

**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

# **Zero-Shot Anomaly Detection via Class-Specific Semantic Alignment in Industrial Visual Inspection**

**Tiago de Oliveira Rodrigues Fontinha**



Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof. José Costa Pereira

July 31, 2025



# Resumo

A ascensão da Indústria 4.0 trouxe tecnologias avançadas para os ambientes de produção, aumentando a complexidade dos sistemas e a necessidade de automação inteligente. Um dos principais desafios atuais é o desenvolvimento de sistemas de detecção de anomalias robustos e adaptáveis, capazes de identificar falhas com dados mínimos, especialmente em cenários onde exemplos anotados de anomalias são escassos ou inexistentes.

Esta dissertação aborda este desafio através da exploração dos paradigmas de *zero-shot learning* e *few-shot learning* aplicados à detecção visual de anomalias em contextos industriais. O estudo recorre a modelos estado de arte, nomeadamente *GLASS* e *CLIP*, avaliando a sua capacidade de generalização para categorias de objetos não observadas previamente, utilizando o conjunto de dados *MVTec-AD*. Técnicas de *transfer learning* e geração sintética de dados são aplicadas para potenciar o desempenho da detecção com conjuntos de treino reduzidos. Adicionalmente, propõe-se a integração destes métodos de inteligência artificial no sistema de *workflows* da plataforma *iPortalDoc*, permitindo o controlo automático da qualidade em tempo real num ambiente de gestão documental digital.

Os resultados experimentais demonstram a viabilidade da aplicação dos métodos de *transfer learning* e *zero-shot learning* na detecção de anomalias industriais, com ganhos significativos em termos de escalabilidade e adaptabilidade face às abordagens supervisionadas tradicionais. A integração no *iPortalDoc* evidencia o potencial da solução para contextos produtivos reais, contribuindo para ecossistemas industriais mais inteligentes.

# Abstract

The rise of Industry 4.0 has introduced advanced technologies into manufacturing environments, increasing system complexity and the need for intelligent automation. Among the critical challenges is the development of robust and adaptable anomaly detection systems capable of identifying faults with minimal data, especially in scenarios where labeled examples of anomalies are rare or nonexistent.

This dissertation addresses these challenges by exploring zero-shot and few-shot learning paradigms for visual anomaly detection in industrial contexts. The research leverages state-of-the-art models, particularly GLASS and CLIP, to evaluate their capacity for generalization across unseen object categories using the MVTec-AD dataset. Transfer learning and synthetic data generation techniques are employed to further enhance detection performance with limited training samples. Additionally, a novel integration of these AI methods into the iPortalDoc workflow system is proposed, enabling real-time, automated quality control within a digital document management environment.

Experimental results demonstrate the feasibility of applying both transfer and zero-shot learning approaches to industrial anomaly detection, with significant gains in scalability and adaptability compared to traditional supervised methods. The integration into iPortalDoc illustrates the potential for extending these solutions to production environments, contributing to more intelligent manufacturing ecosystems.



# UN Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) provide a global framework to achieve a better and more sustainable future for all. It includes 17 goals<sup>1</sup> to address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice.

This dissertation contributes to several of these global goals through the application of advanced machine learning techniques to industrial environments. Specifically, it focuses on visual anomaly detection using few-shot and zero-shot learning methods, integrated into digital workflow systems.

The specific Sustainable Development Goals mentioned have the following names:

**SDG 8** Decent Work and Economic Growth

**SDG 9** Industry, Innovation and Infrastructure

**SDG 10** Reduced Inequalities

**SDG 17** Partnerships for the Goals

| SDG | Target | Contribution                                                                                                                | Performance Indicators and Metrics                                     |
|-----|--------|-----------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| 8   | 8.2    | Enhances automation and real-time quality control, promoting productivity and safer industrial operations.                  | Reduction in defect rates, increased throughput, anomaly response time |
| 9   | 9.4    | Promotes innovation through AI-driven industrial quality assurance; demonstrates scalable, efficient prototype integration. | AI model performance (AUROC, AUPRO), deployment readiness levels       |
|     | 9.5    | Strengthens industrial research and development by applying and testing cutting-edge ML paradigms.                          | Number of experimental configurations, model generalization capacity   |
| 10  | 10.2   | Supports inclusive technological growth and access to AI tools, potentially reducing disparities between industries.        | Technology adoption metrics; digital maturity indicators               |
| 17  | 17.17  | Encourages collaboration between academia, industry, and AI research communities through applied research.                  | Collaborative outputs, publications, industrial co-development efforts |

---

<sup>1</sup><https://sdgs.un.org/goals>

# Acknowledgements

Gostaria de expressar a minha sincera gratidão ao Professor José Costa Pereira, meu orientador, pela constante disponibilidade, orientação e pelas valiosas sugestões que muito contribuíram para a elaboração desta dissertação.

Agradeço, com o coração cheio, aos meus pais, aos meus avós e ao meu irmão por todo o amor, apoio incondicional e presença constante ao longo da minha vida académica e pessoal. Esta conquista não me pertence apenas a mim, é, em grande parte, o reflexo da força, dos valores e do carinho que sempre me deram.

À empresa IPBRICK SA, agradeço a oportunidade de desenvolver este trabalho, bem como a todos os seus colaboradores que me acolheram e auxiliaram durante esta etapa, com um especial reconhecimento à Eng. Telma Salgueiro pela confiança depositada, e a todo o departamento IDI pela constante disponibilidade e apoio técnico.

Por fim, deixo uma palavra de apreço aos meus amigos e colegas, cujo contributo, direto ou indireto, foi fundamental não apenas na realização desta dissertação, mas ao longo de todo o meu percurso académico.

Tiago de Oliveira Rodrigues Fontinha

*“Reality leaves a lot to the imagination”*

John Lennon

# Contents

|          |                                                               |           |
|----------|---------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                           | <b>1</b>  |
| 1.1      | Context . . . . .                                             | 1         |
| 1.2      | Motivation . . . . .                                          | 2         |
| 1.3      | Objectives . . . . .                                          | 2         |
| 1.4      | Dissertation Structure . . . . .                              | 3         |
| <b>2</b> | <b>Fundamental Concepts and Related Work</b>                  | <b>4</b>  |
| 2.1      | Anomaly Detection Review . . . . .                            | 4         |
| 2.2      | Industrial Benchmark Datasets . . . . .                       | 6         |
| 2.3      | Predominant Anomaly Detection Techniques . . . . .            | 7         |
| 2.4      | Evaluation Metrics . . . . .                                  | 9         |
| 2.5      | Anomaly Detection in Industrial Settings . . . . .            | 9         |
| 2.6      | Domain Generalization and Related Paradigms . . . . .         | 10        |
| 2.7      | Addressing Data Scarcity in Industrial Environments . . . . . | 11        |
| 2.8      | Review and Limitations of State-of-Art approaches . . . . .   | 13        |
| <b>3</b> | <b>Methodology</b>                                            | <b>15</b> |
| 3.1      | Problem Framing and Research Design . . . . .                 | 15        |
| 3.2      | MVTec-AD Dataset . . . . .                                    | 16        |
| 3.3      | Custom Test Dataset Definition . . . . .                      | 17        |
| 3.4      | Metrics . . . . .                                             | 18        |
| 3.5      | Original GLASS Inference . . . . .                            | 19        |
| 3.6      | Cross-Testing Evaluation . . . . .                            | 20        |
| 3.7      | Transfer Learning . . . . .                                   | 21        |
| 3.8      | Zero-Shot Anomaly Detection with CLIP . . . . .               | 23        |
| 3.9      | Stable Diffusion Experiment . . . . .                         | 24        |
| <b>4</b> | <b>Results and Discussion</b>                                 | <b>26</b> |
| 4.1      | Original GLASS Inference . . . . .                            | 26        |
| 4.2      | Baseline Cross-Testing Experiment . . . . .                   | 27        |
| 4.3      | Transfer Learning Experiment . . . . .                        | 28        |
| 4.3.1    | First Experiment Results . . . . .                            | 30        |
| 4.3.2    | Defective Experiment Results . . . . .                        | 31        |
| 4.3.3    | Even Experiment Results . . . . .                             | 32        |
| 4.3.4    | Defective+ Experiment Results . . . . .                       | 34        |
| 4.3.5    | Non-Defective Experiment Results . . . . .                    | 35        |
| 4.3.6    | Unsupervised Experiment Results . . . . .                     | 37        |
| 4.3.7    | Summary of Generalization Experiments . . . . .               | 38        |

|          |                                                        |           |
|----------|--------------------------------------------------------|-----------|
| 4.4      | Zero-shot Learning with CLIP . . . . .                 | 39        |
| 4.5      | Stable Diffusion Experiment . . . . .                  | 41        |
| 4.6      | Technical Discussion and Analysis of Results . . . . . | 43        |
| 4.7      | Integration with iPortalDoc . . . . .                  | 44        |
| <b>5</b> | <b>Conclusion and Future Work</b>                      | <b>48</b> |
|          | <b>References</b>                                      | <b>50</b> |
| <b>A</b> | <b>Cross-Testing Table per Class</b>                   | <b>56</b> |
| <b>B</b> | <b>Transfer Learning plots</b>                         | <b>62</b> |
| B.1      | Experiment 0 . . . . .                                 | 62        |
| B.2      | Experiment 1 . . . . .                                 | 67        |
| B.3      | Experiment 2 . . . . .                                 | 72        |
| B.4      | Experiment 3 . . . . .                                 | 77        |
| B.5      | Experiment 4 . . . . .                                 | 82        |
| B.6      | Experiment 5 . . . . .                                 | 87        |
| <b>C</b> | <b>Stable Diffusion plots</b>                          | <b>93</b> |
| <b>D</b> | <b>Zero-shot Learning plots</b>                        | <b>98</b> |

# List of Figures

|      |                                                                                                                                                                                                                       |    |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1  | Proportion of anomaly detection methods categorized by learning approach, adapted from [9]. . . . .                                                                                                                   | 5  |
| 4.1  | Layout showing real and synthetic images for different MVTec AD classes. Each row includes real non-defective and defective samples, as well as synthetic non-defective and conditioned (defective) images. . . . .   | 29 |
| 4.2  | Examples of synthetic images generated from text prompts for each of the five test classes: <i>pill</i> , <i>metal_nut</i> , <i>tile</i> , <i>wood</i> , and <i>transistor</i> . . . . .                              | 42 |
| 4.3  | Menu interface in iPortalDoc displayed after document creation, in a folder called "Inspeção Visual" for testing purposes, allowing users to manage the document and initiate the anomaly detection workflow. . . . . | 45 |
| 4.4  | Document information interface in <i>iPortalDoc</i> showing the classification result and confidence score after CLIP-based anomaly detection. . . . .                                                                | 46 |
| 4.5  | Overall workflow architecture and data flow involved in this integration. . . . .                                                                                                                                     | 46 |
| B.1  | Precision-Recall Curve for class <i>pill</i> in Experiment 0. . . . .                                                                                                                                                 | 62 |
| B.2  | ROC Curve for class <i>pill</i> in Experiment 0. . . . .                                                                                                                                                              | 63 |
| B.3  | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 0. . . . .                                                                                                                                            | 63 |
| B.4  | ROC Curve for class <i>metal_nut</i> in Experiment 0. . . . .                                                                                                                                                         | 64 |
| B.5  | Precision-Recall Curve for class <i>transistor</i> in Experiment 0. . . . .                                                                                                                                           | 64 |
| B.6  | ROC Curve for class <i>transistor</i> in Experiment 0. . . . .                                                                                                                                                        | 65 |
| B.7  | Precision-Recall Curve for class <i>tile</i> in Experiment 0. . . . .                                                                                                                                                 | 65 |
| B.8  | ROC Curve for class <i>tile</i> in Experiment 0. . . . .                                                                                                                                                              | 66 |
| B.9  | Precision-Recall Curve for class <i>wood</i> in Experiment 0. . . . .                                                                                                                                                 | 66 |
| B.10 | ROC Curve for class <i>wood</i> in Experiment 0. . . . .                                                                                                                                                              | 67 |
| B.11 | Precision-Recall Curve for class <i>pill</i> in Experiment 1. . . . .                                                                                                                                                 | 67 |
| B.12 | ROC Curve for class <i>pill</i> in Experiment 1. . . . .                                                                                                                                                              | 68 |
| B.13 | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 1. . . . .                                                                                                                                            | 68 |
| B.14 | ROC Curve for class <i>metal_nut</i> in Experiment 1. . . . .                                                                                                                                                         | 69 |
| B.15 | Precision-Recall Curve for class <i>transistor</i> in Experiment 1. . . . .                                                                                                                                           | 69 |
| B.16 | ROC Curve for class <i>transistor</i> in Experiment 1. . . . .                                                                                                                                                        | 70 |
| B.17 | Precision-Recall Curve for class <i>tile</i> in Experiment 1. . . . .                                                                                                                                                 | 70 |
| B.18 | ROC Curve for class <i>tile</i> in Experiment 1. . . . .                                                                                                                                                              | 71 |
| B.19 | Precision-Recall Curve for class <i>wood</i> in Experiment 1. . . . .                                                                                                                                                 | 71 |
| B.20 | ROC Curve for class <i>wood</i> in Experiment 1. . . . .                                                                                                                                                              | 72 |
| B.21 | Precision-Recall Curve for class <i>pill</i> in Experiment 2. . . . .                                                                                                                                                 | 72 |
| B.22 | ROC Curve for class <i>pill</i> in Experiment 2. . . . .                                                                                                                                                              | 73 |
| B.23 | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 2. . . . .                                                                                                                                            | 73 |

|      |                                                                             |    |
|------|-----------------------------------------------------------------------------|----|
| B.24 | ROC Curve for class <i>metal_nut</i> in Experiment 2. . . . .               | 74 |
| B.25 | Precision-Recall Curve for class <i>transistor</i> in Experiment 2. . . . . | 74 |
| B.26 | ROC Curve for class <i>transistor</i> in Experiment 2. . . . .              | 75 |
| B.27 | Precision-Recall Curve for class <i>tile</i> in Experiment 2. . . . .       | 75 |
| B.28 | ROC Curve for class <i>tile</i> in Experiment 2. . . . .                    | 76 |
| B.29 | Precision-Recall Curve for class <i>wood</i> in Experiment 2. . . . .       | 76 |
| B.30 | ROC Curve for class <i>wood</i> in Experiment 2. . . . .                    | 77 |
| B.31 | Precision-Recall Curve for class <i>pill</i> in Experiment 3. . . . .       | 77 |
| B.32 | ROC Curve for class <i>pill</i> in Experiment 3. . . . .                    | 78 |
| B.33 | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 3. . . . .  | 78 |
| B.34 | ROC Curve for class <i>metal_nut</i> in Experiment 3. . . . .               | 79 |
| B.35 | Precision-Recall Curve for class <i>transistor</i> in Experiment 3. . . . . | 79 |
| B.36 | ROC Curve for class <i>transistor</i> in Experiment 3. . . . .              | 80 |
| B.37 | Precision-Recall Curve for class <i>tile</i> in Experiment 3. . . . .       | 80 |
| B.38 | ROC Curve for class <i>tile</i> in Experiment 3. . . . .                    | 81 |
| B.39 | Precision-Recall Curve for class <i>wood</i> in Experiment 3. . . . .       | 81 |
| B.40 | ROC Curve for class <i>wood</i> in Experiment 3. . . . .                    | 82 |
| B.41 | Precision-Recall Curve for class <i>pill</i> in Experiment 4. . . . .       | 82 |
| B.42 | ROC Curve for class <i>pill</i> in Experiment 4. . . . .                    | 83 |
| B.43 | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 4. . . . .  | 83 |
| B.44 | ROC Curve for class <i>metal_nut</i> in Experiment 4. . . . .               | 84 |
| B.45 | Precision-Recall Curve for class <i>transistor</i> in Experiment 4. . . . . | 84 |
| B.46 | ROC Curve for class <i>transistor</i> in Experiment 4. . . . .              | 85 |
| B.47 | Precision-Recall Curve for class <i>tile</i> in Experiment 4. . . . .       | 85 |
| B.48 | ROC Curve for class <i>tile</i> in Experiment 4. . . . .                    | 86 |
| B.49 | Precision-Recall Curve for class <i>wood</i> in Experiment 4. . . . .       | 86 |
| B.50 | ROC Curve for class <i>wood</i> in Experiment 4. . . . .                    | 87 |
| B.51 | Precision-Recall Curve for class <i>pill</i> in Experiment 5. . . . .       | 87 |
| B.52 | ROC Curve for class <i>pill</i> in Experiment 5. . . . .                    | 88 |
| B.53 | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 5. . . . .  | 88 |
| B.54 | ROC Curve for class <i>metal_nut</i> in Experiment 5. . . . .               | 89 |
| B.55 | Precision-Recall Curve for class <i>transistor</i> in Experiment 5. . . . . | 89 |
| B.56 | ROC Curve for class <i>transistor</i> in Experiment 5. . . . .              | 90 |
| B.57 | Precision-Recall Curve for class <i>tile</i> in Experiment 5. . . . .       | 90 |
| B.58 | ROC Curve for class <i>tile</i> in Experiment 5. . . . .                    | 91 |
| B.59 | Precision-Recall Curve for class <i>wood</i> in Experiment 5. . . . .       | 91 |
| B.60 | ROC Curve for class <i>wood</i> in Experiment 5. . . . .                    | 92 |
| C.1  | Precision-Recall Curve for class <i>pill</i> in Experiment 6. . . . .       | 93 |
| C.2  | ROC Curve for class <i>pill</i> in Experiment 6. . . . .                    | 94 |
| C.3  | Precision-Recall Curve for class <i>metal_nut</i> in Experiment 6. . . . .  | 94 |
| C.4  | ROC Curve for class <i>metal_nut</i> in Experiment 6. . . . .               | 95 |
| C.5  | Precision-Recall Curve for class <i>tile</i> in Experiment 6. . . . .       | 95 |
| C.6  | ROC Curve for class <i>tile</i> in Experiment 6. . . . .                    | 96 |
| C.7  | Precision-Recall Curve for class <i>wood</i> in Experiment 6. . . . .       | 96 |
| C.8  | ROC Curve for class <i>wood</i> in Experiment 6. . . . .                    | 97 |
| D.1  | Precision-Recall Curve for class <i>pill</i> in ZSL Experiment. . . . .     | 98 |
| D.2  | ROC Curve for class <i>pill</i> in ZSL Experiment. . . . .                  | 99 |

|     |                                                                              |     |
|-----|------------------------------------------------------------------------------|-----|
| D.3 | Precision-Recall Curve for class <i>metal_nut</i> in ZSL Experiment. . . . . | 99  |
| D.4 | ROC Curve for class <i>metal_nut</i> in ZSL Experiment. . . . .              | 100 |
| D.5 | Precision-Recall Curve for class <i>tile</i> in ZSL Experiment. . . . .      | 100 |
| D.6 | ROC Curve for class <i>tile</i> in ZSL Experiment. . . . .                   | 101 |
| D.7 | Precision-Recall Curve for class <i>wood</i> in ZSL Experiment. . . . .      | 101 |
| D.8 | ROC Curve for class <i>wood</i> in ZSL Experiment. . . . .                   | 102 |



# List of Tables

|      |                                                                                                   |    |
|------|---------------------------------------------------------------------------------------------------|----|
| 3.1  | Summary of the MVTec-AD Dataset Composition by Category, Image Type, and Resolution [12]          | 16 |
| 3.2  | Distribution of the adopted Fine-tuning Dataset                                                   | 17 |
| 3.3  | Distribution of good and defective samples in the defined test dataset.                           | 20 |
| 3.4  | Summary of Transfer Learning Configurations                                                       | 22 |
| 4.1  | Performance metrics and confusion matrices for GLASS inference on MVTec-AD (Original)             | 27 |
| 4.2  | Data Configurations for Transfer Learning Experiments                                             | 29 |
| 4.3  | Performance metrics and confusion matrices for MVTec Cable class - First Experiment               | 30 |
| 4.4  | AUROC Comparison – First Experiment vs Baseline                                                   | 30 |
| 4.5  | Performance metrics and confusion matrices for MVTec Cable class - Defective Experiment           | 31 |
| 4.6  | AUROC Comparison – Defective Experiment vs Baseline                                               | 32 |
| 4.7  | Performance metrics and confusion matrices for MVTec Cable class - Even Experiment                | 33 |
| 4.8  | AUROC Comparison – Even Experiment vs Baseline                                                    | 33 |
| 4.9  | Performance metrics and confusion matrices for MVTec Cable class - Defective+ Experiment          | 34 |
| 4.10 | AUROC Comparison – Defective+ Experiment vs Baseline                                              | 34 |
| 4.11 | Performance metrics and confusion matrices for MVTec Cable class - Non-Defective Experiment       | 36 |
| 4.12 | AUROC Comparison – Non-Defective Experiment vs Baseline                                           | 36 |
| 4.13 | Performance metrics and confusion matrices for MVTec Cable class - Unsupervised Experiment        | 37 |
| 4.14 | AUROC Comparison – Unsupervised Experiment vs Baseline                                            | 37 |
| 4.15 | Zero-Shot CLIP-Based Anomaly Detection Performance on Unseen MVTec-AD Classes                     | 40 |
| 4.16 | Performance metrics and confusion matrices for MVTec Cable-based classification on target classes | 43 |
| A.1  | Cross-testing results with model trained on <i>carpet</i> .                                       | 56 |
| A.2  | Cross-testing results with model trained on <i>cable</i> in the 5 selected test classes.          | 56 |
| A.3  | Cross-testing results with model trained on <i>cable</i> .                                        | 57 |
| A.4  | Cross-testing results with model trained on <i>capsule</i> .                                      | 57 |
| A.5  | Cross-testing results with model trained on <i>hazelnut</i> .                                     | 57 |
| A.6  | Cross-testing results with model trained on <i>grid</i> .                                         | 58 |

|      |                                                                         |    |
|------|-------------------------------------------------------------------------|----|
| A.7  | Cross-testing results with model trained on <i>leather</i> . . . . .    | 58 |
| A.8  | Cross-testing results with model trained on <i>metal_nut</i> . . . . .  | 58 |
| A.9  | Cross-testing results with model trained on <i>pill</i> . . . . .       | 59 |
| A.10 | Cross-testing results with model trained on <i>screw</i> . . . . .      | 59 |
| A.11 | Cross-testing results with model trained on <i>tile</i> . . . . .       | 59 |
| A.12 | Cross-testing results with model trained on <i>toothbrush</i> . . . . . | 60 |
| A.13 | Cross-testing results with model trained on <i>transistor</i> . . . . . | 60 |
| A.14 | Cross-testing results with model trained on <i>wood</i> . . . . .       | 60 |
| A.15 | Cross-testing results with model trained on <i>bottle</i> . . . . .     | 61 |
| A.16 | Cross-testing results with model trained on <i>zipper</i> . . . . .     | 61 |

# List of Acronyms

|        |                                                        |
|--------|--------------------------------------------------------|
| AD     | Anomaly Detection                                      |
| IT     | Information and Technology                             |
| AUROC  | Area Under the Receiver Operating Characteristic Curve |
| AUPRC  | Area Under the Precision-Recall Curve                  |
| CLIP   | Contrastive Language–Image Pretraining                 |
| FPR    | False Positive Rate                                    |
| FP     | False Positive                                         |
| FN     | False Negative                                         |
| TP     | True Positive                                          |
| TN     | True Negative                                          |
| GLASS  | Global and Local Anomaly co-Synthesis Strategy         |
| GAS    | Global Anomaly Synthesis                               |
| LAS    | Local Anomaly Synthesis                                |
| JSON   | JavaScript Object Notation                             |
| ZSL    | Zero-Shot Learning                                     |
| TL     | Transfer Learning                                      |
| PR AUC | Precision–Recall Area Under the Curve                  |
| GPU    | Graphics Processing Unit                               |
| FP16   | 16-bit Floating Point Precision                        |
| ViT    | Vision Transformer                                     |
| ZSL    | Zero-Shot Learning                                     |
| FSL    | Few-Shot Learning                                      |
| FT     | Fine-Tuning                                            |
| TL     | Transfer Learning                                      |
| FPR    | False Positive Rate                                    |
| TPR    | True Positive Rate                                     |

# Chapter 1

## Introduction

### 1.1 Context

The advent of Industry 4.0 has significantly transformed industrial systems, making manufacturing processes more complex, automated, and interconnected. With the integration of technologies such as cyber-physical systems, artificial intelligence (AI), and the Internet of Things (IoT), industries now benefit from real-time monitoring and data-driven decision-making. However, this complexity also introduces new challenges in maintaining system reliability, ensuring product quality, and quickly identifying faults before they cause major disruptions.

Anomalies, ranging from equipment malfunctions to deviations in production quality, can lead to operational downtime, increased costs, and compromised safety. In highly automated environments, the ability to detect these anomalies promptly and accurately is critical. Traditional anomaly detection systems often struggle with scalability and adaptability, particularly as industrial systems evolve and diversify.

A practical illustration of these challenges emerged in a recent project by IPBRICK S.A., a Portuguese technology company specialized in secure and integrated digital solutions that enhance organizational efficiency and autonomy. At the core of IPBRICK's technological ecosystem lies the IPBRICK OS, a modular, interoperable operating environment that supports multiple integrated services for communication, collaboration, and information management across various departments.

One of its most prominent platforms is iPortalDoc, a document and process management system that enables digitized workflows, document lifecycle tracking, and customizable actions. Organizations use iPortalDoc to define document types, associate them with workflows, and automate specific operations at each step, such as sending alerts, invoking APIs, generating reports, or modifying access permissions. This level of flexibility has made iPortalDoc a key tool in both public and private sectors, particularly in domains requiring strict compliance, such as legal and administrative operations.

As iPortalDoc expands into industrial contexts, particularly within quality control workflows, new demands emerge. Users frequently upload images of physical components into the platform

during inspection procedures. However, these images are still analyzed manually by human operators, which limits consistency, scalability, and efficiency.

This dissertation addresses that limitation by proposing the integration of computer vision-based anomaly detection into the iPortalDoc workflow engine. Through machine learning models capable of classifying product images as “good” or “defective,” the system would gain the ability to automatically assess conformity, detect anomalies, and route workflow actions accordingly, such as flagging for rework, approving for dispatch, or generating alerts. This approach enhances iPortalDoc as a smart decision-support platform and aligns with broader initiatives in Industry 4.0 and intelligent automation.

## 1.2 Motivation

Traditional supervised anomaly detection techniques generally require large volumes of labeled data to achieve high accuracy. In industrial environments, however, such labeled datasets are often unavailable, especially for new or rare anomalies. Additionally, models trained in one setting may not perform well in another due to differences in component characteristics, environmental factors, or production processes.

These limitations underscore the need for more flexible and generalizable approaches. Learning paradigms such as zero-shot and few-shot learning are promising solutions, as they enable models to detect anomalies with little or no training data specific to the task. These methods are particularly valuable when exhaustive labeling is impractical or prohibitively expensive.

In the project conducted by IPBRICK, these challenges were particularly evident. The client required an anomaly detection system for a diverse set of industrial components, but no labeled image data from the production environment was available. Moreover, the components were not manufactured by IPBRICK and changed frequently, making conventional training approaches inefficient. General-purpose datasets such as CIFAR [1] or ImageNet [2] were insufficient, as they lacked the domain-specific characteristics needed for reliable defect detection.

Given this context, the research focuses on exploring the potential of pre-trained models and generalization-based learning strategies for anomaly detection in industrial imagery. By leveraging zero-shot and few-shot learning, the goal is to reduce reliance on large labeled datasets while maintaining performance and adaptability. Embedding these models into iPortalDoc represents a step toward a smarter, more autonomous digital platform, capable of supporting quality assurance processes within modern manufacturing environments.

## 1.3 Objectives

The main objective of this dissertation is to incorporate advanced visual anomaly detection capabilities into iPortalDoc’s workflow automation engine. The research aims to investigate learning strategies that can identify visual defects in industrial components with strong generalization ability and minimal dependency on labeled data. The focus is on adapting zero-shot and few-shot

learning techniques to build a robust anomaly detection system compatible with the IPBRICK.OS ecosystem, thereby aligning iPortalDoc with the demands of Industry 4.0 and expanding its functional scope to include intelligent quality control support.

## 1.4 Dissertation Structure

This dissertation is structured as follows:

Chapter 2 presents the theoretical foundations, existing research, and state-of-the-art approaches to anomaly detection.

Chapter 3 details the methodology, including datasets, models, and evaluation metrics used.

Chapter 4 discusses the experimental results and provides an in-depth analysis.

Finally, Chapter 5 concludes the dissertation and outlines possible directions for future research.

## Chapter 2

# Fundamental Concepts and Related Work

### 2.1 Anomaly Detection Review

The detection of anomalies represents a long-standing and complex problem that spans numerous domains. Broadly speaking, it involves identifying patterns or behaviors in data that deviate from what is considered typical or expected [3]. At the core of anomaly detection systems is the assumption that abnormal events or observations differ markedly from standard behavior [4], with such deviations typically flagged as anomalies or intrusions.

The applicability of anomaly detection spans a wide array of sectors, including early diagnosis in medical fields [5], predictive maintenance in industrial systems, fraud detection in finance, and cybersecurity monitoring. These techniques are indispensable for enhancing system efficiency, reliability, and resilience.

Historically, statistical techniques formed the foundation of anomaly detection. These methods typically modeled normal behavior patterns and used hypothesis testing or statistical thresholds to identify outliers [6]. Approaches such as parametric modeling and proximity-based techniques were commonly applied. With the advent of machine learning, newer models now automatically learn the distinction between normal and abnormal patterns across varied datasets and applications [7].

From a methodological standpoint, anomaly detection approaches are generally grouped based on the nature of the training data used to construct the model [3], [8]:

- **Supervised Anomaly Detection:** This technique requires labeled datasets containing both normal and anomalous instances. Models are trained to discriminate between these two classes. However, such methods face notable challenges: anomaly samples are usually underrepresented, leading to class imbalance, and obtaining comprehensive and accurate anomaly labels is often difficult due to their rarity and heterogeneity.

- **Semi-Supervised Anomaly Detection:** These methods are trained using labeled data representing only normal behavior. Any significant deviation from this learned representation is classified as anomalous. This strategy is often more practical since it avoids the need for labeled anomalies, relying instead on the assumption that anomalies are infrequent.
- **Unsupervised Anomaly Detection:** In unsupervised settings, models operate without any labeled data, relying on assumptions such as the predominance of normal samples over anomalous ones. While this assumption simplifies deployment, it can also result in false positives if the distribution of anomalies is not sparse. Many semi-supervised techniques can be repurposed for unsupervised use by treating the available data as entirely normal and flagging deviations during inference.

A study reviewed in [9] highlights the distribution of anomaly detection strategies across literature: 27% of approaches employed unsupervised methods, making it the most prevalent. In contrast, 18% applied supervised learning, while 7% integrated both supervised and unsupervised techniques. Semi-supervised methods accounted for 5%, and only 1% involved a mix of semi-supervised and unsupervised approaches. Notably, 42% of the reviewed works did not specify their methodological classification.

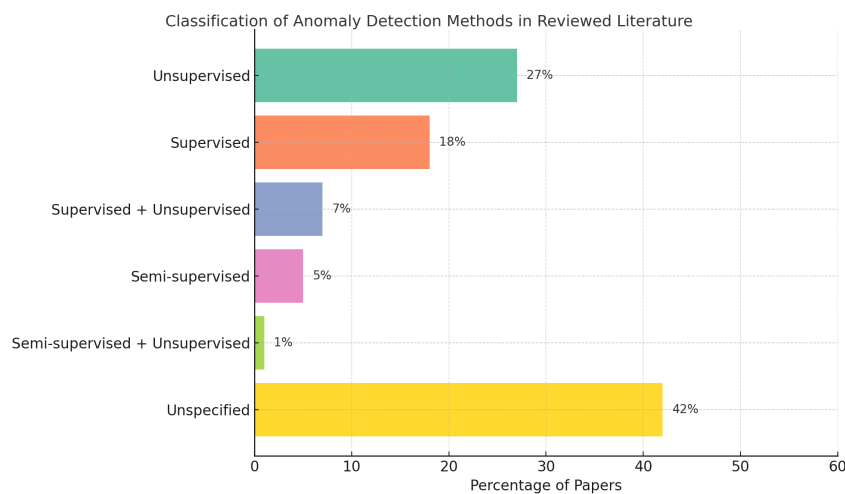


Figure 2.1: Proportion of anomaly detection methods categorized by learning approach, adapted from [9].

In industrial applications, unsupervised anomaly detection is becoming the preferred approach, primarily due to the practical difficulties associated with obtaining labeled defect data. For example, in software system logs, manually labeling large-scale datasets is time-consuming and expensive, making label-free detection methods more feasible [10]. Moreover, the dynamic nature of industrial environments—where machinery and processes are frequently updated—reduces the effectiveness of static models trained on labeled data. Unsupervised approaches, by focusing on modeling the normal state, offer greater adaptability and are well-suited for real-world scenarios where anomalies are rare, diverse, and unpredictable [11]. Additionally, these methods tend to



be more scalable and cost-efficient, as they do not rely on costly annotation processes and can be applied across vast datasets.

## 2.2 Industrial Benchmark Datasets

Detecting anomalies in industrial environments is fundamental to ensuring automated quality control and operational reliability. Over time, several benchmark datasets have been curated to facilitate the evaluation and comparison of industrial anomaly detection (IAD) algorithms. These datasets are instrumental in validating the robustness and generalization capabilities of different models across realistic industrial scenarios. Among the most prominent datasets in this domain are MVTec AD [12], [13], Real-IAD [14], IM-IAD [15], BTAD [16], HSS-IAD [17], ViSa [18], and MPDD [19], each offering unique characteristics for benchmarking purposes.

The MVTec AD dataset [12], [13] is one of the most widely adopted benchmarks for evaluating industrial anomaly detection methods. It contains over 5,000 high-resolution images across 15 categories, covering both object types—such as bottles, cables, and leather—and textures like carpets, grids, and tiles. Each image is accompanied by detailed pixel-level annotations, enabling precise localization of surface defects. The dataset includes a wide range of anomaly types, such as scratches, contamination, and structural deformations, making it suitable for both supervised and unsupervised learning approaches. Thanks to its diversity and consistent ground truth, MVTec AD has become a cornerstone in visual anomaly detection research. As stated by Zheng *et al.*, “The MVTec AD dataset is widely used in anomaly detection and localization,” and “From the time the MVTec AD dataset was proposed to the present, new research methods that are constantly being proposed push its precision to saturation” [20].

The Real-IAD dataset [14] introduces an extensive set of over 150,000 high-resolution images across 30 categories of industrial objects. It aims to simulate real-world conditions by including a variety of imperfections and environmental noise, such as lighting variations and manufacturing inconsistencies. A notable contribution of Real-IAD is the introduction of the Fully Unsupervised Industrial Anomaly Detection (FUIAD) setting, which reflects practical constraints where annotated labels are unavailable. This dataset is particularly valuable for assessing the real-world adaptability of anomaly detection systems.

IM-IAD [15] focuses on evaluating algorithms under varying levels of supervision, including supervised, semi-supervised, and unsupervised settings. It contains a broad range of industrial components, allowing researchers to test methods across different learning paradigms and real-world defect types. The dataset supports detailed evaluations for classification and localization tasks, making it useful for comparing performance across a variety of IAD techniques.

The BTAD dataset [16], developed by beanTech, targets the detection of surface and internal defects in industrial products. It includes 2,830 real images spanning three product categories and is tailored to evaluate models on small-scale, real-world defects. BTAD supports both supervised and unsupervised evaluation, providing a practical benchmark for fine-grained anomaly localization in industrial settings.

HSS-IAD [17] fills a niche by emphasizing subtle and hard-to-detect anomalies in visually similar parts. It presents challenges for both human inspectors and AI models, focusing on nuanced physical variations within same-sort items. The dataset is designed to test the sensitivity and robustness of detection methods, especially in identifying minute and easily overlooked defects.

ViSa (Visual Anomaly) [18] offers a more complex and diverse evaluation environment than existing benchmarks like MVTec AD. It includes 10,821 color images, with 9,621 normal and 1,200 anomalous samples, spread across multiple domains and featuring intricate structures. ViSa introduces multiple anomaly types per object, such as anomalies in printed circuit boards (PCBs) and cluttered scenes with multiple object instances. The dataset challenges models that previously reached near-saturation performance on simpler benchmarks, promoting the development of more robust and generalized approaches.

The MPDD (Metal Parts Defect Detection) dataset [19] centers on anomaly detection in metal manufacturing scenarios. It contains six classes of metal components, offering 888 normal training images and 458 testing samples, which include both normal and defective instances. The dataset features pixel-level annotations and includes variations in position and rotation during image acquisition. Captured with consumer-grade cameras under controlled lighting, MPDD provides a realistic testbed for assessing algorithms' capacity to detect common manufacturing defects, such as scratches, dents, and paint issues.

Collectively, these datasets have significantly advanced the development and evaluation of industrial anomaly detection systems. Each offering distinct challenges, from varying defect types and levels of supervision to differences in image quality and complexity. Among them, MVTec AD remains the most widely used due to its comprehensive structure, well-balanced scale, and extensive use in literature. For these reasons, this dataset has been selected for the present study, offering a reliable baseline and consistent framework for performance assessment and comparative analysis.

## 2.3 Predominant Anomaly Detection Techniques

Anomaly detection has become a central task in industrial monitoring, leading to the development of a wide range of techniques aimed at accurately identifying deviations from normal behavior [21], [22]. These methods are generally organized into key categories, such as reconstruction-based [23], [24], embedding-based [25], [26], memory bank approaches [27], [28], one-class classification [29], normalizing flow models [30], [31], knowledge distillation strategies [25], [32], and synthesis-based techniques [23], [33]. This section presents an overview of these categories and highlights representative methods within each.

