UNIVERSITÀ DEGLI STUDI DI MILANO - BICOCCA
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di Laurea Magistrale in Data Science

Generative AI for Image Captioning Evaluation

Relatore: Prof. Simone Bianco

Correlatore: Prof. Paolo Napoletano

Tesi di Laurea Magistrale di:
Gianluca Cavallaro
Matricola 826049

Anno Accademico 2022-2023

Abstract

In recent years, the field of generative AI has experienced significant advancements. Specifically, text-to-image algorithms have gained immense popularity due to their incredible results. These models can potentially revolutionize many industries. But can they also be effective as validation tools for image captioning model results?

The main objective of this thesis, in fact, is to use text-to-image models to validate the results obtained from a series of image captioning models. The work begins with a dataset containing eight different textual descriptions generated by as many image captioning models for 5,000 images extracted from the COCO dataset. The idea is to implement a series of open-source text-to-image models to generate images from these descriptions, determining the best prompt among the eight based on the similarity between the original image and the regenerated ones.

In this work, five of the most widely used open-source text-to-image models are implemented through specific Python libraries. These models are used to generate a series of images for each caption in the dataset. Subsequently, the similarity between real and regenerated images is assessed through the calculation of CLIPScore, an evaluation metric based on the implementation of the multimodal CLIP model.

The results of this work have provided insights into the best and worst captioning models among those tested. The utility of CLIPScore as a metric for image similarity has been demonstrated, despite its lack of correlation with visual quality. Similarly, it has been shown that other traditional metrics for image similarity are not suitable for this task. Furthermore, in the second part of the work, experiments were conducted with different prompt structures, highlighting how the use of highly detailed descriptions helps generate images much more similar to the original ones compared to those generated from brief and concise descriptions.

All the conclusions presented in this work, however, lack strong consistency due to the limited computational resources available, which allowed for work only on a very small subset of the original dataset. Nevertheless, the proposed methodology can be considered a solid starting point for future developments in the use of text-to-image models as tools for validating the results of image captioning models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, there has been a notable surge of interest and success in the realm of artificial intelligence when it comes to text-to-image algorithms. These algorithms have managed to close the gap between natural language processing (NLP) and computer vision, enabling the creation of visually coherent and realistic images based on textual descriptions. The impressive outcomes produced by these models have captivated millions of users, allured by the ability to unleash their creativity and translate their ideas into visual representations. However, beyond this creative aspect, these algorithms hold the potential to transform a variety of industries, including content creation, digital art, and e-commerce.

Despite their remarkable achievements, it's important to note that text-to-image algorithms are relatively new, but their development has been marked by the rapid pace characteristic of advancements in the field of artificial intelligence. The earliest efforts, employing variational autoencoders (VAE) and generative adversarial networks (GAN), trace back to 2015. Up until 2020, GAN-based models represented the cutting edge in the realm of text-to-image generation, yielding respectable results but often on limited datasets. The real explosion in popularity occurred between 2021 and 2023, a period characterized by the emergence of algorithms based on diffusion models and the continuous release of models producing increasingly astounding results. Models such as DALL-E, Stable Diffusion, Imagen, and Midjourney, particularly as they were made open source, demonstrated to the world the immense potential of text-to-image algorithms.

In this work, the aim is to test this potential, specifically to evaluate the usefulness and effectiveness of text-to-image models as tools for validating the results of image captioning models. The most common metrics for evaluating the outcomes of image captioning algorithms primarily rely on comparing the generated caption with a set of reference descriptions. However, in major benchmark datasets (e.g., MS-COCO), these reference descriptions are often simplistic and inaccurate, incapable of reflecting common natural language usage and meeting user expectations. Consequently, automatic evaluation of image captioning model results doesn't always align well with human judgment. For this reason, by leveraging text-to-image models, this thesis proposes a different perspective, shifting from the language space to the image space. Starting with a dataset that includes eight descriptions generated by as many image captioning models for 5,000 images from the MS-COCO dataset, the idea is to employ text-to-image models to regenerate images based on the dataset's captions, determining the best caption for each original image based on its similarity to the regenerated images. In this regard, the use of various metrics for assessing image similarity is considered, with CLIPScore being identified as the best solution. Finally, while keeping CLIPScore as the reference metric, experiments are conducted to explore the optimal structure of prompts, evaluating how the use of more elab-

orate and detailed prompts can help generate images that are more aligned and faithful to the originals.
To summarize, the two main research questions are as follows:

1. Based on the similarity between the original image and the images regenerated from the dataset captions, which image captioning algorithm produces the best results for each original image? And which ones produce the worst results?

2. Does the use of more in-depth and detailed captions allow for the generation of images that are more faithful to the original, both qualitatively and in terms of CLIPScore, compared to the use of basic prompts?

The thesis is structured as follows. Chapter 2 provides a theoretical background on the emergence and development of text-to-image models, focusing on the fundamental concepts underlying these models. Chapter 3 outlines the methodology followed for the practical part of the work, listing the data, Python libraries, and models used, and defining the various project steps. Chapter 4 presents the results of the different phases of the work, along with a series of considerations explaining their significance. Finally, Chapter 5 draws the conclusions of the work, summarizing the main results and addressing the research questions. Additionally, it also contains some ideas for the future development of this project.

Chapter 2

Theoretical background and related works

2.1 Early works

One of the early pioneering works in the field of text-to-image generation is alignDRAW (Mansimov et al., 2016). The proposed framework is based on the Deep Recurrent Attention Writer architecture (Gregor et al., 2015), which is extended allowing to condition the image generation process through textual descriptions.

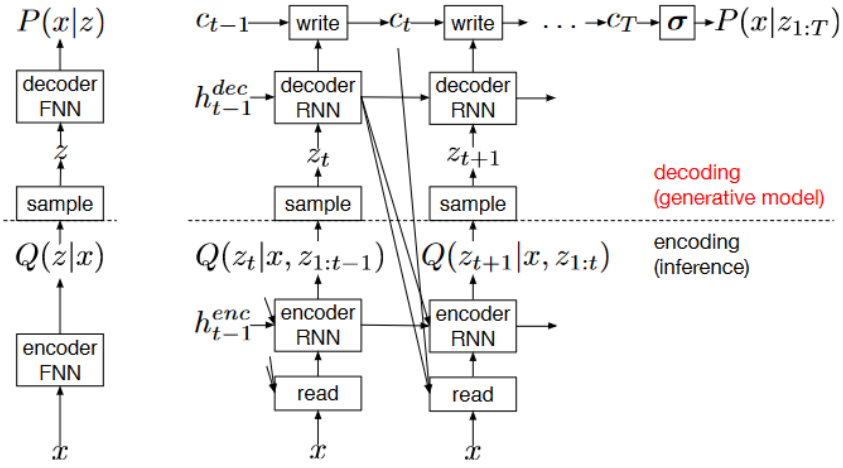

Figure 2.1: Left: Conventional Variational Auto-Encoder. Right: DRAW Network. Image taken from [17].

The DRAW architecture belongs to the family of variational autoencoders: it consists of an encoder, which captures the salient aspects of the input by determining its distribution over latent codes, and a decoder, which receives samples from the code distribution and utilizes them to condition its distribution over the images (Fig. 2.1). The uniqueness of DRAW is that the image is generated iteratively, accumulating the changes emitted by the decoder at each intermediate step into a canvas matrix c_T (Fig. 2.1). At each step, an attention mechanism is employed to constrain the input region of the encoder and the region modified by the decoder. This allows the network to selectively focus on a limited region of the image during each iteration.
In alignDRAW, the previous architecture is extended to include the caption at each intermediate step. The caption y is represented as a sequence of encoded words $y = (y_1, y_2, ..., y_N)$, and then the caption sentence representation is obtained by using a Bidirectional RNN, which transforms

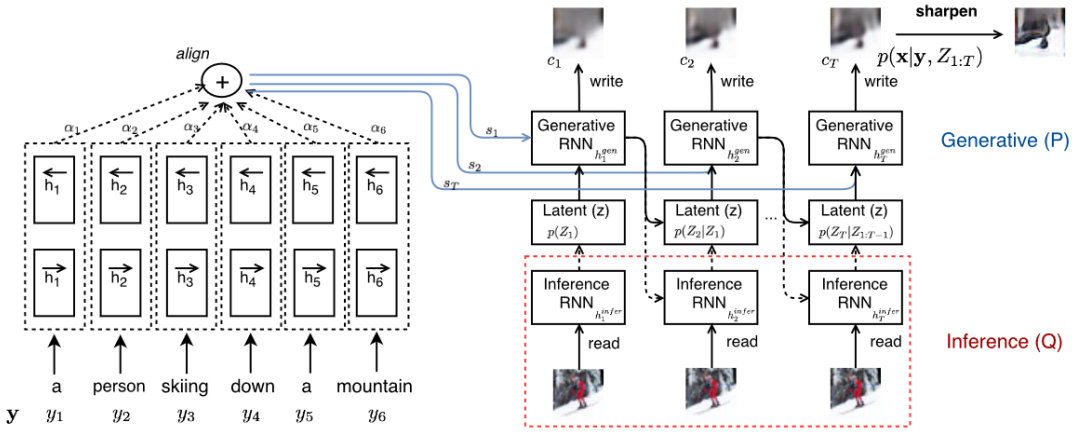

Figure 2.2: AlignDRAW architecture. Image taken from [36].

each word y_i into an m-dimensional vector. Then, at each step, the latent variables Z_t, which in the original network were simple independent spherical Gaussians, are modified to have mean and variance dependent on the caption representation h^{lang} of the previous hidden state (Fig. 2.2). Additionally, the *align* function is defined, which is used to align the input caption with each intermediate step of image generation.
AlignDRAW showed some promising results, being able to handle some basic properties of images, such as colors. For example, although in the training set all the school buses were yellow, when asked to generate a bus of a different color the model can produce a result with a consistent shape and the requested color. However, the images generated by alignDRAW are extremely blurry and lack detail.

2.2 GAN

Between 2015 and 2020, the predominant text-to-image models were based on Generative Adversarial Networks (GANs) (Goodfellow et al., 2014). GANs belong to the class of generative models: their goal is to learn a probability distribution, $p_{model}(x)$, that best approximates an unknown data distribution, $p_{data}(x)$.

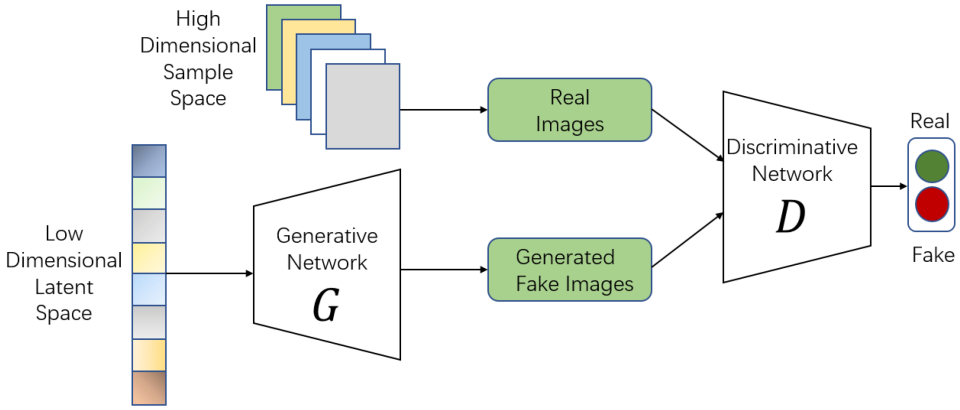

Figure 2.3: Vanilla GANs network architecture.

GANs consist of two neural networks: a generator and a discriminator (Fig. 2.3). The generator is a network whose objective is to generate synthetic data. It is defined by a prior distribution,

$p(z)$, where z is a vector serving as input to the generator function $G(z; \theta^{(G)})$, and $\theta^{(G)}$ represents learnable parameters defining the generator's behavior. The prior distribution $p(z)$ is typically an unstructured distribution, with samples z essentially being noise. The role of the generator is to learn the function $G(z)$ to transform the noise into realistic samples.
On the other hand, the discriminator acts as a binary classifier, aiming to distinguish between real and synthetic data. The discriminator receives input samples x and outputs an estimation $D(x; \theta^{(D)})$ defining whether x is real or fake.
The idea for network training is to have the generator and discriminator compete against each other in a minimax game. The two networks, each characterized by their cost function, are trained simultaneously, seeking to minimize their respective costs. By minimizing its cost function, the generator aims to maximize the probability of fooling the discriminator. Vice versa, the discriminator, by minimizing its cost function, aims to maximize its accuracy in distinguishing between real and fake samples. This adversarial training process improves both networks, with the generator becoming capable of generating increasingly realistic samples, and the discriminator becoming more accurate in recognizing real data.

2.2.1 Generative Adversarial Text to Image

The first approach to text-to-image generation using GANs came in 2016 with GAN-INT-CLS (Reed et al., 2016). The architecture is a deep convolutional generative adversarial network (DC-GAN) where both the generator and discriminator are conditioned on textual features. Regarding the textual features, the idea is to obtain a vector representation of the text that is also visually discriminative. To achieve this, the approach of [44] is followed: a hybrid character-level convolutional recurrent neural network is employed as a text encoder, trained by learning a correspondence function between text and image so that the encoded text has high compatibility with images of the corresponding class and low compatibility with images of other classes.

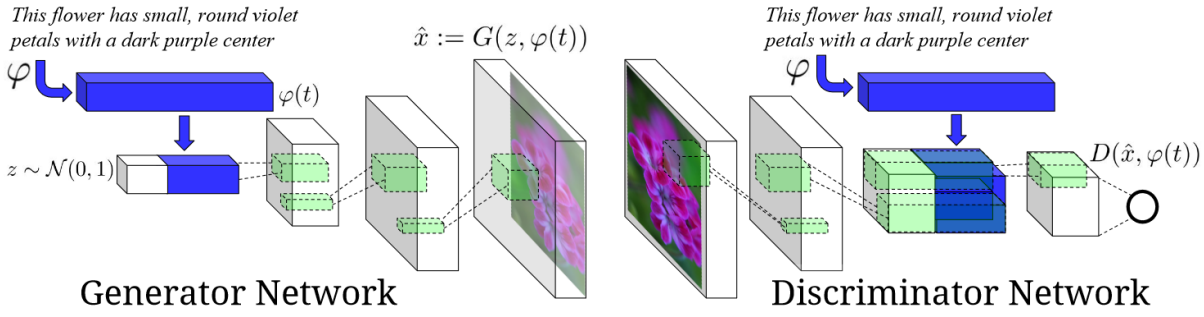

Figure 2.4: Text-conditional GANs network architecture. Image taken from [44].

At this point, the generator produces a synthetic image $\hat{x} \leftarrow G(z, \varphi(t))$, where z is the noise sampled from the prior distribution and $\varphi(t)$ is the caption embedding, conveniently compressed into a 128-dimensional vector.
Furthermore, GAN-INT-CLS introduces a discriminator training method that is used in many other works. The discriminator is trained with three possible inputs: a real image with correct text, a real image with incorrect text, and a fake image with correct text. This training approach allows the discriminator to identify both unrealistic images and the mismatch between the image and the text.
It should be pointed out that additional text embeddings, obtained by interpolating embeddings of descriptions from the training set, are also used in the training process. This addition is

motivated by the property described in [4] and [46], which shows that deep neural networks learn representations where interpolations between embedding pairs tend to be near the data manifold. The interpolated embeddings are synthetic and thus not associated with any real image. However, the discriminator is trained to assess how well an image-text pair matches. The idea is that if the discriminator can successfully work with interpolated text, the generator will be able to learn how to fill in the gaps in the data manifold between different training points.
GAN-INT-CLS is capable of generating plausible images that are at least partially coherent with the descriptions on the CUB and Oxford-102 datasets. Due to the model's construction, the obtained results strongly align with the respective class they belong to. However, on more complex datasets such as MS-COCO, the model generates sharp and fairly diverse images, but whose content still lacks detail and is often inconsistent with the description.

2.2.2 Stacked architectures

2.2.2.1 StackGAN

Reed's work served as a starting point for developing GAN-based models with increasingly satisfying results. These models can be categorized based on their specific characteristics. Stacked architectures are an example.
An unresolved issue was the generation of realistic high-resolution images. Attempts made by simply adding upsampling layers led to unstable training processes and nonsensical outcomes. The main difficulty in generating high-resolution images with GANs is that the support of the natural image distribution and the model's distribution may not overlap in the high-dimensional pixel space [2, 54]. Moreover, this problem worsens as the image resolution increases. The solution to this problem is found in the implementation of stacked architectures. In particular, StackGAN (Zhang et al., 2017) proposes to use two stages of generation: the first stage generates low-resolution images, and the second stage generates realistic high-resolution images by being conditioned not only through the caption but also with the output of the first stage. This way, the Stage-II GAN captures missing information from the Stage-I GAN's output, introducing further details into the image.

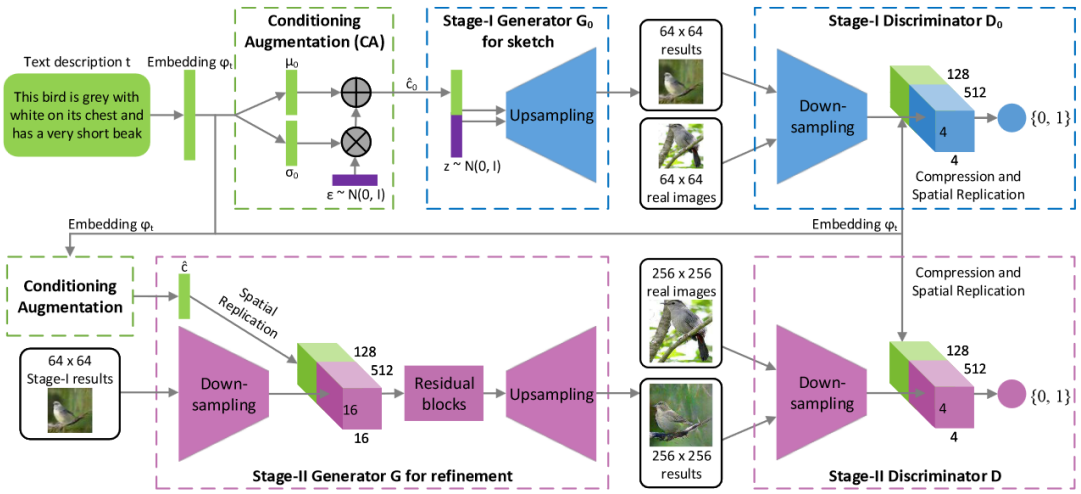

Figure 2.5: StackGAN network architecture. Image taken from [68].

Within this process, Conditioning Augmentation is also introduced. In previous works, the

text embedding is nonlinearly transformed to generate conditioning latent variables as the input of the generator. However, the latent space of the text embedding typically had high dimensionality (>100 dimensions). When limited data are available, this leads to strong discontinuities in the latent data manifold, which is not desirable for learning the generator. To mitigate this problem, the Conditioning Augmentation technique is implemented: instead of using a fixed conditioning text variable c, we sample $\hat{c}$ from an independent Gaussian distribution $N(\mu(\varphi_t), \Sigma(\varphi_t))$, whose mean and covariance matrix are a function of text embedding φ_t. This process yields more training pairs given a small number of image-text pairs, providing more robustness to small perturbations in the conditioning manifold. An additional regularization term is also added to the generator's objective function to avoid overfitting.
Returning to the architecture, the Stage-I GAN generates a low-resolution image that focuses solely on colors and rough shapes (Fig. 2.5). Similar to GAN-INT-CLS, the conditioning variable of the generator and discriminator is obtained by concatenating $\hat{c_0}$ with the noise vector z, sampled from a given distribution, usually Gaussian. At this point, the Stage-II GAN is conditioned on the low-resolution result generated in the previous stage, s_0, and the latent variable $\hat{c}$. In this stage, no noise variable z is considered, assuming that the randomness of the process is preserved by s_0. The variables $\hat{c_0}$ and $\hat{c}$ are generated from the same text embedding φ_t, but Conditioning Augmentation applied in the two stages is implemented using different fully-connected layers to generate different means and variances for the two distributions. This way, the Stage-II GAN learns to capture useful information omitted by the Stage-I GAN.
The improvements brought by StackGAN are evident: compared to GAN-INT-CLS, it can generate images with a resolution up to 256x256 while maintaining a fair level of realism, sharpness, and detail.

2.2.2.2 StackGAN++

The StackGAN architecture is further extended in its second version, StackGAN++ (Zhang et al., 2018). It is a multi-stage architecture designed for both conditional and unconditional tasks. In this architecture, multiple generators are used, sharing a significant portion of their parameters in a "tree-like" structure. The input to the network is the root of the tree, and the various branches are responsible for generating images at progressively larger scales (Fig. 2.6). The entire network is trained jointly, resulting in different but highly correlated distributions for the various branches. This way, positive feedback for a certain distribution can improve the learning of other branches.

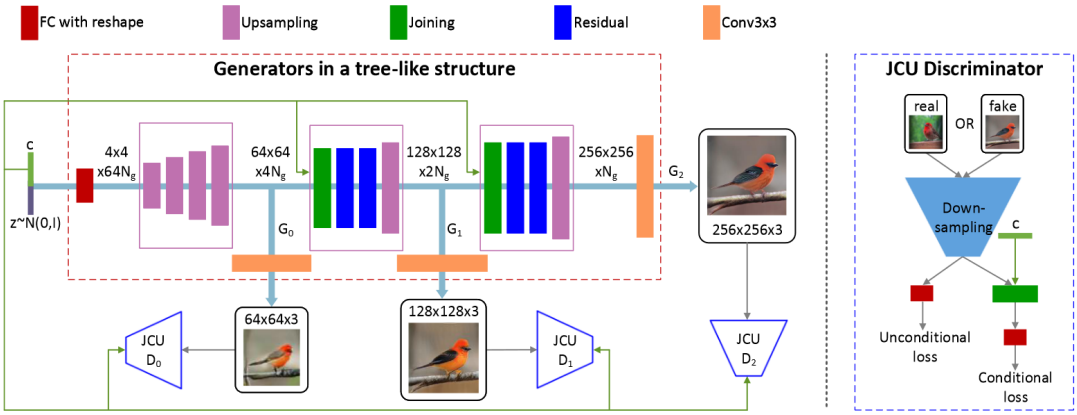

Figure 2.6: StackGAN++ network architecture. Image taken from [69].

Similar to StackGAN, each generator G_i receives as input the latent vector z concatenated with the features obtained from the previous branch, capturing previously omitted information. The first generator, G_0, receives only z as input. Each generator G_i is paired with its discriminator D_i, trained to distinguish between real and fake images and, by receiving the conditioning variable as input, to evaluate whether the image and description match. As mentioned earlier, the network is trained jointly: guided by the discriminators, the generators are optimized to approximate the multi-scale joint distribution. By modeling the data distribution at different scales, if one of the model distributions shares support with the real distribution at a certain scale, the information is propagated through the network, accelerating and stabilizing the training process. For example, if the first branch successfully approximates information about color and basic structure, this information is propagated to the other branches, which can focus on different details. Lastly, as we increase the image resolution the generated images at different scales should share similar basic structure and colors. A color-consistency regularization term is then introduced to keep samples generated from the same input at different generators more consistent in color and thus to improve the quality of the generated images.
The training process of StackGAN++ proves to be much more stable than that of previous models, improving performance, especially for the unconditional generation task. As far as the text-conditional task is concerned, the improvements are not particularly clear, probably because the insertion of textual information at each stage to ensure coherence is an extremely rigid constraint. However, StackGAN++ solves the mode collapse problem that occurred in StackGAN.

2.2.2.3 HDGAN

Following the multi-stage approach, HD-GAN (Zhang et. al, 2018), a model that overcomes the need to use multiple generators, is later published.

