Politecnico di Torino

Ingegneria Informatica
A.a 2023/2024
Sessione di Laurea Dicembre 2024

A Scene Graph-based approach for Text-to-Image generation

Relatori:

Prof. Lia Morra

Candidato:

Leonardo Dardanello

Acknowledgements

As I come to the end of my Master's degree, I feel the need to dedicate a few lines to the people whose various contributions have made this possible. First of all, I would like to thank Professor Lia Morra for her guidance and supervision throughout the duration of my thesis. Thanks to her, I was able to experience first-hand what it means to be involved in a research project. I would also like to thank Dr Pietro Basci for his support and patience, from whom I learned the rigour and precision that the scientific method requires. A special thanks goes to my family: my mother and my brother. For five years they have been my first line of support, helping me in any way they could, without them I would never have made it. To my aunt and uncle and who treated me like their own child and to Luca, for all those evenings when we only needed two old computers to play and time seemed to never pass, thanks to those moments my passion for computers started. To Barbara for being a close advisor and a person of trust, even in the harder times. To Gaia for her help and advices in creating custom images. To Alessandro, Marco, Paola, Alessandro D., Giorgio and Alessandro B., for accompanying me over the years, for all the sessions we prepared together and for cheering us up when we failed an exam, you were an integral part of my everyday life and made even the bad days more bearable. Finally, I would like to express my gratitude my father, grandfather and grandmother, although you left us too soon, you taught me the true values that matter in life, your memory lives on in me. Each person I have thanked has contributed in a unique way to this journey, making it not only possible, but enriching it in many ways. To all of them, my sincere thanks.

Abstract

Recently, the field of artificial intelligence (AI) research has experienced a surge in popularity, both in terms of the quantity of work produced and in the quality of the results obtained. Among these, generative AI models have become among the most popular and appreciated due to their positive performance. In this thesis, the context of Text-to-Image models, i.e. generative models for the creation of images from textual descriptions of the desired image, is first framed. Then, models that can exploit other forms of input, such as pictures, to produce synthetic images are presented. The main contribution of this work is the introduction of an adapter framework based on scene graphs to condition image generation with the aim of exploiting the structured form of the graph to improve object spatial relationships in the generated images. The framework presented consists of a parser capable of converting text into a scene graph, a conditioning adapter based on a Graph Convolutional Network (GCN), and Stable Diffusion 1.5 [38] as a backbone. The whole framework is fine-tuned using a scene graph extracted from captions belonging to a pre-existing dataset. Experiments exploiting more established metrics, such as FID [42], and more modern image-aware benchmarks, such as the TIFA [20] benchmark, are performed on different GCN configurations to determine the more performing GCN adapter configuration. To evaluate the parser, an ad hoc benchmark, specificillay, focused on the spatial relationship in applied. Finally, the conclusion and future work are described

Contents

List of Figures

Chapter 1

Introduction

1.1 Context

Recent developments in artificial intelligence have led to the development of new architectures and models capable of generating images other than those with which they were trained; the term Generative AI has been coined to define these models. Among these architectures, some specialize in the task of "text-to-image synthesis", i.e. the generation of an image from a textual description provided by the user. There are two main approaches to this task: Autoregression and diffusion; however, in recent years diffusion models have obtained better performances and will be the main focus of this work. Although these models are capable of generating high-quality images and can produce different styles, they still have difficulty following the instructions contained in the prompt. Some of the most notable hardships includes spatial object relationships, attribute matching, counting, and reproducing artistic styles. An example of this type of caption is provided below.

```
Spatial object relationships: An apple next to a banana
Attribute matching: A green car with black rims
Counting: Three cups of coffe and two spoons
Artistic style: A sunset depicted in the style of Van Gogh
```

1.2 Purpose

The main purpose of this work is to address the challenges related to the interpretation of spatial relations by the Text-to-Image frameworks, in particular their ability to correctly position objects inside the image. Specifically, the analysis will be conducted on diffusion models. In an attempt to better capture the relationships between the objects in the textual description described by the user, a graph is introduced as a form of structured data, so that the edges of the graph can be exploited as a structural syntactic link between the entities in the image. More precisely, the scene graph, a specific kind of graph, has been chosen as conditioning. A trainable adapter for diffusion models, SG Diffusion, has been developed in order to enable conditioning through a scene graph. The main features of this custom adapter are the ability to extract a scene graph from the user's input text, and to condition the diffusion process through a Graph Convolutional Network (GCN) to better enforce the spatial positions requested by the user.

1.3 Outline

The first chapter provides a general overview of the context of the research area covered by the thesis, together with the motivation behind the creation of this thesis. The second chapter presents the current state of the art in the literature on text-to-image models. It also discusses the problem of evaluating a generative model and how there is currently no single solution. The third chapter provides a background to the knowledge used in this thesis. The fourth chapter introduces SG Diffusion, an adapter for diffusion models that allows the generation of synthetic images exploiting a scene graph. The fifth chapter presents the dataset used for training and validation, and explains the filtering and sampling procedures used. The sixth chapter presents the results obtained by testing several GCNs as adapters, evaluated using the TIFA benchmark and the FID metric. The performance of the fine-tuned model using the most performing GCN is then presented. The seventh chapter presents the conclusion and possible future developments.

Chapter 2

State Of The Art

2.1 State of the Art

The current state of the art regarding Diffusion architectures is characterized by both open source and proprietary models like Stable Diffusion 3 ([12]), PixArt-α ([8]), DALL-E 2 and Midjourney. Autogressive models, while capable of generating detailed and articulated synthetic images, they still perform poorly in terms of object placement, due to the fact that they inherently introduce an unidirectional bias [52] and thus lag behind diffusion models when benchmarked. To enhance and improve the generated images, several methods have been proposed, all of which share a common goal: to condition the diffusion process by providing some extra information. Some methods, like Directed Diffusion ([30]), manipulate the attention weights of the cross-attention layers to stimulate the generation in user-defined bounding-boxes, while other works, like LMD ([26]) employ a "Layout-to-Image" strategy that exploits LLM models to create a layout of the image to generate.

Figure 2.1: First four image (from the right) represent four different denoising step with Directed Diffusion, the two on the left represents the cross-attention maps for the word 'cat' at the first and at the last denoising step. Image reproduced from [30].

These methods, while effective, allow to employ only one single form of external guidance, instead Adapter frameworks like ControlNet ([56]) and IP-Adapter ([54]) consist in add-on networks that directly communicate with the main diffusion model, thus allowing to extract and elaborate information from different sources, such as: Canny Edges,User Scribbles, Human Poses, Depth maps, and Semantic Maps. Furthermore, multiple instances of this adapter modules can be used jointly to guide the diffusion using different sources. Although ControlNet and IP-Adapter can exploit different ways to condition the original model, they still fail in capturing and representing the spatial relationship embedded in the text input. SG-Adapter [46] tries to devise an adapter framework that can effectively exploit relationships between the object described in the textual input. It does so by using a "Scene Graph-to-Image" approach, which uses a Scene Graph extracted directly from the text input to enforce the relationship between the entities of the graph. To further improve its ability to reason with triplets, SG-Adapter introduces a custom attention mask that groups and masks each triplet of the Scene Graph, rather than masking each entity separately. Having evaluated

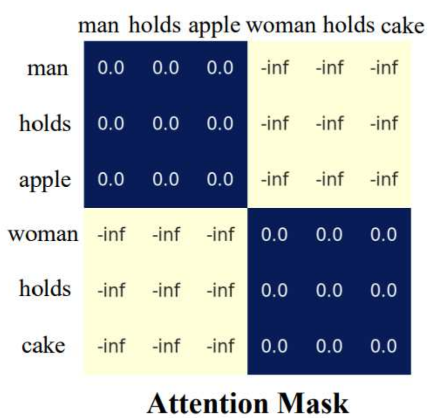


Figure 2.2: Token per token Attention Mask. Image reproduced from [46]

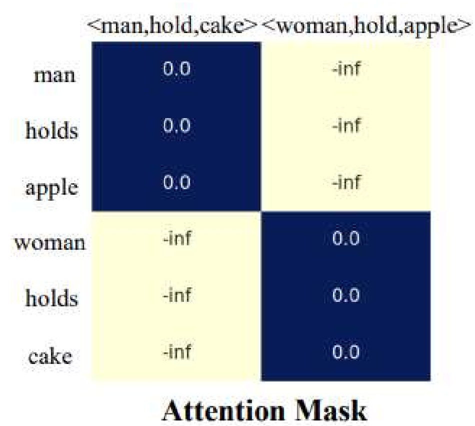


Figure 2.3: SG-Adapter's per Triplet attention Mask, taken from [46]

the current State of the Art, this work aims to improve object placement in synthetic image generation using structured data, such as graphs, taking IP-Adapter and SG-Adapter as a starting point.

2.2 Measurements and Benchmark

In the literature the area of evaluating the efficacy of a Generative model is extensive and lacks a singular, definitive methodology. When generating images, like in “Text-to-Image” applications, traditionally the Fréchet Inception Distance (FID) [19] and the Inception Score (IS) [40] have been used. The FID measure employs two image sets: one synthetic, generated by the model under evaluation, and one comprising real-world images. These are used to fit two Gaussian distributions and to evaluate the similarity between them. Let $G(\mu_r, \Sigma_r)$ be the gaussian distribution estimated on the real-world images and $G(\mu_s, \Sigma_s)$ the gaussian distribution crafted on the synthethic images, the FID is given by:

$$FID(G(\mu_r, \Sigma_r), (\mu_s, \Sigma_s)) = ||\mu_r - \mu_s||_2^2 + Tr(\Sigma_r + \Sigma_s - 2(\Sigma_r \Sigma_s)^{\frac{1}{2}}) \quad (2.1)$$

The closer the generated images are to the real-world image, the lower the FID values. The IS measures the quality and diversity of the image by using a pre-trained model to evaluate if the images are easily classifiable and if they belong to many different classes. These metrics do not measure the accuracy of the entities spatial placement inside the images, thus new metrics and frameworks have been developed to take into consideration the image composition, however a standardized evaluation setup has not emerged yet. The schema presented in Figure 2.4, by Hartwig et al., represents the classification of existing metrics.

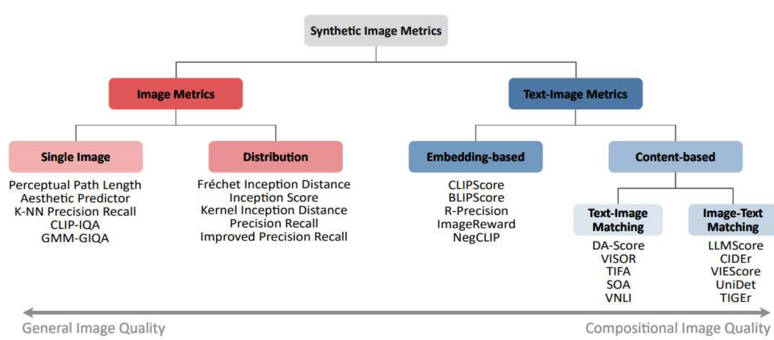


Figure 2.4: Classification of different image evaluation categories. Image by [16]

Both FID and IS metrics considers the images as black-boxes, thus are not suited to perform detailed evaluations of fine-grained picture characteris-

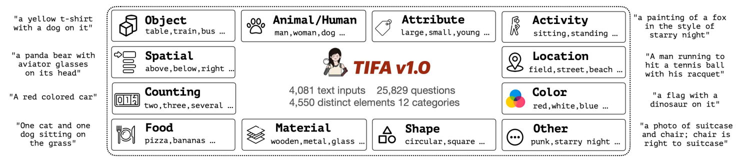


Figure 2.5: Description of the TIFA benchmark. Image by [20]

tics. However, since they are commonly employed in the literature to evaluate model performance, they still remains a useful benchmark to compare models against each other. More recent evaluation setups, exploits a more content-oriented approach to perform a detailed evaluation of the images. These frameworks employs images with an associated textual description to evaluate the correlation between the image and the textual description. One of the most prevalent methods that uses text-image pairs is CLIPScore [18], which is based on CLIP [34], an image-sentence alignment (ISA) model. CLIPScore evaluates whether the content described in the textual input is depicted in the generated images by performing an embedding of both the image and the textual description, the similarity of the embedding is the calculated through a cosine distance. Despite CLIPScore being more content-aware than previous setups, it still fails in providing a detailed compositional evaluation, returning only a numerical score. To properly assess the compliance of the spatial relationships, a more detailed benchmark is needed. Among the exsisting literature, the TIFA [20] benchmark was identified as the most appropriate for the purpose of this work, given its considerable number of benchmarking captions and its broad range of available categories. Other similar TIFA provides a detailed examination of the generated images by employing a VQA model to evaluate the 4081 provided text inputs over more than 25k of question-answer pairs, both in the form of single answer questions. TIFA evaluates the generated image among 12 categories (Object, Spatial, Counting, Food, Animal/Human, Material, Shape, Attribute, Activity, Location, Color, Other), where the "Other" category represents the ability to reproduce particular artistic styles. In order to assign a score to each question, the TIFA faithfullness score is used and expressed as follows:

$$faithfulness(T, I) = \frac{1}{N}\sum_{i=1}^{N} 1[\mathcal{A}^{VQA} = \mathcal{A}_i] \tag{2.2}$$

where N is the number of possible answers, $\mathcal{A}^{VQA}$ is the answer provided by the VQA model and $\mathcal{A}_i$ is the correct answer. Regarding diffusion-based frameworks, recent models like Stable Diffusion 3 [12], DALL-E 2 [35], PixArt-α [8] have been benchmarked with TIFA, their performances are reported in Table 6.7.