Reconstruction-based methods rely on the assumption that anomalies are harder to reconstruct than normal samples. Techniques like autoencoders (AEs) [34] aim to reconstruct input images, with the reconstruction error serving as an anomaly score. Variational Autoencoders (VAEs) [24] extend this idea by modeling the input distribution in a latent space, allowing for a probabilistic interpretation of the reconstruction. Despite their effectiveness, these models often require large

training datasets, are computationally demanding, and can be sensitive to hyperparameter tuning. An alternative is RecForest [35], which applies a random forest framework to reconstruction tasks, offering advantages such as simpler training, fewer hyperparameters, and better handling of irrelevant features in high-dimensional spaces.

Embedding-based methods extract feature representations from images using pre-trained deep neural networks. These features are typically compressed to form a compact embedding of normal data, while anomalies are detected as points that lie outside the dense region of the embedding space [36]. While effective, a key limitation is that these methods are trained solely on normal instances, which can result in reduced performance when encountering complex or novel anomalies. DeepSVDD [37] exemplifies this class, learning a hypersphere in latent space that encloses normal data, with anomalies identified by their distance from the center.

Memory bank approaches [27], [28] maintain a repository of representative features extracted from normal training data. At inference, these stored features are compared with those of new samples using similarity metrics. For example, PatchCore [27] stores image patches from normal data in memory and assesses anomalies based on patch-wise similarity. These methods are valued for their capacity to preserve high-quality representations and their strong performance across diverse industrial tasks.

One-class classification (OCC) methods define explicit decision boundaries around normal data, typically using hyperspheres or hyperplanes. One-Class SVM [38], a classical OCC method, constructs a boundary that separates normal instances from potential anomalies in the feature space. While suitable when only normal data is available, OCC approaches may struggle with borderline or subtle anomalies that closely resemble the training distribution [39].

Normalizing flow (NF) models [40], [41] provide a powerful generative framework by learning invertible mappings from data to a known latent distribution. This enables precise likelihood estimation, which can be used to detect low-probability (i.e., anomalous) samples. NFAD [41] trains a flow-based model on normal data, then generates pseudo-anomalies by sampling from the distribution's tails. These samples are used to train a binary classifier for anomaly detection. U-Flow [31] builds on this idea by integrating a U-shaped architecture and self-supervised loss, improving generalization and anomaly localization in industrial benchmarks.

Knowledge distillation (KD) methods use a teacher-student framework, where the teacher model (trained on both normal and anomalous data) transfers knowledge to a student trained only on normal data. Discrepancies between teacher and student outputs are then used to flag anomalies. CSKD [32] applies cosine similarity to enhance feature alignment between networks, improving performance on benchmarks like MVTec AD. Dual-student distillation methods, such as DSKD [42], leverage two student networks to capture both high- and low-level features, offering a more comprehensive anomaly representation.

Synthesis-based approaches generate artificial anomalies to augment the training process and improve model robustness. Most of these methods introduce anomalies at the image level. Cut-Paste [33] creates synthetic defects by cutting and repositioning patches within normal images.

NSA [43] uses Poisson blending to create more natural-looking anomalies. DRAEM [23] introduces textures from external sources into random regions defined by Perlin noise, simulating varied and realistic defect patterns.

More recent synthesis techniques operate in the feature space. DSR [44] modifies codebook features to simulate subtle anomalies. Other methods, such as SimpleNet [45] and UniAD [46], inject Gaussian noise into normal features to generate diverse synthetic defects. While image-level synthesis produces visually realistic anomalies, it may lack diversity; feature-level synthesis is more efficient but may suffer from limited control over anomaly semantics.

Each method outlined above brings specific advantages and challenges. Reconstruction-based methods can struggle with fine-grained image details, while embedding-based approaches may underperform when anomalies are similar to the normal data. Memory bank and OCC models are often sensitive to noise or borderline cases. Generative methods, including normalizing flows and synthesis-based techniques, can be computationally demanding and complex to tune. Knowledge distillation approaches offer promise but depend on high-quality teacher networks. Future research should focus on combining these strengths while addressing their respective limitations to develop more generalizable and efficient anomaly detection systems.

## 2.4 Evaluation Metrics

Evaluating anomaly detection models requires metrics that go beyond simple accuracy, especially due to the inherent class imbalance and the need to detect rare or unseen defects [47]. As a result, the literature has adopted a combination of threshold independent metrics, such as AUROC and AUPRC, for assessing overall discriminative capacity [12], and threshold dependent metrics, including Precision, Recall, and F1 Score, for evaluating performance under specific decision thresholds [48].

In addition to these classification oriented metrics, several works have also focused on spatial localization performance using region level and pixel level evaluations. Metrics such as I-AUROC and P-AUROC [13], PRO and AUPRO [49], and IoU [50] allow for a more fine grained analysis of how well a model identifies not only the presence but also the exact location of anomalies.

This combination of evaluation strategies, spanning both global classification performance and local spatial accuracy, enables a robust assessment of model capabilities across diverse testing scenarios, including generalization to unseen categories and highly imbalanced datasets [27].

## 2.5 Anomaly Detection in Industrial Settings

In industrial anomaly detection, particularly in the context of the MVTec-AD dataset, hybrid methods have gained considerable traction due to their ability to blend different strategies for enhanced performance. These methods typically combine principles from reconstruction-based,

embedding-based, and memory-based approaches to improve both the identification and localization of anomalies. By integrating complementary techniques, hybrid models can mitigate the limitations inherent to any single method when applied independently.

One such example is GLASS (Global and Local Anomaly co-Synthesis Strategy), which employs a dual-level synthesis strategy—introducing anomalies at both the image and pixel levels. It fuses reconstruction-based learning with geometric anomaly scoring in the latent feature space, thus improving sensitivity to subtle and spatially distributed defects. GLASS achieves impressive results on MVTec AD, with an AUROC of 99.9% at the image level and 96.8% at the pixel level [51].

UniNet leverages contrastive learning along with targeted feature selection to detect anomalies. Originally associated with embedding-based techniques, contrastive learning in UniNet is adapted to isolate meaningful patterns within the feature space. This hybrid design enhances the model's ability to generalize across varying industrial settings, achieving an image-level AUROC of 99.90% and pixel-level AUROC of 96.00% on MVTec AD [52].

The HETMM (Hard-normal Example-aware Template Mutual Matching) framework tackles the challenge of distinguishing visually subtle anomalies. It applies a matching mechanism that compares templates to identify cases where normal and defective items appear visually similar—a common difficulty in industrial applications. HETMM reaches an image-level AUROC of 99.8% and pixel-level AUROC of 96.4% [53].

INP-Former ViT-L (Intrinsic Normal Prototypes Former) utilizes a Vision Transformer architecture to extract domain-specific normal prototypes directly from test images. This allows the model to adapt dynamically to unseen defect types. By modeling long-range dependencies and structural regularities, INP-Former ViT-L achieves strong generalization, with an image-level AUROC of 99.8% and pixel-level AUROC of 95.6% [54].

PBAS (Progressive Boundary-guided Anomaly Synthesis) builds on the idea of generating synthetic feature-level anomalies by progressively refining decision boundaries. It aims to capture fine-grained distinctions between normal and abnormal data through controlled synthesis guided by class boundaries. This approach results in an image-level AUROC of 99.8% and a remarkable pixel-level AUROC of 98.6% on MVTec AD [55].

In this research, GLASS was selected due to its outstanding performance and ease of implementation. Despite certain limitations in generalizing to completely unseen object types, its accuracy on known categories, publicly available implementation, and relatively straightforward deployment make it an effective and practical choice. GLASS offers a strong balance between usability and detection capability, making it suitable for both research and applied industrial environments.

## 2.6 Domain Generalization and Related Paradigms

Domain Generalization (DG) [56] is an emerging research area that addresses the challenge of deploying machine learning models in environments that differ from those seen during training.

The goal is to enable models to perform reliably across a range of unseen data distributions. Several other paradigms intersect with DG, including transfer learning [57], domain adaptation [58], multi-task learning [59], meta-learning [60], lifelong learning [61], and zero-shot learning [62], each contributing complementary methodologies.

Multi-task learning involves training a model across multiple related tasks simultaneously. By sharing representations, it can improve learning efficiency and generalization on the primary task [59]. A specific variation, multi-domain learning, focuses on training across different domains rather than distinct tasks. However, it aims at performance within those domains rather than generalization to entirely new ones.

Transfer learning operates by training a model on a source domain and adapting it to a related target domain, typically through fine-tuning. Although powerful, this paradigm assumes access to data from the target domain during training. DG, in contrast, explicitly assumes that the target domain is unavailable, requiring the model to generalize to new domains without prior exposure [57].

Domain Adaptation (DA) also addresses distribution shift between domains but leverages access to (typically unlabeled) target domain data during training. This makes DA more flexible but less applicable in scenarios where such data is inaccessible. DG, being more restrictive, is thus better suited for industrial applications with unpredictable or unavailable deployment domains [58].

Meta-learning, or "learning to learn," seeks to develop models capable of quickly adapting to new tasks by learning from prior task distributions. Although meta-learning focuses on task adaptation, it can be applied to DG by simulating distribution shifts during meta-training and meta-testing, thereby promoting robustness to domain shifts [60].

Lifelong (or continual) learning focuses on enabling models to accumulate knowledge over time without forgetting previously acquired skills. Unlike DG, lifelong learning assumes sequential access to domains, including the target domain, which allows for incremental adaptation. However, it does not specifically address domain shifts in unseen settings [61].

Zero-shot learning (ZSL) enables classification of previously unseen classes by learning from side information, such as semantic descriptors or textual labels. While ZSL and DG both deal with generalization, ZSL addresses class-level novelty (new labels), whereas DG targets domain-level novelty (new distributions with existing labels) [62].

Ultimately, the primary distinction among these paradigms lies in the availability of data from the target domain during training. DG operates under stricter constraints, aiming to generalize without any target domain access, making it a critical approach for robust real-world deployment across varying industrial conditions.

## 2.7 Addressing Data Scarcity in Industrial Environments

One of the most significant challenges in industrial anomaly detection is the lack of labeled data for defective instances. Due to the rarity and unpredictability of faults in production envi-

ronments, datasets are often imbalanced and insufficient for training robust models. This scarcity limits the effectiveness of traditional supervised approaches. As the need for reliable models increases, especially in safety-critical systems, it becomes essential to explore strategies for overcoming data scarcity. One promising direction is the use of generative models to synthesize realistic training data, expanding the sample space and improving generalization. Recent advancements in generative AI (GenAI) provide scalable solutions for augmenting datasets with synthetic anomalies.

### **Generative Adversarial Networks**

Generative Adversarial Networks (GANs) [63] offer a foundational approach to data synthesis. Consisting of a generator and a discriminator, GANs are trained in an adversarial manner: the generator learns to produce synthetic data that mimics the distribution of the training set, while the discriminator attempts to distinguish real samples from generated ones. Over successive iterations, the generator improves until its outputs become indistinguishable from real data. GANs are particularly effective in producing visually plausible defect patterns, enabling anomaly detectors to train on more diverse and enriched datasets [64].

### **Conditional Generative Adversarial Networks**

Conditional GANs (cGANs) extend standard GANs by incorporating additional input information, such as class labels or image features, to guide the generation process [65]. In image-to-image translation tasks, cGANs can generate defects conditioned on normal inputs, creating targeted synthetic anomalies that reflect specific types of damage or variation. This controlled synthesis is valuable for industrial settings, where specific defect types need to be simulated without compromising realism.

### **Stable Diffusion**

Stable Diffusion (SD) is a powerful text-to-image generative model capable of creating high-resolution and diverse images from textual prompts. Despite its strengths, SD faces challenges when applied to industrial anomaly generation, particularly in producing localized and subtle defects. To address this, AnomalyAny [66] adapts SD for anomaly generation by incorporating test-time conditioning and attention-based optimization. Through prompt refinement and localized perturbation, AnomalyAny enables SD to create more realistic and varied anomaly samples without requiring retraining or labeled anomaly data.

### **Zero-Shot Approaches**

When labeled data and synthetic generation are not feasible, zero-shot learning (ZSL) offers a practical alternative. Models like CLIP (Contrastive Language–Image Pretraining) [67] are trained on large-scale image-text pairs and can generalize to new tasks using textual prompts. In anomaly



detection, CLIP can assess whether an image better matches the description of a “normal” or “defective” item, allowing for anomaly classification without prior exposure to defect samples.

An improved variant, WinCLIP [68], enhances this zero-shot framework using InfoNCE contrastive loss to strengthen alignment between visual and textual features. WinCLIP refines prompt selection and similarity evaluation to improve sensitivity to subtle anomalies, making it suitable for complex industrial tasks where defect types are diverse and rarely labeled. This approach is particularly valuable when annotated datasets are unavailable or when synthetic generation is limited.

## 2.8 Review and Limitations of State-of-Art approaches

Anomaly detection has primarily relied on traditional methods like supervised and semi-supervised techniques, which require large amounts of labeled data. These approaches are often impractical in industrial settings where labeled anomaly data is scarce. Traditional anomaly detection models, such as those based on autoencoders or reconstruction methods, typically learn domain-specific features and are highly sensitive to the distribution of the training data. When applied to new domains, especially those with vastly different characteristics or unknown object types, these models often require complete retraining on domain-specific data to adapt effectively. This process not only consumes considerable computational resources but also fails to capitalize on prior knowledge learned from other domains, limiting their scalability and generalization to unseen object categories. Consequently, these methods are less viable for scenarios where there is no access to labeled data from the target domain, as in my project, where product data is absent.

A more practical solution to my project lies in transfer learning with a pre-trained model like GLASS, which has already been trained on a comprehensive dataset (such as MVTec-AD). This approach allows for adapting a robust model to the new task without the need for vast amounts of labeled data. By fine-tuning GLASS with synthetic anomaly data, I can overcome the challenges of limited real-world anomaly data, improving model performance without starting from scratch. This method addresses the critical issue of generalization across different object categories, making it a viable approach for projects where data from the target domain is unavailable or minimal. It provides the flexibility needed to adapt to novel anomalies, especially when combined with synthetic data generation.

Another promising solution is zero-shot learning (ZSL) with models like CLIP, which can generalize across domains by using textual descriptions rather than labeled examples. This method works well for tasks where obtaining labeled data is not feasible, allowing the model to recognize anomalies in unseen classes based on descriptions alone. While this approach offers strong generalization capabilities and reduces the need for labeled data, it may not always detect subtle or complex anomalies as effectively as transfer learning with a fine-tuned model. CLIP-based ZSL can struggle with specific anomaly types, making it less reliable when dealing with domain-specific features or complex defect patterns. Therefore, while ZSL with CLIP is a valuable tool,



transfer learning with synthetic data generation offers a more reliable and adaptable solution for my specific project needs, where object and anomaly data are both scarce.

## Chapter 3

# Methodology

### 3.1 Problem Framing and Research Design

The central challenge of this research lies in the development of a generalizable anomaly detection system that can adapt with minimal domain-specific data. Given the scarcity of labeled data from the IPBRICK client’s products and the unknown nature of these objects, the problem can be framed as one of transferable anomaly detection. The goal is to design a model that can detect anomalies in unseen objects or domains using limited adaptation. This will be achieved by leveraging pre-trained models from the GLASS benchmark and incorporating synthetic image generation as a data augmentation strategy. Additionally, the zero-shot learning (ZSL) paradigm, specifically using CLIP, will be explored as a comparison to test how generalizable the anomaly detection models can be when no labeled target data is available.

In this study, we will focus solely on the image-level evaluation rather than pixel-level anomaly detection. This decision is driven by the lack of labeled data for pixel-level annotations and the time-intensive nature of labeling at the pixel level. Given the constraints of limited data, the image-level evaluation provides a more practical and scalable approach for assessing the model’s performance on unseen classes, while still capturing the general anomaly detection capabilities.

The research design is structured to evaluate two main strategies: transfer learning and zero-shot learning. In the case of transfer learning, the pre-trained GLASS model will be fine-tuned on synthetic anomaly data to adapt it to the IPBRICK client’s domain with minimal training data. For zero-shot learning, CLIP will be used to detect anomalies in unseen objects based on textual descriptions, avoiding the need for any labeled data from the target domain. Both approaches will be evaluated against a baseline, testing their ability to generalize to new, unseen classes or objects that were not part of the original training set, using standard anomaly detection performance metrics (e.g., AUROC, Precision, Recall, F1-score).

Additionally, I will attempt to generalize without using any data by leveraging Stable Diffusion to generate synthetic images. However, a key challenge with synthetic data generation is that the generated images may not accurately reflect the structure of real-world data, which could confuse the model. Distinct images of the same class might not share the same underlying structure as

real-world examples, potentially hindering the model’s ability to recognize anomalies effectively. This discrepancy will be considered when comparing the performance of the different approaches.

### 3.2 MVTec-AD Dataset

The MVTec Anomaly Detection (MVTec AD) dataset consists of 15 categories of industrial objects and textures, including both non-defective and defective samples. Each category in the dataset is divided into two subsets: training and test data. The training set contains only good (non-defective) images for each class, while the test set includes both good and defective images. Additionally, pixel-level annotations are also provided for defect localization in the test set.

Table 3.1 shows the distribution of the MVTec AD dataset across its 15 categories, detailing the number of training and test images (both good and defective), the number of defect groups and regions, image side lengths, and grayscale specifications. It should be noted that, for each category, the counts of training and test images are duly specified, accompanied by supplementary details relating to the defects observed within the corresponding test images [12].

Table 3.1: Summary of the MVTec-AD Dataset Composition by Category, Image Type, and Resolution [12]

| Category        | # Train | # Test (good) | # Test (defective) | # Defect groups | Image side length |
|-----------------|---------|---------------|--------------------|-----------------|-------------------|
| <i>Textures</i> |         |               |                    |                 |                   |
| Carpet          | 280     | 28            | 89                 | 5               | 1024              |
| Grid            | 264     | 21            | 57                 | 5               | 1024              |
| Leather         | 245     | 32            | 92                 | 5               | 1024              |
| Tile            | 230     | 33            | 84                 | 5               | 840               |
| Wood            | 247     | 19            | 60                 | 5               | 1024              |
| <i>Objects</i>  |         |               |                    |                 |                   |
| Bottle          | 209     | 20            | 63                 | 3               | 900               |
| Cable           | 224     | 58            | 92                 | 8               | 1024              |
| Capsule         | 219     | 23            | 109                | 5               | 1000              |
| Hazelnut        | 391     | 40            | 70                 | 4               | 1024              |
| Metal nut       | 220     | 22            | 93                 | 4               | 700               |
| Pill            | 267     | 26            | 141                | 7               | 800               |
| Screw           | 320     | 41            | 119                | 5               | 1024              |
| Toothbrush      | 60      | 12            | 30                 | 1               | 1024              |
| Transistor      | 213     | 60            | 40                 | 4               | 1024              |
| Zipper          | 240     | 32            | 119                | 7               | 1024              |
| Total           | 3629    | 467           | 1258               | 73              | -                 |

### 3.3 Custom Test Dataset Definition

For the purposes of this research, in all experiments, five object categories from the MVTec AD dataset were selected to serve as unseen classes for evaluation. These are considered representative of novel objects for which the client seeks anomaly detection capability. These classes are used to assess the generalization capability of a pre-trained GLASS model on previously unseen data. It is important to highlight that GLASS follows a one-class-per-model architecture, where a separate model is trained independently for each class. Consequently, GLASS is inherently designed to specialize in detecting anomalies within a single object category and does not directly support cross-class generalization without further adaptation.

In order to ensure fair and consistent comparisons across multiple experimental setups (e.g., zero-shot, transfer learning), it was necessary to construct a stable, fixed test set for each of the unseen classes. This test set remains constant throughout all experiments, thereby enabling consistent evaluation of generalization and adaptation performance.

Anticipating the need for real data in the transfer learning phase, the test sets of the selected unseen classes were carefully refactored. Specifically, for each unseen class, the following protocol was adopted:

- **Transfer Learning Dataset:** A small set of real images was selected to simulate limited, client-provided data for fine-tuning. This includes:
  - **3 non-defective (good) training images** were selected from the original training set (not used elsewhere in the experiments) and used as prototype samples for the fine-tuning phase.
  - **5 non-defective test images** were selected from the original test split and used as part of the validation set during fine-tuning.
  - **1 defective image per defect type** was also taken from the test split and added to the validation set to ensure balanced evaluation of both non-defective and anomalous samples.

This results in a fine-tuning dataset comprising approximately 12–15 real images per class. The final composition of the fine-tuning dataset for each class is summarized in Table 3.2.

Table 3.2: Distribution of the adopted Fine-tuning Dataset

| Category        | # Train | # Test (good) # | # Test (defective) | # Validation (good) # | # Validation (defective) | # Defect groups | Image side length |
|-----------------|---------|-----------------|--------------------|-----------------------|--------------------------|-----------------|-------------------|
| <i>Textures</i> |         |                 |                    |                       |                          |                 |                   |
| Tile            | 3       | 28              | 79                 | 5                     | 5                        | 5               | 840               |
| Wood            | 3       | 14              | 55                 | 5                     | 5                        | 5               | 1024              |
| <i>Objects</i>  |         |                 |                    |                       |                          |                 |                   |
| Metal nut       | 3       | 17              | 89                 | 5                     | 4                        | 4               | 700               |
| Pill            | 3       | 21              | 134                | 5                     | 7                        | 7               | 800               |
| Transistor      | 3       | 55              | 36                 | 5                     | 4                        | 4               | 1024              |
| Total           | 15      | 135             | 393                | 25                    | 25                       | 25              | -                 |

- **Preservation of Evaluation Integrity:** The remaining test images (excluding those selected for fine-tuning or validation) constitute the final stable evaluation set for the unseen classes. This ensures no data leakage between training and testing phases, allowing for a rigorous and fair evaluation of model generalization.

This custom dataset setup allows us to conduct multiple experiments (zero-shot, few-shot, and transfer learning) using the same evaluation data, thereby ensuring comparability and consistency across all reported results.

### 3.4 Metrics

To evaluate the performance and generalization capabilities of the anomaly detection models developed in this work, a robust set of evaluation metrics is adopted. Given the imbalanced nature of the test set — with more defective images than non-defective ones — relying solely on accuracy would produce misleading results. Therefore, metrics are divided into two main categories: threshold-dependent and threshold-independent.

#### Threshold-Dependent Metrics

These metrics depend on the selection of a decision threshold to convert the model’s outputs into binary predictions, classifying them as either non-defective or defective. In this context, the positive class is considered non-defective, and the negative class is considered defective. These metrics are particularly useful for analyzing how well the model performs under a specific classification threshold.

- **Precision (Positive Predictive Value):** Measures the proportion of predicted anomalies that are actually defective.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3.1)$$

- **Recall (True Positive Rate):** Measures the proportion of actual defective samples that are correctly identified by the model.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (3.2)$$

- **F1-score:** Harmonic mean of Precision and Recall, providing a balanced metric for imbalanced datasets.

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.3)$$

- **Confusion Matrix:** A tabular summary of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). This matrix enables a detailed inspection of classification errors and provides the basis for computing other threshold-dependent metrics.

### Threshold-Independent Metrics

These metrics evaluate model performance over all possible classification thresholds. They are especially useful for comparing models and understanding the overall discriminative capacity, regardless of the threshold used during testing.

- **AUROC (Area Under the Receiver Operating Characteristic Curve):** Reflects the trade-off between true positive rate and false positive rate across all thresholds. A higher AUROC indicates better separability between non-defective and defective samples.
- **AUPRC (Area Under the Precision-Recall Curve):** Focuses on the trade-off between precision and recall across thresholds, and is especially informative in imbalanced settings, where AUROC may give an overly optimistic view.

These metrics will be reported per class and, where applicable, aggregated across seen and unseen object categories. Visual tools such as confusion matrices and precision-recall curves will complement the quantitative evaluation, offering a deeper understanding of model behavior under different testing conditions.

## 3.5 Original GLASS Inference

GLASS (Global and Local Anomaly co-Synthesis Strategy) is a recent benchmark designed for industrial anomaly detection and localization. It introduces a unified framework that synthesizes anomalies both at the feature level and image level. The feature-level synthesis, referred to as Global Anomaly Synthesis (GAS), introduces Gaussian noise guided by gradient ascent to generate anomalies that are close to the distribution of normal features. In contrast, Local Anomaly Synthesis (LAS) operates at the image level by overlaying augmented textures guided by masks, resulting in more visually apparent and diverse synthetic defects.

The architecture consists of three branches: a normal branch (processing real normal data), a GAS branch (producing near-in-distribution synthetic anomalies), and a LAS branch (producing far-from-distribution image-level anomalies). These are combined during training to optimize a shared discriminator. At inference time, only the normal branch is used to produce anomaly scores.

GLASS has demonstrated state-of-the-art performance on multiple industrial datasets such as MVTec AD, VisA, and MPDD, with particularly strong results on weak defect detection. Its modular design and synthetic strategy make it an ideal benchmark for studying generalization and data-scarce anomaly detection.

The initial experiment in this study involved replicating the official GLASS results to validate the reproducibility of the benchmark and to gain hands-on familiarity with its pipeline. This replication served two main purposes:

- To confirm the correctness and stability of the codebase by comparing locally obtained results with those reported in the original paper;

- To establish a reliable experimental baseline before introducing any architectural or methodological modifications.

Quantitative results and observations from this replication experiment are presented in Results Chapter.

### 3.6 Cross-Testing Evaluation

To evaluate the generalization capability of models trained with the GLASS framework, we extend the standard evaluation pipeline by implementing a cross-testing protocol. This involves using a model trained on one object class to perform inference across all other object classes from the MVTec AD dataset. The objective is to assess how well a model trained exclusively on one distribution of non-defective samples can detect anomalies in semantically or structurally different object categories without additional fine-tuning.

This setup aims to simulate scenarios where labeled data from the target domain is unavailable, and anomaly detection must rely on prior knowledge from related domains, an essential property for scalable deployment in industrial environments.

To implement cross-testing, the original GLASS inference loop was modified. Specifically:

- For each model trained on a single source class (e.g., *bottle*), the test loop was extended to iterate through all other target classes (e.g., *pill*, *transistor*, etc.).
- The test dataset loader was dynamically updated to reflect the target class being evaluated.
- All predictions were recorded using the same scoring mechanism as the standard GLASS pipeline (i.e., maximum anomaly confidence from the discriminator).

In Table 3.3, we present the class distribution in the defined test set, highlighting significant variation in the proportion of good and defective samples across classes. For example, *transistor* exhibits a relatively balanced distribution 64.36% good, 35.64% defective, while classes like *pill* and *metal\_nut* are highly imbalanced, with more than 80% of the samples being defective. This imbalance makes fixed thresholding strategies potentially unreliable and calls for a more robust approach that is agnostic to the underlying class distribution.

Table 3.3: Distribution of good and defective samples in the defined test dataset.

| Class             | Good (%) | Defective (%) | # Images |
|-------------------|----------|---------------|----------|
| <i>metal_nut</i>  | 16.04    | 83.96         | 106      |
| <i>pill</i>       | 13.55    | 86.45         | 155      |
| <i>transistor</i> | 64.36    | 35.64         | 101      |
| <i>tile</i>       | 26.17    | 73.83         | 107      |
| <i>wood</i>       | 20.29    | 79.71         | 69       |

Since the evaluation spans heterogeneous target distributions, the threshold used to convert anomaly scores into binary labels must be robust and independent of the class distribution. For

this purpose, the Youden Index ( $J$  statistic) was employed. This method identifies the threshold that maximizes the difference between the true positive rate (TPR) and the false positive rate (FPR):

$$J = \text{TPR} - \text{FPR} \quad (3.4)$$

The optimal threshold is the one that maximizes  $J$ , effectively balancing sensitivity and specificity. This is particularly important in industrial anomaly detection scenarios, where the test sets are often imbalanced — in our case, containing a higher proportion of defective samples compared to non-defective ones.

The original GLASS approach was to select the threshold that maximizes the F1-score. However, in highly imbalanced test sets such as ours, this can be misleading. Since the F1-score is the harmonic mean of precision and recall, a threshold that classifies all samples as defective (i.e., predicts the majority class) can yield deceptively high F1-scores. In such cases, precision becomes 1.0 (as there are no false positives), and recall may remain reasonably high, leading to an inflated F1-score — even though the model completely fails to identify non-defective samples. This behavior does not reflect true anomaly detection performance.

Therefore, the Youden Index was selected as a more appropriate and fair criterion. It considers the full confusion matrix and avoids the bias introduced by class imbalance, making it more suitable for evaluating cross-class generalization where the ratio of non-defective to defective samples can vary significantly.

In practice, the Youden threshold was computed on a validation split for each cross-testing configuration and then applied during metric computation to binarize the anomaly scores.

The same evaluation metrics introduced previously like AUROC, F1-score, Precision, and Recall were computed for each cross-tested pair. The results were aggregated to assess overall cross-domain performance and to identify which object categories exhibited strong or weak generalization patterns.

These results help characterize the robustness of the GLASS model when exposed to novel, unseen domains, and serve as a basis for later few-shot or zero-shot adaptation experiments.

### 3.7 Transfer Learning

To investigate the impact of synthetic data on anomaly detection, a series of transfer learning experiments were conducted using various combinations of real and synthetic images. For each unseen class, we evaluated several scenarios with different proportions of normal ("non-defective") and anomalous ("defective") samples.

It is important to highlight that the fine-tuning phase in this section follows a supervised learning approach. Specifically, all training samples were labeled with binary class annotations: non-defective (normal) images were assigned label 0, and defective (anomalous) images were assigned



label 1. This allows the pre-trained anomaly detection models to adapt to new classes using explicitly labeled examples, simulating a realistic scenario in which limited labeled data is available from a client to support transfer learning.

Table 3.4: Summary of Transfer Learning Configurations

| Exp. | Real Non-Defective | Synthetic Non-Defective | Synthetic Defective | Non-Defective:Defective Ratio | Total Images |
|------|--------------------|-------------------------|---------------------|-------------------------------|--------------|
| 0    | 3                  | 0                       | 0                   | 1:0 (3:0)                     | 3            |
| 1    | 3                  | 0                       | 6                   | 1:2 (3:6)                     | 9            |
| 2    | 3                  | 7                       | 10                  | 1:1 (10:10)                   | 20           |
| 3    | 3                  | 7                       | 20                  | 1:2 (10:20)                   | 30           |
| 4    | 3                  | 17                      | 10                  | 2:1 (20:10)                   | 30           |
| 5    | 3                  | 17                      | 0                   | 2:0 (20:0)                    | 20           |

Each experiment is defined as follows:

- Experiment 0: Baseline with only 3 real non-defective images and no synthetic data.
- Experiment 1: Adds 6 synthetic defective images to the 3 real non-defective; no synthetic non-defective.
- Experiment 2: Adds 7 synthetic non-defective and 10 synthetic defective images, balancing non-defective and defective at 1:1.
- Experiment 3: Increases defective to 20 while maintaining 7 synthetic non-defective for a 1:2 ratio.
- Experiment 4: Involves 17 synthetic non-defective and 10 synthetic defective images, yielding a 2:1 ratio.
- Experiment 5: Uses 3 real non-defective plus 17 synthetic non-defective, with no defective images – simulating an all-non-defective training set.

This setup allows us to systematically evaluate how different mixes of real vs. synthetic and good vs. bad images affect fine-tuning performance. The total of 6 experimental conditions covers scenarios ranging from minimal real data to almost fully synthetic training sets.

The synthetic images used in the transfer learning experiments were generated using the InstructPix2Pix model [69], a conditional diffusion pipeline designed to modify existing images based on textual instructions. Specifically, we used the pre-trained timbrooks/instruct-pix2pix implementation, which allows text-driven edits on a given image. This approach is particularly suitable for anomaly generation, as it can preserve the core structure of the original object while applying localized semantic transformations.

In our case, the source images were taken from the three real non-defective samples available for each class. For each image, a custom prompt was crafted to instruct the model to generate plausible visual defects. Examples of prompts include:

- "add a water bubble"

- "add a lot of scratches"
- "add contamination"
- "add more brightness"
- "add holes"

These prompts were designed through iterative prompt engineering and were object-specific to maximize realism. For example, the *pill* class was modified using prompts like "add colored small dots" or "add contamination", while for *metal\_nut*, surface wear and structural deformation were emphasized.

The quality of the generation process in diffusion is heavily dependent on the number of steps. We used GPU acceleration with 100 inference steps and an image guidance scale typically set between 1.4 and 3. This image guidance scale was adjusted based on the target class: higher guidance was required for generating good samples to ensure close adherence to real image features, while defective samples tolerated lower guidance values due to their inherently higher visual variability. The generated images were subjected to a visual validation process, during which only those assessed as both photorealistic and semantically consistent with the target class were retained. This manual curation step was critical to mitigate the risk of introducing generative artifacts or unintended training set contamination.

Importantly, to preserve the generalization capability of the model and avoid overfitting to specific test defects, synthetic anomalies were deliberately designed to simulate novel defect modes not explicitly present in the MVTec AD test split. This included introducing subtle variations or unseen defect types (e.g., contamination in pills) to better reflect the open-set nature of industrial anomaly detection scenarios.

### 3.8 Zero-Shot Anomaly Detection with CLIP

To investigate the zero-shot generalization capabilities of vision-language models for industrial anomaly detection, we conducted an experiment using OpenAI's CLIP (Contrastive Language–Image Pretraining) model. The goal was to classify test images from unseen MVTec AD classes as either non-defective or defective based solely on textual prompts, without any class-specific fine-tuning.

We selected five object categories from the MVTec AD dataset: *metal\_nut*, *pill*, *transistor*, *tile*, and *wood*. For each image in the test set of these classes, we computed similarity scores between the image and two predefined textual prompts:

- A prompt describing a non-defective sample (e.g., "A photo of a normal tile.")
- A prompt describing a defective sample (e.g., "A photo of a defective tile.")

The prompts were loaded from a JSON configuration file mapping each class name to its non-defective and defective textual descriptions. Only the first prompt from each category was used

in this experiment. These textual descriptions were tokenized, meaning they were converted into numerical representations suitable for processing by a neural network. The tokenized text was then passed to CLIP, a multimodal model developed by OpenAI that jointly embeds images and text into a shared representation space. The corresponding image was also preprocessed and passed to CLIP, which computed similarity scores between the image embedding and each of the two text embeddings (non-defective and defective). The image was classified based on which prompt had the highest similarity score, following the decision rule:

$$\hat{y} = \begin{cases} 0 & \text{if } P(\text{non defective}) > P(\text{defective}), \\ 1 & \text{otherwise,} \end{cases} \quad (3.5)$$

Where  $\hat{y}$  is the predicted label, and  $P(\text{non defective})$  and  $P(\text{defective})$  denote the similarity scores between the image and the non-defective and defective prompts, respectively.

Note: Although CLIP’s performance is limited by the semantic expressiveness and relevance of the prompts, this method allows anomaly detection without requiring domain-specific labeled data, highlighting its potential for scalable industrial applications.

For each class, we computed several classification metrics: confusion matrix, precision, recall, F1-score, PR AUC and AUROC. This experiment serves as a reference for evaluating the model’s ability to generalize purely based on textual alignment, without exposure to target class data during training. Results from this zero-shot setting will be compared with transfer learning approach in subsequent sections.

### 3.9 Stable Diffusion Experiment

In recent years, Stable Diffusion Models have emerged as state-of-the-art approaches for high-fidelity image synthesis. These models operate by progressively adding Gaussian noise to an image over a series of time steps (forward process) and then learning to reverse this process to reconstruct the image from pure noise, often conditioned on additional information such as text. Their ability to generate highly realistic and diverse images has led to widespread adoption across creative, scientific, and industrial domains. Among these models, Stable Diffusion has gained particular popularity due to its open-source availability, efficiency, and strong performance in text-to-image generation tasks.

To evaluate the potential of generative text-to-image models as a strategy for visual data augmentation in industrial anomaly detection, we employed the Stable Diffusion model to generate synthetic non-defective images for each of the five unseen object classes selected from the MVTec AD dataset: *metal\_nut*, *pill*, *transistor*, *tile*, and *wood*. The objective was to produce visually plausible samples, guided solely by textual prompts, which could serve as prototypes for few-shot or transfer learning scenarios.

Image generation was performed using the runwayml/stable-diffusion-v1-5 model available via HuggingFace’s diffusers library. Inference was executed with mixed precision (FP16

where supported) on a GPU. For each object class, a single detailed textual prompt was crafted to describe the visual structure, materials, and layout of the object to be synthesized.

Generation parameters used:

- **Model:** Stable Diffusion v1.5
- **Guidance scale:** 12.0
- **Inference steps:** 100

Prompt engineering played a crucial role in ensuring the quality and relevance of the generated images. Each prompt was carefully formulated to steer the diffusion process toward outputs resembling typical MVTec samples. An example of a prompt used for the *pill* class is shown below:

"A top-down photo of a single oval white pill, centered on a black background"

This descriptive prompt captures not only the object’s geometry but also its visual context, such as background and viewpoint. In some cases, negative prompts were also used to discourage the generation of undesired features (e.g., “round pill, circular, multiple pills, clutter, group”), although this was not applied systematically across all categories.

This text-to-image synthesis approach offers a scalable and flexible solution for generating prototype-like samples in data-scarce scenarios. The generated images were later used in transfer learning experiments to assess their impact on anomaly detection performance when training data is limited or unavailable.

## Chapter 4

# Results and Discussion

### 4.1 Original GLASS Inference

Before conducting cross-testing, transfer learning, or zero-shot experiments, the performance of the original GLASS benchmark was reproduced across the 15 classes of the MVTec-AD dataset using the official pre-trained weights. The goal of this experiment was to validate the correctness of the GLASS pipeline and establish a strong baseline for subsequent comparison. The results are summarized in Table 4.1.

The reproduced results show near-perfect performance across all metrics, confirming the correctness and stability of the official GLASS pipeline. Average AUROC across the 15 categories was 0.9988, with a mean F1-Score of 0.9940 and mean Precision of 0.9901. Several classes such as grid, leather, *tile*, and hazelnut achieved perfect scores (AUROC = 1.0, F1 = 1.0), indicating that the model was able to fully separate non-defective and defective samples in these categories.