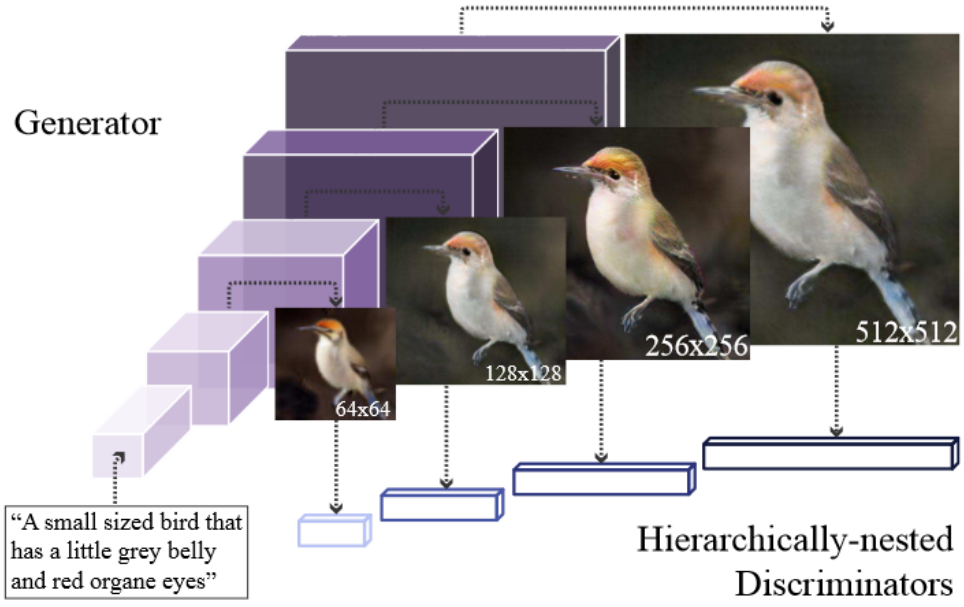

Figure 2.7: Hierarchically-nested adversarial network architecture. Image taken from [71].

The HD-GAN architecture implements a single generator but incorporates a hierarchical structure of discriminators. Each discriminator engages a minimax game with the generator at different intermediate layers, simultaneously pushing the generator to approximate the real data

distribution. Specifically, the idea is to use a generator that is always conditioned on the embedding of the image description and the noise vector, producing different outputs x_i at increasing resolutions. Each of these outputs corresponds to a different discriminator, which competes individually with the generator. Each discriminator is encouraged to learn different discriminative features, with those operating at lower resolutions focusing on the basic structure, while those operating at higher resolutions focusing more on details. Additionally, since the model is trained end-to-end, the low-resolution outputs can also benefit from the knowledge provided by the discriminators operating at higher resolutions.
To ensure both semantic consistency and image fidelity, each discriminator has two branches: one calculates the usual matching-aware pair loss [44] and one calculates a local image loss, used to distinguish between real and fake image patches. Specifically, as the resolution increases, the local image loss is calculated on an increasing number of different patches.
This model is able to surpass the performance achieved by previous models on all datasets, including MS-COCO. Moreover, it can generate images up to a resolution of 512x512 while preserving semantic consistency at all resolutions. The samples generated by HD-GAN are also more diverse and detailed compared to those obtained with the two versions of StackGAN.

2.2.2.4 FusedGAN

The success achieved with stacked architectures stimulates further variations on the theme. In particular, FusedGAN (Bodla et al., 2018) focuses on the aspect of model controllability. In addition to the fidelity and diversity of generated images, a good generative model must also ensure the ability to generate different images by keeping certain factors fixed while others are modified. A highly controllable model is also extremely appealing from a practical standpoint. To achieve this goal it is necessary to more clearly disentangle the different controllable factors in latent space. Taking inspiration from stacked architectures, the idea is to use two cascaded generators: the first unconditioned generator generates the image structure, regardless of the condition; the second conditioned generator adds style details consistent with the condition to the image. This process allows the first generator to be trained without considering the corresponding conditions, exploiting semi-supervised data. This enables learning a better structure prior, which in turn contributes to generating better and more diverse conditioned images.

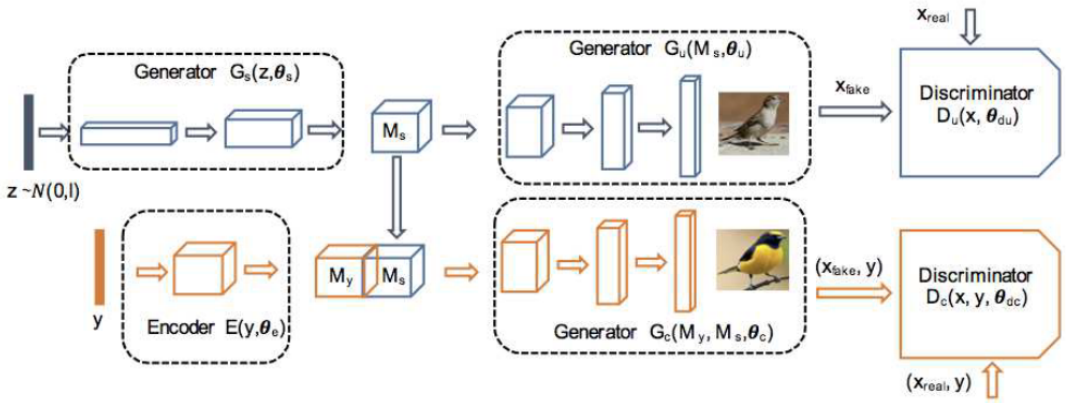

Figure 2.8: FusedGAN network architecture. Image taken from [6].

Therefore, FusedGAN consists of two stages (Fig. 2.8). The first stage performs unconditional generation, producing a feature map that serves as a structure prior to the second stage. The second stage, instead, generates the final conditioned image, using both the condition and the

structure prior as inputs. There is no explicit hierarchy between the two stages, allowing them to be trained simultaneously using an alternating optimization strategy.
This model represents the first attempt to make generative algorithms highly controllable. Previous models led to inconsistent results in terms of controllability, while FusedGAN proves to be quite flexible and reliable, at least on simple datasets like CUB. In addition, FusedGAN can be easily integrated into a stacked architecture, allowing for easy generation of high-resolution images with better performance than StackGAN.

2.2.2.5 PPAN

The latest example of a stacked architecture is PPAN (Gao et al., 2019). PPAN takes inspiration from previous stacked architectures and also integrates an attention mechanism (see 2.2.3.1). The architecture of PPAN includes a single generator and three different discriminators. This way, similarly to what happens with HD-GAN, images at different resolutions are generated in a single stage. The uniqueness is that the generator utilizes a pyramid framework (Fig. 2.9) to enhance multi-scale feature representation.

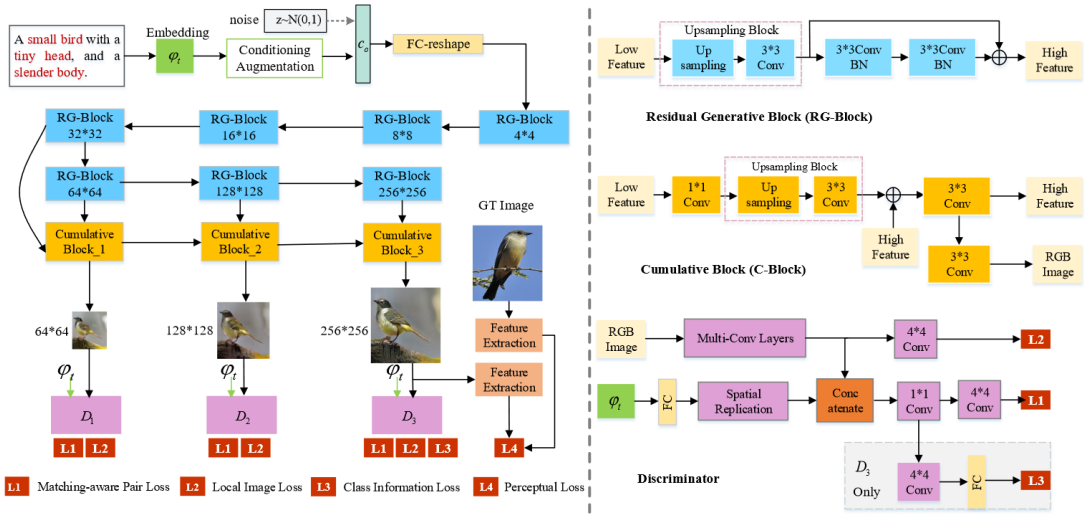

Figure 2.9: PPAN network architecture. Image taken from [15].

Exploiting the pyramid framework has been crucial in enhancing multi-scale feature representation in other computer vision tasks such as semantic segmentation, object detection [34], and super-resolution [30]. The pyramid structure constructs a down-to-top pathway with lateral connections which allows the combination of semantically powerful low-resolution features with semantically weak high-resolution features. This way, the network has rich semantics at each level. PPAN combines features from 32x32 up to 128x128 through three different Cumulative Blocks. However, as we go deeper into the network structure, the generated features become increasingly complex and less interpretable, reducing their discriminative capacity. To address this issue and make the training more stable, three different discriminators are employed, working along the depth of the network. Each discriminator has a branch for matching-aware pair loss, to address semantic consistency, and a branch for local image loss, to address image fidelity. The third discriminator, operating at a resolution of 256x256, also has a third branch that calculates a class information loss to ensure class invariance.
PPAN outperformed the state-of-the-art models of that time on all major datasets. The generated images are more diverse and realistic, managing to preserve a greater number of semantic

details. Additionally, no mode collapse issues are observed, which has been a persistent problem for previous stacked architectures.

2.2.3 Attention

So far, the common approach for encoding textual descriptions is to create a single global vector used to condition the generative process. Despite the encouraging results obtained, fine-grained details related to the meaning of individual words are lost. This prevents the generation of high-quality images, especially as the complexity of the scene increases. The innovation lies in the introduction of an attention mechanism [3] within the generative process.

2.2.3.1 AttnGAN

The first model of this kind is AttnGAN (Xu et al., 2018). AttnGAN is a network that allows attention-driven, multi-stage refinement for fine-grained text-to-image generation. The architecture (Fig. 2.10) is characterized by the introduction of two new components: an attentional generative network and a Deep Attentional Multimodal Similarity Model (DAMSM).

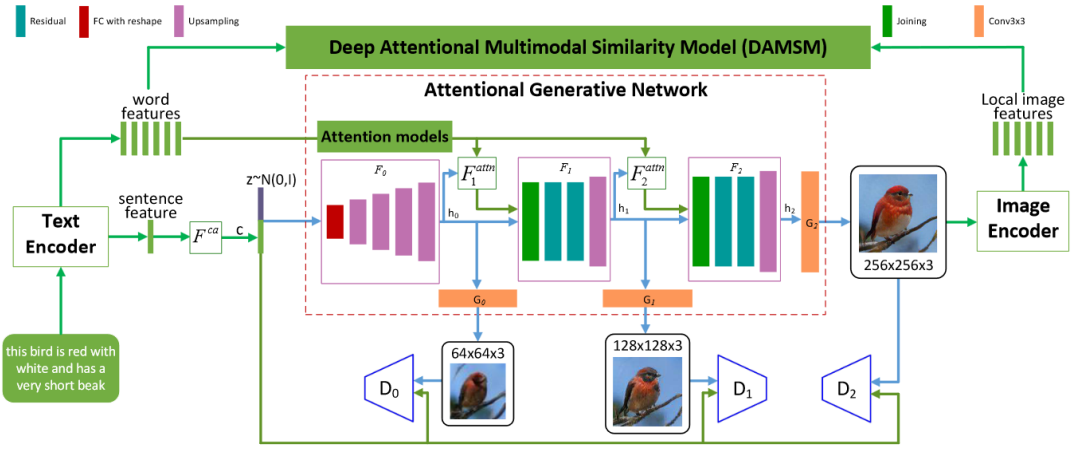

Figure 2.10: AttnGAN network architecture. Image taken from [65].

The first component introduces an attention mechanism in the generative process so that the generator draws different subregions of the images by focusing only on the most relevant words for that specific patch. This is done by encoding both the entire image description into a global sentence vector and individual words into word vectors. The global feature vector is used to generate an initial low-resolution image, while the word vectors are used in subsequent stages to refine the details of individual subregions.
The second component, DAMSM, is used to calculate the similarity between the generated image and the corresponding description, using both the global sentence information and the specific word information.
Specifically, the architecture features m generators that take hidden states as input and generate images of increasing scale. The first generator takes the combination of the noise vector z and the conditioning vector, obtained by applying Conditioning Augmentation to the global sentence vector, as input. In subsequent stages, the attention model comes into play. The attention model at the i^{th} stage, $F_i^{attn}(e, h)$, has two inputs: word features e and image features obtained in the previous stage, h_{i-1}. Each column of h_{i-1} represents the feature vector of a specific sub-region of the image. For each subregion, a word-context vector is calculated

based on the specific hidden features. This word-context vector can be seen as a dynamic representation of the most relevant words for that subregion. At this point, the output of the attention model is concatenated with the respective image features, h_{i-1}, to generate the image. Having multiple levels of conditioning (i.e. whole sentence and individual words), the loss function needs to be appropriately modified to take into account both components. Therefore, in addition to the classical components of unconditional (i.e. real-fake discrimination) and conditional (i.e. image-description matching) loss, the DAMSM loss is added. This is a fine-grained image-text matching loss at the word level, calculated by DAMSM. DAMSM is a model that learns two neural networks mapping image sub-regions and description words into the same semantic space, allowing the calculation of similarity between images and text at the word level. Specifically, the text encoder is a Bi-LSTM that extracts feature vectors for individual words and, in the final layer, combines them to calculate the global sentence vector. The image encoder, on the other hand, is a CNN that maps the image into a series of semantic vectors. The intermediate layers of the CNN learn the local features of the subregions, while deeper layers learn the global features of the image. Therefore, the local features are extracted from an intermediate layer. The similarity between the image and text is calculated based on the similarity matrix for each possible word-subregion pair and on an attention model that calculates region-context vectors for each word. Finally, the relevance of each word to the overall image is calculated, and these values are used to compute the DAMSM loss.
The introduction of DAMSM is a fundamental aspect that has allowed AttnGAN to achieve excellent performance compared to previously proposed models. In particular, DAMSM improves the image conditioning ability on complex scenes, such as those in MS-COCO. Furthermore, this model can be easily combined with stacked architectures (see 2.2.2.5), providing further room for improvement. AttnGAN is also one of the first models to show zero-shot capabilities. In fact, it manages to get decent results on prompts never seen during training, even if in most of these cases the images, despite being sharp and detailed, are not realistic.

2.2.3.2 ControlGAN

The improvement of AttnGAN is once again related to the concept of model controllability. Changing even just one word from the description leads to a completely different image, with changes that also affect attributes other than the explicitly modified one. ControlGAN (Li et al., 2019) is a model that aims to improve the controllability of the algorithm by manipulating visual attributes through textual description without influencing the generation of the rest of the content.
ControlGAN is characterized by three main components: a word-level spatial and channel-wise attention-driven generator, which uses an attention mechanism similar to AttnGAN to generate image subregions according to the most relevant words; a word-level discriminator, which considers the correlation between words and subregions to disentangle different visual attributes; and a perceptual loss used during generation to reduce randomness and force the generator to preserve the parts of the image related to the unchanged text.
The backbone of the model is a multi-stage AttnGAN. The image description is encoded by generating both global features for the sentence and features for individual words. At each stage, the spatial attention introduced in AttnGAN is used, but it is supplemented with innovative channel-wise attention. This module is introduced because the spatial attention module can only correlate words with position, without considering channel information. Experimentally, it is observed that by adding the new component, the channel-wise attention module highly correlates semantically meaningful parts with corresponding words, while spatial attention fo-

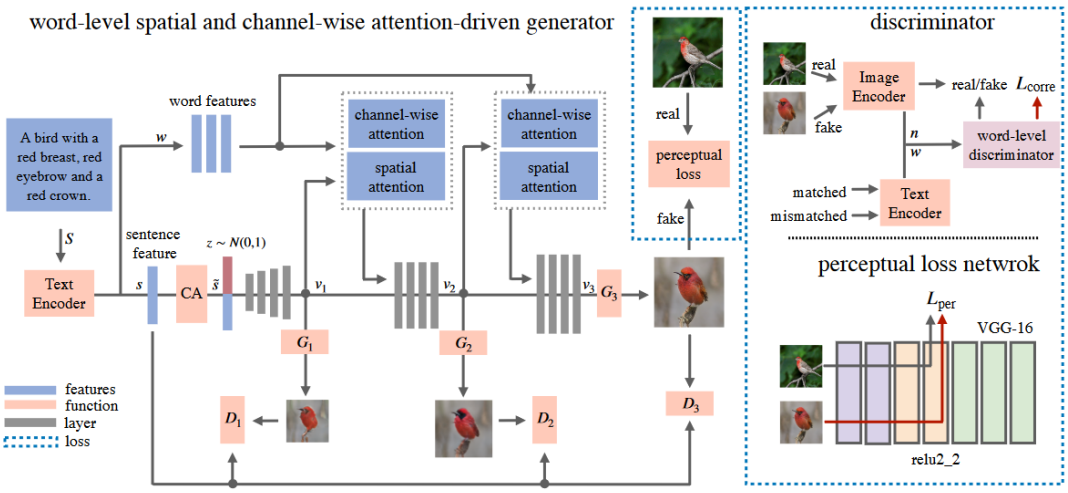

Figure 2.11: ControlGAN network architecture. Image taken from [31].

cuses on color descriptions. Working together, the two modules make it easier to disentangle various visual attributes.

The word-level discriminator is a component that is introduced to explore the correlation between image subregions and each word in the description, providing more specific feedback to the generator. The word-level discriminator takes word features encoded from the original text and a random mismatched text, as well as visual features encoded from the real image and generated images. First, a perception layer aligns the channel dimensions of visual and word features. Then, the image subregion-aware word features are computed, to generate features that take into account the relationship between each word and each subregion of the image. These features are further re-weighted through an importance vector, generated by a word-level self-attention module, to reduce the impact of unimportant words. Finally, a sigmoid function is used to compute the correlation between individual words and the whole image. The sum of these correlations returns the overall correlation between the image and the description. This fine-grained feedback from the word-level discriminator helps supervise the generation and manipulation of each subregion independently, enabling the generator to modify specific parts of the image according to the given text.

Finally, the introduction of perceptual loss during generation becomes necessary. Without adding this term, which constrains "text irrelevant" image regions such as the background, the results tend to be highly random and often semantically inconsistent with the rest of the sentence's content.

2.2.4 CycleGAN

2.2.4.1 MirrorGAN

An alternative approach to text-to-image generation comes from MirrorGAN (Qiao et al., 2019). The idea of this model is to combine text-to-image and image-to-text within the same framework. In fact, if an image generated by T2I is semantically consistent with the given text description, its re-description by I2T should have the same semantics as the given text description. In other words, the generated image should act like a mirror that precisely reflects the underlying text semantics.

The architecture of MirrorGAN is characterized by the presence of three modules (Fig. 2.12): a Semantic Text Embedding Module (STEM), a Global-Local Collaborative Attentive Module

(GLAM), and a Semantic Text Regeneration and Alignment Module (STREAM).

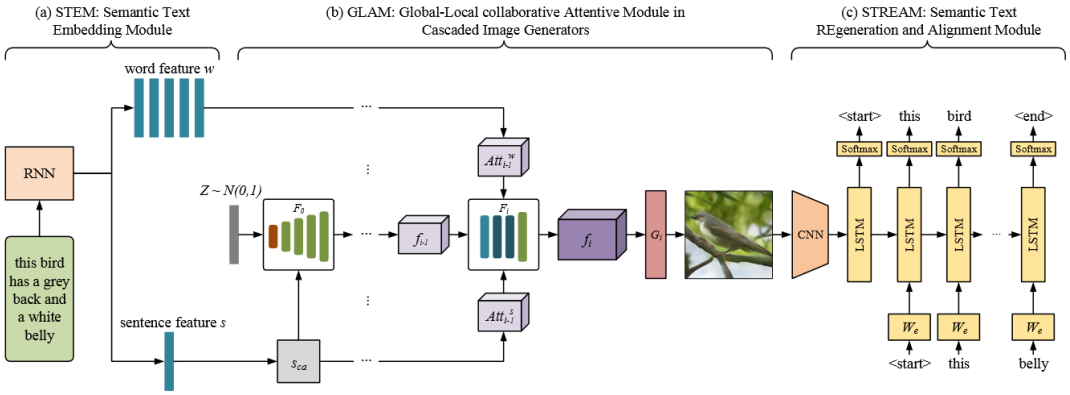

Figure 2.12: MirrorGAN network architecture. Image taken from [39].

STEM uses an RNN to generate the global sentence feature vector, extracted from the last layer, along with the individual word-level feature vectors, extracted by concatenating the hidden states of each word. Conditioning augmentation is also implemented for the same reasons explained in 2.2.2.1.
GLAM implements a cascaded multi-stage generator by sequentially stacking three image generators. The structure of the generators is similar to that used in 2.2.3.1.
Finally, STREAM is responsible for regenerating the image description. It utilizes a widely used captioning framework consisting of an encoder-decoder architecture. The STREAM module is pre-trained, which makes the training of MirrorGAN more stable and enables faster convergence.
Each stage of MirrorGAN involves training the generator and discriminator alternately. In particular, a classic objective function evaluating the realism of the image and its consistency with the description is augmented with a term of CE-based text-semantic reconstruction loss to align the underlying semantics between the re-description of STREAM and the given text description. This term is also used during the pre-training of STREAM.
While not bringing substantial improvements, MirrorGAN improves the results of all previous models on major benchmark datasets, including slight improvements on MS-COCO.

2.2.5 Memory Networks

2.2.5.1 DM-GAN

The previous multi-stage and attention architectures have led to important progress, but they exhibit two main problems. Firstly, the outcome of the generation strongly depends on the quality of the initial low-resolution image: the image refinement process implemented in architectures like StackGAN fails to produce high-quality images if the initial generation is poor. Secondly, each word in a description carries a different level of importance in conveying the image content: so far, models have used the same word representation at every step to refine the image, resulting in an inefficient refinement process. Image information should be taken into account to determine the importance of each word.
DM-GAN (Zhu et al., 2019) aims to address these two problems by drawing inspiration from memory networks, architectures in which information is written to external memory and later retrieved as needed. For the first problem, a memory mechanism is proposed to handle low-quality initial images, where image features are used as queries to retrieve similar images to

refine the initial image. For the second problem, a memory writing gate is proposed to dynamically select the relevant words for the generated image. In general, during each image refinement process, the memory is read and written according to the initial image and its corresponding description.

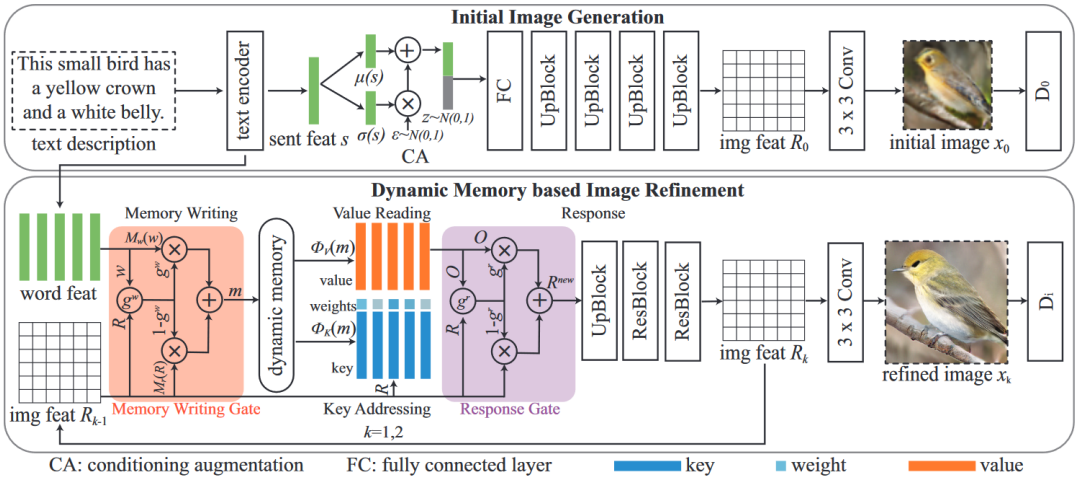

Figure 2.13: DM-GAN network architecture. Image taken from [72].

The architecture of DM-GAN is characterized by two stages (Fig. 2.13). The initial image generation stage takes the text embedding of the textual description concatenated with noise as input to generate the initial image. The dynamic memory image refinement stage adds fine-grained content to the original image to generate photorealistic images. This second stage can be iterated multiple times to generate images with increasingly higher resolution and more details.
The dynamic memory image refinement stage consists of four components:

- The memory writing component stores information related to the image and its description in a value-key structured memory for future reuse. A memory writing gate is used to dynamically select the importance of individual words in the description for the current image. The gate combines the image features from the previous stage with the word features to calculate the importance of each word. The importance value, which is a combination of visual and textual features, is then written into memory;

- The key addressing and value reading components are used to retrieve useful features for image refinement from the memory. Relevant elements are extracted by calculating the similarity probability between each memory slot and each image feature. This probability is used to weigh the elements in memory during the value reading phase;

- The response component controls the process of merging the image features with those retrieved from memory. The merging occurs in a naive manner through a simple concatenation process.

