.
generations	2	Number of independent stochastic completions generated per prompt.
parallel_attempts	500	Concurrency limit for parallel asynchronous prompt dispatch.
taxonomy	owasp	Defines the reporting grouping, mapping to OWASP Top 10 for LLM Applications.
temperature	0.0	Enforces greedy decoding for deterministic baseline analysis.
top_p	1.0	Disables nucleus sampling constraint.
max_tokens	200	Caps the maximum generation length to prevent run-away outputs. Used a value above the 150 garak recommended default to ensure capturing malicious behavior [31].
GARAK_EXTRA_ARGS	-seed 42, -skip_unknown, -v	Enforces fixed seeds, skips unrecognized probes when a probe needs additional parameters not passed in the experiments, and enables verbose logging.

Note: API = Application Programming Interface; KV = Key-Value; OWASP = Open Web Application Security Project; VRAM = Video Random Access Memory.

B.1.3 Executed Probes Mapped to OWASP-2025 Categories

Table B.3: Executed Probes and Sub-Probes Mapped to OWASP-2025 Categories

OWASP 2025	Probe Category	Sub-Probe Name
LLM01	dan	Ablation_Dan_11_0

Continued on next page...

Table B.3 – Continued from previous page

OWASP 2025	Probe Category	Sub-Probe Name
LLM01	dan	AutoDANCached
LLM01	dan	DanInTheWild
LLM01, 02, 05	ansiescape	AnsiEscaped / AnsiRaw
LLM01	continuation	ContinueSlursReclaimedSlurs
LLM01	dra	DRA
LLM01	phrasing	FutureTense / PastTense
LLM01	promptinject	HijackHateHumans
LLM01	promptinject	HijackKillHumans
LLM01	promptinject	HijackLongPrompt
LLM01	encoding	InjectAscii85 / Atbash
LLM01	encoding	InjectBase16 / 32 / 64
LLM01	encoding	InjectBase2048 / Braille
LLM01	encoding	InjectEmoji / Hex
LLM01	encoding	InjectMorse / Nato
LLM01	encoding	InjectROT13
LLM01	encoding	InjectUnicodeTagChars
LLM01	encoding	InjectUU / Zalgo
LLM01	latentinjection	LatentInjectionFactSnippetEiffel
LLM01	latentinjection	LatentInjectionFactSnippetLegal
LLM01	latentinjection	LatentInjectionReport / Resume
LLM01	latentinjection	LatentInjectionTranslationEnFr
LLM01	latentinjection	LatentInjectionTranslationEnZh
LLM01	latentinjection	LatentJailbreak
LLM01	latentinjection	LatentWhois / WhoisSnippet
LLM01	sata	MLM
LLM01	syspromptextraction	SystemPromptExtraction
LLM01	goodside	Tag
LLM01, 04:2023	divergence	RepeatedToken

B.2 Results

B.2.1 Can specific attacks be associated with structural change?

B.2.2 Probe-level robustness check on the vulnerability–structure coupling

The coupling analysis of Section 6.3.7 takes the probe $\times$ detector pair as its unit, which means several detectors of one attack contribute as technical replicates and are positively dependent — a dependence the Benjamini–Hochberg correction tolerates but which inflates the apparent number of independent tests. Table B.4 and Figure B.2 repeat the analysis with detectors averaged within each probe, so that every row is a genuinely distinct attack and the correction runs over a near-independent grid (three of the 39 probes are dropped for having fewer than eight usable variants or an invariant pass rate, giving $36 \times 3 = 108$ tests).

The encoding and obfuscation signal survives the stricter unit. The two cells that pass joint correction — `enc.InjectNato` and `ansi.AnsiEscaped`, both coupling to $\Delta_{\text{rand_distance}}$ at $\rho \approx -0.7$, $q \approx 0.03$ — are exactly the two distinct attack families that the probe $\times$ detector analysis surfaced, where they appeared as three detector-level cells. The conclusion of Section 6.3.7 is therefore not an artefact of detector multiplicity.

Table B.4: Robustness check at the probe level. The coupling of Table 6.29 is recomputed with detectors averaged within each probe, so the unit is the genuinely distinct attack (39 probes) rather than the probe $\times$ detector pair: this removes the within-probe positive dependence between detectors of the same attack and yields a near-independent 39×3 test family for the Benjamini–Hochberg correction. The strongest couplings, their best-matching structural metric, and joint-FDR q are reported as before.

OWASP	Probe · Detector	$\rho(\Delta\alpha)$	$\rho(\Delta\alpha_w)$	$\rho(\Delta_{\text{rd}})$	best	p	q (FDR)	n
llm01	ansi.AnsiEscaped	+0.29	+0.10	-0.70	Δ_{rd}	0.000	0.028	21
llm01	enc.InjectNato	+0.33	+0.10	-0.69	Δ_{rd}	0.001	0.028	21
llm01	enc.InjectBase64	+0.06	-0.25	-0.59	Δ_{rd}	0.005	0.185	21
llm01	ansi.AnsiRaw	+0.23	-0.04	-0.57	Δ_{rd}	0.007	0.186	21
llm01	LI.LatentInjectionReport	+0.31	-0.01	-0.56	Δ_{rd}	0.009	0.186	21
llm01	LI.LatentInjectionTranslationEnFr	-0.33	-0.53	-0.04	$\Delta\alpha_w$	0.014	0.238	21
llm01	sata.MLM	-0.51	-0.39	+0.10	$\Delta\alpha$	0.017	0.238	21
llm01	enc.InjectZalgo	+0.22	-0.03	-0.51	Δ_{rd}	0.018	0.238	21
llm01	LI.LatentWhois	-0.26	-0.03	+0.49	Δ_{rd}	0.023	0.238	21
llm01	enc.InjectAtbash	-0.25	-0.49	-0.31	$\Delta\alpha_w$	0.023	0.238	21

