

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



Generative AI for Ceramics Painting

José Francisco Figueira Mendes Samagaio

MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Supervisor: Carlos Manuel Milheiro de Oliveira Pinto Soares

October 18, 2023

Resumo

A indústria da cerâmica, conhecida pela sua ênfase na estética e na qualidade, está a enfrentar uma procura crescente de desenvolvimento de produtos. A aprendizagem computacional, especificamente a inteligência artificial generativa, pode potencialmente otimizar e melhorar o processo de coloração e design da cerâmica. Esta investigação tem como objetivo desenvolver um novo modelo generativo e explorar a sua aplicabilidade na indústria. O recurso à IA generativa pode resolver o problema da avaliação humana subjectiva e da natureza intensiva em termos de tempo do desenvolvimento de novos produtos. O estudo irá comparar o novo modelo com os modelos existentes mais avançados e avaliar métricas e técnicas. A investigação contribui para o avanço da indústria cerâmica através da introdução de abordagens eficientes e inovadoras ao desenvolvimento e reprodução de cores. O modelo generativo desenvolvido gera com êxito imagens sintéticas altamente realistas e autênticas de material cerâmico com diferentes cores e posições. Esta investigação demonstra a fiabilidade e a eficácia da IA generativa, oferecendo uma solução eficiente em termos de tempo e de recursos, ao mesmo tempo que promove a criatividade e a inovação. Os resultados contribuem para o avanço do desenvolvimento e conceção de cores em cerâmica, abrindo caminho para uma futura implementação prática na indústria. A nova abordagem foi capaz de produzir resultados muito bons, gerando imagens que apresentam um nível de realidade muito elevado e são competitivas com os modelos mais avançados.

Abstract

The ceramics industry, known for its focus on aesthetics and quality, faces increased demand for product development. Machine learning, specifically generative artificial intelligence, can optimize and enhance the colouring of ceramics and the design process. This research aims to develop a new generative model and explore its applicability in the industry. Leveraging generative AI can address the subjective human evaluation and time-intensive nature of new product development. The study will compare the new model with existing state-of-the-art models and evaluate metrics and techniques. The research can contribute to the advancement of the ceramics industry by introducing efficient and innovative approaches to colour development and design. The developed generative model based on Stable Diffusion successfully generates highly realistic and authentic synthetic images of plates with different colours and positions. We showcase the reliability and effectiveness of generative AI, offering a time-efficient and resource-saving solution while fostering creativity and innovation. The new approach delivered excellent results, generating images that present a very high level of reality and are competitive with state-of-the-art models.

Agradecimentos

Gostaria de expressar meus sinceros agradecimentos às pessoas que foram fundamentais na minha jornada acadêmica e pessoal. Em primeiro lugar, gostaria de agradecer minha família, que sempre esteve ao meu lado me apoiou. Meu pai e minha mãe são os pilares da minha vida, que me deram tudo que eu precisei para chegar até aqui. Também quero agradecer ao meu irmão, que me ajudou muito e sempre foi uma fonte constante de motivação, sendo um exemplo que sempre segui. Agradeço imensamente a todos os meus colegas por estes longos 5 anos de luta. Estiveram sempre presentes em todos os momentos, seja nos FEUPCafes como nas longas noites de estudo. A amizade e o apoio que recebi de foi essenciais para superar os desafios e chegar até aqui. Quero dedicar um agradecimento especial à minha namorada, por todo o apoio incondicional que ela me deu ao longo dessa jornada. Ela sempre esteve ao meu lado e sempre me incentivou e acreditou em mim. Também gostaria de agradecer aos meus amigos de longa data da escola. Mesmo que tenhamos nos afastado um pouco, sempre estiveram presentes quando eu precisei. A todos vocês, minha família, colegas, namorada e amigos, expresse minha profunda gratidão. Sem o voço apoio e a presença eu não teria alcançado este marco na minha vida acadêmica. Obrigado por tudo e por serem parte integrante do meu sucesso.

*“If you don’t believe in yourself
no one will do it for you”*

Kobe Bryant

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Goals	2
1.3	Document structure	3
2	Literature Review	5
2.1	SmartColor4Ceramics	5
2.2	Neural Networks	5
2.2.1	Natural language processing	8
2.2.2	Autoencoder	9
2.2.3	Segmentation	11
2.2.4	Convolutional neural networks	12
2.3	AI-based product design	15
2.3.1	Ceramics	15
2.3.2	Product design	15
2.3.3	Colour spaces	16
2.3.4	Generative product design	19
2.4	Summary	19
3	Generative models for product development	21
3.1	Generative adversarial networks	21
3.2	Diffusion models	22
3.2.1	Diffusion models approach	23
3.2.2	Diffusion models components	25
3.2.3	Stable diffusion	27
3.3	Evaluation of generative models	31
3.4	Summary	32
4	Methodology	33
4.1	Hardware	33
4.2	Dataset preparation	34
4.3	Stable-GAN	35
4.3.1	Model architecture	35
4.3.2	Discriminator	36
4.3.3	Hyperparameters	37
4.3.4	Fine-tuning	38
4.4	Fine-tune stable diffusion	38
4.4.1	Fine-tuning techniques	39

4.4.2	Hyperparameters	40
4.4.3	Model training	40
4.5	Evaluations	41
4.5.1	Quantitative evaluation	41
4.5.2	Qualitative evaluation	44
4.6	Summary	45
5	Results	47
5.1	Qualitative evaluation	47
5.1.1	Fine-tuned model selection	47
5.1.2	Hyperparameter	50
5.1.3	Model selection	53
5.2	Quantitative evaluation	54
5.2.1	LoRA	54
5.2.2	Stable-gan	55
5.3	Summary	57
6	Conclusion and Future Work	59
	References	61

List of Figures

1.1	Number of Companies and workers per sub-sector of the Portuguese ceramics industry in 2014	2
1.2	Colour dishes created by generative AI when asked to draw "Real colour dishes"	3
2.1	Flow diagram during Formulation Production for colour verification using the SmartColor4Ceramics solution	6
2.2	Example of a fully connected network	6
2.3	Components of an artificial neuron	7
2.4	Autoencoder architecture	11
2.5	Segmentation of the image. The animal is classified as a cat and separated from the rest of the background	11
2.6	Example of calculating Intersection over Union (IoU) computation, dividing the area of overlap between the bounding boxes by the area of union	13
2.7	Caption features with convolution layers in CNN	13
2.8	Convolution layer transformation	14
2.9	U-Net architecture	14
2.10	U-Net architecture focusing in skip connections	15
2.11	Representation of a matrix with Red;Green;Blue (RGB) colour space	16
2.12	Representation of Lab colour space	17
2.13	Example of ΔE as the distance between two colours	18
3.1	Example of generative architecture using text-to-image	22
3.2	The architecture of Generative Adversarial Networks	22
3.3	Architecture of diffusion model	23
3.4	Architecture of the forward process in diffusion models	24
3.5	Architecture of the reverse process in diffusion models	24
3.6	A simplified architecture of U-Net	25
3.7	A simplified architecture of the autoencoder	26
3.8	A simplified architecture of the text encoder	27
3.9	A simplified architecture of the model	28
3.10	A architecture of the personalised concepts	29
4.1	Demonstration of the images dataset used	34
4.2	Simplify Stable-gan architecture	36
4.3	Simplify Stable-gan architecture with IoU and Delta E loss	38
4.4	Reference image used to calculate the IoU	42
4.5	Example of calculating IoU	43
4.6	Calculating the ΔE	44

5.1	Comparison results between (a) only real and (b) only fake images	48
5.2	Comparison between (a) Dreambooth and (b) Low rank adaption (LoRA).	49
5.3	Result comparison between (a) Dreambooth and (b) LoRA.	49
5.4	Result comparison between (a), (c) Dreambooth and (b),(d) LoRA.	50
5.5	Resulting image for each epoch, incrementing 250 epochs for all concept	51
5.6	Visualization on the effect of Scale and Guidance values	53
5.7	Comparison generated images between (a) LoRA and (b) Sable-GAN.	53
5.8	Comparison results of the questionnaire from (a) LoRA and (b) Sable-GAN.	54
5.9	Resulting outputs from the Stable-GAN	56
5.10	Resulting outputs from the Stable-GAN with the IoU and ΔE loss.	57

List of Tables

4.1	Number of epoch train in each concept	40
4.2	Plate class and corresponding token for the prompt.	41
4.3	Evaluation colours and their respective RGB colour code	44
4.4	The parameters tested in each questionnaire	45
4.5	Values associated with each response	45
5.1	Comparison of the survey results for LoRA and Dreambooth	50
5.2	Optimal Number of Training Steps for Each Concept	52
5.3	Results of the questionnaire on Scale and Guidance hyperparameters	52
5.4	Comparison of Survey Results for Different Models	55
5.5	Survey Results comparing the results for Stable-GAN	55
5.6	Comparison of Survey Results for Stable-gan	57

Abbreviations and Symbols

IoU	Intersection over Union
Lab	CIELAB color space
SAM	Segment Anything
RGB	Red;Green;Blue
GAN	Generative Adversarial Network
DL	Deep Learning
ML	Machine Learning
AI	Artificial Intelligence
DM	Diffusion Models
SD	Stable Diffusion
NLP	Natural language processing
VAE	Variational autoencoder
CNN	Convolutional neural network
NN	Neural network
FID	Fréchet Inception Distance
CLIP	Contrastive Language–Image Pre-training
HPC	High Performance Computing
GPU	Graphics processing unit
BIRME	Bulk Image Resize
CFG	Classifier-Free Guidance
MSE loss	Mean squared error loss
DCGANs	Deep Convolutional Generative Adversarial Networks
BCEloss	Binary Cross Entropy Loss
Adam	Adaptive Moment Estimation

SGD Stochastic Gradient Descent

Dsg Dark slate grey

CLIP-PS Contrastive Language–Image Pre-training - Perceptual Similarity

CIE International Commission on Illumination

PNDM Pseudo numerical methods for diffusion models

L1 L1 Loss

LoRA Low rank adaption

Chapter 1

Introduction

The world of ceramics is vast and rich, dating back thousands of years, from ancient pottery to modern tiles. These versatile inorganic materials are essential to our daily lives, from our dishes to our buildings. The ceramics market is a significant contributor to the global economy, with a market size of 239.53 billion (USD) estimated in 2022 and expected to reach 248.89 billion (USD) in 2023. The market is projected to grow at a compound annual growth rate of 5.2% from 2023 to 2030, reaching 359.35 billion (USD) by 2030 [1]. The ceramics market has several product segments, with ceramic tableware being the most popular one, and it is anticipated to grow at a lucrative pace over the forecast period [2].

The market for tableware ceramics is expected to grow at a compound annual growth rate of 8.5% [3] from 2022 to 2030, reaching 15.00 billion (USD) by 2030 [4]. The use of pottery and ceramic products has risen due to the increasing investment in creating aesthetically pleasing environments in hotels, restaurants and also at people's homes.

Ceramics is a dominant sector in the national industry and significantly impacts the international landscape market, resulting in high total business volume, mainly for export. This leading position is also innovation-driven in a pioneering sense by existing limitations. According to [5], in 2015, the ceramic market in Portugal accounted for about 0.5% of the GDP and 1.2% of the business volume in the manufacturing industry, employing around 15,900 workers through 1,127 companies. Figure 1.1 shows the number of companies and workers per sub-sector of the Portuguese ceramics industry in 2014.

The growing demand for ceramic tableware from hotels, restaurants and domestic needs encourages businesses to expand their production capacity. The emergence of new niches and consumer trends in the ceramics market highlights the need for product differentiation based on cost, functionality, and technological characteristics. Developing products with higher value-added through the reconciliation of new functionalities and designs is essential to meet the specific needs of consumers [6].

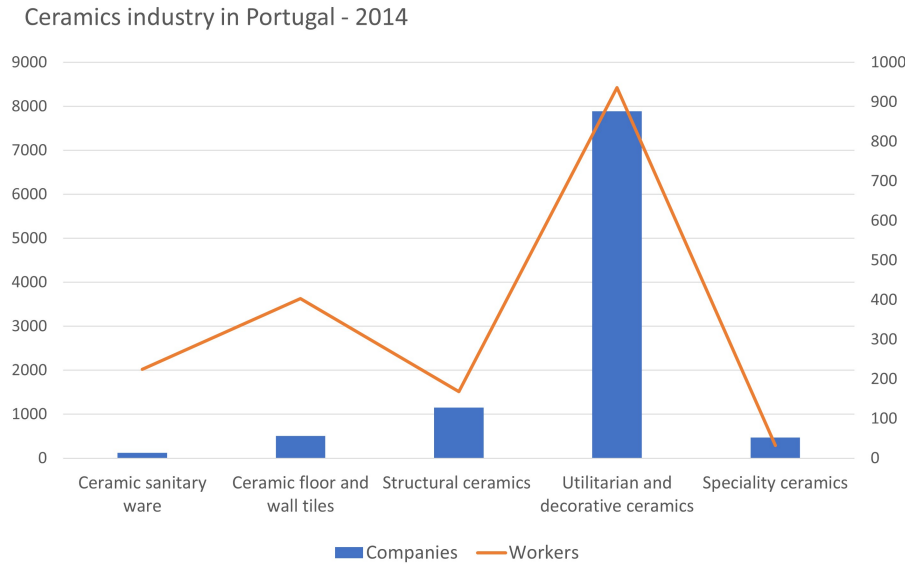


Figure 1.1: Number of Companies and workers per sub-sector of the Portuguese ceramics industry in 2014 [5]

1.1 Motivation

New product development is integral to any industry as it allows companies to stay competitive and respond to evolving consumer needs and preferences [6]. Nevertheless, this process is usually time and resource intensive. Today, new product design and development in the ceramics industry is mainly done by human labour, meaning it is expensive and hard to scale [7].

Artificial intelligence has become one of the biggest disruptors in the consumer market and is widely adapted to various services and products [8]. Recent generative artificial intelligence models can synthesise high-quality, diverse images based on text prompts written in natural language. Sometimes, it can easily create compelling, value-added products with accuracy indistinguishable from the actual photograph. Technology also saves time and resources to create diverse and rich visual content. This presents a promising potential for optimising and improving the colouring and design process in the ceramics industry. An excellent example of the capacity of this technique can be viewed in Figure 1.2.

This dissertation is done with *Fraunhofer Portugal Research Association* [9], which is developing different systems to address the colouring in ceramic problems using multiple techniques.

1.2 Goals

The main objective will be to utilise Diffusion Models (DM) and its application in image generation. Additionally, a novel approach will be developed to be compared to the state-of-art model. The investigation will also evaluate metrics and techniques to assess whether the synthetic images



Figure 1.2: Colour dishes created by generative AI when asked to draw "Real colour dishes"

closely resemble real plates while maintaining their authenticity with essential characteristics. Moreover, the study will examine the ability of AI-generated images to achieve realism.

The ultimate objectives of this research are as follows:

- Investigate the potential of artificial image generation for product design in the ceramic industry.
- Develop a new approach for an image generative model.
- Evaluate the generated models based on the realism and quality of the images.

1.3 Document structure

Chapter 1 summarises the context of this study and its goals.

The 2 describes the theoretical concepts needed to fully understand the document, mainly about the idea of ceramics and the role of design and image in the industry, along with the project developed by *Fraunhofer*, where this investigation inserts itself. It also presents the theoretical concepts and architecture of the proposed work of generative Artificial Intelligence (AI).

Chapter 3 overviews the state-of-the-art methods used to develop generative artificial intelligence models, including available tools and evaluation methods. More specifically, this study aims to compare various generative AI techniques, explicitly focusing on Generative Adversarial Network (GAN) and DM.

Chapter 4 provides a detailed explanation of the steps taken in developing this work. It covers various aspects, from the hardware utilised to the specific steps and tests conducted throughout the project.

Moving on to the [Results](#) Chapter presents a comprehensive overview of all the results gathered during the work. It includes the findings from experiments, analyses, and evaluations conducted throughout the project. These results form the basis for drawing conclusions and making insightful observations.

Lastly, Chapter 6 presents the conclusions derived from the development. This chapter analyses the techniques, and the results accomplish the purposes summarised in [Section 1.2](#).

Chapter 2

Literature Review

To properly build a generative model that satisfies the needs described in Chapter 1.2, it is necessary to comprehend the domain idea and have scientific and technical knowledge about ceramics, design, Machine Learning (ML) and image processing. This Chapter presents a more detailed explanation of the ideas mentioned so the reader can better understand the following work.

2.1 SmartColor4Ceramics

The ceramics industry faces challenges in accurately reproducing colours and decorations for their customers. *Fraunhofer Portugal* and *Matceramica* are collaborating on the SmartColor4Ceramics [10] project to address this. They aim to develop an integrated software solution using machine learning algorithms to optimise the ceramic formulation process [7], Figure 2.1 demonstrated the steps and the Flow diagram for the implementation of the work. By replacing subjective human expertise, the software will recommend the best combination of pigment and manufacturing parameters, reducing costs and improving colour stability. This research is inserted in this project to achieve the same goal through different ways and possibilities. With the development of deep learning and mainly image generation, this research tries to use this technology to optimise the reconstruction of the colour process and the new possibilities that can create.

2.2 Neural Networks

ML is a subfield of artificial intelligence that involves developing algorithms and models that enable computers to learn from data without explicit programming [11]. These algorithms and models process large amounts of data, use statistical techniques to discover patterns and relationships and use them to make predictions and decisions. ML has the potential to revolutionise a wide range of industries by enabling the automation of complex tasks previously performed by humans [?].

Neural network (NN), a subset of ML, have found extensive applications in numerous domains, such as image recognition, Neural language processing (NLP), and predictive analytics

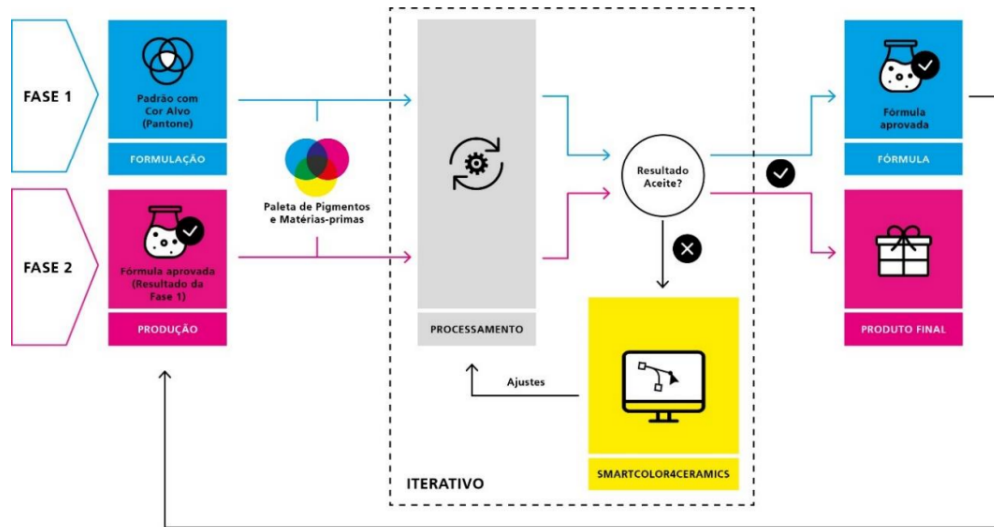


Figure 2.1: Flow diagram during Formulation Production for colour verification using the Smart-Color4Ceramics solution [10]

[12]. They try to replicate the processing and learning abilities of the human brain. With advancements in architecture and computing, NN have become increasingly popular and powerful [13]. These networks consist of artificial neurons arranged in interconnected layers, each playing a specific role in learning [12].

Figure 2.2 shows that NN comprise input, intermediate, and output layers. However, most computations occur in the intermediate layer, where the connections of data points are calculated [12].