With DM-GAN, performance on benchmark datasets is further improved. The generated images show vivid details and a more defined structure. The improvements are especially important on MS-COCO, where DM-GAN proves able to correctly capture the main subject of the image by arranging the rest of the scene in a coherent way.

2.2.6 Siamese Architectures

2.2.6.1 SD-GAN

Another important approach to text-to-image generation is that of siamese architectures. So far, the models used have focused solely on improving the visual quality of images through stacked architectures or attention mechanisms. These methods do not take into account the linguistic diversity that characterizes different possible descriptions of the same image. Using a single description to generate images can lead to highly unstable results, and there is a need for a method that can handle a wide variety of linguistic expressions.

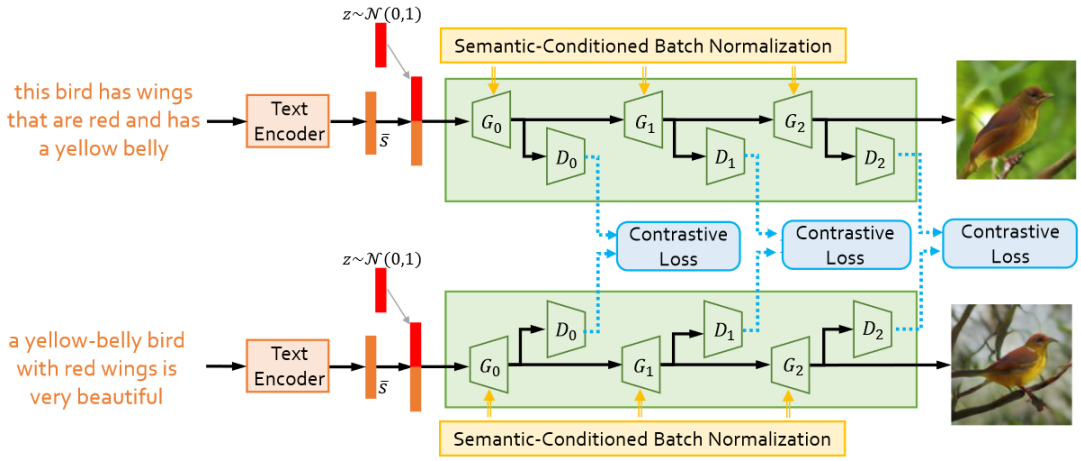

Figure 2.14: Semantic Disentangling GAN network architecture. Image taken from [66].

SD-GAN (Yin et al., 2019) is a model that exploits a siamese architecture (Fig. 2.14), to distill the semantic similarities between different descriptions of the same image for a more consistent generation while maintaining the diversity and details of the different captions for fine-grained refinement. The idea is to treat the discriminator as an image comparator, to preserve the semantic consistency of images generated from different but related descriptions. The scheme used by SD-GAN takes a pair of texts as input and is trained with a contrastive loss: intra-class image pairs (i.e. same ground-truth image, different description) should have the smallest possible distance in the discriminator's feature space, while inter-class pairs (i.e. different ground-truth image, different description) should have the largest possible distance. By using contrastive loss, the network is forced to maintain the common semantics among different texts. Additionally, to utilize the different details of individual descriptions for refinement of the images, a Semantic-Conditioned Batch Normalization (SCBN) layer is introduced in the generator.

To go into more detail, each branch of the siamese architecture adopts a structure inspired by StackGAN and AttnGAN. The description is encoded with a Bi-LSTM, as in StackGAN. The resulting embedding is concatenated with a noise vector, generating the conditioning vector used in a series of hierarchical stages that produce images at increasing resolutions. Each stage also uses the output of the previous stage as conditioning and is followed by its discriminator. Thus, the main innovation is the introduction of contrastive loss, used for the reasons previously described.

The other introduction is that of the conditioned batch normalization layer. The SCBN layer is introduced to realize a batch normalization modulated according to the linguistic cues of the description. The goal is to manipulate the feature map of the generator network by emphasizing,

suppressing, or negating certain features. In this way, it complements the work of the siamese architecture, which focuses on the common semantics while disregarding the unique semantic details of each description. Batch normalization normalizes the mean and standard deviation of each feature channel as follows:

$$BN(x) = \gamma \frac{x - \mu(x)}{\sigma(x)} + \beta$$

where γ and β are affine parameters learned from data. For conditioned batch normalization, it is necessary to introduce modulating parameters, γ_c and β_c, defined based on the conditioning variable:

$$BN(x) = (\gamma + \gamma_c) \frac{x - \mu(x)}{\sigma(x)} + (\beta + \beta_c)$$

In this case, these parameters are calculated taking into account both the global feature sentence vector $\overline{s}$ and the individual word feature vectors w_t.
The results obtained by SD-GAN on benchmark datasets significantly improve the performance of previous models, especially on MS-COCO, where SD-GAN is able to generate much more recognizable and ground-truth-like images.

2.2.7 GAN with Additional Supervision

Up until this point, all the proposed models, while progressively improving performance by adding increasingly specialized components, are characterized by conditioning on a single caption, except for SD-GAN, which uses two descriptions and, unsurprisingly, achieves the best results. Hence, several approaches have been proposed which introduce additional supervision to the generation process. The following models have demonstrated the ability to improve state-of-the-art performance but at the cost of increased complexity.

2.2.7.1 RiFeGAN

The example of SD-GAN has already shown that using multiple descriptions generally leads to higher-quality generated images. The idea behind RiFeGAN (Cheng et al., 2020) is to leverage multiple captions to guide the generative process.
Initially, RiFeGAN uses the caption to retrieve other compatible descriptions by comparing their respective text embeddings. Therefore, the generative process will be guided by a set of n textual descriptions. However, managing a large number of captions is complex. Hence, RiFeGAN introduces a Self-Attentional Embedding Mixture (SAEM) which merges all candidate captions into a single short, concise, and easily manageable embedding.
RiFeGAN uses a three-stage architecture with attention modules. In the first stage, an initial image is synthesized by considering the global embedding produced by SAEM. The two subsequent stages take the features obtained from the previous stage and concatenate them with the features produced by the respective attention module. The attention modules work by calculating attentional features for each of the n descriptions; then SAEM is used again to calculate the overall attentional features.
To guide the training of the network, a Multi-Caps DAMSM loss is used, which is the weighted sum of the DAMSM loss (see 2.2.3.1) for each of the n considered captions.

2.2.7.2 Semantic Layout

Another example of additional supervision is the use of a semantic layout. One of the main challenges in image generation is handling complex scenes characterized by multiple objects, relationships between objects, and specific spatial configurations. The idea, therefore, is not to learn a direct mapping from text to image but to pass through an intermediate representation called a semantic layout. By semantic layout we mean a rough structure of the scene, defining the number of objects, their category, position, size, etc.
A first example of a model of this type is the one proposed by Hong et al. (2018). The model consists of two parts: a layout generator, which produces a label map from the textual description, and an image generator, which converts the layout into an image while considering the text. Furthermore, the generation of the layout is further divided into two phases: first, a bounding box layout is generated using a box generator, and second, the shapes inside each box are refined using a shape generator. In total, there are three networks (i.e. box, shape, and image generator) trained in parallel, each with its respective supervision.

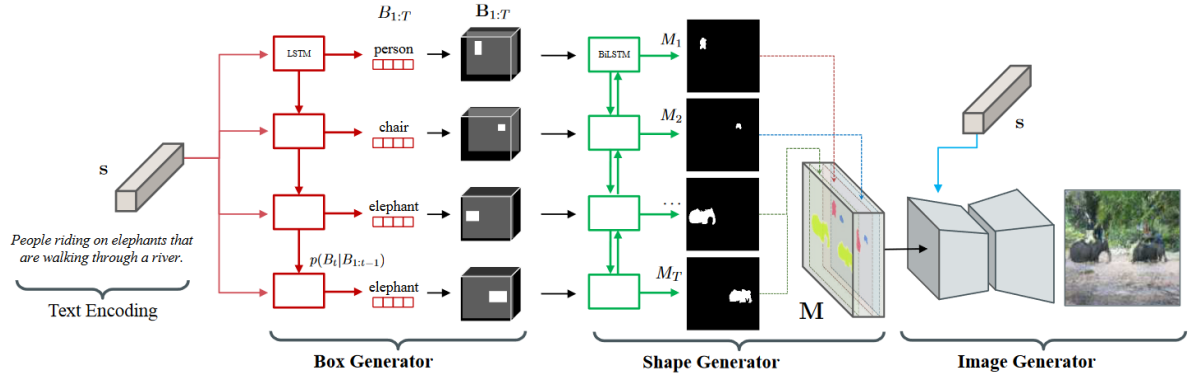

Figure 2.15: Semantic layout models pipeline. Image taken from [23].

Going into detail about the individual components, the box generator is a network that takes as input the embedding s obtained from the image description and produces a set of bounding boxes $B_{1:T} = B_1, ..., B_T$. Each element $B_i = (b_i, l_i)$ consists of information about the spatial position of the box b_i and the object category to be placed l_i. The box generator is trained by minimizing the negative log-likelihood with respect to the ground-truth bounding boxes.
The bounding boxes are then fed into a shape generator, which refines the shape of the object contained in each box and outputs a set of masks $M_{1:T} = M_1, ..., M_T$ that define the shapes of the objects in the entire scene. The shape generation process is implemented by imposing two types of constraints: an instance-wise constraint that ensures that each mask M_i matches the position and class information of the corresponding box B_i, and a global constraint that aligns the shape with the surrounding context. The framework of this component is that of a traditional GAN, with two different discriminators to account for both constraints. Additionally, a perceptual loss is considered, which measures the distance between the generated mask and the ground-truth mask.
At this point, the image generation must adhere to both the text and the produced semantic layout. Therefore, a semantic label map M is generated by aggregating the masks related to individual objects. The map M is given as input to the image generator along with the text and the noise vector. The network used is an encoder-decoder network with some modifications. The layout M is first encoded with downsampling layers to generate layout features. Then, to adaptively select a context relevant to the text, an attention module is applied to the layout

feature. The obtained layout features are then concatenated with the text and noise along the channel dimension and subsequently mapped to an image. The map M is also used within the discriminator to evaluate the alignment between the generated image and the original layout.
Compared with the models already released in 2018, the results are better than StackGAN, with qualitatively more recognizable and semantically more meaningful images. It is demonstrated that the images are strongly correlated with the semantic layout, and the produced layout is crucial in generating realistic images.
An evolution of the previous model comes with ObjGAN (Li et al., 2019). Image generation still involves creating a semantic layout, but in this case, the goal is to use fine-grained word/object level information more efficiently by using a pair of object-driven attentive image generators, an object-wise discriminator, and an innovative object-driven attention mechanism. The proposed image generator takes as input the text and the semantic layout to synthesize high-resolution images through a multi-stage process. At each stage, the new object-driven attention mechanism is utilized, allowing the generator to focus on the most relevant words for the object present in each bounding box region of the image. ObjGAN proposes only two stages, but it can be easily expanded for a deeper refinement process.
The new object-driven attention mechanism leverages class labels of the box to query words in the sentences, forming a word context vector and then synthesizing the image region conditioned on the class label and word context vector. Additionally, patch-wise context vectors related to subregions of the image are still calculated, as in traditional attention modules. The image is divided into uniform and semantically non-significant patches.
As for the discriminator, both a patch-wise discriminator and the new object-wise discriminator are used. The patch-wise discriminator evaluates the realism of individual patches and their alignment with the textual description. The object-wise discriminator, on the other hand, determines the realism and consistency with the corresponding label of the region contained within the bounding box.

2.3 Autoregressive Models

All previous works have the common characteristic of being based on the GAN structure. The introduction of additional modules has progressively improved the quality of the generated images, showing some promising signs even in terms of generalization to unknown image categories. However, these models have always been trained and evaluated on relatively small datasets, such as CUB-200, Oxford-102, and MS-COCO. The impressive results achieved by large-scale autoregressive transformers [58] in various domains, such as text [40], images [8], and audio [11], led to the conviction that the size of the datasets and models used so far has been a major limitation.
Indeed, the extension to large-scale generative models, as well as the use of autoregressive transformers and feature discretization techniques, represents the key to the significant development of text-to-image algorithms.

2.3.1 Pre-training

The main reason that limits the generalization capabilities of GAN-based models is the fact that they are trained on fixed and limited datasets. Increasing the size of these datasets requires a long and resource-intensive phase of data labeling. For this reason, the best approach to increase the size of models is through the use of pre-training techniques.

Pre-training refers to the process of training a model on a large-scale dataset using unsupervised learning tasks, and then fine-tuning it for a specific downstream task. This process is typically performed on a large dataset of unlabeled data, such as text, images, or audio. The idea behind pre-training is to allow the model to learn a representation or useful features from the data that can be transferred to other similar tasks. The concept of pre-training is somewhat inspired by human behavior: we rarely have to learn everything from scratch, but rather transfer and reuse our prior knowledge to handle new concepts and tasks.
As mentioned above, once the pre-training phase is completed, the model is fine-tuned on a smaller labeled dataset for the specific task of interest. In this way, the model benefits from the knowledge and patterns captured during the unsupervised learning phase, improving its performance and achieving faster convergence on the downstream task. Pre-training has proven to be highly effective in many areas of AI, such as natural language processing, computer vision, and speech recognition.

2.3.2 Autoregressive Transformers

Autoregressive transformers are a specific type of neural network that combines the autoregressive modeling approach with the transformer model.
The transformer model, introduced by Vaswani et al. in 2017, is a highly popular approach for various natural language processing tasks. Transformers are characterized by an encoder-decoder architecture, heavily based on the self-attention mechanism and positional encoding. The self-attention mechanism allows to capture relationships and dependencies among words both locally and globally, effectively modeling long-range dependencies. Positional encoding, on the other hand, is introduced to provide the model with information about the position of a specific word in the sequence.
In the context of autoregressive transformers, the model generates output tokens one at a time, conditioning on the previously generated tokens. The autoregressive generation process starts with an initial input and proceeds sequentially, typically from left to right, generating the next token based on the previous ones. Token prediction is usually performed through a softmax layer over a fixed vocabulary, assigning a certain probability to each possible token. The predicted token is appended to the input sequence, and the process is repeated until a sequence of the desired length is generated. The autoregressive nature of this process allows the model to capture sequential dependencies present in the data. By conditioning on the previously generated tokens, the model learns how to generate coherent and contextually relevant sequences.
These models are typically pre-trained on a large corpus of unlabeled data. The most common pretraining task is masked language modeling (MLM), where a certain percentage of input tokens are masked, and the model is trained to predict the original values of the masked tokens based on the surrounding context. Another common pretraining task is left-to-right token prediction, where the model is trained to predict the next token based on the input sequence by minimizing the difference between the prediction and the original value of the token.
Autoregressive transformers can be used for a variety of tasks, including language modeling, text generation, machine translation, and even text-to-image generation. In text-to-image generation, the autoregressive transformer model takes textual descriptions as input and autoregressively generates corresponding image pixels. The model attends to the previously generated tokens, capturing the relevant information necessary to generate coherent and visually appealing images based on the textual input.
Overall, autoregressive transformers combine the power of the transformer model's attention mechanisms with the autoregressive generation process, enabling the generation of high-quality

and contextually relevant sequences in various tasks.

2.3.3 Discrete Representation

Another change concerns the encoding of images. Previous works have always represented images in a continuous space, such as the pixel space. However, transferring this type of representation into the realm of autoregressive transformers poses a problem. The use of continuous features leads the models to require enormous amounts of memory and to focus on modeling imperceptible details rather than what makes recognizable the object we want to draw. Furthermore, a considerable portion of the data we encounter in the real world is discrete: the images themselves contain discrete objects with discrete characteristics.
Therefore, there is a shift towards employing discretization techniques to represent images more efficiently in a discrete latent space with a small number of dimensions.
Discretization involves converting continuous values into a smaller set of discrete values. In text-to-image algorithms, the most commonly used discretization technique is vector quantization (VQ). In vector quantization, the continuous data space is divided into a predefined number of regions, each characterized by a codeword, which represents a significant representative of its respective region. Once the codebook, which consists of all the codewords, is learned, data points are mapped to the nearest codeword, generating a quantized version of the original data. This technique is particularly flexible as it allows learning the codebook directly from the data space, without having to resort to the use of predefined discrete vectors. In generative models, this representation is highly efficient as it reduces the amount of information to store or transmit while preserving the important characteristics of the data.

2.3.4 DALL-E

One of the first text-to-image models to implement an architecture based on an autoregressive transformer pre-trained on a large-scale dataset is DALL-E (Ramesh et al., 2021). Specifically, DALL-E is a transformer with 12 billion parameters, trained on 250 million image-text pairs collected from the internet.
The idea is to train a transformer to autoregressively model both textual tokens and image tokens as a single stream of data. The model training procedure is characterized by two stages:

- In the first stage, a discrete variational autoencoder (dVAE) is trained to compress each 256x256 RGB image into a 32x32 grid of image tokens. Each element of the grid can take one of 8192 possible values. As mentioned in section 2.3.3, this stage reduces the dimensionality of the transformer by a factor of 192 without significantly degrading the image.

- In the second stage, up to 256 BPE-encoded text tokens are concatenated with the 1024 image tokens. BPE stands for Byte Pair Encoding [52], a text encoding algorithm used in language models like GPT-3. The transformer is then trained to model the joint distribution over text tokens and images.

This entire procedure can be seen as maximizing the evidence lower bound [28, 47] on the joint likelihood of the model distribution over images x, captions y, and the tokens z for the encoded RGB image. The distribution is modeled using the following factorization:

$$\ln p_{\theta,\psi}(x,y) \geq E_{z\sim q_{Phi}(z|x)}(\ln p_\theta(x|y,z) - \beta D_{KL}(q_\phi(y,z|x), p_\psi(y,z)))$$

where q_ϕ is the distribution over the 32x32 image tokens generated by the dVAE encoder, p_θ is the distribution over the RGB images generated by the dVAE decoder given the image tokens and p_ψ is the distribution over the text and image tokens modeled by the transformer.
To briefly describe the training process, the first stage involves maximizing the ELBO with respect to (ϕ) and (θ), which corresponds to training the dVAE on images alone. At the end of this stage, the visual codebook for image discretization is learned. In the second stage, on the other hand, (ϕ) and (θ) are fixed, and the objective is to learn the prior distribution over text and image tokens by maximizing the ELBO with respect to (ψ). Therefore, (p_ψ) refers to the 12-billion-parameter transformer. As mentioned earlier, images are encoded into 1024 tokens with a vocabulary size of 8192. Instead, captions are encoded using up to 256 tokens generated by a BPE-encoding algorithm, with a vocabulary size of 16384. Finally, the tokens are concatenated and autoregressively modeled as a single stream of data. In cases where the caption is encoded with less than 256 tokens, the space between the last textual token and the start of image tokens is filled with special tokens. It should be noted that the transformer is a decoder-only model since it takes the image output from the dVAE. In the 64 layers of self-attention in the transformer, each image token can attend to all the text tokens.
The model is evaluated and compared with previous approaches on the MS-COCO dataset. For each caption, N samples are generated, from which the best ones are selected using CLIP (see 2.4.4.2.1), which determines how well the image matches the caption. User studies conducted show that images generated by DALL-E are considered more faithful to the description in 93% of cases and more realistic in 90% of cases compared to GAN-based models. On the MS-COCO dataset, there is an improvement of nearly 2 points in the FID score compared to the state-of-the-art GAN models. All of this is achieved without the model being specifically trained on the same MS-COCO dataset. Overall, the model demonstrates a good level of generalization, even to rather unusual objects. In terms of combinatorial generalization (i.e. generalization of the relationship between objects), there are encouraging signs, but the behavior is still unstable and inconsistent.

2.3.5 CogView

Parallel to DALL-E, the CogView model (Ding et al., 2021) is proposed. Being based on the same principles, CogView is characterized by a transformer with 4 billion parameters, pretrained on a Chinese dataset consisting of 30 million high-quality text-image pairs. Compared to DALL-E, the main contribution brought by CogView is the different way in which it handles the instability problem in the pretraining process due to the significant heterogeneity within the data.
Similarly to DALL-E, the training process is divided into two stages. The first stage degenerates into a discrete auto-encoder, which compresses the images by encoding them into a sequence of discrete tokens belonging to a codebook, learned by minimizing the reconstruction loss. The second stage, consisting of the transformer, takes on the task of modeling by working on sequences of tokens from the discrete codebook.
Unlike DALL-E, text tokenization is performed using the SentencePiece algorithm [29]. The text tokenizer is trained on a large Chinese textual corpus to extract 50,000 text tokens. The image tokenizer, on the other hand, maps the image to a 32x32 grid, with each grid component quantized to the nearest embedding in the codebook. As in DALL-E, the codebook size is 8192.
Regarding the architecture used, the backbone of CogView is a unidirectional transformer (GPT). The input given to the transformer is the concatenated sequence of text and image tokens, to which some special tokens indicating the boundaries of the text and image are added

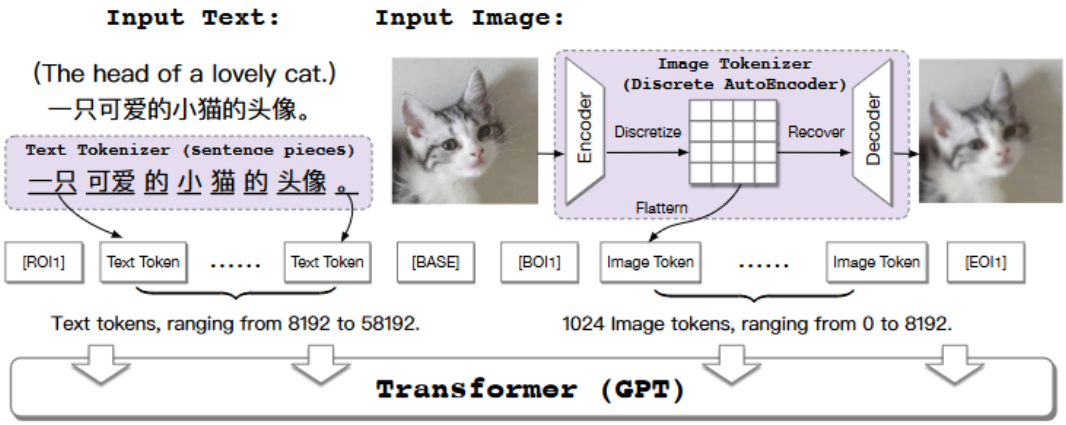

Figure 2.16: CogView framework. Image taken from [12].

(Fig. 2.16). The input sequence is clipped or padded to a fixed length of 1088. The transformer is pre-trained on a left-to-right token prediction task, treating both text and image tokens equally. This is another difference compared to DALL-E, which suggested lowering the loss associated with text tokens. In fact, based on some small-scale experiments, CogView demonstrates how proper text modeling is crucial for the success of the pretraining task. For example, setting the loss weight associated with text tokens to zero results in the model being unable to establish connections between text and image, generating completely unrelated images to the description. It is hypothesized that text modeling abstracts knowledge in hidden layers, which can be efficiently exploited during image modeling.
This leads to the issue of stability. The common choice when pretraining a large model with over 2 billion parameters is to use 16-bit precision, which optimizes GPU memory usage and speeds up computation. However, it can be demonstrated that text-to-image pretraining encounters significant instability problems under 16-bit precision. DALL-E's solution to this problem essentially involves tolerating it: DALL-E manages and stores gains, biases, and embeddings in 32-bit precision. However, this solution is complex and extremely resource-intensive in terms of time and memory. Additionally, it is not supported by most current training methods. In contrast, CogView tackles the problem by attempting to regularize it. Noticing the presence of two types of instability (i.e. overflow and underflow), two regularization techniques are implemented: Precision Bottleneck Relaxation to address the overflow problem, which occurs at two bottleneck operations, and Sandwich LayerNorm for the underflow problem [12].
Despite being trained on much less data and having much fewer parameters, CogView achieves results comparable to those of DALL-E, demonstrating the effectiveness of the different type of encoding used and of the stabilization techniques introduced.

2.4 Diffusion models

In late 2021, another technique emerged as the new state-of-the-art in text-to-image generation. These are denoising diffusion probabilistic models (DDPMs).