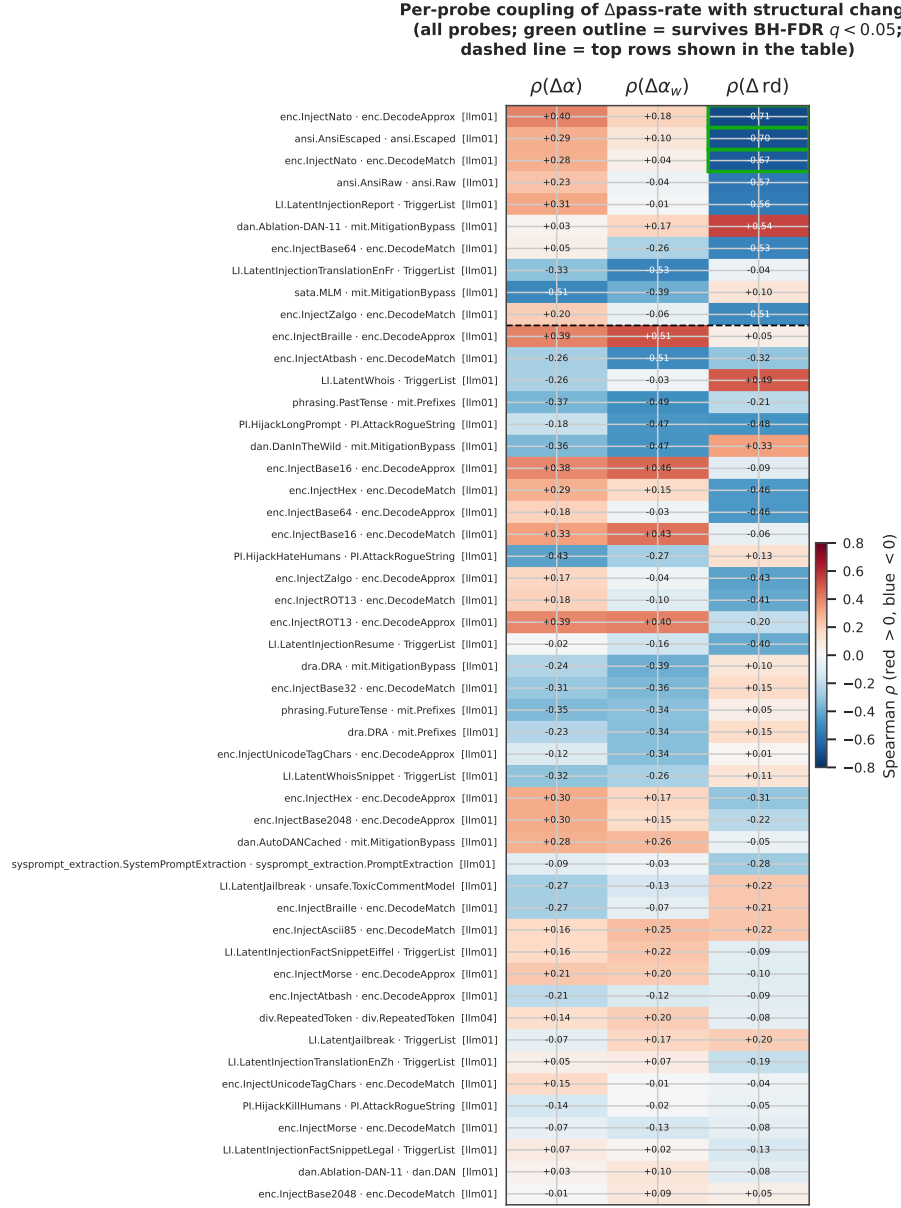


Figure B.1: Complete per-probe Spearman association. For *every* probe with a computable correlation (All combinations, total 50 probes with there corresponding detectors, not just the ten strongest tabulated alongside), the rank correlation of its compression-induced pass-rate change with each of the three global structural changes— $\Delta\alpha$, $\Delta\alpha_{\text{weighted}}$ ($\Delta\alpha_w$), and $\Delta\text{rand_distance}$ (Δrd)—is shown across the 21 family $\times$ level cells, annotated with ρ (red: $\rho > 0$, blue: $\rho < 0$). Probes are ordered by their strongest absolute coupling; the **dashed line separates the ten probes** also listed in Table 6.29 (above) from the remaining forty (below), and the **three cells outlined in green are the couplings that survive joint Benjamini–Hochberg correction** across all $50 \times 3 = 150$ probe–metric tests ($q < 0.05$), including all the used detectors in every probe. The figure makes the table’s claim visible across the entire probe set: strong negative coupling is concentrated in the $\Delta\text{rand_distance}$ column and in the encoding/obfuscation injection probes, while correlations with $\Delta\alpha$ are weak across the tests or don’t survive the BH-FDR tests.

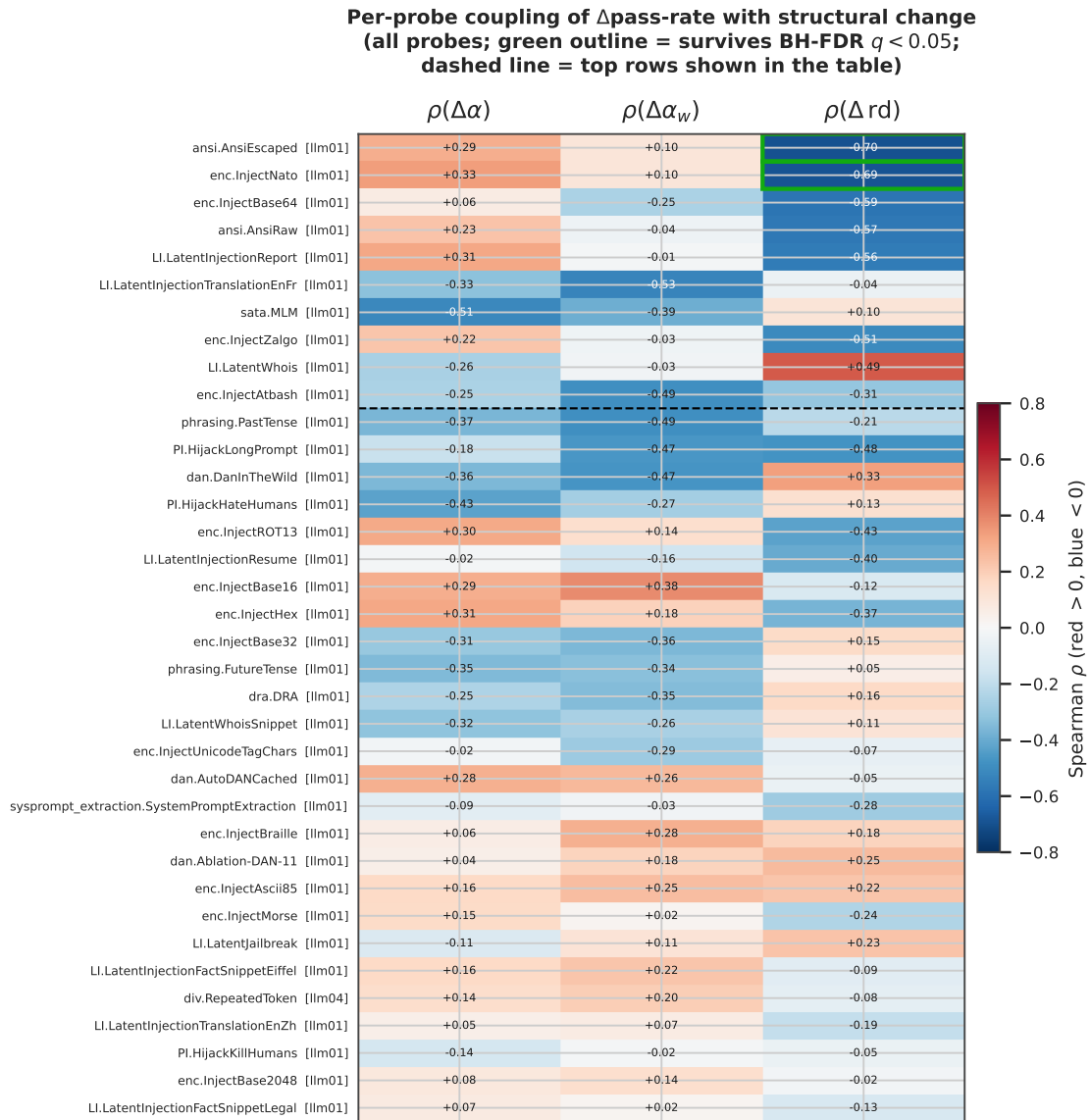


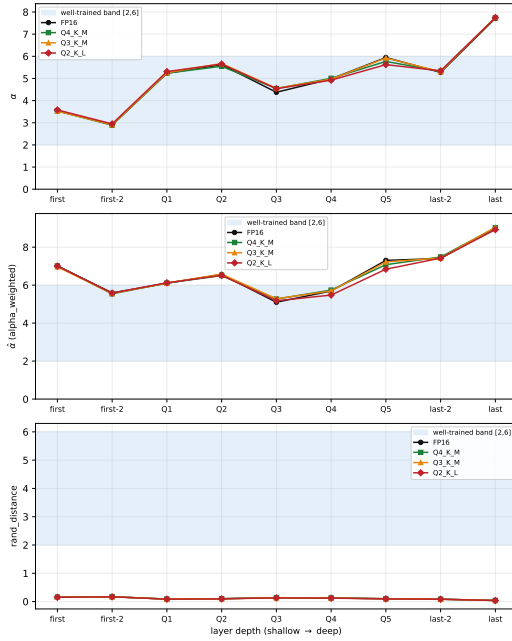
Figure B.2: Complete per-probe Spearman coupling with detectors averaged within each probe, so that each row is one distinct attack. Cells surviving joint Benjamini–Hochberg correction are outlined; the surviving signal is confined to the Δ rand_distance column and to the encoding and obfuscation injection probes, matching the probe $\times$ detector analysis of Section 6.3.7.

B.3 Layers depth and mean changes

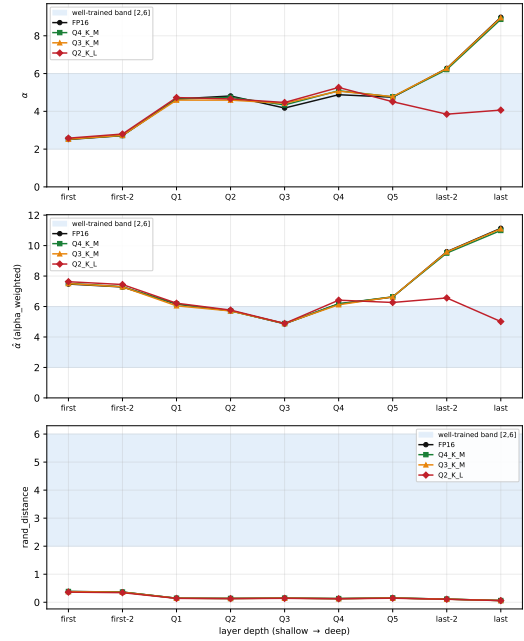
This section provides additional visualizations of the three data-free structural descriptors (α , $\hat{\alpha}$ /alpha_weighted, rand_distance) resolved over nine layer-depth positions (*first*, *first-2*, quintiles Q_1 – Q_5 , *last-2*, *last*), for every model family and its GGUF variants (FP16, Q4_K_M, Q3_K_M, Q2_K_L). They are exclusively auxiliary views: no new statistics are introduced, and all quantities are those analyzed in Chapter 6, shown here at full-layer-depth resolution (bins) per family of models to make the localization findings directly observable.