Chapter 3

Background

3.1 Diffusion Models

Diffusion models are a subset of generative models, more specifically they are a probabilistic generative models that are able to learn how to reverse a degradation process that gradually affects the training data. They have already been deployed and obtained good performance in several different field, such as: image generation ([38],[12],[35]), image-to-image translation ([39]) and image inpainting ([29]). For each diffusive model, the training procedure is divided into two parts: a forward diffusion step and a backward denoising process. In the first stage the training sample, $x_0 \sim p(x_0)$, is progressively transformed into pure Gaussian noise by adding low-level noise at each step, with the amount of noise varying at each step. This process can be represented as the following Markovian process:

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t;\sqrt{1-\beta_t} \cdot x_{t-1}, \beta \cdot \mathrm{I}), \forall t \in \{1, ..., T\} \tag{3.1}$$

where x_t is the training sample at the step t,T are the number of steps, $\beta_1, ..., \beta T$ are hyperparemeters adjusting the variance schedule across the $t-th$ step and I is the identity matrix. The second phase is the denoising step and is characterised by the inverse of the first phase, starting from a sample $x_T \sim \mathcal{N}(0, \mathrm{I})$, the noise is progressively removed until an acceptable image is obtained, according to the following expression:

$$p(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu(x_t, t), \Sigma(x_t, t)\right), \forall t \in \{1, ..., T\} \tag{3.2}$$

where $\mu(x_t, t)$ and $\Sigma(x_t, t)$ are respectively the mean and the variance of the $t-th$ sample at step t.

3.2 Scene Graphs

Images are characterised by a high level of detail and by the presence of intricate relationships between the depicted actors. When trying to extract information from images, previous models such as Faster R-CNN ([37]) employed an isolated approach, generating a spatial description through bounding boxes and focusing exclusively on the objects represented and their position, failing to consider the semantic relationships embedded within the image. An example of this problem is shown in Figure 3.1.

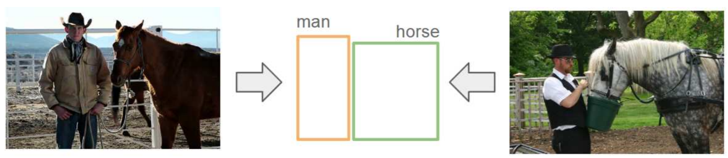


Figure 3.1: (left image) A man standing next to a horse; (right image) A man feeding a horse. Both images produce the same bounding boxes, ignoring completely the semantic concepts. Image by [53].

To tackle this problem Johnson et al., introduced the scene graphs. This approach enables a visually-grounded modelling of an image's content, taking into account entities instances, the attributes of the entities, and relationships between entities. An example of scene graph is shown in Figure 3.2.

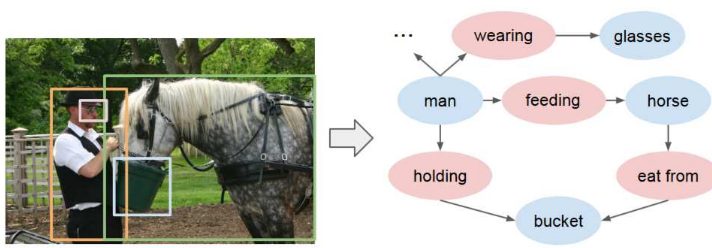


Figure 3.2: Example of scene graph, in blue the entities while in red the relationships. Image by [53].

Due to their capacities, the scene graphs found numerous application and nowadays are employed in several fields such as: 3D scene synthesis ([6]), image retrival ([22]) and visual question answering ([49]).

3.3 Cross-Attention

Cross-attention represents a specific attention mechanism that is employed in a number of fields, including: Machine Translation ([9], Visual Question Answering ([28]) and Text-to-Image generation ([12]). It is a well-documented mechanism, particularly as one of the primary structural components of the Transformers ([50]) architectural framework as shown in 3.3. Given a generic

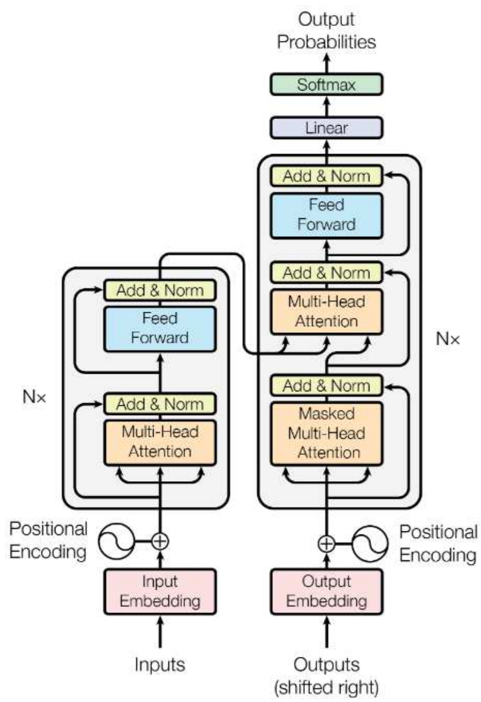


Figure 3.3: Transformers model architecture, [50]

feature vector x the general attention computation, also called self-attention, can be expressed as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.3}$$

where $Q = x \cdot W_Q$, $K = x \cdot W_K$ and $V = x \cdot W_V$. To be more precise, the Query matrix Q encodes what to look for in the Key matrix K that represents a set of keys associated with potential results. Meanwhile, the Value matrix, represented by V, signifies the values of the potential results. A triplet (Q, K, V) is called an attention head. The attention setup can benefit from deploying multiple attention heads because it allows to the jointly attend at different representation subspaces from different positions.

$$\text{MultiHeadAttention}(Q, K, V) = Concat(head_1, ..., head_N) \tag{3.4}$$

where each *head* is computes as 3.3. Cross-attention operates on two distinct feature vectors, namely x_1 and x_2, with the objective of quantifying their degree of similarity. To achieve this, the key and values matrices are calculated using the feature representation of one vector, while the query matrix is computed using the other vector.

3.4 Stable Diffusion 1.5

Stable Diffusion 1.5 is generative diffusion model developed by Rombach et al.[38], in particular is a Latent Diffusion Model (LDM). This particular type of model employs a Variational Autoencoder ([23]), with the Encoder, ε, at one end of the architecture and the Decoder, $\mathcal{D}$ at the other end, to switch between the image pixel space and a latent space where the image information are compressed. Specifically, given an image $x \in \mathbb{R}^{H \times W \times 3}$ in the RGB space, applying the encoder, $z = \varepsilon(x)$, turn x into a latent $z \in \mathbb{R}^{h \times w \times c}$, while the decoder reconstructs the original image from the latent, $\widetilde{x} = \mathcal{D}(z)$. Both the diffusion process and the denoising step are performed inside the latent space, in particular the denoising step in performed by an underlying UNet backbone compose of convolutional and cross-attention layers. The presence of the cross-attention layers allows to condition the denoising step through an external input,in particular the input conditions the K, V and Q parameters of the cross-attention. Some of most common form of conditioning input are: text, semantic map and other images for image-to-image translation. In order for the conditioning input to condition the cross-attention parameters it needs to be ported into a latent, to do so Stable Diffusion 1.5 uses an specific encoder τ_Θ. The whole architecture of SD 1.5 is shown in 3.4.

A major drawback of Stable Diffusion 1.5 is the fact that exploits CLIP [34] to generate the vector representation of the textual input, in fact despite CLIP being an efficient method to extract the textual features it is limited by the amount of token that can take in input. If a textual prompt exceeds

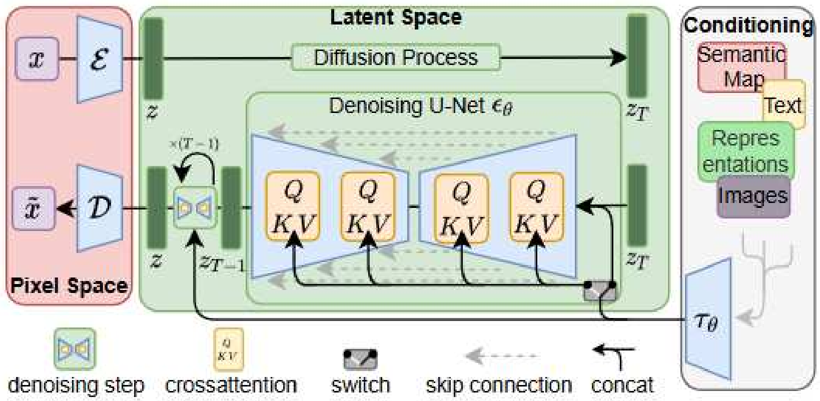


Figure 3.4: Stable Diffusion 1.5 architecture, [38]

77 tokens, it will be truncated by CLIP, thus potentially losing important information useful for a correct image generation.

3.5 Large Language Models

Large Language Models (LLMs) are an attention-based architecture that are able to understand and process human language inputs. The fundamental working unit of an LLMs are tokens, the task of transforming text into tokens is carried out by tokenizers, depending on their granularity tokens can represent whole words ([48]) or sub-words ([43]). Despite their shared conceptual basis, there is no uniform framework. Some of the most prevalent models employ a standard AutoEncoder setup [10], while others utilise a causal decoder architecture or prefix decoder architecture. The causal decoder approach employs a causal attention mask to ensure that each token can only attend to itself or the preceding tokens. In the case of tokens derived from textual inputs, the causality of the masking follows the syntactic ordering of the text, thus proceeding from left to right. The causal decoder approach is exemplified by the GPT family models [33][31], which are among the most well-known examples of this type. Meanwhile the prefix decoder setup drops the constrains of language syntax allowing for a bidirectional attention masking on the prefix token and a unidirectional masking on the newly generated token. A visual representation of the attention patterns associated with each architectural approach is shown in Figure 3.5. In order for an LLM to be

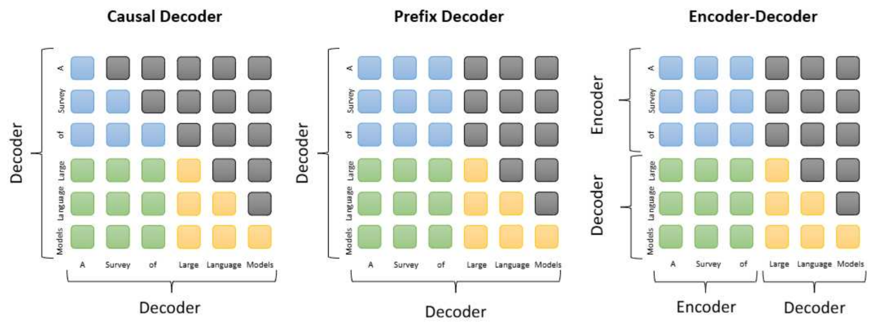


Figure 3.5: the blue tokens represents the attention between prefix tokens,the green tokens indicates the attention between prefix and target tokens, the yellow tokens the attention attention between target tokens and the grey tokens the and masked attention pattern, [57].

able to understand natural language, it is essential that it be exposed to the largest possible amount of data containing the greatest amount of information, and to do this, very large text corpora are used. A corpus is a collection of texts selected and organised usually by similarity. To ensure satisfactory performance, two types of corpus are usually exploited: a more extensive and general corpus consisting of information coming from books [15], conversation [3] and webpages, and a more specialised task including knowledge coming from scientific text and code [2].