2.4.1 Denoising Diffusion Probabilistic Models

Diffusion models are a family of generative models consisting of Markov chains trained with variational inference. The goal of these models is to generate high-quality samples by learning

to invert a data perturbation process, known as the diffusion process.
Diffusion models gained popularity with Ho et al., 2020 and have since attracted more and more interest in the generative models community. In particular, the advent of DDPMs in computer vision can be attributed to two previous works that form their foundations:

- The diffusion probabilistic model proposed in 2015 by Sohl-Dickstein [53] is one of the earliest attempts to model a probability distribution by inverting the Markov diffusion chain that maps the data to a simple distribution. In short, DPMs define a forward process that converts a complex distribution into a much simpler one and then learns to invert this diffusion process. DPMs can be considered the basis of DDPMs;
- Also highly relevant is the score-based generative model (SGM) [55]. SGMs propose perturbing the data with Gaussian noise of varying magnitudes. Then, SGMs generate samples by reducing the noise levels and train the model by estimating the score functions for the noisy data distribution. DDPMs share a very similar objective function to SGMs during training.

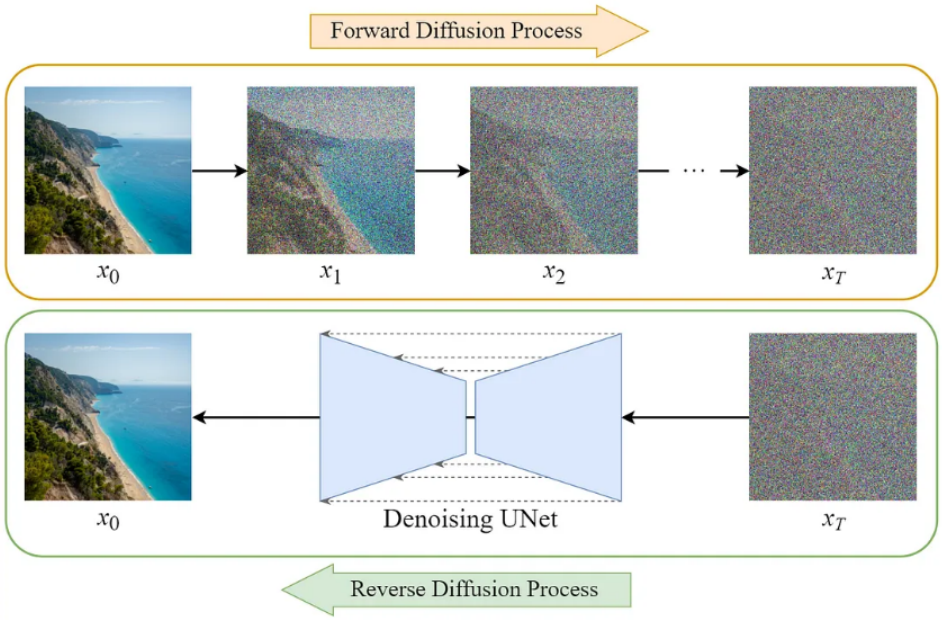

Figure 2.17: Overview of Diffusion models for image synthesis.

Therefore, in the specific case of image synthesis, DDPMs can be defined as a parameterized Markov chain trained using variational inference to produce high-quality images matching the data through a finite number of transitions. Those transitions are learned to reverse a diffusion process, which is a Markov chain that gradually adds noise to the data until the signal is completely destroyed. In particular, the diffusion consists of small amounts of Gaussian noise added to the data for a finite number of steps. Given a data distribution $x_0 \sim q(x_0)$, in the forward pass DDPMs generates x_T in the following way:

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

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

where T is the number of diffusion steps and β_t are hyperparameters. Essentially, what is done in the forward pass is the conversion of an unknown and complex distribution into a much simpler distribution from which it is easier to extract samples. At the end of the forward pass, the images have been mapped to a space outside their subspace and have become completely unrecognizable. At this point, the idea is to reverse the process of diffusion. The reverse pass starts from where the forward pass stopped, sampling a point from the simple distribution. The model is trained to try to return to the data subspace through a series of small transitions that iteratively remove noise from the extracted sample. Once the model is trained, it can start from any point in the simple distribution and bring us back to the data subspace, thus producing images belonging to the original data distribution.

2.4.2 Conditioned and Guided Diffusion Models

Our interest is to generate images that are coherent with their respective textual description. Therefore, it is necessary to introduce this information into the process just described. In diffusion models, the introduction of auxiliary information to guide the sampling process can occur in two ways:

- In conditional diffusion models, the conditioning variable is provided as input to the model during both the forward pass and the reverse pass. The additional information can be in the form of a class label, textual description, or any other type of conditioning variable. By conditioning the model on the provided information, it is capable of generating samples aligned with the desired condition;
- In guided diffusion models, the additional information is utilized only during the reverse pass. During the sampling process, gradients based on the provided condition are introduced. These gradients are used to direct towards generating samples aligned with the desired characteristics.

Guided diffusion models can be further divided based on the type of guidance. Specifically, we can distinguish between classifier-guided diffusion models and classifier-free diffusion models. Classifier-guided diffusion models [10] use an auxiliary classifier that computes a loss signal during the sampling process. For example, the loss signal can be related to how well the generated image matches the textual description. The signal from the classifier is then used to guide the inference process, directing it toward generating an image aligned with the condition. The main drawback of this technique is that the classifier needs to be trained on noisy images, excluding the possibility of plugging in a pre-trained classifier.
On the other hand, classifier-free guidance [22] is a method that completely avoids the use of an auxiliary classifier. Instead, an unconditional DDPM and a conditional DDPM are jointly trained. A single neural network is used to parameterize both models, where the unconditional model can be obtained by simply setting the conditioning variable to a null value. The sampling process is then carried out by leveraging information from both the conditional and unconditional models. In fact, the score used to guide the denoising transitions is a linear combination of conditional and unconditional scores. Despite having a simpler structure, empirical evidence shows that classifier-free guidance is as effective as classifier guidance.

2.4.3 Pixel-space Frameworks

Text-to-image diffusion models can be broadly divided based on the space in which they operate. Methods like GLIDE and Imagen generate images directly from the pixel space. Others,

such as Stable Diffusion or DALL-E 2, first compress the image into a lower-dimensional space and then train the diffusion model in this latent space.

2.4.3.1 GLIDE

The first diffusion model implemented in the field of text-to-image synthesis is GLIDE [37]. It is a standard diffusion model with 3.5 billion parameters and utilizes classifier-free guidance. GLIDE incorporates a transformer with approximately 1.2 billion parameters as the text encoder. To condition the model on textual descriptions, the prompt is first encoded into a sequence of K tokens, which is then passed through the transformer. The final output of the transformer is used as a class embedding to condition the diffusion model. It is worth noting that the output of the diffusion model is an image with a resolution of 64x64. Therefore, a text-conditional upsampling diffusion model is also trained to increase the resolution to 256x256. The upsampling model is conditioned on text in the same way as the base model.
As can be observed, the model is straightforward and simple. However, the photorealism of images generated by GLIDE is preferred in 87% of cases compared to DALL-E, and in 69% of cases, GLIDE images are considered more aligned with the captions. GLIDE can generate images with realistic shadows and reflections, utilizing different artistic styles and combining various objects, each with their respective properties. Additionally, the model also demonstrates significant generalization capabilities.

2.4.3.2 Imagen

Following GLIDE, another model that operates in the pixel space is Google's Imagen [50]. Imagen has a structure conceptually similar to GLIDE, with a diffusion model that generates 64x64 images, followed by two super-resolution diffusion models that generate 256x256 and 1024x1024 images, respectively. All of these diffusion models are conditioned on the text and use classifier-free guidance.
The substantial innovation brought by Imagen lies in the choice of the text encoder. Unlike GLIDE, which trained the text encoder together with the diffusion prior on paired image-text data, Imagen adopts a frozen large language model as the encoder. Freezing the encoder weights reduces computational overhead since the encoder no longer needs to be trained online with the diffusion prior. It also allows to pre-train the encoder on both image-text data and text-only corpora. Notably, pre-training on text-only corpora proves to be a better solution as these datasets are significantly larger than image-text datasets and expose the encoder to a broader and richer text distribution.
Imagen also solves the problem of weights used in classifier-free guidance, already encountered in previous diffusion models. It was observed that increasing the weight of guidance improved the alignment between the image and text but simultaneously led to the generation of highly saturated and unrealistic images. This issue came from the train-test mismatch that arises due to the high guidance weights: at each sampling step, the prediction $\hat{x}_0^t$ must fall within the same range as the training data (i.e. [-1,1]), but empirically it is found that high guidance weights cause the predictions to exceed these limits. Previous works used static thresholding to bring the predictions back into the desired range. In Imagen, a new method of dynamic thresholding is introduced: at each sampling step, the value of s is set to a certain percentile of the absolute pixel values in $\hat{x}_0^t$, and if $s > 1$, then $\hat{x}_0^t$ is thresholded to the range [-s,s] and divided by s. This approach pushes saturated pixels (i.e., those close to -1 and 1) inward, actively preventing pixel saturation at each step and resulting in more natural and non-divergent image generation.

When evaluated on MS-COCO, Imagen's zero-shot performance surpasses that of its predecessors by a large margin, both in terms of photorealism and alignment with the captions. Imagen demonstrates that the size of the text encoder used is critical in achieving good results and scaling the text encoder size has a much more significant impact on performance compared to scaling the dimensions of the diffusion model.

2.4.4 Latent-space Frameworks

The task of image synthesis, even in the case of diffusion models, remains a task with very high computational demands. To give an example, the most powerful diffusion models require hundreds of GPU days for training, and the inference phase is also quite expensive, with the generation of approximately 50k samples taking about 5 days on a single A100 GPU. This makes such models inaccessible to ordinary users. One of the objectives is therefore to increase the accessibility of these models by reducing their computational complexity while still maintaining their high performance. Following the approach taken with autoregressive transformers, the possibility of working in a latent space with a lower number of dimensions compared to the pixel space is being explored for diffusion models as well.

2.4.4.1 Stable Diffusion

The approach followed by Stable Diffusion (Rombach et al., 2022) starts with the analysis of diffusion models trained in the pixel space. It can be observed that for these models the learning process can be broadly divided into two stages: a perceptual compression stage, where high-frequency details are removed but small semantic variations are still learned, and a semantic compression stage, where the model learns the semantics and conceptual composition of the data. The idea is to identify a perceptually equivalent space that is computationally more suitable for training diffusion models.
In Stable Diffusion, the explicit separation of the compression phase and the actual generation phase is proposed. To achieve this, an autoencoding model that learns a space that is perceptually equivalent to the image space but has significantly reduced computational complexity is employed. This approach leads to diffusion models that are computationally much more efficient and general compression models that, once trained, can be reused for different diffusion models and tasks.

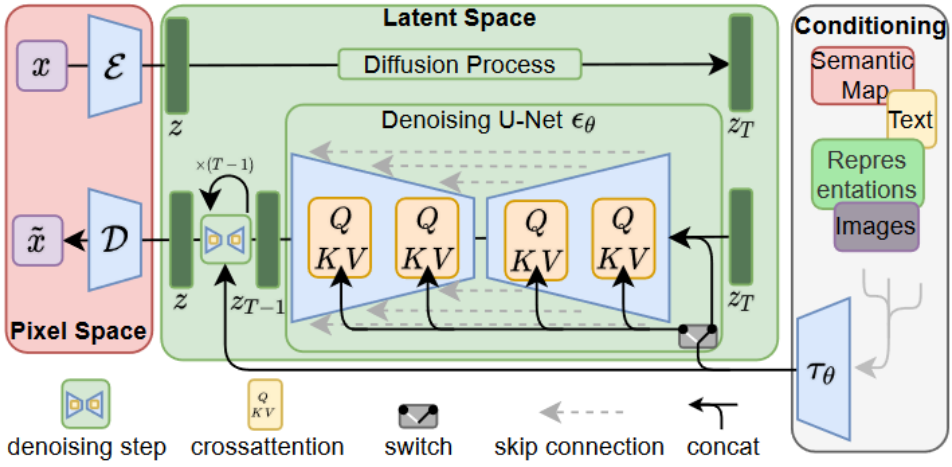

Figure 2.18: Stable Diffusion network architecture. Image taken from [48]

The perceptual compression phase is strongly inspired by the structure of VQ-GAN [14], which will be further discussed in the next paragraph. The utilization of the VQ-GAN framework enhances the realism and naturalness of the synthesized images. Once the compression model is learned, access is gained to a low-dimensional latent space where high-frequency and imperceptible details are abstracted away. By training the generative model in this space, it can focus solely on the truly important semantics. Therefore, Stable Diffusion introduces a diffusion model that, in the forward pass, works to reverse the perturbation process in this latent space. The output of the diffusion model is finally passed to the decoder, which reconstructs the image in the pixel space (Fig. 2.18). The backbone of the model is implemented as a time-conditional UNet [49]. The conditional generation of images is enabled by implementing a cross-attention mechanism [58] in the UNet, which has proven effective for learning attention-based models with various input modalities. Specifically, a domain-specific encoder, such as transformers for text, is introduced to project the conditioning variable y, which can come from different modalities, into an intermediate representation. This intermediate representation is then mapped to the intermediate layers of the UNet using cross-attention layers. This conditioning mechanism offers great flexibility.
The results obtained from Stable Diffusion demonstrate how sampling from the latent space leads to better quality compared to sampling in the pixel space. It achieves an excellent trade-off between efficiency and perceptually good results, with state-of-the-art performance on most benchmark datasets. Particularly, excellent results are obtained for conditioned generation on MS-COCO thanks to the cross-attention mechanism.

2.4.4.1.1 VQ-GAN

As described in 2.3.2, transformers were for a certain period the state of the art for language tasks, extending to text-to-image generation as well. Compared to their predecessors, transformers do not have a built-in prior and are therefore free to learn complex relationships within their inputs. However, this high generality implies that transformers need to learn all these relationships from scratch, whereas other architectures like CNNs are designed to leverage prior knowledge about strong local correlations in images. Therefore, the high expressiveness of transformers corresponds to significant computational costs, which is particularly critical when scaling to high-resolution images with millions of pixels.
Observing that transformers tend to learn convolutional structures [13], one wonders whether it is necessary to learn from scratch all the information regarding local structures and regularities in images every time we train a new model. Assuming that the low-level structure of images is well described by a convolutional architecture, the idea behind VQ-GAN is to combine such architecture with that of transformers. The convolutional approach is used to learn a codebook of context-rich visual parts, while the transformer is subsequently used to model long-range global interactions among these components. By implementing an adversarial approach that ensures that the dictionary of local parts focuses effectively on modeling fundamental local structures, the transformer can concentrate on its strengths, namely modeling long-range relationships, thus opening up the possibility of generating previously out-of-reach high-resolution images.
The learning of the context-rich codebook, through which images are compressed into a discrete representation, is achieved by incorporating the ideas of VQ-VAE (van den Oord et al., 2018) and pushing them to the limit (Fig. 2.19). The original VQ-VAE framework is modified by implementing a patch-based discriminator that aims to distinguish between real and reconstructed images based on a perceptual loss. The perceptual loss replaces the classical

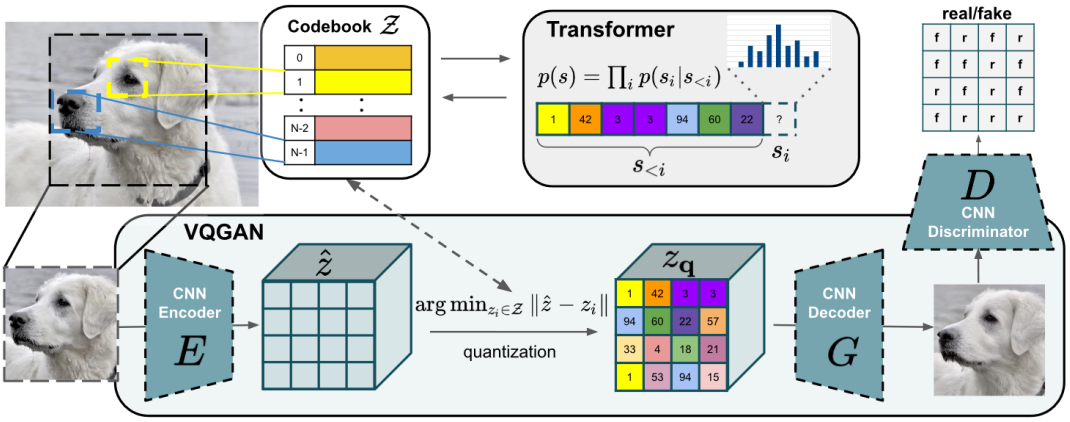

Figure 2.19: Overview of VQ-GAN architecture. Image taken from [14].

reconstruction loss of VQ-VAE, allowing for maintaining good perceptual quality at increasing compression rates. This training procedure significantly reduces the sequence length when unrolling the latent code, enabling the application of powerful transformer models.
As mentioned in the previous paragraph, the VQ-GAN approach [14] serves as the basis for Stable Diffusion in learning the latent space. In that case, the learned context-rich codebook is used to compress images into the latent space, but the generation task is handled by the diffusion model instead of the transformer used in the original implementation of VQ-GAN.

2.4.4.2 DALL-E 2

Another type of text-to-image diffusion model is based on multimodal contrastive models, where image embedding and text encoding are combined in the same representation space. One of the most well-known models of this type is DALL-E 2 (Ramesh et al., 2022), which approaches the problem of text-conditional image generation by implementing CLIP within the framework of diffusion models. The use of CLIP stems from the fact that CLIP embeddings are robust to image distribution shift, possess impressive zero-shot capabilities, and have been fine-tuned to achieve state-of-the-art results in a wide range of vision and language tasks. Before delving into the details of the DALL-E 2 architecture, it is necessary to provide an overview of the main characteristics of CLIP.

2.4.4.2.1 CLIP

CLIP (Radford et al., 2021) stands for Contrastive Language-Image Pretraining. It is a neural network introduced by OpenAI that enables learning the proper alignment between an image and its corresponding textual description. CLIP is designed to learn a joint representation of images and text, making it extremely useful for a wide range of visual and language tasks.
CLIP is trained on a dataset consisting of 400 million image-text pairs collected from various sources available on the internet. Motivated by several works in the field of contrastive representation learning that have shown that contrastive objectives can outperform equivalent predictive objectives [56], CLIP is trained to predict which text, as a whole, is paired with which image, instead of predicting the exact label (i.e. the exact words of the prompt) associated with the image. Therefore, CLIP aims to predict which of the $N \times N$ possible image-text pairs actually occur. To achieve this, CLIP learns a multimodal embedding space by jointly training an image encoder and a text encoder to maximize the cosine similarity of the correct

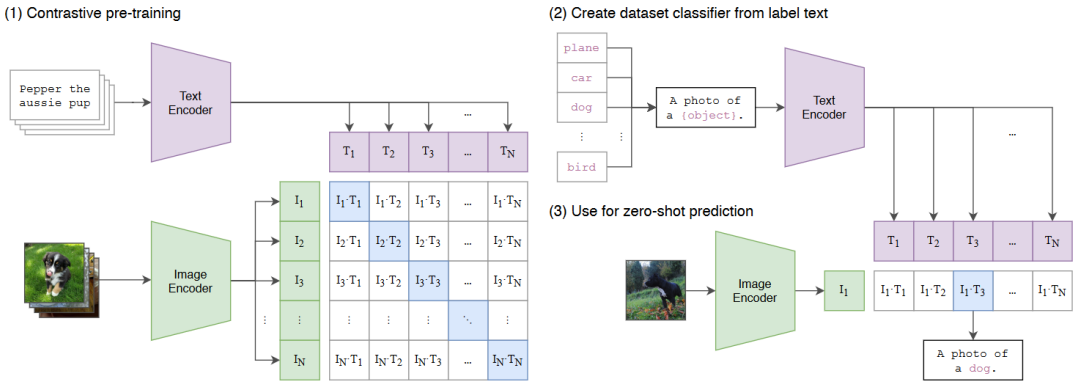

Figure 2.20: Summary of CLIP approach. Image taken from [41].

N pairs and minimize the cosine similarity of the incorrect $N^2 - N$ pairs. For image encoding, either ResNet [18] or ViT [13] are used. The text encoder, on the other hand, is a transformer that performs BPE encoding of the text.
The shared embedding space between images and text allows CLIP to be adapted to various tasks, demonstrating significant zero-shot generalization capabilities.

2.4.4.2.2 DALL-E 2 Framework

Returning to DALL-E 2, the idea is to train a diffusion decoder to invert the CLIP image encoder. To obtain a complete generative model, the CLIP image embedding decoder is combined with a prior model that generates possible CLIP image embeddings from a given caption.
Specifically, DALL-E 2 is trained on a dataset of image-caption pairs (x, y), where x represents the images and y represents their respective descriptions. The model is designed to generate images from text using two components:

- A prior, $P(z_i|y)$, which produces the CLIP image embedding, z_i, conditioned on the caption, y;
- A decoder, $P(x|z_i, y)$, which generates an image, x, conditioned on the CLIP image embedding, z_i, and optionally on the caption, y.

While the decoder allows to obtain images based on their corresponding CLIP image embeddings, the prior allows to learn a model for generating the image embeddings themselves. By combining these two components, the complete generative model, $P(x|y)$, is obtained, which generates the image, x, given the caption, y:

$$P(x|y) = P(x, z_i|y) = P(x|z_i, y)P(z_i|y)$$

Therefore, the generation process involves first sampling z_i from the prior and then sampling x using the decoder.
As the decoder, a diffusion model is used to produce CLIP image-conditional images. The architecture of GLIDE (see 2.4.3.1) serves as a reference, with modifications that involve adding CLIP embeddings to the existing timestep embeddings and projecting the same CLIP embeddings into four additional context tokens, which are concatenated with the sequence obtained from the text encoder of GLIDE. The original text conditioning path of GLIDE is retained, assuming that it allows the diffusion model to learn aspects of natural language that CLIP may not

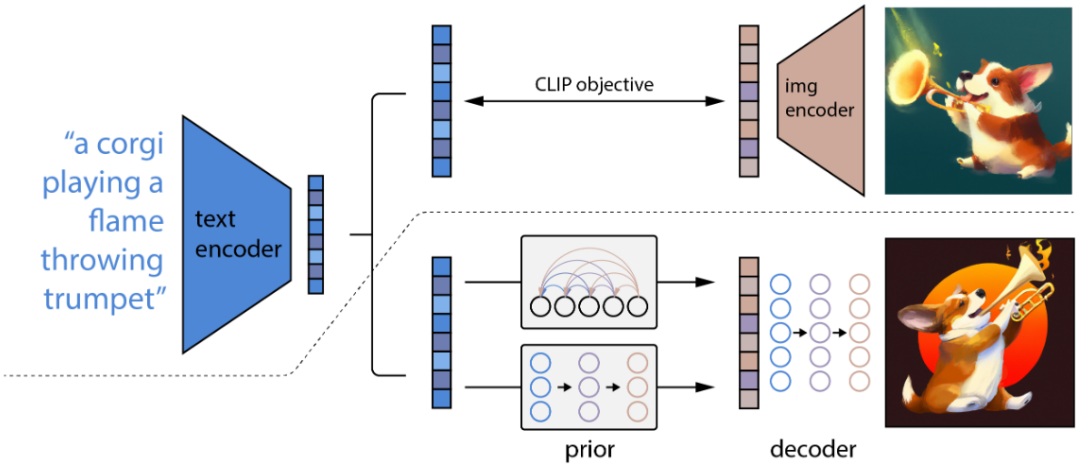

Figure 2.21: Overview of DALL-E 2 framework. Above the dotted line is the CLIP training process; below the dotted line is the text-to-image generation process. Image taken from [42].

capture. Classifier-free guidance is also enabled, and upsampler models are trained to generate images up to a resolution of 1024x1024.
The prior, on the other hand, needs to be able to generate image embeddings z_i from the caption y. Both an autoregressive model and a diffusion model are explored as potential priors. Since the results produced by both approaches are comparable but the diffusion model is computationally more efficient, the diffusion model is used as the prior in DALL-E 2.
The diffusion prior directly models z_i using a conditioned Gaussian diffusion model with the caption y. Specifically, to obtain the prior, a decoder-only transformer is trained with a causal attention mask on a sequence consisting of the encoded caption, the CLIP text embedding (which is a deterministic function of the caption), the embedding for the diffusion timestep, the noised CLIP image embedding, and a final embedding whose output from the transformer is used to predict the unnoised CLIP image embedding. Additionally, to improve the quality of sampling, two samples of z_i are generated by selecting the one with a higher dot product with the CLIP text embedding z_t.
Like the other autoregressive and diffusion models, DALL-E 2 is not directly trained on MS-COCO but is able to easily generalize to the zero-shot validation set. DALL-E 2 achieves new state-of-the-art zero-shot FID values on MS-COCO, generating realistic scenes that are aligned with the prompts and with high diversity.

2.5 Latest Works

The evolution of text-to-image models has been extremely rapid. From GANs to autoregressive models and diffusion models, the available algorithms have quickly allowed to achieve quality levels that were unimaginable just a few years ago. There are still ongoing developments, with improvements being made to the previously described models, as well as the proposal of a series of innovative works that seek to combine the strengths of the outlined techniques. In this paragraph, we want to present some of the most recent works that have opened up further possibilities for improving image synthesis from text.