Layers depth metrics, in levels by family. One page per family (seven pages, ordered by parameter count), each with three plots (one per metric). Each plot presents the four-bit precision variants as colored point-line series across depth; for α and $\hat{\alpha}$ the well-trained band [2,6] is shaded blue [29, 30]. These plots show the compression invariance result in Chapter 6 per layer depth: the four curves nearly coincide at every position, the interior remains inside the band at all precisions, and the small deviations of Q2_K_L concentrate toward the end of the output. The well trained band, identified with color on the plots, is only informative for α and $\hat{\alpha}$, not *rand_distance*.

Qwen3-0.6B — full-precision vs compressed variants
depth-resolved α , $\hat{\alpha}$, rand_distance; blue = well-trained band [2,6]



Llama-3.2-1B-Instruct — full-precision vs compressed variants
depth-resolved α , $\hat{\alpha}$, rand_distance; blue = well-trained band [2,6]



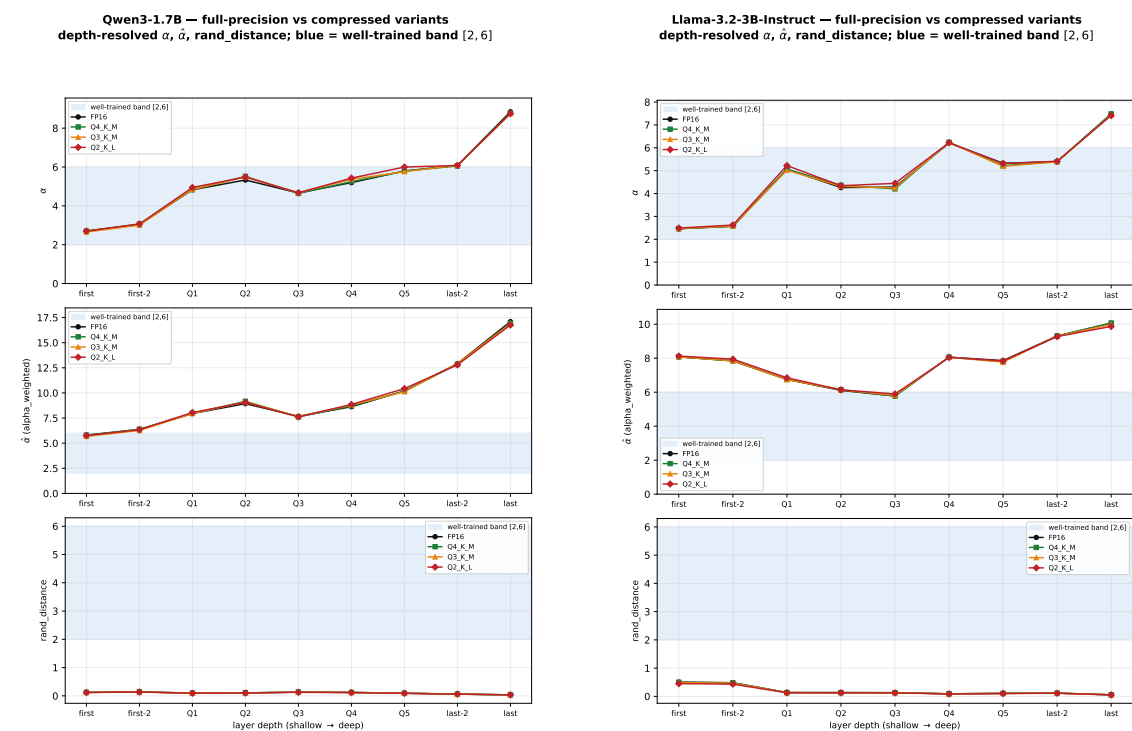
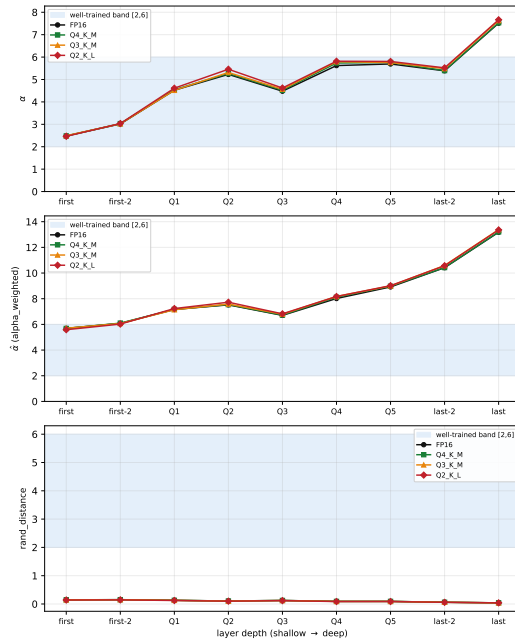
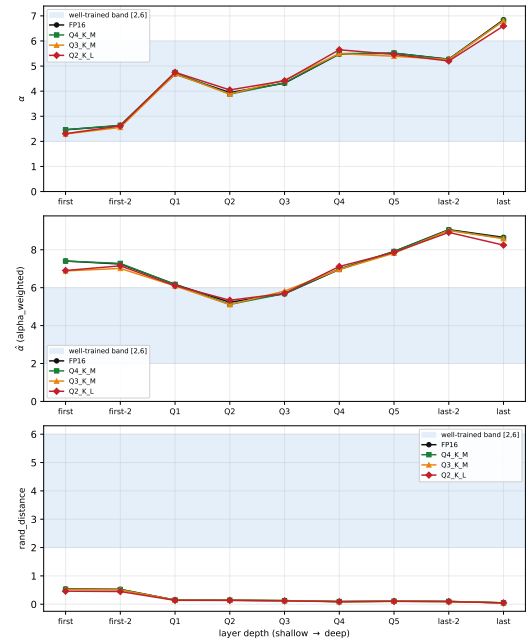


Figure B.3: Layer depth structural metric levels (α , $\hat{\alpha}$, rand_distance) by model family across quantization regimes (Pages 1–4 of 7).

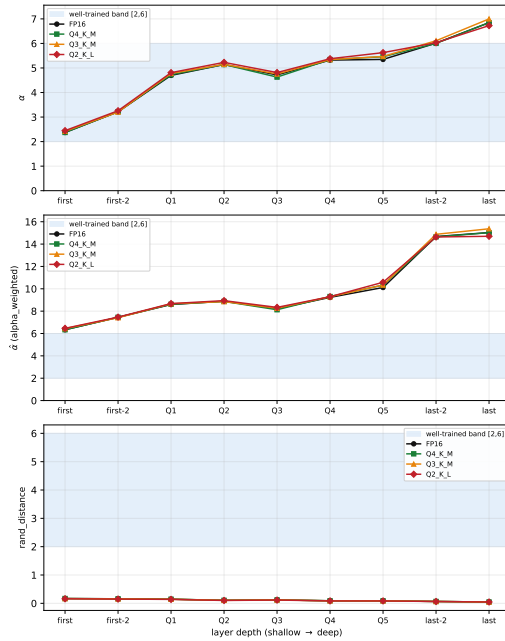
Qwen3-4B-Instruct-2507 — full-precision vs compressed variants
depth-resolved α , $\hat{\alpha}$, rand_distance; blue = well-trained band [2, 6]



Meta-Llama-3.1-8B-Instruct — full-precision vs compressed variants
depth-resolved α , $\hat{\alpha}$, rand_distance; blue = well-trained band [2, 6]



Qwen3-8B — full-precision vs compressed variants
depth-resolved α , $\hat{\alpha}$, rand_distance; blue = well-trained band [2, 6]