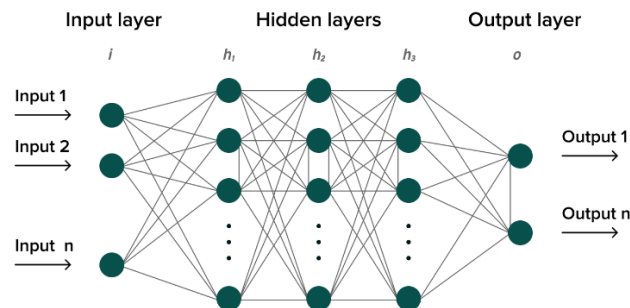


Figure 2.2: Example of a fully connected network [14]

Fundamentally, NN rely on the concept of neurons inspired by the biological ones in the human brain. A neuron receives inputs from other artificial neurons or input data. It computes a weighted sum of these inputs, considering the strength of the connections between neurons. In addition, each neuron has a bias value, influencing its activation. The weighted sum and bias value combination produces the neuron's output. Before passing to the next layer of neurons, the output goes through an activation function, introducing nonlinearity into the network and enabling it to

learn a broader range of patterns from the input data [15]. Figure 2.3 visually represents these neurons.

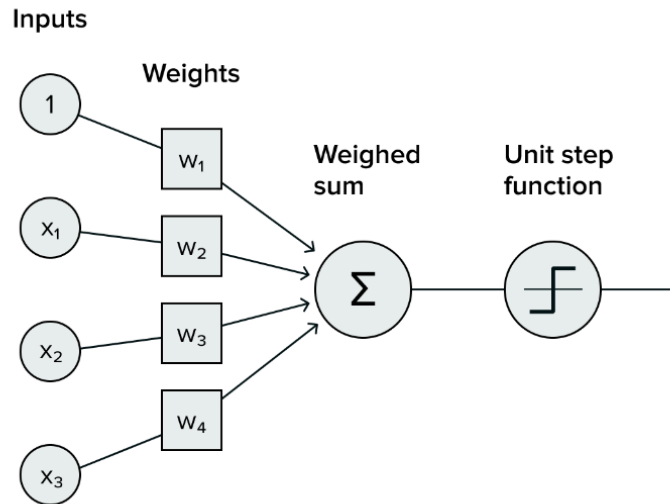


Figure 2.3: Components of an artificial neuron [14]

The neurons in Figure 2.3 combine with other neurons to form the NN. The input layer receives the data that the hidden layers analyse, and through processing, they extract the patterns [15].

Regardless of the layer type, activation function, or network architecture, NN always employ backpropagation as part of their learning process [12]. Backpropagation can be compared to a negative feedback loop, where the model attempts to produce an output for a given problem. Then the training algorithm iteratively adjusts the weights and biases in reverse order, propagating through the layers to minimise the output error. Typically, this tuning is achieved by computing the gradient of the loss function across the different layers, starting from the last layer and moving backwards to the first layer [16]. The math behind NN can be examined in [17].

The learning rate plays a crucial role in the model's weight adjustments during training. It determines the extent to which the model responds to errors [17]. It can be modified before or during training. A high learning rate may cause the model to overshoot optimal weights, while a low learning rate may result in slow convergence. Tuning the learning rate finds a balance between training speed and accuracy. Changing the learning rate during training is expected, starting with a higher rate to generalise important features and progressively lowering it to focus on less-generalised features, preventing the "unlearning" of previous knowledge.

The batch size also affects performance during training and the outcome. This parameter determines the number of samples processed in each training iteration [17]. Larger batch sizes speed up training but can hinder the impact of other hyperparameters like the learning rate. Smaller batch sizes may require more iterations to converge but can enhance the model's generalisation

ability [17]. Choosing an appropriate batch size depends on the available memory, as the batches must fit.

The number of epochs or steps is another hyperparameter that can be adjusted to enhance NN performance. An epoch refers to a complete pass through the entire training dataset in the network [17]. Increasing the number of steps allows the model to capture more complex relationships in the data but may lead to overfitting.

NN performance heavily depends on the data they're trained on. Inadequate or biased training data can disable a model. In contrast, excessive training can lead to overfitting, where the model identifies patterns or relationships that are irrelevant or ineffective in achieving the desired outputs [18].

2.2.1 Natural language processing

NLP is a subfield of artificial intelligence and computational linguistics that developed computational algorithms on the interaction between computers and human language. Its main objective is to enable computers to comprehend and generate natural language input and output it in a way that resembles human communication [19].

Using ambiguity in human language is one of the biggest challenges in NLP. Ambiguity refers to the fact that a given input can have multiple interpretations, depending on the context and the writer. In addition, the use of synonymy increases semantic variability through the usage of different terms with roughly identical meanings.

2.2.1.1 Tokenization

Tokenization is an essential step in NLP, where text is broken down into units called tokens and then passed to numerical values. Tokens are numbers, words and punctuation marks [20]. Therefore, when performing tokenization, it is crucial to consider the interconnectedness of words, domain-specific knowledge, and the proper treatment of morphosyntactic characteristics unique to a language. A trained or rule-based model is required to accomplish this task, incorporating the relevant understanding of the specific domain and grammar rules [21]. Additionally, tokenization algorithms are language-dependent and involve breaking down words into sub-units and converting them into numerical representations.

2.2.1.2 Word Embeddings

After the text is tokenised, these numerical sub-words are passed through the embedding encoder, which employs sophisticated algorithms to transform them into dense vector representations called embeddings. Its primary function is converting the tokens from the tokeniser into meaningful representations, allowing NLP models to understand and analyse text more effectively.

2.2.1.3 Prompt engineering

Prompt engineering is a term used to describe developing and implementing conversational interfaces, which is similar to dialogue with a machine. Usually, the user will use a prompt, watch the result, and then change the suggestion to enhance the development. Thus, prompt engineering is iterative, and the users formulate prompts as probes into the generative models. Since language is free-form and open-ended, there are countless ways to generate images from text prompts.

However, this has the disadvantage of turning image generation into trial and error [22]. To generate an image, it is necessary to create a new text prompt, a process that can seem random and unprincipled. This issue is called prompt engineering in NLP [23], the proper search for prompts retrieving desired outcomes from language models. The two most significant limitations of prompt programming are the difficulty of creating a prompt for a specific type of task and the lack of automated methods [24].

While there have been several studies on prompt engineering for text generation purposes, there needs to be more research done on how users might prompt generative frameworks with natural language for graphic production purposes. So far, progress on prompt engineering for visual tasks has been made informally and on something that has been formed or used for a single, immediate purpose without prior planning.

The textual input prompts must be in a specific format to be effective. The word's order in the prompt can change the output dramatically, meaning it's important to carefully select and compose sentences to achieve a particular visual style in the synthesised image [25] and focusing on the subject and style keywords is more important than connecting words [26]. In terms of the quality of the image, some words have a significant impact. During training, images are accompanied by adjectives that describe the quality of the image. Using these in the prompts can drastically change the quality of the picture. These can be positive (high quality, beautiful, detailed, masterpiece) or negative (imperfect, horrible, awful).

2.2.2 Autoencoder

Autoencoders are a class of NN that work similarly to compression algorithms. It takes some input with very high dimensionality and tries to compress data into a smaller representation. In other words, the basic idea behind this is to learn a compressed representation of an input dataset such that the network's output can be used to reconstruct the original input [27]. The primary objective of a dimensionality reduction method is to identify the best encoder/decoder pair within a given family.

Working with large datasets may be challenging, especially when processing and analysing them. Autoencoders are helpful in this situation [27]. The compressed data is easier and faster to deal with since it is substantially smaller than the original data. This can be very helpful given the enormous data sets involved in image and audio processing operations. Once the task is complete, the autoencoder can decode the compressed data into the original data format.

2.2.2.1 Autoencoder architecture

The encoder and the decoder are the two main components of the autoencoder. The encoder transforms high-dimensional input data into a compressed representation in a lower-dimensional latent space. The encoder, which comprises several layers, converts the input data into a format that can be processed by the rest of the [NN](#). The bottleneck, the desired size for the latent space representation, is reached by the encoder after it performs a series of nonlinear transformations to the raw input data, gradually lowering the data's dimensionality. This reduction is made either by selection or extraction, leaving only some existing features and creating a reduced number of new features based on the old ones [\[27\]](#).

Capturing the most important aspects of the input data while excluding noise and unnecessary information is one of the encoder's primary objectives [\[27\]](#). This is achieved by learning a set of weights that minimise the reconstruction error between the original input data and the compressed representation in the latent space. Therefore, the overall autoencoder architecture would create a data bottleneck that would only allow the main structured portion of the information to pass through and be reconstructed.

The decoder is responsible for transforming the compressed representation in the latent space back into the original input space and reconstructing the original data from the lower-dimensional representation. The decoder comprises several layers to convert the compressed representation in latent space into a format that can be compared to the original input data. The decoder uses the compressed representation to gradually increase the dimensionality of the data until it reaches the input data's original size through a series of nonlinear operations. With the compressed representation in the latent space, the decoder's principal objective is to recover the original input data as precisely as possible. This is achieved by learning a set of weights that minimise the reconstruction error between the original input data and the reconstructed states. By reconstructing the original data, from the compressed representation in the latent space, the decoder can help recover the important features of the input data captured by the encoder. The architecture is represented in [Figure 2.4](#).

To train an autoencoder, it's necessary to specify the number of layers and the activation functions used in each layer and the bottleneck size. These parameters must be tuned to find the optimal configuration for the given dataset. After defining the architecture of the encoder and decoder networks, to ensure that the output of the decoder closely resembles the original input, the encoder and decoder networks are trained together end-to-end and involve optimising the weights of the encoder and decoder networks.

To compute the loss function, it is necessary to look at the reconstructed version at the end of the decoder network and then rebuild the loss with the respective input. Simply comparing pixel-to-pixel differences in the output makes it possible to create a loss function. Two popular loss functions are Mean squared error loss ([MSE loss](#)) and L1 Loss ([L1](#)). [MSE loss](#) is commonly used when the goal is to minimise the overall reconstruction error. In contrast, [L1](#) is often used when reducing the number of outliers in the reconstructed data.

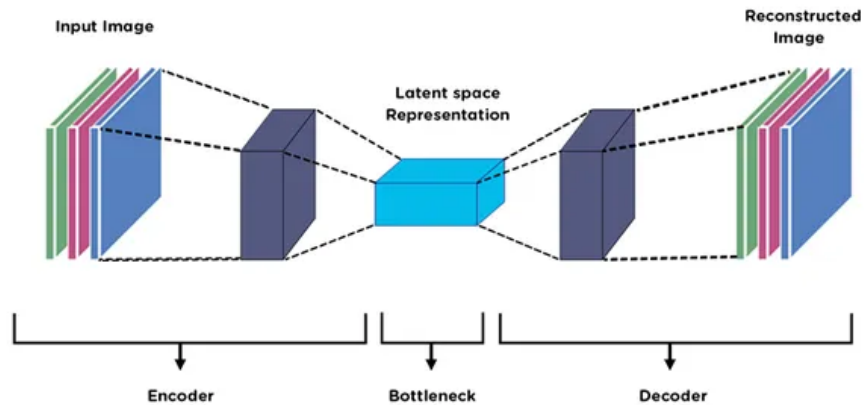


Figure 2.4: Autoencoder Architecture [28]

2.2.3 Segmentation

Image segmentation is a crucial task in computer vision that involves dividing an image into meaningful and distinguishable regions or objects. This process is fundamental in various applications, including object recognition, tracking, detection, medical imaging, and robotics. With the advancement of deep learning, image segmentation techniques have significantly improved accuracy and efficiency [29].

Image segmentation divides an image into meaningful and homogeneous regions or objects based on their inherent characteristics, such as colour, texture, shape, or brightness. By assigning a standard label to pixels belonging to the same category, image segmentation simplifies the representation of it, making it easier to analyse [29], as shown in Figure 2.5.

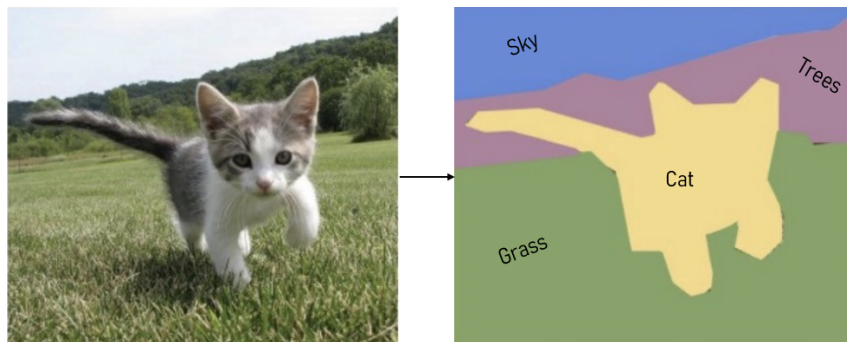


Figure 2.5: Segmentation of the image. The animal is classified as a cat and separated from the rest of the background [30]

2.2.3.1 Segment Anything

The Segmentation Anything project [31] aims to democratise image segmentation, a fundamental task in computer vision. It introduces the Segmentation Anything (SAM) and the SAM 1-Billion

mask dataset (SA-1B), the largest segmentation dataset to date.

SAM is able to do so through a design inspired by **NLP** techniques. It combines an image encoder, a lightweight encoder for prompt conversion, and a lightweight decoder for mask prediction. It allows flexible integration with other systems and can handle interactive and automatic segmentation tasks.

To train **SAM**, the Segment Anything team built the SA-1B dataset using a data engine that combined interactive annotation by human annotators with automatic annotation. The dataset comprises over 1.1 billion high-quality and diverse segmentation masks collected from about 11 million images. It surpasses previous segmentation datasets in scale and quality. The photos were sourced from multiple countries, ensuring better representation across regions and demographics.

2.2.3.2 Evaluation

Object detection and image segmentation are key in computer vision applications. Evaluating the accuracy of these tasks is crucial for developing robust algorithms. This is where **IoU** comes in. **IoU** is a quantitative metric used to evaluate algorithms to assess the alignment between predicted and ground truth regions of interest or bounding boxes [32]. To use this metric, the algorithm must have the ground truth annotation, which is a mask or a bounding box, and the model's predictions must be the same size as the ground truth ones.

IoU is an intuitive metric, and it is computed by determining the ratio of the intersection area between the predicted and ground truth regions to the union area of these regions. Figure 2.6 demonstrates how the calculation is done in a simplified way. It can be mathematically expressed in Equation 1.

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}} \quad (1)$$

The resulting **IoU** value ranges from 0 to 1, where higher values indicate a more significant overlap and a better result. **IoU** is extensively used as an evaluation metric in object detection [33] and image segmentation tasks [29].

Moreover, **IoU** can also be a valuable evaluation metric for generative models that aim to generate images with the same shape and positioning as the original. In this context, the **IoU** can measure the degree of similarity between the generated image and the reference image concerning its spatial alignment, helping to measure the accuracy and ability to reproduce objects' original shapes and positions faithfully.

2.2.4 Convolutional neural networks

Convolutional neural network (**CNN**) are a subset of deep learning **NN** that have revolutionised computer vision tasks, particularly in pattern recognition and image analysis. The strength of **CNN**

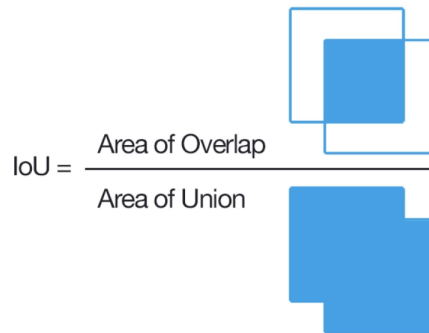


Figure 2.6: Example of calculating **IoU** computation, dividing the area of overlap between the bounding boxes by the area of union [34]

lies in their ability to extract meaningful features from input images by employing specialised filters and leveraging matrix multiplication techniques [35].

The convolutional layer is responsible for convolving learnable filters over the input image. As these filters slide across the input, they perform localised convolution operations that extract relevant features. The output of this layer is a collection of feature maps, each representing a transformed version of the input image [36], as seen in Figure 2.7.

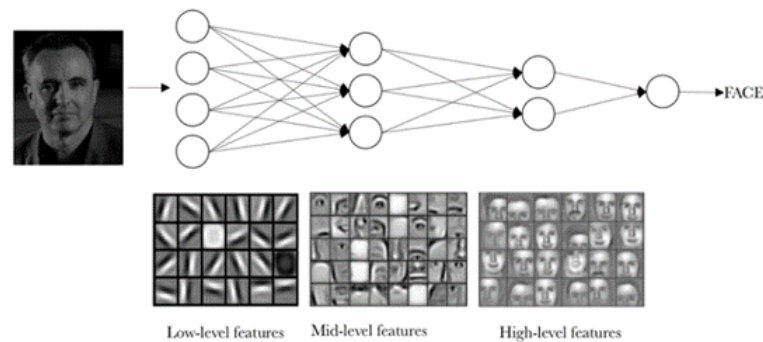


Figure 2.7: Caption features with convolution layers in **CNN** [36]

The convolution layer applies a set of learnable filters to the input. The output of this layer is a set of feature maps, which are transformed representations of the input image. The activation layer introduces nonlinearity to the model by applying an activation function to the output of the convolutional layer. Next is the pooling layer, which downsamples the feature maps produced by the convolution layer, by taking the maximum or average value in a small region of the feature map. This reduces the spatial size of the features while preserving the necessary information [35]. Figure 2.8 demonstrates an example of **CNN** transformation.

CNN can be customised to perform specific functions. U-Net is a **CNN** architecture developed for biomedical image segmentation [38]. U-net is a particular architecture designed for image segmentation, specifically semantic segmentation. In other words, semantic segmentation assigns a label to each pixel in an image, creating a mask showing where that specific object is located in

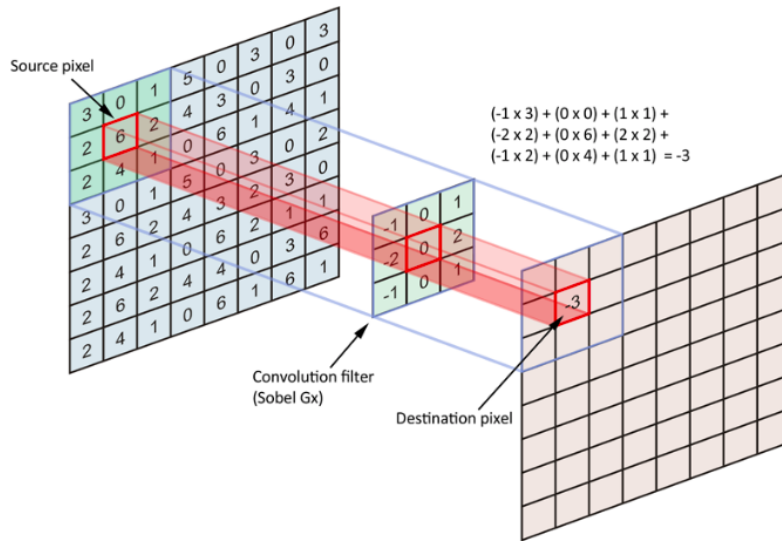


Figure 2.8: Convolution layer transformation [37]

the image [39]. For example, in a picture of a table, semantic Segmentation might be used to label each pixel as belonging to a plate, a glass, or to the table itself.

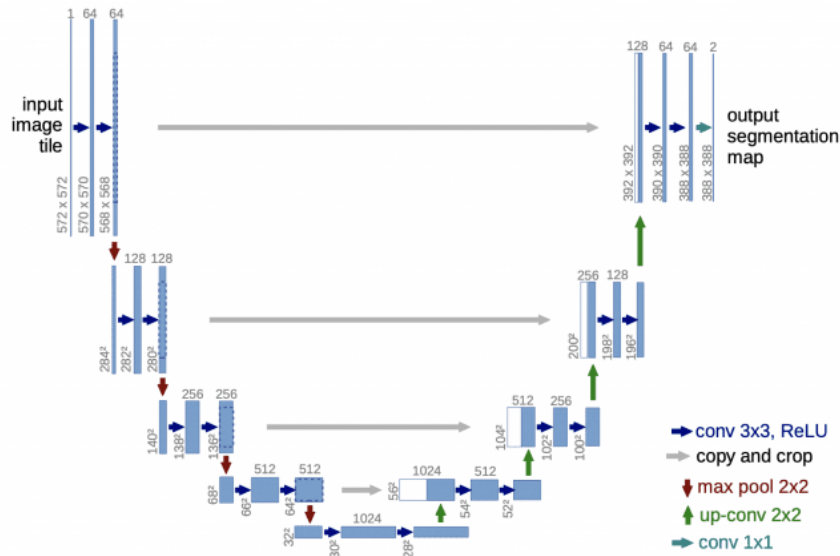


Figure 2.9: U-Net architecture [40]

The U-Net architecture, represented in Figure 2.9, is a NN consisting of three main components: the encoder, the decoder, and skip connections. The encoder's purpose is to extract relevant features from the input image. It achieves this through convolutional and max pooling layers, progressively reducing the input size while increasing the channels. At the lowest level of the "U" shape, the encoder reaches a bottleneck layer that captures the most abstract features of the image, as shown in Figure 2.9.