2.5.1 Diffusion-GAN

GANs have demonstrated the ability to generate high-quality photorealistic images, especially when limited to specific categories. However, GANs have been shown to suffer from convergence and instability problems. Among the many solutions to regularize GAN training, one possibility is to inject noise into the input of the discriminator. This allows to expand the support of both the generator's and discriminator's distributions, thereby preventing the latter from overfitting [2]. Although very useful, this technique is challenging to apply in practice, primarily because finding a suitable noise distribution to use is complicated.

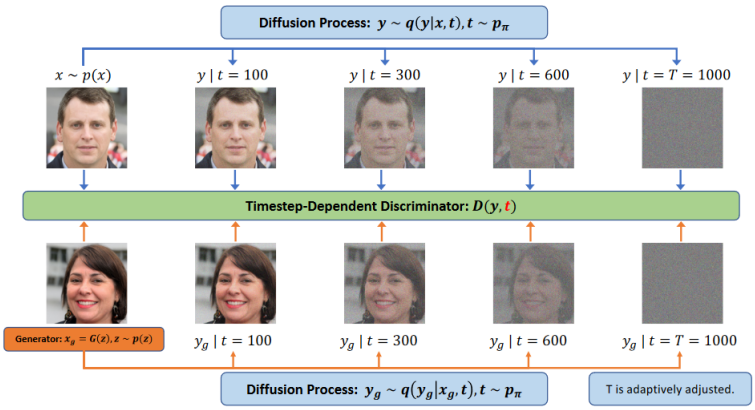

Figure 2.22: Diffusion-GAN flowchart. Image taken from [62].

One of the most recent proposals is the Diffusion-GAN [62], an architecture that utilizes a diffusion process to generate Gaussian-mixture distributed instance noise. In Diffusion-GAN, the input to the diffusion process consists of a real image and a generated image, and the diffusion process simply involves gradually adding noise to the images. The number of diffusion steps is dynamically determined based on the data, and the diffusion process is designed to be differentiable, allowing gradients to propagate from the discriminator to the generator through the diffusion process. Additionally, a timestep-dependent discriminator is designed. It compares the real and fake images for different noise-to-data ratios, learning to distinguish the noisy versions of the images. Building upon the fundamental concept of GANs, in Diffusion-GAN, at each diffusion step the discriminator is trained to distinguish between generated and real images, while the generator aims to generate images that deceive the discriminator. This process has proven to be very useful as it stabilizes the training by addressing the vanishing gradient problem, and improves the efficiency and diversity of the generator. In fact, this architecture outperforms state-of-the-art GAN architectures on almost all benchmark datasets by a considerable margin. Although still distant from the performance of autoregressive models and diffusion models, the combination of GAN architecture with the concept of diffusion introduces an interesting direction for future developments.

2.5.2 GigaGAN

The current major text-to-image models, based on autoregressive and diffusion models, are straightforward to define but, at the same time, are based on an iterative inference process which is computationally quite expensive. On the other hand, GANs are inherently efficient in this regard, as they can generate images with a single inference pass. As mentioned before, GANs have achieved excellent results in modeling a limited number of classes, but scaling to

complex datasets remains a significant challenge. However, with autoregressive and diffusion models owing their success primarily to being ultra-large models trained on large-scale datasets, one wonders if GANs can also benefit from utilizing such resources or if they have reached a plateau in terms of performance.
Various experiments have shown that simple scaling of GAN architecture leads to unstable training. The main problems have been identified and addressed in GigaGAN [26], which achieves stable and scalable training of a GAN with one billion parameters on large-scale datasets.
GigaGAN introduces several substantial modifications to the traditional structure of a GAN. The underlying architecture is based on StyleGAN2 [27]. The first problem to address is scaling the generator's capacity. To handle the highly diverse distribution of large-scale datasets, a solution is proposed that selects convolutional kernels on-the-fly based on text conditioning. Instead of choosing a single convolutional filter, a bank of N filters is instantiated. The most suitable filter is dynamically selected based on the conditioning variable and used in the regular StyleGAN2 convolution pipeline.
Furthermore, since convolutional filters operate within their receptive field, attention layers are implemented to incorporate long-range relationships. Specifically, both self-attention (i.e. image-only) and cross-attention (i.e. image-text) layers are interleaved with the convolutional layers to improve performance.
Moreover, multi-scale training is reintroduced, employing a new scheme that enhances image-text alignment and low-frequency details of generated outputs. Multi-scale training allows the GAN-based generator to effectively utilize parameters in low-resolution blocks, resulting in improved image-text alignment and image quality.

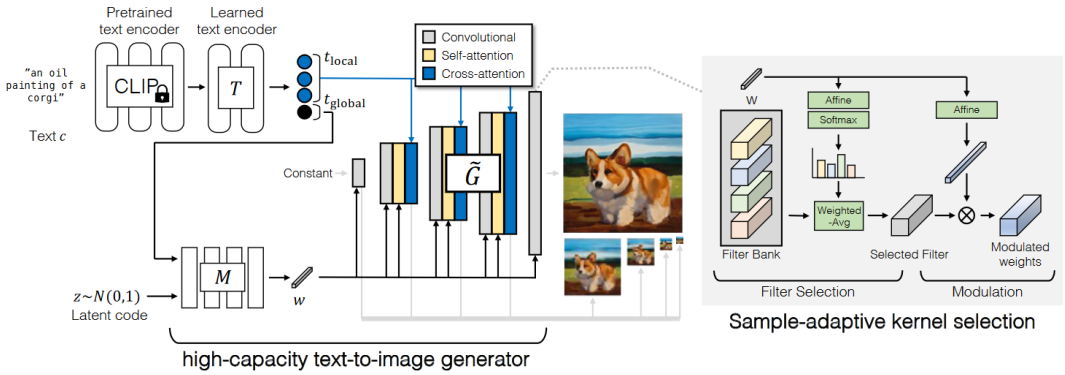

Figure 2.23: GigaGAN network architecture. Image taken from [26].

The objective function is also modified by introducing innovative CLIP-based terms. Those terms are added to the traditional adversarial loss and matching-aware loss, which address image realism and caption alignment respectively, and proved effective in improving model stability.
Lastly, it is worth noting that the GigaGAN framework follows a multi-stage approach, with the first stage generating a 64x64 image and a subsequent stage performing upsampling up to a resolution of 512x512. Both modules prove to be robust enough to be used in a plug-and-play fashion, with the upsampling network efficiently serving as an upsampler for other methods like DALL-E 2.
GigaGAN surpasses the results of previous GANs by a wide margin. With one billion parameters, it is still smaller in size compared to more recent synthesis models (Imagen with 3 billion, DALL-E 2 with 5.5 billion), yet it achieves impressive zero-shot FID results on COCO, out-

performing Imagen, DALL-E 2, and Stable Diffusion. Additionally, GigaGAN is significantly faster during the generation phase compared to autoregressive and diffusion counterparts. This work demonstrates that GANs still hold promise for image synthesis from text and should be considered for future aggressive scaling.

2.6 Metrics

For the evaluation of text-to-image model performance, it is essential to determine context-appropriate metrics. For a long time, the main metrics for evaluating the performance of text-to-image algorithms were the Inception Score (IS) and the Frechet Inception Distance (FID). Another popular approach is manual inspection through user tests. However, with the introduction of multimodal models, the CLIPScore has become particularly popular, as it is a useful metric for evaluating the alignment between an image and its description.

2.6.1 Inception Score

The Inception Score (IS) [51] is a measure of the realism of synthetically generated images. It is widely used as it correlates well with human evaluation of image quality. The IS is calculated based on the output of InceptionV3, a pre-trained image classification model, applied to a sample of generated images (e.g., approximately 30,000). The IS is defined in the range [0, N], where N is the number of possible labels. A higher IS value indicates a sharper and more distinct distribution of images. The IS is maximized when the entropy of the predicted label distribution by InceptionV3 is minimized (i.e., the model is very confident in predicting the various classes), and the predictions are evenly distributed across the classes (i.e., the generated images are sufficiently "diverse").
IS is useful for evaluating the quality and diversity of generated images. However, the main problem is that the score is limited by what InceptionV3, or any other usable classifier, is capable of classifying, which is directly linked to the data on which the classifier is pre-trained. If you attempt to generate something outside the known classes of the classifier, you will consistently obtain a low IS, regardless of the image quality. Furthermore, the classification network may not necessarily extract relevant features related to the concept of image quality. There is evidence that CNN-based models like InceptionV3 heavily rely on local image textures for classification, while coarse shapes are not particularly important. This can lead to low-quality images still receiving high scores. For example, the generation of people with two heads might not be properly penalized. Finally, the IS fails to capture intra-class diversity.

2.6.2 Frechet Inception Distance

Another metric for evaluating the quality of generated images is the Frechet Inception Distance (FID) [20]. Unlike the IS, which only assesses the distribution of generated images, FID compares this distribution to that of real images.
What FID does is compare the mean and variance of the distributions estimated by the deepest layer of InceptionV3 for both real and generated images. The considered layer is close to the output node that directly corresponds to real-world objects, enabling it to mimic human perception of similarity.
Specifically, assuming a Gaussian distribution for convenience, considering the probability distributions $N(\mu, \Sigma)$ and $N(\mu', \Sigma')$ for real and generated images, respectively, their FID is calculated as follows:

$$d_F(N(\mu, \Sigma), N(\mu', \Sigma'))^2 = ||\mu - \mu'||_2^2 + tr(\Sigma + \Sigma' - 2(\Sigma^{\frac{1}{2}} \cdot \Sigma' \cdot \Sigma^{\frac{1}{2}})^{\frac{1}{2}})$$

The goal is to have a lower FID value, indicating that the two distributions are very similar. FID has been widely adopted because of its consistency with human inspection and sensitivity to small changes in the real distribution. Furthermore, unlike IS, FID is able to evaluate intra-class diversity by detecting intra-class mode collapse. The main issues with FID are related to the Gaussian assumption in the metric calculation, which may not hold in practice, and its high bias. It has been shown that FID is statistically biased, meaning that the expected value on a finite dataset is not its true value. It is necessary to evaluate FID on a sufficiently large sample of data, typically more than 50,000 images, to avoid overestimating the score.

2.6.3 Manual Inspection

As said before, a popular approach to evaluate generative models is manual inspection via user tests. While this approach can directly investigate image quality, it is rather limited in assessing sample diversity. It is also subjective as the reviewers may incorporate their own biases and opinions. It also requires knowledge of what is realistic and what is not for the target domain which might be hard to learn in some cases (e.g. medical domain). Furthermore, it is constrained by the number of images that can be reviewed in a reasonable time prohibiting its usage during model development. Finally, due to the subtle differences in experimental protocols (e.g. user interface design, fees, duration), it is often difficult to replicate the results across different publications.

2.6.4 CLIPScore

As mentioned in chapter 2.4.4.2.1, CLIP is a cross-modal retrieval model trained on over 400 million image-text pairs collected from the internet. The idea is to leverage the multimodal space described by CLIP to quantitatively evaluate the compatibility between an image and its potential description.
Although it does not explicitly evaluate the quality or diversity of the image generated by the generative model, the CLIPScore [19] allows for an effective comparison of the content between an image and text, thereby assessing their alignment. The compatibility of an image-text pair can be thought of as the semantic similarity between the image and text. The CLIPScore is calculated simply through the cosine similarity between the visual CLIP embedding and the textual CLIP embedding. As a result, the CLIPScore is defined in the range [0, 1], with values close to 1 indicating a high correlation between the two modalities and, thus, correct alignment. The CLIPScore is strongly correlated with human judgment. Therefore, it has quickly become an excellent solution for evaluating the outputs of generative models in both text-to-image and image captioning fields.

Chapter 3

Methodology

This work makes use of five open-source text-to-image models. The goal is to exploit these models to validate the results of eight image captioning models, trying to identify the best caption, that is the one that can reconstruct an image as faithful as possible to the original. This chapter lists all the models used with their specifics, the dataset used, the evaluation metrics implemented, and the analyses performed.

3.1 Overview

The project has three phases. Initially, a literature review phase is performed to understand existing text-to-image algorithms (Chapter 2), their main features and limitations, and to select some suitable models for subsequent analysis.
After that, using each of the five selected models (Section 3.3), images are generated using the caption of the custom dataset at hand. The goal of this phase is to compare the images reconstructed using the caption with the corresponding original images, so as to identify the best captioning model. The comparison between original and generated images is done with CLIPScore.
Finally, an analysis is conducted on the use of enriched prompts. Observing that the prompts in the dataset do not capture stylistic details of the image, we want to evaluate whether adding this information to the prompts can result in images that are more similar to the originals and, therefore, whether text-to-image models benefit from using longer and more detailed prompts.

3.2 Experimental Setup

3.2.1 Computational Resources

The generation of images and the subsequent evaluation are carried out through Python scripts. All the models used for this work can be easily implemented in Google Colab, a cloud-based Jupyter notebook environment that runs in a web browser and allows anyone with internet access to experiment with machine learning and coding for artificial intelligence. However, the free version of Google Colab has limitations on daily GPU usage. Therefore, the work was conducted on a university virtual machine, which is accessed remotely via SSH.
The virtual machine features an Nvidia GeForce GTX 1070 GPU with 8 GB of RAM and implements CUDA in version 12.2. Furthermore, it does not have any limitations on GPU usage time.

3.2.2 Python Dependencies

The programming language used is Python (version 3.10.12). In addition to common usage libraries, several specific libraries were used for this project. Specifically, **Torch** (*2.0.1+cu118*) and **Torchvision** (*0.15.2*+cu118) are required for the environment setup. For the use of Stable Diffusion and DreamlikeArt, the libraries **Diffusers** (*0.19*) and **Huggingface-Hub** (*0.16.4*) are necessary. These two models also require the libraries **Accelerate** (*0.21*), **Transformers** (*4.25.1*), and **SciPy** (*1.8.0*) for code optimization. Kandinsky[1], GLIDE[2] and CLIP[3] are implemented through their respective GitHub repositories. Finally, DALLE-Mini is implemented using the library **dalle-mini** (*0.1.4*). This model also requires the libraries **Jax** and **Jaxlib** in the specific version *0.3.25*.

3.3 Text-to-Image Model Selection

The models selected for this work are five of the most popular and high-performing open-source text-to-image models. None of these models have usage limits or require a paid subscription. Below is a description of each model:

- **Stable Diffusion 2.1**: it is the largest open-source text-to-image model available. It is the latest and most up-to-date version of Stable Diffusion and is easily accessible through HuggingFace[4]. It is trained on the LAION-5B dataset, which consists of 5,85 billion image-text pairs. It allows for 768×768 image generation with the available GPU;
- **DreamlikeArt Photoreal**: it is a model derived from Stable Diffusion 1.5, a Stable Diffusion version trained on the LAION-2B dataset. Dreamlike Photoreal has undergone an extensive fine-tuning process, leveraging the power of a dataset consisting of photorealistic images generated by other AI models or user-contributed data, made by dreamlike.art[5]. It allows for 768×768 image generation with the available GPU;
- **Kandinsky 2.1**: it is a Russian text-to-image model that inherits best practices from Dall-E 2 and Latent Diffusion while introducing some new ideas. The model is trained on the LAION HighRes dataset, a subset of LAION-5B consisting of approximately 170 million images with a resolution of 1024 x 1024 or higher, and subsequently fine-tuned on the creators' internal datasets. More information about the model can be found on the GitHub page. However, the available GPU resources allow for generating images with a resolution not exceeding 256×256;
- **GLIDE**: the small, filtered-data version of the model described in section 2.4.3.1. It is trained on a dataset where several filters were applied in order to remove all images of people, violent objects, and hate symbols. Allows generation of images up to 256x256 resolution;
- **DALL-E mini**: it is a reproduction of the model described in section 2.3.4 made by the AI community. The model is significantly smaller than the original model and replaces some original components with similar pre-trained components. It's trained on three

[1] https://github.com/ai-forever/Kandinsky-2
[2] https://github.com/openai/glide-text2im
[3] https://github.com/openai/CLIP
[4] https://huggingface.co/stabilityai/stable-diffusion-2-1
[5] https://dreamlike.art/

different datasets, totaling approximately 15 million image-text pairs. More detailed information can be found on the model card[6]. Although the results are worse compared to those of the original model, it was decided to use DALLE Mini to have examples of generation from an autoregressive model as well.

The models proposed are not necessarily among the most high-performing ones available. Very popular models like Midjourney, DALLE 2, or the recent Stable Diffusion XL offer significantly better performance in terms of the quality of generated images, but they all require higher resources than what is available or the payment of a subscription. Additionally, as can be easily observed from the sizes of the datasets used for their training, it should be emphasized that the performance offered by Stable Diffusion and DreamlikeArt is significantly superior to that of the other three proposed models.

3.4 Dataset

For this experiment, a hand-crafted dataset created in a previous work is used. The dataset consists of 5000 elements and 9 attributes. Specifically, the first attribute corresponds to the image URL of the MS-COCO dataset. The other eight attributes are text descriptions referring to the image, generated through as many image captioning models.

Image_URL	OFA_Huge	BLIP_2	GIT_Large	ExpansionNet_v2	ViT_GPT2	Best	Ensemble	New_ensemble
http://images.cocodataset.org/val2014/COCO_val...	"a bathroom with a wooden shelf next to a toilet"	"a bathroom with a toilet and a sink"	"a bathroom with a toilet and a shelf with bas...	"a bathroom with a toilet and a sink"	"a bathroom with a toilet, sink, and shelf "	"a bathroom with a toilet and a shelf with bas...	"A bathroom with a toilet and a wooden shelf w...	"A bathroom with a toilet, a sink, and a woode...
http://images.cocodataset.org/val2014/COCO_val...	"a man wearing a black shirt and a green tie"	"a man wearing a tie with leaves on it"	"a man wearing glasses and a green tie."	"a man wearing a black shirt and a tie"	"a man wearing a tie and a shirt "	"a man wearing a black shirt and a green tie"	"The man was wearing a black shirt, glasses, a...	"A man wearing a black shirt, glasses, and a t...
http://images.cocodataset.org/val2014/COCO_val...	"two men sitting on a motorcycle at a traffic ...	"a black and white photo of two men on a motor...	"a man and a woman sitting on a motorcycle."	"a black and white photo of a large train larg...	"a man riding a motorcycle down a street "	"two men sitting on a motorcycle at a traffic ...	"A black and white photo of two men sitting on...	"A man and a woman sitting on a black and whit...
http://images.cocodataset.org/val2014/COCO_val...	"a row of motorcycles parked on the side of a ...	"a row of motorcycles parked on a cobblestone ...	"a row of motorcycles parked on a narrow street."	"a group of motorcycles parked in front of bui...	"a row of motorcycles parked in front of a bui...	"a row of motorcycles parked on a narrow street."	"A row of motorcycles parked on a narrow cobbl...	"A group of motorcycles parked in a row on a c...
http://images.cocodataset.org/val2014/COCO_val...	"a bedroom with a tv and a white bed"	"a living room with a television, a chair, a d...	"a bed sitting in front of a bookcase with a t...	"a bedroom with a television and a table with ...	"a living room with a tv and a bookcase "	"a living room with a television, a chair, a d...	"The image is of a living room featuring a bed...	"The image is of a bedroom with a television, ...

Figure 3.1: The hand-built dataset used.

The first five models, OFA_Huge [61], BLIP_2 [32], GIT_Large [60], ExpansionNet_v2 [25], and ViT_GPT2 [24], are captioning models that excel in terms of performance, reproducibility, scalability, and popularity. The last three models, Best, Ensemble, and New_ensemble, represent the results achieved in [5]. Specifically, Best corresponds to the caption with the highest BLIPScore, a metric that computes the overall matching score between the image embedding and the caption embedding [5], among those generated by the previous five models. Therefore, it will always be a copy of one of the other models' descriptions. Instead, Ensemble and New_ensemble represent two possible descriptions obtained by merging, for each image, the two captions with the highest BLIPScore among those generated by the other models. The merging is done using GPT_3, one of the largest and most efficient LLMs available [7].
Given the limited computational resources available, only a subset of 100 images was actually used in this project. In the future, with access to greater computational resources, this work can be easily extended to the entire dataset to obtain more consistent results.

[6]https://wandb.ai/dalle-mini/dalle-mini/reports/DALL-E-Mini-Explained-with-Demo–Vmlldzo4NjIxODA

3.5 Image Generation and Evaluation

3.5.1 Generation with Original Prompts

As mentioned earlier, the first analysis aims to identify the best image captioning model among the eight provided in the available dataset. To accomplish this, the five selected text-to-image models are used. For each model, 10 images are generated for each caption. With eight possible descriptions, this means 80 images are generated for each original image of the MS-COCO dataset (Figure 3.2). In Table 3.1 are some details about the generation time for each model.

Figure 3.2: The image generation process.

Table 3.1: Generation time for different text-to-image models

Model	Time per image (s)	Time per captioning model [10 images] (min)	Total time [8000 images] (GPU days)
Stable Diffusion	77	12.83	7.13
DreamlikeArt	100	16.67	9.26
Kandinsky	20	3.33	1.85
GLIDE	20	3.33	1.85
DALLE Mini	6	1	0.55

3.5.2 Generation with Enriched Prompts

The second analysis focuses on the use of more detailed prompts. The descriptions in the dataset are often brief and concise, and they rarely provide stylistic details regarding the photographic style, the color of objects, or the arrangement of the scene. Even the Ensemble and New_ensemble models, whose prompts were obtained by merging two descriptions precisely with the aim of testing the effectiveness of more detailed prompts, do not include such characteristics. Therefore, the goal is to test whether the use of longer and more detailed prompts,

constructed by adding stylistic information, can somehow improve the alignment between the original images and the reconstructed ones and impact the CLIPScore.
Given the very limited size of the dataset used in this study, the "enriched" prompts were manually written, aiming to describe the original images in the most accurate way possible.

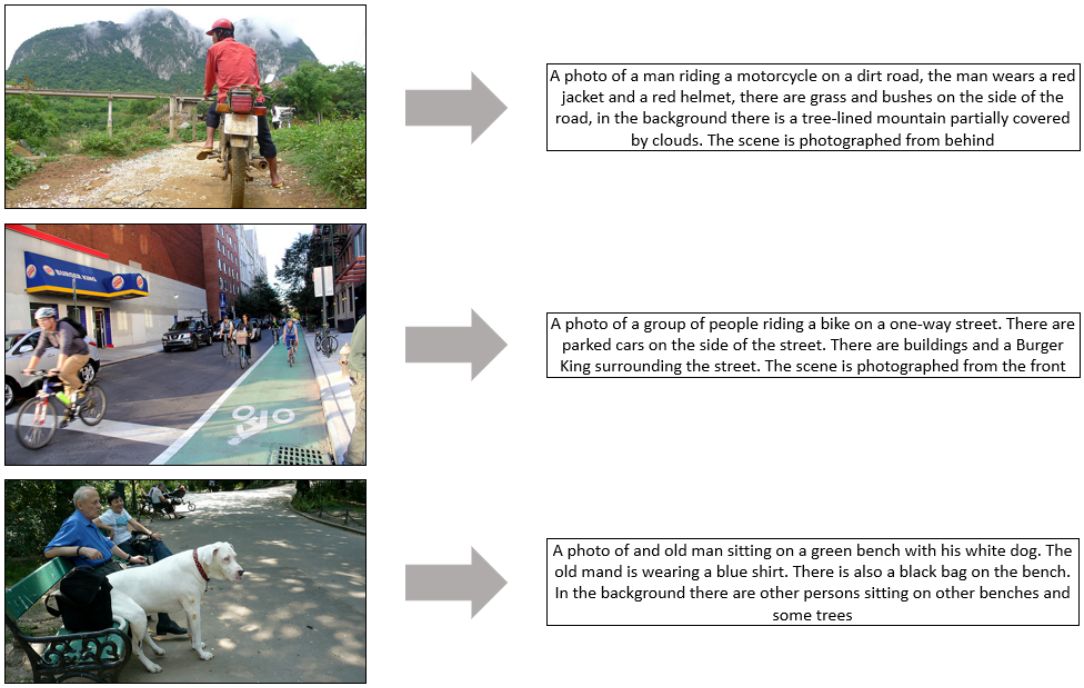

Figure 3.3: Examples of hand-written detailed prompts.

Of course, in the perspective of future expansion of this work, manual writing is not a sustainable process. In this regard, in section 4.3, a way to make this process more easily scalable is proposed through the use of tools like ChatGPT as a prompt generator.