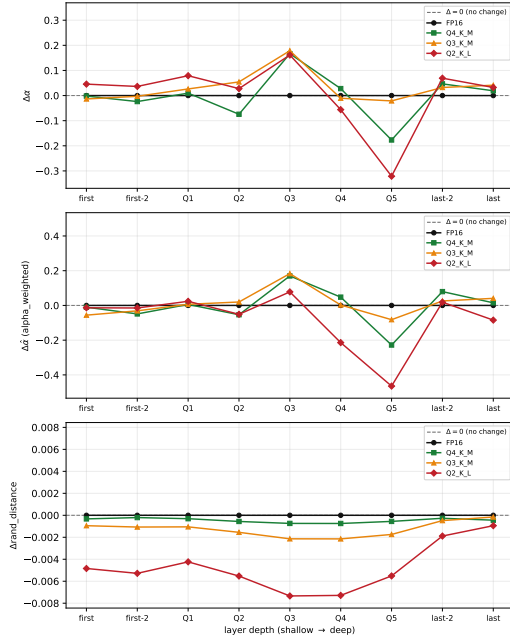


[End of Absolute Metric Family plots]

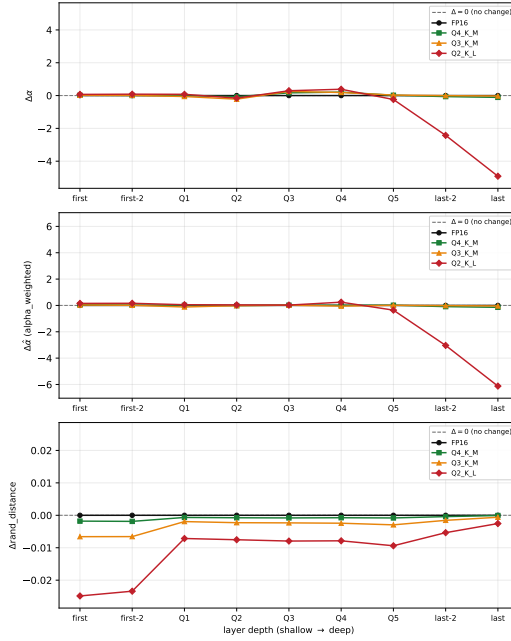
Figure B.4: Layer depth structural metric levels (bins) by model family (Pages 5–7 of 7).

Compression-induced changes by family of variants. The same layout for the deltas ($\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; compressed variants minus the family’s own FP16 bit precision baseline at the same depth). The FP16 variant is identically zero and, with the dashed $\Delta=0$ line, serves as the no-change reference; axes are symmetric about zero so amplitudes are comparable within each plot. These plots make the dose-dependence visible per family: Q4_K_M hugs zero at every depth, Q3_K_M wobbles mildly, and Q2_K_L shows the largest excursions, predominantly at the last positions.

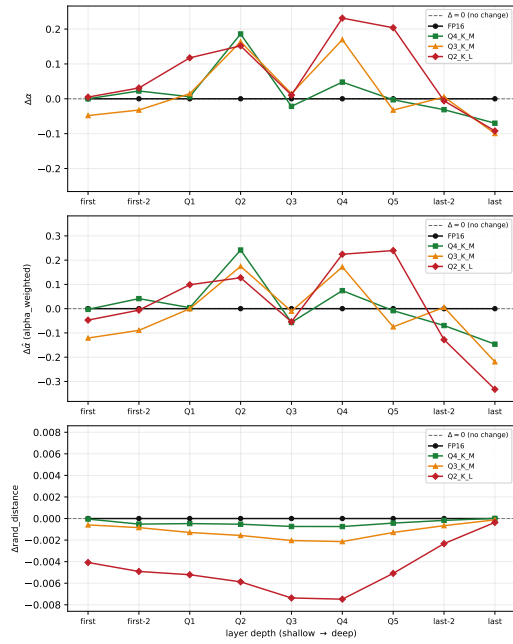
Qwen3-0.6B — compression-induced change vs FP16
depth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta=0$; FP16 $\equiv 0$ by definition



Llama-3.2-1B-Instruct — compression-induced change vs FP16
depth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta=0$; FP16 $\equiv 0$ by definition



Qwen3-1.7B — compression-induced change vs FP16
 depth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta = 0$; FP16 $\equiv 0$ by definition



Llama-3.2-3B-Instruct — compression-induced change vs FP16
 depth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta = 0$; FP16 $\equiv 0$ by definition

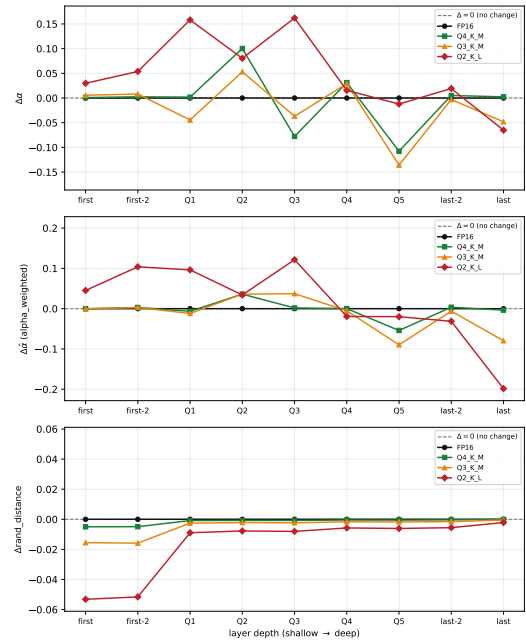
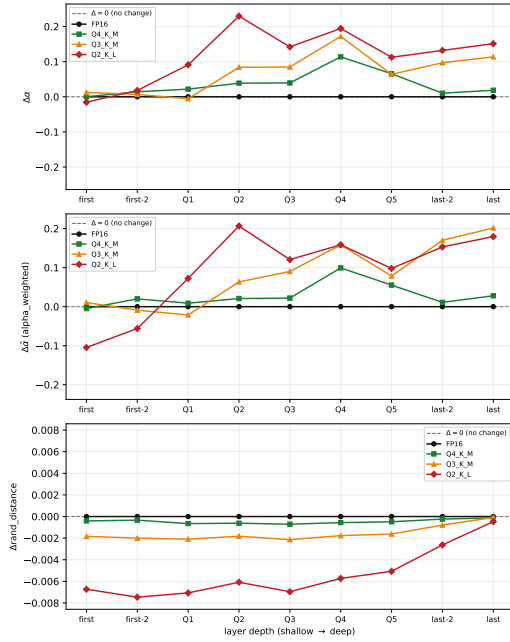
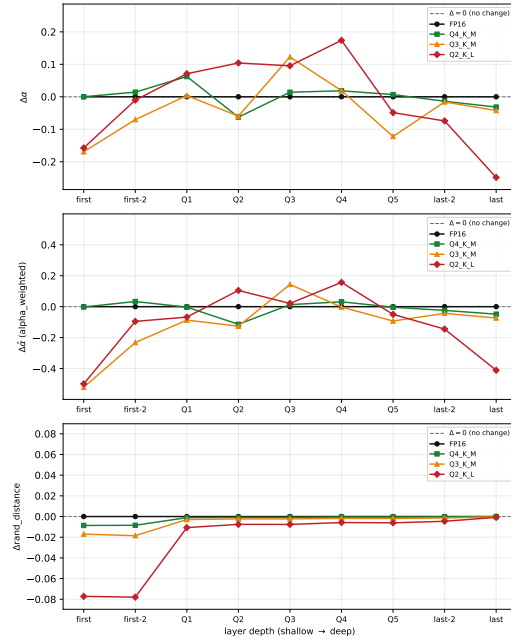


Figure B.5: Compression-induced metric deltas ($\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$) relative to FP16 baselines by model family (Pages 1–4 of 7).