On the other hand, the decoder is responsible for reconstructing by utilising convolutional

layers followed by upsampling layers. The upsampling operation increases the spatial resolution of the feature maps while reducing the number of feature channels, resulting in an output size matching the input.

A vital feature of the U-Net architecture is skip connections, and this can be seen in Figure 2.10, which allow the direct transfer of selected features from the encoder to the decoder. These skip connections skip some layers in the network and feed the output of one layer as input to the subsequent layers. This transfer of information assists the decoder in generating a more accurate segmentation mask [41].

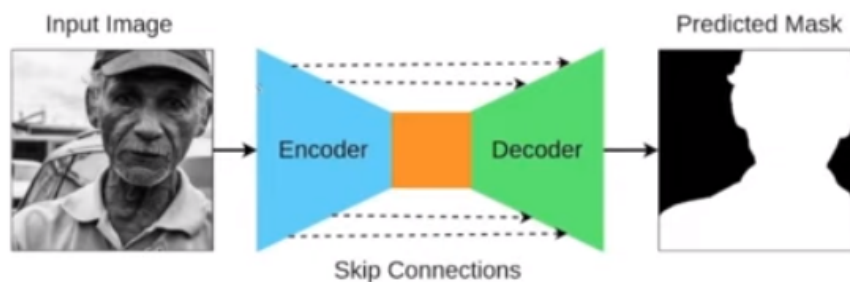


Figure 2.10: U-Net architecture focusing in skip connections [40]

U-Net has proven highly effective, especially in scenarios with limited training samples [38]. Its architecture, with the encoder-decoder structure and skip connections, facilitates precise Segmentation of objects or regions in images, making it particularly suitable for biomedical image analysis and similar tasks. This CNN is the central part of the generative model chosen in this work.

2.3 AI-based product design

2.3.1 Ceramics

Ceramics are versatile inorganic and nonmetallic materials created through moulding and firing clays at high temperatures. They have many applications, from decorative tiles to high-tech materials for electronics and engineering. Pottery, a popular use of ceramics, has been an integral part of human culture for centuries. While it lost some popularity with the emergence of metal and plastic, pottery remains widely used for dishes and cooking. The ceramic industry is environmentally friendly, adhering to regulations and promoting energy efficiency. Ceramic products are recyclable and have a long lifespan. Recycling waste generated during production contributes to a circular economy and a more sustainable future [42].

2.3.2 Product design

In today's highly competitive global market, product design is crucial in attracting customers and gaining a competitive edge [43]. Designing products that stimulate positive emotions and

reflect individuals' lifestyles is a growing trend [44]. Aesthetics, uniqueness, and emotional appeal are essential factors that differentiate products and drive consumer interest [45]. For ceramic designers, incorporating humanized design concepts and considering consumer preferences allows them to create personalized and visually engaging products that express taste and personality [43].

2.3.3 Colour spaces

Images are composed of pixels, the minor units of an image. Each pixel is assigned a specific value, determined by the colour space used to represent the image and has a particular location or set of coordinates arranged in an array or grid. Typically, a set of values corresponding to the colour space represents each pixel's colour value. For instance, in the **RGB** model, each pixel is represented by a combination of red, green, and blue values that determine the pixel's overall colour. Different colour spaces have varying gamuts or ranges of colours they can represent [46].

2.3.3.1 RGB

The most common and well-known model is **RGB**, which stands for "red, green, blue". Its foundation is the theory of additive colour mixing, which allows the creation of a wide range of values by combining different colours. **RGB** combines red, green, and blue light in varying intensities. Each colour channel in the model has a range of values from 0 to 255, as presented in Figure 2.11, with 0 representing no intensity and 255 representing maximum intensity and setting all three channels to their total weight of 255 results in white light. The result will be black if all three tracks are set to 0. Changing the intensities of the three channels can create all the colours the human eye sees [47].

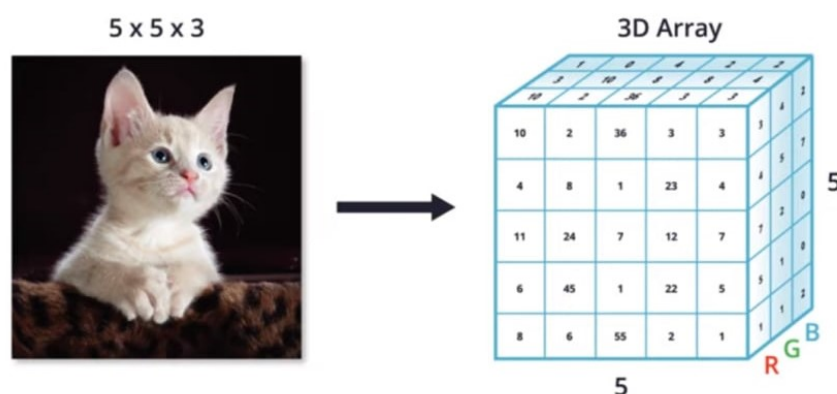


Figure 2.11: Representation of a matrix with **RGB** colour space [47]

It is widely used in digital images because it can faithfully reproduce various colours. It is also relatively easy to implement, making it a popular choice for many applications. The model is only suitable for some applications, and it may not be ideal for printing or professional design work where colour accuracy is essential.

2.3.3.2 CIELAB

It was created in 1976 by the International Commission on Illumination. The Lab* colour space is defined by the L^* , a^* , and b^* , where L^* represents perceptual lightness (ranging from 0 to 100), a^* represents the distance of the colour from red/green (running from -128 to 127), and b^* represents the distance of the colour from blue/yellow (ranging from -128 to 127). Moreover, they are represented by three-dimensional coordinates (L^* , a^* , b^*) in the Lab* space [48], which is illustrated in Figure 2.12.

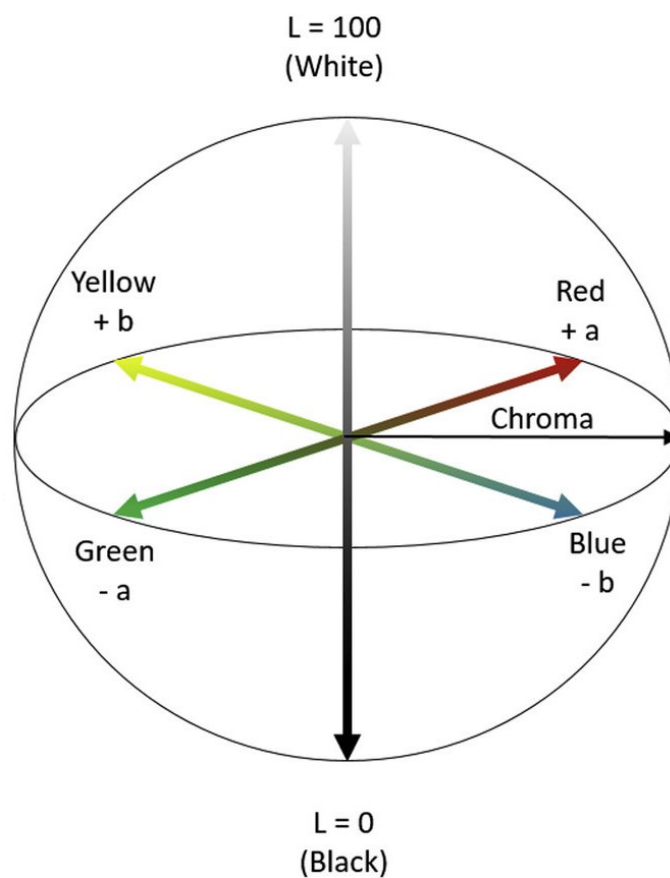


Figure 2.12: Representation of Lab colour space [49]

This is considered a perceptually uniform model, meaning that the differences between the values in the Lab* space are proportional to the differences the human eye perceives. This contrasts with the RGB, where colour differences are not necessarily proportional to the human eye's perceived differences.

However, it does have some limitations, such as being a three-dimensional colour space. This can make processing images in Lab* more complex. Despite this, the Lab* colour space is still widely utilised due to its high accuracy and precision. This model is commonly used in fields such as industry.

2.3.3.3 Delta E

When working with colours, it is essential to ensure that the intended colours are reproduced consistently across different platforms. In 1976, International Commission on Illumination (CIE) created a formula for assessing and quantifying the accuracy of colour reproduction. Delta E is a reliable method to assess and quantify the accuracy of colour reproduction. Delta E, often represented as *DeltaE*, is a metric that measures the difference between two colours. It provides a numerical value and is measured on a scale from 0 to 100, where 0 is exactly the same colour, and 100 indicates precisely the opposite [50].

- ≤ 1 : Not perceptible by human eyes.
- 1 - 2: Perceptible through close observation.
- 2 - 10: Perceptible at a glance.
- 11 - 49: Colors are more similar than opposite.
- ≥ 50 : Colors are more opposites than similares.

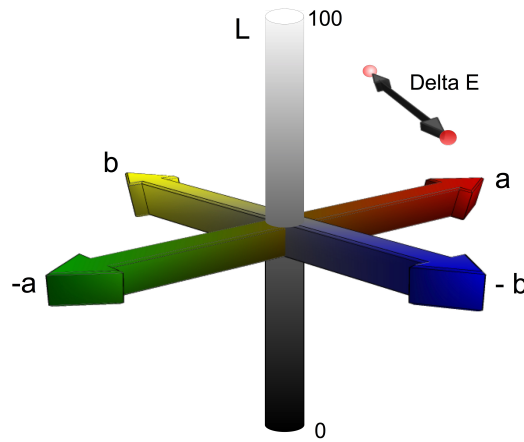


Figure 2.13: Example of delta E as the distance between two colours [51]

Image 2.13 demonstrates how ΔE is calculated in the Lab space. Delta E is based on delta L^* , Delta a^* and delta b^* colour values. The 1976 formula was the first to be released and will be used in the following work. The CIE76 colour difference formula is defined as shown in 2,[52].

$$\Delta E_{ab}^* = \sqrt{(L_2^* - L_1^*)^2 + (a_2^* - a_1^*)^2 + (b_2^* - b_1^*)^2} \quad (2)$$

In conclusion, Delta E is crucial for evaluating colour accuracy and ensuring consistent colour reproduction, providing a standardised measurement of colour.

2.3.4 Generative product design

The use of generative design algorithms in product development has the potential to revolutionise the industry [53]. Traditionally, design teams approach complex optimisation problems iteratively, but this process is time-consuming and expensive, often resulting in suboptimal designs [53]. Generative AI-powered design systems can explore many design possibilities quicker [53]. Several methods have been tested in the last few years concerning the actual applications of generative models in design. This section presents related work on product development and innovation in artificial creation.

Generative design is a technique that permits you to generate and explore numerous design solutions while maintaining a consistent style. It allows for exploring various layout opportunities and answers that conventional processes could not discover. Even though human involvement is still an indispensable element of the creative design process, the field of generative art aims to improve the process of designing and optimising solutions. Although GAN have demonstrated excellent overall performance in image creation and editing, they have been the pioneer on this topic, already existing same studies that have attempted to create a theoretical framework for adopting product design. In the enterprise, GAN has displayed that it is possible to generate a brand-new product created by merging existing products.

The study [54] train the generator with different types and car brands to create a new format similar to the existing ones, suggesting to designers operating for brand-spanking new marketplace segments. Another example is the [55], which uses CGANs to generate customised shoe designs from the dataset UT-Zap50K, which has 50,025 images of shoes categorised by functional type, which allows a wide range of designs and concepts. And for examining numerous colours, textures, and styles to be carried out to simple cartoon snapshots of style merchandise primarily based totally on traits entered as text [56]. It is possible to change a product's colour by applying colour or a pattern in a small part of the product, and the AI can fill the rest, resulting in a new way of testing and verification.

A study on consumer behaviour [57] shows that participants were far more interested in AI-generated designs than professionally made products and were willing to pay more. However, these participants preferred to know the product's creator in advance. If the mention of AI was hidden, they felt tricked and uncomfortable, which means informing consumers about the product's creation in advance is better.

2.4 Summary

In this chapter, the necessary information is provided to comprehend the scope of the work, along with a discussion on applied techniques, specifically focusing on the key components of the generative model. Subsequently, the following chapter delves into the existing approaches utilized in generative models.

Chapter 3

Generative models for product development

This chapter introduces the main methods for developing and assessing generative models. The initial sections will explain the fundamental concepts involved in constructing the generator, aiming to provide a comprehensive understanding of the capabilities of these models and the techniques employed in this field. In addition, evaluation methods are also discussed at the end of this chapter.

Generative models are a technology that uses AI methods to create and generate new data, which can be text, images, or other media in response to common language requests [58]. They use generative models to sample new data based on the training dataset [59]. This State-of-the-art models require massive computational resources and are trained on amounts of data. One of the main motivations of the generative model development is to produce software qualified for handling Turing's Imitation Game [60], which tests a machine's ability to reveal intelligent behaviour equivalent to or indistinguishable from a human.

Generative AI systems include text-based systems such as ChatGPT and Bard, as well as artificial intelligence art systems including DALL-E [61], Make-a-scene [62], DALL-E 2 [63] and Imagen [64], these images generative model will be the main focus on this research.

Recently, significant progress has been accomplished in this direction with the emergence of Deep Learning (DL) models [65]. Several models already have great results in this function, such as GAN [59] and, recently introduced, DM [66]. Typically, these image generative models follow the architecture of Figure 3.1.

3.1 Generative adversarial networks

Ian Goodfellow first introduced GAN in 2014 [59]. The GAN architecture consists of two neural networks working together: generator and discriminator. The generator and discriminator are trained simultaneously like a two-player game, where they compete. The generator tries to generate samples with the most realistic synthetic images possible that the discriminator cannot

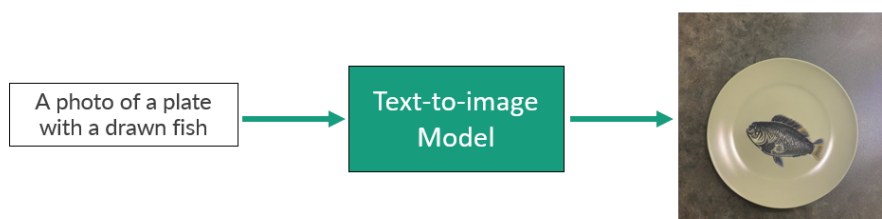


Figure 3.1: Example of generative architecture using text-to-image

distinguish from actual examples. The discriminator also tries to identify whether a given sample is real or fake correctly. During training, the dataset is never shown to the generator, only having the gradient of the discriminator as input. The process of training GAN involves optimising both the generator and discriminator. Ideally, the generator reaches an equilibrium point when the generator fully learns the training set, and the discriminator cannot tell if the input image is real [59]. In Figure 3.2, it is possible to see the architecture of GAN.

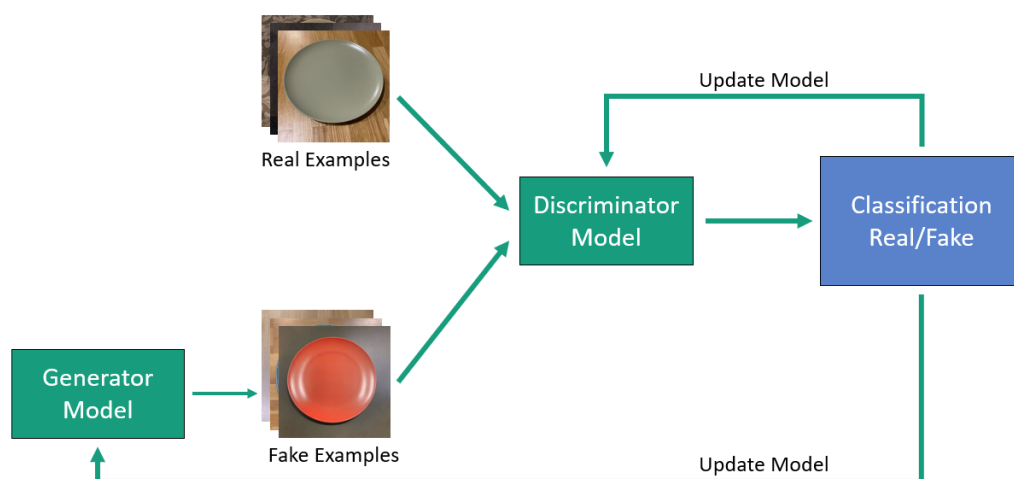


Figure 3.2: The architecture of Generative Adversarial Networks [67]

These networks show that they are capable of generating artificial images. For this reason, GAN is one of the state-of-the-art models with the most image-generation tasks. [68, 69]. Several variations of the GAN developed since its introduction have been designed for more specific tasks.

Although this model is the most common, a new one called the diffusion model was developed recently. It performs better on some tasks and has recently gained much attention.

3.2 Diffusion models

In 2015, Sohl Dickstein's publication [70] presented a new framework for deep unsupervised learning, the DM. These models are inspired by a fundamental principle of thermodynamics that describes how molecules diffuse from a dense region to a less dense area, increasing entropy. Diffusion models have achieved great success in image generation [71, 72, 73], outperforming GAN

in fidelity and diversity in image synthesis, providing a wider variety of images [74]. Additionally, compared to GAN, they have demonstrated that they resist mode collapse, which happens when the generator over-optimises for a particular discriminator, resulting in a small set of output types.

Essentially, they define a sequence of Markov diffusion steps by continuously adding Gaussian noise and then trying to reconstruct the image by inverting this methodology [66]. The fundamental idea, inspired by nonequilibrium statistical physics, is to systematically and slowly destroy the structure in a data distribution through an iterative forward diffusion process. Then learn a reverse diffusion process that restores the system in data, yielding a highly flexible and tractable generative model of the data [75]. It does this by subtracting the estimated noise from the original image and repeating this process a few times.

3.2.1 Diffusion models approach

A Diffusion Model, as shown in Figure 3.3, comprises a forward process, represented as X_0 to X_T , in which a picture is gradually noised, X_T , and a reverse function, defined as X_T to X_0 , in which noise is transformed back into a sample from the target distribution [76].

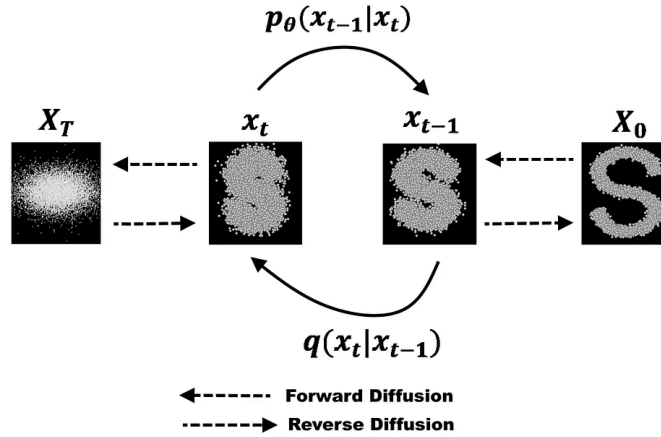


Figure 3.3: Architecture of diffusion model [76]

Forward diffusion process

The forward diffusion process consists in adding a small amount of Gaussian noise to the sample in T steps in the data point $X_0 \sim q(x)$, producing a Markov chain of latent variables of noisy samples $x_1, \dots, x_T$, as seen in Figure 3.4. In image processing, Gaussian noise is made by adding random variations with a Gaussian distribution to the intensity values of the pixels in an image. Combining this with the Markov chain, the conditional probability density can be computed as in Equation 3, which can be simplified in Equation 4.

$$q(x_t|x_{t-1}) = N(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t I) \quad (3)$$

$$q(x_{0:T}|x_0) = q(x_0) \prod_{t=1}^T q(x_t|x_{t-1}) \quad (4)$$

The hyperparameter b ensures that x_T is nearly an isotropic Gaussian for sufficiently large T [77]. The data sample x_0 gradually loses its distinctive features as the step size t becomes larger when $T \rightarrow \infty$ is equivalent to an isotropic Gaussian distribution [78].