3.6 LLama 3.1 8b

LLama 3.1 is causal decoder LLM family of models developed by Abhimanyu Dubey et al. [1] at Meta and publically released as open-source. The whole LLama 3.1 series of model is composed of three different versions with increasing numbers of parameters. The smallest version has 8 billion parameters, the intermediate one has 70 billion of parameters while the largest one has 405 parameters. Each version is characterized by an 128,000 tokens window and from a native support for reasoning, coding, multilingual, and tool tasks. A grid showing the performance of all the LLama3.1 series models is represented in Figure 3.6. Due to the considerable complexity and the substantial number of parameters involved, the models were trained in an ad-hoc computing facility at Meta, utilising 16,000 Nvidia H100 GPUs for the largest

Category	Benchmark	Llama 3 8B	Gemma 2 9B	Mistral 7B	Llama 3 70B	Mixtral 8x22B	GPT 3.5 Turbo	Llama 3 405B	Nemotron 4 340B	GPT-4 (0125)	GPT-4o	Claude 3.5 Sonnet
General	MMLU (5-shot)	69.4	**72.3**	61.1	**83.6**	76.9	70.7	87.3	82.6	85.1	89.1	**89.9**
	MMLU (0-shot, CoT)	**73.0**	72.3△	60.5	**86.0**	79.9	69.8	88.6	78.7◁	85.4	**88.7**	88.3
	MMLU-Pro (5-shot, CoT)	**48.3**	–	36.9	**66.4**	56.3	49.2	73.3	62.7	64.8	74.0	**77.0**
	IFEval	**80.4**	73.6	57.6	**87.5**	72.7	69.9	**88.6**	85.1	84.3	85.6	88.0
Code	HumanEval (0-shot)	**72.6**	54.3	40.2	**80.5**	75.6	68.0	89.0	73.2	86.6	90.2	**92.0**
	MBPP EvalPlus (0-shot)	**72.8**	71.7	49.5	**86.0**	78.6	82.0	88.6	72.8	83.6	87.8	**90.5**
Math	GSM8K (8-shot, CoT)	**84.5**	76.7	53.2	**95.1**	88.2	81.6	**96.8**	92.3◇	94.2	96.1	96.4◇
	MATH (0-shot, CoT)	**51.9**	44.3	13.0	**68.0**	54.1	43.1	73.8	41.1	64.5	**76.6**	71.1
Reasoning	ARC Challenge (0-shot)	83.4	**87.6**	74.2	**94.8**	88.7	83.7	**96.9**	94.6	96.4	96.7	96.7
	GPQA (0-shot, CoT)	32.8	–	28.8	**46.7**	33.3	30.8	51.1	–	41.4	53.6	**59.4**
Tool use	BFCL	**76.1**	–	60.4	84.8	–	**85.9**	88.5	86.5	88.3	80.5	**90.2**
	Nexus	**38.5**	30.0	24.7	**56.7**	48.5	37.2	**58.7**	–	50.3	56.1	45.7
Long context	ZeroSCROLLS/QuALITY	81.0	–	–	90.5	–	–	**95.2**	–	**95.2**	90.5	90.5
	InfiniteBench/En.MC	65.1	–	–	78.2	–	–	**83.4**	–	72.1	82.5	–
	NIH/Multi-needle	98.8	–	–	97.5	–	–	98.1	–	**100.0**	**100.0**	90.8
Multilingual	MGSM (0-shot, CoT)	**68.9**	53.2	29.9	**86.9**	71.1	51.4	**91.6**	–	85.9	90.5	**91.6**

Figure 3.6: Performance of the LLama3.1 series across several benchmark, Image by [1]

model. In terms of the architectural design, the smallest model is composed of 32 layers, a token cardinality of 4,096 dimensions and 32 attention heads. It also employs the SwiGLU activation function ([45]) and a peak learning rate of 3×10^{-4}. A gloabal overwiev of the archictecure is presented in 3.7

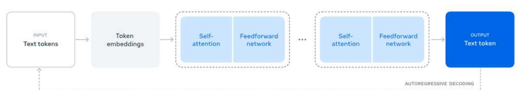


Figure 3.7: Architecture schematic of the LLama3.1 herd of models. Image taken from [1]

3.7 Graph Convolutional Network

In recent years, Graph Convolutional Networks have become increasingly popular due to their ability to exploit and extract information from graph-structured data formats. Most notably, they have found application in fields like recommender system [55], computer vision [5], protein interaction [13], chemistry [36], traffic prediction [21] and social analysis [25]. They were first introduced by Kipf and Welling [24] with the aim of exploiting the message-passing technique on graphs; in particular, each pair of nodes connected by an edge will exchange its information, represented by the node feature representation, to combine it with the other nodes' information by an aggregation function. An example of the message-passing technique is shown in Figure 3.8. Given an undirected graph $G = (V, E)$ with N nodes $v_i \in V$ and edges

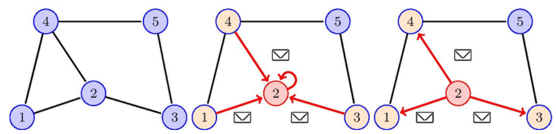


Figure 3.8: Example of the message passing technique. Image from [4]

$(v_i, v_j) \in E$, its adjacency matrix $A \in \mathbb{R}^{N \times N}$, the degree matrix $D_{ii} = \sum_j A_{ij}$ and h the input feature representation, the layer-wise formulation of the convolutional layer can be expressed as:

$$h^{'} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}hW\right) \tag{3.5}$$

where $\widetilde{A} = A + I_N$ is the adjacency matrix with the self-loops added, $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$, W is a layer-specific weight matrix and $\sigma(\cdot)$ represents the $ReLU(\cdot) = max(0, \cdot)$ activation function. Another approach, developed by Morris et al.[32], aims to improve the graph classification task by proposing to perform the message passing between neighbouring subgraphs instead of neighbouring nodes, in order to better capture structural information that is not visible at the node level. This approach creates a de facto k-hierarchical graph neural network (GNN), where k represents the hierarchical degree as shown in Figure 3.9. At each hierarchical level, each node feature, x is aggregated according to the following layer-wise expression:

$$x_i^{'} = W_1 x_i + W_2 \sum_{j \in N(i)} e_{i,j} \cdot x_j \tag{3.6}$$

where e_{ij} represent the weight of the edge between the $i-th$ node and the $j-th$ node, W_1 and W_2 are two learnable weight matrix, one to parametrize the self-loops and one to parametrize every other connection.

3.8 Graph Attention Network

Inspired by the recent work on transformers architectures [50], Graph Attention Networks introduces a novel approach to node classification using a self-attending mechanism capable of attending to each node's neighbours.Veličković

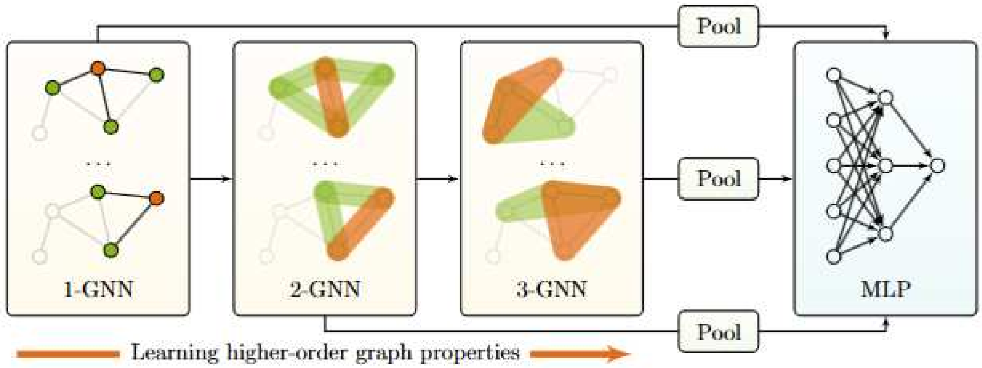


Figure 3.9: Example of 3-GNN for graph classification, the information extracted by the hierarchical framework is than used by the MLP to classify the graph. Image from [32]

et al. [51] developed a framework that employs the concatenation of multiple attention heads to obtain the feature representation of each successive layer. In particular, given an initial feature representation $h = \{\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_N}\}, \overrightarrow{h_i} \in \mathbb{R}^F$ with F as the feature dimension and $h^i = \{\overrightarrow{h^i_1}, \overrightarrow{h^i_2}, ..., \overrightarrow{h^i_N}\}, \overrightarrow{h^i_i} \in \mathbb{R}^{F'}$, a linear transformation parametrized by a weight matrix $W \in \mathbb{R}^{F \times F'}$ is applied at every node to increase their expressive power. A shared weight-attention mechanism, a, is applied to obtain the weights, e_{ij}, of each edge between the $i-th$ node and its $j-th$ immediate neighbours $N(i)$.

$$e_{ij} = a(W\overrightarrow{h}_i, W\overrightarrow{h}_j) \tag{3.7}$$

Veličković et al.[51] used as attention mechanisms a single-layer fully connected feed-forward neural network parametrized by a weight vector $\overrightarrow{a} \in \mathbb{R}^{2F'}$. Using this setup the attention coefficients computed by the attention mechanism can be expressed as:

$$\alpha_{ij} = \frac{exp(\text{LeakyRelu}(\overrightarrow{a^T}[\text{W}\overrightarrow{h_i}||\text{W}\overrightarrow{h_j}]))}{\sum_{k \in N_i} exp(\text{LeakyRelu}(\overrightarrow{a^T}[\text{W}\overrightarrow{h_i}||\text{W}\overrightarrow{h_k}]))} \tag{3.8}$$

thus, the feature representation for the $i-th$ layer and the $k-th$ attention head is given by:

$$\overrightarrow{h}'_i = \overset{K}{\underset{k=1}{\Big\|}} \sigma \left(\sum_{j \in N(i)} \alpha_{ij}^k W^k \overrightarrow{h}_j \right) \tag{3.9}$$

where α_{ij}^k is the attention coefficient for $k-th$ attention head and $\|$ indicates the concatenation operation. For the last layer the concatenation is dropped and replaced with an averaging operation.

$$\overrightarrow{h}'_i = \sigma \left(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in N(i)} \alpha_{ij}^k W^k \overrightarrow{h}_j \right) \tag{3.10}$$

Shaked Brody et al.[44] found that the approach developed by Veličković et al [51], while effective, did not compute a dynamic form of attention, in fact the query node of the attention mechanism was left unconditioned. This leads to a static form of attention that heavily limits the potentiality of the method. To solve this problem, Shaked Brody et al. propose two simple corrections: apply a after LeakyRelu and employ W after the concatenation. Following these changes the attention coefficients can be computed as:

$$\alpha_{ij} = \frac{exp(\overrightarrow{a^T}\text{LeakyRelu}(\text{W}[\overrightarrow{h_i}||\overrightarrow{h_j}]))}{\sum_{k \in N_i} exp(\overrightarrow{a^T}\text{LeakyRelu}(\text{W}[\overrightarrow{h_i}||\overrightarrow{h_k}]))} \tag{3.11}$$

By employing these modifications each node can process a dynamic attention for each node representation.

Chapter 4

Method

4.1 Adapter pipeline

Haven explored the State-of-the-Art methods and the background knowledge that is required to fully understand the proposed adapter, this section will introduce the general architecture of the adapter pipeline. The first step of the process is the extraction of the Scene Graph (SG) from the user text input; this is done inside the Scene Graph Parser module. In this module a Large Language Model (LLM) is used to identify the entities present in the text prompt and to obtain a list of triplets representing the Scene Graph. Each node of the Scene Graph is associated with an extracted entity or relationship. However, in order to obtain a Scene Graph that correctly describes the text prompt, all the co-references between the entities must be resolved. The term co-reference is used when two different words are used to indicate the same entity, if this task is not properly taken care of, the set of extracted entities could be redundant; this will lead to an incorrect amount of nodes in the Scene Graph.

```
 Example of Co-Reference:
John found a kitten outside, and he decided to bring it home.

The words "John" and "he" refers to the same entity thus are
   co-reference
```

To avoid this pitfall, the Parser Module uses the LLM to resolve the co-references, ensuring that all mentions are correctly matched to each instance. The Parser module outputs an index-mapped entity list and an adjacency list representing the Scene Graph, an example of said format can be seen in

```
      caption : "The image shows a room with a pile of rubble and debris, including a pile of wood, a red object, and a
                 pile of dirt. The room appears to be in a state of disrepair, with the rubble occupying a significant
                 portion of the space. The red object is situated in the middle of the room, surrounded by the debris, while
                 the pile of wood is located on the left side of the room. The pile of dirt is located on the right side of
                 the room, adjacent to the pile of wood."

     Entities : { "room": 0, "pile of rubble": 1, "debris": 2, "wood": 3, "red object": 4, "dirt": 5, "left side of room": 6,
                    "right side of room": 7, "middle of room": 8 }

adjacency list : [[0, 9], [9, 1], [1, 10], [10, 0], [0, 9], [9, 2], [2, 10], [10, 0], [0, 9], [9, 3], [3, 10], [10, 0], [0, 9
                 [9, 4], [4, 10], [10, 0], [0, 9], [9, 5], [5, 10], [10, 0], [3, 11], [11, 6], [5, 11], [11, 7], [4, 10], [10 8]]
```

Figure 4.1: SG Parser example output

Figure 4.1.

The second step in the pipeline is the encoding of the graph, that is done by the Graph Embedding module. This part of the pipeline takes advantage of the CLIP model [34] to obtain a vector representation of the features of each node. It is worth noting that encoding each node, rather than directly at the text prompt, avoids the 77 token bottleneck (see Chapter 3.4)that Stable Diffusion 1.5 suffers from. The Graph Convolutional Network (GCN) [24] instead, has its own weight updated with each backward pass. The main objective of the GCN is to exploit its convolutional layer to exchange information between each node of the graph. The core concept behind the GCN is that each node of the graph "shares" its embedded Scene and becomes "aware" of the other nodes so that the final image can correctly represent the user's request. After the last layer of the GCN, a new embedding is obtained for each node. The final step of the adapter is the injection of the node features into the Stable Diffusion 1.5 [38] U-Net. This is done by passing the node feature into multiple cross-attention layers, one for each layer of the U-Net. The global structure of the pipeline can be seen in Figure 4.2.

4.1.1 Scene Graph Parser

The Scene Graph Parser has the key task of extracting information from the user text input and crafting the Entity List and the Scene Graph. As can be seen in Figure 4.3, it is composed by two main elements: the Large Language Model Parser and the Post-Processing Module. The Large Language Model Parser si equipped with two prompts: a Main Prompt 8.1.1 and a Secondary Prompt 8.1.2, the latter is used when the first one produces a incomplete or badly-formatted output. The Post-Processing Module is used to identify errors in the raw ouput generated by the LLM and, when necessary, to recall the LLM itself with the proper prompt. In case of errors in the SG or in the

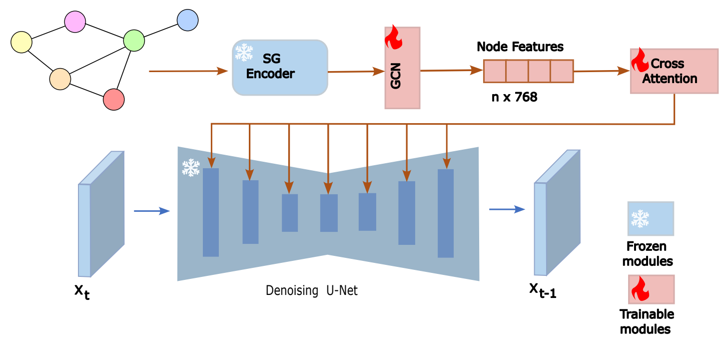


Figure 4.2: Global structure of the SG adapter, original image by [54]

entity list, when generating the raw output with the Secondary Prompt, the Post-Processing module acts a last fail-safe measure correcting the persistent errors. Finally the Post-Processing Module outputs the Adjacency List of the Scene Graph along with the Entity List, the Relationship List and the Scene Graph itself.