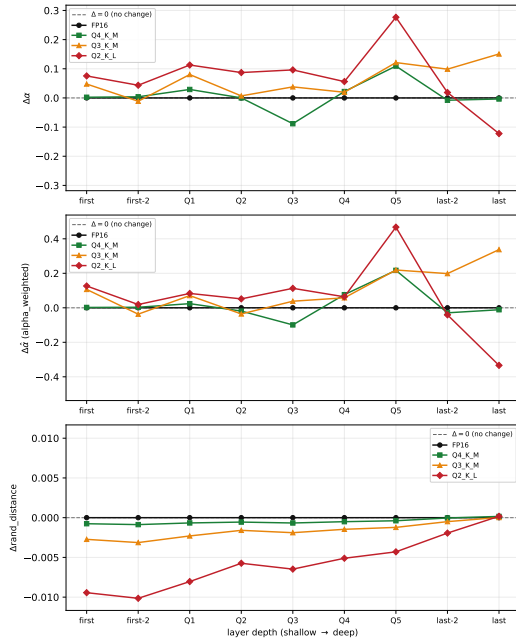
Qwen3-4B-Instruct-2507 — compression-induced change vs FP16
 lepth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta = 0$; FP16 $\equiv 0$ by definition



Meta-Llama-3.1-8B-Instruct — compression-induced change vs FP16
 lepth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta = 0$; FP16 $\equiv 0$ by definition



Qwen3-8B — compression-induced change vs FP16
 lepth-resolved $\Delta\alpha$, $\Delta\hat{\alpha}$, $\Delta\text{rand_distance}$; dashed line = $\Delta = 0$; FP16 $\equiv 0$ by definition



[End of Delta Metric Family plots]

Granular Structural Diagnostics. To illustrate the tracking of a metric at the extreme, layer by layer, Figure B.6 presents the raw, unbinned trajectory of the Heavy-Tailed Self-Regularization (HTSR) spectral exponent α across all 420 individual layers of the Qwen3-0.6B Q4_K_M quantized variant. An analysis of the empirical distribution reveals that the network maintains structural stability, yielding a global mean exponent of $\mu_\alpha \approx 5.22$, which sits safely within the theoretically optimal, well-trained spectral band of $[2, 6]$ (with $\approx +2$, preferable), established in the HTSR and SETOL literature [29, 30].

Despite this stable tendency, the high-resolution perspective exposes massive localized fluctuations, spanning an absolute minimum of $\min_\alpha \approx 2.08$ (approaching the boundary of the optimal heavy-tailed layer) to a maximum spike of $\max_\alpha \approx 12.29$. These extreme spikes represent localized over-parameterization or under-trained layer states and demonstrate the value of tracking discrete layer IDs rather than relying solely on global network averages [29, 30].

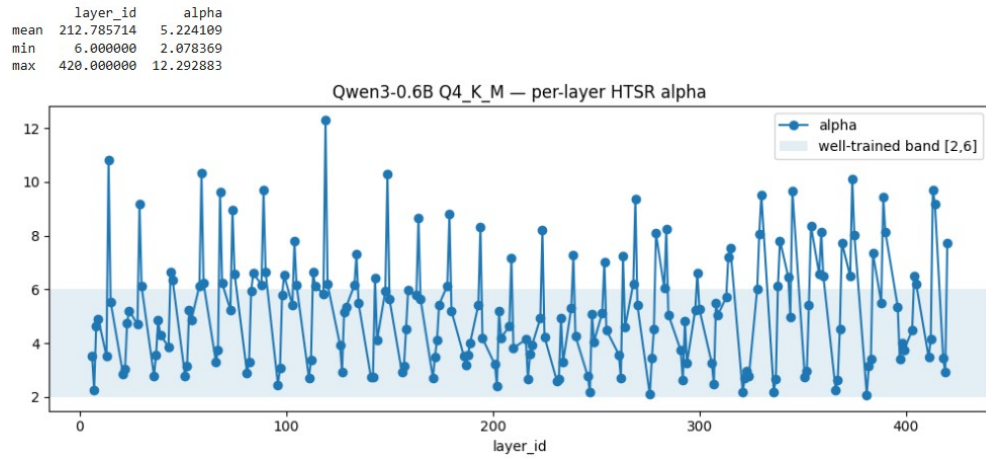


Figure B.6: High-resolution architectural diagnostic profile extracted by the Veritas Analytics stage. The plot tracks the HTSR α exponent across all 420 individual layer IDs for the Qwen3-0.6B Q4_K_M variant, mapped against the shaded $[2, 6]$ well-trained HTSR and SETOL band ($\mu_\alpha = 5.224$, $\min_\alpha = 2.078$, $\max_\alpha = 12.293$).

B.4 Quantized model on Hugging Face

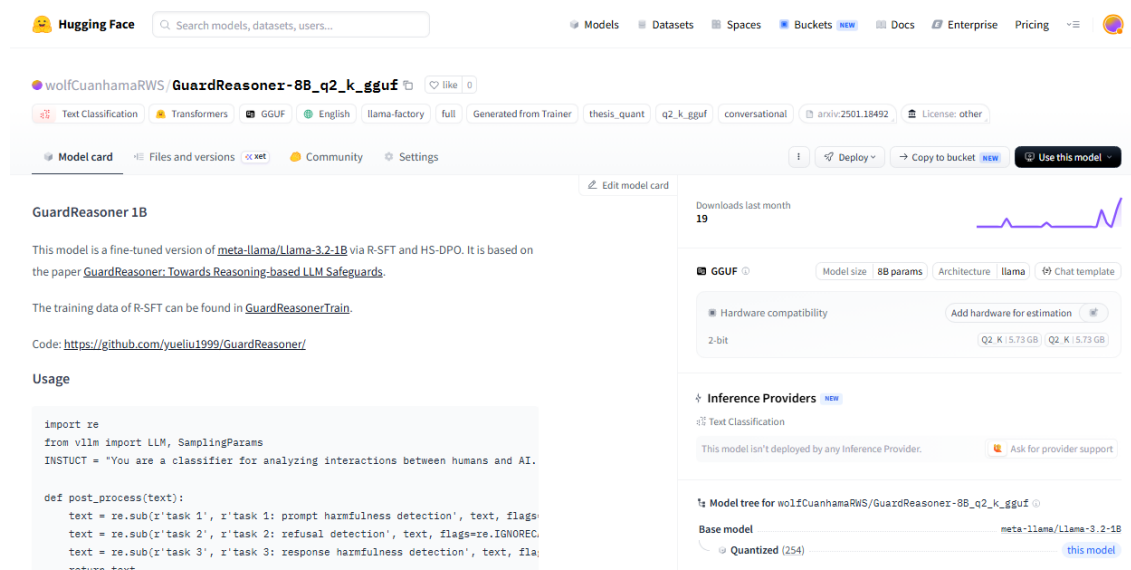


Figure B.7: Hugging Face Hub model repository interface for the deployed GuardReasoner GGUF artifact. The snapshot details the model card overview, metadata categorization tags (e.g., gguf, q2_k_gguf), and file tree tracking for the compressed 2-bit variant within the remote coordinate path `wolfCuanhamaRWS/GuardReasoner-8B_q2_k_gguf`. This serves as concrete verification of the framework’s remote asset registration, metadata synchronization, and target deployment compliance under the retrieve-or-compute mechanics of Functional Requirement 4 (FR4).

References

- [1] Juergen Schmidhuber. Annotated History of Modern AI and Deep Learning. <https://arxiv.org/abs/2212.11279v3>, December 2022.
- [2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. arXiv:1706.03762.
- [3] Mengwei Xu, Wangsong Yin, Dongqi Cai, Rongjie Yi, Daliang Xu, Qipeng Wang, Bingyang Wu, Yihao Zhao, Chen Yang, Shihe Wang, Qiyang Zhang, Zhenyan Lu, Li Zhang, Shangguang Wang, Yuanchun Li, Yunxin Liu, Xin Jin, and Xuanzhe Liu. A survey of resource-efficient LLM and multimodal foundation models. *arXiv preprint arXiv:2401.08092*, 2024.
- [4] The 2026 AI Index Report | Stanford HAI. <https://hai.stanford.edu/ai-index/2026-ai-index-report>.
- [5] Eoghan Cunningham and Derek Greene. Knowledge transfer, knowledge gaps, and knowledge silos in citation networks. *PLOS One*, 20(8):e0329302, August 2025.
- [6] Michael Latzer. Digitalization, AI and the rise of techno-religion: Transhumanist promises and the challenge to Enlightenment. *Telecommunications Policy*, 50(2):103115, March 2026.
- [7] Murray Shanahan. Talking about large language models. *Communications of the ACM*, 67(2):68–79, 2024.
- [8] Rachel Thomas and David Uminsky. The problem with metrics is a fundamental problem for AI. *arXiv preprint arXiv:2002.08512*, 2020.
- [9] Kenneth O. Stanley and Joel Lehman. *Why Greatness Cannot Be Planned: The Myth of the Objective*. Springer, Cham, Switzerland, 2015.
- [10] Joel Lehman and Kenneth O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011.
- [11] Shivalika Singh, Yiyang Nan, Alex Wang, Daniel D’Souza, Sayash Kapoor, Ahmet Üstün, Sanmi Koyejo, Yuntian Deng, Shayne Longpre, Noah A. Smith, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. The leaderboard illusion. *arXiv preprint arXiv:2504.20879*, 2025.
- [12] Víctor Mayoral-Vilches. Cybersecurity AI: The dangerous gap between automation and autonomy. *arXiv preprint arXiv:2506.23592*, 2025.
- [13] Huijun Zhou, Xiaohan Zhang, Haozhe Zhang, Haoyang Zhang, Mi Zhang, and Min Yang. A first measurement study on authentication security in real-world remote MCP servers, 2026.
- [14] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.
- [15] Stan Franklin and Art Graesser. Is it an agent, or just a program? A taxonomy for autonomous agents. In *Intelligent Agents III: Agent Theories, Architectures, and Languages*, volume 1193 of *Lecture Notes in Computer Science*, pages 21–35. Springer, 1997.
- [16] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, and Yiwen Ding. The rise and potential of large language model based agents: a survey. *Science China Information Sciences*, 68(2):121101, 2025.