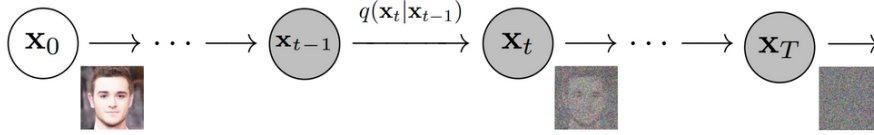


Figure 3.4: Architecture of the forward process in diffusion models [77]

Reverse diffusion process

The Reverse Diffusion technique involves training the model to reverse the forward function and generate new data, as illustrated in Figure 3.5. However, achieving this requires efficient computation of $q(x_{t-1}|x_t)$ when working with the entire dataset. By reversing the previous process and sampling, the model can recreate the sample from a Gaussian noise input $x_T \sim N(0, I)$, as depicted in Equation 5, which can be simplified in Equation 6.

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t) \sum_\theta(x_t, t)) \quad (5)$$

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t) \quad (6)$$

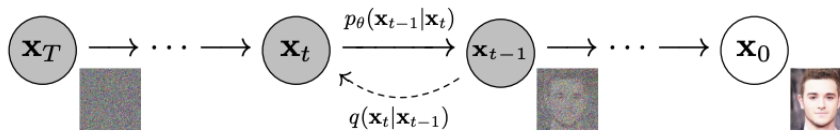


Figure 3.5: Architecture of the reverse process in diffusion models [77]

The slow reverse denoising procedure in diffusion models is a disadvantage. Also, since these models work in pixel space, which becomes excessively expensive when creating high-resolution images, they take a lot of memory [79]. Rombach [66] recently introduced latent diffusion models to reduce the computational complexity of training diffusion models. This variation of the diffusion model works on the low-dimensional latent space instead of the original high-dimensional

input, making training more efficient. It uses the same criteria as the diffusion model but changes the encoded input to a latent representation. Using a decoder network, the noise predictor processes the latent representation to produce new data up-sampled back to the original dimensions. Some models are already in use, for example, DALL·E 2 [63] from OpenAI and Imagen [64] from Google. These models have shown efficacy on high-quality synthetic data and may be applied to various tasks.

3.2.2 Diffusion models components

The three main parts of latent diffusion models are the text encoder, autoencoder, and convolutional neural network.

U-Net

To reverse the diffusion process, it is necessary to calculate how much noise is added to the image using a model called U-Net. The noise predictor can estimate the noise in an image from a completely noisy image. The U-Net gradually subtracts noise in the latent space over several steps to reach the desired output. Each step reduces the amount of noise added to the latent representation until the final de-noised output is reached.

The U-Net model takes two types of inputs: noisy latent and text embedding. When creating images from a text description, the noisy latent inputs are generated by the Variational autoencoder (VAE) encoder for image-to-image tasks or pure noise input. The text embedding is based on Contrastive Language–Image Pre-training (CLIP). The model’s main output is the prediction of noise residual, which is used to compute the expected de-noised image representation. This predicted noise residual helps improve the quality of the output image by reducing the noise present in the original noisy latent input. The simplified architecture Figure 3.6 shows how these inputs and outputs are connected.

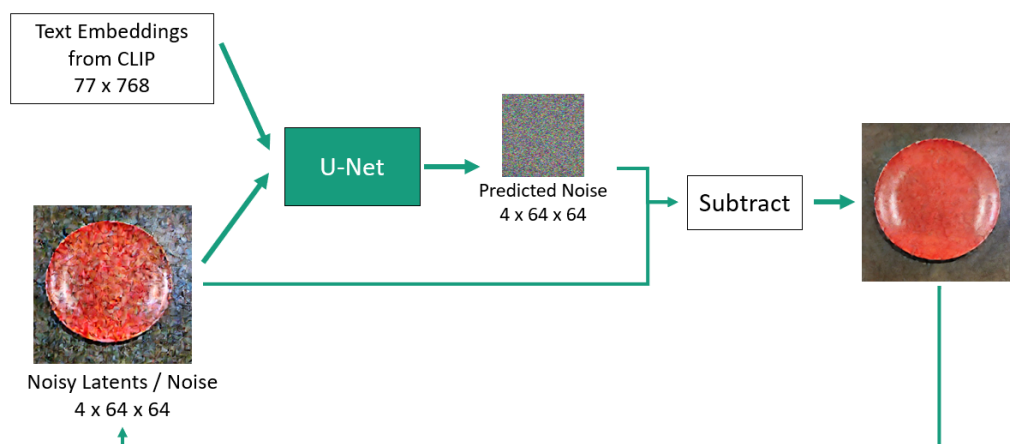


Figure 3.6: A simplified architecture of U-Net

Autoencoder

As a latent diffusion model, the Stable Diffusion (SD) first compresses the image in latent space instead of working in high-dimensional image space. The latent space is 48 times smaller, benefiting from fewer data to reduce the computational power. This is done using a technique called the VAE. The encoder and decoder are two components of the VAE model. The low-dimensional latent representation of the image is generated using the encoder and will be used as input for the U-Net model. Instead, the decoder turns the latent representation into a Figure, 2.2.2.

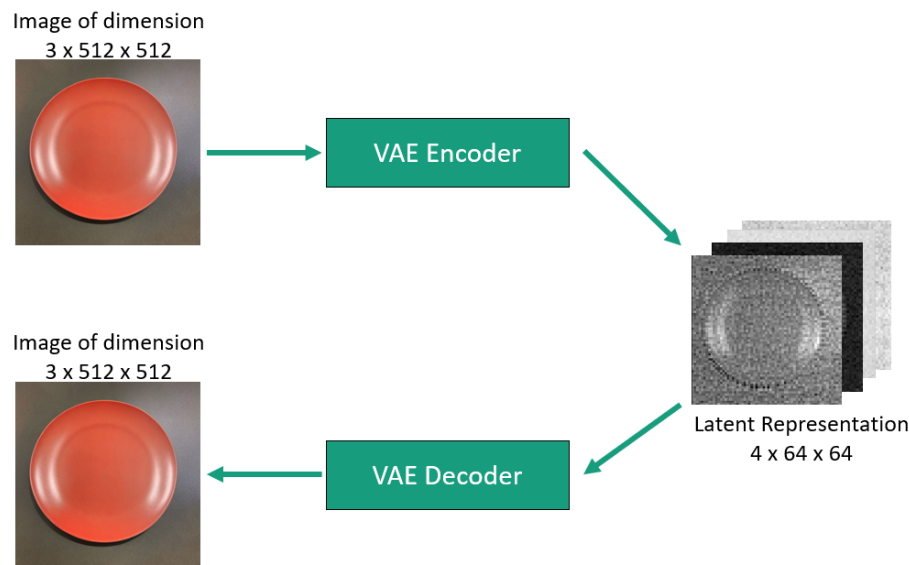


Figure 3.7: A simplified architecture of the autoencoder

In the example Figure 3.7, the encoder compresses the image into smaller dimensions while the decoder recovers the original image from the compressed one. The encoder obtains latent image representations for the forward process during latent diffusion training. In the reverse diffusion process, denoising latent representations are generated, then converted back into images by the VAE decoder during inference. Latent text-to-image models exclusively utilize the VAE decoder.

Text encoder

For the model to be text-to-image, it must have some conditioning to make the resulting image an interpretation of the prompt. This is where the text encoder comes in. The text encoder is responsible for converting the text into a vector representation such that a machine can understand and to a sequence of latent text embeddings, which the U-Net can understand. In simple terms, the tokeniser converts each word in the prompt into a token. Then, each token is converted into a value vector known as an embedding. The embeddings are then processed by the text encoder and ready for use by the noise predictor. During the training phase, the text encoder itself is not trained. Rather, an already pre-trained text encoder is utilised. This enables the utilisation of powerful pre-trained language models without requiring extra training. The utilisation of different

architectures for text embedding is common. For instance, Imagen [64] uses the T5-XXL [80], while SD uses the CLIP ViT-L/14 model [81]. Figure 3.8 clearly understands this architecture.

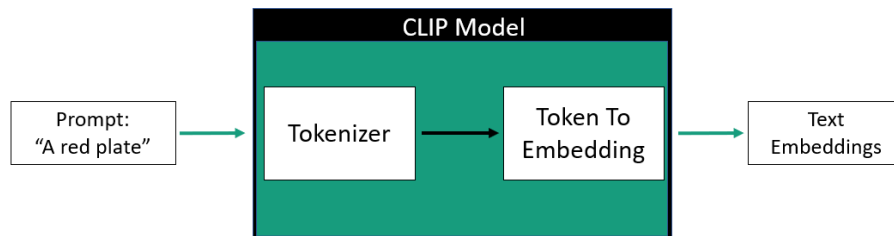


Figure 3.8: A simplified architecture of the text encoder

A scheduler, a crucial component of the diffusion model, is an algorithmic process of adding noise to the latent representation. The purpose of a scheduler is to determine how much noise to add to the latent at a given step in the diffusion process. The Pseudo numerical methods for diffusion models (PNDM) [82] is chosen as the scheduler because it accelerates the inference process while maintaining high sample quality and solving differential equations on manifolds. Using pseudo-numerical methods, PNDM generate higher-quality synthetic images with fewer steps than traditional schedulers [82].

Diffusion models steps

Figure 3.9 shows that the diffusion model receives a text prompt and returns an image in text-to-image. It starts by generating a random tensor in the latent space and can be controlled by the random number generator's seed. The seed is from which the initial Gaussian noisy latent is generated. It's the final image in latent space, but it is all noise. The U-Net takes the latent noisy image and text prompt as input predicts the noise and then subtracts the latent noise from the latent image. This becomes the new latent image. This process is repeated for the number of sampling steps. Finally, the VAE decoder converts the latent image into pixel space, resulting in the final image.

3.2.3 Stable diffusion

Stable Diffusion is a DL technique that generates images from text [66]. It is an open-source model, meaning that the code and weights are publicly available. The model was developed by the CompVis group at LMU Munich and was released in August 2022 in collaboration with Stability AI. It offers versatile functionality, including tasks like inpainting, outpainting, and image-to-image translations guided by a text prompt.

SD operates within the latent space, where input images are transformed via Diffusion and decoded into output images. The decoding process is conditioned on either text or images.

Choosing which model to use depends on the study's specific needs. However, despite the array of new functionalities in version 2.0, none are suitable for the present study. According to

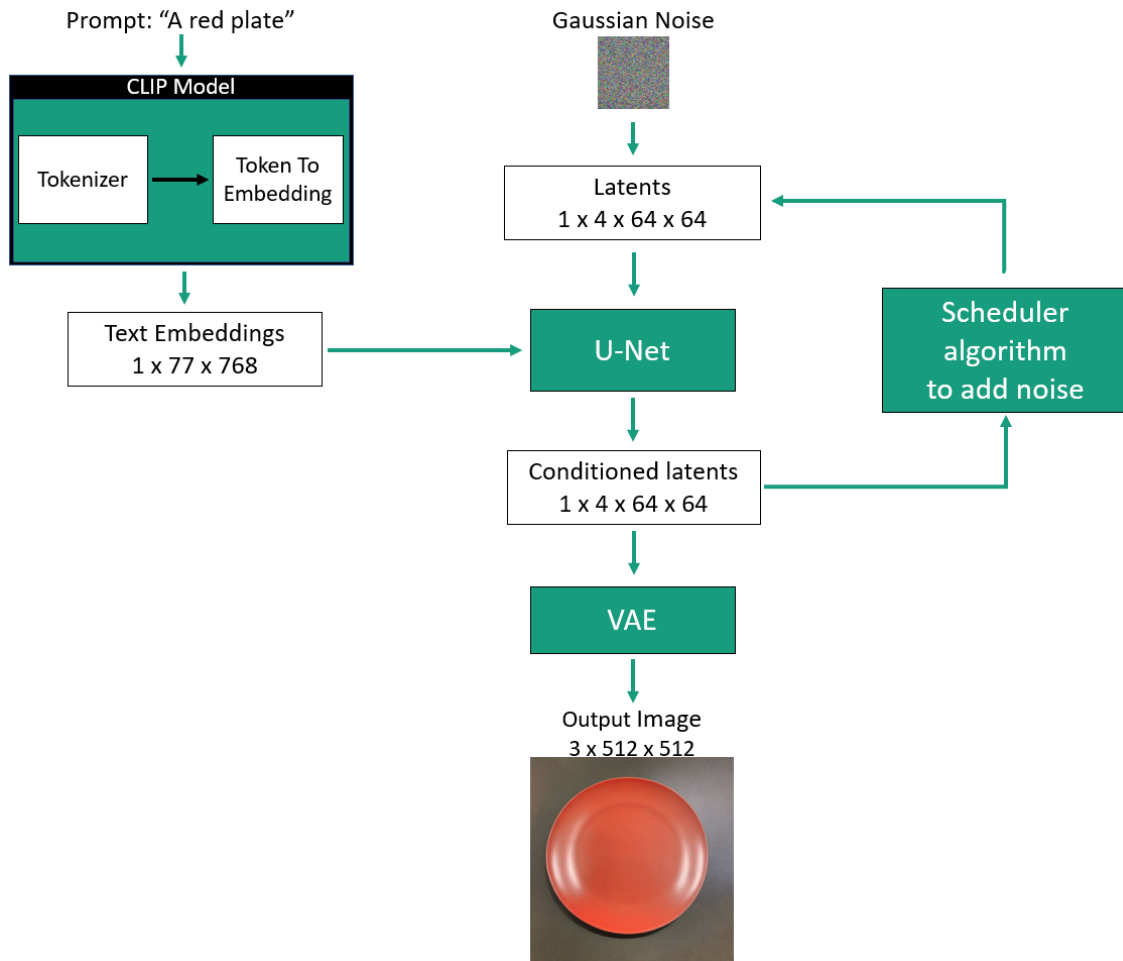


Figure 3.9: A simplified architecture of the model

[83], version 1.5 was reported to offer the most flexibility and its ability to produce the highest quality images overall. This study was conducted on the 1.5.

An additional feature of the SD is the insertion of classifier guidance. Classifier-Free Guidance (CFG) is the setting that controls how closely SD should follow the text prompt [84]. This parameter is applied in text-to-image and image-to-image generation and can vary between values. The higher the CFG value, the more strictly it will follow the prompt. A value of 1 will give SD almost complete freedom, whereas values above 15 are restrictive. The default value is 7, which offers a good balance between following the concept and having independence. It is possible to set CFG as high as 999 or even to negative. A negative CFG means to generate the opposite of the text prompt [85].

Instead of using class labels, CFG uses image captions to guide and train the diffusion model. The classifier component is incorporated into the noise predictor U-Net, achieving "classifier-free" guidance in image generation. CFG uses the text prompt to guide image generation at each step, balancing quality, alignment with the prompt, and input diversity. By calculating the difference between images with and without the prompt, the model knows in which direction to incorporate

the prompt. Adding the scaled difference to the text-less image allows the model to extrapolate towards the text information for prompt-guided image generation [86].

Although it sounds straightforward, using CFG has some inevitable trade-offs: Colour saturation increases as CFG increases; Contrast increases as CFG increases; Above a specific CFG value, output images become blurrier, resulting in a loss of detail. Increasing the number of sampler steps can counteract the decrease in output image quality at higher CFG values. However, this leads to longer processing times. It is essential to find a balance between CFG and simple steps to get the best output images while minimising memory and processing time.

Generative models possess immense potential to generate diverse outputs. However, their knowledge might be constrained. Frequently, the generation process heavily relies on the images encountered during training. While general pre-trained models are highly adaptable, specific applications may necessitate further training. SD is a public and open-source model, has a custom architecture and is supplied with pre-trained model weights to fine-tune the model. Fine-tuning a pre-trained model is an effective way to improve the model's performance for a specific task. It involves using a pre-trained model as a starting point and updating its weights and biases to minimise the loss function until the model's output matches the expected task. Furthermore, fine-tuning large models improves their performance on various tasks. It achieves state-of-the-art results with less computational power and time than training a model from scratch [87].

Diffusion models have created new opportunities for generating images from descriptions in natural language. Yet, these templates could be more adaptable when producing original concept images or altering their appearance. This begs the question of how language instruction models can be used to create representation from user input. One possible solution is to train a SD model to learn a specific concept, as seen in Figure 3.10. There are four ways to achieve this. Understanding them is crucial for building accurate and reliable models that can capture novel ideas in a manner controlled by text-to-image pipelines. The following topics will explore these two methods, Dreambooth and Low rank adaption, and their benefits and limitations.

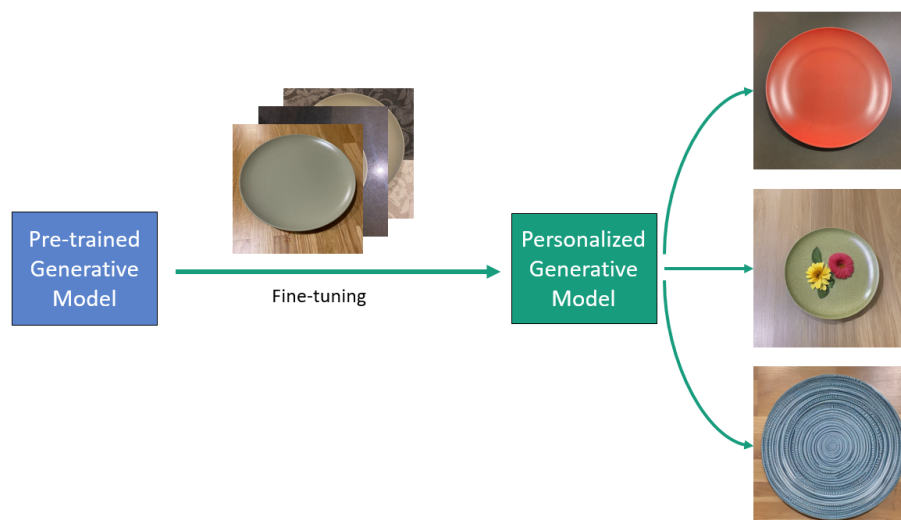


Figure 3.10: A architecture of the personalised concepts

Dreambooth

Dreambooth [73] is one of the most utilised fine-tuning techniques. It injects a custom subject developed by Google. Similar to textual Inversion, Dreambooth aims to use a small number of images to identify a subject uniquely and generate it in the output domain of a model. It does this by fine-tuning the entire model to link a unique identifier with a specific concept, resulting in an excellent visual representation of the subject.

Dreambooth works by altering the structure of the model. It receives two inputs, the concept and the unique identifier. It takes the sentence and converts it into text embedding. Then pass the text embedding, a pretty noisy sample image and another without noise. During training, try to get the model to take the noisy one with the text embedding and output the less noisy one, comparing the two. Create a loss with that comparison and adjust the weights. It fine-tunes the model to be capable of understanding the new concept. To prevent the model from associating a specific instance with a class name, such as "plate", creating a new class with a name that does not exist is recommended.

Dreambooth is flexible regarding input image capture settings and can accommodate variations in the subject's context, such as location, species, colour, shape, stance, expression, material, and other semantic adjustments. This technique can generate more precise results but can only develop specific subjects. This approach is highly effective in achieving its intended purpose, but the downside is that it requires a lot of storage and could be more efficient in terms of storage.

As an open-source tool, Dreambooth has given rise to several variants. One of the variations is from ShivamShrirao [88]. This model can generate high-quality images. To train the model is recommended to use 30 training images of each type. It uses a learning rate of $1e-6$ and mixed precision of fp16. This model has some advantages: primarily, it can generate amazing images relatively quickly, making it efficient. Also, it can train more than one concept, in this case, allowing training with different plates in one model, saving resources and time compared to using other models for each object. It does this by including a JSON list with the names of the tokens and the classes. Furthermore, pausing and resuming the training for more steps during training is possible if the result is not satisfactory. The author suggests that the number of steps equals 100 multiplied by the number of training images. In other words, with 30 photos, the number of steps is 3000.

XavierXiao's approach [89], although one of the first to appear, still needs to be updated, and the results could be better. The author uses "sks" defined in the code to create the token. Like the ShivamShrirao model, it has a learning rate of $1.0e-6$. The advantage of this model is that it requires a small number of steps, 800 steps, but with lower steps, it also performs well.