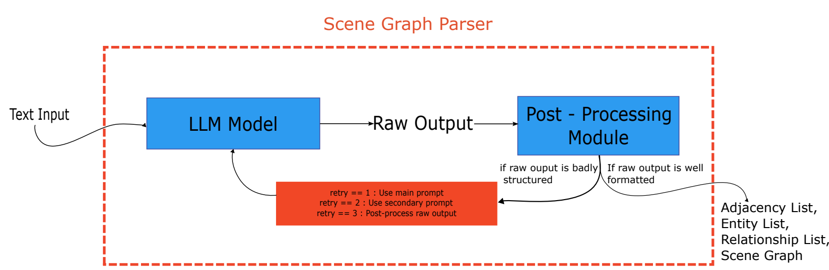


Figure 4.3: Structure of the Scene Graph Parser

Large Language Model Parser

This sub-module is composed of a Large Language Model (LLM), specifically LLama 3.1 8b Instruct [1]. This model was chosen for a number of reasons, including its open source nature, its compact number of weights, and its high level of reasoning capability. Being an "Instruct" type of model, it supports chat roles for structuring the prompt, in particular supports four roles:

- System: To sets the context in which to interact with the model;
- User: To represent human interaction with the model;
- Ipython: To mark messages with the output of a tool call when sent back to the model;
- Assistant: To represent the response generated by the model, based on the context provided in the System, Ipython and User prompts.

After performing some tests to establish the hyperparameters, the result showed the best efficacy with the Temperature parameter set to 0.4 and the Top_p parameter set to 0.9. In order to properly extract the Scene Graph, multiple Prompt Engineering strategy have been leveraged, in particular a combination of Chain-Of-Thought (CoT) and Few-Shot Learning. After several tests and experiments, it was found that captions containing multiple instances or long paragraphs of text produced incomplete output, in particular they exceeded the maximum token length of the LLM without completing the SG or the entity list (EL). To solve this problem, a two-prompts approach was implemented: a more precise Main Prompt (8.1.1), with the aim of extracting and numerating each Entity and Relationship, and a Secondary Prompt (8.1.2), more focused into grouping Entities into a single one, used when after two attempts the Main Prompt still can't produce a complete output. Lastly, in the Main Prompt, the co-reference resolution is performed by the assigned LLM model, in particular the model is prompted with using a numerical progressive subscript (####_1,####_2, ...) in order to correctly disambiguate each instance.

Post-Processing Module

Although in the precedent sub-module several tests were performed in order to minimize variability of the LLM output, some captions resulted too semantically complex for the Parser to Handle, producing incorrectly formatted output. To try tacking this errors at first the LLM model is called again with the Main Prompt, then if the errors persist the Secondary Prompt is used. This mechanism is put in place in order to maximize the amount of information that can be preserved while minimizing the need to enforce automatic corrections to the raw output. In the event that even when adopting the Secondary Prompt the raw output is incorrectly formatted or present some inconsistencies the Post-Processing modules enforces some automatic corrections. The most common inconsistencies were in the length of the Scene Graph triplets and a mismatch between the entities in the Entity List and those in the Scene Graph. To address these issues, the Post-Processing block standardises the length of all triplets with a length greater than three to three. During this step, all the triplets with length two (tuples) are left untouched and maintain their original dimension. This is done to improve the mapping the possession and attribute type of relationship as they carry important information for the image generation but not always are correctly expressed into triplets by the LLM.

```
Caption: This image displays: A white dog wearing a fur hat
   with a red star on it. The dog is looking at the camera.
   There is a green towel on the floor in the background. The
    floor is covered in black carpet. The background wall is
   blurry and unfocused. The image is a photograph."

Example of triplets with legth 2 (tuple) representing
   attributes:
["background wall", "is blurry and unfocused"]
```

Subsequently, the entity coming from the Entity List and the ones coming from the Scene Graph are checked to see if they match, if an entity from the Entity List does not match any entity from the Scene Graph than the entity is discarded. If an Entity from the Scene Graph does not have a corresponding match with any of the entity present inside the Entity List, the whole Entity List is discarded, a new one will be built using the entity of the Scene Graph. A schema of the operation performed in the Post-Processing Module is reported in Figure 4.4

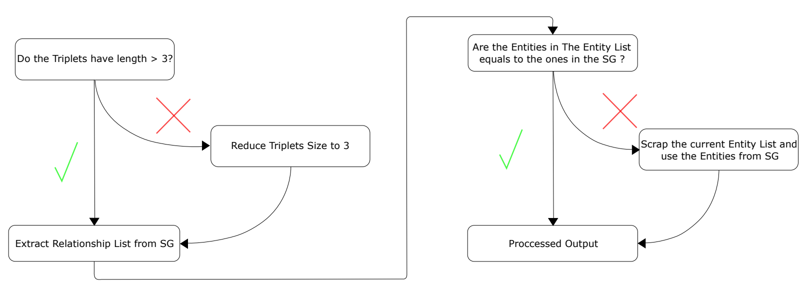


Figure 4.4: Post-Processing Module Schematics

The way the Relationship are treated is also accounted for, after having checked the length of the each triplet a Relationship List is extracted from all the Relationship that have a length of three. As with Entities, the same Relationship is numbered using a progressive numerical index in order to distinguish different instances of the same Relationship. Unlike the Entities, the enumeration is not performed by the LLM model. The final step of this module associates with every node (both Entities and Relationship) an integer index so that the Scene Graph can be rewritten in the form of a numerical adjacency list.

4.1.2 Graph Encoder

Once that the information contained in the text prompt has been successfully turned into a well-formatted Scene Graph, it is ready to be transformed into a numerical representation. The graph encoder module transforms all textual features associated with each node into a numerical embedding. In order to produce said embedding a CLIP Model is exploited [34], in particular the "ViT-L/14" version. This particular version is the one used in the original Stable Stable Diffusion v1.5 setup and thus is compatible with it, in particular it returns n tokens each with a length of 768, where n stands for the number of nodes in the graph.

4.1.3 Graph Convolutional Network

In order to exploit the information embedded into the relationship between the each pair of nodes a Graph Convolutional Network [24] is used. After a series of preliminary experiments testing different configurations of convolutional layers and number of layers, the final architecture ended up using the convolution type introduced by Kipf and Welling 3.7 with two layers. The convolutional layers adopted does not change the feature cardinality. Together with those layers, the GCN adapter is composed of a linear layer in input, mapping from $\mathbb{R}^d$ to $\mathbb{R}^{2d}$ where d=768, and another final linear in output, applied to reverse the initial transformation. Regarding the activation function, the GELU [17] was used after each linear and convolutional layer.

4.1.4 Node Padding

The results of a series of experiments demonstrated that the presence of graph scenes that contained a limited number of nodes posed a challenge for the model. In particular, the images produced by scene graphs of this type were unable to reproduce the main subjects of the image or place them in the context described by the textual prompt. Through the execution of a series of tests, it was empirically determined that for all graphs with a number of nodes below a certain threshold, additional nodes were introduced into the graph with null associated semantic content (' '). The number of nodes below which to insert this form of padding is, in fact, to be regarded as a hyper-parameter. One potential explanation for this phenomenon is that the cross-attention mechanism leverages the weights associated with these null nodes to extract information introduced during the training phase, and subsequently utilizes the information encoded in the weights during the inference phase.

4.1.5 Cross-Attention Layers

The final step of the Adapter is the injection of the output of the CGN into the Stable Diffusion v1.5 U-Net. This is done by adding a layer of Cross-Attention for every layer of the U-Net, in particular the Cross-Attention operator will be performed amongst the graph features c_g and the feature Z coming from the U-Net. Let's now define Q = ZW_q, K = $c_g W_k$ and V = $c_g W_V$ as parameters for the cross-attention function.

$$Z = Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.1}$$

where d_k is a normalization factor. Since the weights from the U-Net are being kept frozen, the weight matrix W_q won't be updated during training, thus leaving the matrices W_v and W_k as the only learnable parameters.

Chapter 5

Dataset

5.1 Dataset

To train and evaluate the proposed architecture, a new custom dataset was created, combining caption and image pairs from different sources. Following the results of Singla et al. [47], which showed how a dataset with more complete and descriptive captioning improves the performance of generative models by enhancing their expressiveness, the dataset they developed - PixelProse - was chosen as the first data source.

5.1.1 PixelProse

PixelProse [47] is a dataset consisting of over 16 million synthetically generated captions. It includes image-caption pairs from multiple datasets:

- CC12M (9.1M) [7]
- CommonPool (6.5M) [14]
- RedCaps (1.3M) [11]

To obtain a more dense and descriptive representation of the image, Singla et al., recaptioned each image textual description with Gemini 1.0 Pro Vision.

5.2 Data Filtering

PixelProse incorporates a multitude of images and captions from a variety of sources. Consequently, the collection is characterised by a considerable degree of heterogeneity, necessitating a rigorous filtering process to ensure the removal of irrelevant or superfluous elements. In particular, the following filters were applied:

- For PixelProse:
  - width and lenght >512
  - watermark class = 1 (images without watermark)
  - aesthetics score >4.5 for the images
  - aspect ratio <= 1.5 (3:2/2:3 or 5:3/3:5)
  - max 3 ’\n’ in the caption (max three paragraph)
  - only characters of the english alphabet in the caption
  - no special characters
  - the image is not of a slide from a presentation
  - max 79 ”,”

In regard to the aesthetic score, Pixelprose employs LAION-Aesthetics Predictor V2 [41], which has been fine-tuned on a high-quality image dataset. An example of images associated with an aesthetics score is shown in Figure 5.1.

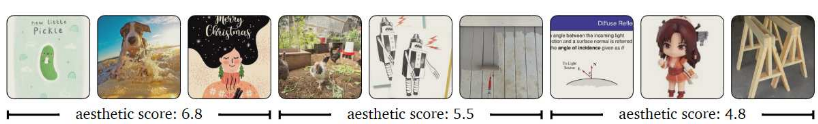


Figure 5.1: Example of image quality with different aesthetics score

5.3 Data Sampling

Initially, the starting datasets is very large, containing 12 million for PixelProse. After the filtering process, there are 2.406.514 records left for PixelProse so with a target of 150.000 records, it is necessary to do some sampling.

Taking into account the large number of samples available, in order to try to obtain as complete a representation of the information as possible, and to make the execution times as efficient as possible, random sampling was chosen as strategy. Among the sampled entries, a validation split of 5000 images was selected.

5.4 Dataset Analysis

In this section some statistics are reported on the dataset that was used for training. The analysis aims to provide information on the structure of the dataset, including the distribution of key variables, the presence of missing values, and any significant patterns or anomalies.