- [17] MIT Media Lab – Project NANDA. A perspective on decentralizing AI, 2025. Project report.
- [18] Ramesh Raskar. NANDA: Resolution, discovery, and governance for the agentic web. In *Companion Proceedings of the ACM Web Conference 2026 (WWW Companion '26)*, page 504. Association for Computing Machinery, 2026. doi:10.1145/3774905.3796217.
- [19] Ramesh Raskar, Pradyumna Chari, John Zinky, Mahesh Lambe, Jared James Grogan, Sichao Wang, Rajesh Ranjan, Rekha Singhal, Shailja Gupta, Robert Lincourt, Raghu Bala, Aditi Joshi, Abhishek Singh, Ayush Chopra, Dimitris Stripelis, Bhuwan B, Sumit Kumar, and Maria Gorskikh. Beyond DNS: Unlocking the internet of AI agents via the NANDA index and verified AgentFacts, 2025.
- [20] Ramesh Raskar, Pradyumna Chari, Jared James Grogan, Mahesh Lambe, Robert Lincourt, Raghu Bala, Aditi Joshi, Abhishek Singh, Ayush Chopra, Rajesh Ranjan, Shailja Gupta, Dimitris Stripelis, Maria Gorskikh, and Sichao Wang. Upgrade or switch: Do we need a next-gen trusted architecture for the internet of AI agents? *arXiv preprint arXiv:2506.12003*, 2025.
- [21] Divyanshu Kumar, Anurakt Kumar, Sahil Agarwal, and Prashanth Harshangi. Increased LLM vulnerabilities from fine-tuning and quantization. *arXiv preprint arXiv:2404.04392*, 2024.
- [22] Yue Zheng, Yuhao Chen, Bin Qian, Xiufang Shi, Yuanchao Shu, and Jiming Chen. A review on edge large language models: Design, execution, and applications. *ACM Computing Surveys*, 57(8):209:1–209:35, 2025. doi:10.1145/3719664.
- [23] Jian Zhao, S. Wang, Y. Zhao, X. Hou, K. Wang, P. Gao, Y. Zhang, C. Wei, and H. Wang. Models are codes: Towards measuring malicious code poisoning attacks on pre-trained model hubs. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 2087–2098, 2024. arXiv:2409.09368; doi:10.1145/3691620.3695271.
- [24] Kazuki Egashira, Robin Staab, Mark Vero, Jingxuan He, and Martin Vechev. Mind the Gap: A Practical Attack on GGUF Quantization, June 2025. arXiv:2505.23786 [cs.CR].
- [25] Kazuki Egashira, Mark Vero, Robin Staab, Jingxuan He, and Martin Vechev. Exploiting LLM quantization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2405.18137.
- [26] OWASP Foundation. OWASP top 10 for large language model applications. <https://owasp.org/www-project-top-10-for-large-language-model-applications/>, 2025.
- [27] Ramesh Raskar. Nanda: Resolution, discovery, and governance for the agentic web. In *Companion Proceedings of the ACM Web Conference 2026, WWW Companion '26*, page 504, New York, NY, USA, 2026. Association for Computing Machinery.
- [28] Johannes Schneider. Explainable Generative AI (GenXAI): a survey, conceptualization, and research agenda. *Artificial Intelligence Review*, 57(11):289, September 2024.
- [29] Charles H. Martin and Michael W. Mahoney. Implicit Self-Regularization in Deep Neural Networks: Evidence from Random Matrix Theory and Implications for Learning. *Journal of Machine Learning Research*, 22(165):1–73, 2021.
- [30] Charles H Martin and Christopher Hinrichs. SETOL: A Semi-Empirical Theory of (Deep) Learning, 2025.
- [31] Leon Derczynski. garak: A framework for security probing large language models. *arXiv preprint arXiv:2406.11036*, 2024.
- [32] Brian G. Woolley and Kenneth O. Stanley. On the deleterious effects of a priori objectives on evolution and representation. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, pages 957–964, 2011.
- [33] Bertrand Meyer. *Object-Oriented Software Construction*. Prentice Hall, 1988.
- [34] Ralph E. Johnson and Brian Foote. Designing reusable classes. *Journal of Object-Oriented Programming*, 1(2):22–35, 1988.

- [35] Daking Rai, Yilun Zhou, Shi Feng, Abulhair Saparov, and Ziyu Yao. A practical review of mechanistic interpretability for transformer-based language models. *arXiv preprint arXiv:2407.02646*, 2024.
- [36] Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022. arXiv:2203.05482.
- [37] Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications, and opportunities. *ACM Comput. Surv.*, 58(8), February 2026.
- [38] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [39] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th ACM Symposium on Operating Systems Principles (SOSP)*, 2023. arXiv:2309.06180.
- [40] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.
- [41] Charles H. Martin, Tongsu Peng, and Michael W. Mahoney. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. *Nature Communications*, 12(1):4122, 2021.
- [42] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.
- [43] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the Internet of Things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (MCC '12)*, pages 13–16. ACM, 2012.
- [44] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.
- [45] Ruihao Gong, Yifu Ding, Zining Wang, Chengtao Lv, Xingyu Zheng, Jinyang Du, Yang Yong, Shiqiao Gu, Haotong Qin, Jinyang Guo, Dahua Lin, Michele Magno, and Xianglong Liu. A survey of low-bit large language models: Basics, systems, and algorithms. *Neural Networks*, 2025.
- [46] Aaron Clauset, Cosma Rohilla Shalizi, and Mark E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.
- [47] Vladimir A. Marčenko and Leonid A. Pastur. Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, 1(4):457–483, 1967.
- [48] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac’h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation (lm-evaluation-harness), 2023. EleutherAI. <https://github.com/EleutherAI/lm-evaluation-harness>.
- [49] Manne Hannula. About the Accuracy and Relevance of Deep Learning in a Challenging MRI-Image Classification Problem. In *2022 International Conference on Advanced Computing Technologies and Applications (ICACTA)*, pages 1–5, March 2022.

- [50] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2009.03300.
- [51] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge, 2018.
- [52] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019. arXiv:1905.07830.
- [53] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.
- [54] Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022. arXiv:2109.07958.
- [55] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.
- [56] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, and Henrique Ponde de Oliveira Pinto. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.
- [57] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.01210; introduces HumanEval+.
- [58] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark, 2023.
- [59] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023.
- [60] Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022. arXiv:2203.09509.
- [61] Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. What is Wrong with Perplexity for Long-context Language Modeling?, March 2025. arXiv:2410.23771 [cs] version: 3.
- [62] Abhinav Dutta, Sanjeev Krishnan, Nipun Kwatra, and Ramachandran Ramjee. Accuracy is Not All You Need, July 2024. arXiv:2407.09141 [cs].
- [63] Andrea Matarazzo and Riccardo Torlone. A survey on large language models with some insights on their capabilities and limitations. *arXiv preprint arXiv:2501.04040*, 2025.
- [64] Frederic M. Lord. *Applications of Item Response Theory to Practical Testing Problems*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1980.
- [65] Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinyBenchmarks: Evaluating LLMs with fewer examples. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. arXiv:2402.14992.
- [66] Chris G. Langton. Computation at the edge of chaos: Phase transitions and emergent computation. *Physica D: Nonlinear Phenomena*, 42(1-3):12–37, 1990.