Joe Penna [90], a feature film director, creates other variations for digital artists to train their styles and characters. It is inspired by XavierXiao's repo [89], specifying to train people's faces. The new prompt must consist of the token, followed by the class. A disadvantage is that this model is inefficient, requiring a Graphics processing unit (GPU) with 24GB of VRAM. This model can easily overfit, resulting in images similar to the training data. To solve this is necessary to retrain the model with fewer steps or use better photos of the concept.

Low rank adaption

Low-Rank Adaption [91], **LoRA** is a novel technique that enhances **SD** models by a method to fine-tune weights for the text-encoder and the U-Net. The cross-attention layers are critical in **SD** models because they are the point where the image and the prompt interact. In **LoRA**, researchers found it sufficient to fine-tune only this part of the model to achieve good training results.

The technique behind **LoRA** involves breaking down the weight matrices of the cross-attention layers into two smaller (low-rank) matrices. This approach reduces the number of parameters stored in the model file and allows for quicker training times. **LoRA** achieves this by inserting new layers into the model and updating only these new layers instead of updating the entire model's weights. It works likewise to Dreambooth, but rather than update the weights in the mode. It updates the new weight on the new layers. The hyperparameter that has the most impact on this technique is scale. This hyperparameter influences the effect of the layer and can vary between 0 and 1, where 0 indicates no tracking, and 1 indicates extended tracking.

Developing diverse versions with various features results from **LoRA** open-source nature. The most promising known version is from cloneofsimo [92]. The author recommends training the model 500 steps per training image, although lower steps also show good results. With ten images, the recommended is 5000. In the Dreambooth case, the learning rate of the U-Net is $1e-6$, but this is way too small for training **LoRA**. The author suggests using $1e-4$ for the U-Net and $5e-5$ for **CLIP**. The U-Net's learning rate in the Dreambooth scenario is $1e-6$, but this is relatively low to train **LoRA**. The author suggests using $1e-4$ for the U-Net and $5e-5$ for **CLIP**. In **LoRA**, it is possible to change the effect of fine-tuning. This means by varying α is possible to vary the weight that the model has in the final result. This model can also support inpainting. The Cloneofsimo model is much smaller and twice faster at fine-tuning **SD** models than the Dreambooth method.

3.3 Evaluation of generative models

As the outputs of the models are images, it cannot be easy to evaluate them. Evaluating images is challenging due to their subjective nature and the complexity of visual perception. Images require a human assessment to test their quality, unlike objective data, such as numerical values. This subjectivity originates from differences in individual preferences, cultural backgrounds, and artistic interpretations. The multidimensional nature of images, encompassing elements like composition, lighting, and colour, further complicates evaluation. Furthermore, contextual factors and intended use must be considered when assessing image quality. The scalability of manual assessment and the variability of human perception pose additional challenges. While objective metrics exist, they often fail to capture higher-level aspects of image quality. Therefore, a combination of subjective human judgment and automated metrics is usually employed to provide a more comprehensive evaluation of images.

The automatic techniques for evaluation were first developed to assess **GAN**. Salimans introduced the Inception Score in 2016 [93], which has become one of the most widely used metrics. The principle behind this is that an excellent generative model should produce images that are high

in quality (realistic) and diverse (cover a wide range of object categories). It quantifies the proximity between two probability distributions [94] by computing the difference between the predicted and marginal class distributions of the generated images.

Another metric is the Fréchet Inception Distance (FID). The FID [95] is often used as a benchmark for the evaluation of the performance of generative AI [96]. This metric works by measuring the distance between entities in the generated images and those of the actual pictures, calculating the comparison of the generated image with a CNN trained to recognise objects in images.

In addition to these, some metrics also consider the prompt. CLIP [97], created by OpenAI, measures both visual similarity and semantic meaning. Based on the view that a produced image should be visually similar to the input text and semantically meaningful. For its use, the generated image is first encoded into a feature vector using CLIP. Then, a set of reference images is selected based on their similarity to the input text, and each reference image is encoded into a feature vector using CLIP. Finally, the Contrastive Language–Image Pre-training - Perceptual Similarity (CLIP-PS) score is computed as the average similarity between the feature vectors of the generated image and the reference images.

These automated asses techniques try to close the gap to human visual review. Therefore, it also should conduct Human-based experiences for a complete picture review. To this end, should be defined metrics and ways of quantifying them. There are various ways to assess the photorealism and quality of an image.

One alternative is that the user is presented with a model generation and a reference image and asked to choose which image looks more photorealistic. Beyond this evaluation, another method is to show real and fake pictures and ask to assess whether they are "real", "indefinite", and "unreal". By presenting real and fake images to the users, this method allows them to assess their ability to differentiate between the two, helping to understand how well people can perceive and identify fake visual content. Including the option "indefinite" responses also provide insights into the difficulty level in distinguishing between authentic and generated images or potential room for improvement. Thus, the image quality assessment should consider human judgement, even though this can be time-consuming.

3.4 Summary

This chapter introduces generative models for product development, focusing on text-to-image generation. It covers vital models like GAN and DM, explaining their basic concepts and evaluation methods. The SD model, its components, and fine-tuning techniques are discussed, including Dreambooth and LoRA. This chapter provides a concise overview of generative models, setting the stage for further exploration in product development.

Chapter 4

Methodology

This chapter describes the available resources, the selected tools and the methodology taken to develop the generative model. Furthermore, the last Section presents how the generator's performance was evaluated and validated.

The project consists of two primary phases. The initial phase primarily involves fine-tuning the [SD](#), and the second is the evaluation methods performed during the work. It is structured as follows:

- Preparing the dataset: The process of acquiring, preparing and selecting the data set to be suitable for the techniques used,
- Fine-tuned model: The approach and algorithms used to create a fine-tuned version of [SD](#) by utilizing the Dreambooth and [LoRA](#) technology and the pretended results.
- Hyperparameter tuning: Perform tests with different hyperparameters to define the best ones that give the finest combination of results.
- Train the combined model: Run and test the new model with the pre-defined parameters.
- Evaluation: The method by which the generated images will be validated to calculate the model performance in qualitative and quantitative terms

4.1 Hardware

Developing and implementing generative models is computationally intensive due to the complex tasks involved. These models require significant computational power for efficient calculations. To meet these demands, the use of [GPU](#) is crucial. [GPU](#) excel in parallel computing, making them well-suited for the distributed computational operations required by generative models. *Fraunhofer's* High Performance Computing ([HPC](#)) infrastructure, including [GPU](#), provided the necessary computing resources for training and inference processes, significantly reducing the time needed for these tasks.

The computational work relies on PyTorch [98], an open-source deep learning framework known for its flexibility and ease of use. PyTorch’s extensive ecosystem of libraries and tools efficiently leverages parallel computing on GPU, resulting in improved model performance. In collaboration with StabilityAI [99], Hugging Face [100] provides pre-trained models with advanced architecture and weights. These models, seamlessly integrated with PyTorch, served as a starting point for the research and facilitated the implementation and fine-tuning of the models in Python. The Stable-Diffusion-v1-5 model [101] was chosen based on its performance and fine-tuning capabilities, as mentioned in an article from StableAi [102].

4.2 Dataset preparation

For preparing the dataset, the recommendations and compatibility of Dreambooth also apply to LoRA. They require the input images to be square with a specific size, 512 x 512, and the pictures that don’t meet the requirement are automatically cropped, losing some essential features.

The images used to train the models were from real plates. It was required to build the dataset by taking pictures of the dishes with different characteristics, specifically, different angles, lights and backgrounds. The photos were taken on an iPhone 11 [103]. After collecting the images, resizing them to the required size was necessary. The Bulk Image Resize (BIRME) [104] is an online tool to resize images to a specific size. Although all photos were in PNG format, the training images can be JPGs or PNGs.



(a) Concept_1



(b) Concept_2



(c) Concept_3



(d) Concept_4

Figure 4.1: Demonstration of the images dataset used

As seen in Figure 4.1 shows the four plates used for the dataset preparation. All chosen plates are entirely different, each with a unique feature. The concept_3 is a black rectangular plate. The concept_2 has a unique style with trees and a lot of colours. On the opposite side, the concept_4 is a simple circular white plate with a wavy texture. And finally, the concept_1 is the most neutral plate, making it the most effective one to manipulate.

4.3 Stable-GAN

This section presents a novel approach that combines SD and GAN, referred to as Stable-GAN. This new model is based on the principle of training GAN, using its discriminator and SD as a generator to improve the quality of the created samples. This new approach aims to test the benefit of using a discriminator to support the training of the SD. The following sections describe the tools selected and the methodology used to develop this.

4.3.1 Model architecture

This involved creating a new generation model by combining the DM and GAN models. The training process of the GAN played a significant role in this integration. Instead of using a typical GAN generator, in this case, the generator is the SD model, while the discriminator remains the same NN.

This new methodology was chosen because the DM outputs show very good results, outperforming GAN generator in some tasks [74]. By leveraging these superior outputs, this model can benefit by producing more accurate and desirable results for specific images. The discriminator has great importance in the guidance of the generator [93]. Training the discriminator with a dataset and the generated images could guide SD to create images more similar to the dataset images.

In this particular study, version 1.5 of the diffusers was selected as the pre-trained model. The SD training process was inspired by the LoRA technique, which focuses on optimizing the cross-attention layers of the U-Net, which is the crucial part where the image and the prompt interact. Incorporating LoRA fine-tuning makes the training process more manageable and not so computationally heavy, as only the layers experience significant adjustments. Consequently, the capacity and performance of the original model are minimally affected, allowing for targeted modifications to enhance specific aspects of its functionality.

To achieve this integration, the generated samples from the SD are input alongside the dataset photos into the discriminator. The discriminator then evaluates the authenticity of these samples and produces corresponding outputs. Based on these, an error is calculated and then used for backpropagation.

In addition to the conventional loss function utilized by U-Net, which typically entails computing the MSE loss between the predicted and actual noise [66], the loss calculation is augmented by including the discriminator's error. By integrating the discriminator's error, the U-Net adapts

its parameters and enhances its capacity to generate samples of heightened realism. This back-propagation process enables the U-Net to acquire knowledge from the discriminator's evaluation and refine its generative capabilities.

This augmented loss function provides a more comprehensive measure of the model's performance, taking into account both its generative and discriminatory aspects.

Figure 4.2 provides a simplified illustration of how this training process is conducted, depicting the integration and information flow between the stable diffusion model and the discriminator.

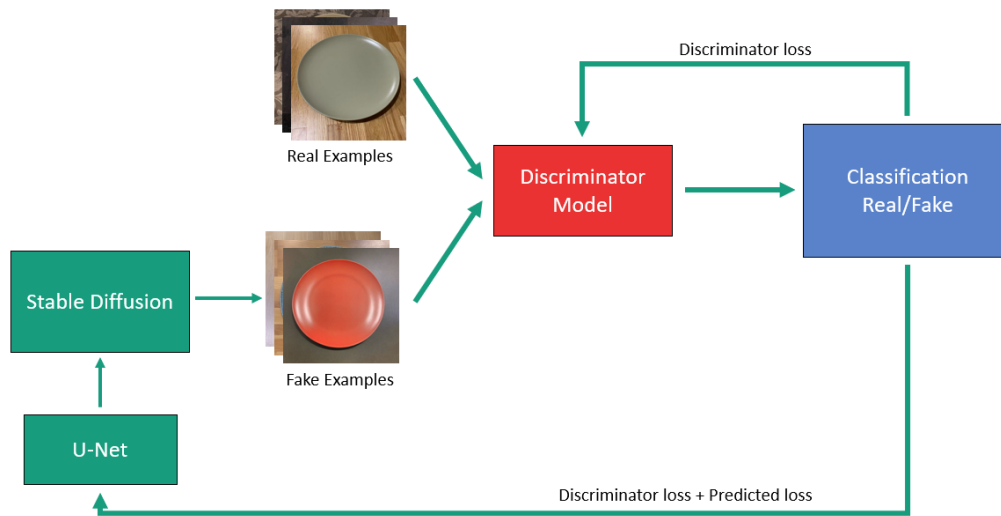


Figure 4.2: Simplify Stable-gan architecture

4.3.2 Discriminator

The discriminator is crucial in distinguishing between actual and generated samples in the generator framework. The discriminator is responsible for learning the underlying patterns and features in the data, allowing it to assess the authenticity of the images.

In Stable-GAN, the discriminator was implemented as a CNN consisting of convolutional and fully connected layers. This architecture is designed to capture hierarchical representations of the input data and extract discriminative features. Inspired by the successful Deep Convolutional Generative Adversarial Networks (DCGANs) approach [105], the discriminator architecture is straightforward to implement and yields excellent results [106].

It consists of multiple CNN that employ chain convolutions, which help reduce the input's spatial dimensions and focus on relevant information. Following each convolutional layer, an activation function, ReLU, is applied to introduce a non-linearity, and batch normalisation is used to improve the stability of the training process.

In contrast to traditional fully connected layers, the discriminator replaces them with global average pooling. This technique summarises the spatial information of the image into a single value, capturing the overall features and preserving the global context. The output layer of the

discriminator consists of a sigmoid-activated fully connected layer, producing a probability value that indicates the likelihood of the input image as real or fake.

To train the discriminator, labels are assigned to the input images. Authentic images are labelled as 1, representing their authenticity, while fake images are labelled as 0, indicating their artificial nature.

During the forward pass, the discriminator evaluates the authenticity of images by passing them through its architecture. The output of the discriminator represents its estimation of the actual photos. A loss function, typically binary cross-entropy, is then applied to compare the discriminator's output with the actual labels, resulting in the computation of the real loss. The mean of the discriminator's output is also calculated, providing the actual score.

Similarly, the forward pass is performed for the fake images, generating the discriminator's output and computing the fake loss. The mean of the discriminator's result for fake images is obtained as the fake score. The total loss is calculated as the sum of the actual and generated losses, reflecting the discriminator's overall performance in distinguishing between real and fake images.

To update the discriminator parameters, backpropagation is applied to calculate the parameter gradients for the total loss. The optimiser then updates the discriminator parameters based on the calculated gradients.

4.3.3 Hyperparameters

To prepare for the model training, specific parameters need to be defined. The Binary Cross Entropy Loss (**BCEloss**) calculates the discriminator's error. This loss function is commonly employed in **ML** and **DL** algorithms, particularly in binary classification tasks [107]. It quantifies the dissimilarity or error between predicted probabilities and target probabilities. In binary classification, each instance belongs to one of two classes. The **BCEloss** function computes the loss by comparing the predicted probability with the target probability for each class. The **BCEloss** function employs the binary cross-entropy formula to calculate the loss, represented by the following equation 7.

$$BCELoss = -[target * \log(predicted) + (1 - target) * \log(1 - predicted)] \quad (7)$$

The optimiser selected was the Adaptive Moment Estimation (**Adam**) optimiser. It is an extension of Stochastic Gradient Descent (**SGD**). It has gained popularity in deep learning due to its adaptive learning rate and efficient convergence properties. Proposed by Kingma [108], **Adam** uses adaptive learning rates based on exponentially decaying averages of past gradients and their squared gradients. **Adam** is widely utilised in deep learning due to its efficient convergence and ability to handle sparse gradients and non-stationary objectives [108]

When training a discriminator with a small dataset, selecting an appropriate learning rate is crucial for achieving practical and stable training. In this case, a learning rate of 0.00001 has been

chosen. With a small dataset, a lower learning rate helps prevent overfitting by making smaller parameter updates based on limited data [109]. It enables a more thorough exploration of the dataset and aids the model in identifying subtle patterns. Although convergence might be slower, prioritising stability and avoiding overfitting is essential.

Additionally, it is necessary to define the learning rate of the U-Net model. In this case, the recommended learning rate by the author of the LoRA [92] technique has been utilised, $1e-4$.

4.3.4 Fine-tuning

Instead of solely training the model with the discriminator, an improvement has been made by incorporating the IoU and delta E. The model's representation is illustrated in Figure 4.3.

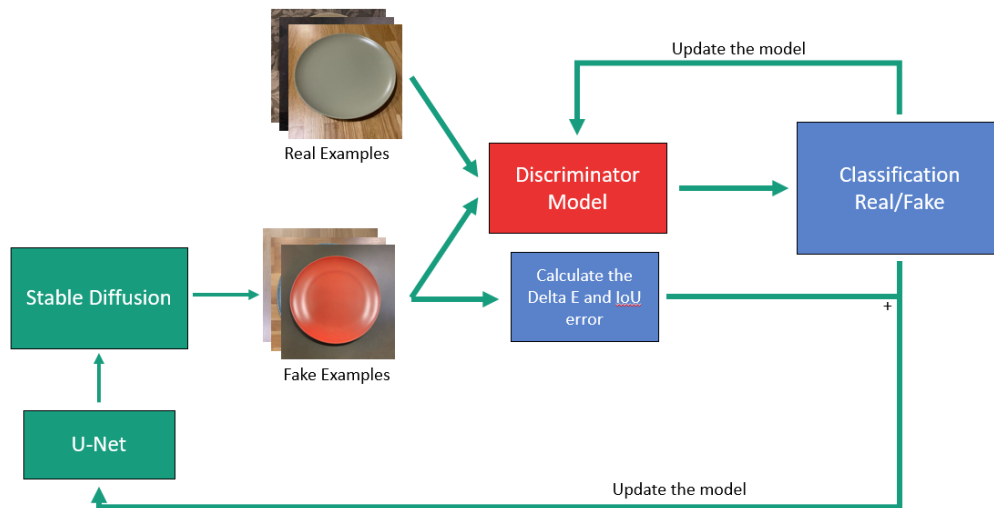


Figure 4.3: Simplify Stable-gan architecture with IoU and Delta E loss

This modification aims to train the model to capture both colour and position information accurately. It is crucial to calculate the IoU error and delta E using a reference image and a colour reference during each training epoch and use this error as losses.

To ensure that these losses do not heavily influence the overall loss of the model, their weight has been set to 0.2 times the loss. Consequently, these losses are integrated with the U-Net loss of the network and propagated throughout its architecture.

4.4 Fine-tune stable diffusion

The following sections explore the setups to approach the SD model, the challenges of the applied methods, and how to fine-tune the model using Dreambooth and LoRA. A fine-tuned model will be biased toward generating images similar to the dataset while maintaining the versatility of the original model.

4.4.1 Fine-tuning techniques

This section examines the effectiveness of refined techniques in choosing the appropriate models. The authors' suggested default settings were used to evaluate the models to ensure a fair comparison [88, 92]. For running Dreambooth, a GPU with 24GB VRAM is recommended, while a GPU with 12GB VRAM is suitable for LoRA.

The quality of the synthetic images was assessed by visually inspecting each experiment's results to determine the range of image instances that yielded the most promising outcomes. Even though Dreambooth's original paper mentioned good results with 3 to 5 training images [73], it does not identify the perfect number. For popular repository owners such as Joepenna [90] and XavierXau [89], the recommended number of training images is suggested to be in the interval of 20 and 30.

For Dreambooth, the ShivamShrirao [88] model was selected. This model has many advantages compared to other models. Firstly, it can generate excellent images in a relatively short time, making it efficient for the research purposes of this study. Moreover, it has the ability to train multiple tokens, enabling the training of all the different classes within a single model. This approach saves time and resources compared to using separate models for each plate class.

Additionally, this model provides the convenience of continuing training from a previous model, which is a beneficial feature. Furthermore, the model demonstrates a superior capability to handle and replicate the background of the images, an essential aspect of achieving image realism. Considering all these factors, the ShivamShrirao model [88] demonstrates to be the most suitable choice for the requirements of this work.

For LoRA models, the cloneofsimo [92] model was used for training. Although it is a different model, the author recommends using the same dataset configuration as Dreambooth [110]. All the images were already prepared for training in this case.

It was determined that the LoRA model outperformed DreamBooth in several aspects [110]. One advantage of LoRA is its lower compute requirements, making it accessible and cost-effective to train on different hardware configurations. This aspect allows it to leverage LoRA capabilities without straining computational resources.

Furthermore, the output generated by LoRA is significantly smaller than DreamBooth's. The trained LoRA outputs are compact, facilitating easier sharing, storage, and reuse. The weights for the new layers in LoRA can be saved as a single file, weighing approximately 2-200 MB. This file size is remarkably smaller than the original size of the UNet model used in DreamBooth, which is around 2-7 GB, providing practical benefits in terms of storage and processing [111].

Considering these factors, including training efficiency, resource requirements, output size, and experimental capabilities, LoRA emerged as the optimal choice for our generative model development. Its ability to efficiently generate high-quality images with smaller outputs made it a superior option to DreamBooth. The decision to adopt LoRA was made to align with the research goals and requirements.