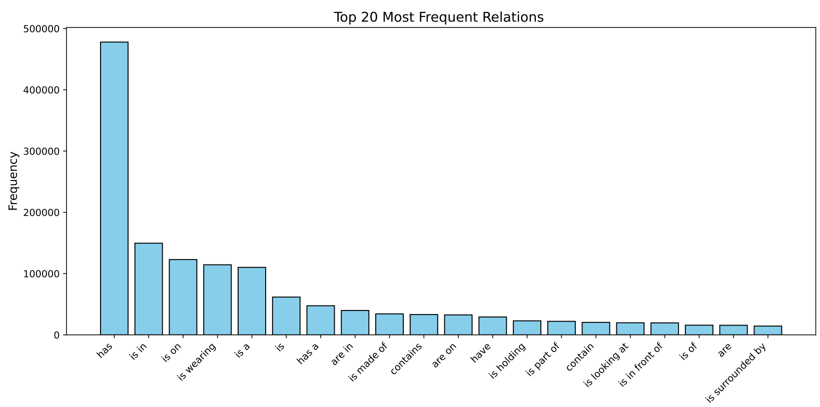


Figure 5.2: Top 20 Most Frequent Relationships

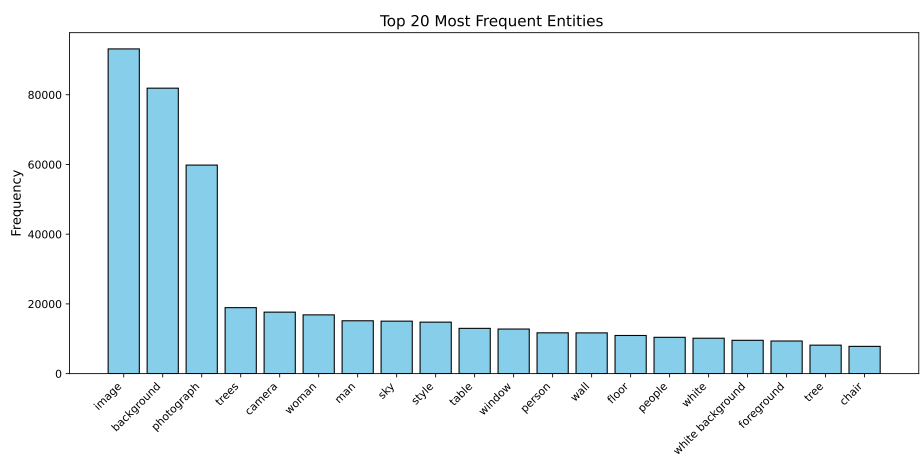


Figure 5.3: Top 20 Most Frequent Entities

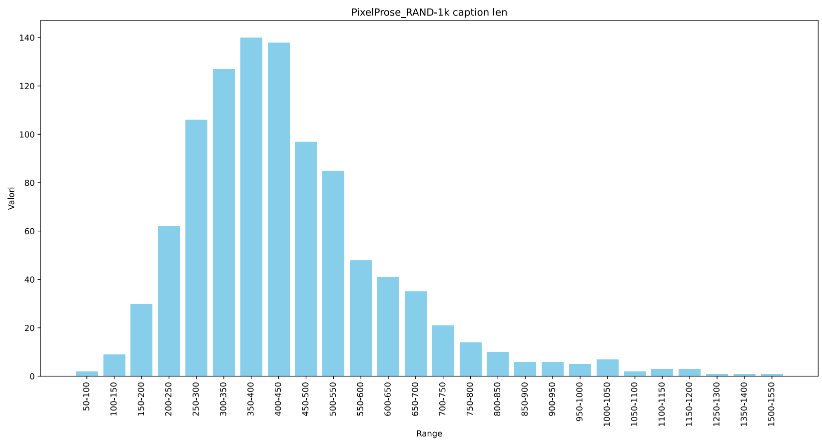


Figure 5.4: Caption length distribution

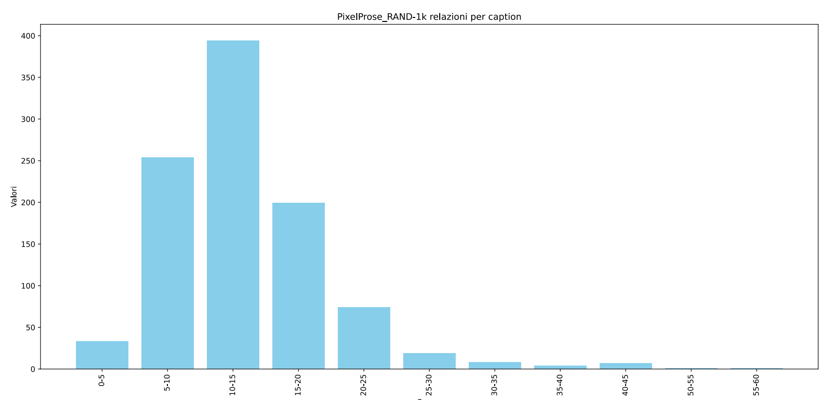


Figure 5.5: Relationships per caption

As can be seen from Figure 5.2, the words "image" and "background appear" to be among the most frequent. This is due to the fact that most of PixelProse's captioning starts with "This image shows ...", whereas the expression "In the background ...." is used for the description of the background. Being a very descriptive and richly detailed dataset, the graphs generated by PixelProse tend to have many nodes, often with multiple entities. An example of graphs generated from PixelProse caption can be seen in Figures 6.4, 6.5, 6.8, 6.9. 6.9, 6.13. Statistical analysis has shown that on average a PixelProse caption generates a caption with 40 nodes between entities and relationships.

Chapter 6

Results

6.1 Parser Benchmark

In order to properly assess the reasoning capability of the SG-Parser, an ad hoc human-generated caption benchmark is introduced. To evaluate the Parser in a thorough manner, different categories of captions were included: in particular, the captions in the benchmark are divided according to their length and number of relations. The threshold values for caption length and the spatial relationship number for each section of the benchmark were determined by observing the caption length and relationships per caption distribution in PixelProse. In total, four distinct caption categories were generated, each consisting of 10 captions. To further evaluate the model's performance, each category was divided into two subcategories based on the number of entity instances present in the textual prompt. An overview of the composition of the benchmark is shown in Table 6.1, while an example of the caption benchmark is reported in Appendix 8.2.

	Caption Lenght	
Number of Relationships	Captions with lenght <150 and 15 relationships	Captions with lenght <150 and 5 relationships
	Captions with 150<=lenght<=300 and 15 relationships	Captions woth 150<=lenght<=300 and 5 relationships

Table 6.1: Categories of the Parser Benchmark

Following the introduction of the benchmark employed to evaluate the parser, the ensuing section presents the results obtained.

- Caption length $<$ 150 characters, 3 spatial relationships, one instance per entity
  - Parsed successfully at the first time with Main Prompt: 10
  - Parsed successfully at the second time with Main Prompt: 0
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length $<$ 150 characters,3 spatial relationships, max 3 instance per entity
  - Parsed successfully at the first time with Main Prompt: 9
  - Parsed successfully at the second time with Main Prompt: 1
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length $<$ 150 characters,5 spatial relationships, one instance per entity
  - Parsed successfully at the first time with Main Prompt: 9
  - Parsed successfully at the second time with Main Prompt: 1
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length $<$ 150 characters,3 spatial relationships, max 3 instance per entity
  - Parsed successfully at the first time with Main Prompt: 9
  - Parsed successfully at the second time with Main Prompt: 0
  - Parsed successfully with the Secondary Prompt: 1
  - Both prompts failed, correction to the raw output were enforced: 0

- Caption length 150−300 characters,3 spatial relationships, one instance per entity
  - Parsed successfully at the first time with Main Prompt: 10
  - Parsed successfully at the second time with Main Prompt: 0
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length 150−300 characters,3 spatial relationships, max 3 instance per entity
  - Parsed successfully at the first time with Main Prompt: 10
  - Parsed successfully at the second time with Main Prompt: 0
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length 150−300 characters,5 spatial relationships, one instance per entity
  - Parsed successfully at the first time with Main Prompt: 9
  - Parsed successfully at the second time with Main Prompt:1
  - Parsed successfully with the Secondary Prompt: 0
  - Both prompts failed, correction to the raw output were enforced: 0
- Caption length 150−300 characters,5 spatial relationships, max 3 instance per entity
  - Parsed successfully at the first time with Main Prompt: 8
  - Parsed successfully at the second time with Main Prompt: 0
  - Parsed successfully with the Secondary Prompt: 1
  - Both prompts failed, correction to the raw output were enforced: 1

6.2 GCN Comparison

Before training the entire adapter pipeline as a whole, a set of experiments were performed to find the best GCN configuration. The analysis included three main factors: the type of convolution layer, the number of convolutional layers and the amount of training images. The candidate types of convolutional layers are: the convolution operator by Kipf and Welling (3.5), the one developed by Morris et al. (3.6) and the attention framework introduced by Shaked Brody (3.10). For the sake of simplicity and readablity, from now on these operators will be addressed as GCNConv, GraphConv and GATv2Conv. To reduce the computational complexity and evaluate the configuration purely on the GCN performance, during this analysis, the weights of Stable Diffusion 1.5 are kept frozen. In order to determine the number of steps at which each configuration performed best, an upper limit of 5000 denoising steps was set, and the steps at which the quality of the generated images was best were observed. The steps for each configuration can be found in Table 6.2. Each configuration was benchmarked using TIFA and FID metric.

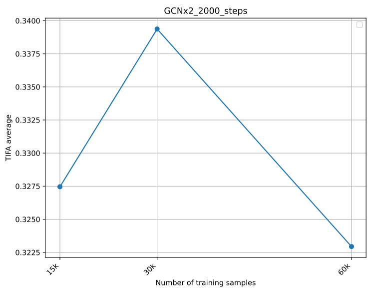
conv layer type	15k images	30k images	60k images
GCNConv-2layers	2000 steps	2000 steps	2,000 steps
GCNConv-4layers	2500 steps	2500 steps	2500 steps
GraphConv-2layers	4500 steps	4500 steps	4500 steps
GraphConv-4layers	4500 steps	4500 steps	4500 steps
GATv2Conv-2layers	2000	2000	2000

Table 6.2: Training steps for each configuration tested, the values were determined experimentally

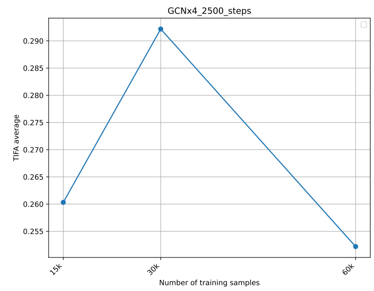
6.2.1 GCN Configuration Analysis with TIFA

Having established the number of iterations for each configuration, this section aims to demonstrate the impact of scaling on each configuration as the quantity of training images available increases.

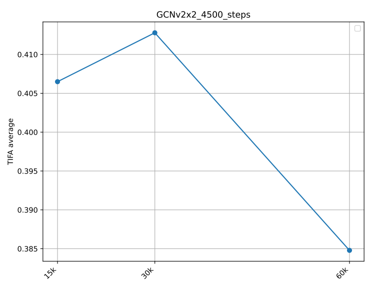
To facilitate comparison, the values obtained with TIFA for each category are in Table 6.3 for all configurations tested.



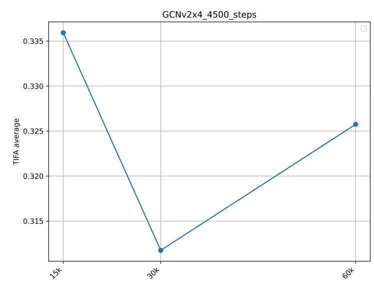
(a) GCN with two convolutional layers



(b) GCN with four convolutional layers



(a) GraphConv with two convolutional layers



(b) GraphConv with four layer

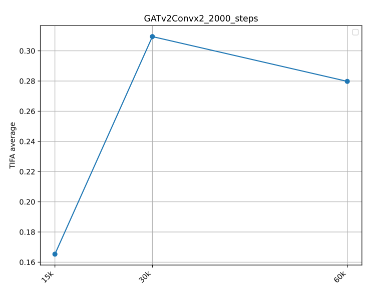


Figure 6.3: GATv2Conv with two layer

15K training samples													
conv layer type	animal/human	object	location	activity	color	spatial	attribute	food	counting	material	shape	other	average	stdev
GCNConv-2layers	0.385	0.270	0.305	0.287	0.317	0.487	0.429	0.234	0.288	0.468	0.348	0.338	0.327	0.245
GCNConv-4layers	0.278	0.213	0.260	0.211	0.262	0.418	0.361	0.179	0.287	0.349	0.290	0.284	0.260	0.225
GraphConv-2layers	0.538	0.350	0.411	0.365	0.389	0.487	0.543	0.338	0.354	0.493	0.377	0.358	0.406	0.257
GraphConv-4layers	0.436	0.281	0.348	0.244	0.285	0.441	0.512	0.256	0.315	0.512	0.420	0.303	0.336	0.245
GATv2Conv-2layers	0.140	0.128	0.087	0.147	0.257	0.347	0.214	0.094	0.150	0.258	0.478	0.159	0.166	0.169
30K trainig images														
conv layer type	animal/human	object	location	activity	color	spatial	attribute	food	counting	material	shape	other	average	stdev
GCNConv-2layers	0.434	0.275	0.335	0.275	0.305	0.450	0.485	0.247	0.318	0.478	0.290	0.289	0.340	0.248
GCNConv-4layers	0.352	0.237	0.243	0.242	0.282	0.445	0.411	0.192	0.275	0.407	0.304	0.245	0.292	0.235
GraphConv-2layers	0.542	0.365	0.406	0.368	0.358	0.494	0.552	0.319	0.359	0.560	0.435	0.378	0.413	0.264
GraphConv-4layers	0.404	0.251	0.322	0.225	0.283	0.427	0.494	0.206	0.269	0.459	0.209	0.234	0.312	0.236
GATv2Conv-2layers	0.409	0.243	0.258	0.220	0.298	0.429	0.473	0.225	0.282	0,388	0.333	0.313	0.309	0.236
60K training images														
conv layer type	animal/human	object	location	activity	color	spatial	attribute	food	counting	material	shape	other	average	stdev
GCNConv-2layers	0.350	0.274	0.323	0.277	0.326	0.478	0.432	0.224	0.309	0.455	0.203	0.313	0.323	0.245
GCNConv-4layers	0.286	0.208	0.236	0.189	0.264	0.397	0.348	0.198	0.239	0.368	0.312	0.204	0.252	0.224
GraphConv-2layers	0.452	0.341	0.420	0.338	0.379	0.484	0.506	0.321	0.357	0.493	0.333	0.338	0.385	0.258
GraphConv-4layers	0.403	0.267	0.334	0.257	0.274	0.441	0.494	0.241	0.309	0.388	0.420	0.343	0.326	0.247
GATv2Conv-2layers	0.364	0.213	0.230	0.202	0.266	0.430	0.444	0.190	0.212	0.359	0.247	0.194	0.280	0.230

Table 6.3: TIFA values for different GCN divided by amount of training samples

From both the plots and from the TIFA values it is noticeable a degradation in performance with a number of training samples greater than 30.000, The GAT configuration with four layers wasn't evaluated with TIFA nor with FID due to already the very poor performance of the 2-layer version.