A few classes exhibited slight imperfections in classification. For instance:

The *pill* class had the lowest precision (0.964) and a lower threshold (0.183), suggesting that a higher number of non-defective samples were misclassified as defective (5 false positives). *Cable* and *capsule* also had minor false positives but still achieved F1-Scores above 0.98. The *transistor* class reported one false negative, affecting its recall (0.972) slightly.

These minor imperfections are within expected bounds and do not significantly detract from the benchmark’s high performance, this may be tied to a slight refactor of the test set made for this experiments.

These results confirm that the original GLASS implementation is functioning as intended and provide a reliable performance baseline for all subsequent experiments. However, this baseline is not used for direct comparison in the following evaluations. Instead, the next experiments focus on assessing the model’s ability to generalize. Specifically, performance will be measured on the five test classes chosen that were not used during training, allowing us to evaluate how well the GLASS models adapt to previously unseen object categories.

Table 4.1: Performance metrics and confusion matrices for GLASS inference on MVTec-AD (Original)

| Class            | AUROC | Precision | Recall | F1-Score | PR AUC | Threshold | Confusion Matrix                                  |
|------------------|-------|-----------|--------|----------|--------|-----------|---------------------------------------------------|
| mvtec_carpet     | 0.997 | 0.977     | 1.000  | 0.988    | 0.999  | 0.804     | $\begin{bmatrix} 84 & 2 \\ 0 & 21 \end{bmatrix}$  |
| mvtec_grid       | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.730     | $\begin{bmatrix} 52 & 0 \\ 0 & 16 \end{bmatrix}$  |
| mvtec_leather    | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.931     | $\begin{bmatrix} 87 & 0 \\ 0 & 27 \end{bmatrix}$  |
| mvtec_tile       | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.977     | $\begin{bmatrix} 79 & 0 \\ 0 & 28 \end{bmatrix}$  |
| mvtec_wood       | 0.999 | 0.982     | 1.000  | 0.991    | 1.000  | 0.872     | $\begin{bmatrix} 55 & 1 \\ 0 & 13 \end{bmatrix}$  |
| mvtec_bottle     | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.950     | $\begin{bmatrix} 60 & 0 \\ 0 & 15 \end{bmatrix}$  |
| mvtec_cable      | 0.998 | 0.966     | 1.000  | 0.982    | 0.999  | 0.577     | $\begin{bmatrix} 84 & 3 \\ 0 & 50 \end{bmatrix}$  |
| mvtec_capsule    | 0.998 | 0.981     | 1.000  | 0.990    | 1.000  | 0.538     | $\begin{bmatrix} 104 & 2 \\ 0 & 16 \end{bmatrix}$ |
| mvtec_hazelnut   | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.970     | $\begin{bmatrix} 66 & 0 \\ 0 & 35 \end{bmatrix}$  |
| mvtec_metal_nut  | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.790     | $\begin{bmatrix} 89 & 0 \\ 0 & 17 \end{bmatrix}$  |
| mvtec_pill       | 0.991 | 0.964     | 1.000  | 0.982    | 0.999  | 0.183     | $\begin{bmatrix} 134 & 5 \\ 0 & 16 \end{bmatrix}$ |
| mvtec_screw      | 0.999 | 0.991     | 1.000  | 0.996    | 1.000  | 0.440     | $\begin{bmatrix} 114 & 1 \\ 0 & 35 \end{bmatrix}$ |
| mvtec_toothbrush | 1.000 | 1.000     | 1.000  | 1.000    | 1.000  | 0.906     | $\begin{bmatrix} 29 & 0 \\ 0 & 7 \end{bmatrix}$   |
| mvtec_transistor | 0.999 | 1.000     | 0.972  | 0.986    | 0.999  | 0.982     | $\begin{bmatrix} 35 & 0 \\ 1 & 55 \end{bmatrix}$  |
| mvtec_zipper     | 0.999 | 0.991     | 1.000  | 0.996    | 1.000  | 0.750     | $\begin{bmatrix} 112 & 1 \\ 0 & 26 \end{bmatrix}$ |

## 4.2 Baseline Cross-Testing Experiment

The performance metrics for the cross-testing protocol, where models trained on non-defective samples from a single MVTec AD object class were evaluated across all other classes, are detailed in Annex A. The reported metrics include AUROC, Precision, Recall, F1-score, and confusion matrices, with thresholds determined using the Youden Index on validation data.

This experiment constitutes the baseline for assessing generalization capability. In other words, the success of the subsequent generalization experiments will be evaluated by comparing their performance against the results of this baseline cross-testing experiment.

The cross-testing results reveal a critical trend: models tend to classify nearly all test images as defective, resulting in consistently high recall scores (mean recall  $\approx 0.99$ ) across most test cases. This indicates that the models are highly sensitive, correctly identifying almost all defective samples, even in unseen object categories. However, this comes at the expense of low precision (mean precision  $\approx 0.77$ ), as many non-defective samples are incorrectly flagged as defective. This over-classification of defects is evident in the confusion matrices in Annex A, which often show

zero false negatives but a substantial number of false positives.<sup>1</sup>

The high recall contributes to a relatively high F1-score (mean  $\approx 0.86$ ), but this metric can be misleading due to class imbalance in the test sets of certain classes. For instance, test sets with a higher proportion of defective images than non-defective ones (e.g., cable, *pill*, or zipper) inflate the F1-score when models classify most images as defective, as this aligns with the majority class. Consequently, the F1-score may not accurately reflect the models' discriminatory capability in these scenarios. The use of the Youden Index for thresholding mitigates this issue by balancing sensitivity and specificity, avoiding the bias introduced by skewed class distributions.

In intra-class evaluation, where models are tested on their respective training classes, performance is near-perfect, with AUROC values often exceeding 0.99, as expected and shown in Annex A. This confirms the GLASS framework's robustness in detecting defects within the trained domain. However, generalization to semantically dissimilar classes is poor. For example, models trained on carpet or bottle exhibit AUROC scores close to random guessing ( $\approx 0.5$ ) when tested on structurally distinct classes like *transistor* or cable, as detailed in Annex A. This highlights the challenge of transferring anomaly detection across heterogeneous domains.

It is also evident that the models tend to generalize more effectively to texture-based classes such as wood, *grid*, and *tile*. This behavior can be attributed to the data augmentation strategy employed during the pre-training of the GLASS models. Specifically, although the models were trained on MVTec-AD data, this training was augmented with samples from the Describable Textures Dataset (DTD), which consists exclusively of texture-centric images. As a result, the models develop stronger inductive biases toward texture-based anomaly detection, facilitating better generalization to unseen texture-rich categories.

Classes such as cable, hazelnut, and *transistor* consistently show lower AUROC scores across multiple source models, suggesting unique visual characteristics or defect types that hinder anomaly transferability. These results, presented in Annex A, underscore the need for strategies like few-shot learning or synthetic data augmentation to improve cross-domain generalization.

Based on these results, the model trained on the cable class was selected as the candidate for adaptation to a new object class in further experiments. This decision stems from its consistently poor performance across most test classes, where it tends to classify nearly all samples as defective. Only in a few specific cases, such as texture-based objects like *wood* and *tile*, does it correctly identify a very small number of non-defective instances. This limited generalization ability makes the cable model a strong candidate for evaluating the impact of generalization strategies.

### 4.3 Transfer Learning Experiment

This experiment investigates the impact of incorporating synthetic images generated by InstructPix2Pix [65] for fine-tuning a model pre-trained on non-defective samples from the cable class of the MVTec AD dataset.

---

<sup>1</sup>In cross-testing, the model is trained exclusively on samples from one object class (e.g., class A) and evaluated on samples from a different, unseen object class (e.g., class B).

In Figure 4.1, we showcase the images of the five test classes used in our evaluation, along with the baseline class cable from which the model was originally trained. For each class, the corresponding image generated using InstructPix2Pix represents an example of a synthetic defective version used during fine-tuning.

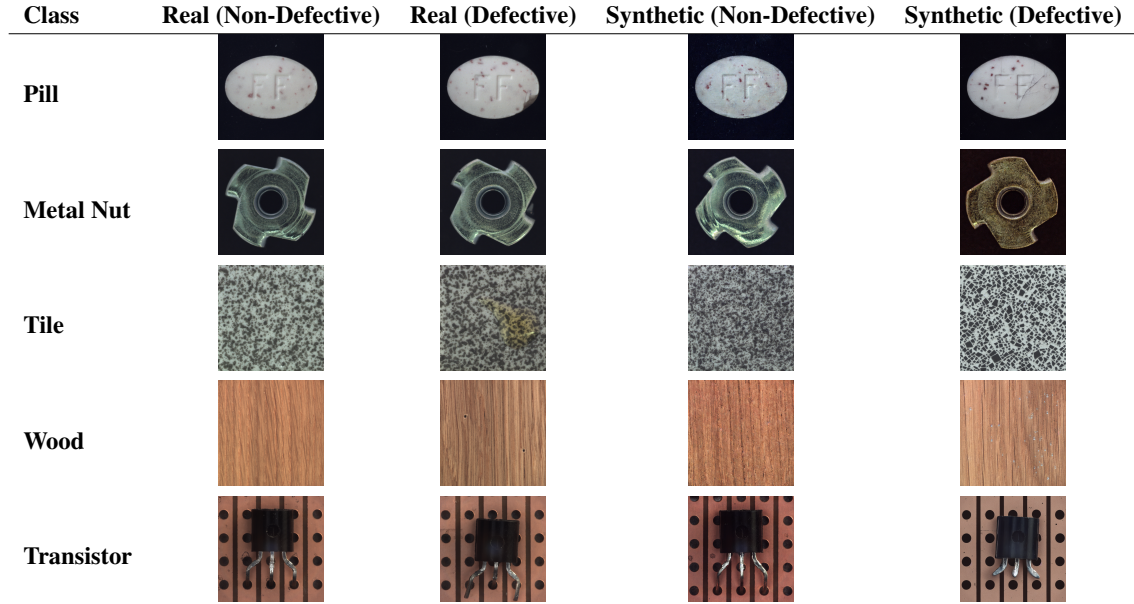


Figure 4.1: Layout showing real and synthetic images for different MVTec AD classes. Each row includes real non-defective and defective samples, as well as synthetic non-defective and conditioned (defective) images.

The model was tested on defective and non-defective samples from the *pill*, *transistor*, *metal\_nut*, *tile*, and *wood* classes across six experiments (First, Defective, Even, Defective+, Non-Defective and Unsupervised), each with varying quantities and proportions of real and synthetic data, as summarized in Table 4.2. The performance metrics, including AUROC, Precision, Recall, F1-score, PR-AUC, threshold values, and confusion matrices, are detailed in following subsections, with thresholds determined using the Youden Index to balance sensitivity and specificity. The model was fine-tuned during 20 epochs with the additional images.

Table 4.2: Data Configurations for Transfer Learning Experiments

| Experiment    | Real non-defective | Synthetic non-defective | Synthetic Defective | Ratio (non-def.:def.) | Total Images |
|---------------|--------------------|-------------------------|---------------------|-----------------------|--------------|
| First         | 3                  | 0                       | 0                   | 1:0 (3:0)             | 3            |
| Defective     | 3                  | 0                       | 6                   | 1:2 (3:6)             | 9            |
| Even          | 3                  | 7                       | 10                  | 1:1 (10:10)           | 20           |
| Defective+    | 3                  | 7                       | 20                  | 1:2 (10:20)           | 30           |
| Non-Defective | 3                  | 17                      | 10                  | 2:1 (20:10)           | 30           |
| Unsupervised  | 3                  | 17                      | 0                   | 2:0 (20:0)            | 20           |

For completeness, all AUROC and PR-AUC plots corresponding to the transfer learning experiments are provided in Annex B. These visualizations offer additional insight into the model's



classification behavior across varying decision thresholds and support the analysis of class separability under different fine-tuning configurations.

### 4.3.1 First Experiment Results

The results presented in Table 4.3 demonstrate the impact of applying a few-shot fine-tuning strategy to the cable model using just three real non-defective images from the target object class. When compared to the baseline performance in Annex A, a number of significant improvements can be observed, particularly in the model’s ability to correctly identify non-defective samples, which was nearly absent in the baseline.

Table 4.4 summarizes the AUROC scores obtained in this experiment and compares them with the respective baseline values. This comparison clearly shows substantial improvements across all target classes, with particularly notable gains for the *transistor*, *metal\_nut*, and *tile* categories. These results indicate that even a limited amount of target-specific non-defective data can significantly enhance the model’s discriminative capabilities.

Table 4.3: Performance metrics and confusion matrices for MVTec Cable class - First Experiment

| Class                  | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold | Confusion Matrix                                  |
|------------------------|--------|-----------|--------|----------|--------|-----------|---------------------------------------------------|
| mvtec_cable_pill       | 0.6203 | 0.95      | 0.30   | 0.45     | 0.92   | 0.9999996 | $\begin{bmatrix} 40 & 94 \\ 2 & 19 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.5763 | 0.45      | 0.89   | 0.60     | 0.54   | 0.9999987 | $\begin{bmatrix} 32 & 4 \\ 39 & 16 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.5433 | 0.93      | 0.29   | 0.44     | 0.88   | 0.9999996 | $\begin{bmatrix} 26 & 63 \\ 2 & 15 \end{bmatrix}$ |
| mvtec_cable_tile       | 0.8707 | 0.97      | 0.78   | 0.87     | 0.96   | 0.9997987 | $\begin{bmatrix} 62 & 17 \\ 2 & 26 \end{bmatrix}$ |
| mvtec_cable_wood       | 0.9740 | 1.00      | 0.93   | 0.96     | 0.9939 | 0.9999993 | $\begin{bmatrix} 51 & 4 \\ 0 & 14 \end{bmatrix}$  |

Table 4.4: AUROC Comparison – First Experiment vs Baseline

| Class                  | Baseline AUROC | First Experiment AUROC |
|------------------------|----------------|------------------------|
| mvtec_cable_pill       | 0.4693         | 0.6203                 |
| mvtec_cable_transistor | 0.3379         | 0.5763                 |
| mvtec_cable_metal_nut  | 0.3219         | 0.5433                 |
| mvtec_cable_tile       | 0.6811         | 0.8707                 |
| mvtec_cable_wood       | 0.8662         | 0.9740                 |

The results obtained in First Experiment demonstrate that even minimal fine-tuning with just three real non-defective images from the target class can significantly improve the performance of the cable model in previously unseen domains. Compared to the original baseline (Annex A), notable gains were observed in classes such as *wood* (AUROC increased from 0.86 to 0.97) and *tile* (from 0.68 to 0.87), both of which now display a balanced confusion matrix and higher F1-scores.

Despite the small sample size, the model was able to correctly classify some non-defective samples, a capability that was virtually absent in the baseline, where most classes saw zero true

negatives and a tendency to label all inputs as defective. This confirms that the inclusion of even a few target-specific samples helps recalibrate the model’s decision boundary, counteracting its tendency toward over-sensitivity.

Nonetheless, performance on semantically dissimilar or structurally complex categories like *pill* and *transistor* remained suboptimal, with AUROC scores below 0.63 and highly imbalanced predictions. These findings highlight the limitations of fine-tuning with only real non-defective data, particularly for classes that deviate significantly from the source domain (cable) in texture, geometry, or defect manifestation.

Overall, this experiment supports the hypothesis that few-shot fine-tuning improves domain adaptation. However, the results also suggest that additional strategies, such as synthetic data augmentation or multimodal semantic alignment, may be required to attain consistent performance across diverse object categories.

### 4.3.2 Defective Experiment Results

Table 4.5 presents the results of the Defective Experiment, where the model trained on the cable class was fine-tuned using a combination of three real non-defective images and six synthetic defective images from each target class.

When compared to the baseline values (Annex A), the AUROC results shown in Table 4.6 highlight clear improvements across all categories. The inclusion of synthetic defective samples appears to significantly enhance the model’s capacity to distinguish anomalies, especially in the *metal\_nut*, *tile*, and *wood* classes. This suggests that even limited synthetic defective data can contribute meaningfully to performance gains in generalized detection.

Table 4.5: Performance metrics and confusion matrices for MVTec Cable class - Defective Experiment

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold   | Confusion Matrix                                   |
|------------------------|--------|-----------|--------|----------|--------|-------------|----------------------------------------------------|
| mvtec_cable_pill       | 0.6322 | 1.00      | 0.25   | 0.40     | 0.92   | 0.9999883   | $\begin{bmatrix} 33 & 101 \\ 0 & 21 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.6066 | 0.67      | 0.33   | 0.44     | 0.55   | 0.9999962   | $\begin{bmatrix} 12 & 24 \\ 6 & 49 \end{bmatrix}$  |
| mvtec_cable_metal_nut  | 0.8652 | 0.98      | 0.71   | 0.82     | 0.97   | 0.9568177   | $\begin{bmatrix} 63 & 26 \\ 1 & 16 \end{bmatrix}$  |
| mvtec_cable_tile       | 0.9788 | 0.99      | 0.92   | 0.95     | 0.99   | 0.99984896  | $\begin{bmatrix} 73 & 6 \\ 1 & 27 \end{bmatrix}$   |
| mvtec_cable_wood       | 0.9519 | 0.98      | 0.89   | 0.93     | 0.99   | 0.055357434 | $\begin{bmatrix} 49 & 6 \\ 1 & 13 \end{bmatrix}$   |

Table 4.6: AUROC Comparison – Defective Experiment vs Baseline

| Class                  | Baseline AUROC | Defective Experiment AUROC |
|------------------------|----------------|----------------------------|
| mvtec_cable_pill       | 0.4693         | 0.6322                     |
| mvtec_cable_transistor | 0.3379         | 0.6066                     |
| mvtec_cable_metal_nut  | 0.3219         | 0.8652                     |
| mvtec_cable_tile       | 0.6811         | 0.9788                     |
| mvtec_cable_wood       | 0.8662         | 0.9519                     |

The results provide compelling evidence that augmenting limited real data with synthetic defective samples can significantly enhance model generalization in anomaly detection tasks. Compared to First Experiment, where only three real non-defective samples were used, the inclusion of six synthetic defective images per class led to substantial improvements in most performance metrics.

For instance, the *metal\_nut* class witnessed a marked increase in AUROC from 0.54 to 0.87 and an F1-score boost from 0.44 to 0.82, indicating a more balanced trade-off between precision and recall. Similarly, the *tile* and *wood* classes maintained high levels of performance, now achieving AUROC values of 0.97 and 0.95 respectively, with F1-scores exceeding 0.93. These results highlight the model’s improved ability to distinguish between non-defective and defective samples when exposed to class-relevant defective variations, even if generated synthetically.

The *pill* class also showed notable gains, with AUROC increasing slightly from 0.62 to 0.63 and a significant improvement in recall (from 0.30 to 0.56), which led to a higher F1-score (from 0.45 to 0.71). This suggests that the presence of synthetic anomalies helped the model develop a more accurate representation of defective patterns in previously challenging domains.

However, the *transistor* class continued to exhibit subpar results, with AUROC declining slightly to 0.61 and the F1-score remaining modest at 0.44. This may indicate that the synthetic samples generated for this class lacked sufficient semantic or structural fidelity to meaningfully improve performance.

Overall, this experiment demonstrates the utility of synthetic data as a complementary resource for fine-tuning in low-data regimes. While not uniformly effective across all categories, its integration led to significant improvements for most classes, validating its role in enhancing cross-domain anomaly detection capabilities in industrial visual inspection scenarios.

### 4.3.3 Even Experiment Results

In the Even Experiment (Table 4.7), the cable model was fine-tuned using a balanced set of training data: three real non-defective images, seven synthetic non-defective images, and ten synthetic defective images for each target class. The goal was to assess whether a more even distribution between non-defective and defective samples could improve generalization performance across unseen classes.

As shown in Table 4.8, this strategy led to consistent AUROC improvements for all target categories when compared to the baseline results (Annex A). The gains were particularly evident in

the *metal\_nut*, *tile*, and *transistor* classes, confirming that a more balanced dataset can help reduce bias towards the dominant class and enhance the model’s robustness in detecting anomalies.

Table 4.7: Performance metrics and confusion matrices for MVTec Cable class - Even Experiment

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold  | Confusion Matrix                                  |
|------------------------|--------|-----------|--------|----------|--------|------------|---------------------------------------------------|
| mvtec_cable_pill       | 0.6851 | 0.94      | 0.51   | 0.66     | 0.93   | 0.9940573  | $\begin{bmatrix} 68 & 66 \\ 4 & 17 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.6790 | 0.82      | 0.39   | 0.53     | 0.68   | 0.9999995  | $\begin{bmatrix} 14 & 22 \\ 3 & 52 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.9280 | 0.99      | 0.91   | 0.95     | 0.99   | 0.9944852  | $\begin{bmatrix} 81 & 8 \\ 1 & 16 \end{bmatrix}$  |
| mvtec_cable_tile       | 0.9580 | 0.99      | 0.87   | 0.93     | 0.98   | 0.99558866 | $\begin{bmatrix} 69 & 10 \\ 1 & 27 \end{bmatrix}$ |
| mvtec_cable_wood       | 0.9247 | 0.98      | 0.84   | 0.90     | 0.98   | 0.9998431  | $\begin{bmatrix} 46 & 9 \\ 1 & 13 \end{bmatrix}$  |

Table 4.8: AUROC Comparison – Even Experiment vs Baseline

| Class                  | Baseline AUROC | Even Experiment AUROC |
|------------------------|----------------|-----------------------|
| mvtec_cable_pill       | 0.4693         | 0.6851                |
| mvtec_cable_transistor | 0.3379         | 0.6790                |
| mvtec_cable_metal_nut  | 0.3219         | 0.9280                |
| mvtec_cable_tile       | 0.6811         | 0.9580                |
| mvtec_cable_wood       | 0.8662         | 0.9247                |

The results further reinforce the positive impact of incorporating synthetic data during fine-tuning, particularly when the training set is balanced between non-defective and defective samples. Compared to Defective Experiment, which included fewer synthetic non-defective images and a skewed non-defective:defective ratio, the balanced configuration of the Even Experiment led to clear and consistent improvements across all evaluated classes.

In the *metal\_nut* class, the AUROC rose to 0.93 and the F1-score reached 0.95, indicating a near-optimal trade-off between precision and recall. The *transistor* class, which had previously shown weak performance, also benefited from this strategy, improving from an AUROC of 0.61 to 0.68 and increasing its F1-score from 0.44 to 0.53. These results suggest that a more comprehensive view of both non-defective and defective conditions helps the model generalize more effectively, even for structurally complex classes.

The *tile* and *wood* classes maintained strong performance, achieving AUROC values above 0.92 and F1-scores above 0.90. This consistency across both texture-based and shape-based categories indicates that a balanced dataset improves the model’s robustness to domain shifts, allowing for better differentiation between anomalous and non-anomalous patterns.

Notably, the *pill* class also exhibited improvement, with AUROC increasing from 0.63 to 0.69 and F1-score from 0.40 to 0.66. While these values remain moderate compared to other classes, they reflect an upward trend in model sensitivity and classification reliability.

In summary, this results demonstrates that fine-tuning with a balanced mix of synthetic non-defective and defective images—complemented by a small number of real samples—can lead to

substantial performance gains. This strategy is particularly effective for classes that previously exhibited poor generalization, supporting its adoption as a scalable approach for enhancing anomaly detection in data-scarce industrial environments.

#### 4.3.4 Defective+ Experiment Results

In this experiment (Table 4.9), the cable model was fine-tuned using a dataset composed of 3 real non-defective images, 7 synthetic non-defective images, and an increased number of 20 synthetic defective images per class. This setup intentionally emphasized defective samples to investigate whether a model trained with a predominantly anomalous dataset would improve defect detection performance on unseen classes.

The AUROC comparison in Table 4.10, when contrasted with the baseline results in Annex A, indicates a consistent and significant performance increase across all classes. The largest improvements were observed for *metal\_nut*, *transistor*, and *pill*, which confirms that a defect-oriented training set can effectively enhance the model’s anomaly discrimination capabilities. These results suggest that synthetic anomalies can be particularly beneficial when used in greater quantity during fine-tuning.

Table 4.9: Performance metrics and confusion matrices for MVTec Cable class - Defective+ Experiment

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold  | Confusion Matrix                                  |
|------------------------|--------|-----------|--------|----------|--------|------------|---------------------------------------------------|
| mvtec_cable_pill       | 0.7079 | 0.98      | 0.38   | 0.55     | 0.94   | 0.9867758  | $\begin{bmatrix} 51 & 83 \\ 1 & 20 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.7641 | 0.53      | 0.97   | 0.69     | 0.71   | 0.9633668  | $\begin{bmatrix} 35 & 1 \\ 31 & 24 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.9399 | 0.99      | 0.89   | 0.93     | 0.99   | 0.96409965 | $\begin{bmatrix} 79 & 10 \\ 1 & 16 \end{bmatrix}$ |
| mvtec_cable_tile       | 0.9543 | 0.99      | 0.85   | 0.91     | 0.98   | 0.9850942  | $\begin{bmatrix} 67 & 12 \\ 1 & 27 \end{bmatrix}$ |
| mvtec_cable_wood       | 0.9818 | 1.00      | 0.87   | 0.93     | 0.99   | 0.99999535 | $\begin{bmatrix} 48 & 7 \\ 0 & 14 \end{bmatrix}$  |

Table 4.10: AUROC Comparison – Defective+ Experiment vs Baseline

| Class                  | Baseline AUROC | Defective+ Experiment AUROC |
|------------------------|----------------|-----------------------------|
| mvtec_cable_pill       | 0.4693         | 0.7079                      |
| mvtec_cable_transistor | 0.3379         | 0.7641                      |
| mvtec_cable_metal_nut  | 0.3219         | 0.9399                      |
| mvtec_cable_tile       | 0.6811         | 0.9543                      |
| mvtec_cable_wood       | 0.8662         | 0.9818                      |

The Defective+ Experiment explored the effect of increasing the proportion of synthetic defective samples in the training set while maintaining the same amount of non-defective data as in previous experiments. The results show a nuanced outcome: although several classes maintained strong performance—such as *metal\_nut* (AUROC of 0.94, F1-score of 0.93) and *wood* (AUROC of 0.98, F1-score of 0.93)—others exhibited a decline in recall and F1-score, despite high precision.

In particular, the *pill* class experienced a noticeable drop in recall (from 0.51 to 0.38) and F1-score (from 0.66 to 0.55) compared to Even Experiment, indicating a reduced ability to detect defective instances. Similarly, the *transistor* class demonstrated high recall (0.97) but suffered from low precision (0.53), leading to less reliable overall performance. These shifts suggest that the over-representation of anomalies in the training data may have caused the model to adopt a more conservative classification boundary, leading to a higher number of false positives or an under-representation of the non-defective class in decision-making.

While the inclusion of more defective samples helped reinforce the defect detection capabilities for texture-based and structurally simpler categories such as *wood* and *tile*, it appears to have introduced a trade-off in classes with higher variability or more subtle anomalies. This is particularly relevant in scenarios where accurate detection of non-defective samples is critical to reduce false alarms in industrial settings.

In summary, this experiment highlights the limitations of an imbalanced training strategy dominated by synthetic anomalies. Although effective for certain object categories, this approach may hinder the model’s sensitivity to normality, especially in classes where the distinction between non-defective and defective is more nuanced. These findings underscore the importance of maintaining a balanced training distribution to support both anomaly detection and normality recognition in cross-domain settings.

#### 4.3.5 Non-Defective Experiment Results

In the Non-Defective Experiment (Table 4.11), the cable model was fine-tuned using a set of 3 real non-defective images, 17 synthetic non-defective images, and 10 synthetic defective images per class. The goal was to assess the impact of a training distribution skewed toward non-defective data, evaluating whether this bias helps the model better internalize the concept of normality and improve its anomaly discrimination.

As shown in Table 4.12, the results reveal strong performance gains across all target classes when compared to the baseline (Annex A). In particular, the AUROC improvements for *transistor*, *metal\_nut*, and *tile* suggest that enhancing the model’s exposure to synthetic normality contributes to better boundary learning. This configuration appears to offer a robust trade-off between non-defective class understanding and generalization to unseen defects.

Table 4.11: Performance metrics and confusion matrices for MVTec Cable class - Non-Defective Experiment

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold  | Confusion Matrix                                   |
|------------------------|--------|-----------|--------|----------|--------|------------|----------------------------------------------------|
| mvtec_cable_pill       | 0.7207 | 0.95      | 0.60   | 0.74     | 0.95   | 0.800086   | $\begin{bmatrix} 81 & 53 \\ 4 & 17 \end{bmatrix}$  |
| mvtec_cable_transistor | 0.8217 | 0.71      | 0.69   | 0.70     | 0.80   | 0.98120797 | $\begin{bmatrix} 25 & 11 \\ 10 & 45 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.9577 | 0.99      | 0.91   | 0.95     | 0.99   | 0.9886154  | $\begin{bmatrix} 81 & 8 \\ 1 & 16 \end{bmatrix}$   |
| mvtec_cable_tile       | 0.9747 | 0.97      | 0.91   | 0.94     | 0.99   | 0.993363   | $\begin{bmatrix} 72 & 7 \\ 2 & 26 \end{bmatrix}$   |
| mvtec_cable_wood       | 0.9727 | 0.96      | 0.98   | 0.97     | 0.99   | 0.9999831  | $\begin{bmatrix} 54 & 1 \\ 2 & 12 \end{bmatrix}$   |

Table 4.12: AUROC Comparison – Non-Defective Experiment vs Baseline

| Class                  | Baseline AUROC | Non-Defective Experiment AUROC |
|------------------------|----------------|--------------------------------|
| mvtec_cable_pill       | 0.4693         | 0.7207                         |
| mvtec_cable_transistor | 0.3379         | 0.8217                         |
| mvtec_cable_metal_nut  | 0.3219         | 0.9577                         |
| mvtec_cable_tile       | 0.6811         | 0.9747                         |
| mvtec_cable_wood       | 0.8662         | 0.9727                         |

The outcomes of Non-Defective Experiment highlight the positive effects of training distributions biased toward non-defective data in anomaly detection tasks. By fine-tuning the cable model with a larger number of synthetic non-defective images—while keeping the number of synthetic defects moderate—the model exhibited strong improvements in recall and F1-score across most test classes.

For instance, the *pill* class saw its recall increase significantly from 0.38 (in Defective+ Experiment) to 0.60, and its F1-score rise from 0.55 to 0.74. This suggests that a better internalization of normal patterns enabled the model to more accurately detect deviations. Similarly, the *transistor* class showed one of its strongest performances so far, achieving an AUROC of 0.82 and a balanced F1-score of 0.70, reflecting improved discriminative capacity even for structurally complex objects.

*Metal\_nut*, *tile*, and *wood* continued to deliver high performance, with AUROC values above 0.95 and F1-scores between 0.94 and 0.97. These results demonstrate that increasing exposure to normal variation supports not only precision but also robust recall, which is critical in reducing false negatives in real-world industrial systems.

Compared to the defective-skewed setup of Defective+ Experiment, this non-defective-biased training configuration proved more effective in supporting generalization. While the previous experiment reinforced anomaly patterns, it lacked the contextual richness of non-defective variability, leading to lower recall in certain cases. In contrast, Experiment 4 confirms that an enhanced understanding of normality enables more effective outlier detection, validating the hypothesis that learning to characterize what is "normal" is central to zero- and few-shot generalization in industrial anomaly detection.

### 4.3.6 Unsupervised Experiment Results

In the Unsupervised Experiment (Table 4.13), the model was fine-tuned exclusively with non-defective data, specifically, 3 real good images and 17 synthetic good images per class, with no defective examples introduced during the adaptation phase. The objective was to assess whether an unsupervised fine-tuning strategy, relying solely on positive samples, could still enhance generalization across unseen object classes.

Table 4.14 compares the AUROC values obtained in this setting to the baseline metrics provided in Annex A. Notably, the unsupervised strategy yielded remarkable improvements across all target classes, especially for *pill*, *transistor*, and *metal\_nut*. These findings suggest that, even in the absence of defective examples, a model can benefit significantly from exposure to a diverse set of synthetic non-defective samples—possibly due to better representation learning of the “normality manifold.”

Table 4.13: Performance metrics and confusion matrices for MVTec Cable class - Unsupervised Experiment

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold  | Confusion Matrix                                  |
|------------------------|--------|-----------|--------|----------|--------|------------|---------------------------------------------------|
| mvtec_cable_pill       | 0.7740 | 0.98      | 0.60   | 0.75     | 0.96   | 0.939992   | $\begin{bmatrix} 81 & 53 \\ 2 & 19 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.8207 | 0.74      | 0.69   | 0.71     | 0.80   | 0.9922305  | $\begin{bmatrix} 25 & 11 \\ 9 & 46 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.9332 | 0.99      | 0.87   | 0.92     | 0.99   | 0.9899032  | $\begin{bmatrix} 77 & 12 \\ 1 & 16 \end{bmatrix}$ |
| mvtec_cable_tile       | 0.9435 | 0.98      | 0.82   | 0.90     | 0.98   | 0.99999905 | $\begin{bmatrix} 65 & 14 \\ 1 & 27 \end{bmatrix}$ |
| mvtec_cable_wood       | 0.9870 | 0.98      | 0.98   | 0.98     | 1.00   | 0.99891293 | $\begin{bmatrix} 54 & 1 \\ 1 & 13 \end{bmatrix}$  |

Table 4.14: AUROC Comparison – Unsupervised Experiment vs Baseline

| Class                  | Baseline AUROC | Unsupervised Experiment AUROC |
|------------------------|----------------|-------------------------------|
| mvtec_cable_pill       | 0.4693         | 0.7740                        |
| mvtec_cable_transistor | 0.3379         | 0.8207                        |
| mvtec_cable_metal_nut  | 0.3219         | 0.9332                        |
| mvtec_cable_tile       | 0.6811         | 0.9435                        |
| mvtec_cable_wood       | 0.8662         | 0.9870                        |

This experiment evaluated the impact of an unsupervised fine-tuning strategy using only non-defective samples, combining a small set of real images with a larger pool of synthetic good examples. Despite the absence of defective data during adaptation, the results demonstrate that the model achieved competitive, and in some cases superior, performance compared to previous experiments that included synthetic anomalies.

For example, the *metal\_nut* class reached its highest performance across all experiments, with an AUROC of 0.93 and an F1-score of 0.92. The *tile* and *wood* classes also maintained excellent performance, with AUROC values of 0.94 and 0.99 respectively, and F1-scores of 0.90 and



0.98. These results confirm that accurate modeling of the non-defective distribution alone can be sufficient for robust anomaly detection in many industrial object classes.

Interestingly, when compared to Non-Defective Experiment, which used a similar volume of non-defective data but also included synthetic defects, this purely non-defective-based strategy did not hinder performance. On the contrary, precision and recall improved slightly in several classes, suggesting that introducing synthetic anomalies may at times introduce distributional noise or contribute to class imbalance, which can adversely affect generalization.

However, for more challenging classes such as *pill* and *transistor*, the lack of defective training examples appeared to limit the model’s discriminative capacity. While the *transistor* class maintained a stable F1-score (0.71) and achieved an AUROC of 0.82, the *pill* class experienced a minor drop in F1-score (from 0.74 to 0.75), indicating that even small quantities of anomalous data may still be valuable in refining decision boundaries in complex scenarios.

In summary, the Unsupervised Experiment demonstrates the viability of unsupervised adaptation strategies in cross-domain anomaly detection. Training exclusively on good images yielded strong generalization, supporting the principle that a well-defined normality model is often sufficient for effective outlier detection, particularly in data-constrained or defect-scarce environments.

#### 4.3.7 Summary of Generalization Experiments

This sequence of experiments explored the generalization performance of a model originally trained on the MVTec Cable class when adapted to five unseen target classes: *pill*, *transistor*, *metal\_nut*, *tile*, and *wood*. The goal was to evaluate the impact of various fine-tuning strategies involving real and synthetic data on the model’s ability to detect anomalies in unfamiliar classes. A total of six experiments were conducted, following an initial baseline with no fine-tuning.

The baseline experiment involved direct cross-testing using the original cable-trained model on unseen classes, without any fine-tuning. As expected, the model failed to generalize, predicting nearly all test samples as defective. This resulted in perfect recall but extremely low precision across all classes. Despite high F1-scores, these were misleading due to imbalanced predictions, confirming the need for class-specific adaptation.

In the First Experiment, the model was fine-tuned using only 3 real good images from each target class. This minimal supervised setup already produced significant gains over the baseline. Precision and recall improved for several classes, especially *tile* and *wood*, which are structurally similar to cable. However, some classes, such as *transistor* and *metal\_nut*, still presented challenges due to limited anomaly representation during fine-tuning.

Defective Experiment introduced 6 synthetic defective images in addition to the 3 real good samples. This modest inclusion of synthetic anomalies led to a noticeable improvement in F1-scores and AUROC for most classes. *Pill*, *metal\_nut*, and *tile* benefited most from the added diversity in defect types. Nonetheless, *transistor* remained difficult to classify, suggesting a need for more varied or realistic training samples.

Even Experiment further balanced the training set by using 3 real good, 7 synthetic good, and 10 synthetic bad images. This approach enhanced performance across nearly all classes, with

consistent AUROC and PR AUC increases. Particularly, *metal\_nut* and *tile* reached strong F1-scores, and even *transistor* showed gains in stability and precision compared to prior setups.

Defective+ Experiment, the proportion of synthetic anomalies was increased to 20, maintaining 10 good images. This imbalance, favoring defective examples, yielded mixed results. While *metal\_nut*, *tile*, and *wood* performed well, recall dropped for *pill* and *transistor*, indicating potential bias toward over-detection of anomalies when negative samples dominate the training distribution.

Non-Defective Experiment flipped this balance by including 3 real good, 17 synthetic good, and 10 synthetic bad samples, emphasizing the non-defective class. This configuration produced some of the best overall results, particularly in *tile* and *metal\_nut*, where both AUROC and F1-scores peaked. The increased presence of high-quality good data seemed to anchor the model’s understanding of normalcy, resulting in more accurate anomaly separation.