4.4.2 Hyperparameters

Hyperparameters play a crucial role in fine-tuning the performance of a model and achieving optimal results. Understanding and selecting appropriate hyperparameters is essential in ML. One of the most influential hyperparameters is the ratio between the number of training images and the learning rate [112]. For Dreambooth, the recommended base configuration for the learning rate is $5e-6$ [88], while for LoRA, a learning rate of $1e-4$ is suggested [92].

In addition to the learning rate, two other essential hyperparameters are the number of epochs and the CFG Scale. The number of steps refers to the times the entire data set has to be run through the training. The CFG scale determines the level of adherence the model should have to the provided prompt.

Once the model selection process, specifically LoRA, was completed, the next crucial step was determining the appropriate number of steps for training. Initially, experiments were conducted using the recommended configuration of 100 times the number of training images. For instance, the concept_1 model was trained with 3000 steps, while the other concepts utilized 2000 steps. However, it was necessary to explore different values to identify the optimal number of steps for all models.

Table 4.1 shows the number of steps used to train the dataset. To better understand the results obtained with different values of epochs, values below and above the recommended values were chosen. The concept_1 model was trained with 4000 epochs to determine the ideal number, while the other models were trained with 3000 steps. A checkpoint was saved in intervals of 250 steps, and 10 images were generated in each step.

Table 4.1: Number of epoch train in each concept

Class	Steps
concept_1	4000
concept_2	3000
concept_3	3000
concept_4	3000

After determining the optimal number of steps, the next phase involves selecting more specific hyperparameters. In this stage, the focus is on exploring the effects of adjusting the Guidance and Scale. To investigate these variations, a series of tests were conducted with different Guidance values, specifically 7, 7.5, 8 and 10. The scale parameter was tested during the training with values of 0.7, 0.8 and 0.9. These values were chosen to test their influence and what to expect with values near the recommended, which is 0.9 [92]. To evaluate the impact of these hyperparameters, 50 images were generated for each combination of Scale and Guidance settings.

4.4.3 Model training

The next step is training after preparing the dataset, defining all the hyperparameters, and

selecting the models. Several experiments were conducted to determine the factors influencing the models' performance during training. To apply the Dreambooth and LoRA techniques, it is necessary to identify the plate with a symbol that the model does not know. Each plate is designated by a "concept" followed by a number, separated by "_". The name of the plate and the class type (in this case, "plate") are used for the token in the model prompt. The table 4.2 illustrates the plate names and their corresponding prompts. Following these guidelines can effectively apply the Dreambooth and LoRA techniques and achieve desired results during training.

Table 4.2: Plate class and corresponding token for the prompt.

Class	Text Prompt
concept_1	concept_1 plate
concept_2	concept_2 plate
concept_3	concept_3 plate
concept_4	concept_4 plate

After the dataset is prepared, hyperparameters are defined, and models are selected, the next crucial step is training. In order to decide between the LoRA and Dreambooth techniques, questionnaires were administered to compare the results. To acquire the representations of the surveys, 50 images were generated for each testing parameter and colour test, and the most realistic results were chosen through visual inspection.

The prompt used to generate the images for model comparison was the following:

$$Aphoto\ of\ "concept\ name" \quad (4.1)$$

4.5 Evaluations

Two key factors were considered to evaluate the results: the realism of the fake images and the model's capacity to accurately replicate the colour and position of the plates. The authenticity of the generated images is essential, as the model should be capable of producing visually plausible images that closely resemble real ones. Further validation of the authenticity of the results was aimed through the development of various questionnaires throughout this work. Simultaneously, the model's ability to accurately generate images with specific colours and faithfully reproduce their unique characteristics is crucial for its utility in this context. A quantitative assessment was employed to evaluate the position and colour accuracy. This assessment aimed to provide a systematic and measurable analysis of the model's performance in replicating the precise position and colour attributes.

4.5.1 Quantitative evaluation

The evaluation aimed to measure the precision of the generated images by comparing them to

authentic photographs. A detailed analysis assessed the position accuracy, comparing the generated plate positions to the corresponding ones in the reference photo. This comparison allowed for a thorough assessment of how well the model could generate the license plate positions accurately.

Additionally, the generated images' colours were carefully evaluated against the desired colours to ensure faithful colour reproduction. By conducting an accurate comparison, the degree of colour accuracy achieved by the model was quantified. This evaluation process provided valuable insights into how well the model could reproduce exact colours. The subsequent subsections will explore how this evaluation was conducted and the methodology and metrics employed.

4.5.1.1 Position Accuracy

For the position evaluation, a set of 50 images was generated specifically for this purpose. In order to facilitate accurate comparison, masks were created for each generated image as well as the corresponding actual images using the SAM technique [31]. These masks were carefully designed to highlight the area occupied by the plates in each image.

The Intersection over Union metric was employed to measure the accuracy of the generated image positions. The IoU metric, discussed in section 2.2.3.2, calculates the degree of overlap between the generated mask and the mask derived from the actual photo. By utilizing the authentic image masks as a reference, the IoU values provided an objective measure of how well the model generated the positions of the plates. To illustrate the calculation process, refer to Figure 4.5, which showcases an example of calculating the IoU. The reference image is represented in Figure 4.4.



Figure 4.4: Reference image used to calculate the IoU

This approach evaluated the position accuracy, allowing for a comprehensive analysis of the model's performance in generating accurate plate positions.

4.5.1.2 Color Accuracy

In addition to a thorough assessment of the accuracy of the plate position, an evaluation was

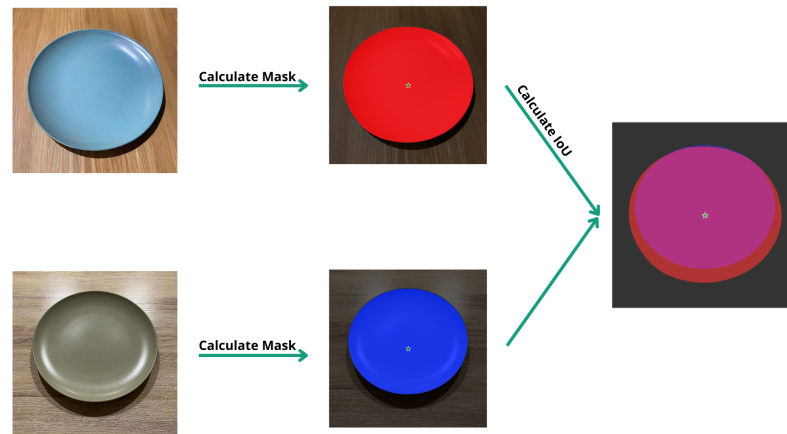


Figure 4.5: Example of calculating IoU

also carried out to ensure faithful and authentic colour reproduction in the generated images. This assessment employed the delta E metric.

A selection of 20 pixels from the centre of each plate was made to assess the colour accuracy of the generated images. This selection aimed to capture a representative sample of the plaque colour, allowing for a reliable comparison of colour accuracy. As colour variations and changes across the entire plate could significantly impact the overall colour perception, only 20 pixels from the centre of the plate were chosen for this comparison. The evaluation assessed the average colour code by considering multiple pixels rather than a single point. The centre of the plate was determined based on previously generated masks, ensuring consistency and accuracy in the pixel selection process. This approach facilitated an objective and systematic evaluation of colour reproduction on different plates.

The level of colour variation and precision can be evaluated by determining the delta E value. Once the chosen pixels are acquired, they are transformed into CIELAB color space (**Lab**) format for delta E calculation. Subsequently, a matrix with the corresponding colour values in **Lab** is created, matching the dimensions of the selected area. The ultimate step involves calculating the delta E for these values and computing the average to obtain the conclusive outcome, as seen in Figure 4.6.

Specific colours were intentionally chosen for assessment to ensure a comprehensive evaluation and encompass a diverse range of colours. These colours included black and white, red, green, blue, orange, and Dark slate grey (**Dsg**). This selection was made purposefully, as these colours represent a significant spectrum of plate variations commonly encountered in real-world scenarios. Black and white correspond to opposite ends of the spectrum, while red, green, and blue are the primary colours. The chosen colour is shown in table 4.3. Additionally, orange and **Dsg** represent mixed colours. By evaluating a variety of colours, the assessment encompassed a broader range of potential challenges, thereby providing a more comprehensive understanding of the model's ability to reproduce colours accurately.

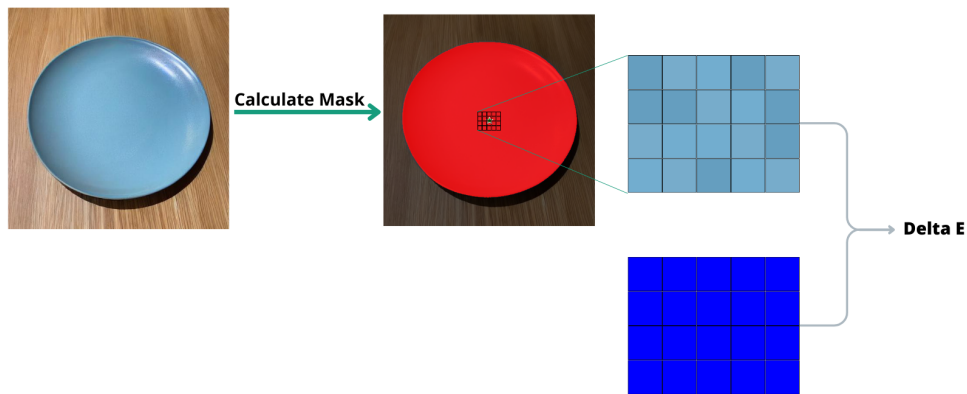
Figure 4.6: Calculating the ΔE

Table 4.3: Evaluation colours and their respective RGB colour code

Name	RGB code	Colour
Black	(0, 0, 0)	
White	(255, 255, 255)	
Red	(255, 0, 0)	
Green	(0, 255, 0)	
Blue	(0, 0, 255)	
Orange	(255, 165, 0)	
Dsg	(47, 79, 79)	

By employing these evaluation techniques, a quantitative analysis was conducted to assess the accuracy of the generated images regarding position and colour reproduction. Using the IoU metric for position evaluation and the delta E metric for colour evaluation, these evaluations provided valuable insights into the model's performance and suitability for clinical use.

4.5.2 Qualitative evaluation

As the output results are images, evaluating can be subjective and challenging. To overcome this challenge, seven surveys were conducted over eight weeks. The primary objective of these surveys was to evaluate the plausibility of the generated images.

The questionnaires were distributed to the employees of the *Fraunhofer* researchers, who were asked to carefully examine each image and determine whether it was real or generated. Participants were also encouraged to indicate if they found any photos indistinguishable, meaning they could not confidently differentiate between authentic and generated images.

Each week, a questionnaire was distributed, consisting of 12 shots. Two of these surveys were explicitly designed to test the plausibility of the evaluations and participants' responses. One contains only fake images deliberately incorporating noticeable elements that indicate their

artificial nature. Conversely, the other survey consisted solely of authentic photos. These two sets of pictures served as benchmarks for comparative analysis.

These forms were performed to evaluate various variables shown in Table 4.4. The first four questionnaires were conducted to assess and test the Dreambooth and LoRA models' quality. The fifth questionnaire evaluated the parameter changes in the LoRA model. Additionally, the sixth and the seventh forms served as a baseline and were only performed for the credibility of this evaluation method. Finally, the last evaluate the performance of the generated images of the proposed model and LoRA.

Table 4.4: The parameters tested in each questionnaire

Survey	Evaluating
1	Real; Dreamboth
2	Real; LoRA
3	Real; Dreambooth; LoRA
4	Real; Dreambooth; LoRA
5	LoRA parameters
6	Generated
7	Real
8	LoRA; Stable-gan

For better data analysis, values have been assigned to the results. Table 4.5 shows the values assigned to each representation. This allows for calculating the average that each image obtained. This average is calculated as the sum of all the values obtained divided by the number of answers given. The higher the average, the more realistic the image. In other words, the closer it is to 3, the more authentic it is, and vice versa.

Table 4.5: Values associated with each response

Response	Value
Unreal	1
Indistinguishable	2
Real	3

The surveys were intended to collect subjective opinions on the realism and quality of the images generated. By involving human participants in the evaluation process, the study sought to obtain reliable judgments regarding the visual plausibility of the generated images.

4.6 Summary

This chapter focuses on developing the generative model, precisely fine-tuning Stable Diffusion. The resources used, including GPU and PyTorch, are discussed, along with the selected tools

and methodology. The process involves dataset preparation, hyperparameter tuning, and evaluation. Fine-tuning is performed using Dreambooth and LoRA models. Hyperparameters such as learning rate and training steps are adjusted to optimize the models. In addition, the chapter concludes with evaluation methods, including quantitative metrics and subjective surveys, to assess the realism and accuracy of the generated images.

Chapter 5

Results

This chapter presents the results from the experiments conducted to develop the generative model for product development, including comparing fine-tuned models, evaluating the automatic techniques and reviews by *Fraunhofer* workers. The results of fine-tuning [SD](#) with different configurations to optimise the model's performance are discussed in the subsequent sections, followed by the results of the new generative model.

5.1 Qualitative evaluation

As mentioned in section [4.5.2](#), two test questionnaires were conducted only with real and generated images, whose results are shown in Figure [5.1](#), respectively.

We observe that in Form 7, 55.24% of the obtained answers were correct and that only about 11% voted as "Unreal". This shows that the participants were more inclined towards the images being true. In Form 6, 37.78% of the votes were given to "Real", demonstrating that the opinions were divided. This shows that there is a trend for the majority of the panel to decide correctly.

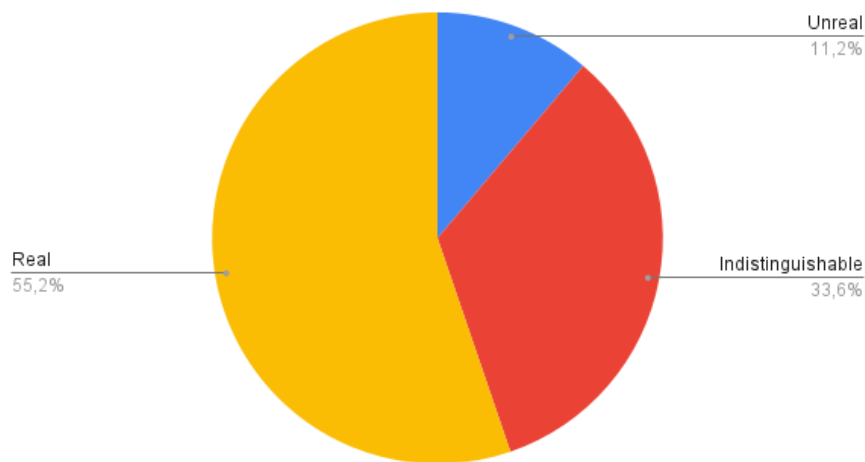
5.1.1 Fine-tuned model selection

The first step in the research project was to select the most suitable model. To accomplish this, a series of experiments were conducted, comparing the performance of both models with the recommended configurations from the authors. The focus was training the four specific concepts. Some of the results can be observed in Image [5.2](#).

For the decision-making process, four surveys were conducted to compare images generated by both models. Each survey consisted of a set of images, with one-third being real images and the remaining two-thirds generated by either DreamBooth or [LoRA](#). The first survey exclusively featured images generated by Dreambooth, and the second solely contained [LoRA](#) generated photos. The results of these questionnaires are represented in Figure [5.4](#).

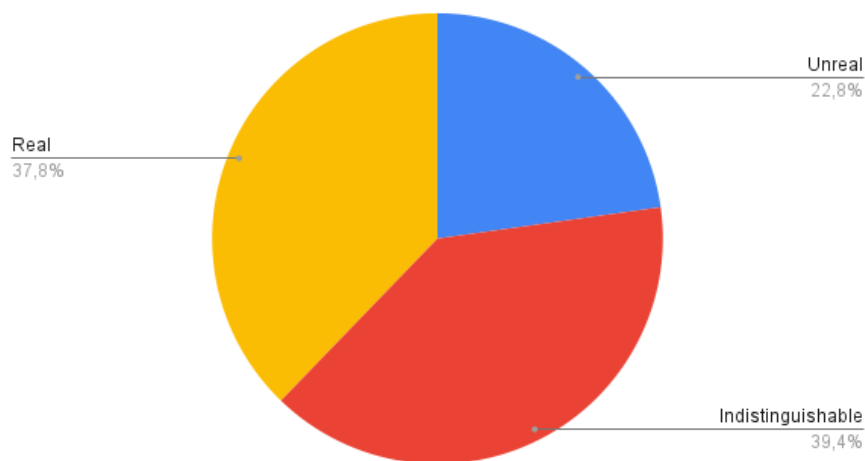
It is crucial to analyze the evaluation rates separately to compare the results of the first and second questionnaires. In the initial study, [LoRA](#) received a rating of 37.6% for generating images perceived as "Real" and 22.4% for "Unreal", while DreamBooth obtained ratings of 37.5%

Survey 7 - Only Real



(a) Survey with only Real images

Survey 6 - Only generated



(b) Survey with only Fake images

Figure 5.1: Comparison results between (a) only real and (b) only fake images

and 24.6%, respectively. These ratings were remarkably close, suggesting a similar performance between the two models in generating realistic images. Two additional surveys were conducted to gain a more comprehensive understanding and break this tie, combining images from both models within the same poll. These two new surveys included pictures from both models. The qualitative evaluation results are presented in Figure 5.4.

With the values established in table 4.5, table 5.1 presents the average result for each model in each survey.

Examining the data in the tables reveals that the LoRA model outperformed DreamBooth regarding image generation. In Survey 3 (4.4), the percentage of images rated as "Unreal" was



Figure 5.2: Comparison between (a) Dreambooth and (b) LoRA.

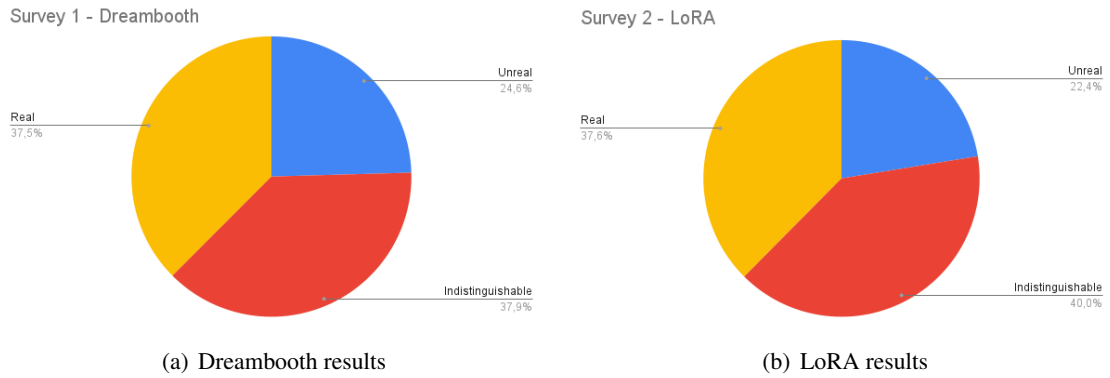


Figure 5.3: Result comparison between (a) Dreambooth and (b) LoRA.

significantly lower for LoRA (23.9%) compared to DreamBooth (33.7%), indicating that LoRA produced more realistic images. Similarly, in Survey 4 (4.4), LoRA performed better, with only 14% of pictures rated as "Unreal" compared to DreamBooth's 29.7%. Moreover, the average ratings from all four surveys consistently favoured LoRA over DreamBooth. The average rating for LoRA ranged from 2.15 to 2.4, while DreamBooth's average rating ranged from 2.04 to 2.14. These findings suggest that, on average, the images generated by LoRA were perceived as more realistic and visually appealing. The statistically lower percentage of images rated as "Unreal" and the higher average ratings obtained by LoRA, provide strong evidence of its superior performance compared to DreamBooth.