3.5.3 Image Similarity Evaluation

The comparison between the generated images and their corresponding original images is based on the calculation of the CLIPScore. As mentioned in 2.6.4, the CLIPScore is designed to assess the alignment between text and image through the cosine similarity of their respective CLIP embeddings. Therefore, by calculating the cosine similarity between two image embeddings, it is possible to evaluate the alignment between two images.
The model used is the one made available by OpenAI[7]. Firstly, the two images to be compared are processed to extract their respective CLIP embeddings. These embeddings are then used to calculate the cosine similarity. Having generated 10 images for each prompt, the CLIPScore is first computed for each synthetic image with respect to the original, and then an average CLIPScore is calculated for each captioning model.
The calculation of the CLIPScore for the selected subset of 100 images requires approximately 14 hours of GPU time per text-to-image model.

[7]https://github.com/openai/CLIP

Chapter 4

Results and Discussion

4.1 Image Similarity Metrics

As mentioned in section 3.5.3, the chosen metric for evaluating the similarity between real and synthetic images is the CLIPScore. With the strengths and weaknesses of this metric to be analyzed in detail in the upcoming paragraphs, it is necessary to emphasize that it was not the only metric considered.
Firstly, the use of common metrics in the text-to-image domain was considered, such as Inception Score or FID. However, as described in 2.6, IS is not suitable for the purpose of this experiment since it focuses solely on evaluating the quality and diversity of images generated through a classification model, without capturing the similarity between real and synthetic images. On the other hand, FID aims to compare real and synthetic images by comparing their respective distributions and therefore represents a possible solution. However, FID requires a sufficiently large set of images (typically more than 50,000 images) to avoid overestimating the score, making it impossible to draw consistent conclusions given the size of the subset used. In the future, by expanding this work to a larger dataset, it will be possible to calculate the FID for the eight captioning models more consistently, to assess its potential effectiveness as a metric for this task.

Another possibility is represented by the use of traditional image similarity metrics. Below are the metrics tested, along with a brief description:

- **Root Mean Square Error (RMSE)**: measures the amount of change per pixel between two images. RMSE values are non-negative, and a value of 0 means the images being compared are identical;
- **Peak Signal-to-Noise Ratio (PSNR)**: measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. PSNR is usually expressed in terms of the logarithmic decibel scale;
- **Structural Similar Index Measure (SSIM)** [64]: quantifies image quality degradation caused by processing, such as data compression, or by losses in data transmission. SSIM is based on visible structures in the image. In other words, SSIM actually measures the perceptual difference between two similar images. The algorithm does not judge which of the two is better;
- **Feature Similarity Indexing Method (FSIM)** [70]: was developed with a view to comparing the structural and feature similarity measures between restored and original objects. It is based on phase congruency and gradient magnitude;

- **Information theoretic-based Statistic Similarity Measure (ISSM)** [1]: a hybrid approach that incorporates information theory (i.e. Shannon entropy) with a statistic (i.e. SSIM) as well as a distinct structural feature provided by edge detection;
- **Spectral Angle Mapper (SAM)** [67]: a physically-based spectral classification. The algorithm determines the spectral similarity between two spectra by calculating the angle between the spectra and treating them as vectors in a space with dimensionality equal to the number of bands. Smaller angles represent closer matches to the reference spectrum;
- **Universal Image Quality Index (UIQ)** [63]: is designed by modeling any image distortion as a combination of three factors: loss of correlation, luminance distortion, and contrast distortion.

These metrics are implemented using the Python package *image-similarity-measures*[1].
The use of these metrics immediately raised two problems, closely related to their definition. All the metrics described above are designed to quantify the effect of image processing techniques. They are used to compare an image with its edited copy or with an image of the same subject taken under different environmental conditions (i.e., different angles, different lighting conditions, etc.). They do this by comparing the two images through functions of the image's pixels, either by considering the simple difference between pixels (RMSE and PSNR) or by using more complex functions (i.e., SSIM uses the local mean and standard deviation of the pixels of the two images, as well as the covariance between the two images; FSIM calculates Phase Congruency and Magnitude Gradient; SAM compares image spectra). In the present case, these metrics would be applied to images that do not meet these requirements: images generated by the implemented models, even when they reproduce the desired subject and conditions, may differ from the original in scene composition, style, colors, shadows, and other features. Furthermore, the implementation of such metrics requires that the two images to be compared have exactly the same dimensions. This poses a problem for two reasons: firstly, the generative models used are designed to produce square images, and when tasked with generating rectangular images or images with dimensions that are not multiples of 8, they tend to produce artifacts or repetitions that degrade the overall image quality. Secondly, the original images in the MS-COCO dataset have variable dimensions, which makes automating the generation process complicated as a constant size cannot be enforced. For all these reasons, the use of these metrics has been avoided.

4.2 Generation with Original Prompts

4.2.1 Qualitative Results

The images in the MS-COCO dataset belong to various categories: photos of people, animals, simple objects, rooms in a house, as well as complex scenes that include multiple objects and specific relationships.
In this initial phase, considering the images generated by the five implemented models, a qualitative analysis is conducted regarding the quality of the images and their alignment with the corresponding original images. The objective is to identify which entities and relationships the generative models can accurately represent and which ones pose greater challenges.

[1] https://pypi.org/project/image-similarity-measures/

Figure 4.1: Examples of generated images depicting the interior of a room. The corresponding prompts are given in Appendix 2.

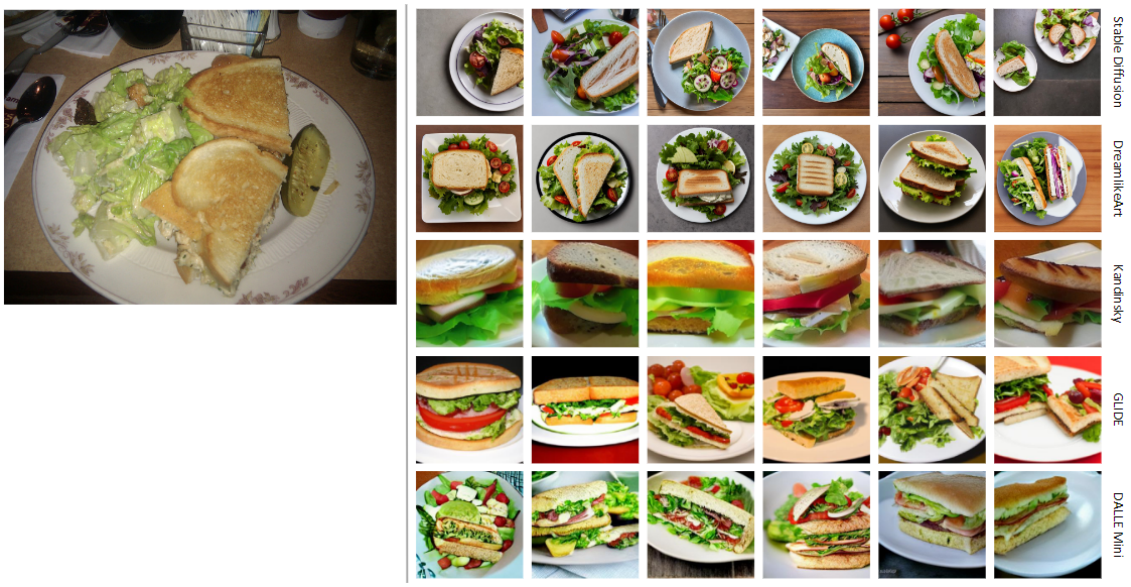

Figure 4.2: Examples of generated images representing a simple object. The corresponding prompts are given in Appendix 2.

The first category analyzed includes images of single objects or simple interior scenes, such as those shown in Figures 4.1 and 4.2. Focusing on image realism, for subjects of this type, all five models perform quite well. The depicted objects exhibit colors, shapes, and proportions that closely resemble reality and are appropriately positioned in space. Nevertheless, variations exist among the different models. The two most robust models in our evaluation, Stable Diffusion and DreamlikeArt, can produce images at a resolution of 768×768, resulting in highly detailed and sharp images. Kandinsky, while sharing a similar architecture with the previous two models, is constrained by available computational resources and can only generate images at a resolution of 256×256, leading to less detailed and sharp visuals. Nonetheless, these images still effectively convey the primary subject of the scenes. Even GLIDE and DALLE Mini,

the two models with more limited capabilities in our set, demonstrate the ability to generate coherent and well-structured scenes within this image category, albeit with a reduced level of definition and sharpness.
Yet, the most evident distinctions arise when contrasting the synthetic images with their matching originals. The generated images fail to capture the same style, lighting, or perspective as the original photo. This could be because the prompts produced by captioning models, which guide image generation, tend to be quite straightforward, focusing solely on describing the primary scene subject and disregarding stylistic elements. In fact, looking at the prompts corresponding to the two examples provided (Appendix 2), it is noticeable that there is no reference to the fact that both images have poor lighting or to the perspective from which the photos were taken. This aspect of the prompts inspired the analysis described in 4.3, where we used more detailed prompts to see if adding stylistic details to the descriptions used in image generation could make the results resemble the originals more closely.

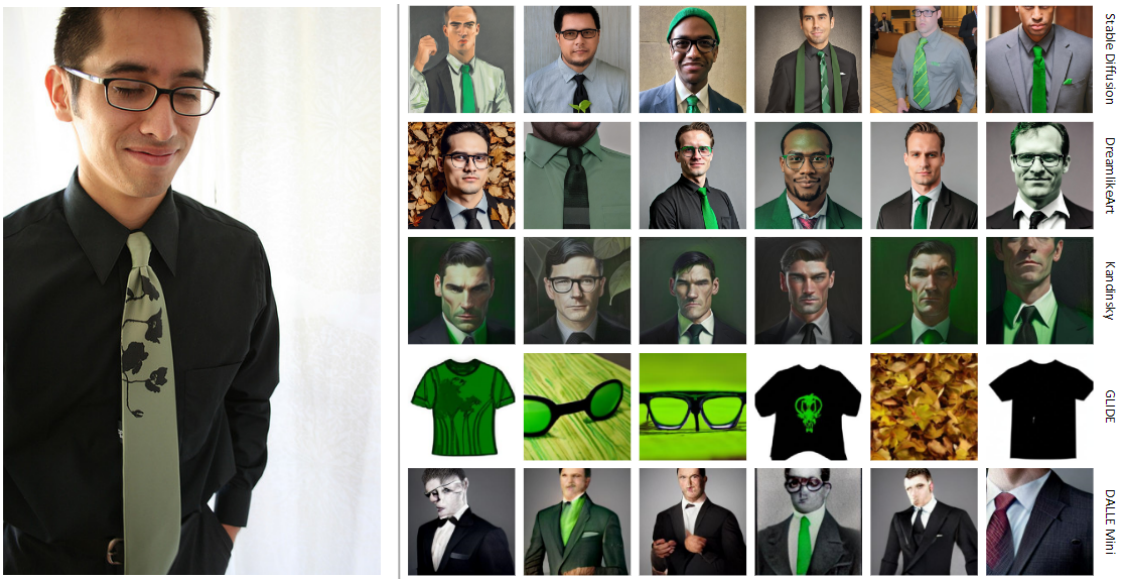

Figure 4.3: Examples of generated images representing a person. The corresponding prompts are given in Appendix 2.

One of the most challenging subjects for generative models has always been humans, especially their faces. In this study, we observed that, at least for the two best-performing models, when tasked with creating images of individual people, the results are fairly satisfactory. Stable Diffusion, Kandinsky, and especially DreamlikeArt, a model designed explicitly for realistic images, can generate convincing, well-proportioned faces without significant issues (see Figure 4.3). Other items like glasses are correctly placed and easily distinguishable from the person. However, there are some issues with the colors and focus in some of the generated images. For instance, when the captions specify the green color of the tie, it can sometimes result in a greenish tint across the entire image (e.g., in the fourth and sixth images from DreamlikeArt and all images from Kandinsky). On the other hand, in some cases, the model may focus solely on the shirt and tie, neglecting to generate the face. With these types of subjects, the difference in performance between the first three and the last two models becomes noticeable. DALLE Mini manages to capture the primary subject and occasionally gets some features like colors right, but the generated images are distorted and lack detail. In contrast, GLIDE faces more limitations due to its training on a restricted dataset that excludes images of people. The model's smaller size and limited understanding of the world hinder its ability to depict object

characteristics and perform positional tasks, especially when people are involved. As we'll see later, this pattern repeats. Nevertheless, it's important to highlight that, despite its limitations in composing scenes, the model can still capture other described elements in the prompts, such as the shirt, glasses, colors, and even tie patterns.

Figure 4.4: Examples of generated images representing an animal. The corresponding prompts are given in Appendix 2.

Very similar results are obtained for images belonging to the category of animals (Figure 4.4). Shapes and proportions are generally correct, especially for the first three models. In this case, even GLIDE generates fairly realistic images, with results comparable to those of DALLE Mini: both models, in fact, produce overall satisfactory shapes, but there are often noticeable artifacts that distort the main subject of the scene.

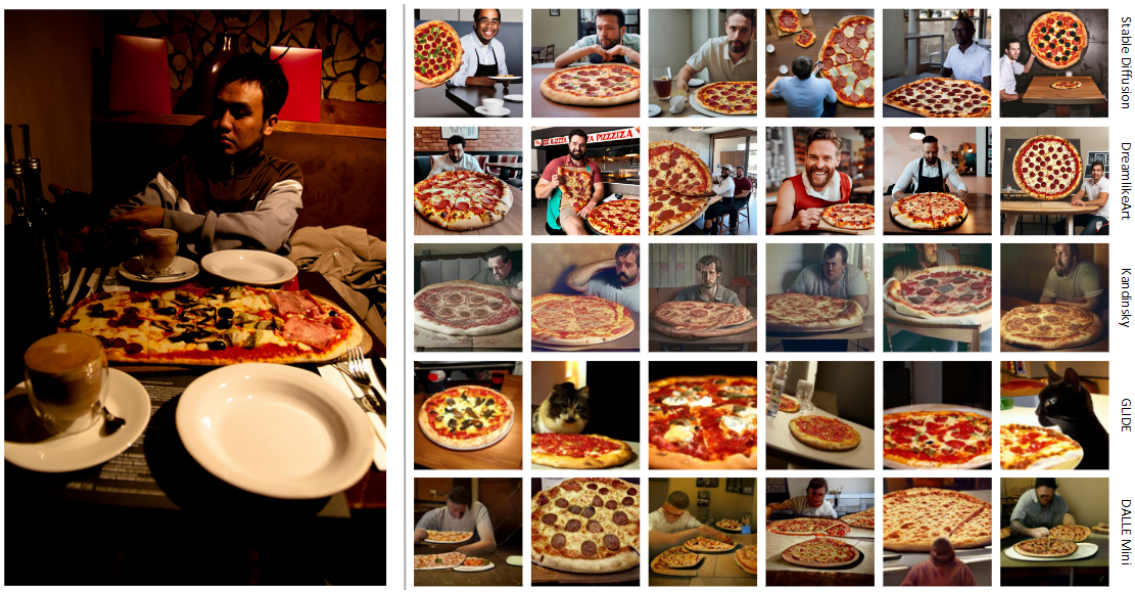

Figure 4.5: Examples of generated images representing a complex scene of a man, sitting at a table eating pizza. The corresponding prompts are given in Appendix 2.

In terms of quality, there is a significant decline when dealing with complex scenes, as exemplified in Figure 4.5 and 4.6, which involve the generation of multiple objects and the realization of specific relationships. For the first example, while the models can accurately recognize the image's subjects, the man and the pizza, the overall composition of the scene often lacks realism. Objects are frequently placed in unrealistic positions, exhibit disproportionate sizes, and, sometimes, different elements seem to blend together (for instance, in the third image from DreamlikeArt). Moreover, the quality and realism of generated faces and bodies tend to deteriorate even for models like Stable Diffusion and DreamlikeArt that previously delivered satisfactory results when generating those subjects in isolation. Among the other models, GLIDE correctly identifies one of the two subjects in the scene but demonstrates once again an incapability to generate people. Another characteristic of GLIDE is noticeable: being unable to generate humans, it often generates dogs or cats when asked to generate a person. This behavior is observed in many cases and is probably due to the fact that, in terms of features, these two entities are the closest to those of the person that the model is unable to generate.

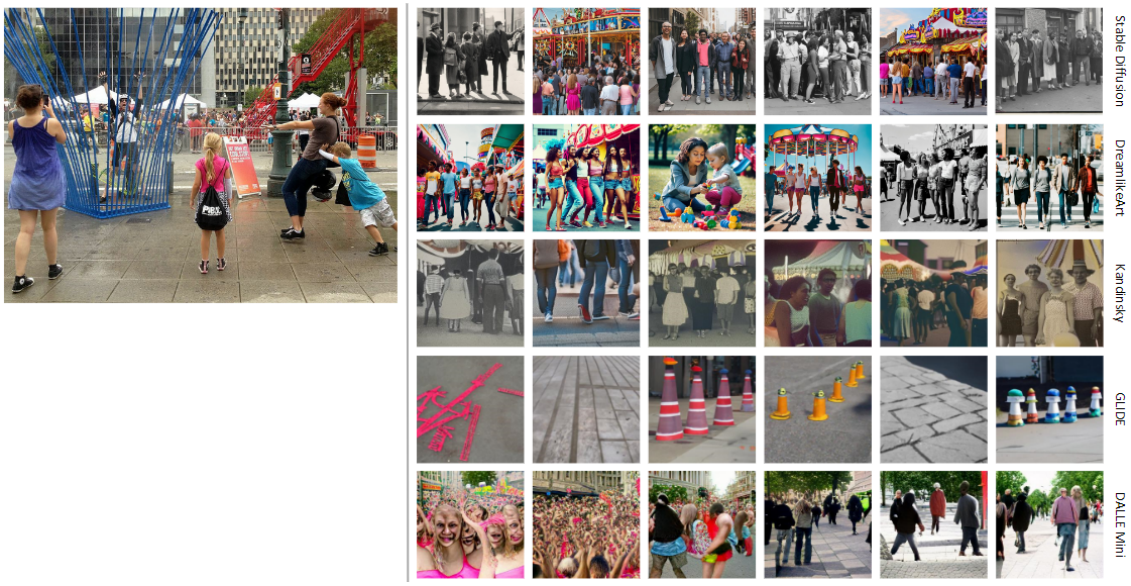

Figure 4.6: Examples of generated images representing a complex scene of a crowd of people. The corresponding prompts are given in Appendix 2.

For even more complex scenes, such as the one in the second example, the situation worsens. Stable Diffusion and DreamlikeArt maintain an acceptable level of quality, although, with a high number of subjects, the quality of individual bodies and faces deteriorates significantly. Kandinsky and DALLE Mini also generate vaguely recognizable scenes, but their content is almost entirely unrealistic. GLIDE, for extremely complex scenes, exhibits all the limitations due to its limited capacity and ends up generating artifacts that are essentially nonsensical. In general, for subjects of this kind, all the difficulties of these models for tasks deeply based on compositionality come to the surface.
Furthermore, in the case of such scenes, the deviation from the original image becomes more pronounced. The prompts employed lack the necessary level of detail and precision required to describe the actual scene's composition accurately. Consequently, when these prompts are used to generate new images, the resulting scenes exhibit significant visual disparities from the originals. A more comprehensive discussion of this topic will follow in the next section and in section 4.3.

4.2.2 CLIPScore Evaluation

After generating all the images, we conducted a methodical analysis of the CLIPScore. In order to ensure consistency, we generated 10 images for each description. For each image, we initially extracted the corresponding CLIP feature vector. Subsequently, we calculated the CLIPScore by measuring cosine similarity against the CLIP feature vector of the corresponding original image. Lastly, we computed the average CLIPScore for the 10 images generated for each of the eight captioning models. This process was executed for all five generative models.

Table 4.1: The absolute frequency of each captioning model as the model with the highest and lowest average CLIPScore for the 100 images in the dataset. For each text-to-image model, the best and worst values are highlighted in green and red, respectively.

	Stable Diffusion		DreamlikeArt		Kandinsky		GLIDE		DALLE Mini	
	Best	Worst	Best	Worst	Best	Worst	Best	Worst	Best	Worst
OFA_Huge	10	11	12	8	7	15	13	15	8	9
BLIP_2	5	17	10	12	6	13	**17**	12	11	7
GIT_Large	14	6	10	11	18	12	5	**18**	15	6
ExpansionNet_v2	16	19	11	15	11	19	12	16	13	14
ViT_GPT2	9	**33**	11	**31**	8	**25**	16	17	**17**	**31**
Best	7	6	14	6	14	8	12	9	14	5
Ensemble	**20**	5	12	8	17	1	16	7	16	13
New_ensemble	19	3	**20**	9	**19**	7	9	6	6	15

In the initial analysis, we examined how frequently each captioning model achieved the highest and lowest average CLIPScore. The results are reported in Table 4.1. Given the differences highlighted previously in terms of performance, the behaviors of the five text-to-image models are quite variable. The most significant difference is observed in the case of GLIDE, but this is consistent with the model's limited capabilities, which hinder consistent evaluations for some images, such as those where people appear.
As for the best average score, there is no clear preference for any particular model. This is because, very often, the prompts for all eight models are nearly identical to each other. Consequently, the images generated for each prompt only differ due to the stochasticity of the text-to-image models, which, even when starting from the same prompt, generate slightly different images in each iteration, leading to subtle variations in the CLIPScore, as shown in Figure 4.7.

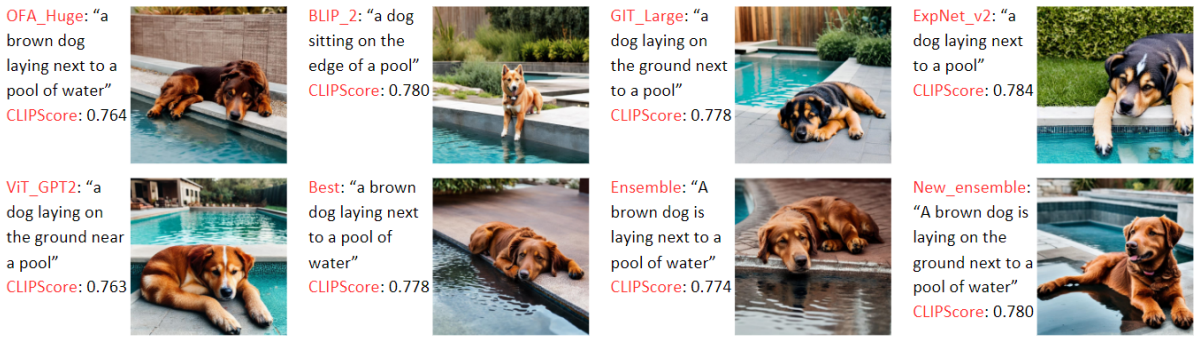

Figure 4.7: Example of images generated from very similar prompts.

However, in general, at least for Stable Diffusion, DreamlikeArt, and Kandinsky, the more consistent models appear to be **Ensemble** and **New_ensemble**. Remembering that the prompts for these two models are obtained by merging the top two prompts generated by other captioning

models, this result suggests that more comprehensive and elaborate prompts not only describe the scene more accurately but also provide more useful guidance to generative models in faithfully reproducing the original image. Instead, all text-to-image models, except for GLIDE, highlight **ViT_GPT2** as the worst captioning model in at least 25% of the cases. This result is consistent with what was obtained in [5], where ViT_GPT2 was identified as the captioning model that produced the least accurate descriptions among those tested. In any case, the consistency of the conclusions presented must necessarily be enhanced by expanding the analysis to a larger dataset.

Table 4.2: Average CLIPScore per model for the images in 4.8. For each model, the best and worst scores are indicated, respectively, in bold and underlined.

Model	OFA_Huge	BLIP_2	GIT_Large	ExpansionNet_v2	ViT_GPT2	Best	Ensemble	New_ensemble
Stable Diffusion	0.9248	0.9206	0.9034	0.9293	<u>0.8857</u>	0.9405	**0.9519**	0.9399
DreamlikeArt	**0.9521**	0.9388	<u>0.9069</u>	0.9453	0.9223	0.9453	0.9440	0.9304
Kandinsky	0.8249	0.8821	0.8739	<u>0.7569</u>	0.8181	0.8819	**0.8840**	0.8713
GLIDE	0.9074	**0.9289**	0.8921	<u>0.8751</u>	0.8839	0.8912	0.9034	0.8844
DALLE	0.8803	0.8849	<u>0.8347</u>	0.8824	0.8660	0.8779	0.8882	**0.8959**

Figure 4.8: Best average CLIPScore images per model. The corresponding prompts are given in Appendix 2.