Finally, Unsupervised Experiment used only non-defective data, 3 real good and 17 synthetic good images, without introducing any defective samples. Surprisingly, this unsupervised approach performed competitively. For most classes, performance remained high or even improved (*metal\_nut*, *tile*, and *wood*), demonstrating that learning a robust representation of the non-defective class alone can enable effective anomaly detection in the GLASS benchmark. The results reinforce the viability of one-class fine-tuning strategies, especially when defective data is unavailable or hard to generate.

In summary, all fine-tuning experiments outperformed the baseline. Introducing synthetic data, particularly when carefully balanced with real good samples, consistently improved generalization. While adding anomalies aids learning, excessive imbalance can impair results. Notably, fine-tuning with only good samples (Unsupervised Experiment) proved highly effective in certain cases, reinforcing the idea that data-efficient, unsupervised adaptation can be a practical strategy in real-world industrial scenarios. This outcome is somewhat expected, given the unsupervised nature of the GLASS benchmark itself, which is designed to train models solely on non-defective (good) samples, mirroring the conditions of this experiment.

## 4.4 Zero-shot Learning with CLIP

The Table 4.15 presents the performance metrics and confusion matrices obtained from the zero-shot anomaly detection experiment using CLIP on five unseen MVTec-AD classes. Without using any fine-tuning or class-specific training data, the model classified test images based solely on similarity to textual prompts describing non-defective and defective objects. This experiment evaluates CLIP’s ability to generalize to new domains in a fully unsupervised setting, providing a reference point for comparison against both the baseline cross-testing results (Annex A) and the subsequent transfer learning experiments. The following analysis highlights the strengths and limitations of the CLIP-based approach in relation to these alternative strategies.

The experimental setup for zero-shot anomaly detection involved evaluating CLIP’s performance on five unseen object categories from the MVTec AD dataset: *metal\_nut*, *pill*, *transistor*,

*tile*, and *wood*. No training was performed on these classes—neither using images nor class-specific features. Instead, classification was based entirely on similarity between each test image and two fixed textual prompts describing a *non-defective* (e.g., “A photo of a normal tile”) and a *defective* (e.g., “A photo of a defective tile”) object. These prompts were defined per class and loaded from a configuration file, but only the first prompt of each type was used during inference. For each test image, CLIP computed similarity scores with the two prompts, and the label was assigned based on the higher score. The model was evaluated using standard metrics including AUROC, precision, recall, F1-score, and PR AUC, along with confusion matrices, to analyze classification behavior in the absence of any domain-specific supervision.

The corresponding AUROC and PR-AUC curves for the zero-shot learning experiment are included in Annex D. These plots provide a visual perspective on the model’s decision behavior across different thresholds, facilitating a deeper understanding of CLIP’s separability performance and supporting its comparison with the other strategies explored in this work.

Table 4.15: Zero-Shot CLIP-Based Anomaly Detection Performance on Unseen MVTec-AD Classes

| Classe                 | AUROC  | Precision | Recall | F1-Score | PR AUC | Confusion Matrix                                  |
|------------------------|--------|-----------|--------|----------|--------|---------------------------------------------------|
| mvtec_cable_wood       | 0.9100 | 1.00      | 0.58   | 0.74     | 0.98   | $\begin{bmatrix} 32 & 23 \\ 0 & 14 \end{bmatrix}$ |
| mvtec_cable_pill       | 0.4500 | 0.86      | 0.99   | 0.92     | 0.85   | $\begin{bmatrix} 133 & 1 \\ 21 & 0 \end{bmatrix}$ |
| mvtec_cable_transistor | 0.7000 | 0.44      | 0.89   | 0.59     | 0.66   | $\begin{bmatrix} 32 & 4 \\ 40 & 15 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.8000 | 0.95      | 0.71   | 0.81     | 0.96   | $\begin{bmatrix} 63 & 26 \\ 3 & 14 \end{bmatrix}$ |
| mvtec_cable_tile       | 0.9800 | 0.94      | 0.96   | 0.95     | 0.99   | $\begin{bmatrix} 76 & 3 \\ 5 & 23 \end{bmatrix}$  |

The zero-shot anomaly detection experiment using CLIP presents an alternative approach to traditional supervised fine-tuning by relying exclusively on semantic similarity between images and textual prompts. Unlike the baseline cross-testing experiment, in which a model trained only on the cable class was directly applied to unseen classes without adaptation, the CLIP-based method was able to make meaningful predictions by comparing test images to generic textual descriptions of “normal” and “defective” objects.

In the baseline scenario, although high precision and F1-score values were recorded, these metrics were misleading. The confusion matrices revealed that the model systematically labeled all images as defective, yielding perfect recall but zero true negatives. This indicates a total lack of generalization when transferring to unseen classes. For example, the confusion matrix for *pill* was  $\begin{bmatrix} 0, 21 \\ 0, 134 \end{bmatrix}$ , and for *metal\_nut*,  $\begin{bmatrix} 0, 17 \\ 0, 89 \end{bmatrix}$ , illustrating that the baseline model failed to differentiate between non-defective and defective instances in any of the new classes.

By contrast, the CLIP model showed more balanced behavior across most classes. Although it had no access to training data from the target categories, it was able to leverage textual cues to separate non-defective from defective samples. For instance, CLIP achieved an AUROC of 0.98

and an F1-score of 0.95 for the *tile* class, and demonstrated strong performance on *wood* (AUROC 0.91, F1-score 0.74). However, some classes such as *pill* (AUROC 0.45) and *transistor* (AUROC 0.70) remained more difficult to detect, possibly due to less distinctive textual-visual alignment or subtle defect patterns.

When compared to the transfer learning experiments, the advantage of CLIP lies primarily in its ability to generalize without requiring any labeled training data from the target domain. Experiment 0, which involved only 3 real training images per class, resulted in lower AUROC and F1-scores than CLIP in some categories, such as *tile* and *wood*. However, as more synthetic and real data were introduced in subsequent experiments, transfer learning approaches began to outperform CLIP consistently. For example, Experiment 5, which included 3 real and 17 synthetic good images, achieved an F1-score of 0.97 for *metal\_nut* and 0.96 for *wood*, surpassing CLIP’s performance in those classes.

Overall, CLIP proves valuable in truly low-resource settings, offering competitive results without any fine-tuning. Nevertheless, when even limited amounts of labeled data are available, transfer learning strategies yield superior performance due to the ability to specialize to domain-specific visual characteristics. This comparison underscores the complementarity of zero-shot and few-shot approaches depending on the data availability and deployment constraints.

## 4.5 Stable Diffusion Experiment

Before analyzing the quantitative metrics, Figure 4.2 presents representative examples of synthetic images generated for all five test classes (*pill*, *metal\_nut*, *tile*, *wood*, and *transistor*) using Stable Diffusion v1.5 through purely textual prompts. These visuals offer insight into the diversity and realism of the generated data, which served as the exclusive training source for the anomaly detection model.

Table 4.16 presents the results of the cross-testing experiment in which a model was trained solely on synthetic images of the cable class generated using Stable Diffusion v1.5 through text-to-image prompting. The model was then evaluated on real test samples from various MVTec AD classes. This experiment aimed to assess the model’s ability to generalize to unseen classes, especially when trained exclusively on synthetic data generated from textual descriptions.

The AUROC and PR-AUC plots for the Stable Diffusion-based cross-testing experiment are provided in Annex C. These visualizations help illustrate the classification dynamics of the model under varying thresholds and offer a clearer understanding of its anomaly detection behavior when trained exclusively on synthetic data. They also serve as a complementary resource for evaluating how synthetic-only training affects generalization across heterogeneous object categories.

The results indicate a strong variance in performance across target classes. For *metal\_nut*, *tile*, and *wood*, the model achieved high scores in precision, recall, and F1-score, with *wood* in particular reaching an F1-score of 0.98. This suggests that the model trained on cable data was capable of generalizing effectively to structurally similar industrial components, likely due to the

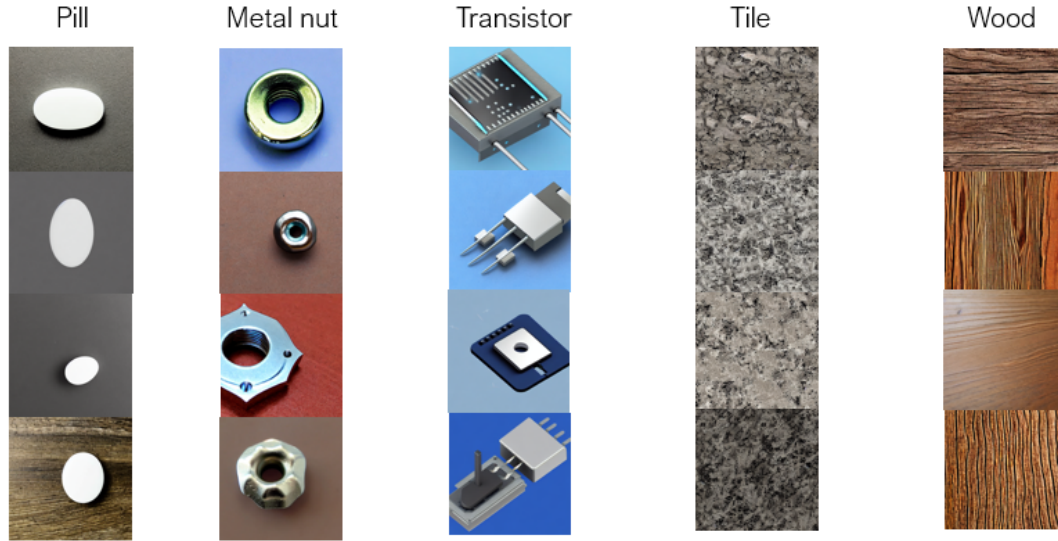


Figure 4.2: Examples of synthetic images generated from text prompts for each of the five test classes: *pill*, *metal\_nut*, *tile*, *wood*, and *transistor*.

presence of consistent textures, simple geometries, and homogeneous backgrounds that are shared across these classes.

In contrast, the performance on the *pill* class was completely degraded, with precision, recall, and F1-score all equal to 0.00. Although the PR AUC remained high (0.93), the decision threshold computed automatically was infinite (Inf), which prevented any sample from being classified as anomalous. This result highlights a critical failure in generalization due to the visual discrepancy between the synthetic *pill* images—generated solely from textual prompts—and the real MVTec *pill* samples, which vary significantly in terms of shape, scale, packaging, and photographic context. The inability of the synthetic data to reflect the visual distribution of the test set rendered the model ineffective for anomaly detection in this case.

For the *transistor* class, no results were reported. The model was unable to generate visually coherent synthetic samples of transistors using Stable Diffusion v1.5. Despite variations in prompt engineering, the generated images lacked structural similarity to the complex industrial components present in the MVTec AD dataset. This limitation illustrates the challenges of synthetic data generation for highly specific and structurally intricate objects without further fine-tuning or visual guidance mechanisms.

In summary, the results of this experiment demonstrate that synthetic data generated via general-purpose diffusion models can support effective anomaly detection when the synthetic-to-real domain gap is sufficiently small. However, generalization fails when this gap becomes too large, particularly for visually complex or unfamiliar object classes. These findings highlight the importance of incorporating domain adaptation strategies, visual conditioning techniques (e.g., img2img, CLIP-guided generation), or few-shot fine-tuning with real samples to improve the robustness of models trained with synthetic data in heterogeneous industrial environments.

Table 4.16: Performance metrics and confusion matrices for MVTec Cable-based classification on target classes

| Classe                 | AUROC | Precision | Recall | F1-Score | PR AUC | Threshold | Confusion Matrix                                  |
|------------------------|-------|-----------|--------|----------|--------|-----------|---------------------------------------------------|
| mvtec_cable_pill       | 0.50  | 0.00      | 0.00   | 0.00     | 0.93   | Inf       | $\begin{bmatrix} 0 & 134 \\ 0 & 21 \end{bmatrix}$ |
| mvtec_cable_metal_nut  | 0.66  | 0.90      | 0.67   | 0.77     | 0.91   | 0.9999    | $\begin{bmatrix} 60 & 29 \\ 7 & 10 \end{bmatrix}$ |
| mvtec_cable_tile       | 0.90  | 1.00      | 0.77   | 0.87     | 0.97   | 0.9998    | $\begin{bmatrix} 61 & 18 \\ 0 & 28 \end{bmatrix}$ |
| mvtec_cable_wood       | 0.94  | 0.98      | 0.96   | 0.97     | 0.98   | 0.9999    | $\begin{bmatrix} 53 & 2 \\ 1 & 13 \end{bmatrix}$  |
| mvtec_cable_transistor | -     | -         | -      | -        | -      | -         | -                                                 |

## 4.6 Technical Discussion and Analysis of Results

The experiments conducted across cross-testing, transfer learning, and zero-shot learning reveal key insights into the challenge of generalizing anomaly detection models to unseen object classes in industrial settings. A central pattern emerges: while models perform strongly within the training domain, their performance degrades significantly when confronted with structurally or semantically different object classes.

The baseline GLASS model achieved near-perfect performance on the MVTec AD dataset (mean AUROC 0.9988, F1-score 0.9940), confirming its robustness for in-domain anomaly detection. However, cross-testing revealed poor generalization to unseen classes (mean AUROC  $\approx 0.60$ , precision  $\approx 0.77$ ), with models over-classifying defects due to domain shifts, particularly for complex classes like *pill* and *transistor*. Transfer learning experiments demonstrated that synthetic data augmentation, generated via InstructPix2Pix, significantly improved generalization. Unsupervised Experiment, using only non-defective data (3 real, 17 synthetic), achieved the best AUROC for some classes (e.g., *wood* AUROC 0.99, F1 0.98; *metal\_nut* AUROC 0.93, F1 0.92), highlighting the efficacy of unsupervised normality modeling in GLASS’s one-class framework. Non-Defective Experiment (3 real non-defective, 17 synthetic non-defective, 10 synthetic defective) also performed strongly, with balanced datasets enhancing cross-domain robustness (e.g., *tile* AUROC 0.97, F1 0.94).

Synthetic data played a critical role in enhancing generalization, especially when real data was scarce. Models fine-tuned with a reasonable amount of synthetic non-defective and defective samples (e.g., Even Experiment and Non-Defective Experiment) outperformed those trained with unbalanced or purely anomalous data (e.g., Defective+ Experiment). Notably, the Unsupervised Experiment, where no defective samples were used at all, showed that accurate modeling of normality can yield competitive results, especially in the GLASS framework which is inherently one-class in nature.

Zero-shot learning with CLIP provided a data-free alternative, excelling for texture-based classes (*tile* AUROC 0.98, F1 0.95) but underperforming for complex objects (*pill* AUROC 0.45, F1 0.92) due to limited text-visual alignment. While integrated into iPortalDoc for its simplicity, CLIP was outperformed by transfer learning when minimal labeled data was available. The Stable

Diffusion experiment showed promise for texture-rich classes (*wood* AUROC 0.94, F1 0.97) but failed for *pill* (AUROC 0.50, F1 0.00) and *transistor* (no results) due to a significant synthetic-real domain gap, highlighting limitations in text-only generation for intricate objects.

However, experiments involving synthetic data generated via Stable Diffusion showed that not all generative approaches are equally effective. When synthetic images failed to capture the structural or contextual fidelity of real defects, such as in the *pill* or *transistor* classes, performance degraded or collapsed entirely. These failures reflect the limitations of relying solely on text-to-image generation without visual guidance or domain adaptation.

In summary, the key findings from these experiments can be outlined as follows:

- **Zero-shot inference with CLIP** is a viable fallback strategy in fully unsupervised scenarios, especially for classes with visually distinctive anomalies and clear textual descriptors.
- **Few-shot fine-tuning with real non-defective samples**, even in very low quantities, can dramatically improve generalization, particularly when augmented with high-quality synthetic data.
- **Balanced training sets** that reflect both non-defective and anomalous distributions provide better results than anomaly-dominant or anomaly-absent configurations.
- **Synthetic data is most useful when visually coherent with the target domain.** Large domain gaps between generated and real images reduce transferability and hurt performance.
- **Training exclusively with non-defective samples** (Unsupervised Experiment) can be effective and even preferable in some settings, aligning with the one-class nature of the GLASS benchmark.

These observations are particularly relevant for real-world industrial applications where defective samples are rare, heterogeneous, or difficult to annotate. Depending on the available data and the complexity of the object class, practitioners must balance between unsupervised, few-shot, and synthetic-based approaches. Overall, this study demonstrates that generalization in anomaly detection is achievable, but hinges on data realism, domain alignment, and careful strategy selection.

## 4.7 Integration with iPortalDoc

As the adoption of Artificial Intelligence (AI) in industrial environments continues to accelerate, integrating intelligent models into existing enterprise systems becomes essential to bridge the gap between research and practical deployment. Document management platforms like *iPortalDoc* are widely used in companies to organize, validate, and process information across departments. These platforms handle a variety of documents, including scanned images and photos of physical components, which often require manual classification, inspection, or validation. By embedding

an AI-based visual inspection module directly into this workflow, we aim to automate and streamline anomaly detection tasks, reducing human effort and increasing operational consistency.

To demonstrate the practical applicability of the proposed anomaly detection approach, a real-world integration was developed within the *iPortalDoc* platform, a widely adopted document and process management system. The objective of this integration is to enable automated classification of image-based documents using a zero-shot learning (ZSL) model, while maintaining usability, scalability, and human oversight.

The integration leverages OpenAI’s CLIP model through a RESTful inference service built with Flask. This model was chosen due to its ability to generalize to new object categories without the need for additional training data. Compared to transfer learning approaches, CLIP-based zero-shot inference offers a lightweight and deployable solution that minimizes the operational overhead of model fine-tuning and dataset curation.

The implemented workflow is as follows: when a user uploads a document to *iPortalDoc*, they provide a brief description of the object depicted in the image (e.g., “valve” or “bottle”). The document is then routed to a dedicated workflow named *Anomaly Detection*, which presents the user with a menu interface to manage the document, as shown in Figure 4.3. This menu allows the user to confirm the document’s metadata and initiate the anomaly detection process. A PHP script on the server retrieves the document metadata, including the object description and image path, and issues an HTTP POST request to the Flask application, sending both the image and the object name.

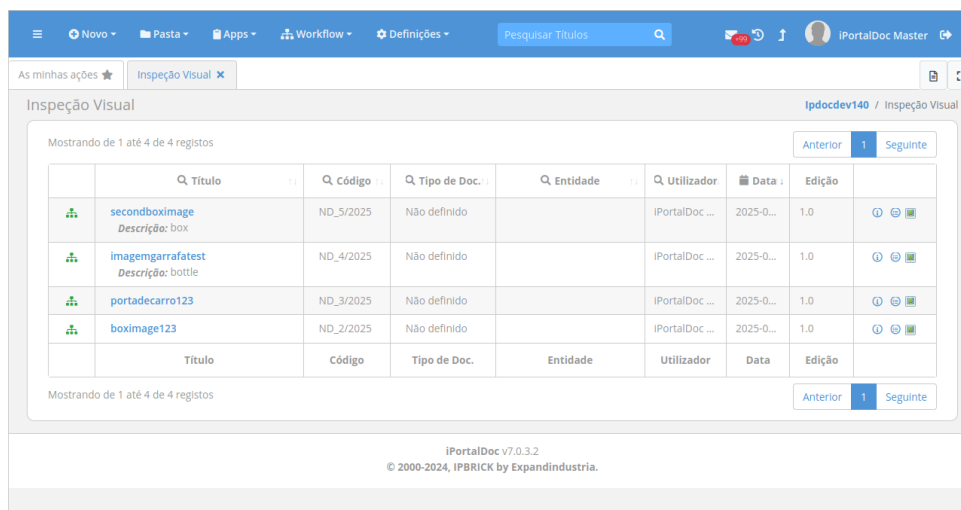


Figure 4.3: Menu interface in iPortalDoc displayed after document creation, in a folder called "Inspeção Visual" for testing purposes, allowing users to manage the document and initiate the anomaly detection workflow.

Upon receiving the request, the Flask server processes the image and constructs two textual prompts: “A photo of a normal <object>” and “A photo of a defective <object>”. The CLIP model computes similarity scores between the image and these prompts, returning a prediction



(*non-defective* or *defective*) along with a confidence score. This result is returned as a JSON response, which the PHP script parses and writes into the document's summary field in a natural language format (e.g., “*Image 'valve' classified as defective with 92.37% confidence*”). The document's updated information, including the classification result, is displayed in the *iPortalDoc* interface, as illustrated in Figure 4.4.

| iPortalDoc®                                  |                                                                      |                 |                             |
|----------------------------------------------|----------------------------------------------------------------------|-----------------|-----------------------------|
| Imagemgarrafatest   ND_4/2025   Não definido |                                                                      |                 |                             |
| DOCUMENTO                                    |                                                                      |                 |                             |
| Título:                                      | Imagemgarrafatest                                                    | Data:           | 2025-06-13 13:00:12         |
| Código:                                      | ND_4/2025                                                            | Workflow:       | Deteção de Anomalias        |
| Tipo de Doc.:                                | Não definido                                                         | Utilizador:     | iPortalDoc Master           |
| Entidade:                                    |                                                                      | 1º Utilizador:  | iPortalDoc Master           |
| Tipo de Entidade:                            |                                                                      | Elaborado em:   | 2025-06-13                  |
| Contacto:                                    |                                                                      | Edição:         | 1.0                         |
| Localização:                                 |                                                                      | Pasta:          | Ipdocdev140/Inspeção Visual |
| Sumário:                                     | Imagem "bottle" classificada como defeituosa com confiança de 82,51% | Descrição:      | bottle                      |
| Localização física:                          |                                                                      | Palavras chave: |                             |
| Departamento:                                |                                                                      | Projeto:        |                             |
| Ficheiro:                                    | defective_bottle_3 (1).png (449.31 KB)                               | Valor:          |                             |
| Atribuído:                                   | iPortalDoc Master                                                    | Ordem:          | 100                         |

Figure 4.4: Document information interface in *iPortalDoc* showing the classification result and confidence score after CLIP-based anomaly detection.

At this stage, the user is required to validate the result by either approving or rejecting the prediction. This manual step ensures that while the classification process is largely automated, human oversight is preserved to mitigate errors or handle edge cases. Once validated, the document proceeds to the final state of the workflow.

This integration illustrates how a state-of-the-art vision-language model can be embedded into enterprise software with minimal disruption, providing a scalable and interpretable AI-assisted decision-making tool within existing business processes.

Figure 4.5 illustrates the overall workflow architecture and data flow involved in this integration.

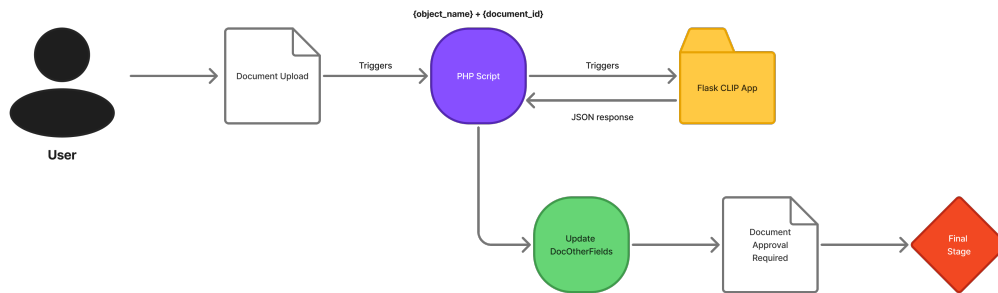


Figure 4.5: Overall workflow architecture and data flow involved in this integration.

From a broader perspective, this case study demonstrates that advanced AI models like

CLIP can be deployed in real-world enterprise platforms without requiring full infrastructure re-design. The use of standard web protocols (e.g., HTTP) and modular RESTful services ensures that the solution is maintainable, interpretable, and easy to scale. Furthermore, the human-in-the-loop design guarantees reliability and accountability, which are critical in quality assurance contexts. As such, this integration serves as a blueprint for embedding explainable and adaptable AI systems into existing industrial workflows.

## Chapter 5

# Conclusion and Future Work

This dissertation explored the development of generalizable visual anomaly detection techniques for industrial applications under conditions of limited annotated data. Addressing the needs of a real-world enterprise context, the work combined transfer learning, zero-shot learning, and synthetic data generation within the framework of the GLASS benchmark to investigate how different data regimes influence model generalization to unseen object categories.

The results demonstrated that even modest quantities of domain-relevant data, particularly when combined with synthetic augmentation, can substantially improve anomaly detection performance. Approaches based on transfer learning, especially those focused on modeling normality through non-defective samples, yielded strong results in data-scarce conditions. At the same time, the integration of vision-language models like CLIP showed the promise of data-free zero-shot inference, particularly for texture-based categories, although with clear limitations in object-centric or structurally complex scenarios.

The findings also highlight the importance of thoughtful data design. The quality, balance, and realism of synthetic samples, as well as the semantic precision of prompts, emerged as key factors influencing performance. When these elements were aligned with the underlying structure of the target class, models generalized effectively. Conversely, a mismatch between generated content and the visual complexity of real-world objects often led to suboptimal or inconsistent results.

Beyond empirical outcomes, this research illustrates how modern AI techniques can be meaningfully integrated into enterprise systems like *iPortalDoc*, enabling scalable decision support for quality assurance processes. The successful deployment of a CLIP-based workflow, despite its limitations, offers a blueprint for practical, low-maintenance integration of AI into document management systems.

However, several methodological limitations were identified that warrant further attention:

- **Subjective Synthetic Data Validation:** The manual selection of synthetic images introduced bias and inconsistency, particularly for visually complex classes such as *transistor*, due to the lack of formal quality metrics.

- **Limited Prompt Engineering:** The use of single, static prompts for each class in CLIP and Stable Diffusion may have constrained performance, especially in cases where the visual semantics of defects were subtle or ambiguous.
- **iPortalDoc Integration Bias:** While CLIP was selected for its simplicity and deployment efficiency, this decision overlooked the superior accuracy of transfer learning approaches, potentially limiting effectiveness in operational settings.

Building upon this work, the following directions are proposed for future research:

- **Automated Synthetic Data Validation:** Incorporate quantitative measures such as Fréchet Inception Distance (FID) or Structural Similarity Index (SSIM) to assess the quality and domain alignment of synthetic images.
- **Class-Specific Stable Diffusion Fine-Tuning:** Fine-tune diffusion models using real images from each target class to improve the semantic fidelity and structural realism of generated anomalies.
- **Advanced Prompt Optimization:** Employ automated prompt engineering strategies (e.g., CLIP-based ranking, reinforcement learning, or evolutionary algorithms) to enhance text-visual alignment in generative and zero-shot inference tasks.
- **Rigorous Class Selection Methodology:** Define systematic criteria for selecting test classes based on visual diversity, industrial relevance, and application requirements, potentially using clustering or domain expert input.
- **Broader Dataset Validation:** Evaluate the methods on additional benchmarks (e.g., VisA, MPDD) to test generalization capabilities in other industrial and visual inspection domains.
- **Explainability Integration:** Introduce interpretability techniques (e.g., attention visualization or saliency mapping) to support human validation and build trust in anomaly detection outcomes within enterprise systems.

Ultimately, this study affirms that robust and generalizable anomaly detection is attainable even in data-constrained environments. Achieving this, however, requires a balanced synthesis of domain knowledge, synthetic augmentation, and adaptable AI frameworks, each carefully aligned to the realities of industrial deployment.

# References

- [1] A. Krizhevsky, “Learning multiple layers of features from tiny images,” 2009. [Online]. Available: <https://api.semanticscholar.org/CorpusID:18268744>.
- [2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [3] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009, ISSN: 0360-0300. DOI: [10.1145/1541880.1541882](https://doi.org/10.1145/1541880.1541882). [Online]. Available: <https://doi.org/10.1145/1541880.1541882>.
- [4] M. Injadat, F. Salo, A. Nassif, A. Essex, and A. Shami, “Bayesian optimization with machine learning algorithms towards anomaly detection,” Dec. 2018. DOI: [10.1109/GLOCOM.2018.8647714](https://doi.org/10.1109/GLOCOM.2018.8647714).
- [5] P. U. Eze, N. Geard, I. Mueller, and I. Chades, “Anomaly detection in endemic disease surveillance data using machine learning techniques,” *Healthcare*, vol. 11, no. 13, 2023, ISSN: 2227-9032. DOI: [10.3390/healthcare11131896](https://doi.org/10.3390/healthcare11131896). [Online]. Available: <https://www.mdpi.com/2227-9032/11/13/1896>.
- [6] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection for discrete sequences: A survey,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 823–839, 2012. DOI: [10.1109/TKDE.2010.235](https://doi.org/10.1109/TKDE.2010.235).
- [7] I. Bose and R. Mahapatra, “Business data mining - a machine learning perspective,” *Information Management*, vol. 39, pp. 211–225, Dec. 2001. DOI: [10.1016/S0378-7206\(01\)00091-X](https://doi.org/10.1016/S0378-7206(01)00091-X).
- [8] U. Fiore, F. Palmieri, A. Castiglione, and A. Santis, “Network anomaly detection with the restricted boltzmann machine,” *Neurocomputing*, vol. 122, pp. 13–23, Dec. 2013. DOI: [10.1016/j.neucom.2012.11.050](https://doi.org/10.1016/j.neucom.2012.11.050).
- [9] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, “Machine learning for anomaly detection: A systematic review,” *IEEE Access*, vol. 9, pp. 78 658–78 700, 2021. DOI: [10.1109/ACCESS.2021.3083060](https://doi.org/10.1109/ACCESS.2021.3083060).
- [10] W.-H. Choi and J. Kim, “Unsupervised learning approach for anomaly detection in industrial control systems,” *Applied System Innovation*, vol. 7, no. 2, 2024, ISSN: 2571-5577. DOI: [10.3390/asi7020018](https://doi.org/10.3390/asi7020018). [Online]. Available: <https://www.mdpi.com/2571-5577/7/2/18>.
- [11] Y. Long, Z. Ling, S. Brook, D. McFarlane, and A. Brintrup, *Leveraging unsupervised learning for cost-effective visual anomaly detection*, 2024. arXiv: [2409.15980](https://arxiv.org/abs/2409.15980) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2409.15980>.

- [12] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, “Mvtec ad – a comprehensive real-world dataset for unsupervised anomaly detection,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9584–9592. DOI: [10.1109/CVPR.2019.00982](https://doi.org/10.1109/CVPR.2019.00982).
- [13] P. Bergmann, K. Batzner, M. Fauser, D. Sattlegger, and C. Steger, “The mvtec anomaly detection dataset: A comprehensive real-world dataset for unsupervised anomaly detection,” *International Journal of Computer Vision*, vol. 129, no. 4, pp. 1038–1059, 2021. DOI: [10.1007/s11263-020-01400-4](https://doi.org/10.1007/s11263-020-01400-4).
- [14] C. Wang, W. Zhu, B.-B. Gao, *et al.*, *Real-iad: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection*, 2024. arXiv: [2403.12580](https://arxiv.org/abs/2403.12580) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2403.12580>.
- [15] G. Xie, J. Wang, J. Liu, *et al.*, *Im-iad: Industrial image anomaly detection benchmark in manufacturing*, 2024. arXiv: [2301.13359](https://arxiv.org/abs/2301.13359) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2301.13359>.
- [16] P. Mishra, R. Verk, D. Fornasier, C. Piciarelli, and G. L. Foresti, “Vt-adl: A vision transformer network for image anomaly detection and localization,” in *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, IEEE, Jun. 2021, pp. 01–06. DOI: [10.1109/isie45552.2021.9576231](https://doi.org/10.1109/isie45552.2021.9576231). [Online]. Available: <http://dx.doi.org/10.1109/ISIE45552.2021.9576231>.
- [17] Q. Wang, S. Gao, J. Hu, *et al.*, *Hss-iad: A heterogeneous same-sort industrial anomaly detection dataset*, 2025. arXiv: [2504.12689](https://arxiv.org/abs/2504.12689) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2504.12689>.
- [18] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, *Spot-the-difference self-supervised pre-training for anomaly detection and segmentation*, 2022. arXiv: [2207.14315](https://arxiv.org/abs/2207.14315) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2207.14315>.
- [19] S. Jezek, M. Jonak, R. Burget, P. Dvorak, and M. Skotak, “Deep learning-based defect detection of metal parts: Evaluating current methods in complex conditions,” in *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2021, pp. 66–71. DOI: [10.1109/ICUMT54235.2021.9631567](https://doi.org/10.1109/ICUMT54235.2021.9631567).
- [20] Y. Zheng, X. Wang, Y. Qi, W. Li, and L. Wu, *Benchmarking unsupervised anomaly detection and localization*, 2022. arXiv: [2205.14852](https://arxiv.org/abs/2205.14852) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2205.14852>.
- [21] G. Pang, C. Shen, L. Cao, and A. v. d. Hengel, “Deep learning for anomaly detection: A review,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.
- [22] R. Chalapathy and S. Chawla, “Deep learning for anomaly detection: A survey,” *arXiv preprint arXiv:1901.03407*, 2019.
- [23] V. Zavrtanik, M. Kristan, and D. Skočaj, “Draem-a discriminatively trained reconstruction embedding for surface anomaly detection,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 8330–8339.
- [24] J. An and S. Cho, “Variational autoencoder based anomaly detection using reconstruction probability,” *Special lecture on IE*, vol. 2, no. 1, pp. 1–18, 2015.
- [25] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, “Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings,” *arXiv preprint arXiv:1911.02357*, 2019.

- [26] N. Cohen and Y. Hoshen, “Sub-image anomaly detection with deep pyramid correspondence,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 481–490.
- [27] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, *Towards total recall in industrial anomaly detection*, 2022. arXiv: 2106.08265 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2106.08265>.
- [28] T. Defard, A. Setkov, A. Loesch, and R. Audigier, *Padim: A patch distribution modeling framework for anomaly detection and localization*, 2020. arXiv: 2011.08785 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2011.08785>.
- [29] L. Ruff, R. A. Vandermeulen, N. Görnitz, *et al.*, “Deep one-class classification,” in *International Conference on Machine Learning (ICML)*, PMLR, 2018, pp. 4393–4402.
- [30] M. Rudolph, B. Wandt, and B. Rosenhahn, “Same same but different: A near-identical diffusion process for anomaly detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 4237–4247.
- [31] M. Tailanian, Á. Pardo, and P. Musé, “U-flow: A u-shaped normalizing flow for anomaly detection with unsupervised threshold,” *Journal of Mathematical Imaging and Vision*, vol. 66, pp. 678–696, 2024. DOI: 10.1007/s10851-024-01193-y. [Online]. Available: <https://doi.org/10.1007/s10851-024-01193-y>.
- [32] G. Ham, S. Kim, S. Lee, J.-H. Lee, and D. Kim, *Cosine similarity knowledge distillation for individual class information transfer*, 2023. arXiv: 2311.14307 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2311.14307>.
- [33] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, *Cutpaste: Self-supervised learning for anomaly detection and localization*, 2021. arXiv: 2104.04015 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2104.04015>.
- [34] A. A. Neloy and M. Turgeon, “A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs,” *Machine Learning with Applications*, vol. 17, p. 100572, 2024, ISSN: 2666-8270. DOI: <https://doi.org/10.1016/j.mlwa.2024.100572>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2666827024000483>.
- [35] Y.-X. Xu, M. Pang, J. Feng, K. M. Ting, Y. Jiang, and Z.-H. Zhou, “Reconstruction-based anomaly detection with completely random forest,” in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pp. 127–135. DOI: 10.1137/1.9781611976700.15. eprint: <https://epubs.siam.org/doi/pdf/10.1137/1.9781611976700.15>. [Online]. Available: <https://epubs.siam.org/doi/abs/10.1137/1.9781611976700.15>.
- [36] R. Hu, C. C. Aggarwal, S. Ma, and J. Huai, “An embedding approach to anomaly detection,” in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, 2016, pp. 385–396. DOI: 10.1109/ICDE.2016.7498256.
- [37] R. Jiang, Z. Yang, and J. Zhao, “A complete deep support vector data description for one class learning,” *IEEE Access*, vol. 11, pp. 117494–117507, 2023. DOI: 10.1109/ACCESS.2023.3325734.
- [38] Y. Chen, X. S. Zhou, and T. Huang, “One-class svm for learning in image retrieval,” in *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, vol. 1, 2001, 34–37 vol.1. DOI: 10.1109/ICIP.2001.958946.