Considering these statistical findings, the LoRA model is the preferred option for the research project. Although the differences are insignificant, the results indicate that LoRA is better. The decision to adopt LoRA was made after considering and analysing its advantages, aligning with

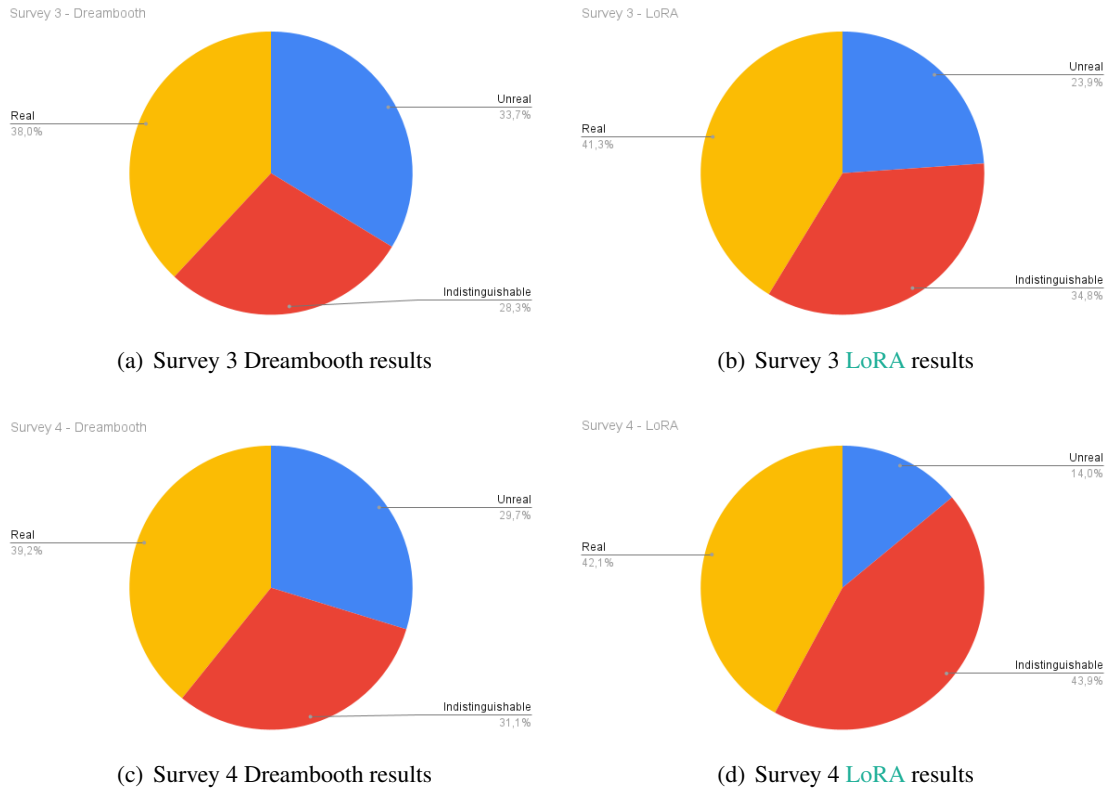


Figure 5.4: Result comparison between (a), (c) Dreambooth and (b),(d) LoRA.

Table 5.1: Comparison of the survey results for LoRA and Dreambooth

Values	Survey 1		Survey 2	
	Real	Dreambooth	Real	LoRA
Unreal	15	54	4	46
Indistinguishable	52	84	15	82
Real	45	86	49	77
Average	2.27	2.14	2.66	2.15

Values	Survey 3			Survey 4		
	Real	Dreambooth	LoRA	Real	Dreambooth	LoRA
Unreal	5	31	22	11	22	8
Indistinguishable	22	26	32	23	23	25
Real	65	35	38	42	29	24
Average	2.65	2.04	2.17	2.2	2.1	2.4

the research goals and requirements.

5.1.2 Hyperparameter

5.1.2.1 Number of steps

As stated in 4.4.2, the most realistic picture from each checkpoint was chosen from the image generated with different steps. The results of these training iterations can be observed in Figure 5.5.

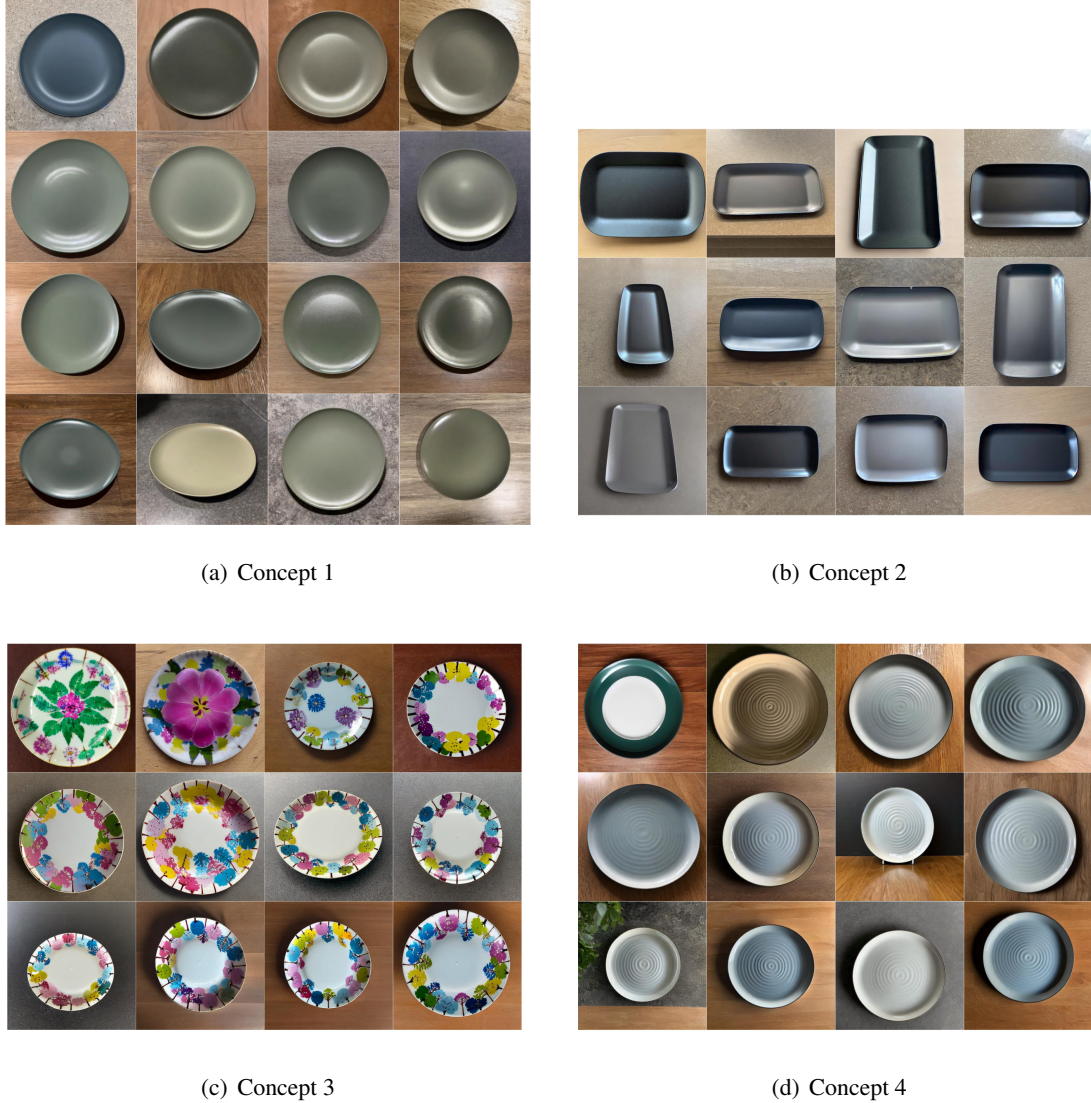


Figure 5.5: Resulting image for each epoch, incrementing 250 epochs for all concept

The results demonstrate that increasing the number of steps enables the model to capture more intricate features from the training data, leading to more realistic images. Based on the findings from Figure 5.5, it was evident that datasets with more detailed designs, such as concept_3, required more steps to achieve optimal results. In the case of concept_3, the results exhibited quality at 2000 steps and significantly improved at 3000 steps. On the other hand, concept_1, which has a simpler design and fewer steps, already achieved impressive results, showing significant improvement from just 1000 steps. For concepts 2 and 4, which are also relatively basic, good results were

obtained with 1000 steps. However, concept_4 continued to improve with higher step counts, while concept_2 demonstrated signs of overfitting and produced poor results at higher step counts.

Nevertheless, balancing the number of steps and available computational resources is crucial. Higher step counts result in longer training times and increased computational costs. Therefore, careful consideration must be given to selecting an appropriate number of steps that aligns with the specific task and dataset.

Considering Fig. 5.5, it was made decisions on the number of steps for each concept, presented in the table 5.2.

Table 5.2: Optimal Number of Training Steps for Each Concept

Class	Steps
concept_1	2000
concept_2	1500
concept_3	3000
concept_4	3000

5.1.2.2 Scale and guidance

Subsequently, to get the results of the methodology from 4.4.2, a survey was conducted using 12 selected images from visual inspection. These images were carefully chosen to provide a representative sample of the generated outputs across different parameter values and are represented in Figure 5.6. The results of the forms are described in the table 5.3.

Table 5.3: Results of the questionnaire on Scale and Guidance hyperparameters

Scale	Guidance				Average
	7	7.5	8	10	
0.7	2,33	2.11	2.39	2.47	2,33
0.8	2,56	2.67	2.39	2.1	2,43
0.9	2,17	2.06	2.06	2.44	2,18
Average	2,35	2,28	2,28	2,34	

After analyzing the survey results in the table 5.3 and considering visual inspection, we found that Scale 0.8 and Guidance 7.5 consistently received higher ratings than other combinations. This suggests that this particular combination yielded the best-performing generated outputs. The average ratings within each Scale value showed that Scale 0.8 had a slightly higher rating range than Scale 0.7 and 0.9. Similarly, the average ratings within each Guidance value indicated that Guidance 7 had a slightly higher rating range than others.

Based on the results, the LoRA model, a guidance value of 7.5, and a scale value of 0.8 were selected as the optimal configuration. Subsequently, further tests were conducted using these specific parameter settings to investigate their effects and evaluate the model's performance.

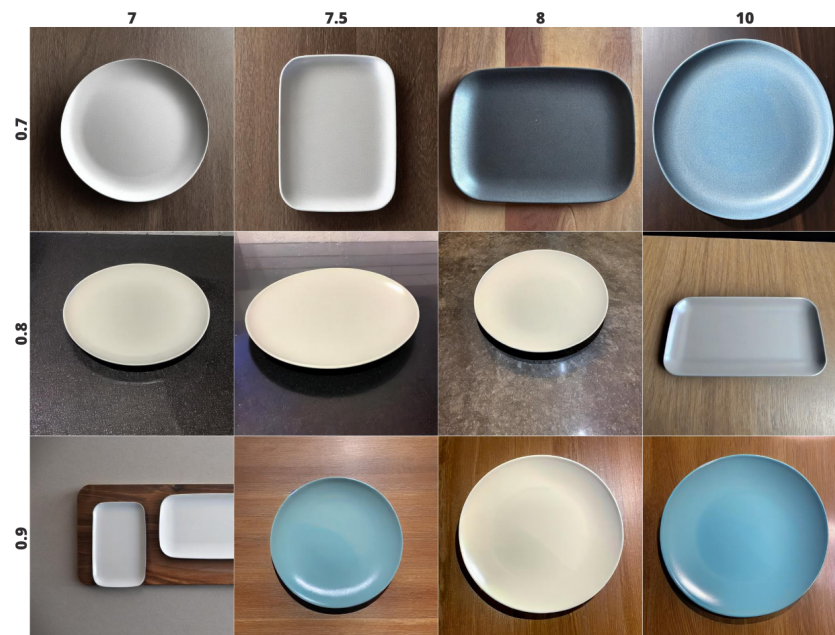


Figure 5.6: Visualization on the effect of Scale and Guidance values

5.1.3 Model selection

Figure 5.7 presents a selection of images that were used to construct survey number 8 (4.4), allowing a comparison of the generated images from LoRA and Stable-GAN in terms of their resemblance to reality.

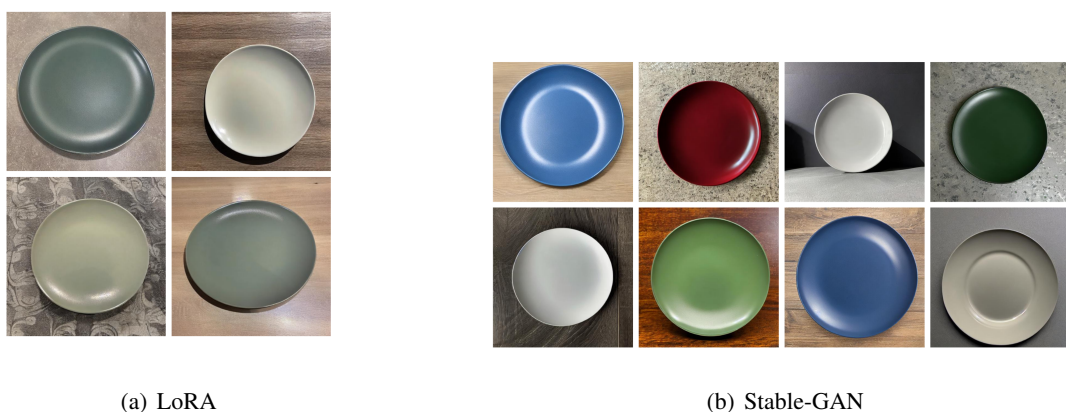


Figure 5.7: Comparison generated images between (a) LoRA and (b) Stable-GAN.

Following the selection of these images, Figure 5.8 showcases the questionnaire outcomes, with the participant's feedback providing further insights.

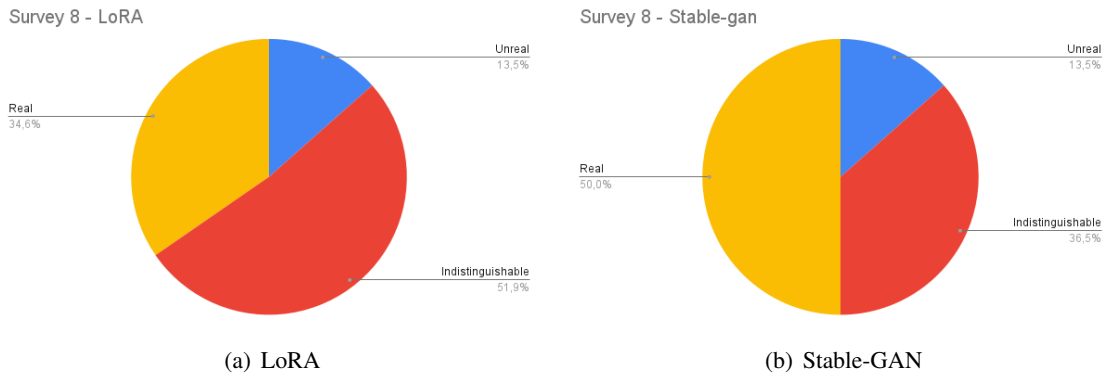


Figure 5.8: Comparison results of the questionnaire from (a) LoRA and (b) Stable-GAN.

Comparing the two models, LoRA and Stable-GAN, results indicate some differences in their performance when generating images. In terms of generating "Unreal" images, both models received 13.5%. This suggests they produced photos that the participants considered realistic. Regarding "indistinguishable", Stable-GAN received 36.5% of the votes, while LoRA received 52%. This indicates that LoRA produced results that are harder for participants to distinguish from fake pictures. Regarding generating "Real" images, Stable-GAN outperformed LoRA. Stable-GAN received 50%, while LoRA received only 35% of the votes. This indicates that the images generated by Stable-GAN were considered more realistic and authentic by the participants, closely resembling real photographs.

Overall, the results suggest that Stable-GAN performed better than LoRA in generating images perceived as more realistic. It received higher percentages of votes in the "Real" categories and less in "indistinguishable", indicating its ability to generate images that closely resemble real photographs.

5.2 Quantitative evaluation

The generated plates' position accuracy and colour reproduction were evaluated simultaneously to assess their overall performance.

The colour reproduction evaluation generated a diverse set of 50 images, encompassing the primary colours of red, blue, and green and additional colours such as black, white, orange, and dark blue. This selection allowed for a comprehensive assessment of the model's ability to reproduce different colours accurately. The colour distance was calculated by comparing the generated plates and the corresponding colours' RGB values, enabling quantitative analysis of colour fidelity.

5.2.1 LoRA

Table 5.4 presents the IoU values for position accuracy, the ΔE values for colour reproduction, and the average values across all concepts of images generated by LoRA.

Table 5.4: Comparison of Survey Results for Different Models

Colour		Results	
Name	Code	ΔE	IoU
White	[0,0,0]	10.355	0.660
Black	[255,255,255]	9.374	0.601
Red	[255,0,0]	10.303	0.650
Green	[0,255,0]	10.493	0.617
Blue	[0,0,255]	9.989	0.675
Orange	[255,165,0]	9.429	0.593
Dsg	[47,79,79]	10.026	0.643
Average		9.996	0.634

The average IoU for position accuracy across all concepts is 0.634, indicating a good alignment of the generated plates with the ground truth. The individual concept results range from 0.601 to 0.675, demonstrating consistent and reliable performance regarding position accuracy.

Regarding colour reproduction, the average ΔE across all concepts is 9.996, suggesting a good colour match between the generated plates and the ground truth colours. The individual concept results range from 9.374 to 10.493, reflecting consistent performance in reproducing colours across different concepts.

The visual inspection revealed that the models successfully learned the diverse styles, colours, and textures present in the training dataset. The quantitative evaluation confirmed the generated plates' reasonable position accuracy and good colour reproduction.

5.2.2 Stable-gan

As section 4.3 mentioned, the first test was only trained with the discriminator's loss. The resulting image can be seen in 5.9. Table 5.5 shows the results of these tests.

Table 5.5: Survey Results comparing the results for Stable-GAN

Colour	Results	
Name	ΔE	IoU
White	38.430	0.6280
Black	31.078	0.5635
Red	76.518	0.6217
Green	101.533	0.6055
Blue	113.414	0.6177
Orange	68.045	0.5801
Dsg	20.876	0.6362
Average	64.27	0.607

The outcome of the individual concepts varies between 20.876 and 113.414. The higher values indicate more significant deviations from the actual colours, suggesting that it did not display good



Figure 5.9: Resulting outputs from the Stable-GAN

colour reproduction accuracy. The average value of ΔE across all concepts is 64.27, providing a poor overall measure for colour reproduction performance.

Regarding the position accuracy, the results of the individual concepts vary between 0.5635 and 0.6362, which indicates slight variations in the performance of the different concepts. The average of **IoU** across all concepts is 0.607.

According to the findings, Stable-GAN demonstrates nearly identical position accuracy compared to the fine-tuned results, indicating that both models possess similar capabilities. However, the results were significantly inferior regarding colour representation, suggesting that this model fails to capture colours accurately.

As mentioned in sec ??, for the second test, the model was tuned with the discriminator's loss along with **IoU** and delta E's loss. The generated images can be seen in Figure 5.10. This experiment must have very high computational power to train the model. An epoch takes about 1 minute per epoch. Since the recommended [88] is approximately 100 times the number of training images, in the case of concept_1, it would be 3000 epochs take 50 hours. For this reason, only one experiment was done with this model. For this, concept_1 was used, with 3000 epochs, focusing on the black colour. After the training, 50 images were generated to study the result.

The result can be visualized in Table 5.6.

The 5.6 table shows that the accuracy in positioning the plate has dramatically increased compared to all previous tests. The average **IoU** result was 0.77. However, the result obtained for ΔE was 51.97, which is higher than previously.

These results show that, with the addition of the losses. Although the colour reproduction



Figure 5.10: Resulting outputs from the Stable-GAN with the IoU and ΔE loss.

Table 5.6: Comparison of Survey Results for Stable-gan

Colour	Results	
Name	ΔE	IoU
Black	51.974	0.774

results were not the best, the results obtained for the positioning were much better, demonstrating that this approach achieved better results for one of the analysed parameters.

5.3 Summary

This study proposed a new model for generative AI, and the results obtained were highly competitive with the state-of-the-art methods. The newly introduced model demonstrated its capability to produce realistic and visually appealing images for product development. The performance of the proposed model indicates its potential for advancement in the field, showcasing promising results.