6.2.2 GCN Configuration Analysis with FID

In order to evaluate the various possible configurations from another perspective, a more well-established and proven metrics was used, the Frechet Inception Distance (FID) 2.1. In particular, this analysis employed a FID version developed with the pytorch library, proposed in [42]. Following the method used by Shen et al., the FID is computed using 5000 images from the COCO validation set [27] and 5000 validation images from the processed PixelProse introduced in 5.1. The results are presented in tabular form below, broken down according to the number of training images.

conv layer type	FID
GCNConv-2layers	84.00
GCNConv-4layers	87.27
GraphConv-2layers	80.94
GraphConv-4layers	87.22
GATv2Conv-2layers	91.15

Table 6.4: FID values with 15K training samples, the lower the better

conv layer type	FID
GCNConv-2layers	80.867
GCNConv-4layers	92.10
GraphConv-2layers	83.51
GraphConv-4layers	91.80
GATv2Conv-2layers	89.24

Table 6.5: FID values with 30K training samples, the lower the better

conv layer type	FID
GCNConv-2layers	91.42
GCNConv-4layers	96.04
GraphConv-2layers	86.67
GraphConv-4layers	93.48
GATv2Conv-2layers	87.56

Table 6.6: FID values with 60K training samples, the lower the better

6.2.3 Qualitative GCN comparison

In this section are reported an example of the image generated by the different GCN versions,the corresponding scene graph is also represented above them. The image are from the validation split of the dataset introduced earlier 5.1. To better visualize the entities and the relationships the nodes are color coded: light blue for the entities, while red for the relationships. In case of multiple instances of a relationships or of an entity the subscript notation described in 4.1.1 is used.

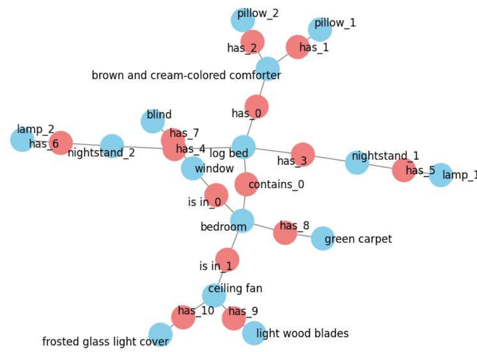


Figure 6.4: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

GCNConv **GraphConv** **GATv2Conv**

A bedroom with a log bed, a ceiling fan, and a window. The bed has a brown and cream-colored comforter with two matching pillows and two matching nightstands with lamps on either side of the bed. The window has a blind and there is a green carpet on the floor. The ceiling fan has light wood blades and a frosted glass light cover.

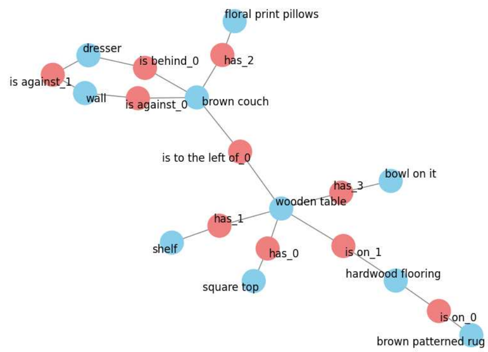


Figure 6.5: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

GCNConv

GraphConv

GATv2Conv

This image displays a wooden table with a square top and a shelf below it. The table is to the left of a brown couch with floral print pillows. The table has a bowl on it. There is a brown patterned rug on the floor, and hardwood flooring underneath it. There is a wooden dresser against the wall behind the couch. There are no people in the image.

6.3 SceneGraph Diffusion Finetuning Setup

Training a model correctly is crucial to achieving the best performance from the model. From the analysis done in the earlier section, the selected GCN of choice ended up being a GCNConv setup with 2 convolutional layer. This choice was made due GCNConv presenting more aesthetic images while still achieving decent results on the TIFA benchmark if confronted against other configuration. Two linear layers, one before and one after the convolutional layer, complete the GCN architecture for the whole adapter training. Having selected the GCN setup, the next step is to fine-tune it together with the Stable Diffusion 1.5 backbone. The proposed architecture was finetuned using 60.000 entries of the dataset described in 5.1 for 10.000 steps, the learning was increased in a linear manner value until it reached the value of 1×10^{-4}, after that it was kept constant, a batch size of 16 was used. Moreover, to remove the effect of a potential bad weight initialization, the GCN was subject to 500 steps of warm-up. The padding described in 4.1.4 was also used with a minimal number of nodes equal to 15.

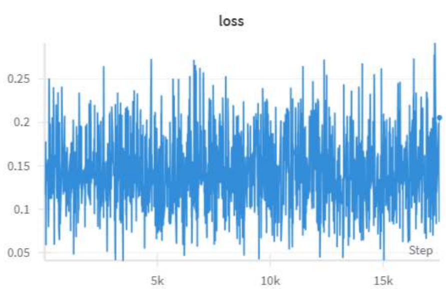


Figure 6.6: Training loss after 40.000 steps, the selected checkpoint was stopped after 10.000 steps

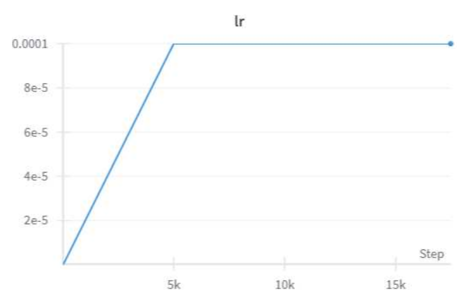


Figure 6.7: Learning rate curve during training

6.3.1 SG Diffusion Qualitative Results on Validation Set

To better compare visually the image generated by the Adapter, in this section are presented some images produced by SG Diffusion alongside images produced by other State-Of-The-Art architectures. The caption are from the validation spli of the earlier introduced 5.1 version of PixelProse.

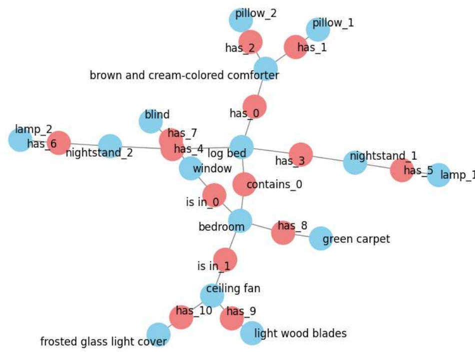
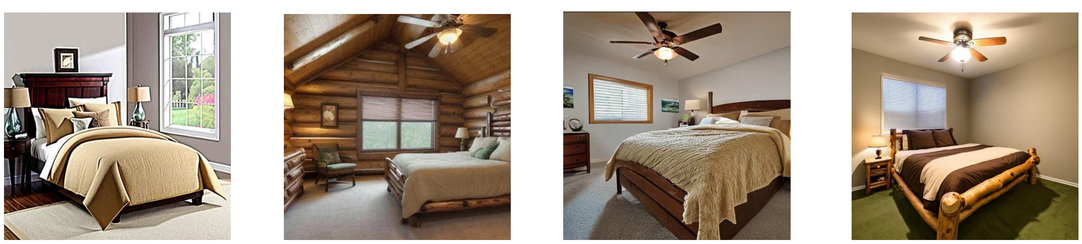


Figure 6.8: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion **SD 1.5** **SD 2.1** **SD 3 (medium)**



This image displays: A bedroom with a log bed, a ceiling fan, and a window. The bed has a brown and cream-colored comforter with two matching pillows and two matching nightstands with lamps on either side of the bed. The window has a blind and there is a green carpet on the floor. The ceiling fan has light wood blades and a frosted glass light cover.

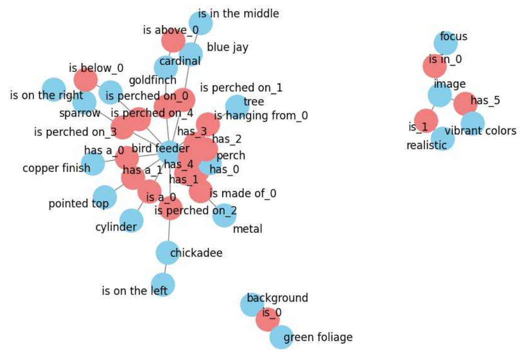
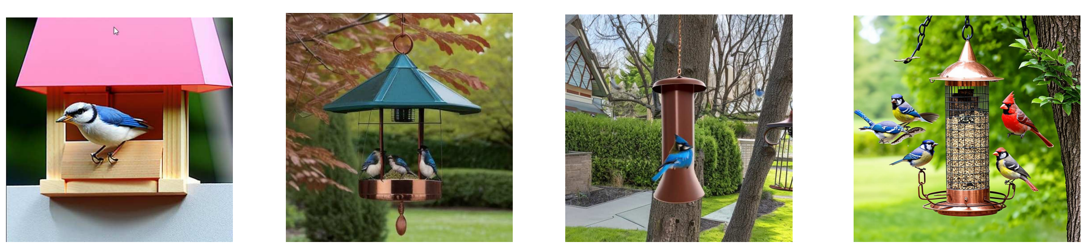


Figure 6.9: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion **SD 1.5** **SD 2.1** **SD 3 (medium)**



This image displays a bird feeder hanging from a tree in a garden. The bird feeder is made of metal and has a copper finish. It is cylindrical and has a pointed top. There are four perches for birds to land on. There are five birds in the image. They are a cardinal, a blue jay, a chickadee, a sparrow, and a goldfinch. The birds are all perched on the bird feeder. The cardinal is at the top, the blue jay is in the middle, the chickadee is on the left, the sparrow is on the right, and the goldfinch is below the sparrow. The background of the image is green foliage. The image is in focus and the colors are vibrant. The style of the image is realistic.

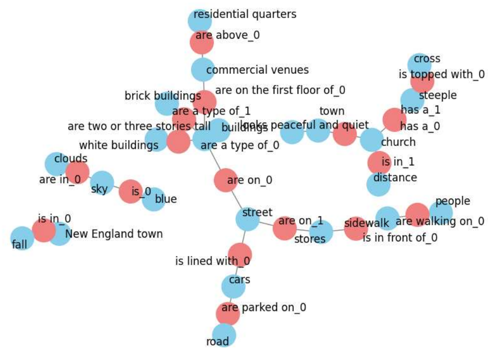


Figure 6.10: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion **SD 1.5** **SD 2.1** **SD 3 (medium)**

This image displays a quaint New England town in the fall. The street is lined with cars, parked on either side of the road. The buildings are mostly white, with some brick buildings as well. The buildings are two or three stories tall, with commercial venues on the first floor, and possibly residential quarters above. There are a few people walking on the sidewalk in front of the stores. The sky is blue and there are some puffy white clouds. The town looks very peaceful and quiet. In the distance, there is a white church steeple. The steeple is topped with a cross.

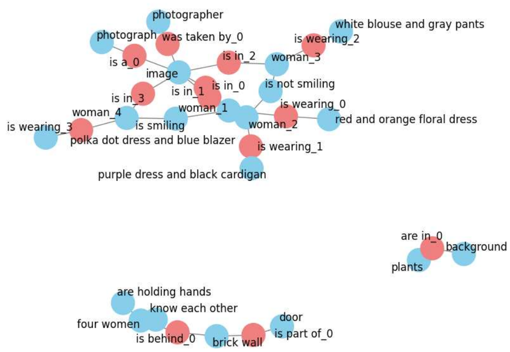


Figure 6.11: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion **SD 1.5** **SD 2.1** **SD 3 (medium)**

This image displays: Four women who appear to know each other, as they are holding hands in the center of the picture. They are standing in front of a brick wall with a door, and there are some plants in the background. One woman is wearing a red and orange floral dress, another is wearing a purple dress with a black cardigan, the third is wearing a white blouse with gray pants, and the fourth is wearing a polka dot dress with a blue blazer. The woman in the red and orange dress and the woman in the polka dot dress are smiling, the other two women are not smiling, but none of the women are frowning or showing any signs of distress. The background of the image is blurry, so it is not possible to make out any details. There is no text on the image. The image is a photograph, and it was taken by an unknown photographer.

Looking at the images, it is possible to observe that the proposed framework, while still producing images of inferior quality if confronted with the other ones, doesn't fail to capture the main subject of the picture, its main attributes and the overall surrounding scene. In particular in the first and in the third images reported above the model displayed an understanding of the scene perspective comparable with the other models. While in the last figure reported the model correctdely represented the number of instances of the same entity in the picture, performing better ,on this aspect, than Stable DIffusion 2.1.

6.3.2 SG Diffusion Qualitative Results on TIFA Captions

The image shown below are generated using captions from the TIFA benchmark, the captions reported are far shorter than the ones from the PixelProse dataset, this lead to the generation of smaller scene graphs with fewer nodes

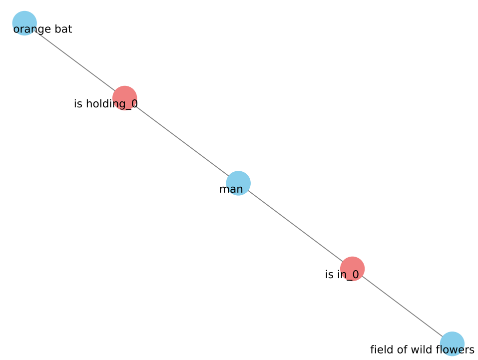
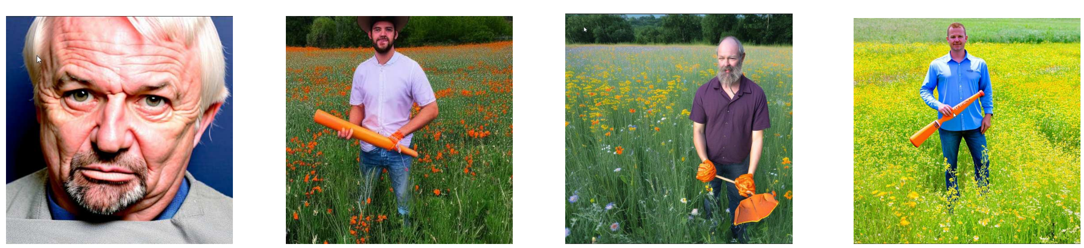


Figure 6.12: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion | **SD 1.5** | **SD 2.1** | **SD 3 (medium)**



A man in a fields of wilf flower holding an orange bat.