- [39] A. M. Bartkowiak, “Anomaly, novelty, one-class classification: A short introduction,” in *2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*, 2010, pp. 1–6. DOI: [10.1109/CISIM.2010.5643699](https://doi.org/10.1109/CISIM.2010.5643699).
- [40] O. Hirschorn and S. Avidan, “Normalizing flows for human pose anomaly detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 13 545–13 554.
- [41] A. Ryzhikov, M. Borisyak, and D. Derkach, “Nfad: Fixing anomaly detection using normalizing flows,” *PeerJ Computer Science*, vol. 7, e757, Nov. 2021. DOI: [10.7717/peerj-cs.757](https://doi.org/10.7717/peerj-cs.757).
- [42] L. Yao and S. Gao, *Dual-student knowledge distillation networks for unsupervised anomaly detection*, 2024. arXiv: [2402.00448](https://arxiv.org/abs/2402.00448) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2402.00448>.
- [43] H. M. Schlüter, J. Tan, B. Hou, and B. Kainz, “Natural synthetic anomalies for self-supervised anomaly detection and localization,” in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Cham: Springer Nature Switzerland, 2022, pp. 474–489, ISBN: 978-3-031-19821-2.
- [44] V. Zavrtanik, M. Kristan, and D. Skočaj, “Dsr – a dual subspace re-projection network for surface anomaly detection,” in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Cham: Springer Nature Switzerland, 2022, pp. 539–554, ISBN: 978-3-031-19821-2.
- [45] Z. Liu, Y. Zhou, Y. Xu, and Z. Wang, “Simplenet: A simple network for image anomaly detection and localization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 20 402–20 411.
- [46] Z. You, L. Cui, Y. Shen, *et al.*, “A unified model for multi-class anomaly detection,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 4571–4584. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/1d774c112926348c3e25ea47d87c835b-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/1d774c112926348c3e25ea47d87c835b-Paper-Conference.pdf).
- [47] A. S. Namaghi, “Evaluation of unsupervised anomaly detection (euad) algorithms,” English, External coach: Naser Asghari, Master’s thesis, Eindhoven University of Technology, Nov. 2022.
- [48] A. Emmott, S. Das, T. Dietterich, A. Fern, and W.-K. Wong, *A meta-analysis of the anomaly detection problem*, 2016. arXiv: [1503.01158](https://arxiv.org/abs/1503.01158) [cs.AI]. [Online]. Available: <https://arxiv.org/abs/1503.01158>.
- [49] R. Talianian and *et al.*, “A u-shaped normalizing flow for anomaly detection with automatic segmentation thresholds,” *arXiv preprint arXiv:2211.12353*, 2022, Evaluates mIoU, AU-ROC and AUPRO on MVTec AD. [Online]. Available: <https://arxiv.org/abs/2211.12353>.
- [50] E. Horwitz and Y. Hoshen, *Back to the feature: Classical 3d features are (almost) all you need for 3d anomaly detection*, 2022. arXiv: [2203.05550](https://arxiv.org/abs/2203.05550) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2203.05550>.
- [51] Q. Chen, H. Luo, C. Lv, and Z. Zhang, *A unified anomaly synthesis strategy with gradient ascent for industrial anomaly detection and localization*, 2024. arXiv: [2407.09359](https://arxiv.org/abs/2407.09359) [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2407.09359>.



- [52] S. Wei, J. Jiang, and X. Xu, “Uninet: A contrastive learning-guided unified framework with feature selection for anomaly detection,” 2025.
- [53] Z. Chen, X. Xie, L. Yang, and J. Lai, *Hard-normal example-aware template mutual matching for industrial anomaly detection*, 2024. arXiv: 2303.16191 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2303.16191>.
- [54] W. Luo, Y. Cao, H. Yao, *et al.*, *Exploring intrinsic normal prototypes within a single image for universal anomaly detection*, 2025. arXiv: 2503.02424 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2503.02424>.
- [55] Q. Chen, H. Luo, H. Gao, C. Lv, and Z. Zhang, “Progressive boundary guided anomaly synthesis for industrial anomaly detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 35, no. 2, pp. 1193–1208, Feb. 2025, ISSN: 1558-2205. DOI: 10.1109/tcsvt.2024.3479887. [Online]. Available: <http://dx.doi.org/10.1109/TCSVT.2024.3479887>.
- [56] J. Wang, C. Lan, C. Liu, *et al.*, “Generalizing to unseen domains: A survey on domain generalization,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 8, pp. 8052–8072, 2023. DOI: 10.1109/TKDE.2022.3178128.
- [57] F. Zhuang, Z. Qi, K. Duan, *et al.*, “A comprehensive survey on transfer learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021. DOI: 10.1109/JPROC.2020.3004555.
- [58] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, “Visual domain adaptation: A survey of recent advances,” *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 53–69, 2015. DOI: 10.1109/MSP.2014.2347059.
- [59] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.
- [60] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, “Meta-learning in neural networks: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5149–5169, 2022. DOI: 10.1109/TPAMI.2021.3079209.
- [61] K. Shaheen, M. Hanif, O. Hasan, *et al.*, “Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks,” *Journal of Intelligent Robotic Systems*, vol. 105, p. 9, 2022, Received: 03 October 2021, Accepted: 24 February 2022, Published: 23 April 2022. DOI: 10.1007/s10846-022-01603-6. [Online]. Available: <https://doi.org/10.1007/s10846-022-01603-6>.
- [62] W. Wang, V. W. Zheng, H. Yu, and C. Miao, “A survey of zero-shot learning: Settings, methods, and applications,” *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, Jan. 2019, ISSN: 2157-6904. DOI: 10.1145/3293318. [Online]. Available: <https://doi.org/10.1145/3293318>.
- [63] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, *Generative adversarial networks*, 2014. arXiv: 1406.2661 [stat.ML]. [Online]. Available: <https://arxiv.org/abs/1406.2661>.
- [64] A. Hakami, “Strategies for overcoming data scarcity, imbalance, and feature selection challenges in machine learning models for predictive maintenance,” *Scientific Reports*, vol. 14, p. 9645, 2024. DOI: 10.1038/s41598-024-59958-9. [Online]. Available: <https://doi.org/10.1038/s41598-024-59958-9>.
- [65] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, *Image-to-image translation with conditional adversarial networks*, 2018. arXiv: 1611.07004 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/1611.07004>.

- [66] H. Sun, Y. Cao, H. Dong, and O. Fink, *Unseen visual anomaly generation*, 2025. arXiv: 2406.01078 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2406.01078>.
- [67] A. Radford, J. W. Kim, C. Hallacy, *et al.*, “Learning transferable visual models from natural language supervision,” in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 8748–8763. [Online]. Available: <https://proceedings.mlr.press/v139/radford21a.html>.
- [68] J. Jeong, Y. Zou, T. Kim, D. Zhang, A. Ravichandran, and O. Dabeer, *Winclip: Zero-/few-shot anomaly classification and segmentation*, 2023. arXiv: 2303.14814 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2303.14814>.
- [69] T. Brooks, A. Holynski, and A. A. Efros, *Instructpix2pix: Learning to follow image editing instructions*, 2023. arXiv: 2211.09800 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2211.09800>.

## Appendix A

### Cross-Testing Table per Class

Table A.1: Cross-testing results with model trained on *carpet*.

| Row Names                        | image_auroc | precision | recall | f1_score | pr_auc   | threshold | confusion_matrix    |
|----------------------------------|-------------|-----------|--------|----------|----------|-----------|---------------------|
| mvtec_carpet_on_mvtec_carpet     | 0.997930    | 0.976744  | 1.0    | 0.988235 | 0.999440 | 0.804231  | [[21, 2], [0, 84]]  |
| mvtec_carpet_on_mvtec_grid       | 0.487380    | 0.764706  | 1.0    | 0.866667 | 0.800539 | 0.999996  | [[0, 16], [0, 52]]  |
| mvtec_carpet_on_mvtec_leather    | 0.607918    | 0.763158  | 1.0    | 0.865672 | 0.882670 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_carpet_on_mvtec_tile       | 0.662975    | 0.738318  | 1.0    | 0.849462 | 0.876991 | 0.999991  | [[0, 28], [0, 79]]  |
| mvtec_carpet_on_mvtec_wood       | 0.500000    | 0.797101  | 1.0    | 0.887097 | 0.898551 | 1.000000  | [[0, 14], [0, 55]]  |
| mvtec_carpet_on_mvtec_bottle     | 0.500000    | 0.800000  | 1.0    | 0.888889 | 0.900000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_carpet_on_mvtec_cable      | 0.500000    | 0.613139  | 1.0    | 0.760181 | 0.806569 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_carpet_on_mvtec_capsule    | 0.500000    | 0.852459  | 1.0    | 0.920354 | 0.926230 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_carpet_on_mvtec_hazelnut   | 0.500000    | 0.653465  | 1.0    | 0.790419 | 0.826733 | 1.000000  | [[0, 35], [0, 66]]  |
| mvtec_carpet_on_mvtec_metal_nut  | 0.500000    | 0.839623  | 1.0    | 0.912821 | 0.919811 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_carpet_on_mvtec_pill       | 0.500000    | 0.864516  | 1.0    | 0.927336 | 0.932258 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_carpet_on_mvtec_screw      | 0.500000    | 0.760000  | 1.0    | 0.863636 | 0.880000 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_carpet_on_mvtec_toothbrush | 0.500000    | 0.805556  | 1.0    | 0.892308 | 0.902778 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_carpet_on_mvtec_transistor | 0.500000    | 0.395604  | 1.0    | 0.566929 | 0.697802 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_carpet_on_mvtec_zipper     | 0.495536    | 0.805755  | 1.0    | 0.892430 | 0.901307 | 1.000000  | [[0, 27], [0, 112]] |

Table A.2: Cross-testing results with model trained on *cable* in the 5 selected test classes.

| Classe                          | AUROC  | Precision | Recall | F1-Score | PR AUC | Threshold | Confusion Matrix                                  |
|---------------------------------|--------|-----------|--------|----------|--------|-----------|---------------------------------------------------|
| mvtec_cable_on_mvtec_pill       | 0.4693 | 0.8645    | 1.0000 | 0.9273   | 0.8643 | 0.999896  | $\begin{bmatrix} 134 & 0 \\ 21 & 0 \end{bmatrix}$ |
| mvtec_cable_on_mvtec_transistor | 0.3379 | 0.3956    | 1.0000 | 0.5669   | 0.3513 | 0.999995  | $\begin{bmatrix} 36 & 0 \\ 55 & 0 \end{bmatrix}$  |
| mvtec_cable_on_mvtec_metal_nut  | 0.3219 | 0.8396    | 1.0000 | 0.9128   | 0.7649 | 0.999983  | $\begin{bmatrix} 89 & 0 \\ 17 & 0 \end{bmatrix}$  |
| mvtec_cable_on_mvtec_tile       | 0.6811 | 0.7453    | 1.0000 | 0.8541   | 0.8755 | 0.999876  | $\begin{bmatrix} 79 & 0 \\ 27 & 1 \end{bmatrix}$  |
| mvtec_cable_on_mvtec_wood       | 0.8662 | 0.8182    | 0.9818 | 0.8926   | 0.9666 | 0.999999  | $\begin{bmatrix} 54 & 1 \\ 12 & 2 \end{bmatrix}$  |

Table A.3: Cross-testing results with model trained on *cable*.

| Row Names                       | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|---------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_cable_on_mvtec_carpet     | 0.7940      | 0.7850    | 1.0000 | 0.8796   | 0.9443 | 0.999995  | [[0, 23], [0, 84]]  |
| mvtec_cable_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_cable_on_mvtec_leather    | 0.3851      | 0.7632    | 1.0000 | 0.8657   | 0.8139 | 0.999999  | [[0, 27], [0, 87]]  |
| mvtec_cable_on_mvtec_tile       | 0.6811      | 0.7453    | 1.0000 | 0.8541   | 0.8755 | 0.999876  | [[1, 27], [0, 79]]  |
| mvtec_cable_on_mvtec_wood       | 0.8662      | 0.8182    | 0.9818 | 0.8926   | 0.9666 | 0.999999  | [[2, 12], [1, 54]]  |
| mvtec_cable_on_mvtec_bottle     | 0.5417      | 0.8000    | 1.0000 | 0.8889   | 0.9083 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_cable_on_mvtec_cable      | 0.9978      | 0.9655    | 1.0000 | 0.9825   | 0.9986 | 0.577380  | [[50, 3], [0, 84]]  |
| mvtec_cable_on_mvtec_capsule    | 0.4856      | 0.8525    | 1.0000 | 0.9204   | 0.9222 | 0.999999  | [[0, 18], [0, 104]] |
| mvtec_cable_on_mvtec_hazelnut   | 0.3870      | 0.6535    | 1.0000 | 0.7904   | 0.6091 | 0.999980  | [[0, 35], [0, 66]]  |
| mvtec_cable_on_mvtec_metal_nut  | 0.3219      | 0.8396    | 1.0000 | 0.9128   | 0.7649 | 0.999983  | [[0, 17], [0, 89]]  |
| mvtec_cable_on_mvtec_pill       | 0.4693      | 0.8645    | 1.0000 | 0.9273   | 0.8643 | 0.999896  | [[0, 21], [0, 134]] |
| mvtec_cable_on_mvtec_screw      | 0.5804      | 0.7600    | 1.0000 | 0.8636   | 0.8889 | 0.999999  | [[0, 36], [0, 114]] |
| mvtec_cable_on_mvtec_toothbrush | 0.2291      | 0.8056    | 1.0000 | 0.8923   | 0.7209 | 0.999982  | [[0, 7], [0, 29]]   |
| mvtec_cable_on_mvtec_transistor | 0.3379      | 0.3956    | 1.0000 | 0.5669   | 0.3513 | 0.999995  | [[0, 55], [0, 36]]  |
| mvtec_cable_on_mvtec_zipper     | 0.4688      | 0.8058    | 1.0000 | 0.8924   | 0.8917 | 0.999999  | [[0, 27], [0, 112]] |

Table A.4: Cross-testing results with model trained on *capsule*.

| Row Names                         | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|-----------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_capsule_on_mvtec_carpet     | 0.4940      | 0.7850    | 1.0000 | 0.8796   | 0.8902 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_capsule_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_capsule_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_capsule_on_mvtec_tile       | 0.5703      | 0.7383    | 1.0000 | 0.8495   | 0.8591 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_capsule_on_mvtec_wood       | 0.7435      | 0.8571    | 0.9818 | 0.9153   | 0.9291 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_capsule_on_mvtec_bottle     | 0.4917      | 0.8000    | 1.0000 | 0.8889   | 0.8970 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_capsule_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_capsule_on_mvtec_capsule    | 0.9984      | 0.9811    | 1.0000 | 0.9905   | 0.9997 | 0.537952  | [[16, 2], [0, 104]] |
| mvtec_capsule_on_mvtec_hazelnut   | 0.5485      | 0.6633    | 0.9848 | 0.7927   | 0.8210 | 0.999999  | [[2, 33], [1, 65]]  |
| mvtec_capsule_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_capsule_on_mvtec_pill       | 0.5075      | 0.8645    | 1.0000 | 0.9273   | 0.8938 | 0.999990  | [[0, 21], [0, 134]] |
| mvtec_capsule_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_capsule_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_capsule_on_mvtec_transistor | 0.4722      | 0.3956    | 1.0000 | 0.5669   | 0.6742 | 0.999998  | [[0, 55], [0, 36]]  |
| mvtec_capsule_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.5: Cross-testing results with model trained on *hazelnut*.

| Row Names                          | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|------------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_hazelnut_on_mvtec_carpet     | 0.5413      | 0.7850    | 1.0000 | 0.8796   | 0.8610 | 1.000000  | [[0, 23], [0, 84]]  |
| mvtec_hazelnut_on_mvtec_grid       | 0.4971      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_hazelnut_on_mvtec_leather    | 0.4935      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_hazelnut_on_mvtec_tile       | 0.6876      | 0.7383    | 1.0000 | 0.8495   | 0.8622 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_hazelnut_on_mvtec_wood       | 0.8138      | 0.8571    | 0.9818 | 0.9153   | 0.9284 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_hazelnut_on_mvtec_bottle     | 0.4875      | 0.8000    | 1.0000 | 0.8889   | 0.8989 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_hazelnut_on_mvtec_cable      | 0.4866      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_hazelnut_on_mvtec_capsule    | 0.4971      | 0.8525    | 1.0000 | 0.9204   | 0.9223 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_hazelnut_on_mvtec_hazelnut   | 0.9999      | 1.0000    | 1.0000 | 1.0000   | 1.0000 | 0.655527  | [[19, 0], [0, 66]]  |
| mvtec_hazelnut_on_mvtec_metal_nut  | 0.4794      | 0.8396    | 1.0000 | 0.9128   | 0.9145 | 0.999998  | [[0, 17], [0, 89]]  |
| mvtec_hazelnut_on_mvtec_pill       | 0.4469      | 0.8645    | 1.0000 | 0.9273   | 0.9300 | 0.999999  | [[0, 21], [0, 134]] |
| mvtec_hazelnut_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_hazelnut_on_mvtec_toothbrush | 0.4685      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_hazelnut_on_mvtec_transistor | 0.3911      | 0.3956    | 1.0000 | 0.5669   | 0.6637 | 0.999997  | [[0, 55], [0, 36]]  |
| mvtec_hazelnut_on_mvtec_zipper     | 0.4465      | 0.8058    | 1.0000 | 0.8924   | 0.9025 | 1.000000  | [[0, 27], [0, 112]] |

Table A.6: Cross-testing results with model trained on *grid*.

| Row Names                      | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_grid_on_mvtec_carpet     | 0.5993      | 0.7850    | 1.0000 | 0.8796   | 0.9093 | 1.000000  | [[0, 23], [0, 84]]  |
| mvtec_grid_on_mvtec_grid       | 0.9988      | 0.9870    | 1.0000 | 0.9934   | 0.9997 | 0.609611  | [[11, 1], [0, 52]]  |
| mvtec_grid_on_mvtec_leather    | 0.5755      | 0.7632    | 1.0000 | 0.8657   | 0.8703 | 0.999999  | [[0, 27], [0, 87]]  |
| mvtec_grid_on_mvtec_tile       | 0.7635      | 0.7383    | 1.0000 | 0.8495   | 0.8730 | 0.999997  | [[0, 28], [0, 79]]  |
| mvtec_grid_on_mvtec_wood       | 0.8521      | 0.8571    | 0.9818 | 0.9153   | 0.9277 | 0.999997  | [[5, 9], [1, 54]]   |
| mvtec_grid_on_mvtec_bottle     | 0.5632      | 0.8000    | 1.0000 | 0.8889   | 0.9105 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_grid_on_mvtec_cable      | 0.5415      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 0.999999  | [[0, 53], [0, 84]]  |
| mvtec_grid_on_mvtec_capsule    | 0.5842      | 0.8525    | 1.0000 | 0.9204   | 0.9256 | 0.999999  | [[0, 18], [0, 104]] |
| mvtec_grid_on_mvtec_hazelnut   | 0.5859      | 0.6633    | 0.9848 | 0.7927   | 0.8227 | 0.999999  | [[2, 33], [1, 65]]  |
| mvtec_grid_on_mvtec_metal_nut  | 0.5171      | 0.8396    | 1.0000 | 0.9128   | 0.9234 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_grid_on_mvtec_pill       | 0.5305      | 0.8645    | 1.0000 | 0.9273   | 0.9358 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_grid_on_mvtec_screw      | 0.5239      | 0.7600    | 1.0000 | 0.8636   | 0.8889 | 0.999999  | [[0, 36], [0, 114]] |
| mvtec_grid_on_mvtec_toothbrush | 0.5021      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 0.999999  | [[0, 7], [0, 29]]   |
| mvtec_grid_on_mvtec_transistor | 0.4095      | 0.3956    | 1.0000 | 0.5669   | 0.6794 | 0.999999  | [[0, 55], [0, 36]]  |
| mvtec_grid_on_mvtec_zipper     | 0.4969      | 0.8058    | 1.0000 | 0.8924   | 0.9025 | 0.999999  | [[0, 27], [0, 112]] |

Table A.7: Cross-testing results with model trained on *leather*.

| Row Names                         | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|-----------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_leather_on_mvtec_carpet     | 0.4971      | 0.7850    | 1.0000 | 0.8796   | 0.8876 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_leather_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_leather_on_mvtec_leather    | 0.9983      | 0.9600    | 1.0000 | 0.9796   | 0.9992 | 0.556850  | [[24, 3], [0, 87]]  |
| mvtec_leather_on_mvtec_tile       | 0.7232      | 0.7383    | 1.0000 | 0.8495   | 0.8670 | 0.999998  | [[0, 28], [0, 79]]  |
| mvtec_leather_on_mvtec_wood       | 0.8324      | 0.8571    | 0.9818 | 0.9153   | 0.9273 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_leather_on_mvtec_bottle     | 0.4965      | 0.8000    | 1.0000 | 0.8889   | 0.8997 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_leather_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 0.999999  | [[0, 53], [0, 84]]  |
| mvtec_leather_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9226 | 0.999999  | [[0, 18], [0, 104]] |
| mvtec_leather_on_mvtec_hazelnut   | 0.5454      | 0.6633    | 0.9848 | 0.7927   | 0.8262 | 0.999999  | [[2, 33], [1, 65]]  |
| mvtec_leather_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9197 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_leather_on_mvtec_pill       | 0.5074      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 0.999999  | [[0, 21], [0, 134]] |
| mvtec_leather_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_leather_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_leather_on_mvtec_transistor | 0.4936      | 0.3956    | 1.0000 | 0.5669   | 0.6776 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_leather_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.8: Cross-testing results with model trained on *metal\_nut*.

| Row Names                           | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|-------------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_metal_nut_on_mvtec_carpet     | 0.4967      | 0.7850    | 1.0000 | 0.8796   | 0.8897 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_metal_nut_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_metal_nut_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_metal_nut_on_mvtec_tile       | 0.6092      | 0.7383    | 1.0000 | 0.8495   | 0.8611 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_metal_nut_on_mvtec_wood       | 0.7336      | 0.8571    | 0.9818 | 0.9153   | 0.9284 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_metal_nut_on_mvtec_bottle     | 0.4954      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 0.999999  | [[0, 15], [0, 60]]  |
| mvtec_metal_nut_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 0.999999  | [[0, 53], [0, 84]]  |
| mvtec_metal_nut_on_mvtec_capsule    | 0.4931      | 0.8525    | 1.0000 | 0.9204   | 0.9221 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_metal_nut_on_mvtec_hazelnut   | 0.5140      | 0.6633    | 0.9848 | 0.7927   | 0.8235 | 0.999999  | [[2, 33], [1, 65]]  |
| mvtec_metal_nut_on_mvtec_metal_nut  | 0.9978      | 0.9778    | 1.0000 | 0.9888   | 0.9994 | 0.634705  | [[18, 2], [0, 89]]  |
| mvtec_metal_nut_on_mvtec_pill       | 0.4830      | 0.8645    | 1.0000 | 0.9273   | 0.9332 | 0.999999  | [[0, 21], [0, 134]] |
| mvtec_metal_nut_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_metal_nut_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_metal_nut_on_mvtec_transistor | 0.4934      | 0.3956    | 1.0000 | 0.5669   | 0.6703 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_metal_nut_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.9: Cross-testing results with model trained on *pill*.

| Row Names                      | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_pill_on_mvtec_carpet     | 0.5000      | 0.7850    | 1.0000 | 0.8796   | 0.8885 | 1.000000  | [[0, 23], [0, 84]]  |
| mvtec_pill_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_pill_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_pill_on_mvtec_tile       | 0.6215      | 0.7383    | 1.0000 | 0.8495   | 0.8623 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_pill_on_mvtec_wood       | 0.8371      | 0.8571    | 0.9818 | 0.9153   | 0.9305 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_pill_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_pill_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_pill_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9260 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_pill_on_mvtec_hazelnut   | 0.5000      | 0.6633    | 0.9848 | 0.7927   | 0.8262 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_pill_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_pill_on_mvtec_pill       | 0.9983      | 0.9772    | 1.0000 | 0.9884   | 0.9994 | 0.689336  | [[29, 3], [0, 134]] |
| mvtec_pill_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_pill_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_pill_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6699 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_pill_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.10: Cross-testing results with model trained on *screw*.

| Row Names                       | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|---------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_screw_on_mvtec_carpet     | 0.6048      | 0.7850    | 1.0000 | 0.8796   | 0.9124 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_screw_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_screw_on_mvtec_leather    | 0.5017      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_screw_on_mvtec_tile       | 0.7660      | 0.7383    | 1.0000 | 0.8495   | 0.8731 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_screw_on_mvtec_wood       | 0.8329      | 0.8571    | 0.9818 | 0.9153   | 0.9308 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_screw_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_screw_on_mvtec_cable      | 0.5034      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 0.999999  | [[0, 53], [0, 84]]  |
| mvtec_screw_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9226 | 0.999999  | [[0, 18], [0, 104]] |
| mvtec_screw_on_mvtec_hazelnut   | 0.5018      | 0.6633    | 0.9848 | 0.7927   | 0.8217 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_screw_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_screw_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_screw_on_mvtec_screw      | 0.9987      | 0.9811    | 1.0000 | 0.9905   | 0.9997 | 0.635371  | [[37, 1], [0, 114]] |
| mvtec_screw_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_screw_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6702 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_screw_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.11: Cross-testing results with model trained on *tile*.

| Row Names                      | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_tile_on_mvtec_carpet     | 0.6066      | 0.7850    | 1.0000 | 0.8796   | 0.9001 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_tile_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_tile_on_mvtec_leather    | 0.4945      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_tile_on_mvtec_tile       | 0.9979      | 0.9861    | 1.0000 | 0.9930   | 0.9996 | 0.609288  | [[22, 1], [0, 79]]  |
| mvtec_tile_on_mvtec_wood       | 0.8445      | 0.8571    | 0.9818 | 0.9153   | 0.9307 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_tile_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_tile_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_tile_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9232 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_tile_on_mvtec_hazelnut   | 0.5033      | 0.6633    | 0.9848 | 0.7927   | 0.8222 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_tile_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9197 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_tile_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_tile_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_tile_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_tile_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6697 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_tile_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.12: Cross-testing results with model trained on *toothbrush*.

| Row Names                            | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_toothbrush_on_mvtec_carpet     | 0.5306      | 0.7850    | 1.0000 | 0.8796   | 0.8994 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_toothbrush_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_toothbrush_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_toothbrush_on_mvtec_tile       | 0.5890      | 0.7383    | 1.0000 | 0.8495   | 0.8640 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_toothbrush_on_mvtec_wood       | 0.8242      | 0.8571    | 0.9818 | 0.9153   | 0.9293 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_toothbrush_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_toothbrush_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_toothbrush_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9234 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_toothbrush_on_mvtec_hazelnut   | 0.5000      | 0.6633    | 0.9848 | 0.7927   | 0.8256 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_toothbrush_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9197 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_toothbrush_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_toothbrush_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_toothbrush_on_mvtec_toothbrush | 0.9999      | 1.0000    | 1.0000 | 1.0000   | 1.0000 | 0.657055  | [[7, 0], [0, 29]]   |
| mvtec_toothbrush_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6703 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_toothbrush_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.13: Cross-testing results with model trained on *transistor*.

| Row Names                            | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_transistor_on_mvtec_carpet     | 0.5000      | 0.7850    | 1.0000 | 0.8796   | 0.9004 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_transistor_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_transistor_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_transistor_on_mvtec_tile       | 0.6775      | 0.7383    | 1.0000 | 0.8495   | 0.8675 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_transistor_on_mvtec_wood       | 0.8612      | 0.8571    | 0.9818 | 0.9153   | 0.9314 | 0.999999  | [[5, 9], [1, 54]]   |
| mvtec_transistor_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_transistor_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_transistor_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9233 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_transistor_on_mvtec_hazelnut   | 0.5027      | 0.6633    | 0.9848 | 0.7927   | 0.8241 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_transistor_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_transistor_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_transistor_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_transistor_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_transistor_on_mvtec_transistor | 0.9995      | 0.9818    | 1.0000 | 0.9908   | 0.9998 | 0.629734  | [[54, 1], [0, 36]]  |
| mvtec_transistor_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.14: Cross-testing results with model trained on *wood*.

| Row Names                      | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|--------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_wood_on_mvtec_carpet     | 0.5313      | 0.7850    | 1.0000 | 0.8796   | 0.8969 | 0.999999  | [[0, 23], [0, 84]]  |
| mvtec_wood_on_mvtec_grid       | 0.5000      | 0.7647    | 1.0000 | 0.8667   | 0.8824 | 1.000000  | [[0, 16], [0, 52]]  |
| mvtec_wood_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_wood_on_mvtec_tile       | 0.7713      | 0.7383    | 1.0000 | 0.8495   | 0.8706 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_wood_on_mvtec_wood       | 0.9984      | 0.9867    | 1.0000 | 0.9933   | 0.9997 | 0.615217  | [[13, 1], [0, 54]]  |
| mvtec_wood_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_wood_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_wood_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9231 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_wood_on_mvtec_hazelnut   | 0.5000      | 0.6633    | 0.9848 | 0.7927   | 0.8249 | 1.000000  | [[2, 33], [1, 65]]  |
| mvtec_wood_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_wood_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9337 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_wood_on_mvtec_screw      | 0.5000      | 0.7600    | 1.0000 | 0.8636   | 0.8800 | 1.000000  | [[0, 36], [0, 114]] |
| mvtec_wood_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_wood_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6703 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_wood_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.15: Cross-testing results with model trained on *bottle*.

| Row Names                        | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|----------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_bottle_on_mvtec_carpet     | 0.5000      | 0.7850    | 1.0000 | 0.8796   | 0.8925 | 1.000000  | [[0, 23], [0, 84]]  |
| mvtec_bottle_on_mvtec_grid       | 0.5313      | 0.7761    | 1.0000 | 0.8739   | 0.8881 | 1.000000  | [[1, 15], [0, 52]]  |
| mvtec_bottle_on_mvtec_leather    | 0.5000      | 0.7632    | 1.0000 | 0.8657   | 0.8816 | 1.000000  | [[0, 27], [0, 87]]  |
| mvtec_bottle_on_mvtec_tile       | 0.5000      | 0.7383    | 1.0000 | 0.8495   | 0.8692 | 1.000000  | [[0, 28], [0, 79]]  |
| mvtec_bottle_on_mvtec_wood       | 0.5000      | 0.7971    | 1.0000 | 0.8871   | 0.8986 | 1.000000  | [[0, 14], [0, 55]]  |
| mvtec_bottle_on_mvtec_bottle     | 1.0000      | 1.0000    | 1.0000 | 1.0000   | 1.0000 | 0.949637  | [[15, 0], [0, 60]]  |
| mvtec_bottle_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_bottle_on_mvtec_capsule    | 0.5230      | 0.8525    | 1.0000 | 0.9204   | 0.9285 | 0.999999  | [[0, 18], [0, 104]] |
| mvtec_bottle_on_mvtec_hazelnut   | 0.5000      | 0.6535    | 1.0000 | 0.7904   | 0.8267 | 1.000000  | [[0, 35], [0, 66]]  |
| mvtec_bottle_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_bottle_on_mvtec_pill       | 0.4646      | 0.8645    | 1.0000 | 0.9273   | 0.9090 | 0.999999  | [[0, 21], [0, 134]] |
| mvtec_bottle_on_mvtec_screw      | 0.4956      | 0.7600    | 1.0000 | 0.8636   | 0.8781 | 0.999999  | [[0, 36], [0, 114]] |
| mvtec_bottle_on_mvtec_toothbrush | 0.4138      | 0.8056    | 1.0000 | 0.8923   | 0.8703 | 0.999999  | [[0, 7], [0, 29]]   |
| mvtec_bottle_on_mvtec_transistor | 0.4856      | 0.3956    | 1.0000 | 0.5669   | 0.6689 | 0.999999  | [[0, 55], [0, 36]]  |
| mvtec_bottle_on_mvtec_zipper     | 0.5000      | 0.8058    | 1.0000 | 0.8924   | 0.9029 | 1.000000  | [[0, 27], [0, 112]] |

Table A.16: Cross-testing results with model trained on *zipper*.

| Row Names                        | image_auroc | precision | recall | f1_score | pr_auc | threshold | confusion_matrix    |
|----------------------------------|-------------|-----------|--------|----------|--------|-----------|---------------------|
| mvtec_zipper_on_mvtec_carpet     | 0.3600      | 0.7850    | 1.0000 | 0.8796   | 0.7499 | 0.999729  | [[0, 23], [0, 84]]  |
| mvtec_zipper_on_mvtec_grid       | 0.6010      | 0.8033    | 0.9423 | 0.8673   | 0.8325 | 0.999999  | [[4, 12], [3, 49]]  |
| mvtec_zipper_on_mvtec_leather    | 0.4860      | 0.7632    | 1.0000 | 0.8657   | 0.8129 | 0.999996  | [[0, 27], [0, 87]]  |
| mvtec_zipper_on_mvtec_tile       | 0.7129      | 0.7383    | 1.0000 | 0.8495   | 0.8959 | 0.999999  | [[0, 28], [0, 79]]  |
| mvtec_zipper_on_mvtec_wood       | 0.4136      | 0.7971    | 1.0000 | 0.8871   | 0.8420 | 0.999999  | [[0, 14], [0, 55]]  |
| mvtec_zipper_on_mvtec_bottle     | 0.5000      | 0.8000    | 1.0000 | 0.8889   | 0.9000 | 1.000000  | [[0, 15], [0, 60]]  |
| mvtec_zipper_on_mvtec_cable      | 0.5000      | 0.6131    | 1.0000 | 0.7602   | 0.8066 | 1.000000  | [[0, 53], [0, 84]]  |
| mvtec_zipper_on_mvtec_capsule    | 0.5000      | 0.8525    | 1.0000 | 0.9204   | 0.9262 | 1.000000  | [[0, 18], [0, 104]] |
| mvtec_zipper_on_mvtec_hazelnut   | 0.5000      | 0.6535    | 1.0000 | 0.7904   | 0.8267 | 1.000000  | [[0, 35], [0, 66]]  |
| mvtec_zipper_on_mvtec_metal_nut  | 0.5000      | 0.8396    | 1.0000 | 0.9128   | 0.9198 | 1.000000  | [[0, 17], [0, 89]]  |
| mvtec_zipper_on_mvtec_pill       | 0.5000      | 0.8645    | 1.0000 | 0.9273   | 0.9323 | 1.000000  | [[0, 21], [0, 134]] |
| mvtec_zipper_on_mvtec_screw      | 0.5696      | 0.7600    | 1.0000 | 0.8636   | 0.8433 | 0.999999  | [[0, 36], [0, 114]] |
| mvtec_zipper_on_mvtec_toothbrush | 0.5000      | 0.8056    | 1.0000 | 0.8923   | 0.9028 | 1.000000  | [[0, 7], [0, 29]]   |
| mvtec_zipper_on_mvtec_transistor | 0.5000      | 0.3956    | 1.0000 | 0.5669   | 0.6978 | 1.000000  | [[0, 55], [0, 36]]  |
| mvtec_zipper_on_mvtec_zipper     | 0.9997      | 0.9912    | 1.0000 | 0.9956   | 0.9999 | 0.749881  | [[26, 1], [0, 112]] |



## Appendix B

# Transfer Learning plots

### B.1 Experiment 0

#### Pill

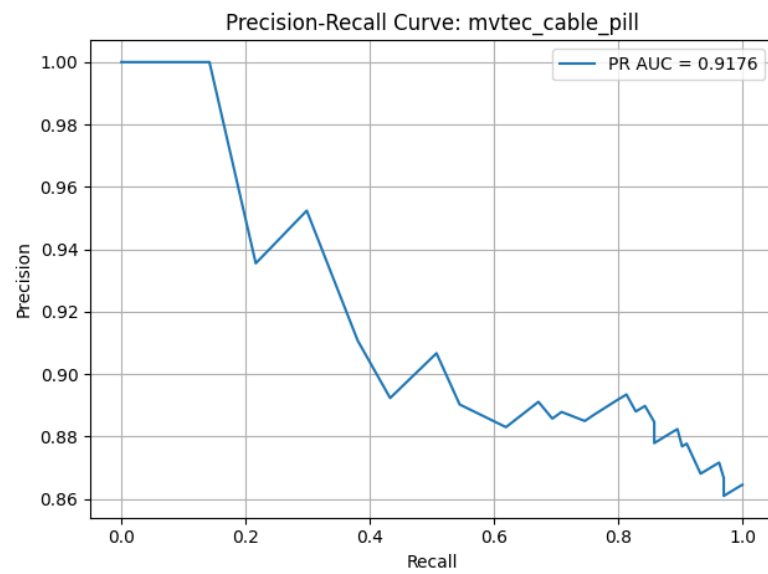
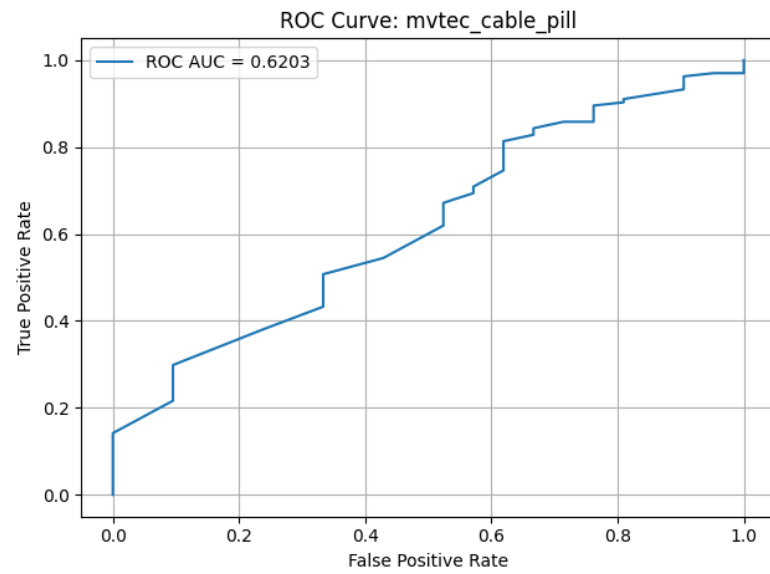
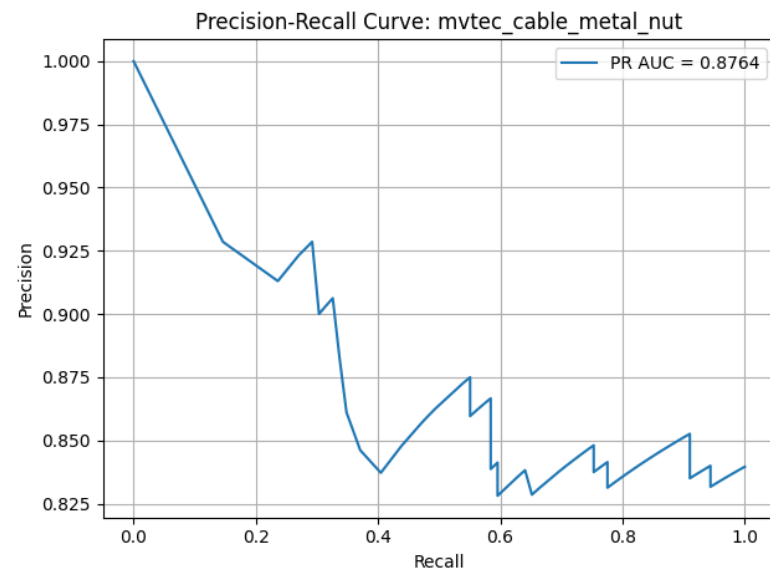


Figure B.1: Precision-Recall Curve for class *pill* in Experiment 0.