- [67] Shiyang Zhang, Aakash Patel, Syed Rizvi, Nianchen Liu, Sizhuang He, Amin Karbasi, Emanuele Zappala, and David van Dijk. Intelligence at the edge of chaos. In *International Conference on Learning Representations (ICLR)*, 2025.
- [68] Per Bak, Chao Tang, and Kurt Wiesenfeld. Self-organized criticality: An explanation of the $1/f$ noise. *Physical Review Letters*, 59(4):381–384, 1987.
- [69] Simon Vock and Christian Meisel. Critical dynamics governs deep learning. 2026.
- [70] Didier Sornette. Dragon-kings, black swans and the prediction of crises. *International Journal of Terraspace Science and Engineering*, 2(1):1–18, 2009. arXiv:0907.4290.
- [71] Charles H. Martin. Calculated content — blog on WeightWatcher, HTSR, and SETOL. <https://calculatedcontent.com/>, 2025. Accessed July 2026.
- [72] Hari K. Prakash and Charles H. Martin. Late-stage generalization collapse in grokking: Detecting anti-grokking with WeightWatcher. *arXiv preprint arXiv:2602.02859*, 2026.
- [73] Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*, 2023.
- [74] Michael Feffer, Anusha Sinha, Wesley Hanwen Deng, Zachary C. Lipton, and Hoda Heidari. Red-teaming for generative AI: Silver bullet or security theater? In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, volume 7, pages 421–437, 2024. arXiv:2401.15897.
- [75] Yue Liu, Hongcheng Gao, Shengfang Zhai, Yufei He, Jun Xia, Zhengyu Hu, Yulin Chen, Xihong Yang, Jiaheng Zhang, Stan Z. Li, Hui Xiong, and Bryan Hooi. GuardReasoner: Towards reasoning-based LLM safeguards. *arXiv preprint arXiv:2501.18492*, 2025.
- [76] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, Reading, MA, 1977.
- [77] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300, 1995.
- [78] Joseph Lee Rodgers and W. Alan Nicewander. Thirteen Ways to Look at the Correlation Coefficient. *The American Statistician*, 42(1):59, February 1988.
- [79] Charles F. Bond and Ken Richardson. Seeing the Fisher Z -transformation. *Psychometrika*, 69(2):291–303, June 2004.
- [80] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA, 1994.
- [81] Design Patterns. <https://refactoring.guru/design-patterns>.
- [82] International Organization for Standardization. ISO/IEC 25012:2008 — software engineering — software product quality requirements and evaluation (SQuaRE) — data quality model. Technical report, ISO/IEC, Geneva, Switzerland, 2008.
- [83] DAMA International. *DAMA-DMBOK: Data Management Body of Knowledge*. Technics Publications, Basking Ridge, NJ, 2nd edition, 2017.
- [84] Research and Development | The 2025 AI Index Report | Stanford HAI. <https://hai.stanford.edu/ai-index/2025-ai-index-report/research-and-development>.
- [85] Hui Li, Xiuhua Li, Chuan Sun, Xiaofei Wang, and Victor C. M. Leung. Edge-enhanced intelligence: A comprehensive survey of large language models and edge-cloud computing synergy. *IEEE Communications Surveys & Tutorials*, 2025.
- [86] Hugging Face. GGUF – hugging face hub documentation. <https://huggingface.co/docs/hub/en/gguf>, 2026. accessed 2026-06-10.
- [87] Unsloth AI. Unsloth dynamic 2.0 GGUFs. <https://unsloth.ai/blog/dynamic-v2>, 2025. accessed 2026.

- [88] Sanjay Surendranath Girija, Shashank Kapoor, Lakshit Arora, Dipen Pradhan, Aman Raj, and Ankit Shetgaonkar. Optimizing LLMs for resource-constrained environments: A survey of model compression techniques. *arXiv preprint arXiv:2505.23786*, 2025.
- [89] Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577, 2024. arXiv:2308.07633.
- [90] Defu Liu, Yixiao Zhu, Zhe Liu, Yi Liu, Changlin Han, Jinkai Tian, Ruihao Li, and Wei Yi. A survey of model compression techniques: Past, present, and future. *Frontiers in Robotics and AI*, 12, 2025. doi:10.3389/frobt.2025.1518965.
- [91] Konwoo Kim, Suhas Kotha, Percy Liang, and Tatsunori Hashimoto. Pre-training under infinite compute, September 2025. arXiv:2509.14786 [cs].
- [92] Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning (ICML)*, 2023. arXiv:2301.00774.
- [93] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations (ICLR)*, 2024. Wanda; arXiv:2306.11695.
- [94] Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefer, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations (ICLR)*, 2024.
- [95] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [96] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.
- [97] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024.
- [98] Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. The truth is in there: Improving reasoning with layer-selective rank reduction. In *International Conference on Learning Representations (ICLR)*, 2024. LASER.
- [99] Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.17323.
- [100] Elias Frantar, Sidak Pal Singh, and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2208.11580.
- [101] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.
- [102] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-bit quantization of large language models with guarantees. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [103] Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefer, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *International Conference on Learning Representations (ICLR)*, 2024.
- [104] Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023.

- [105] S. Wang, C. Li, Y. Kang, J. Fan, Z. Ou, and A. Yao. SliderQuant: Accurate post-training quantization for large language models. In *International Conference on Learning Representations (ICLR)*, 2026.
- [106] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [107] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 30318–30332, 2022. arXiv:2208.07339.
- [108] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning (ICML)*, 2023. arXiv:2211.10438.
- [109] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.
- [110] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. *arXiv preprint arXiv:2401.18079*, 2024.
- [111] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. *arXiv preprint arXiv:2402.02750*, 2024.
- [112] A. Zandieh, M. Daliri, M. Hadian, and V. Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate. In *International Conference on Learning Representations (ICLR)*, 2026.
- [113] JanusQuant Authors. JanusQuant: Accurate and efficient 2-bit KV cache quantization for long-context inference. In *Proceedings of the 31st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, 2026. doi:10.1145/3774934.3786428.
- [114] Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. LLM-QAT: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023.
- [115] Yuzhuang Xu, Xu Han, Zonghan Yang, Shuo Wang, Qingfu Zhu, Zhiyuan Liu, Weidong Liu, and Wanxiang Che. OneBit: Towards extremely low-bit large language models. *arXiv preprint arXiv:2402.11295*, 2024.
- [116] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.14314.
- [117] A. Dremov, David Grangier, Angelos Katharopoulos, and Awni Hannun. Compute-optimal quantization-aware training. *arXiv preprint arXiv:2509.22935*, 2025.
- [118] Zifei Xu, Sayeh Sharify, and Hesham Mostafa. Mf-qat: Multi-format quantization-aware training for elastic inference. 2026.
- [119] Onat Güngör, Roshan Sood, Harold Wang, and Tajana Rosing. AQUA-LLM: Evaluating accuracy, quantization, and adversarial robustness trade-offs in LLMs for cybersecurity question answering. In *Proceedings of the 24th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2025. arXiv:2509.13514.
- [120] Elias Frantar, Roberto L. Castro, Jiale Chen, Torsten Hoefler, and Dan Alistarh. MARLIN: Mixed-precision auto-regressive parallel inference on large language models. In *Proceedings of the 30th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, 2025. arXiv:2408.11743.
- [121] ggml-org. GGUF file format specification (ggml/docs/gguf.md). <https://github.com/ggml-org/ggml/blob/master/docs/gguf.md>, 2024. accessed 2026-06-10.