Figures 4.8 and 4.9 show the examples referring to the best and worst results in terms of average CLIPScore for each text-to-image model.
As you can see in figure 4.8, images that achieve the best CLIPScore depict very simple subjects, such as a traffic light, a distant flock of birds, and a zebra. For all the images shown the CLIPScore is very high, close to or above 0.9 (Table 4.2), indicating a very good alignment between the feature vectors of the original image and those of the generated images.
As previously shown, within each group, there is limited variability in CLIPScore values since the eight captions are all very similar to each other. However, when considering the traffic light image, it can be seen that the models that have the lowest average score are **GIT_Large** and **ViT_GPT2**: this happens because the prompts for these two models are the only ones

that don't mention the presence of the sky in the background (Appendix 2), which is missing in the corresponding generated images. Similarly, when looking at the image of the flock of birds, **OFA_Huge**, **ExpansionNet_v2**, and **ViT_GPT2** do not specify in the description that the image should be in black and white (see Appendix 2). As a result, their generated images are in color and receive the lowest average scores. This demonstrates how CLIPScore is a useful metric for this type of analysis, as it can detect even subtle differences like the ones we've seen in these examples.

Table 4.3: Average CLIPScore per model for the images in 4.9. For each model, the best and worst scores are indicated, respectively, in bold and underlined.

Model	OFA_Huge	BLIP_2	GIT_Large	ExpansionNet_v2	ViT_GPT2	Best	Ensemble	New_ensemble
Stable Diffusion	0.4476	<u>0.3906</u>	0.4318	0.4010	0.4185	0.4290	**0.5452**	0.4771
DreamlikeArt	0.4382	0.4264	<u>0.4151</u>	0.4238	0.4663	0.4369	0.4578	**0.4781**
Kandinsky	0.4602	0.4700	0.4727	<u>0.4396</u>	0.4676	0.4545	**0.4954**	0.4656
GLIDE	0.3768	0.4044	0.3646	<u>0.3509</u>	0.3687	**0.4172**	0.3690	0.3745
DALLE	0.5224	0.5342	0.4789	0.4946	<u>0.4773</u>	0.5129	**0.5449**	0.5282

Figure 4.9: Worst average CLIPScore images per model. The corresponding prompts are given in Appendix 2.

This observation is supported by examining the values corresponding to the images shown in Figure 4.9. The generated images are qualitatively very different from the originals, and accordingly, the CLIPScore is significantly lower compared to what was obtained for the images in the previous example. Furthermore, in this case, it also highlights the metric's ability to effectively evaluate the content of the scene. Indeed, GLIDE, not being able to generate human figures, only places the outline elements in the scene, such as the ball or the brick wall in the background, creating an image where the central entity is missing. This absence results in obtaining extremely low CLIPScore values even compared to those of the other images shown in Figure 4.9.
These poor results confirm the models' difficulty in generating complex scenes, but also the fact that the prompts used do not describe in sufficient detail and accuracy the composition of these scenes.

Another crucial observation is that CLIPScore lacks a correlation with visual image quality. Since it relies on comparing feature vectors, it can effectively evaluate content similarity but falls short in assessing the realism and overall quality of the image.

Figure 4.10: Examples of non-correlation of CLIPScore with image quality. The best value achieved by the captioning models is reported for each text-to-image model.

This is evident by looking at the examples given in Figure 4.10. The CLIPScores for the various models are all quite similar, an indication that the feature vectors of the various images are not far apart, but it is evident that in terms of detail and sharpness the images of Stable Diffusion and DreamlikeArt are vastly superior to those of the other models. Thus, since the CLIPScore is not suitable for evaluating overall image quality, in order to compare the images both in terms of content and quality, in the future it will be necessary to pair it with a metric that can effectively evaluate the composition and realism of the scene.

4.3 Generation with Enriched Prompts

In the previous section, the emphasis was placed on two key aspects: the visual quality of the generated images and their conformity to the original images. While enhancing the quality of synthetic images is mainly driven by the adoption of more advanced models and the utilization of increased computational resources, the alignment issue provides greater opportunities for experimentation.
When comparing real and synthetic images, the most notable differences arise in terms of stylistic details: black-and-white photos reproduced in color, close-ups becoming full-length photos, objects reproduced in a different color or material, and different perspectives of the photo. Moreover, there persists the challenge of effectively generating intricate scenes full of numerous entities, relationships, and details, a task that generative models often struggle with in terms of quality. Notably, these characteristics have been rarely addressed in the prompts employed thus far (Appendix 2). However, it is plausible that CLIPScore, which assesses image alignment based on feature vectors, could exhibit improved alignment performance by faithfully reproducing these stylistic features. This hypothesis is further supported by the observation that among the captioning models tested so far, the most satisfactory results were achieved by Ensemble and New_ensemble, whose prompts are given by the fusion of multiple descriptions and are thus the longest and most articulate.
The goal of this section, therefore, is to assess how much the use of "enriched" prompts affects the alignment of the generated images with the originals. As mentioned in section 3.5.2, due

to the limited size of the dataset, the prompts used in this part were written manually trying to describe the scene as accurately as possible. In the description, all the essential elements of the image are specified, including the composition of the scene in terms of relationships between entities, colors and materials that characterize different objects, as well as stylistic features such as perspective, lighting, and atmosphere.
As previously done, multiple images were generated for each given prompt. The CLIPScore was computed for each of these generated images in relation to the original image. Subsequently, the average CLIPScore was determined. This average CLIPScore value was then compared to the values obtained from the eight captioning models in two steps: first, to provide an initial assessment of the new prompt's efficacy, it was compared to the overall average score of the eight models. Following this, as appropriate, further comparisons were made to assess whether it was higher or lower than the scores of all eight models. The results are shown in Figure 4.11:

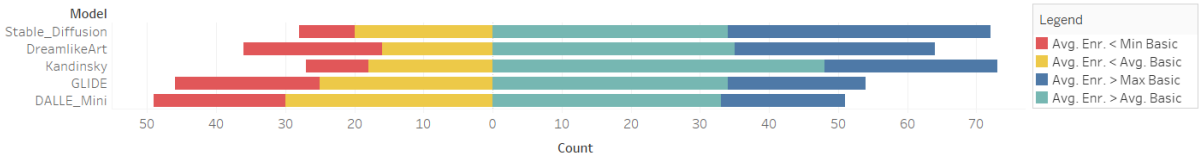

Figure 4.11: Comparison of CLIPScore with enriched prompts and basic prompts.

In the case of Stable Diffusion, DreamlikeArt, and Kandinsky, the use of enriched prompts resulted in higher scores than the average of the eight captioning models in roughly 70% of instances. This suggests an overall positive impact of employing detailed prompts. Moreover, in about 25% of cases, these scores surpassed those of all eight models. However, there were also instances where the average scores obtained with enriched prompts were lower than those of all eight models. In contrast, for GLIDE and DALLE Mini, the scores improved in approximately half of the cases and declined in the other half. This outcome is not surprising and is primarily linked to the capabilities of these two models. For relatively simple scenes, where the original prompts were already sufficiently accurate and did not require extensive modifications, GLIDE and DALLE Mini had previously demonstrated good performance. On the other hand, due to their limited size, these two models had previously exhibited reduced effectiveness in handling complex scenes, even with the initial prompts. Consequently, further increasing the length and complexity of these descriptions did not benefit these models; in fact, it is possible that it stressed their difficulties.

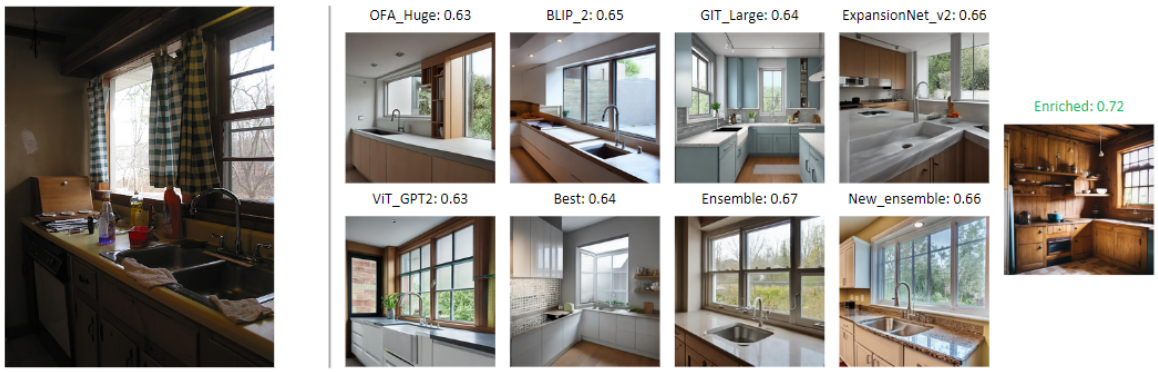

Figure 4.12: Generation with enriched prompt - Example 1.

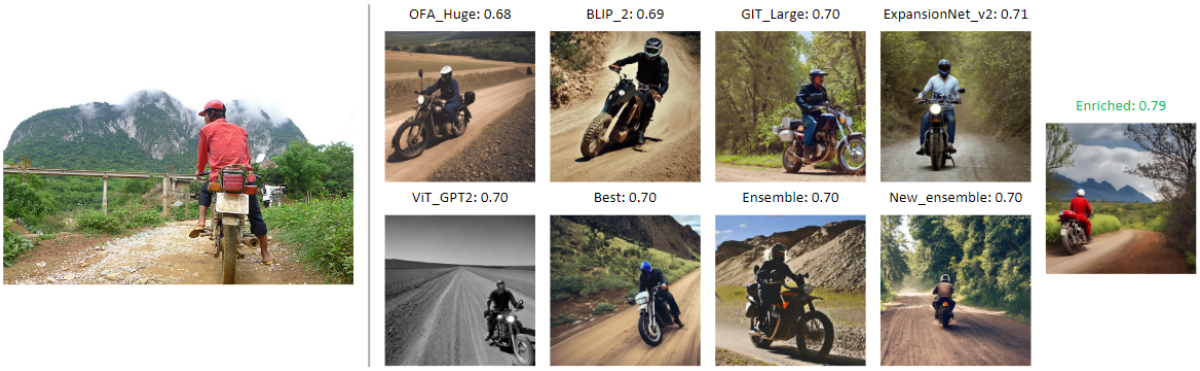

Figure 4.13: Generation with enriched prompt - Example 2.

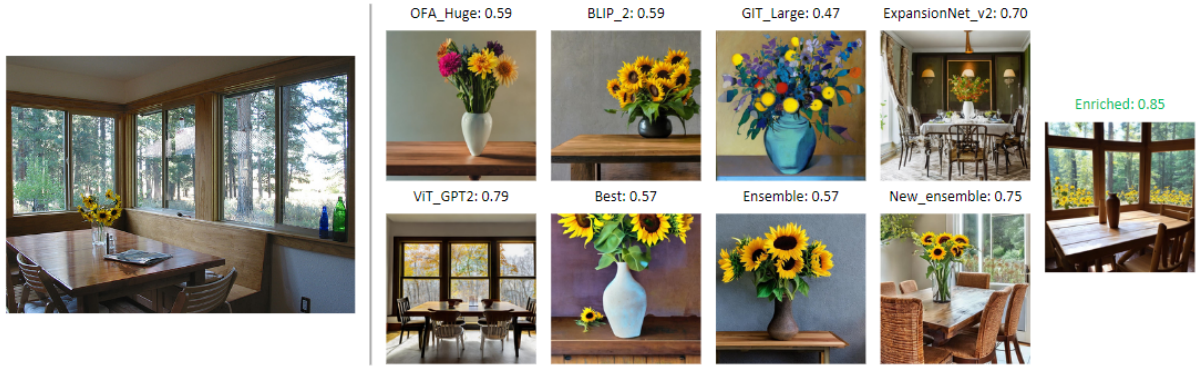

Figure 4.14: Generation with enriched prompt - Example 3.

Figure 4.15: Generation with enriched prompt - Example 4.

The figures 4.12, 4.13, 4.14, and 4.15 show some examples in which the use of enriched prompts has led to a significant improvement in CLIPScore. The examples are taken from Stable Diffusion, DrealikeArt, and Kandinsky. From a visual standpoint, one can immediately observe a significantly closer resemblance to the original image. This is further validated by a higher CLIPScore. When examining these images, it becomes apparent that the enhancements made to the prompts that have the greatest impact on improving alignment are those related to the colors and materials associated with the objects within the scene. As demonstrated in the examples presented in Figures 4.12 and 4.13, text-to-image models adeptly capture these features, resulting, respectively, in the creation of a wooden kitchen and a red jacket and helmet, aligning with their respective enriched prompts. This aspect is also evident in Figure 4.14,

where the specification of wooden table and chairs has a noticeable effect. In this instance, however, it is evident that the terminology chosen in the description plays a pivotal role. As detailed in Appendix 2, the enriched prompt explicitly states that the photo depicts a cabin in a forest, not a dining room as described in the best prompts generated by the eight captioning models. This disparity significantly contributes to reproducing an environment that closely mirrors the original. Likewise, as depicted in Figure 4.15, another aspect that the models can fairly accurately reproduce is the overall atmosphere of the scene. In this case, the portrayal of a snowy environment results in an improved CLIPScore.
Additionally, it is important to emphasize that these models frequently exhibit the capability to recreate background elements when specified in the description. In Figure 4.13, trees and bushes are generated along the road, alongside a mountain veiled by distant clouds. Figure 4.14 displays trees surrounding the cabin, while Figure 4.15 reveals bare trees and an abandoned farmhouse in the background. These elements are marginal within the scene and may have minimal impact on the alignment of feature vectors and the CLIPScore. Nevertheless, they qualitatively contribute to creating a more satisfying image.

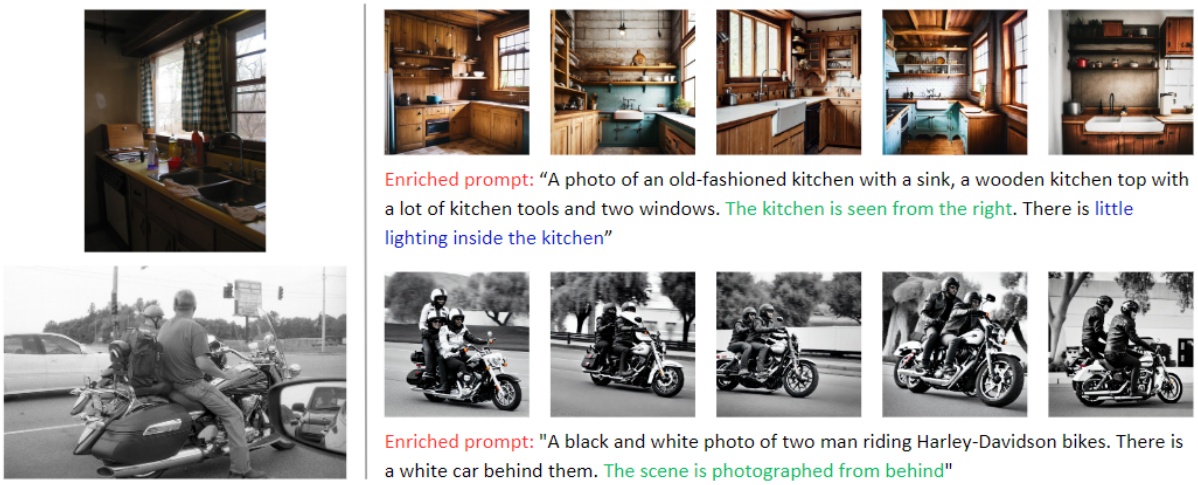

Figure 4.16: Examples for the perspective and lighting of the scene. The corresponding information in the prompt is highlighted, respectively, in green and blue.

Nonetheless, there are several features incorporated into enriched prompts that do not produce the intended outcomes, and in certain instances, they even result in a decline in the CLIPScore. The specification of the photo's perspective yields inconsistent results. While it may seem that this information is faithfully reproduced in the examples depicted in Figures 4.12 and 4.13, Figure 4.16 displays five images generated from the same enriched prompt, where the perspective lacks consistency and often does not align with the requested one. Another intricate aspect to convey is the scene's lighting. As evident in the initial of the two examples in Figure 4.16, the generative model struggles to accurately replicate the desired lighting conditions.
Another significant challenge involves the creation of complex scenes. The primary difficulty in handling scenes of this complexity arises from the tendency, when enriching the prompt with additional information, to place excessive emphasis on new details, thereby diverting attention away from the distinctive elements of the image. In the most challenging scenarios, this surplus of information can even lead to the model's incapability of accurately representing all the required elements. This scenario is exemplified in the instance depicted in Figure 4.17: within the detailed description, there is a mention of a bookshelf that should be partially visible in the background. However, the model proceeds to generate the entire bookshelf, making it a domi-

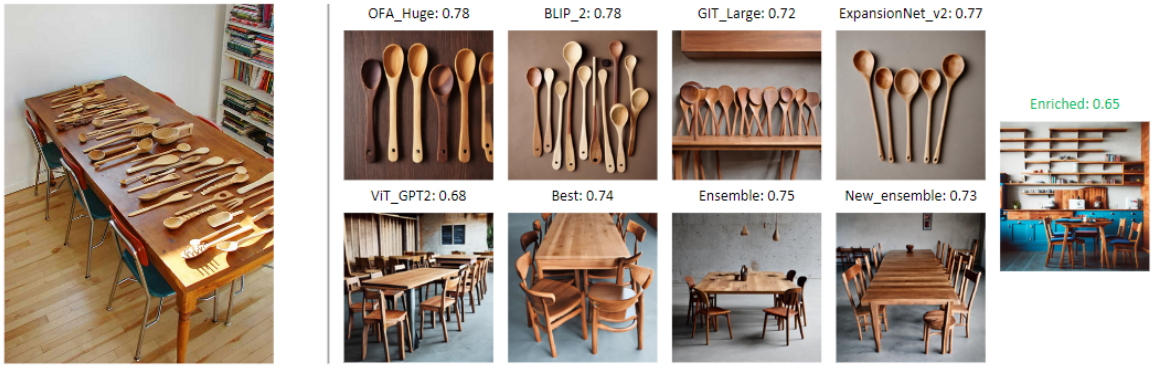

Figure 4.17: Complex scene described with an enriched prompt - Example 1.

nant feature within the image. As a result, this deviation from the original in terms of features ultimately leads to a reduction in the CLIPScore.
Even when attempting to accurately describe the composition of the scene, the results are not satisfactory. As illustrated in the example in Figure 4.18, the enriched prompt outlined the relative positions of objects within the scene, yet the model fails in positioning the various entities precisely as intended. Consequently, even with an intricately detailed prompt, the resulting image closely resembles those generated by basic prompts (see Appendix 2), and the CLIPScore remains essentially unaltered.

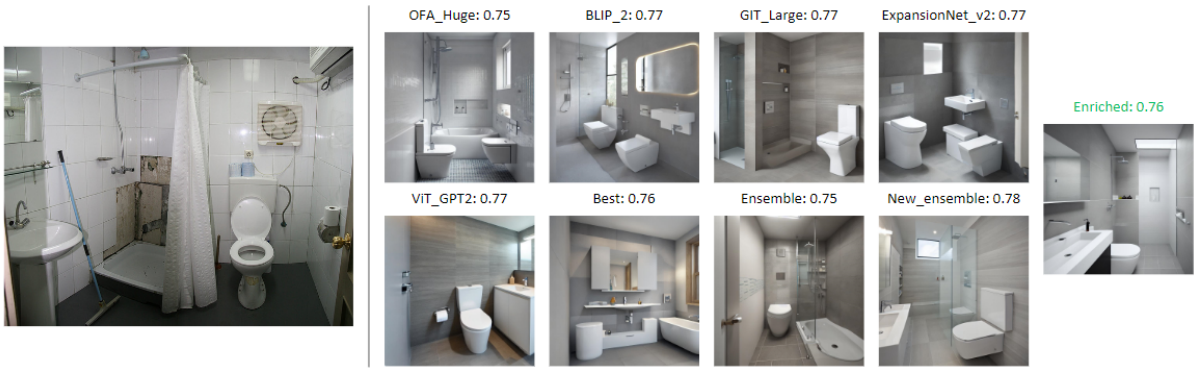

Figure 4.18: Complex scene described with an enriched prompt - Example 2.

In summary, as widely demonstrated, prompts generated by available captioning models are quite basic. While they generally succeed in accurately identifying the main subject of the scene, they often fail to convey all the necessary details to reproduce intricate scenes and all the stylistic elements that contribute to generating an image that is qualitatively and visually similar to the original counterpart. From this perspective, the provided examples illustrate the potential offered by the use of enriched prompts to improve the alignment between synthetic and real images, but they also highlight some important limitations. For this reason, it is crucial to delve deeper into the topic of prompt structure. As with the rest of this study, finding the optimal prompt structure requires extending this work to a larger dataset.
In this regard, the issue of generating enriched prompts needs to be addressed. As the dataset size increases, the solution of manually writing descriptions would quickly become unsustainable. To overcome this challenge, one idea could be to incorporate ChatGPT into the prompt generation process. The concept is to turn ChatGPT into a prompt generator using a script that offers a set of good prompts and guidelines for making high-quality prompts. With this

approach, by providing ChatGPT with the initial image concept, it can then expand the description based on the provided examples and guidelines. An example of this process is shown below:

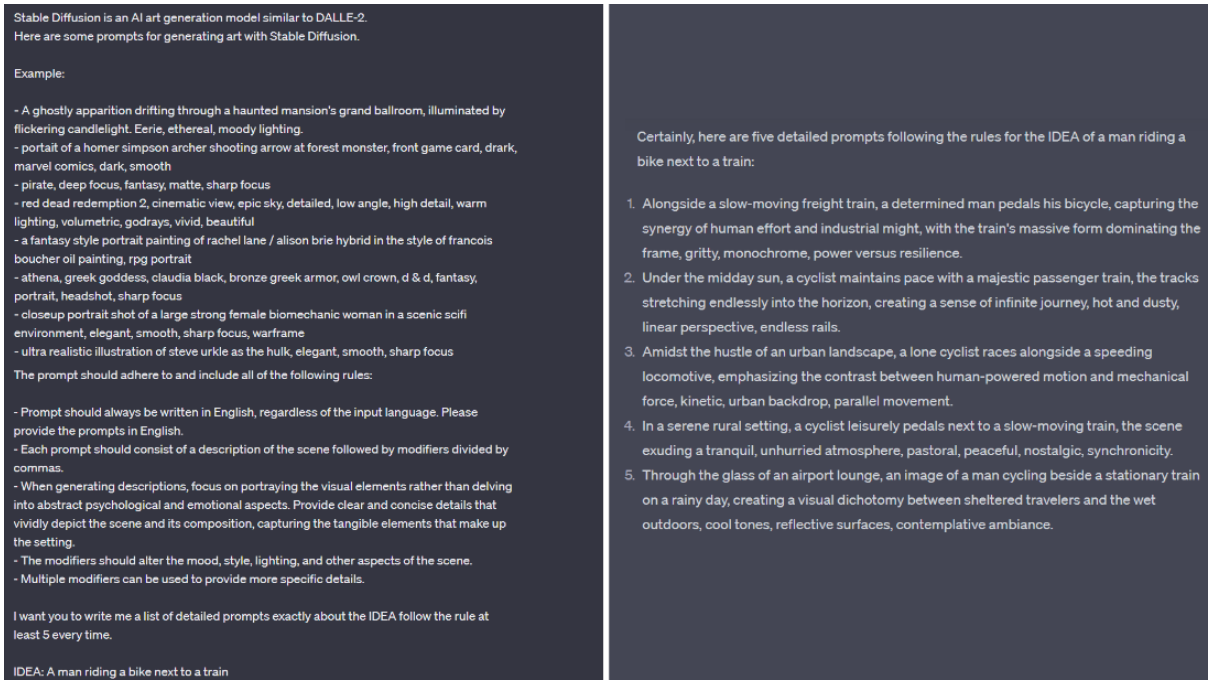

Figure 4.19: ChatGPT as a prompt generator.

Starting from a brief description, which could also be one of the prompts used for the tested captioning models, it is possible to obtain detailed descriptions that follow the provided instructions. This approach is highly flexible, as it can be easily adapted to test different syntactic structures (for example, evaluating whether the order of information given in the prompt affects the importance assigned to each of them) or specific rules (for example, ensuring that all prompts begin with the phrase 'a picture of...' or specifying the name of an artist to draw inspiration from for the image's style). Moreover, more and more gold standard examples can be quickly added as reference points. Equally important, it would make the process of generating these prompts much more efficient, automatable, and scalable.