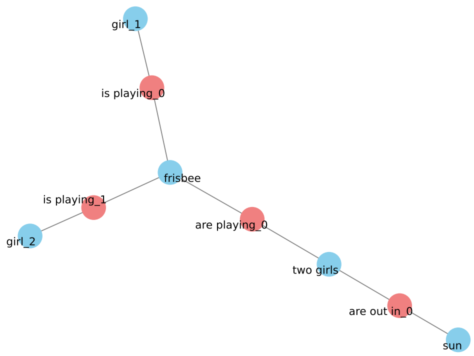


Figure 6.13: Scene graph parsed by the SG Parser, in red the nodes representing relationships, in light blue the nodes representing the entities.

SG Diffusion **SD 1.5** **SD 2.1** **SD 3 (medium)**

Two girls playing frisbee out in the sun.

As can be observed, the images generated by the TIFA benchmark are of lesser quality if compared with the ones from the other models, the proposed architecture still maintains its ability to represents the main actor of the caption, but fails in showing the same stylistic abilities than the other ones.

6.3.3 Adapter Performance

Although a qualitative comparison can provide a general and intuitive analysis of model performance, it remains subjective and non-quantifiable. It is therefore necessary to apply a quantitative benchmark to objectively evaluate the different models. Consequently, the same metrics that were previously outlined are employed to assess the final adapter performance in comparison to other existing State-Of-The-Art architectural designs. The TIFA values of the comparison models and of SG Diffusion were obtained by executing the benchmark five times. Each time with a different initial seed, in order to exclude the initial seed from the factors that may influence the final result. The seed that were used were randomly chosen and they were same for all the tested architectures, they are 0,17,355,1710,40000. The final score for each category is the average of all scores for that category. The results are showned in Table 6.7.

model	animal/object	object	location	activity	color	spatial	attribute	food	counting	material	shape	other	average	stdev
Stable Diffusion 1.5	0.824	0.779	0.822	0.782	0.782	0.723	0.778	0.795	0.670	0.828	0.550	0.682	0.787	0.224
Stable Diffusion 2.1	0.853	0.823	0.842	0.819	0.830	0.758	0.783	0.842	0.747	0.852	0.754	0.786	0.821	0.202
Stable Diffusion 3 (medium)	0.921	0.904	0.876	0.891	0.949	0.830	0.758	0.783	0.747	0.852	0.754	0.786	0.821	0.202
PixArt-α	0.887	0.823	0.830	0.817	0.859	0.762	0.778	0.872	0.749	0.861	0.659	0.736	0.823	0.198
SG Diffuser	0.533	0.413	0.437	0.390	0.484	0.460	0.515	0.414	0.397	0.550	0.420	0.393	0.451	0.269

Table 6.7: Table reporting the scores of the TIFA benchmark for different models, each model is tested 5 times with fixed seeds

model	FID
Stable DIffusion 1.5	48.77
Stable Diffusion 2.1	47.55
Stable Diffusion 3 (medium)	44.39
Pix-Art-alpha	45.43
SG Diffusion	71.48

Despite the efforts made and the changes introduced, the results obtained by both metrics are not satisfactory. A possible reason for this result is the fact that TIFA mainly uses shorter captions than the ones the model was trained on, while FID is ineffective in evaluating spatial relationships within the image and therefore fails to detect a change in the compositional quality of the images.

Chapter 7

Conclusions

7.1 Final Conclusion

This work explored the possibility of conditioning diffusion models for the Text-to-Image task via scene graphs. In particular, SG Diffusion was introduced, a framework capable of taking as input the textual description of the image to be generated, extracting the scene graph from the text through a parser, using a GCN adapter as a method for exchanging information between entities present in the same image, and finally conditioning the denoising process of Stable Diffusion 1.5. In particular, the proposed method avoids the 77 token limits of Stable DIffusion 1.5 by using node encoding instead of text description encoding. When benchmarked on TIFA the model performed poorly, a possible explanations for this behaviur is the number of nodes generated by the TIFA captions. This hypothesis is validated by the fact that imaged generated using PixelProse caption had led to good qualitative results as can be observed in 6.3.1. In fact images produced using PixelProse validation set didn't fail to capture the main subject of the image and, depending on the complexity, included also most of the spatial relationship that involved the main subject. Moreover the adapter proved to able to capture multiple instances of the same entity, in certain occasion performing even better than other comparison models, while in other images the model was able to capture the prospective requested by the caption as well as the other comparison models. Although the performance indicated by the metrics used did not show results that exceeded or matched the baseline model, a visual comparison of the images produced by SG Diffusion shows a good adherence to the original prompt.

7.2 Future works

In order to address the problems highlighted in the approach analysed, one possible route involves the introduction of heterogeneous graphs capable of representing different types of relationships by means of different arcs. Alternatively, another direction of research includes the introduction of a specific token, in addition to those of the nodes, which encapsulates and carries an embedding of the entire graph structure, thus allowing the underlying backbone to have multiple levels of granularity with which to exploit the graph information.

Chapter 8

Appendix

8.1 Parser Prompts

In this section are reported the prompts used by the Knowledge Graph Parser

8.1.1 Main Prompt

```
{"role": "system", "content": "Your task is to extract an
    \"Entity List\" from the caption provided, taking
   care to solve coreferences inside the text, and then
   to generate a \"Knowledge Graph\" using the entities
   contained in the \"Entity List\". Entities represent
   the main concepts or objects being described in the
   provided caption. These entities are the subjects and
   objects of the relationships' triplets in the \"
   Knowledge Graph\". The lists need to be in the
   following format: \"Entity List\": [\"entity_1\", \"
   entity_2\", \"entity_3\", ...], \"Knowledge Graph\":
   [[\"entity\", \"relationship\", \"entity\"], [\"entity
   \", \"relationship\", \"entity\"], ...]. The
   relationships' triplets in the \"Knowledge Graph\"
   must contain three elements. Do NOT generate triplets
   with more than three elements. Do NOT use bold or
   italic text. Do NOT format the list as a bullet point
   list or numbered list. If you found some Entities with
    the same name but that reference to different
   instances use a numerical progressive index. An
   example on how to indicate two entities that represent
```

```
    two different instances is the following: \"Entity
    List\": [\"cat_1\", \"cat_2\"]."},
{"role": "user", "content": "Caption: \"A family of four
    is sitting at a table in a restaurant. There are two
    adults, one child, and one grandmother. The adults are
     sitting on either side of the child. The grandmother
    is sitting across from the child. The table is set
    with a variety of food. There are two plates of rice,
    two plates of curry, and a plate of flatbread. There
    are also two glasses of water. The family is all
    smiling and talking. The background is a blur. The
    image is a photograph.\""},
{"role": "assistant", "content": "\"Entity List\": [\"
    family\", \"table\", \"restaurant\", \"four members\",
     \"adult_1\", \"adult_2\", \"child\", \"grandmother\",
     \"variety of food\", \"plate_1\", \"plate_2\", \"rice
    \", \"plate_3\", \"plate_4\", \"curry\", \"plate_5\",
    \"flatbread\", \"glass_1\", \"glass_2\", \"water\", \"
    image\", \"background\", \"blur\", \"photograph\"], \"
    Knowledge Graph\": [[\"family\", \"is sitting at\", \"
    table\"], [\"family\", \"has\", \"four members\"], [\"
    table\", \"is in\", \"restaurant\"], [\"adult_1\", \"
    is part of\", \"family\"], [\"adult_2\", \"is part of
    \", \"family\"], [\"child\", \"is part of\", \"family
    \"], [\"grandmother\", \"is part of\", \"family\"],
    [\"adult_1\", \"is sitting on either side of\", \"
    child\"], [\"adult_2\", \"is sitting on either side of
    \", \"child\"], [\"grandmother\", \"is sitting across
    from\", \"child\"], [\"table\", \"is set with\", \"
    variety of food\"], [\"plate_1\", \"is on\", \"table
    \"], [\"plate_2\", \"is on\", \"table\"], [\"plate_3
    \", \"is on\", \"table\"], [\"plate_4\", \"is on\", \"
    table\"], [\"plate_5\", \"is on\", \"table\"], [\"
    plate_1\", \"contain\", \"rice\"], [\"plate_2\", \"
    contain\", \"rice\"], [\"plate_3\", \"contain\", \"
    curry\"], [\"plate_4\", \"contain\", \"curry\"], [\"
    plate_5\", \"contains\", \"flatbread\"], [\"glass_1\",
     \"is on\", \"table\"], [\"glass_2\", \"is on\", \"
    table\"], [\"glass_1\", \"contain\", \"water\"], [\"
    glass_2\", \"contain\", \"water\"], [\"background\",
    \"is a blur\"], [\"image\", \"is a\", \"photograph
    \"]]"},
{"role": "user", "content": "Caption: \"This image
    displays: A group of seven men in military uniforms
    walking towards the camera. The men are wearing
```

```
    various military uniforms, including two men in green
    and brown camouflage uniforms, three men in tan and
    brown camouflage uniforms, one man in a grey and black
     military uniform, and one man in a white and blue
    military uniform. The men are walking in a group, with
     the man in the white and blue camouflage uniform in
    the front. The men are walking on a concrete path,
    with a tan building in the background. The building
    has a sign on it.\""},
{"role": "assistant", "content": "\"Entity List\": [\"
    group\", \"seven men\", \"man_1\", \"man_2\", \"man_3
    \", \"man_4\", \"man_5\", \"man_6\", \"man_7\", \"
    green and brown camouflage uniforms\", \"tan and brown
     camouflage uniforms\", \"grey and black military
    uniform\", \"white military uniform\", \"concrete path
    \", \"background\", \"tan building\", \"sign\"], \"
    Knowledge Graph\": [[\"group\", \"of\", \"seven men
    \"], [\"man_1\", \"is wearing\", \"green and brown
    camouflage uniforms\"], [\"man_2\", \"is wearing\", \"
    green and brown camouflage uniforms\"], [\"man_3\", \"
    is wearing\", \"tan and brown camouflage uniforms\"],
    [\"man_4\", \"is wearing\", \"tan and brown camouflage
     uniforms\"], [\"man_5\", \"is wearing\", \"tan and
    brown camouflage uniforms\"], [\"man_6\", \"is wearing
    \", \"grey and black military uniform\"], [\"man_7\",
    \"is wearing\", \"white and blue military uniform\"],
    [\"seven men\", \"are walking in\", \"group\"], [\"
    man_7\", \"is in front\"], [\"tan building\", \"is in
    \", \"background\"], [\"seven men\", \"are walking on
    \", \"concrete path\"], [\"tan building\", \"has a\",
    \"sign\"]]"}
```

8.1.2 Secondary Prompt

```
{"role": "system", "content": "Your task is to extract an
     \"Entity List\" from the caption provided, taking
    care to solve coreferences inside the text, and then
    to generate a \"Knowledge Graph\" using the entities
    contained in the \"Entity List\". Entities represent
    the main concepts or objects being described in the
    provided caption. These entities are the subjects and
    objects of the relationships' triplets in the \"
    Knowledge Graph\". The lists need to be in the
```

```
    following format: \"Entity List\": [\"entity_1\", \"
    entity_2\", \"entity_3\", ...], \"Knowledge Graph\":
    [[\"entity\", \"relationship\", \"entity\"], [\"entity
    \", \"relationship\", \"entity\"], ...]. The
    relationships' triplets in the \"Knowledge Graph\"
    must contain three elements. Do NOT generate triplets
    with more than three elements. Do NOT use bold or
    italic text. Do NOT format the list as a bullet point
    list or numbered list. If you found some Entities with
     the same name but that reference to different
    instances consider them as a group. An example on how
    to indicate ten entities that represent ten different
    instances is the following: \"Entity List\": [\"group
    of ten cats\"]. Do NOT enumerate instances."},
{"role": "user", "content": "Caption: \"A family of four
    is sitting at a table in a restaurant. There are two
    adults, one child, and one grandmother. The adults are
     sitting on either side of the child. The grandmother
    is sitting across from the child. The table is set
    with a variety of food. There are two plates of rice,
    two plates of curry, and a plate of flatbread. There
    are also two glasses of water. The family is all
    smiling and talking. The background is a blur. The
    image is a photograph.\""},
{"role": "assistant", "content": "\"Entity List\": [\"
    family\", \"table\", \"restaurant\", \"four members\",
     \"two adults\", \"child\", \"grandmother\", \"variety
     of food\", \"two plates_1\", \"rice\", \"two plates_2
    \", \"curry\", \"one plate\", \"flatbread\", \"two
    glasses\", \"water\", \"image\", \"background\", \"
    blur\", \"photograph\"], \"Knowledge Graph\": [[\"
    family\", \"is sitting at\", \"table\"], [\"family\",
    \"has\", \"four members\"], [\"table\", \"is in\", \"
    restaurant\"], [\"two adults\", \"is part of\", \"
    family\"], [\"child\", \"is part of\", \"family\"],
    [\"grandmother\", \"is part of\", \"family\"], [\"two
    adults\", \"is sitting on either side of\", \"child
    \"], [\"grandmother\", \"is sitting across from\", \"
    child\"], [\"table\", \"is set with\", \"variety of
    food\"], [\"two plates_1\", \"is on\", \"table\"], [\"
    two plates_2\", \"is on\", \"table\"], [\"one plate\",
     \"is on\", \"table\"], [\"two plates_1\", \"contain
    \", \"rice\"], [\"two plates_2\", \"contain\", \"curry
    \"], [\"one plate\", \"contains\", \"flatbread\"], [\"
    two glasses\", \"is on\", \"table\"], [\"two glasses
```

```
    \", \"contain\", \"water\"], [\"background\", \"is a
    blur\"], [\"image\", \"is a\", \"photograph\"]]"},
{"role": "user", "content": "Caption: \"This image
    displays: A group of seven men in military uniforms
    walking towards the camera. The men are wearing
    various military uniforms, including two men in green
    and brown camouflage uniforms, three men in tan and
    brown camouflage uniforms, one man in a grey and black
     military uniform, and one man in a white and blue
    military uniform. The men are walking in a group, with
     the man in the white and blue camouflage uniform in
    the front. The men are walking on a concrete path,
    with a tan building in the background. The building
    has a sign on it.\""},
{"role": "assistant", "content": "\"Entity List\": [\"
    group of seven men\", \"military uniforms\", \"two men
    \", \"three men\", \"one man_1\", \"one man_2\", \"
    green and brown camouflage uniforms\", \"tan and brown
     camouflage uniforms\", \"grey and black military
    uniform\", \"white military uniform\", \"concrete path
    \", \"background\", \"tan building\", \"sign\"], \"
    Knowledge Graph\": [[\"group of seven men\", \"in\",
    \"military uniforms\"], [\"two men\", \"is wearing\",
    \"green and brown camouflage uniforms\"], [\"three men
    \", \"is wearing\", \"tan and brown camouflage
    uniforms\"], [\"one man_1\", \"is wearing\", \"grey
    and black military uniform\"], [\"one man_2\", \"is
    wearing\", \"white and blue military uniform\"], [\"
    group of seven men\", \"are walking in\", \"group\"],
    [\"one man_2\", \"is in front\"], [\"tan building\",
    \"is in\", \"background\"], [\"group of seven men\",
    \"are walking on\", \"concrete path\"], [\"tan
    building\", \"has a\", \"sign\"]]"}
```