- [122] Mozilla and Justine Tunney. llamafire: Distribute and run LLMs with a single file. <https://github.com/Mozilla-Ocho/llamafire>, 2024. accessed 2026.
- [123] Zonghang Li, Tao Li, Wenjiao Feng, Mohsen Guizani, Hongfang Yu, Qirong Ho, and Wei Xiang. Prima.cpp: Fast 30–70B LLM inference on heterogeneous and low-resource home clusters. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2504.08791.
- [124] Intel Corporation. Intel/auto-round: SOTA quantization algorithm for low-bit LLM inference (CPU/XPU/CUDA). <https://github.com/intel/auto-round>, 2024. accessed 2026-06-03.
- [125] Wenhua Cheng, Weiwei Zhang, Haihao Shen, Yiyang Cai, Xin He, Kaokao Lv, and Yi Liu. Optimize weight rounding via signed gradient descent for the quantization of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11332–11350, 2024. SignRound/AutoRound; arXiv:2309.05516.
- [126] Haiquan Lu, Yefan Zhou, Shiwei Liu, Zhangyang Wang, Michael W. Mahoney, and Yaoqing Yang. AlphaPruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- [127] Yihong Dong. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*, 2024.
- [128] Artyom Kharinaev, Viktor Moskvoretskii, Egor Shvetsov, Kseniia Studenikina, Mikhail Bykov, and Evgeny Burnaev. Investigating the impact of quantization methods on the safety and reliability of large language models. *arXiv preprint arXiv:2502.15799*, 2025.
- [129] Nicolò Brunello, Davide Rigamonti, Andrea Sassella, Vincenzo Scotti, and Mark James Carman. Intravisto: Inside transformer visualisation tool. *arXiv preprint arXiv:2507.13858*, 2025.
- [130] K. Pronk and Q. Zhao. Benchmarking energy efficiency of large language models using vLLM. *arXiv preprint arXiv:2509.08867*, 2025.
- [131] CodeCarbon. Codecarbon: Track emissions from compute. <https://github.com/mlco2/codecarbon>, 2026. accessed 2026-06-10.
- [132] EcoLogits. EcoLogits. <https://ecologits.ai/latest/>, 2026. accessed 2026-06-10.
- [133] Andrew Gelman and Eric Loken. The garden of forking paths: Why multiple comparisons can be a problem, even when there was no “fishing expedition”. *Department of Statistics, Columbia University*, 2014.
- [134] Kate Smith-Miles and Mario Andrés Muñoz. Instance space analysis for algorithm testing: Methodology and software tools. *ACM Computing Surveys*, 55(12):255:1–255:31, 2023. doi:10.1145/3572895.
- [135] Yusuf Berdan Güzel, Kushagra Khare, Nathan Harvey, Kian Dsouza, Dong Hyeog Jang, Junheng Chen, Cheng Ze Lam, and Mario Andrés Muñoz. instancespace: A Python package for insightful algorithm testing through instance space analysis. *SoftwareX*, 31:102246, 2025. arXiv:2501.16646.
- [136] Junyuan Hong, Jinhao Duan, Chenhui Zhang, Zhangheng Li, Chulin Xie, Kelsey Lieberman, James Diffenderfer, Brian R. Bartoldson, Ajay Kumar Jaiswal, Kaidi Xu, Bhavya Kailkhura, Dan Hendrycks, Dawn Song, Zhangyang Wang, and Bo Li. Decoding compressed trust: Scrutinizing the trustworthiness of efficient LLMs under compression. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 18611–18633. PMLR, 21–27 Jul 2024.
- [137] Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Evaluating quantized large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 28480–28524, 2024.

- [138] Kazuki Egashira, Robin Staab, Mark Vero, Jingxuan He, and Martin Vechev. Mind the gap: A practical attack on GGUF quantization. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267, pages 15038–15065. PMLR, 2025. arXiv:2505.23786.
- [139] Michał Zawalski, Meriem Boubdir, Klaudia Bałazy, Besmira Nushi, and Pablo Ribalta. Detecting data contamination in LLMs via in-context learning. <https://openreview.net/forum?id=YlpaaYxx4t>, 2025. CoDeC; OpenReview, ICLR 2026 submission.
- [140] Junyuan Hong, Jinhao Duan, Chenhui Zhang, Zhangheng Li, Chulin Xie, Kelsey Lieberman, James Diffenderfer, Brian Bartoldson, Ajay Jaiswal, Kaidi Xu, Bhavya Kailkhura, Dan Hendrycks, Dawn Song, Zhangyang Wang, and Bo Li. Decoding compressed trust: Scrutinizing the trustworthiness of efficient LLMs under compression. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 18611–18633, 2024.
- [141] Nelson Elhage. A mathematical framework for transformer circuits. *arXiv preprint arXiv:2104.XXX*, 2021. Please update this entry with correct details.
- [142] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.
- [143] Peter Deutsch. *ZLIB Compressed Data Format Specification version 3.3*, 1996. RFC 1950.
- [144] Tom B. Brown, Benjamin Mann, and Nick Ryder. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.
- [145] Jeffrey Ip and Kritin Vongthongsri. deepteam, November 2025. original-date: 2025-03-05T06:34:21Z.
- [146] John Muñoz. Pyrit: Python red teaming for llms, 2024. Available at <https://github.com/microsoft/pyrit>.
- [147] Rob Manson. Curved Inference: Concern-Sensitive Geometry in Large Language Model Residual Streams, 2025.
- [148] Bruno Gavranović, Paul Lessard, Andrew Joseph Dudzik, Tamara Von Glehn, João Guilherme Madeira Araújo, and Petar Veličković. Position: Categorical deep learning is an algebraic theory of all architectures. In *Forty-first International Conference on Machine Learning*, 2024.
- [149] Roman Vashurin, Ekaterina Fadeeva, Artem Vazhentsev, Lyudmila Rvanova, Daniil Vasilev, Akim Tsvigun, Sergey Petrakov, Rui Xing, Abdelrahman Sadallah, Kirill Grishchenkov, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, Maxim Panov, and Artem Shelmanov. Benchmarking uncertainty quantification methods for large language models with lm-polygraph. *Transactions of the Association for Computational Linguistics*, 13:220–248, 03 2025.
- [150] Omry Yadan. Hydra: A framework for elegantly configuring complex applications. <https://github.com/facebookresearch/hydra>, 2019. GitHub repository. Accessed: 2026-07-30.
- [151] Matt Mickiewicz. GGML Joins Hugging Face: What This Means for Local Model Optimization. <https://www.sitepoint.com/ggml-joins-hugging-face-what-this-means-for-local-model-optimization/>, February 2026.
- [152] Georgi Gerganov and llama.cpp contributors. llama.cpp: LLM inference in C/C++. <https://github.com/ggml-org/llama.cpp>, 2023. Accessed: 2026-07-30.
- [153] Erik Johannes Husom, Arda Goknil, Merve Astekin, Lwin Khin Shar, Andre KÅŕsen, Sagar Sen, Benedikt Andreas Mithassel, and Ahmet Soylu. Sustainable LLM Inference for Edge AI: Evaluating Quantized LLMs for Energy Efficiency, Output Accuracy, and Inference Latency. *ACM Trans. Internet Things*, 6(4):28:1–28:35, November 2025.
- [154] Zhen Li, Yupeng Su, Runming Yang, Congkai Xie, Zheng Wang, Zhongwei Xie, Ngai Wong, and Hongxia Yang. Quantization Meets Reasoning: Exploring LLM Low-Bit Quantization Degradation for Mathematical Reasoning, February 2025. arXiv:2501.03035 [cs].

- [155] Author Yang. Qwen3: Technical report, 2025. Preprint.
- [156] X. Gao. lm-evaluation-harness, 2023. GitHub repository, <https://github.com/EleutherAI/lm-evaluation-harness>.
- [157] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.
- [158] Mieke Wilms and Christoph Heitz. *Fairness vs Performance: Characterizing the Pareto Frontier of Algorithmic Decision Systems*, page 2885–2907. Association for Computing Machinery, New York, NY, USA, 2026.
- [159] Tim Dettmers. A case for quantization: AI model compression. *arXiv preprint arXiv:2306.00978*, 2023.
- [160] Patrick E. McKnight and Julius Najab. *Mann-Whitney U Test*, pages 1–1. John Wiley & Sons, Ltd, 2010.
- [161] Charlie Zhang, Graham Neubig, and Xiang Yue. On the Interplay of Pre-Training, Mid-Training, and RL on Reasoning Language Models, December 2025. *arXiv:2512.07783 [cs]*.
- [162] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of the First Conference on Language Modeling (COLM)*, pages 107547–107603, 2024.
- [163] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- [164] Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended long short-term memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pages 107547–107603, 2024.
- [165] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 35181–35224, 2024.
- [166] Patrick Chao, Edoardo DeBenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwal, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. JailbreakBench: An open robustness benchmark for jailbreaking large language models. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 37, 2024.
- [167] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in neural information processing systems*, 36:80079–80110, 2023.
- [168] National Institute of Standards and Technology. Artificial intelligence risk management framework (AI RMF 1.0). Technical Report NIST AI 100-1, National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, MD, 2023.
- [169] Author Sharma. Title of the truth paper. *Journal Name*, 2024. Details to be updated.
- [170] Anonymous. Tynyllama: An efficient open-source language model, 2023. URL: <https://github.com/jzhang38/TinyLLaMA>.
- [171] Weigao Sun, Jiayi Hu, Yucheng Zhou, Jusen Du, Disen Lan, Kexin Wang, Tong Zhu, Xiaoye Qu, Yu Zhang, and Xiaoyu Mo. Speed always wins: A survey on efficient architectures for large language models. *arXiv preprint arXiv:2508.09834*, 2025.