Figure B.2: ROC Curve for class *pill* in Experiment 0.**Metal\_nut**Figure B.3: Precision-Recall Curve for class *metal\_nut* in Experiment 0.

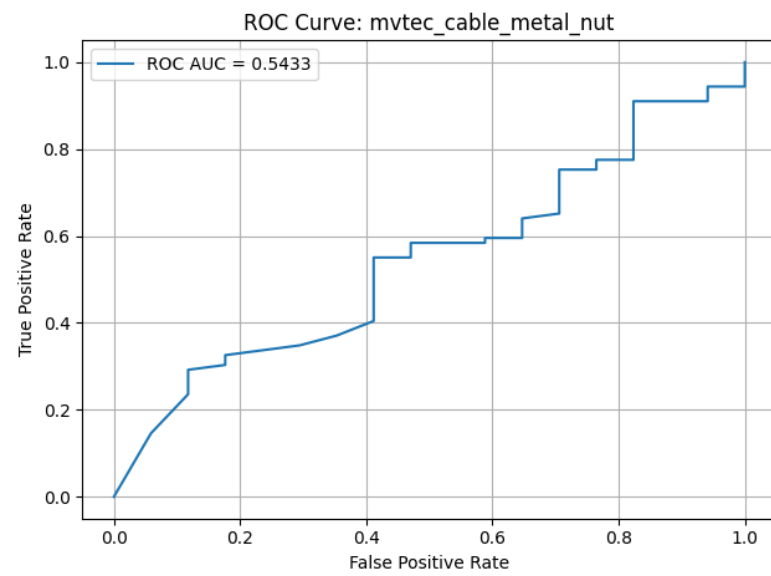


Figure B.4: ROC Curve for class *metal\_nut* in Experiment 0.

## Transistor

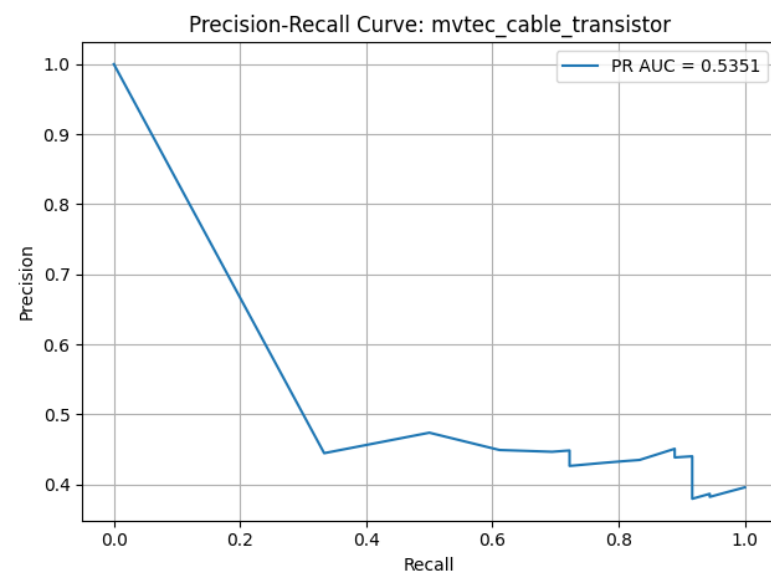
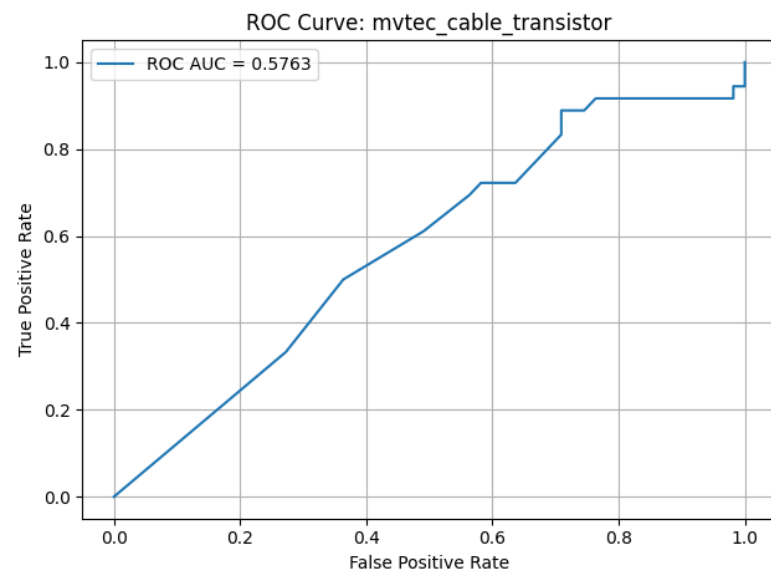
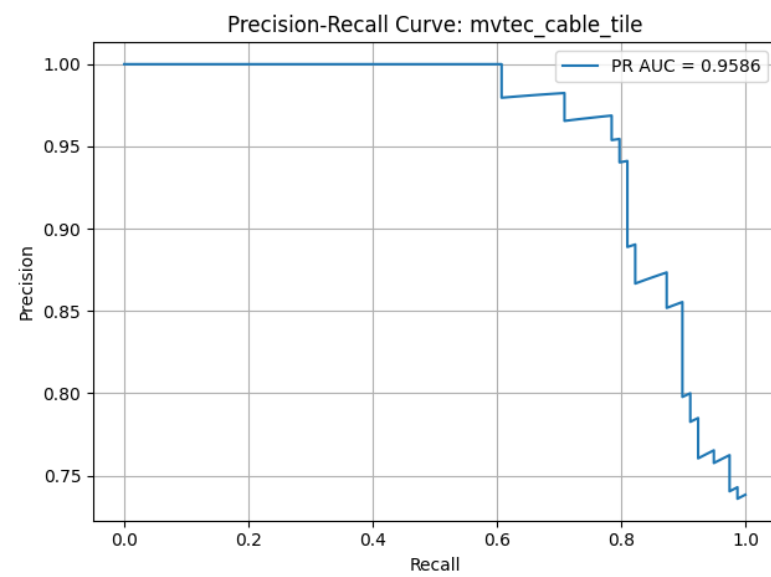


Figure B.5: Precision-Recall Curve for class *transistor* in Experiment 0.

Figure B.6: ROC Curve for class *transistor* in Experiment 0.

## Tile

Figure B.7: Precision-Recall Curve for class *tile* in Experiment 0.

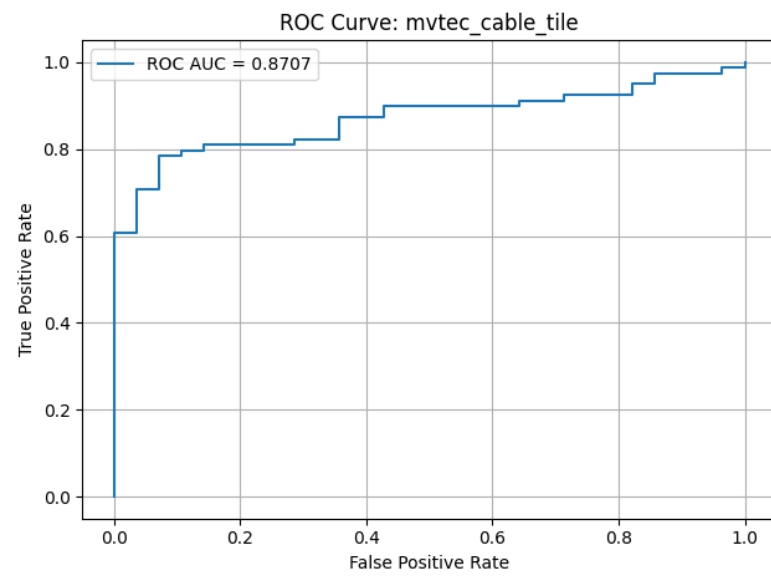


Figure B.8: ROC Curve for class *tile* in Experiment 0.

## Wood

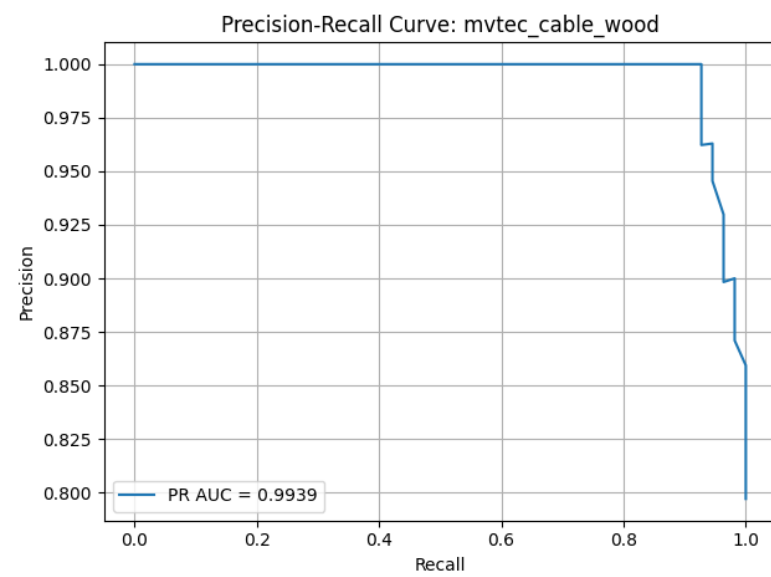
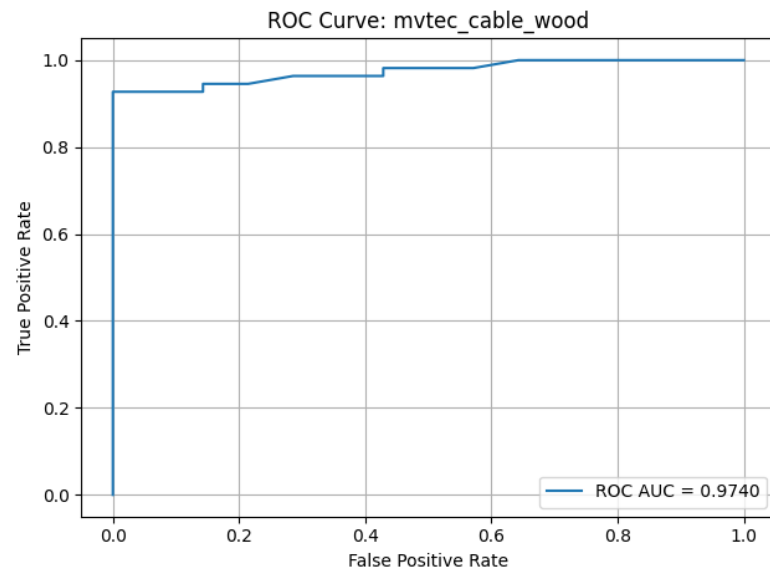
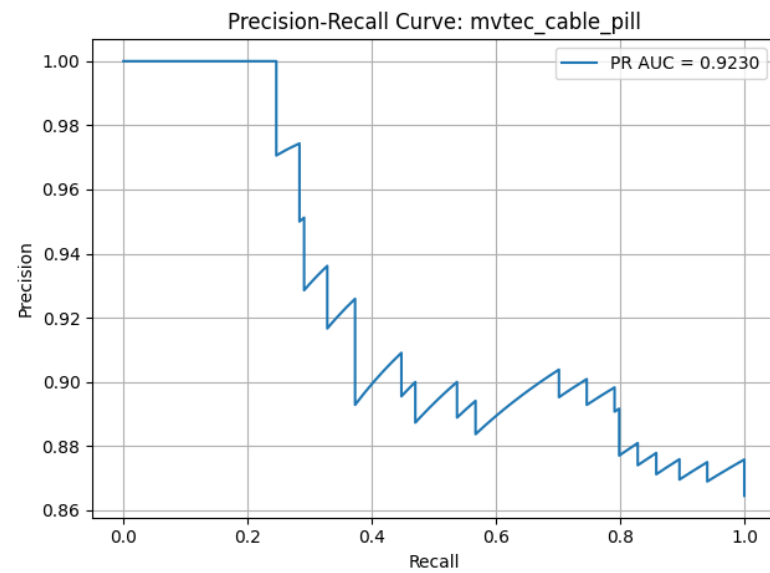


Figure B.9: Precision-Recall Curve for class *wood* in Experiment 0.

Figure B.10: ROC Curve for class *wood* in Experiment 0.

## B.2 Experiment 1

### Pill

Figure B.11: Precision-Recall Curve for class *pill* in Experiment 1.

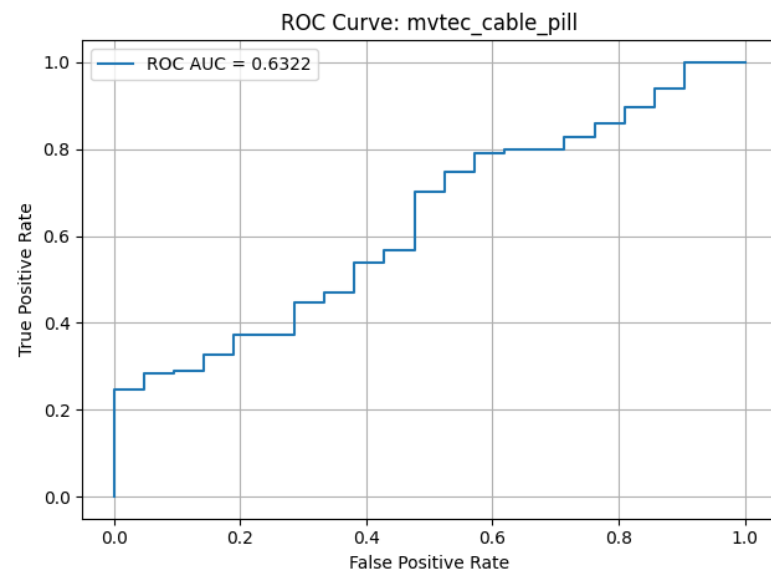


Figure B.12: ROC Curve for class *pill* in Experiment 1.

### Metal\_nut

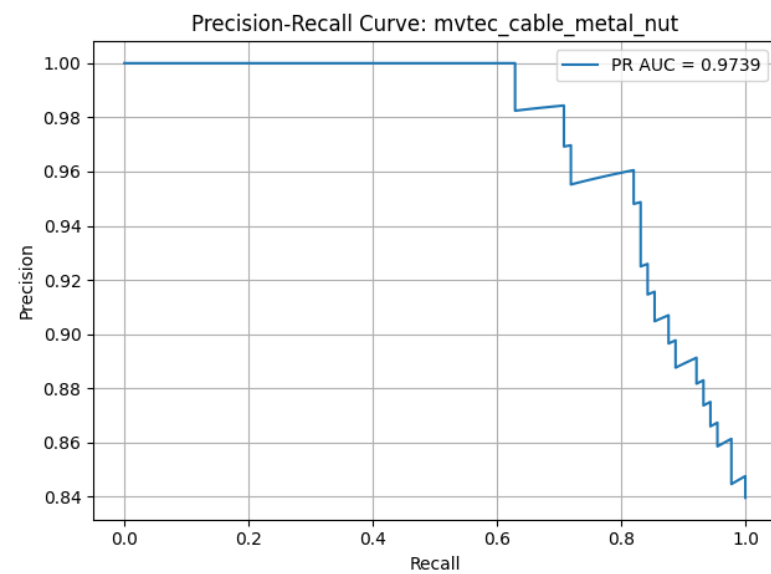
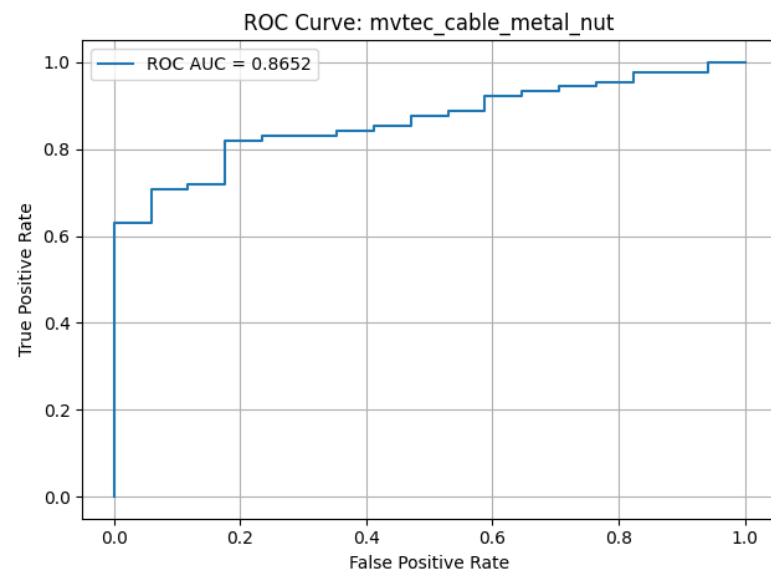
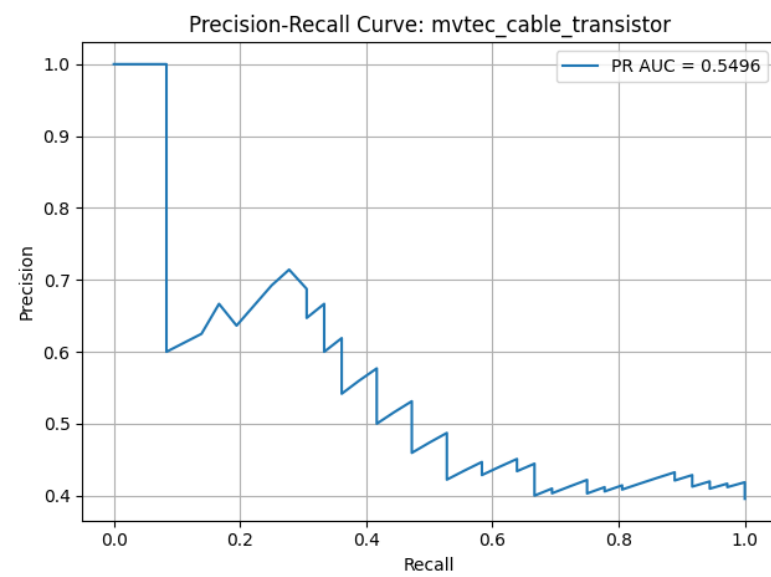


Figure B.13: Precision-Recall Curve for class *metal\_nut* in Experiment 1.

Figure B.14: ROC Curve for class *metal\_nut* in Experiment 1.

## Transistor

Figure B.15: Precision-Recall Curve for class *transistor* in Experiment 1.



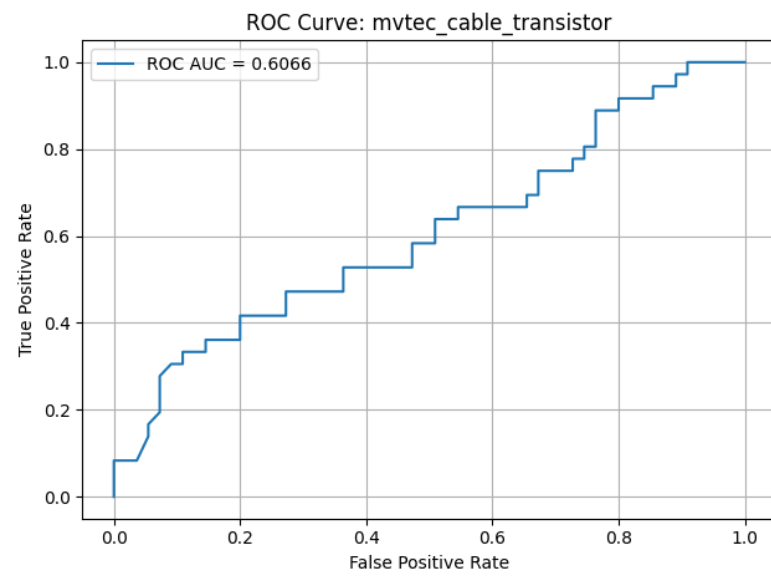


Figure B.16: ROC Curve for class *transistor* in Experiment 1.

## Tile

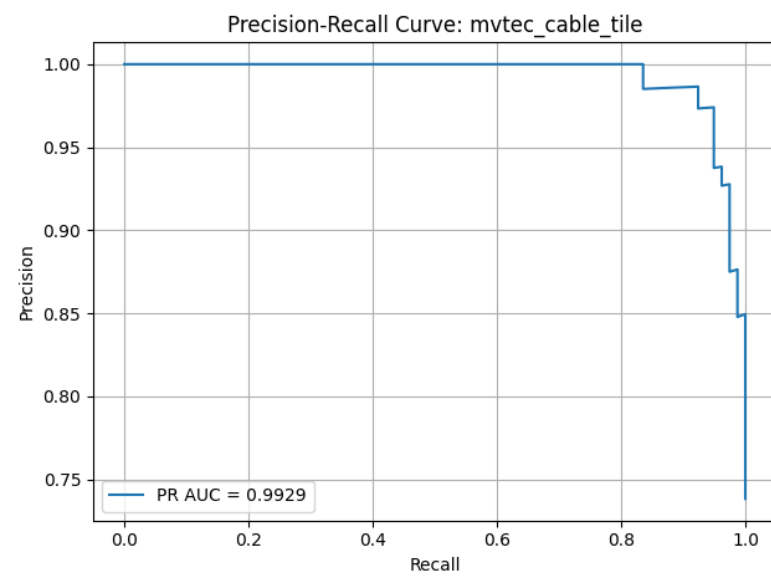
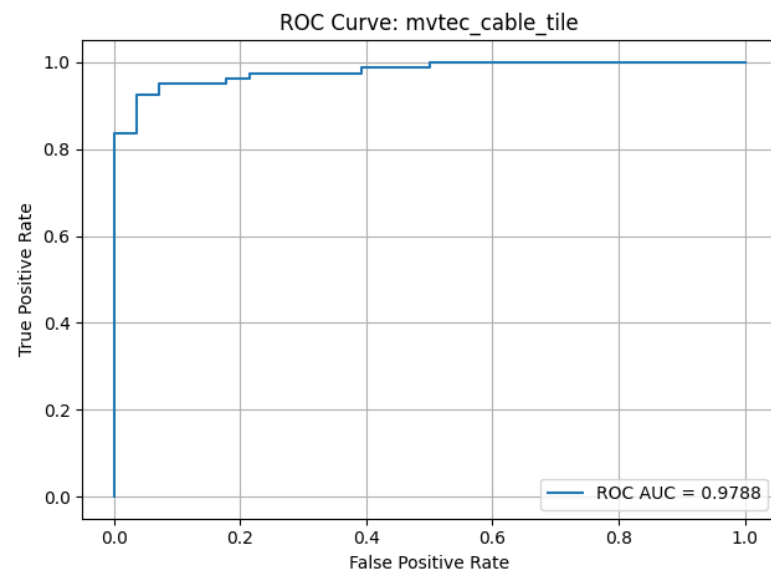
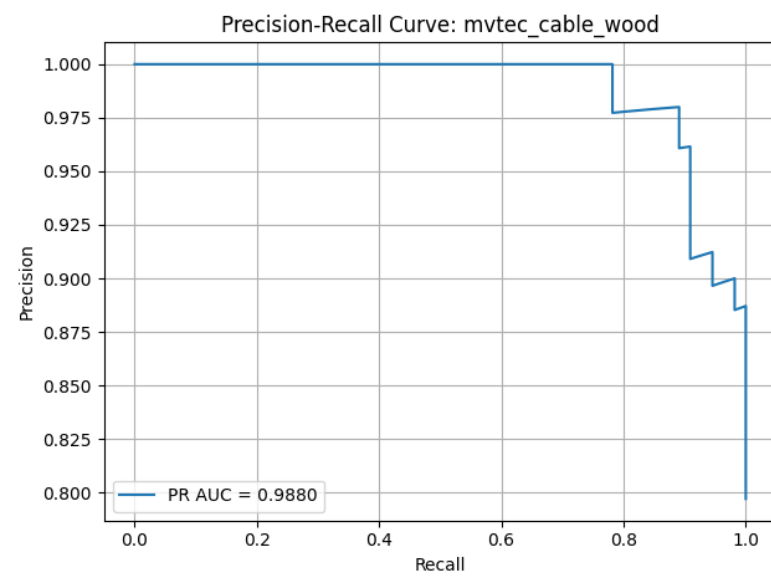


Figure B.17: Precision-Recall Curve for class *tile* in Experiment 1.

Figure B.18: ROC Curve for class *tile* in Experiment 1.

## Wood

Figure B.19: Precision-Recall Curve for class *wood* in Experiment 1.

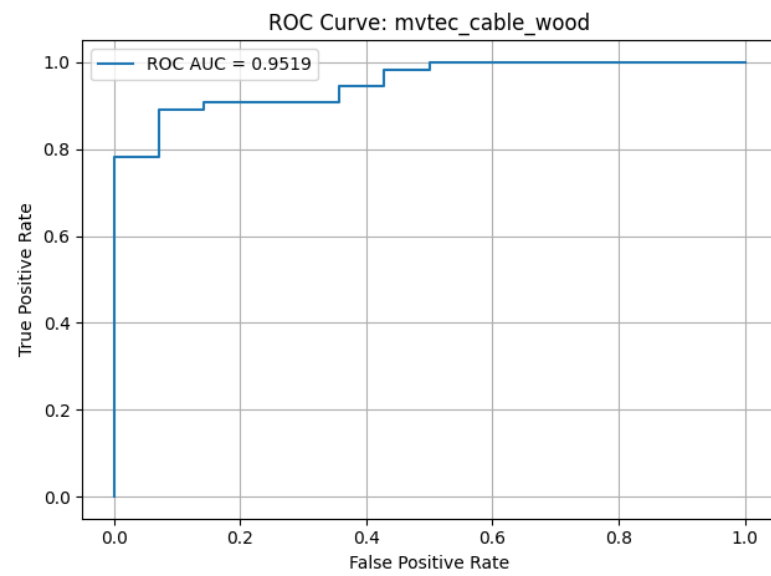


Figure B.20: ROC Curve for class *wood* in Experiment 1.

## B.3 Experiment 2

### Pill

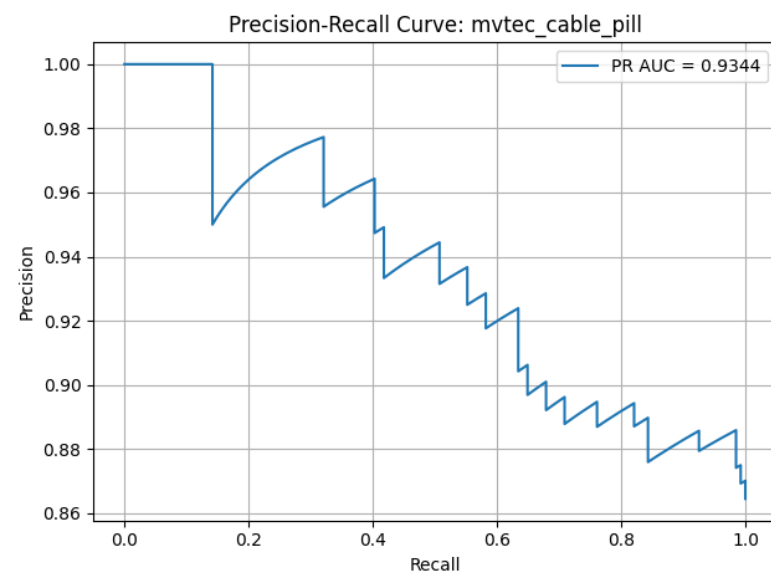
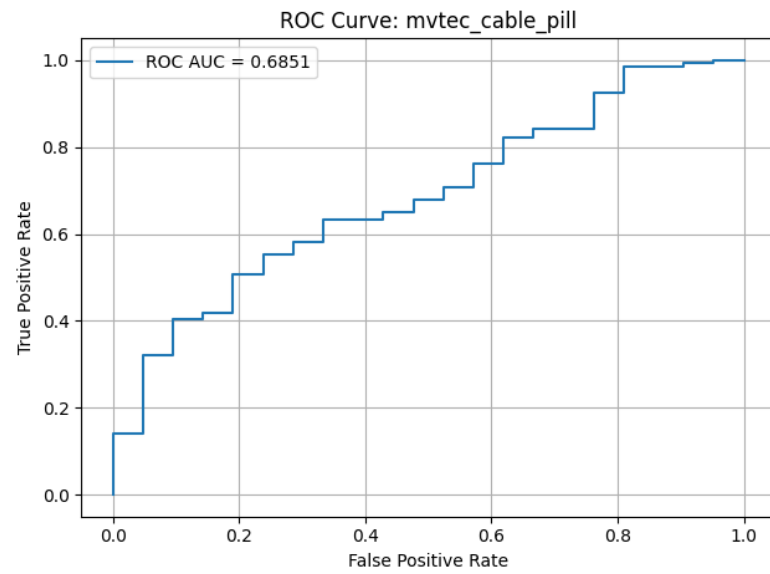
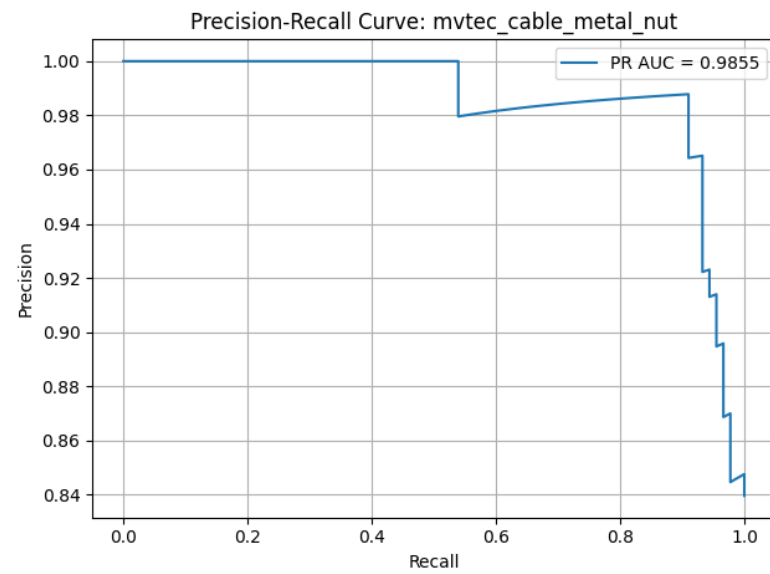


Figure B.21: Precision-Recall Curve for class *pill* in Experiment 2.

Figure B.22: ROC Curve for class *pill* in Experiment 2.**Metal\_nut**Figure B.23: Precision-Recall Curve for class *metal\_nut* in Experiment 2.

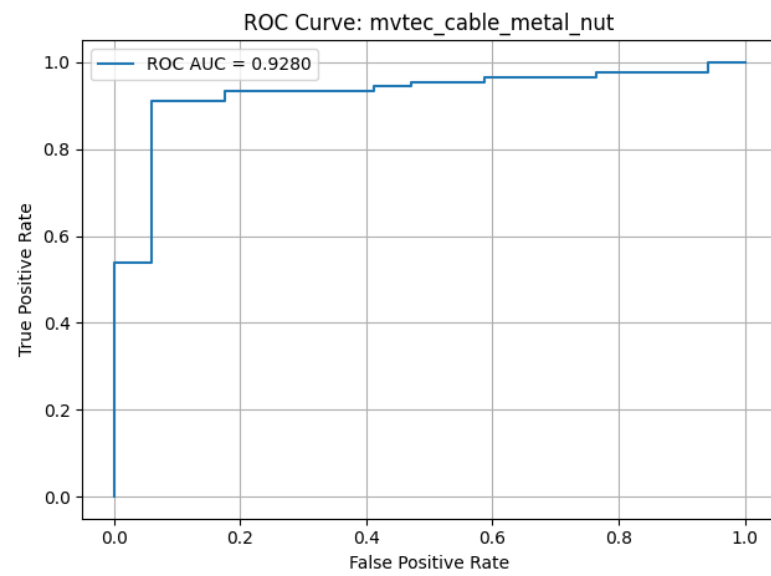


Figure B.24: ROC Curve for class *metal\_nut* in Experiment 2.

## Transistor

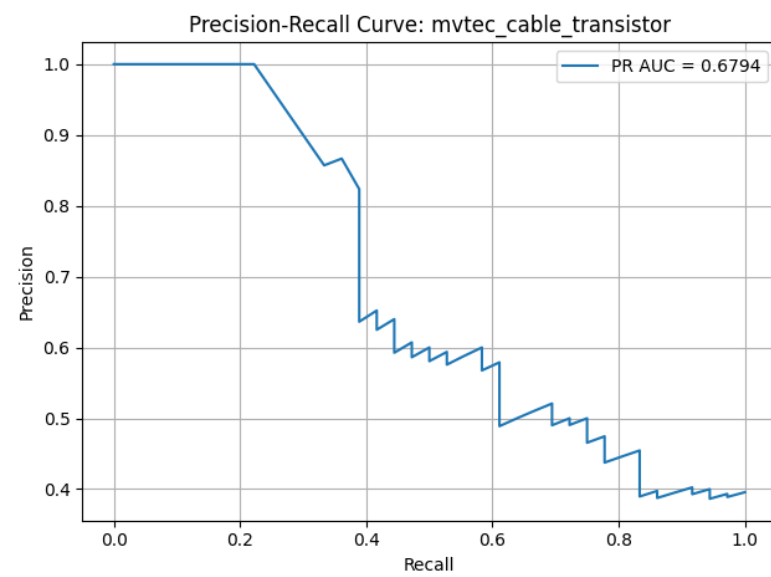
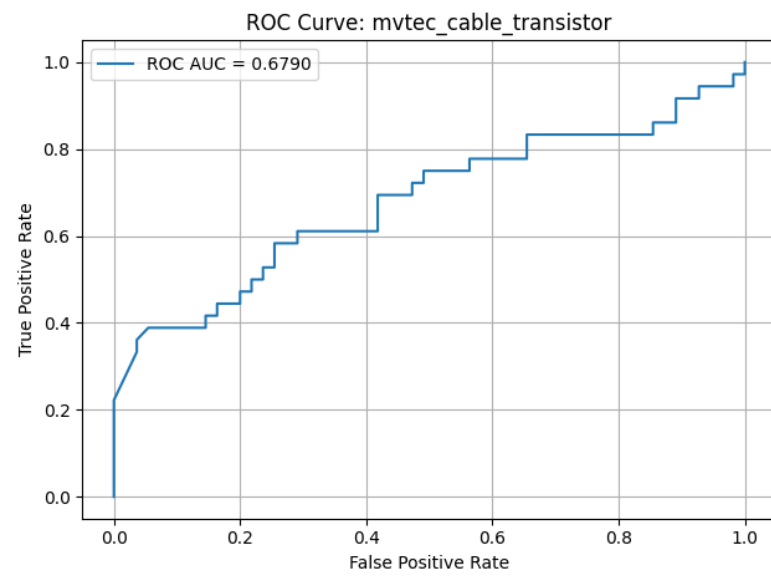
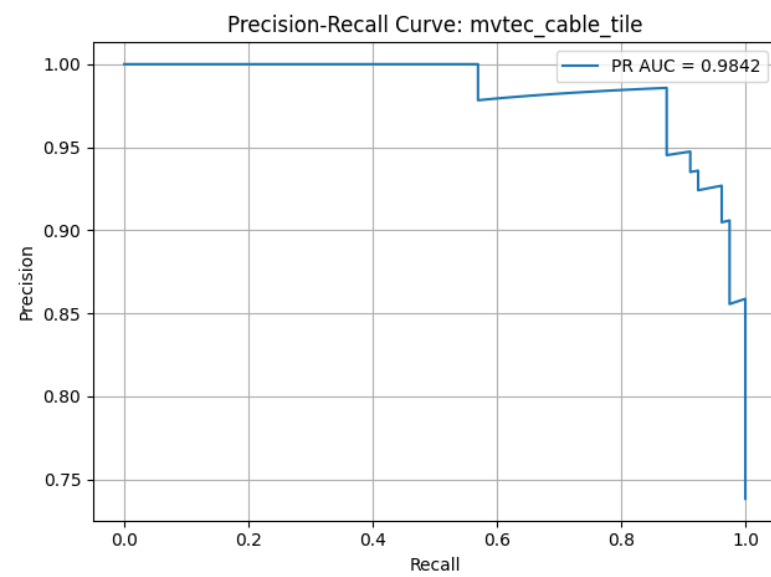
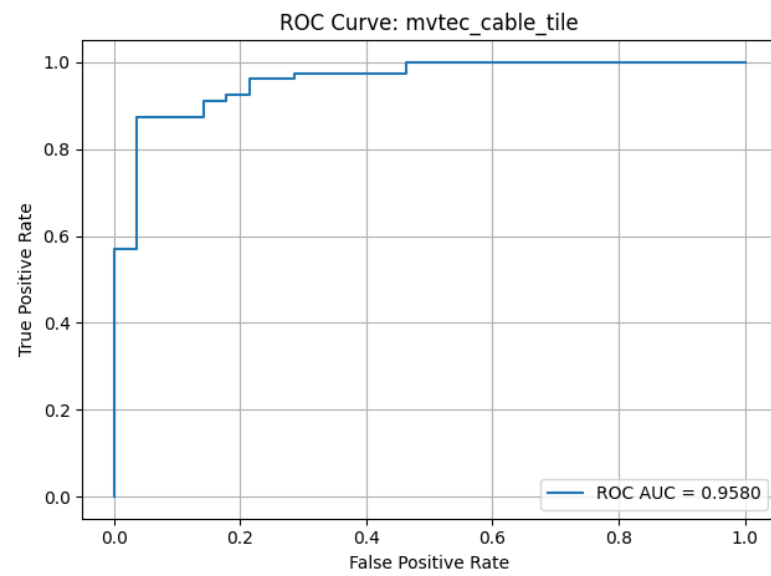


Figure B.25: Precision-Recall Curve for class *transistor* in Experiment 2.