Chapter 5

Conclusions and Future Works

The aim of this thesis was to assess the utility of generative text-to-image models in evaluating the performance of image captioning models. In particular, the objective was to utilize prompts generated by eight image captioning models, applied to images from the MS-COCO dataset, to generate synthetic images and identify the most effective prompt based on the similarity between the original image and the generated ones.
The findings from this study have showcased the potential of text-to-image models as evaluation tools while also highlighting certain challenges. Firstly, it's evident that there is a lack of metrics ideally suited for this purpose. Traditional metrics commonly employed for evaluating text-to-image models do not assess the similarity between pairs of images, and conventional image similarity metrics fall short as they rely on basic pixel-level functions that do not account for variations in style or different compositions of the same scene. The only metric that proved suitable in this context is CLIPScore, which is capable of evaluating the alignment between feature vectors, even across different modalities, using cosine similarity. However, CLIPScore does not correlate with the quality and realism of the generated images. For instance, when using the same prompt, models with vastly different features and capabilities, such as Stable Diffusion and DALLE Mini, can achieve nearly identical scores simply because their feature vectors are very similar, even though there are significant differences in realism and overall quality between their generated images. Therefore, to comprehensively compare images, CLIPScore should be complemented with a metric capable of evaluating the overall quality and composition of the scenes, such as metrics based on image segmentation techniques. Nonetheless, CLIPScore could prove valuable in improving the results of image captioning models. By adopting a similar approach to CycleGAN, where images are regenerated based on the newly generated description, CLIPScore calculated between the original image and the regenerated one could serve as an effective loss function, guiding the captioning model towards generating an optimal description.
Before identifying the optimal captioning model, we conducted a qualitative assessment of the outcomes generated by the five implemented text-to-image models. In general, all models demonstrated strong performance when dealing with simple subjects (e.g., a traffic light, a kitchen, a pastry shop window). However, notable challenges emerged when generating human bodies, particularly in the case of faces. Additionally, complex scenes presented significant issues, including low quality in individual elements, unrealistic scene compositions, and the presence of generation artifacts. These shortcomings were more pronounced and occurred more frequently in the case of smaller-scale models, such as GLIDE and DALL-E Mini. In the future, with access to more substantial computational resources, it may be possible to alleviate this problem by adopting more powerful models, such as the extended version of GLIDE or the

XL variants of Stable Diffusion.
Returning to the examination of the best captioning model, the top-performing models are Ensemble and New_ensemble. The prompts for these models are generated by combining the two prompts with the highest BLIPScore from a selection that includes OFA_Huge, BLIP_2, GIT_Large, ExpansionNet_v2, and ViT_GPT2. Consequently, they represent the most comprehensive and sophisticated prompts available, illustrating that, overall, they are better at faithfully conveying the characteristics of the original image. Conversely, the model yielding the worst results was ViT_GPT2, which, as also indicated in the findings of [5], tends to provide descriptions of image content with the lowest fidelity and fluency. However, it's worth noting that the conclusions drawn in this aspect are not entirely consistent due to the limited size of the dataset used. In the future, with access to more substantial computational resources, expanding the study to include more than 100 images will be necessary to enhance the robustness of the results.
For the final experiment, an attempt was made to enhance the alignment between the original and synthetic images. Given that CLIPScore does not strongly correlate with the overall image quality, the focus shifted toward the structure of prompts. From this perspective, it has been demonstrated that enriched prompts enable the reproduction of various stylistic elements that basic prompts tend to overlook. The utilization of these enriched prompts has resulted in CLIPScore improvements in several instances. However, it's important to highlight that the structure of prompts plays a pivotal role. Certain characteristics, such as perspective and lighting, are challenging to transfer to the generative model, and in some cases, the addition of new information can divert attention away from the central elements of the scene, leading to a decrease in the CLIPScore. To identify the optimal prompt structure, a larger dataset is required, and the analysis should be made more systematic and scalable, potentially by implementing tools such as ChatGPT as a prompt generator.

Appendix 1: Benchmark Dataset

Oxford-102

The Oxford-102 dataset is one of the most widely used datasets for the text-to-image generation task, especially with GAN-based models. It is a dataset published in 2008 [38], consisting of 102 categories of flowers, with a total of more than 8,000 images. Initially used for classification and segmentation tasks, in 2015 [45] the dataset undergoes an annotation task to create 10 fine-grained text descriptions for each image, making it one of the main reference datasets for the text-to-image task.

Caltech-UCSD Bird (CUB)

The Caltech-UCSD Birds-200-2011 (CUB-200-2011) dataset [59] is the extended version of the CUB-200 dataset, produced and published by Caltech University. The dataset contains over 11,000 images of birds, divided into 200 categories. Images are annotated with information related to part location, binary attributes, and bounding boxes. Like Oxford-102, 10 text descriptions for each image were added in 2015 thanks to the contribution of Reed et al., making the dataset suitable for text-to-image generation tasks.

MS-COCO

The Microsoft-COCO [35] dataset is a large-scale dataset published in 2014 and widely used for various image-related tasks such as image classification, semantic segmentation, object detection, and text-to-image generation. The dataset collects more than 120,000 images of complex everyday scenes containing common objects in their natural context. Given the variety of categories and images contained within it, MS-COCO has been one of the most challenging datasets for text-to-image generation.

Appendix 2: Prompts

This section will show the prompts of the eight captioning models for all the examples shown in the project. Next to each image are the eight prompts with the names of the respective models highlighted in red. In addition, in green, the manually written enriched prompt is also reported.

- OFA_Huge: "a plate with a sandwich and a salad on it"
- BLIP_2: "a plate with a sandwich and a salad on it"
- GIT_Large: "a sandwich cut in half on a plate"
- ExpansionNet_v2: "a plate with a sandwich and a salad on a table"
- ViT_GPT2: "a sandwich with lettuce, tomato, and cheese on a plate"
- Best: "a plate with a sandwich and a salad on a table"
- Ensemble: "A plate with a sandwich and a salad on it is placed on a table"
- New_ensemble: "A plate with a sandwich and a salad on it is placed on a table"
- Enriched prompt: "A photo of a white plate with a sandwich and some salad"

- OFA_Huge: "a sheep standing in a field of grass"
- BLIP_2: "a sheep standing on top of a lush green field"
- GIT_Large: "a sheep standing in a field of grass"
- ExpansionNet_v2: "a sheep standing in the grass in a field"
- ViT_GPT2: "a sheep standing in a field of grass"
- Best: "a sheep standing in the grass in a field"
- Ensemble: "A sheep stands in a field of grass"
- New_ensemble: "A sheep is standing in a field of grass"
- Enriched prompt: "A photo of a sheep in a grass field with some trees blurred in the background"

- OFA_Huge: "a zebra is eating grass in a field"
- BLIP_2: "a black and white photo of a zebra grazing in the grass"
- GIT_Large: "a zebra eating grass in a field with a sky background"
- ExpansionNet_v2: "a black and white photo of a zebra grazing in the grass"
- ViT_GPT2: "a zebra standing in a field of grass"
- Best: "a black and white photo of a zebra grazing in the grass"
- Ensemble: "A black and white photo of a zebra grazing in the grass captures the beauty of nature"
- New_ensemble: "A black and white photo of a zebra grazing in the grassy field"
- Enriched prompt: "A black and white close up of a zebra eating grass in a meadow"

- OFA_Huge: "a traffic light with a blue sky with clouds"
- BLIP_2: "a close up of a traffic light with clouds in the background"
- GIT_Large: "a traffic light with three lights on it"
- ExpansionNet_v2: "two traffic lights on a pole with a blue sky"
- ViT_GPT2: "a traffic light with a bunch of traffic lights"
- Best: "two traffic lights on a pole with a blue sky"
- Ensemble: "A close-up of two traffic lights on a pole against a blue sky with clouds in the background"
- New_ensemble: "A close-up of two traffic lights on a pole against a blue sky with clouds is the best descriptive caption for the image"
- Enriched prompt: "A close up photo of a traffic light. In the background you can see a clear sky with some clouds"

- OFA_Huge: "a kitchen with a sink and a window"
- BLIP_2: "a kitchen with a sink and a window"
- GIT_Large: "a kitchen with a sink, dishwasher, and window"
- ExpansionNet_v2: "a kitchen with a sink and a window"
- ViT_GPT2: "a kitchen with a sink, stove, and window"
- Best: "a kitchen with a sink, dishwasher, and window"
- Ensemble: "A kitchen with a sink, dishwasher, and window, offering a view of the outdoors"
- New_ensemble: "This kitchen features a sink, dishwasher, and window"
- Enriched prompt: "A photo of an old-fashioned kitchen with a sink, a wooden kitchen top with a lot of kitchen tools and two windows. The kitchen is seen from the right. There is little lighting inside the kitchen"

- OFA_Huge: "a group of people standing on a sidewalk near a carnival"
- BLIP_2: "a group of people standing on a sidewalk"
- GIT_Large: "a group of people standing on a sidewalk"
- ExpansionNet_v2: "a group of people standing on a sidewalk"
- ViT_GPT2: "a woman and a child playing with a toy in a park"
- Best: "a group of people standing on a sidewalk near a carnival"
- Ensemble: "A group of people standing on a sidewalk near a carnival, taking in the sights and sounds of the festivities"
- New_ensemble: "A group of people are standing on a sidewalk near a carnival"
- Enriched prompt: "A photo of a square filled with people. In the foreground there is a woman seen from behind who takes a picture of a man with his hands raised, a little girl in a pink shirt and a boy in a blue shirt who pushes his mother into his back"

- OFA_Huge: "a wooden table with a vase of flowers on top of it"
- BLIP_2: "a wooden table with a vase of sunflowers on it"
- GIT_Large: "a table with a vase of flowers on it"
- ExpansionNet_v2: "a dining room with a table and chairs with flowers"
- ViT_GPT2: "a dining room table with a window"
- Best: "a wooden table with a vase of sunflowers on it"
- Ensemble: "A wooden table with a vase of sunflowers atop it"
- New_ensemble: "A dining room with a wooden table topped with a vase of flowers, such as sunflowers, provides a beautiful and inviting atmosphere"
- Enriched prompt: "A photo of the inside of a cabin in the woods, with a wooden table with a vase of sunflowers on it, wooden chairs around the table, and windows looking outside. Outside you can see several trees"

- OFA_Huge: "a flock of birds flying through a cloudy sky"
- BLIP_2: "a black and white photo of birds flying over a body of water"
- GIT_Large: "a black and white photo of birds flying in the sky"
- ExpansionNet_v2: "a group of birds flying over the water"
- ViT_GPT2: "a flock of birds flying over a body of water"
- Best: "a black and white photo of birds flying over a body of water"
- Ensemble: "A black and white photo of birds flying over a body of water and in the sky"
- New_ensemble: "A black and white photo of a group of birds flying over a body of water"
- Enriched prompt: "A black and white photo of a flock of black birds flying across the sky"

- OFA_Huge: "a man wearing a black shirt and a green tie"
- BLIP_2: "a man wearing a tie with leaves on it"
- GIT_Large: "a man wearing glasses and a green tie"
- ExpansionNet_v2: "a man wearing a black shirt and a tie"
- ViT_GPT2: "a man wearing a tie and a shirt"
- Best: "a man wearing a black shirt and a green tie"
- Ensemble: "The man was wearing a black shirt, glasses, and a green tie"
- New_ensemble: "A man wearing a black shirt, glasses, and a tie with leaves on it"
- Enriched prompt: "A medium close up photo of a man wearing a black shirt, black pants and a green tie with black flowers. The man is white, has no beard and has short black hair. The man is wearing glasses and his eyes are closed"

- OFA_Huge: "a man sitting at a table with a pizza"
- BLIP_2: "a man sitting at a table with a large pizza"
- GIT_Large: "a man sitting at a table with a pizza on it"
- ExpansionNet_v2: "a man sitting at a table with a pizza"
- ViT_GPT2: "a man sitting at a table with a pizza"
- Best: "a man sitting at a table with a large pizza"
- Ensemble: "A man sitting at a table with a large pizza on it"
- New_ensemble: "A man is sitting at a table with a large pizza"
- Enriched prompt: "A photo of a man sitting at a restaurant table. On the table is a pizza, two empty plates, and two glasses containing brown liquid. The atmosphere in the restaurant is gloomy, the scene is dimly lit"

- OFA_Huge: "a wooden table with many wooden spoons on it"
- BLIP_2: "a table with a lot of wooden spoons on it"
- GIT_Large: "a table with many wooden spoons and a few chairs"
- ExpansionNet_v2: "a wooden table with wooden spoons on it"
- ViT_GPT2: "a table with a bunch of wooden chairs around it"
- Best: "a table with many wooden spoons and a few chairs"
- Ensemble: "A table with many wooden spoons and a few chairs, displaying a lot of them on its surface"
- New_ensemble: "A wooden table with many wooden spoons and a few chairs on it"
- Enriched prompt: "A photo of wooden kitchen tools arranged on a wooden table. There are six chairs with blue seat and red back around the table. A bookshelf with lots of books can be partially seen in the background. The table and the bookshelf are in a room with white walls and wooden floor"

- OFA_Huge: "a young man holding a basketball next to a basketball hoop"
- BLIP_2: "a man holding a basketball in front of a brick wall"
- GIT_Large: "a young man holding a basketball in his hand"
- ExpansionNet_v2: "a man holding a basketball on a court"
- ViT_GPT2: "a young man is holding a basketball in his hand"
- Best: "a man holding a basketball in front of a brick wall"
- Ensemble: "A young man is holding a basketball in front of a brick wall next to a basketball hoop"
- New_ensemble: "A young man is holding a basketball in his hand in front of a brick wall, next to a basketball hoop"
- Enriched prompt: "A stock photo of a basketball player holding a basketball with a hoop behind him. The player has blonde hair and wears a white and green uniform"

- OFA_Huge: "a boy wearing bananas on his head and wearing a costume"
- BLIP_2: "a man in a costume with a bunch of bananas on his head"
- GIT_Large: "a woman wearing a bunch of bananas on her head"
- ExpansionNet_v2: "a young boy wearing a bunch of bananas"
- ViT_GPT2: "a man wearing a banana hat and holding a banana"
- Best: "a boy wearing bananas on his head and wearing a costume"
- Ensemble: "A boy and a man wearing costumes with bananas on their heads"
- New_ensemble: "A young boy wearing a costume with a bunch of bananas on his head"
- Enriched prompt: "A close up of a boy wearing a light blue and purple shirt, a necklace made out of bananas and a hat made out of bananas. Behind him there are two men dressed in black looking at him. The background is blurred"

- OFA_Huge: "a herd of sheep grazing in a field"
- BLIP_2: "a herd of sheep standing in a field next to a house"
- GIT_Large: "a herd of sheep standing next to a fence"
- ExpansionNet_v2: "a herd of sheep standing in a field of grass"
- ViT_GPT2: "sheep are standing in a field"
- Best: "a herd of sheep standing in a field next to a house"
- Ensemble: "A herd of sheep standing in a field next to a house and a fence"
- New_ensemble: "A herd of sheep is standing in a field of grass next to a house or fence"
- Enriched prompt: "A photo of a flock of sheep in a snowy meadow. In the background you can see a ruined farmhouse and several bare trees"

- OFA_Huge: "a white bathroom with a shower and a toilet"
- BLIP_2: "a bathroom with a toilet a sink and a shower"
- GIT_Large: "a bathroom with a shower, toilet and a shower"
- ExpansionNet_v2: "a bathroom with a toilet and a sink"
- ViT_GPT2: "a bathroom with a toilet, sink, and shower stall"
- Best: "a bathroom with a toilet a sink and a shower"
- Ensemble: "A bathroom with a toilet, sink, and shower stall, including a shower"
- New_ensemble: "This bathroom features a white shower, toilet and sink"
- Enriched prompt: "A photo of a white bathroom. At the back of the bathroom is a shower to the left and a toilet to the right. In front of the shower there is a sink. There are some broken tiles on the bathroom wall"

- OFA_Huge: “a man riding a motorcycle on a dirt road”
- BLIP_2: “a person riding a motorcycle on a dirt road”
- GIT_Large: “a man riding a motorcycle down a dirt road”
- ExpansionNet_v2: “a man riding a motorcycle on a dirt road”
- ViT_GPT2: “a man riding a motorcycle on a dirt road”
- Best: “a man riding a motorcycle down a dirt road”
- Ensemble: “A man riding a motorcycle down a dirt road”
- New_ensemble: “A man is riding a motorcycle down a dirt road”
- Enriched prompt: “A photo of a man riding a motorcycle on a dirt road, the man wears a red jacket and a red helmet, there are grass and bushes on the side of the road, in the background there is a tree-lined mountain partially covered by clouds. The scene is photographed from behind”

Bibliography

[1] Mohammed Abdulameer Aljanabi et al. “Design of a hybrid measure for image similarity: a statistical, algebraic, and information-theoretic approach”. In: *European Journal of Remote Sensing* 52.sup4 (2019), pp. 2–15.

[2] Martin Arjovsky and Léon Bottou. “Towards principled methods for training generative adversarial networks”. In: *arXiv preprint arXiv:1701.04862* (2017).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014).

[4] Yoshua Bengio et al. “Better mixing via deep representations”. In: *International conference on machine learning*. PMLR. 2013, pp. 552–560.

[5] Simone Bianco et al. “Improving Image Captioning Descriptiveness by Ranking and LLM-based Fusion”. In: *arXiv preprint arXiv:2306.11593* (2023).

[6] Navaneeth Bodla, Gang Hua, and Rama Chellappa. “Semi-supervised FusedGAN for conditional image generation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 669–683.

[7] Tom Brown et al. “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[8] Mark Chen et al. “Generative pretraining from pixels”. In: *International conference on machine learning*. PMLR. 2020, pp. 1691–1703.

[9] Jun Cheng et al. “Rifegan: Rich feature generation for text-to-image synthesis from prior knowledge”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 10911–10920.

[10] Prafulla Dhariwal and Alexander Nichol. “Diffusion models beat gans on image synthesis”. In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794.

[11] Prafulla Dhariwal et al. “Jukebox: A generative model for music”. In: *arXiv preprint arXiv:2005.00341* (2020).

[12] Ming Ding et al. “Cogview: Mastering text-to-image generation via transformers”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 19822–19835.

[13] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[14] Patrick Esser, Robin Rombach, and Bjorn Ommer. “Taming transformers for high-resolution image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 12873–12883.

[15] Lianli Gao et al. “Perceptual pyramid adversarial networks for text-to-image synthesis”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 8312–8319.

[16] Ian Goodfellow et al. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[17] Karol Gregor et al. “Draw: A recurrent neural network for image generation”. In: *International conference on machine learning*. PMLR. 2015, pp. 1462–1471.

[18] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[19] Jack Hessel et al. “Clipscore: A reference-free evaluation metric for image captioning”. In: *arXiv preprint arXiv:2104.08718* (2021).

[20] Martin Heusel et al. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *Advances in neural information processing systems* 30 (2017).

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[22] Jonathan Ho and Tim Salimans. “Classifier-free diffusion guidance”. In: *arXiv preprint arXiv:2207.12598* (2022).

[23] Seunghoon Hong et al. “Inferring semantic layout for hierarchical text-to-image synthesis”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7986–7994.

[24] NLP Connect. vit-gpt2-image-captioning (revision 0e334c7). 2022. URL: `https://huggingface.co/nlpconnect/vit-gpt2-image-captioning`.

[25] Jia Cheng Hu, Roberto Cavicchioli, and Alessandro Capotondi. “Expansionnet v2: Block static expansion in fast end to end training for image captioning”. In: *arXiv preprint arXiv:2208.06551* (2022).

[26] Minguk Kang et al. “Scaling up gans for text-to-image synthesis”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 10124–10134.

[27] Tero Karras et al. “Analyzing and improving the image quality of stylegan”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8110–8119.

[28] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[29] Taku Kudo and John Richardson. “Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing”. In: *arXiv preprint arXiv:1808.06226* (2018).

[30] Wei-Sheng Lai et al. “Deep laplacian pyramid networks for fast and accurate super-resolution”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 624–632.

[31] Bowen Li et al. “Controllable text-to-image generation”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[32] Junnan Li et al. “Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models”. In: *arXiv preprint arXiv:2301.12597* (2023).

[33] Wenbo Li et al. “Object-driven text-to-image synthesis via adversarial training”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12174–12182.

[34] Tsung-Yi Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[35] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 740–755.

[36] Elman Mansimov et al. “Generating images from captions with attention”. In: *arXiv preprint arXiv:1511.02793* (2015).

[37] Alex Nichol et al. “Glide: Towards photorealistic image generation and editing with text-guided diffusion models”. In: *arXiv preprint arXiv:2112.10741* (2021).

[38] Maria-Elena Nilsback and Andrew Zisserman. “Automated flower classification over a large number of classes”. In: *2008 Sixth Indian conference on computer vision, graphics & image processing*. IEEE. 2008, pp. 722–729.

[39] Tingting Qiao et al. “Mirrorgan: Learning text-to-image generation by redescription”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1505–1514.

[40] Alec Radford and Jeffrey Wu. “Rewon child, david luan, dario amodei, and ilya sutskever. 2019”. In: *Language models are unsupervised multitask learners. OpenAI blog* 1.8 (2019), p. 9.

[41] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[42] Aditya Ramesh et al. “Hierarchical text-conditional image generation with clip latents”. In: *arXiv preprint arXiv:2204.06125* 1.2 (2022), p. 3.

[43] Aditya Ramesh et al. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[44] Scott Reed et al. “Generative adversarial text to image synthesis”. In: *International conference on machine learning*. PMLR. 2016, pp. 1060–1069.

[45] Scott Reed et al. “Learning deep representations of fine-grained visual descriptions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 49–58.

[46] Scott Reed et al. “Learning to disentangle factors of variation with manifold interaction”. In: *International conference on machine learning*. PMLR. 2014, pp. 1431–1439.

[47] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. “Stochastic backpropagation and approximate inference in deep generative models”. In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[48] Robin Rombach et al. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: `1505.04597 [cs.CV]`.

[50] Chitwan Saharia et al. “Photorealistic text-to-image diffusion models with deep language understanding”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 36479–36494.

[51] Tim Salimans et al. “Improved techniques for training gans”. In: *Advances in neural information processing systems* 29 (2016).

[52] Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural machine translation of rare words with subword units”. In: *arXiv preprint arXiv:1508.07909* (2015).

[53] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *International conference on machine learning*. PMLR. 2015, pp. 2256–2265.

[54] Casper Kaae Sønderby et al. “Amortised map inference for image super-resolution”. In: *arXiv preprint arXiv:1610.04490* (2016).

[55] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *arXiv preprint arXiv:2011.13456* (2020).

[56] Yonglong Tian, Dilip Krishnan, and Phillip Isola. “Contrastive multiview coding”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer. 2020, pp. 776–794.

[57] Aaron Van Den Oord, Oriol Vinyals, et al. “Neural discrete representation learning”. In: *Advances in neural information processing systems* 30 (2017).

[58] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[59] Catherine Wah et al. “The caltech-ucsd birds-200-2011 dataset”. In: (2011).

[60] Jianfeng Wang et al. “Git: A generative image-to-text transformer for vision and language”. In: *arXiv preprint arXiv:2205.14100* (2022).

[61] Peng Wang et al. “Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework”. In: *International Conference on Machine Learning*. PMLR. 2022, pp. 23318–23340.

[62] Zhendong Wang et al. “Diffusion-gan: Training gans with diffusion”. In: *arXiv preprint arXiv:2206.02262* (2022).

[63] Zhou Wang and Alan C Bovik. “A universal image quality index”. In: *IEEE signal processing letters* 9.3 (2002), pp. 81–84.

[64] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.

[65] Tao Xu et al. “Attngan: Fine-grained text to image generation with attentional generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1316–1324.

[66] Guojun Yin et al. “Semantics disentangling for text-to-image generation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 2327–2336.

[67] Roberta H Yuhas, Alexander FH Goetz, and Joe W Boardman. “Discrimination among semi-arid landscape endmembers using the spectral angle mapper (SAM) algorithm”. In: *JPL, Summaries of the Third Annual JPL Airborne Geoscience Workshop. Volume 1: AVIRIS Workshop*. 1992.

[68] Han Zhang et al. “Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5907–5915.

[69] Han Zhang et al. “Stackgan++: Realistic image synthesis with stacked generative adversarial networks”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018), pp. 1947–1962.

[70] Lin Zhang et al. “FSIM: A feature similarity index for image quality assessment”. In: *IEEE transactions on Image Processing* 20.8 (2011), pp. 2378–2386.

[71] Zizhao Zhang, Yuanpu Xie, and Lin Yang. “Photographic text-to-image synthesis with a hierarchically-nested adversarial network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6199–6208.

[72] Minfeng Zhu et al. “Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5802–5810.