8.2 Parser Benchmark's caption

Here are listed some of the captions used in the Parser Benchmark

8.2.1 Caption Lenght < 150, 3 spatial relationships, one instance per entity

```
"A dog rests under the shady tree, next to a wooden bench,
   and in front of a sparkling blue pond surrounded by
   flowers."
"She stands under the bright umbrella, next to an antique
   lamppost, and in front of the cafe' window filled with
   pastries and plants."
```

8.2.2 Caption Lenght < 150, 3 spatial relationships, max 3 instances per entity

```
"Two large cakes are displayed on the a table, while guests
   gather around the dance table, and streamers hang from the
    ceiling above them."
"Three racing cars are zooming around the track, with one
   driver ahead the other two, the crowd is cheering from the
    grandstands above."
```

8.2.3 Caption Lenght < 150, 5 spatial relationships, one instance per entity

```
"Under the neon lights an office worker is sitting at his
   desk, in front of the computer, next to the window. A cup
   of coffee is on the desk."
"A dragon is flying over a castle, between two clouds, above
   the towers while having its tail wrapped around an enemy.
   A cannon is in the background."
```

8.2.4 Caption Lenght < 150, 5 spatial relationships, one instance per entity

```
"A gold miner is inside a mine, under a lamp, on the walls
   two rocks are shining. The first one is on the right,
   while the other is on the left."
"Two astronauts are on the moon, one is near the lander,
   while the other is inside a crater. The Earth is in the
   background, between the stars."
```

8.2.5 Caption Lenght 150−300, 3 spatial relationships, one instance per entity

```
"A man stands on the rooftop of a tower, looking out over the
    sparkling city below. Next to him, a telescope is aimed
   at the stars. Beneth the tower a there is a well curated
   garden, with colorful flowers blooming in the pots and
   plants along the edges of the garden."
"An ancient temple stands proudly in the center, surrounded
   by towering columns. To the left, a quiet river flows
   beneath the stone bridge, while a small garden blooms next
    to the entrance, with wildflowers swaying gently under
   the breeze."
```

8.2.6 Caption Lenght 150−300, 3 spatial relationships, max 3 instances per entity

```
"Three friends are gathered around a campfire, roasting
   marshmallows. To the left, a tent is pitched, while a dog
   sits beside the fire, waiting for a treat. The stars are
   shining brightly above them."
"The hiker pauses at the summit, gazing out at the valley
   below. To the right, a rocky outcrop juts out from the
   mountain, while another hiker stands beside him, resting
   on a boulder. Their backpacks are neatly placed next to
   them as they enjoy the view."
```

8.2.7 Caption Lenght 150−300, 5 spatial relationships, one instance per entity

```
"A car is driving along a busy road. To the left of the car
   there is a pole with a sign on it. The sign says "No right
    turn on red". Behind the car a red and blue lorry has
   stopped and to the right of the car there is a motorbike
   with a woman on it."
"A tiny insect and a spider are crawling alongside onto a
   leaf. The leaf is from an hibiscus plant placed next to a
   sunflower inside a vase. Both plants are in a corner of
   vibrant anf full-of-life garden. Beside the hibiscus plant
    a gardener is filling the watering can."
```

8.2.8 Caption Lenght 150−300, 5 spatial relationships, max 3 instance per entity

```
"A golden retiever barked excitedly near the wooden fence
   that bordered the backyard, its tail wagging as its ball
   rolled under the picnic table sitting to the left of the
   weathered garden shed. Another golden retiever is lounging
    peacefully on the porch above the yard, next to a rocking
    chair."
"Three giraffes are enclosed by metal fences in a zoo. One of
    the giraffes is placed in the background of the enclosure
   , above a huge gray rock. The other two giraffes are
   placed one on the right and one on the left, below the
   rock, they both are eating some tropical tree leaves."
```

Bibliography

[1] Abhimanyu Dubey et al. *The Llama 3 Herd of Models*. 2024.

[2] Jacob Austin et al. "Program synthesis with large language models". In: *arXiv preprint arXiv:2108.07732* (2021).

[3] Jason Baumgartner et al. "The pushshift reddit dataset". In: *Proceedings of the international AAAI conference on web and social media*. Vol. 14. 2020, pp. 830–839.

[4] Stefan Bloemheuvel, Jurgen van den Hoogen, and Martin Atzmueller. "A computational framework for modeling complex sensor network data using graph signal processing and graph neural networks in structural health monitoring". In: *Applied Network Science* 6 (Dec. 2021). DOI: `10.1007/s41109-021-00438-8`.

[5] Pingping Cao et al. 2022.

[6] Angel Chang, Manolis Savva, and Christopher D Manning. "Learning spatial knowledge for text to 3D scene generation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 2028–2038.

[7] Soravit Changpinyo et al. *Conceptual 12M: Pushing Web-Scale Image-Text Pre-Training To Recognize Long-Tail Visual Concepts*. 2021.

[8] Junsong Chen et al. *PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis*. 2023. arXiv: `2310.00426 [cs.CV]`.

[9] Heeyoul Choi, Kyunghyun Cho, and Yoshua Bengio. "Fine-grained attention mechanism for neural machine translation". In: *Neurocomputing* 284 (2018), pp. 171–176. ISSN: 0925-2312. DOI: `https://doi.org/10.1016/j.neucom.2018.01.007`. URL: `https://www.sciencedirect.com/science/article/pii/S0925231218300225`.

[10] Hyung Won Chung et al. "Scaling instruction-finetuned language models". In: *Journal of Machine Learning Research* 25.70 (2024), pp. 1–53.

[11] Karan Desai, Gaurav Kaul, and Justin Johnson Zubin Aysola. *RedCaps: web-curated image-text data created by the people, for the people.* 2021.

[12] Patrick Esser et al. *Scaling Rectified Flow Transformers for High-Resolution Image Synthesis*. 2024.

[13] Alex Fout et al. "Protein interface prediction using graph convolutional networks". In: *Advances in neural information processing systems* 30 (2017).

[14] Samir Yitzhak Gadre et al. *DataComp: In search of the next generation of multimodal datasets*. 2023.

[15] Leo Gao et al. "The pile: An 800gb dataset of diverse text for language modeling". In: *arXiv preprint arXiv:2101.00027* (2020).

[16] Sebastian Hartwig et al. *A Survey on Quality Metrics for Text-to-Image Models*. 2024.

[17] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2023. arXiv: 1606.08415 [cs.LG]. URL: https://arxiv.org/abs/1606.08415.

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

[19] Martin Heusel et al. *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. 2018. arXiv: 1706.08500 [cs.LG]. URL: https://arxiv.org/abs/1706.08500.

[20] Yushi Hu et al. "TIFA: Accurate and Interpretable Text-to-Image Faithfulness Evaluation with Question Answering". In: *arXiv preprint arXiv:2303.11897* (2023).

[21] Weiwei Jiang and Jiayun Luo. "Graph neural network for traffic forecasting: A survey". In: *Expert Systems with Applications* 207 (2022), p. 117921.

[22] Justin Johnson et al. "Image retrieval using scene graphs". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2015, pp. 3668–3678.

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

[24] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2017.

[25] Chang Li and Dan Goldwasser. “Encoding social information with graph convolutional networks forpolitical perspective detection in news media”. In: *Proceedings of the 57th annual meeting of the association for computational linguistics*. 2019, pp. 2594–2604.

[26] Long Lian et al. “LLM-grounded Diffusion: Enhancing Prompt Understanding of Text-to-Image Diffusion Models with Large Language Models”. In: *arXiv preprint arXiv:2305.13655* (2023).

[27] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015.

[28] Haotian Liu et al. “Visual Instruction Tuning”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 34892–34916. URL: `https://proceedings.neurips.cc/paper_files/paper/2023/file/6dcf277ea32ce3288914faf369fe6de0-Paper-Conference.pdf`.

[29] Andreas Lugmayr et al. “Repaint: Inpainting using denoising diffusion probabilistic models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11461–11471.

[30] Wan-Duo Kurt Ma et al. “Directed Diffusion: Direct Control of Object Placement through Attention Guidance”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.5 (Mar. 2024), pp. 4098–4106. DOI: `10.1609/aaai.v38i5.28204`. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/28204`.

[31] Ben Mann et al. “Language models are few-shot learners”. In: *arXiv preprint arXiv:2005.14165* 1 (2020).

[32] Christopher Morris et al. *Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks*. 2021.

[33] Alec Radford. “Improving language understanding by generative pre-training”. In: (2018).

[34] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021.

[35] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022.

[36] Patrick Reiser et al. *Graph neural networks for materials science and chemistry*. 2022.

[37] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2016), pp. 1137–1149.

[38] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2021. arXiv: 2112.10752 [cs.CV].

[39] Chitwan Saharia et al. "Palette: Image-to-image diffusion models". In: *ACM SIGGRAPH 2022 conference proceedings*. 2022, pp. 1–10.

[40] Tim Salimans et al. *Improved Techniques for Training GANs*. 2016.

[41] Christoph Schuhmann et al. "Laion-5b: An open large-scale dataset for training next generation image-text models". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 25278–25294.

[42] Maximilian Seitzer. *pytorch-fid: FID Score for PyTorch*. https://github.com/mseitzer/pytorch-fid. Version 0.3.0. Aug. 2020.

[43] Rico Sennrich, Barry Haddow, and Alexandra Birch. *Neural Machine Translation of Rare Words with Subword Units*. 2016. arXiv: 1508.07909 [cs.CL]. URL: https://arxiv.org/abs/1508.07909.

[44] Eran Yahav Shaked Brody Uri Alon. *How Attentive are Graph Attention Networks?* 2022.

[45] Noam Shazeer. *GLU Variants Improve Transformer*. 2020. arXiv: 2002.05202 [cs.LG]. URL: https://arxiv.org/abs/2002.05202.

[46] Guibao Shen et al. *SG-Adapter: Enhancing Text-to-Image Generation with Scene Graph Guidance*. 2024.

[47] Vasu Singla et al. *From Pixels to Prose: A Large Dataset of Dense Image Captions*. 2024.

[48] Xinying Song et al. *Fast WordPiece Tokenization*. 2021. arXiv: 2012.15524 [cs.CL]. URL: https://arxiv.org/abs/2012.15524.

[49] Damien Teney, Lingqiao Liu, and Anton van Den Hengel. "Graph-structured representations for visual question answering". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1–9.

[50] Ashish Vaswani et al. *Attention Is All You Need*. 2017.

[51] Petar Veličković et al. *Graph Attention Networks*. 2018.

[52] Jing Xiong et al. "Autoregressive Models in Vision: A Survey". In: *arXiv preprint arXiv:2411.05902* (2024).

[53] Danfei Xu et al. "Scene graph generation by iterative message passing". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5410–5419.

[54] Hu Ye et al. "IP-Adapter: Text Compatible Image Prompt Adapter for Text-to-Image Diffusion Models". In: (2023).

[55] Rex Ying et al. "Graph convolutional neural networks for web-scale recommender systems". In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 974–983.

[56] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. *Adding Conditional Control to Text-to-Image Diffusion Models*. 2023.

[57] Wayne Xin Zhao et al. *A Survey of Large Language Models*. 2024. arXiv: `2303.18223 [cs.CL]`. URL: `https://arxiv.org/abs/2303.18223`.