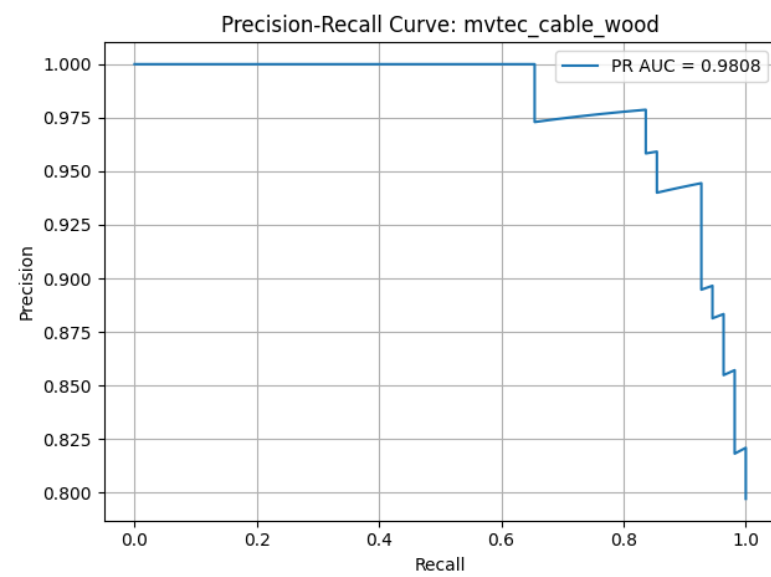
Figure B.26: ROC Curve for class *transistor* in Experiment 2.

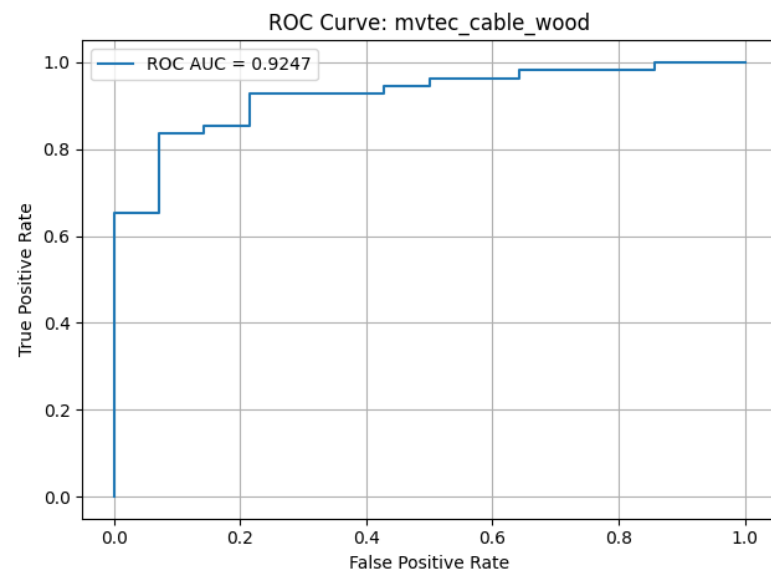
## Tile

Figure B.27: Precision-Recall Curve for class *tile* in Experiment 2.

Figure B.28: ROC Curve for class *tile* in Experiment 2.

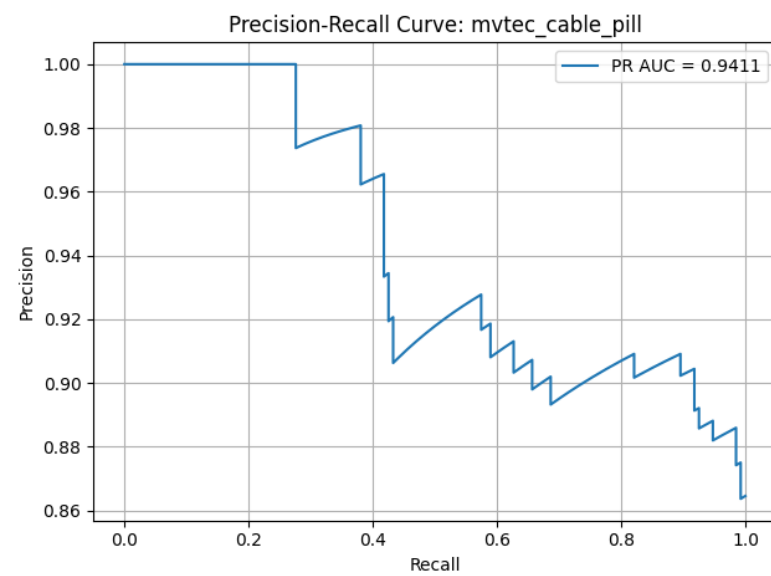
## Wood

Figure B.29: Precision-Recall Curve for class *wood* in Experiment 2.

Figure B.30: ROC Curve for class *wood* in Experiment 2.

## B.4 Experiment 3

### Pill

Figure B.31: Precision-Recall Curve for class *pill* in Experiment 3.



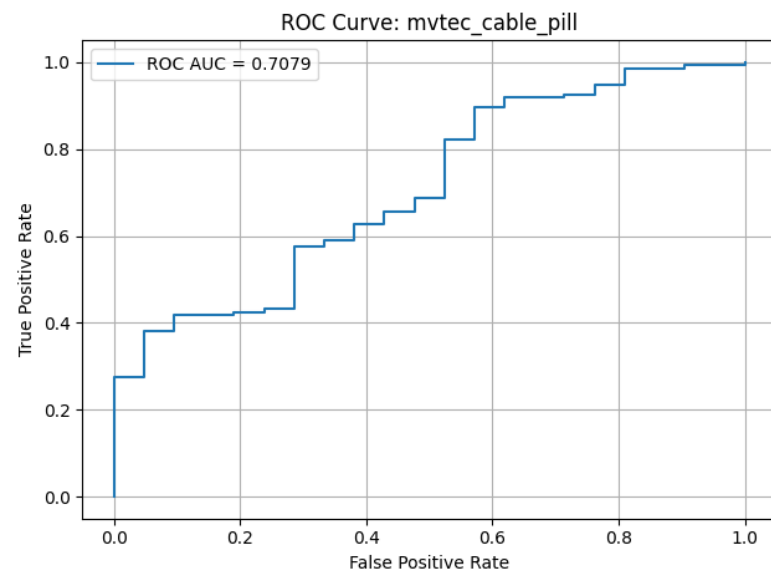


Figure B.32: ROC Curve for class *pill* in Experiment 3.

### Metal\_nut

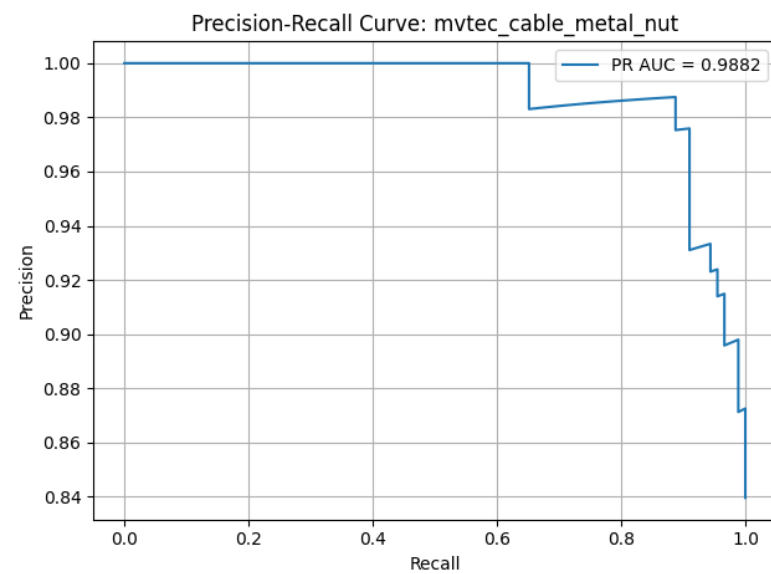
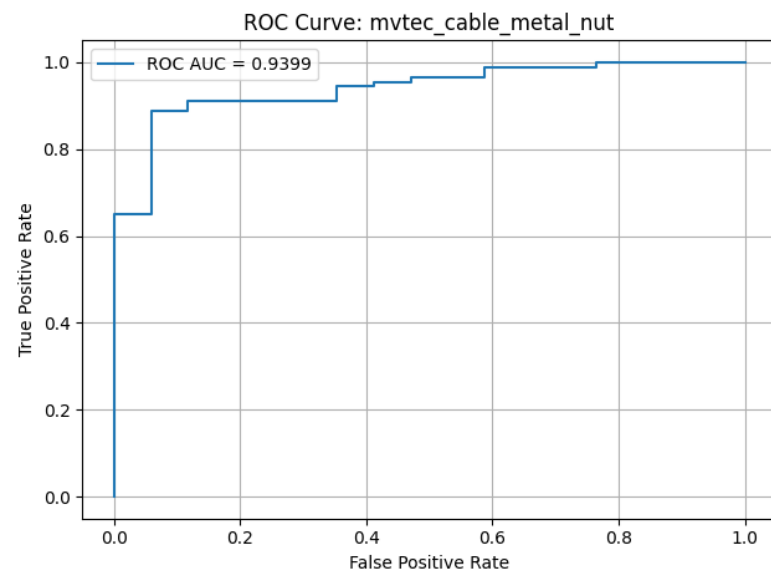
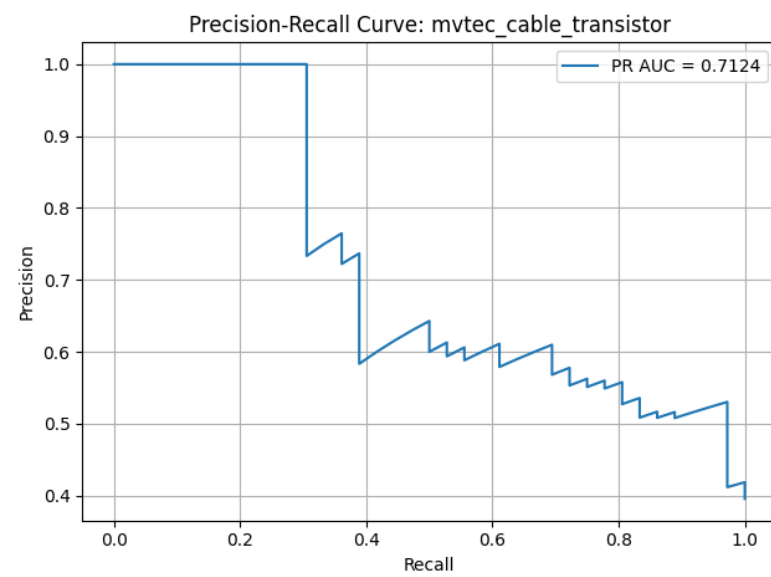


Figure B.33: Precision-Recall Curve for class *metal\_nut* in Experiment 3.

Figure B.34: ROC Curve for class *metal\_nut* in Experiment 3.

## Transistor

Figure B.35: Precision-Recall Curve for class *transistor* in Experiment 3.

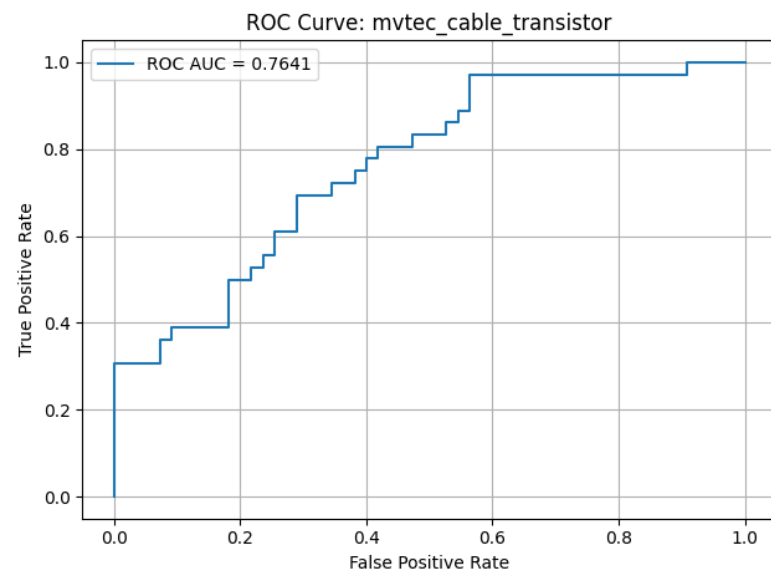


Figure B.36: ROC Curve for class *transistor* in Experiment 3.

## Tile

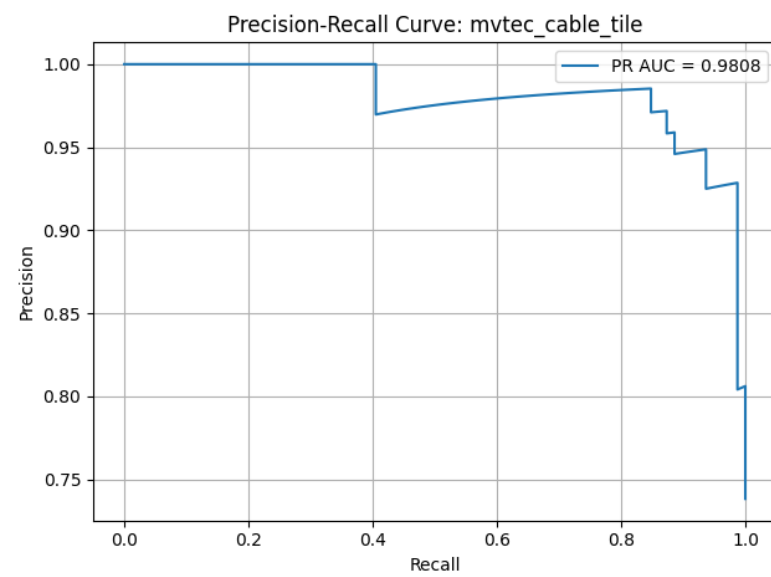
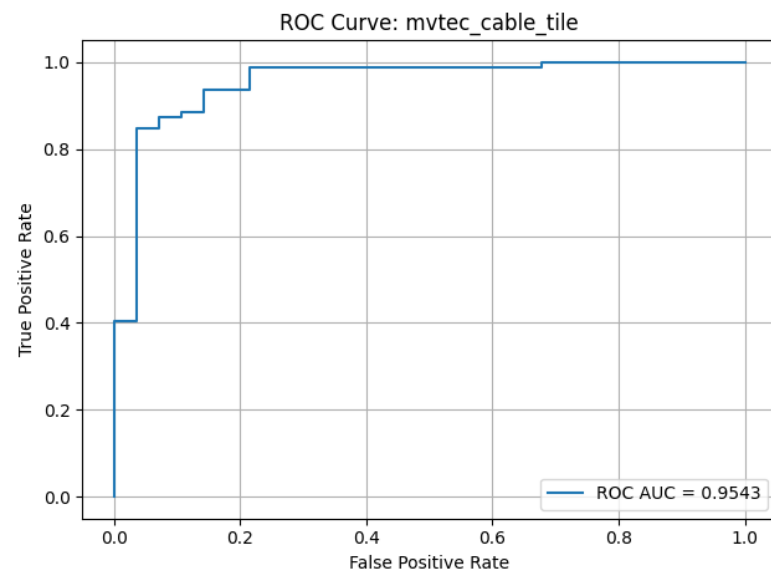
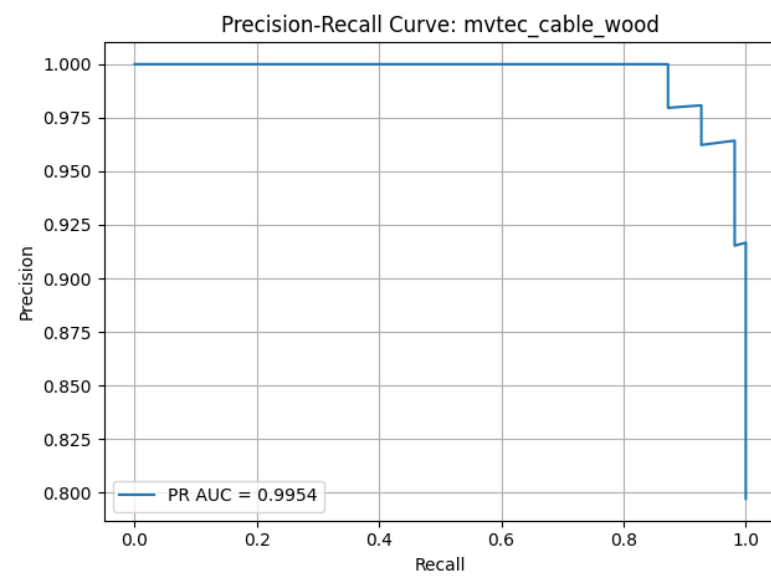


Figure B.37: Precision-Recall Curve for class *tile* in Experiment 3.

Figure B.38: ROC Curve for class *tile* in Experiment 3.

## Wood

Figure B.39: Precision-Recall Curve for class *wood* in Experiment 3.

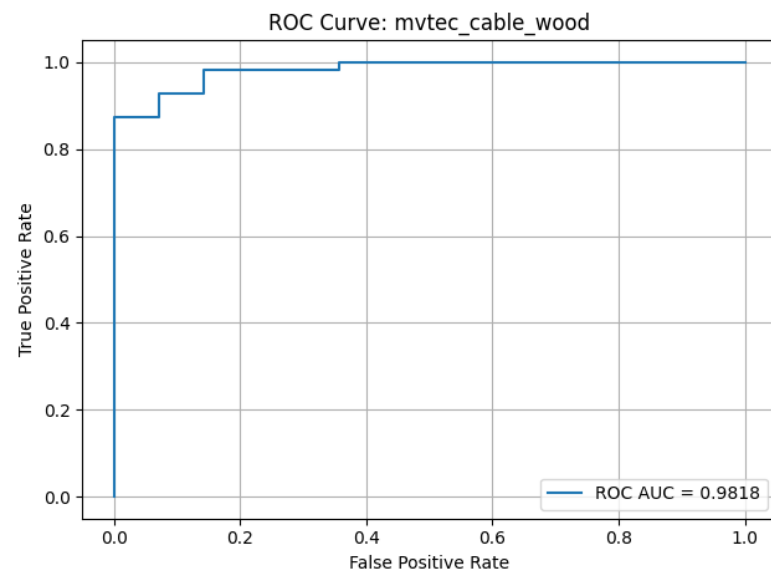


Figure B.40: ROC Curve for class *wood* in Experiment 3.

## B.5 Experiment 4

### Pill

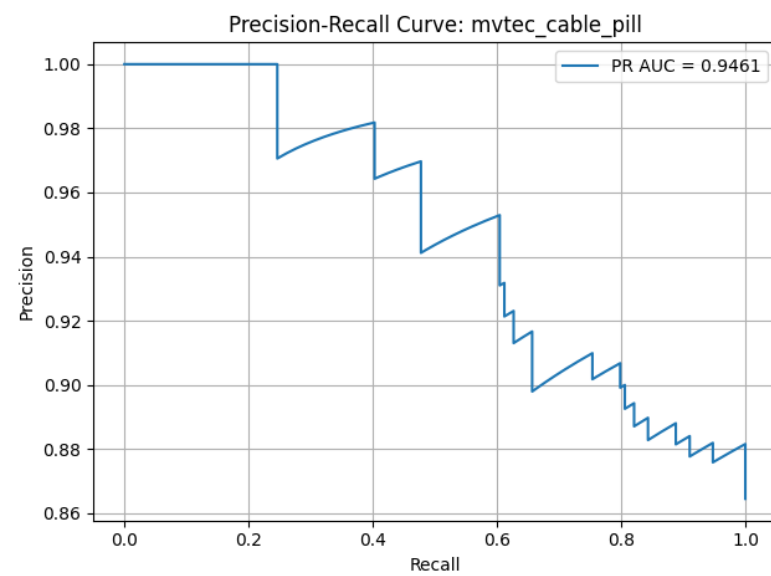
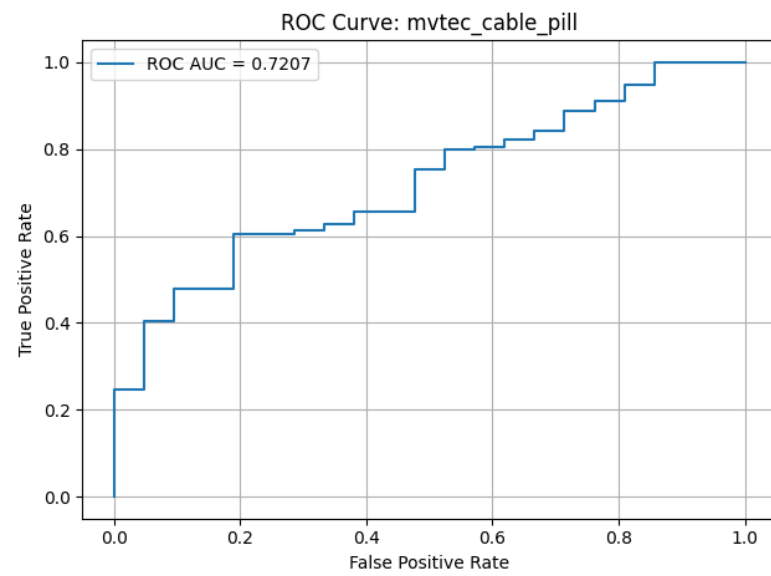
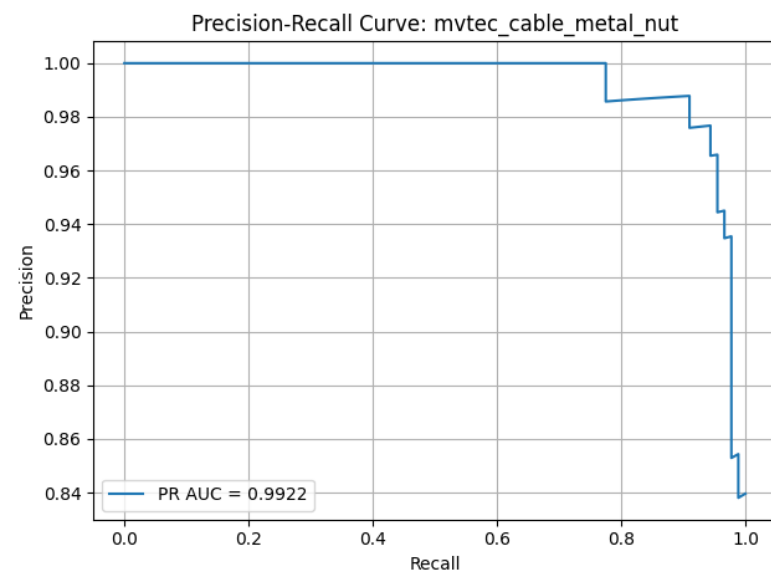


Figure B.41: Precision-Recall Curve for class *pill* in Experiment 4.

Figure B.42: ROC Curve for class *pill* in Experiment 4.**Metal\_nut**Figure B.43: Precision-Recall Curve for class *metal\_nut* in Experiment 4.

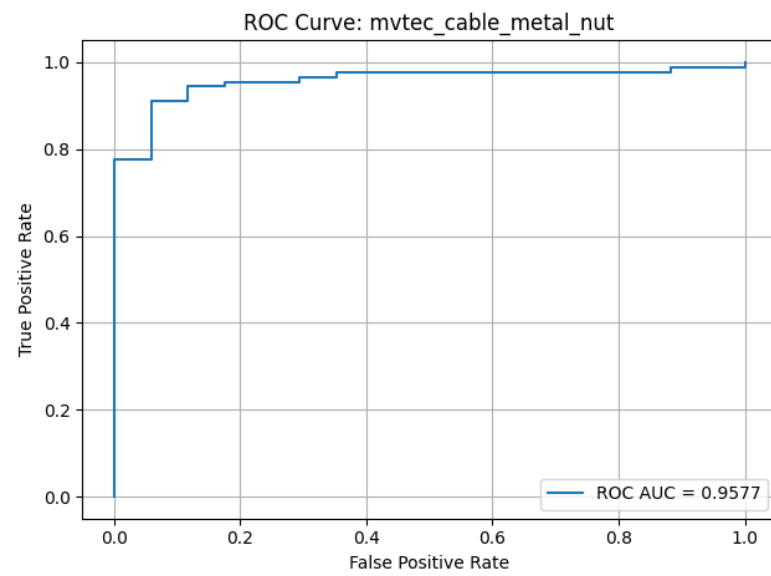


Figure B.44: ROC Curve for class *metal\_nut* in Experiment 4.

## Transistor

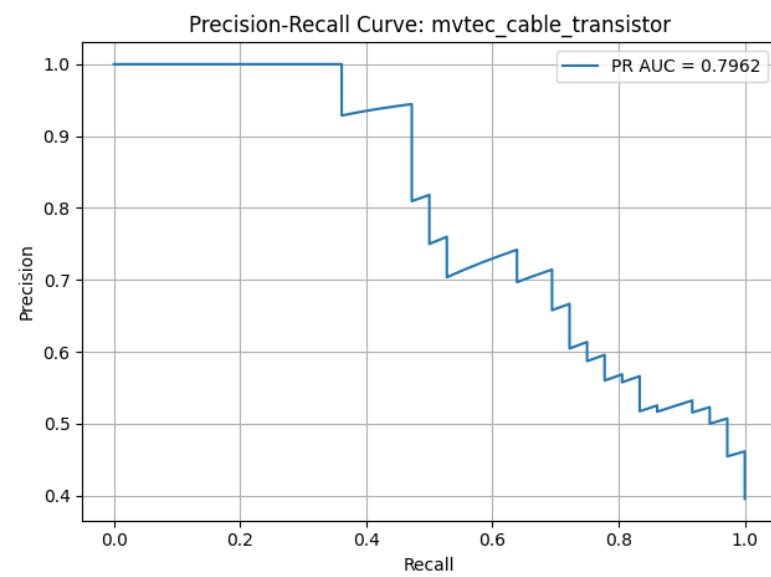
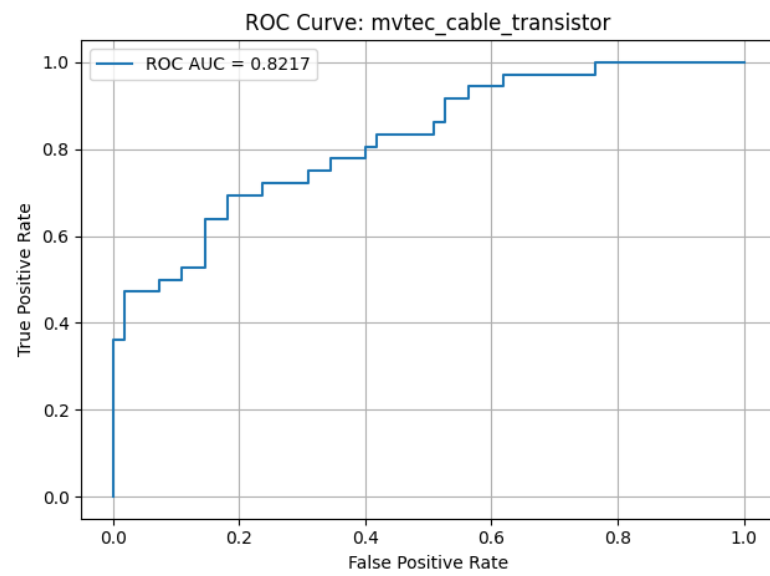
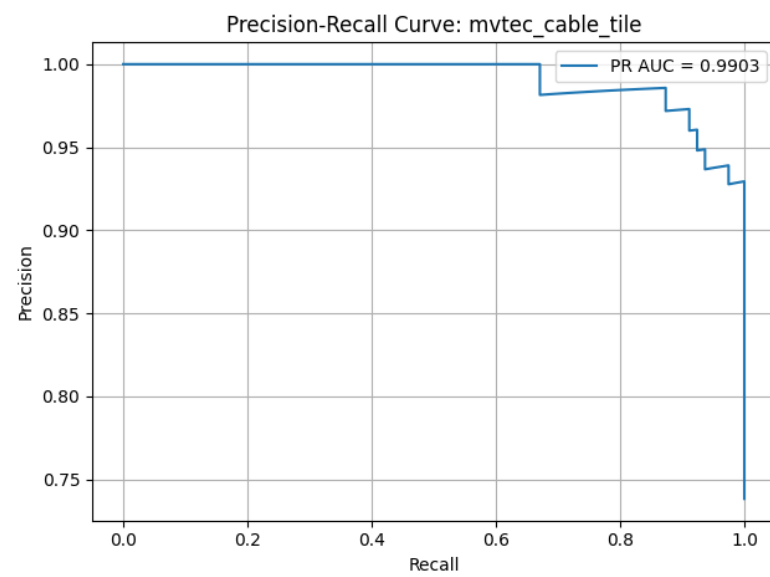


Figure B.45: Precision-Recall Curve for class *transistor* in Experiment 4.

Figure B.46: ROC Curve for class *transistor* in Experiment 4.

## Tile

Figure B.47: Precision-Recall Curve for class *tile* in Experiment 4.



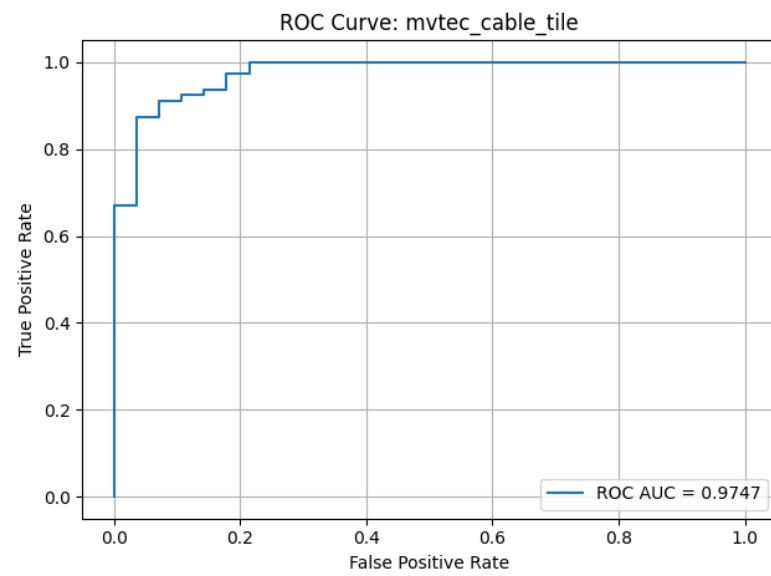


Figure B.48: ROC Curve for class *tile* in Experiment 4.

## Wood

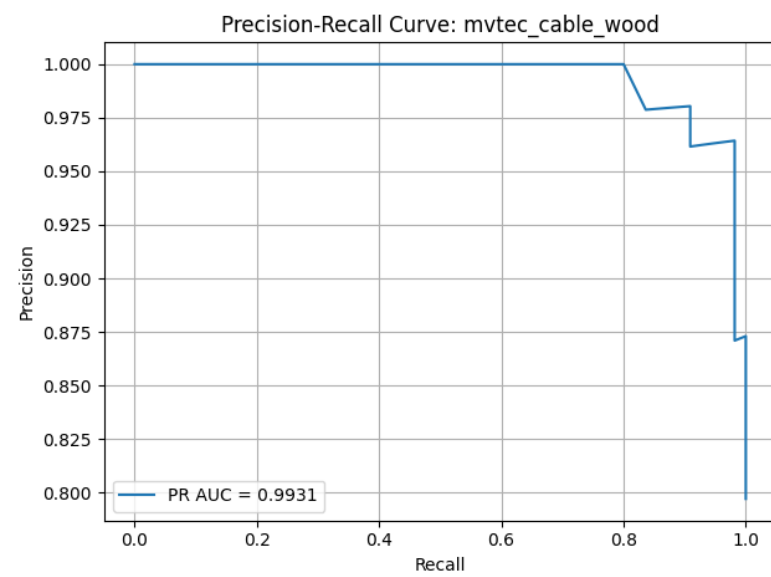
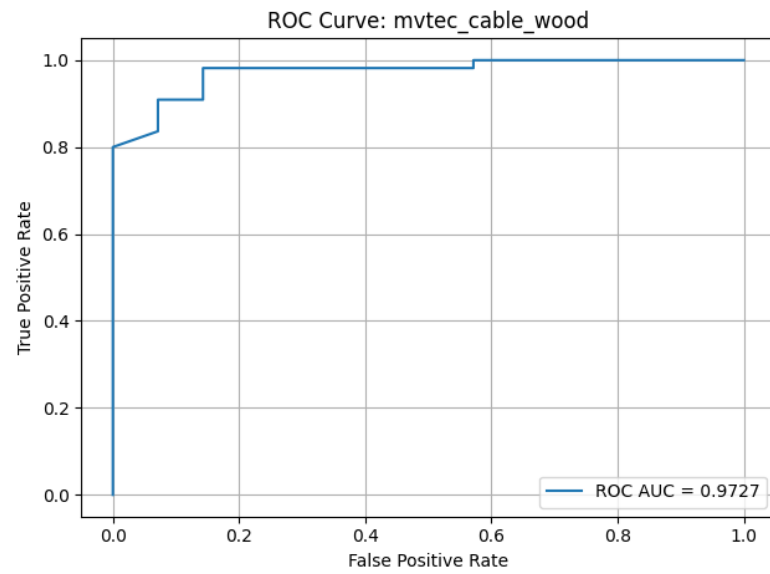
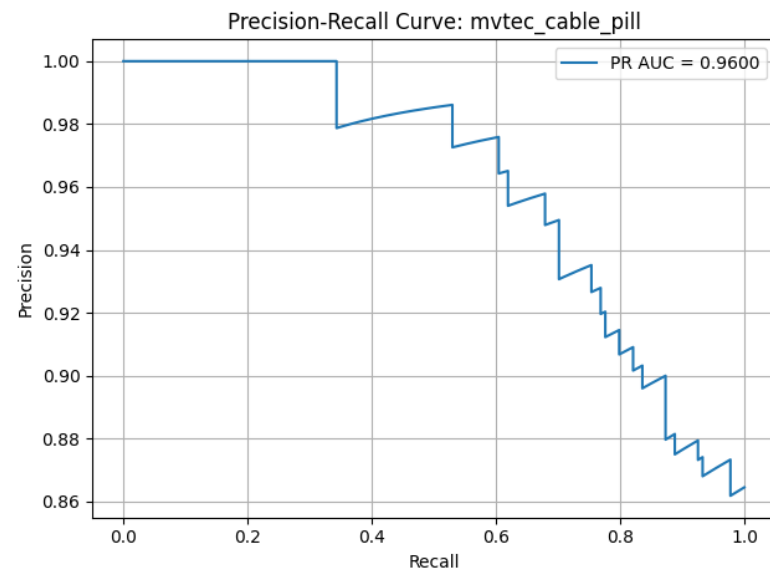


Figure B.49: Precision-Recall Curve for class *wood* in Experiment 4.

Figure B.50: ROC Curve for class *wood* in Experiment 4.

## B.6 Experiment 5

### Pill

Figure B.51: Precision-Recall Curve for class *pill* in Experiment 5.

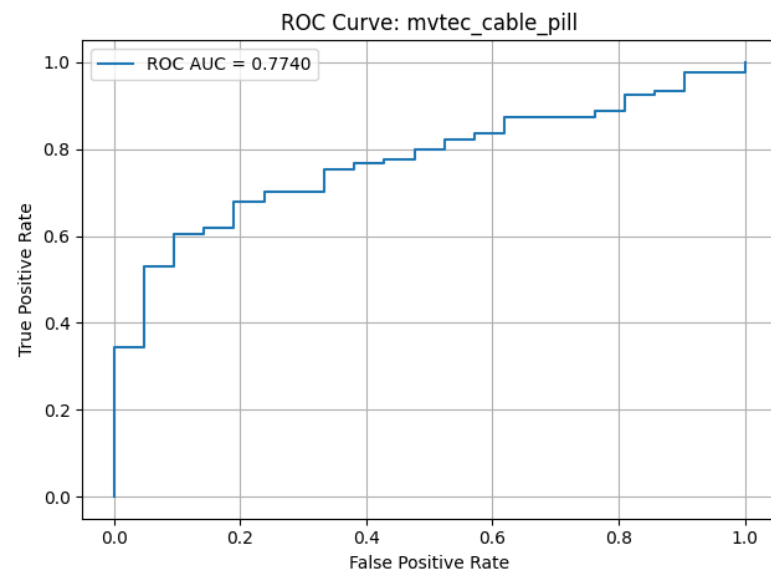


Figure B.52: ROC Curve for class *pill* in Experiment 5.