This chapter explores the model selection and method essential for the generative model. It begins by picking the LoRA model as the preferred choice over DreamBooth, considering the visual appeal and training efficiency factors. It discusses determining optimal hyperparameters, including the number of training steps, scale, and guidance values. In addition, it also analyses the outputs generated by the Stable-GAN model. It furthermore describes the evaluation method, divided into two parts: the quantitative testing and the questionnaires.

Overall, the [Results](#) chapter provides insights into the capabilities and performance of the generative model in producing realistic and visually appealing images for product development. In conclusion, the generator agent developed in this work showed promising results, although subsequent improvements can be made.

Chapter 6

Conclusion and Future Work

This dissertation focused on exploring the potential of artificial intelligence in the ceramics industry, specifically in the context of new product design and development. By leveraging generative AI models, the research aimed to optimize the colouration and design process while reducing costs and scalability challenges associated with human labour. A novel approach was developed and compared against existing state-of-the-art models, focusing on assessing the realism and quality of the generated ceramic images.

The conclusion of this thesis highlights the successful exploration of Diffusion Models for product design, specifically using SD. The model was fine-tuned to generate synthetic images of plates with different colours and positions that were highly realistic and indistinguishable from authentic images. Evaluating these synthetic images for realism was crucial to ensure the reliability and trustworthiness of the generated results. This research showcased the capability of the generated images, revealing that the models can create images that closely resemble real ones and are difficult to distinguish. These findings suggest that generative AI has the potential to assist humans in various tasks, particularly in product design.

In addition, the research proposed a new model that yielded competitive results compared to the state-of-the-art approaches. Despite its computational intensity, the new model demonstrated promising outcomes in terms of image realism. It proved its capability to generate visually appealing and realistic ceramic images for product development, even surpassing the tested techniques in terms of reality. The results showcased the potential of the new model and its ability to rival existing state-of-the-art approaches.

However, there are opportunities for future work to improve further and expand upon these findings. Firstly, enhancing the dataset for fine-tuning by including a more extensive and diverse range of images could improve the model's training and generalization capabilities. Additionally, allocating more time for testing with additional parameters and variables could produce more refined results.

It is also essential to test the practical application of this technique in real-world scenarios, particularly within the context of the *Fraunhofer* project mentioned. This would provide valuable insights into the practicality and potential of the approach in real design projects.

Further model improvements can be improved by exploring different datasets and testing the model with various parameters to determine the most suitable configuration. Extensive experimentation and analysis are needed to validate this new model's potential and identify improvement areas. Additionally, investigating the influence of the discriminator, including training it with different parameters and architectures, would provide valuable insights into its impact on the model's overall performance.

This research establishes a solid foundation for applying Diffusion Models in generating product design images. With continued research and development, this approach can significantly impact the design and reproduction of colours, enabling accurate and reliable visualization of specific attributes in product design.

References

- [1] Ceramics market size, share growth analysis report, 2030. URL: <https://www.grandviewresearch.com/industry-analysis/ceramics-market>.
- [2] Ceramics market trends and industry statistics 2022-2032. URL: <https://www.factmr.com/report/3846/ceramics-market>.
- [3] Pottery ceramics market size worth 15.00 billion by 2030. URL: <https://www.grandviewresearch.com/press-release/global-pottery-ceramics-market>.
- [4] Ceramics market size, trends and global forecast to 2032. URL: <https://www.thebusinessresearchcompany.com/report/ceramics-global-market-report>.
- [5] PwC Portugal. Capacitação da indústria da cerâmica portuguesa. URL: www.pwc.pt.
- [6] Pottery ceramics market size, share trends report, 2030. URL: <https://www.grandviewresearch.com/industry-analysis/pottery-ceramics-market-report>.
- [7] Smartcolor4ceramics-otimização do processo de (re)criação de cor em vidrados cerâmicos introdução.
- [8] Vasile Daniel Păvăloaia and Sabina Cristiana Necula. Artificial intelligence as a disruptive technology—a systematic literature review. *Electronics* 2023, Vol. 12, Page 1102, 12:1102, 2 2023. URL: <https://www.mdpi.com/2079-9292/12/5/1102/html><https://www.mdpi.com/2079-9292/12/5/1102>, doi:10.3390/ELECTRONICS12051102.
- [9] Fraunhofer portugal. URL: <https://www.fraunhofer.pt/>.
- [10] Sc4c.: Smartcolors for ceramics.
- [11] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, 2018. doi:10.1109/ICCUBEA.2018.8697857.
- [12] Is deep learning part of ai? - reason.town. URL: <https://reason.town/is-deep-learning-part-of-ai/>.
- [13] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. URL: <http://dx.doi.org/10.1016/j.neunet.2014.09.003>, doi:10.1016/j.neunet.2014.09.003.

- [14] What are convolutional neural networks? URL: <https://serokell.io/blog/introduction-to-convolutional-neural-networks>.
- [15] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature* 2015 521:7553, 521:436–444, 5 2015. URL: <https://www.nature.com/articles/nature14539>, doi:10.1038/nature14539.
- [16] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature* 1986 323:6088, 323:533–536, 1986. URL: <https://www.nature.com/articles/323533a0>, doi:10.1038/323533a0.
- [17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [18] Reconciling modern machine learning and the bias-variance trade-off. *arXiv*, 4:1–16, 2018. URL: <http://arxiv.org/abs/1812.11118>.
- [19] Kevin Bretonnel Cohen. Biomedical natural language processing and text mining. *Methods in Biomedical Informatics: A Pragmatic Approach*, pages 141–177, 1 2014. doi:10.1016/B978-0-12-401678-1.00006-3.
- [20] Christopher D. Manning, Hinrich Schütze, and Gerhard Weikurn. Foundations of statistical natural language processing. *ACM SIGMOD Record*, 31:37–38, 9 2002. URL: <https://dl.acm.org/doi/10.1145/601858.601867>, doi:10.1145/601858.601867.
- [21] M. Hassler and G. Fliedl. Text preparation through extended tokenization. *WIT Transactions on Information and Communication Technologies*, 37:13–21, 6 2006. URL: <http://www.witpress.com/elibrary/wit-transactions-on-information-and-communication-technologies/37/16699>, doi:10.2495/DATA060021.
- [22] Jonas Oppenlaender. A taxonomy of prompt modifiers for text-to-image generation. 4 2022. URL: <https://arxiv.org/abs/2204.13988v2>.
- [23] Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. *Conference on Human Factors in Computing Systems - Proceedings*, 2 2021. URL: <https://arxiv.org/abs/2102.07350v1>, doi:10.1145/3411763.3451760.
- [24] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. 10 2021. URL: <https://arxiv.org/abs/2110.08207v3>.
- [25] Robin Rombach, Andreas Blattmann, and Björn Ommer. Text-guided synthesis of artistic images with retrieval-augmented diffusion models. 7 2022. URL: <https://arxiv.org/abs/2207.13038v1>.

- [26] A traveler's guide to the latent space. URL: <https://tinyurl.com/yw7jyhnh>.
- [27] Walter Hugo Lopez Pinaya, Sandra Vieira, Rafael Garcia-Dias, and Andrea Mechelli. Autoencoders. *Machine Learning: Methods and Applications to Brain Disorders*, pages 193–208, 3 2020. URL: <https://arxiv.org/abs/2003.05991v2>, doi:10.1016/B978-0-12-815739-8.00011-0.
- [28] Basics of autoencoders. autoencoders (ae) are type of. . . | by deepak birla | medium. URL: <https://medium.com/@birla.deepak26/autoencoders-76bb49ae6a8f>.
- [29] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:640–651, 11 2014. URL: <https://arxiv.org/abs/1411.4038v2>, doi:10.1109/TPAMI.2016.2572683.
- [30] 020 overview of semantic segmentation methods - master data science 08.11.2021. URL: <https://datahacker.rs/020-overview-of-semantic-segmentation-methods/>.
- [31] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.
- [32] Intersection over union (iou) for object detection - pyimage-search. URL: <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>.
- [33] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:779–788, 6 2015. URL: <https://arxiv.org/abs/1506.02640v5>, doi:10.1109/CVPR.2016.91.
- [34] Intersection over union (iou) | hasty.ai. URL: <https://hasty.ai/docs/mp-wiki/metrics/iou-intersection-over-union>.
- [35] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *International Journal for Research in Applied Science and Engineering Technology*, 10:943–947, 11 2015. URL: <https://arxiv.org/abs/1511.08458v2>, doi:10.22214/ijraset.2022.47789.
- [36] Vision transformers: The end of convolutional neural networks? URL: <https://www.plainconcepts.com/vision-transformer/>.
- [37] An intuitive guide to convolutional neural networks. URL: <https://tinyurl.com/36wxvm6t>.
- [38] Weihao Weng and Xin Zhu. U-net: Convolutional networks for biomedical image segmentation. *IEEE Access*, 9:16591–16603, 5 2015. URL: <https://arxiv.org/abs/1505.04597v1>, doi:10.48550/arxiv.1505.04597.
- [39] Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S. Lew. A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, 7:87–93, 6 2018. URL: <https://link.springer.com/article/10.1007/s13735-017-0141-z>, doi:10.1007/S13735-017-0141-Z/FIGURES/3.

- [40] U-net image segmentation in keras - pyimagesearch. URL: <https://pyimagesearch.com/2022/02/21/u-net-image-segmentation-in-keras/>.
- [41] Tiange Xiang, Chaoyi Zhang, Xinyi Wang, Yang Song, Dongnan Liu, Heng Huang, and Weidong Cai. Towards bi-directional skip connections in encoder-decoder architectures and beyond. *Medical Image Analysis*, 78:102420, 5 2022. doi:10.1016/J.MEDIA.2022.102420.
- [42] Portuguese creativity and know how in ceramics is above the world average | s. bernardo ceramics. URL: <https://tinyurl.com/39x4nzaa>.
- [43] Donald Norman. *Emotional Design: Why We Love (or Hate) Everyday Things*, volume 27. 01 2004.
- [44] Paul Hekkert, Dirk Snelders, and Piet Wieringen. 'most advanced, yet acceptable': Typicality and novelty as joint predictors of aesthetic preference in industrial design. *British journal of psychology (London, England : 1953)*, 94:111–24, 03 2003. doi:10.1348/000712603762842147.
- [45] Nathan Crilly, James Moultrie, and P John Clarkson. Seeing things: consumer response to the visual domain in product design. URL: www.elsevier.com/locate/destud, doi:10.1016/j.destud.2004.03.001.
- [46] Scanning basics - what is a pixel? URL: <https://www.scantips.com/basics1b.html>.
- [47] Machine learning - going furthur with cnn part 2 - dev community. URL: <https://dev.to/sandeepbalachandran/machine-learning-going-furthur-with-cnn-part-2-41km>.
- [48] Brief explanation of delta e or delta e* - hunterlab. URL: <https://support.hunterlab.com/hc/en-us/articles/203023559-Brief-Explanation-of-delta-E-or-delta-E->.
- [49] The cielab color space diagram. the cielab, or cie l* a* b*, color... | download scientific diagram. URL: <https://tinyurl.com/yxt2kytz>.
- [50] What is delta e? and why is it important for color accuracy? - viewsonic library. URL: <https://www.viewsonic.com/library/creative-work/what-is-delta-e-and-why-is-it-important-for-color-accuracy/>.
- [51] Delta e: A key to understanding lightfastness readings | just paint. URL: <https://justpaint.org/delta-e/>.
- [52] Learn l. URL: <http://zschuessler.github.io/DeltaE/learn/>.
- [53] How generative design could reshape the future of product development. URL: <https://tinyurl.com/yk44pkmx>.
- [54] Alexander Burnap, Ye Liu, Yanxin Pan, Honglak Lee, Richard Gonzalez, and Panos Y Papalambros. Estimating and exploring the product form design space using deep generative models, 2016.

- [55] Jaime Deverall, Jiwoo Lee, and Miguel Ayala. Using generative adversarial networks to design shoes: The preliminary steps, 2017.
- [56] Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Texturegan: Controlling deep image synthesis with texture patches. 6 2017. URL: <http://arxiv.org/abs/1706.02823>.
- [57] Kwonsang Sohn, Christine Eunyoung Sung, Gukwon Koo, and Ohbyung Kwon. Artificial intelligence in the fashion industry: consumer responses to generative adversarial network (gan) technology. *International Journal of Retail and Distribution Management*, 49:61–80, 1 2021. doi:10.1108/IJRDM-03-2020-0091.
- [58] Generative ai. URL: <https://generativeai.net/>.
- [59] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *COMMUNICATIONS OF THE ACM*, 63, 2020. doi:10.1145/3422622.
- [60] A. M. TURING. I.—computing machinery and intelligence. *Mind*, LIX:433–460, 10 1950. URL: <https://academic.oup.com/mind/article/LIX/236/433/986238>, doi:10.1093/MIND/LIX.236.433.
- [61] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. 2 2021. URL: <https://arxiv.org/abs/2102.12092v2>.
- [62] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13675 LNCS:89–106, 3 2022. URL: <https://arxiv.org/abs/2203.13131v1>, doi:10.1007/978-3-031-19784-0_6.
- [63] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. 4 2022. URL: <https://arxiv.org/abs/2204.06125v1>, doi:10.48550/arxiv.2204.06125.
- [64] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photo-realistic text-to-image diffusion models with deep language understanding. 5 2022. URL: <http://arxiv.org/abs/2205.11487>.
- [65] Scott Reed, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran REEDSCOT, Bernt Schiele, and Honglak Lee SCHIELE. Generative adversarial text to image synthesis. 2016.
- [66] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2022-June:10674–10685, 12 2021. URL: <https://arxiv.org/abs/2112.10752v2>, doi:10.48550/arxiv.2112.10752.
- [67] Top 10 deep learning algorithms you should know in 2023. URL: <https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm>.

- [68] Piotr Teterwak, Aaron Sarna, Dilip Krishnan, Aaron Maschinot, David Belanger, Ce Liu, and William T Freeman. Boundless: Generative adversarial networks for image extension.
- [69] Kaiwen Cui, Gongjie Zhang, Fangneng Zhan, Jiaxing Huang, and Shijian Lu. Fbc-gan: Diverse and flexible image synthesis via foreground-background composition.
- [70] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *32nd International Conference on Machine Learning, ICML 2015*, 3:2246–2255, 3 2015. URL: <https://arxiv.org/abs/1503.03585v8>.
- [71] Chitwan Saharia, William Chan, Huiwen Chang, Chris A Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. URL: <https://doi.org/10.1145/3528233.3530757>, doi:10.1145/3528233.3530757.
- [72] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. 12 2021. URL: <https://arxiv.org/abs/2112.10741v3>, doi:10.48550/arxiv.2112.10741.
- [73] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. 8 2022. URL: <https://arxiv.org/abs/2208.12242v1>, doi:10.48550/arxiv.2208.12242.
- [74] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 11:8780–8794, 5 2021. URL: <https://arxiv.org/abs/2105.05233v4>, doi:10.48550/arxiv.2105.05233.
- [75] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *32nd International Conference on Machine Learning, ICML 2015*, 3:2246–2255, 3 2015. URL: <https://arxiv.org/abs/1503.03585v8>.
- [76] Diffusion models made easy. understanding the basics of denoising... | by j. rafid sid-diqui, phd | towards data science. URL: <https://towardsdatascience.com/diffusion-models-made-easy-8414298ce4da>.
- [77] Introduction to diffusion models for machine learning. URL: <https://www.assemblyai.com/blog/diffusion-models-for-machine-learning-introduction/>.
- [78] What are diffusion models? | lil'log. URL: <https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>.
- [79] Stable diffusion with diffusers. URL: https://huggingface.co/blog/stable_diffusion.
- [80] Adam Roberts, Colin Raffel, and Noam Shazeer Google. How much knowledge can you pack into the parameters of a language model? URL: <https://goo.gle/t5-checkpoints>.

- [81] sentence-transformers/clip-vit-l-14 · hugging face. URL: <https://huggingface.co/sentence-transformers/clip-ViT-L-14>.
- [82] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. 2 2022. URL: <https://arxiv.org/abs/2202.09778v2>.
- [83] Diffusion models in practice. part 1: The tools of the trade - deepsense.ai. URL: <https://deepsense.ai/diffusion-models-in-practice-part-1-the-tools-of-the-trade/>.
- [84] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. URL: <https://openreview.net/pdf?id=qw8AKxfYbI>.
- [85] How does stable diffusion work? - stable diffusion art. URL: https://stable-diffusion-art.com/how-stable-diffusion-work/#What_is_CFG_value.
- [86] Guide: Stable diffusion’s cfg scale explained. URL: <https://tinyurl.com/593566pb>.
- [87] Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don’t stop pretraining: Adapt language models to domains and tasks. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, 4 2020. URL: <https://arxiv.org/abs/2004.10964v3>, doi:10.18653/v1/2020.acl-main.740.
- [88] Shivamshrira0/diffusers: diffusers: State-of-the-art diffusion models for image and audio generation in pytorch. URL: <https://github.com/ShivamShrirao/diffusers>.
- [89] Xavierxiao/dreambooth-stable-diffusion: Implementation of dreambooth (<https://arxiv.org/abs/2208.12242>) with stable diffusion. URL: <https://github.com/XavierXiao/Dreambooth-Stable-Diffusion>.
- [90] Joepenna/dreambooth-stable-diffusion: Implementation of dreambooth (<https://arxiv.org/abs/2208.12242>) by way of textual inversion (<https://arxiv.org/abs/2208.01618>) for stable diffusion (<https://arxiv.org/abs/2112.10752>). tweaks focused on training faces, objects, and styles. URL: <https://github.com/JoePenna/Dreambooth-Stable-Diffusion>.
- [91] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. 6 2021. URL: <https://arxiv.org/abs/2106.09685v2>.
- [92] cloneofsim0/lora: Using low-rank adaptation to quickly fine-tune diffusion models. URL: <https://github.com/cloneofsim0/lora>.
- [93] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. URL: <https://github.com/openai/>.
- [94] Jonathon Shlens. Notes on kullback-leibler divergence and likelihood. 4 2014. URL: <https://arxiv.org/abs/1404.2000v1>.

- [95] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 2017-December:6627–6638, 6 2017. URL: <https://arxiv.org/abs/1706.08500v6>, doi:10.48550/arxiv.1706.08500.
- [96] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2 2018. URL: <https://arxiv.org/abs/1802.03446v5>, doi:10.48550/arxiv.1802.03446.
- [97] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. 2 2021. URL: <https://arxiv.org/abs/2103.00020v1>, doi:10.48550/arxiv.2103.00020.
- [98] Pytorch. URL: <https://pytorch.org/>.
- [99] Stability ai. URL: <https://stability.ai/>.
- [100] Hugging face – the ai community building the future. URL: <https://huggingface.co/>.
- [101] runwayml/stable-diffusion-v1-5 · hugging face. URL: <https://huggingface.co/runwayml/stable-diffusion-v1-5>.
- [102] Stable diffusion 1 vs 2 - what you need to know. URL: <https://www.assemblyai.com/blog/stable-diffusion-1-vs-2-what-you-need-to-know/>.
- [103] iphone 11 - technical specifications - apple (by). URL: <https://www.apple.com/by/iphone-11/specs/>.
- [104] Birme - bulk image resizing made easy 2.0 (online free). URL: <https://www.birme.net/>.
- [105] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 11 2015. URL: <https://arxiv.org/abs/1511.06434v2>.
- [106] Dcgan tutorial — pytorch tutorials 2.0.1+cu117 documentation. URL: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html.
- [107] Binary classification using new pytorch best practices, part 2: Training, accuracy, predictions – visual studio magazine. URL: <https://visualstudiomagazine.com/articles/2022/10/14/binary-classification-using-pytorch-2.aspx>.
- [108] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 12 2014. URL: <https://arxiv.org/abs/1412.6980v9>.
- [109] Sebastian Thrun and Lorian Pratt. *Learning to Learn: Introduction and Overview*, pages 3–17. Springer US, Boston, MA, 1998. URL: https://doi.org/10.1007/978-1-4615-5529-2_1, doi:10.1007/978-1-4615-5529-2_1.

- [110] Introducing lora: A faster way to fine-tune stable diffusion - replicate - replicate. URL: <https://replicate.com/blog/lora-faster-fine-tuning-of-stable-diffusion>.
- [111] What are lora models and how to use them in automatic1111 - stable diffusion art. URL: <https://stable-diffusion-art.com/lora/>.
- [112] Training stable diffusion with dreambooth using diffusers. URL: <https://huggingface.co/blog/dreambooth>.