### Metal\_nut

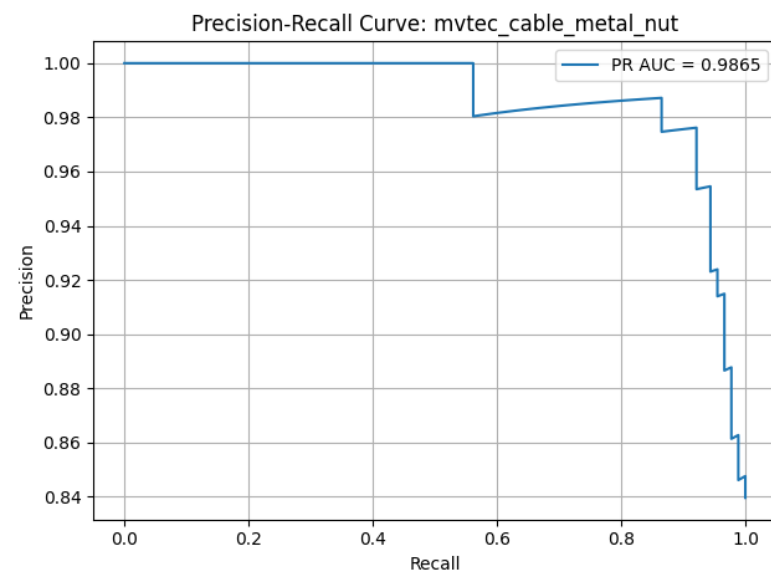
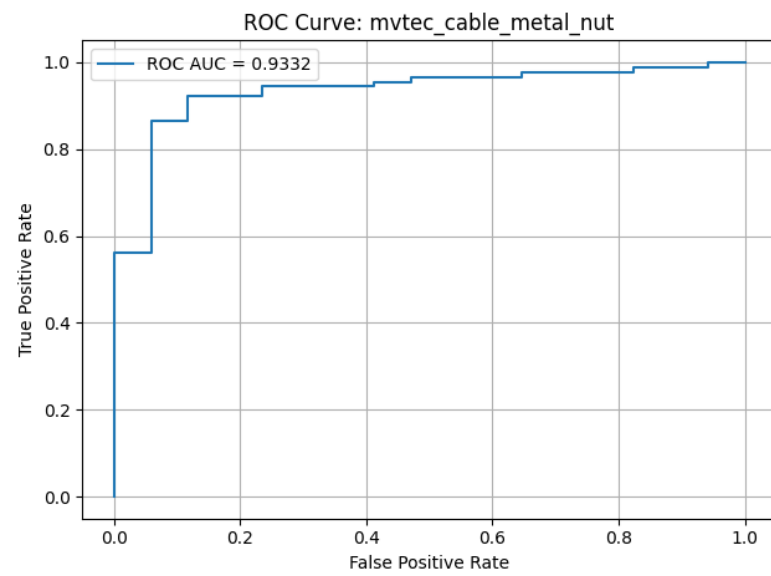
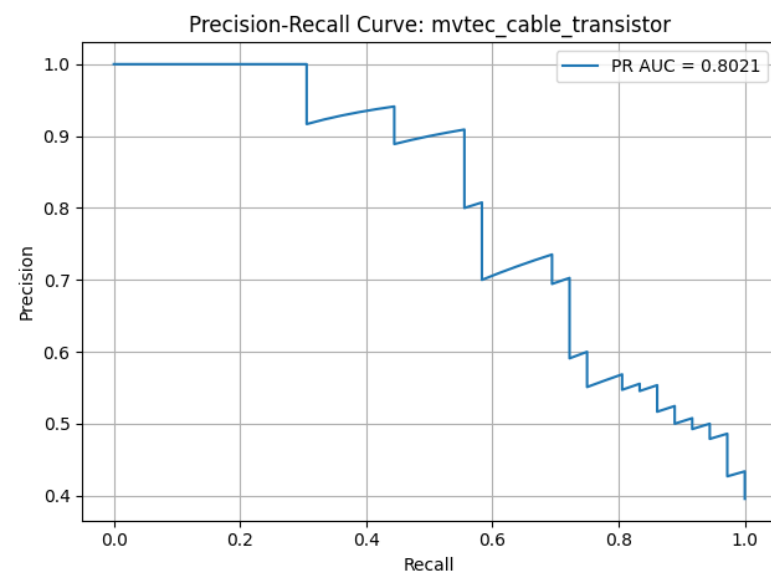


Figure B.53: Precision-Recall Curve for class *metal\_nut* in Experiment 5.

Figure B.54: ROC Curve for class *metal\_nut* in Experiment 5.

## Transistor

Figure B.55: Precision-Recall Curve for class *transistor* in Experiment 5.

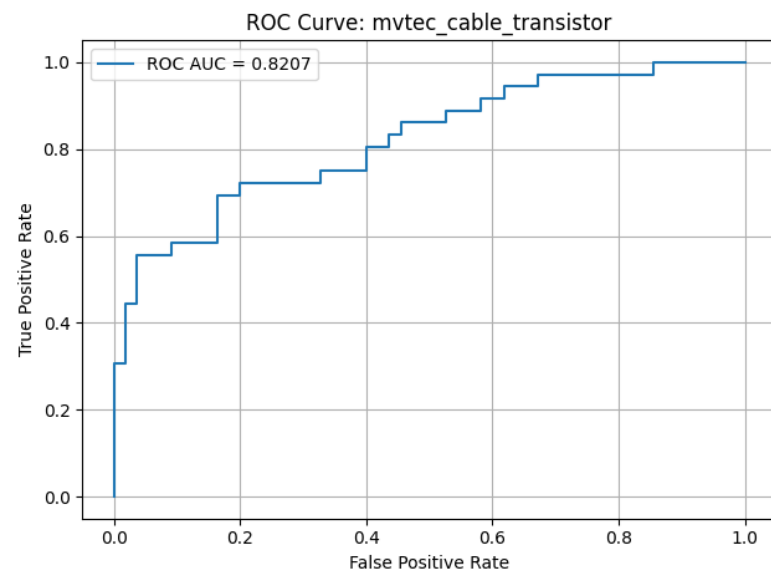


Figure B.56: ROC Curve for class *transistor* in Experiment 5.

## Tile

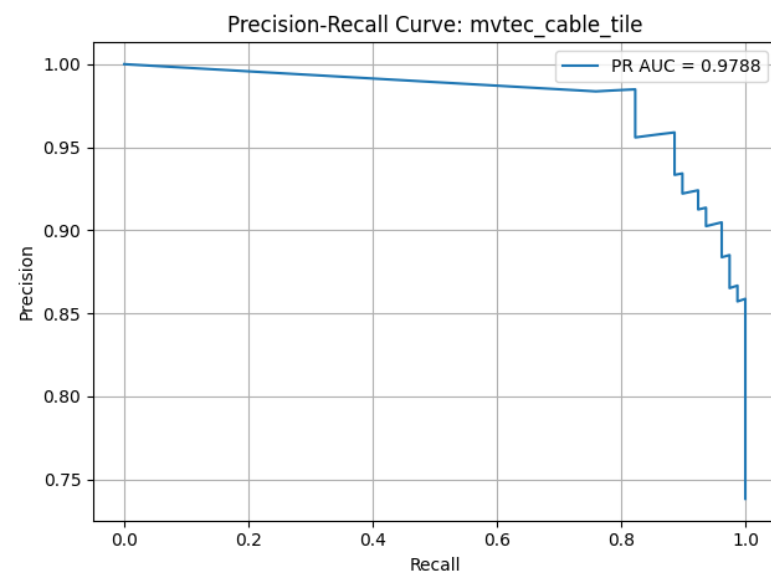
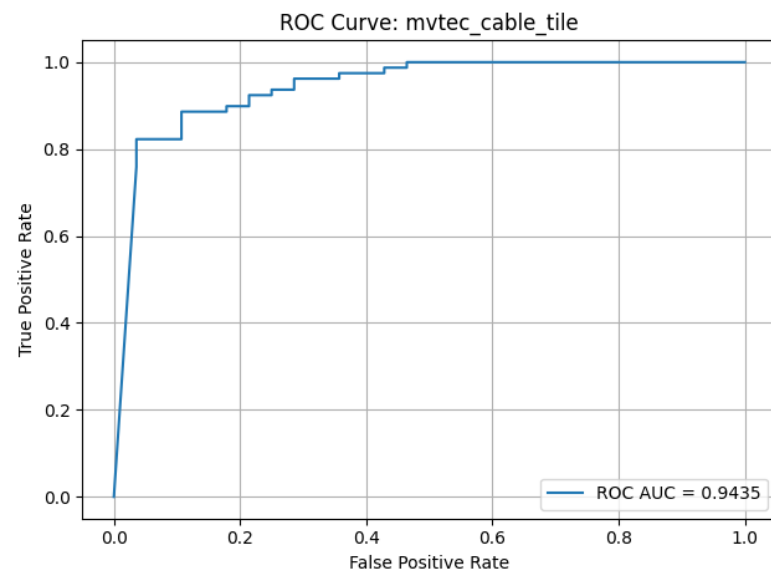
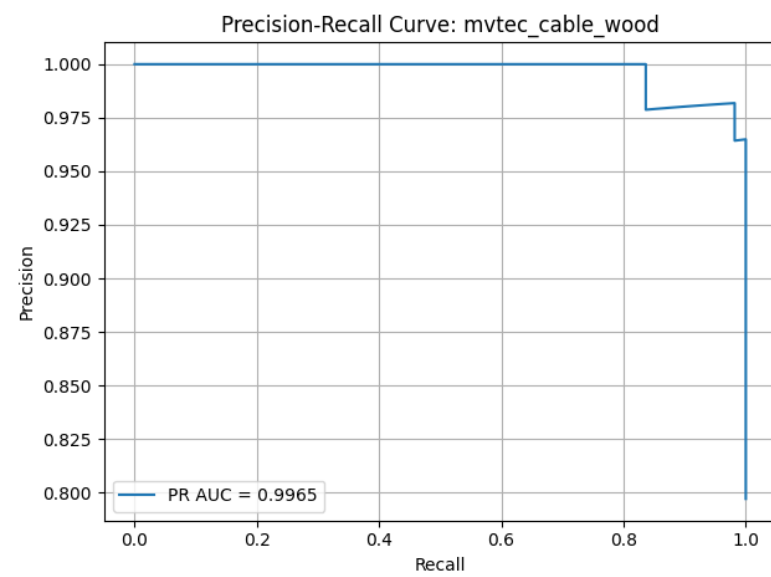


Figure B.57: Precision-Recall Curve for class *tile* in Experiment 5.

Figure B.58: ROC Curve for class *tile* in Experiment 5.

## Wood

Figure B.59: Precision-Recall Curve for class *wood* in Experiment 5.

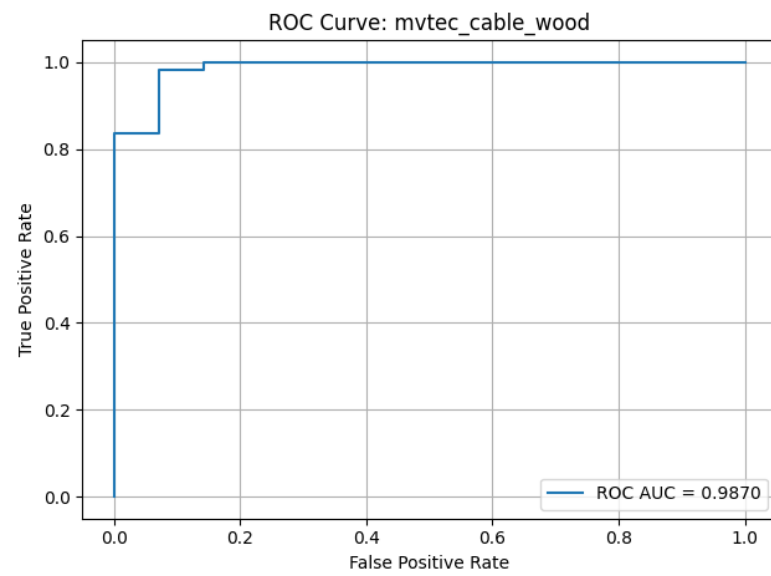


Figure B.60: ROC Curve for class *wood* in Experiment 5.

## Appendix C

### Stable Diffusion plots

#### Pill

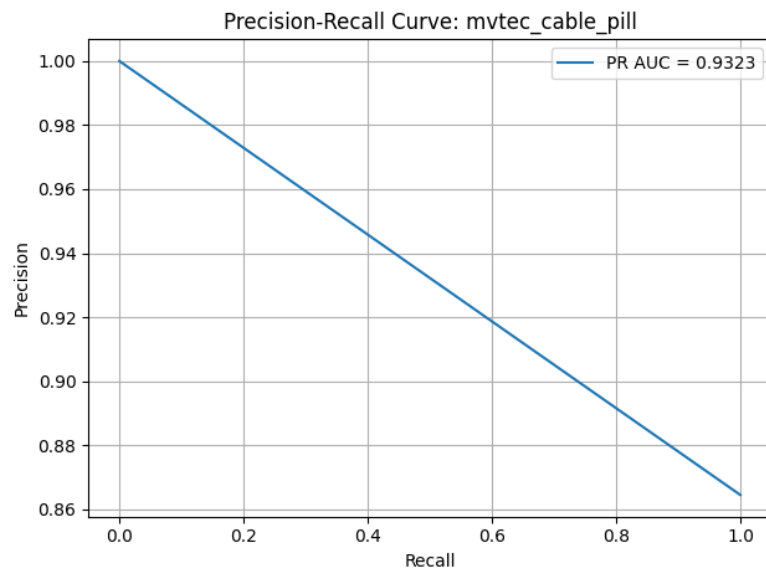


Figure C.1: Precision-Recall Curve for class *pill* in Experiment 6.



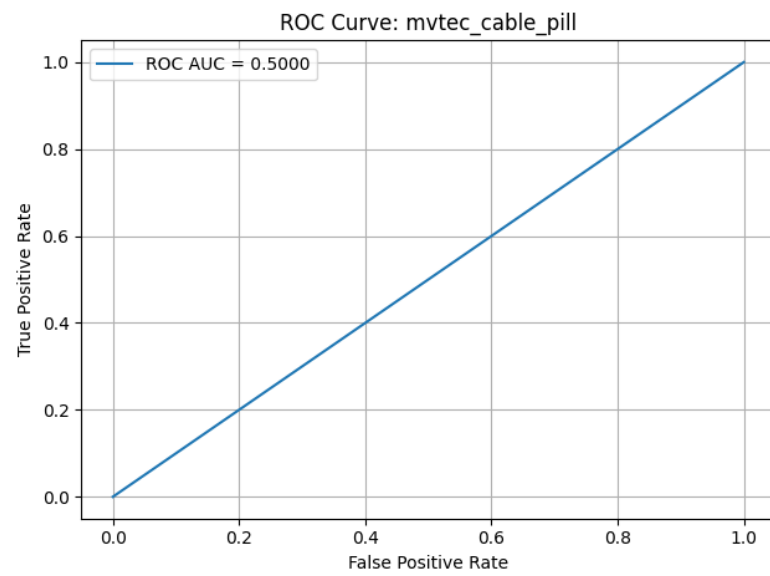


Figure C.2: ROC Curve for class *pill* in Experiment 6.

### Metal\_nut

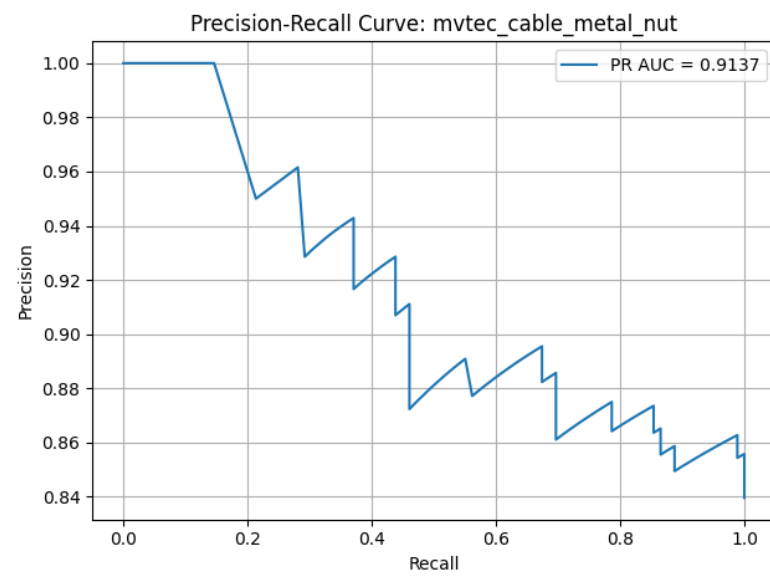


Figure C.3: Precision-Recall Curve for class *metal\_nut* in Experiment 6.

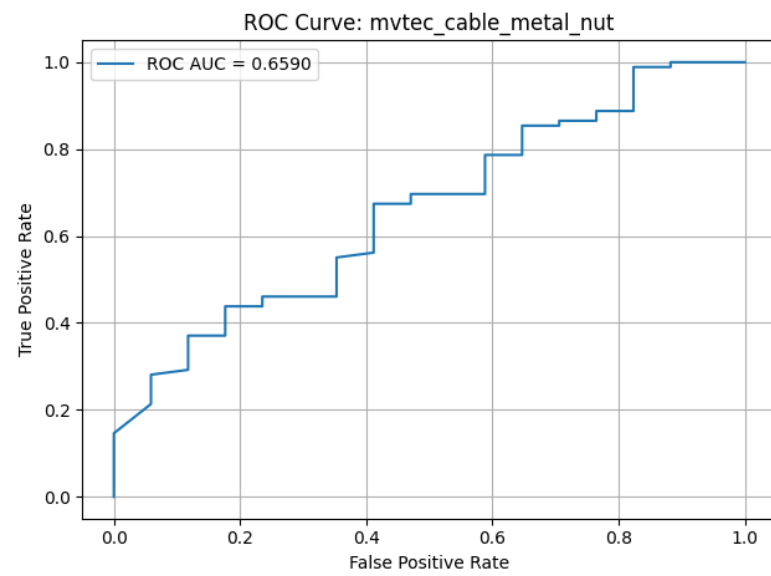


Figure C.4: ROC Curve for class *metal\_nut* in Experiment 6.

## Tile

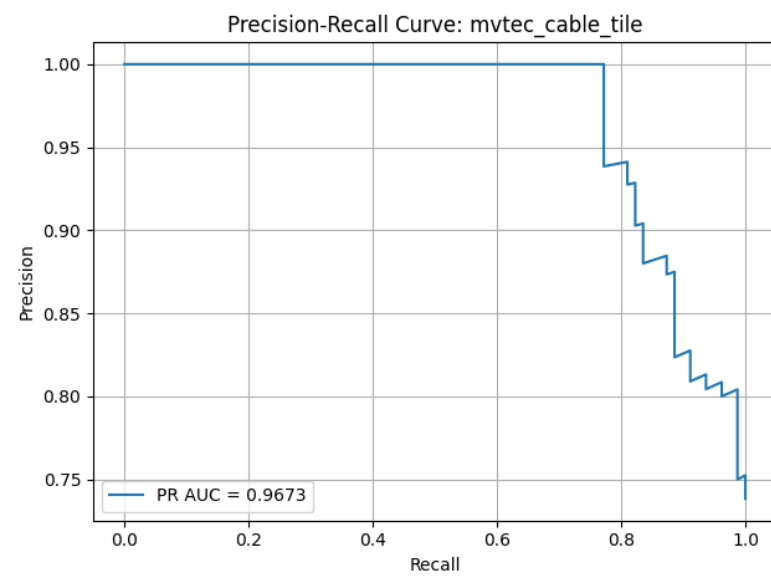


Figure C.5: Precision-Recall Curve for class *tile* in Experiment 6.

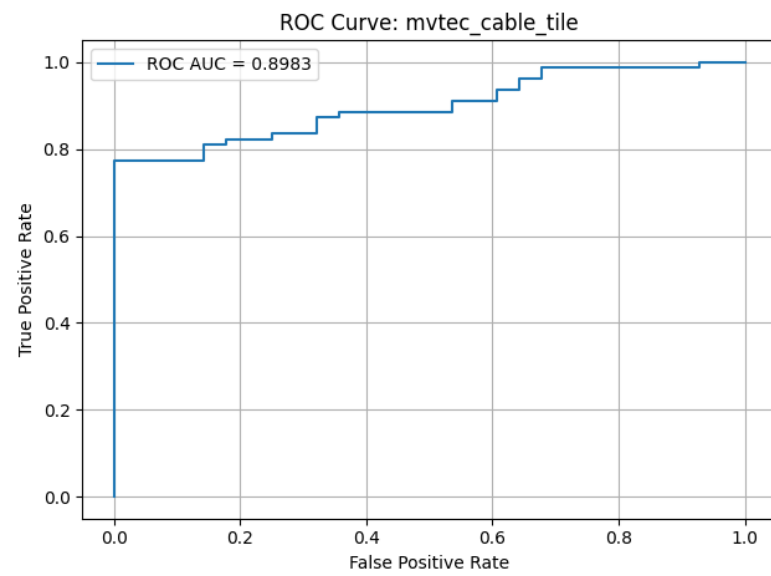


Figure C.6: ROC Curve for class *tile* in Experiment 6.

## Wood

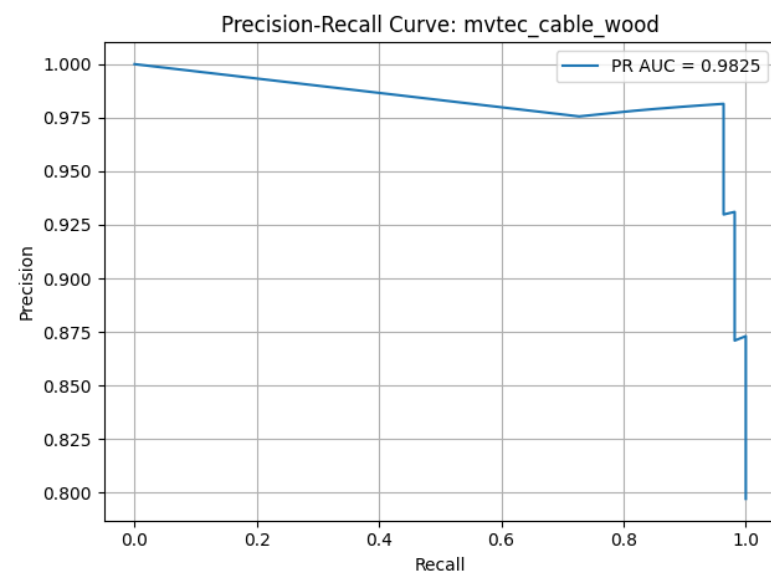


Figure C.7: Precision-Recall Curve for class *wood* in Experiment 6.

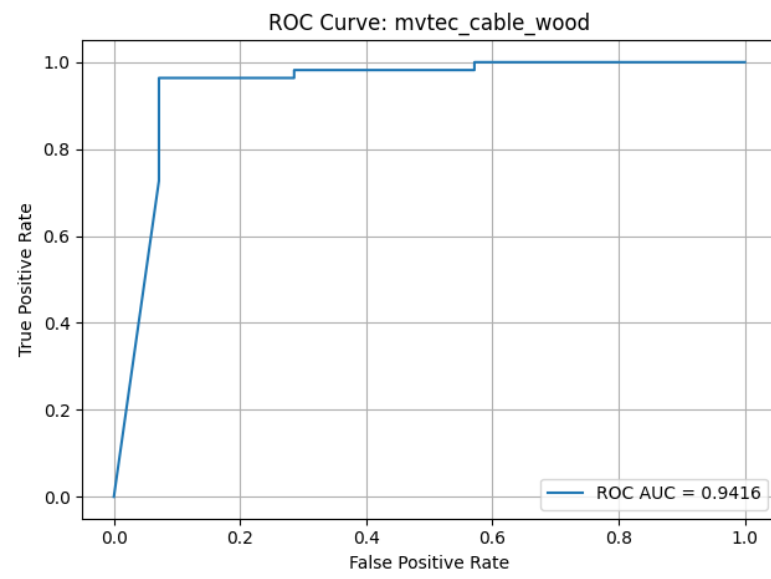


Figure C.8: ROC Curve for class *wood* in Experiment 6.

## Appendix D

### Zero-shot Learning plots

Pill

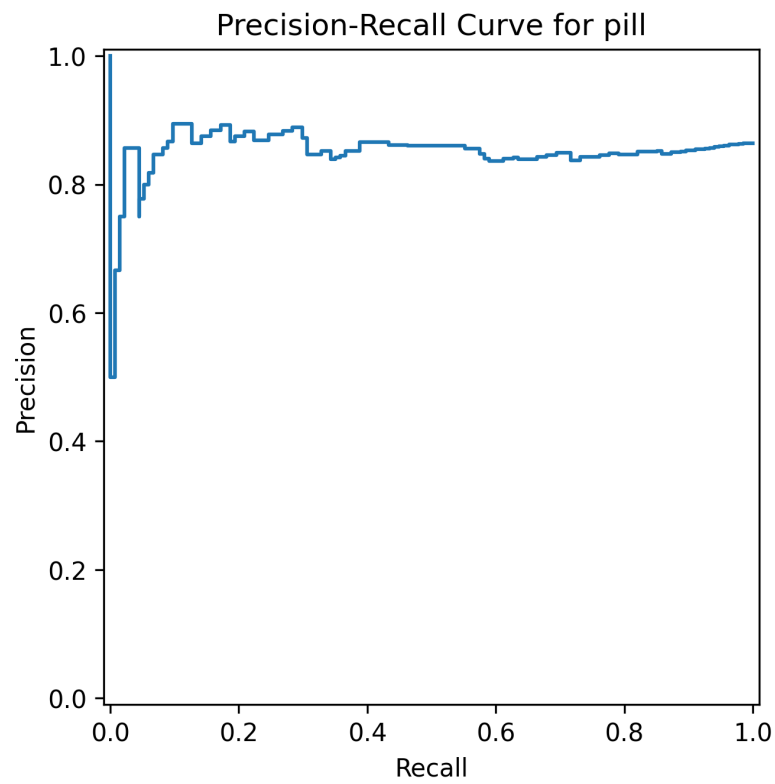


Figure D.1: Precision-Recall Curve for class *pill* in ZSL Experiment.

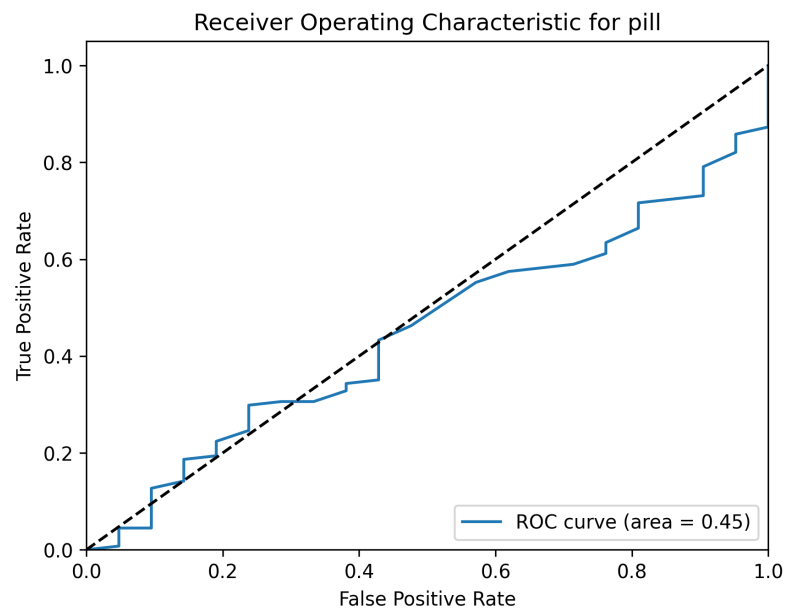


Figure D.2: ROC Curve for class *pill* in ZSL Experiment.

### **Metal\_nut**

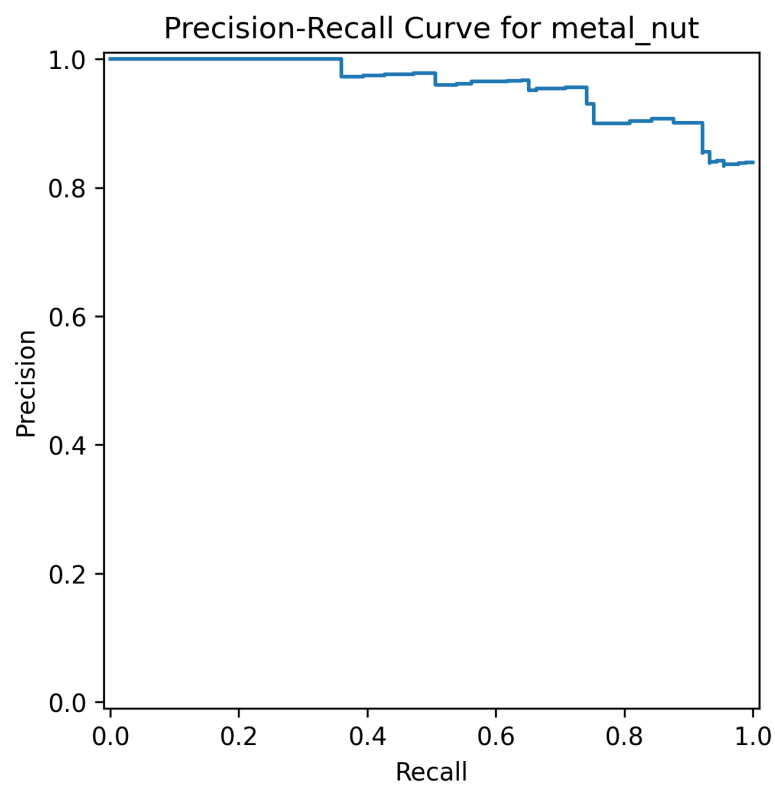


Figure D.3: Precision-Recall Curve for class *metal\_nut* in ZSL Experiment.

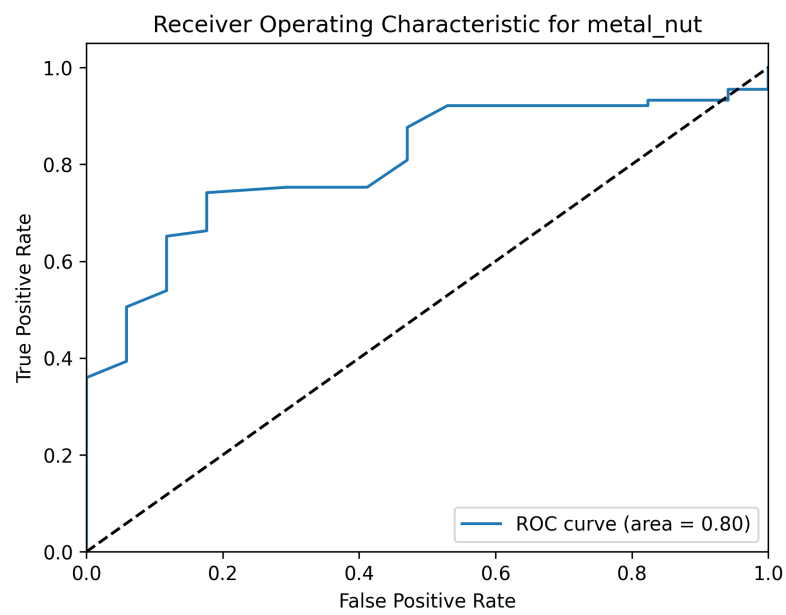


Figure D.4: ROC Curve for class *metal\_nut* in ZSL Experiment.

## Tile

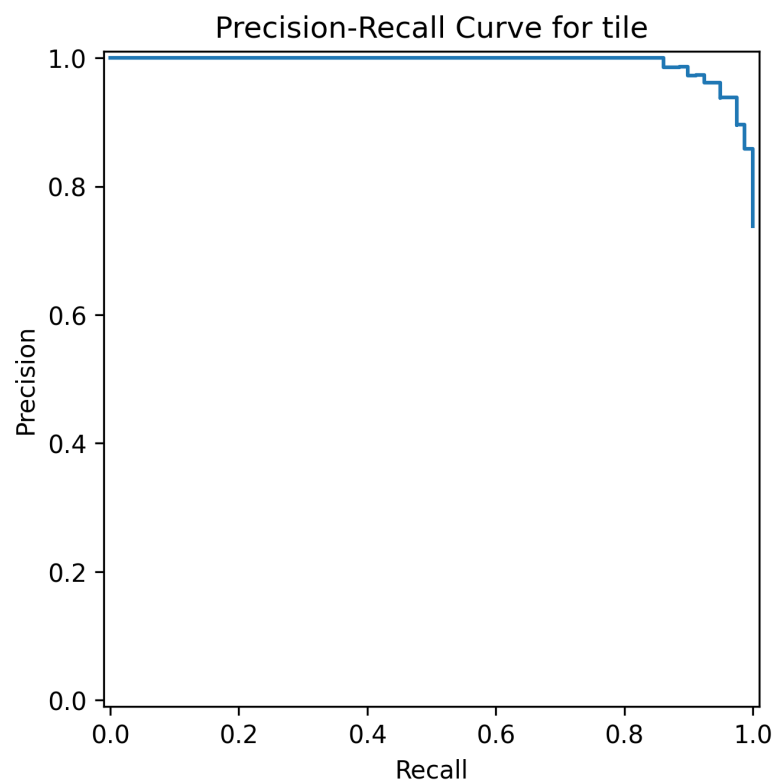


Figure D.5: Precision-Recall Curve for class *tile* in ZSL Experiment.

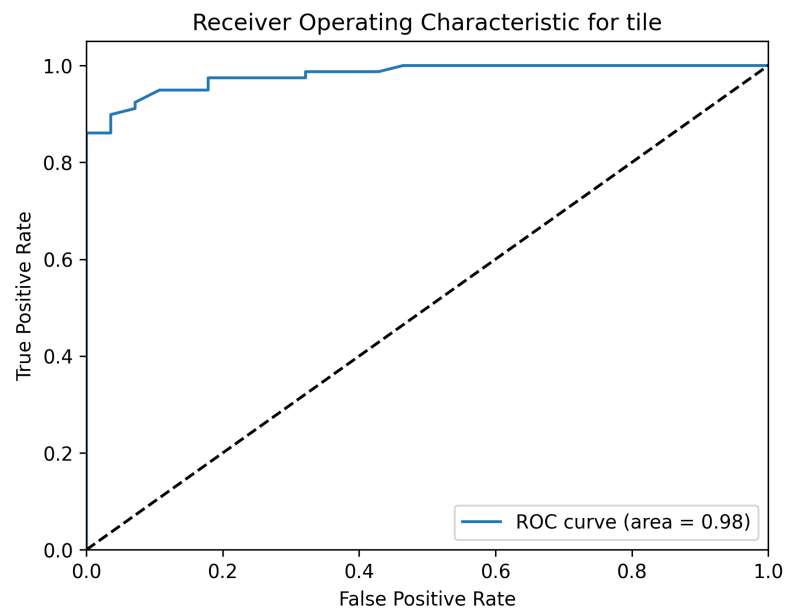


Figure D.6: ROC Curve for class *tile* in ZSL Experiment.

## Wood

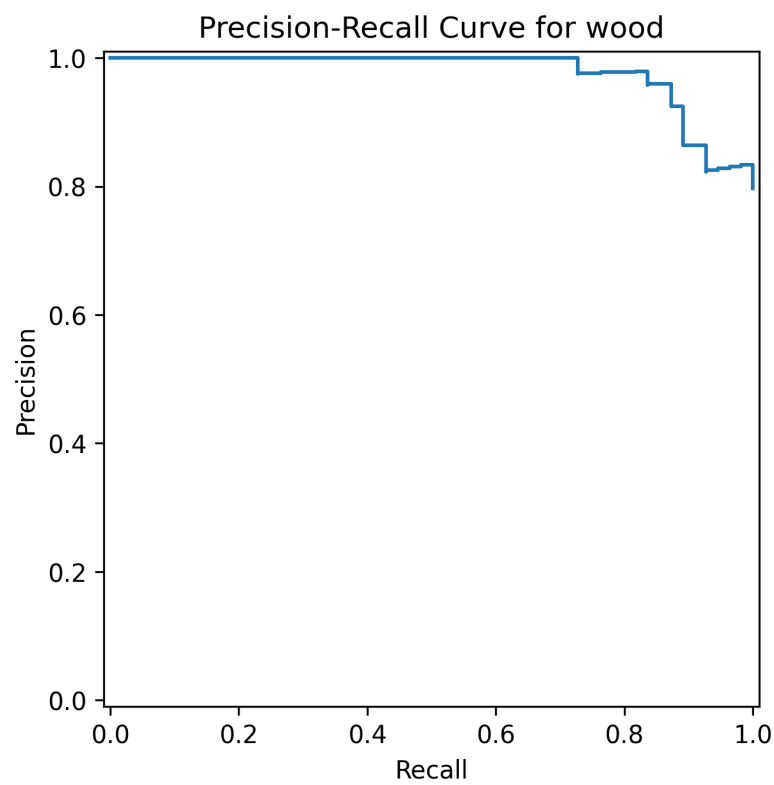


Figure D.7: Precision-Recall Curve for class *wood* in ZSL Experiment.



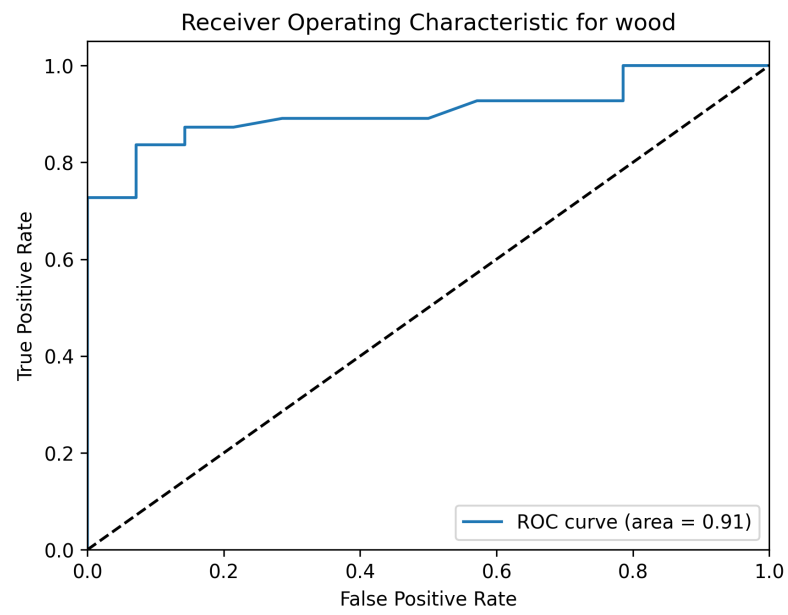


Figure D.8: ROC Curve for class *wood* in ZSL Experiment.