NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

MSc: DATA SCIENCE AND INFORMATION TECHNOLOGIES

Masters Thesis

Enhancing and unifying Vision-Language tasks with Diffusion models

Christos Morfopoulos

Athens
October 2023

Masters Thesis

Enhancing and unifying Vision-Language tasks with Diffusion models

Christos Morfopoulos

A.M. : 7115152100011

Supervisor : Yannis Avrithis

Examination Committee:

Harris Papageorgiou

Vassilis Katsouros

Abstract

In computer vision, image captioning is a challenging task, in which the goal is to bridge the gap between visual content and natural language understanding. Image captioning, as the name suggests, is the process where a descriptive caption is automatically generated from an image. Another challenging task is visual question answering, where a user can ask questions about an image and receive meaningful answers. In recent years, there is a lot of effort in the research community to improve both processes, by introducing different architectures and methods. Image captioning and visual question answering are two very related vision-language tasks. However, they are treated individually.

In this thesis, we follow a lightweight approach for image captioning and we made a thorough investigation of all its components. We then extend that method to handle visual question answering tasks. Finally, we introduce a unified model, which is trained via multitask learning on both image captioning and visual question answering. This single model can handle both tasks at inference, achieving competitive performance in both. Surprisingly, although multitask learning often leads to inferior performance in the individual tasks, in our case it even improves performance.

In another direction, we employ the power of diffusion generative models to boost the performance of our image captioning and visual question answering models. Using diffusion, we generate images from the existing captions of each training set to create new, synthetic datasets. By controlling each generated image to be similar to the existing one corresponding to the caption, we verify that the synthetic datasets can assist to improve the performance of captioning, as well as visual question answering in the presence of multitask learning.

Acknowledgements

I would like to thank my supervisor, Yannis Avrithis, for his guidance and support in this project. I am deeply grateful to Yannis, his assistance was of vital importance and his suggestions in the field of Artificial Intelligence serve as standards that I greatly admire.

Most of the experiments of this project and the development of the source code was made in the resources of the IARAI, Institute of Advanced Research in Artificial Intelligence. I am deeply grateful to the IARAI team, who gave me the opportunity to utilise their computational resources and finally conduct all the required experiments to fulfil this project.

Finally, I would like to thank my family and my partner, who support me through all the stages of this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The last years, we have noticed revolutionary improvements in Artificial Intelligence. Generally, in Computer Vision, the dominance of Convolutional Neural Networks was questionable, where the CNN architecture was considered state of the art a decade ago. In Nature Language Processing, a significant improvement was made though "Attention Is All you need" paper [32], in which the Transformer architecture was first introduced along with the attention mechanism. The transformer architecture consists of an Encoder and a Decoder and the models who adapted this architecture had a remarkable performance in many NLP tasks, outperforming all the previous NLP architectures such as RNN and LSTM [28]. However, the transformer architecture had a great impact in the field of Computer Vision as well, where the research community tried to replace the CNN architectures with the powerful Vision-Transformer architectures.

In the same way, the domain of image captioning was affected by the transformer architecture. The initial image captioning architectures have utilised certain approaches, combining pre-trained CNN models with LSTMs or RNNs [28]. However, the overall performance of those models was insufficient and the errors of their outputs were often noticed. Therefore, the research community was looking for alternative techniques, knowing that the effectiveness of those models had certain limitations. After the release of the transformer architecture, the encoder and decoder [32] elements were employed for image captioning tasks and the results were incredible, overtaking all the previous models in terms of performance. Most of the time, the position of the encoder is set to a vision model, whereas the position of the decoder is set to a language model. The

binding between the encoder and the decoder is succeeded through the cross attention mechanism.

Additionally, the domain of Visual Question Answering [2] has adapted standard approaches to answer their desired visual questions. A common technique is to perceive this challenge as a classification task. In other words, in this approach you have to integrate additional Neural Network layers along with a softmax layer in order to classify the input image and question to a particular answer. Obviously, this approach has limitations and its biggest issue is that it defines a closet set of answers. Therefore, for Visual Question Answering tasks, the Transformer architecture is not commonly applied and the enriched vocabulary of a pre-trained decoder is underutilised.

1.2 Motivation

With the rise of the encoder and the decoder via transformer architecture, the domain of image captioning has many aspects under investigation. Certain questions arise from the research community with respect to that particular mechanism. Which is the optimal model? Which encoder or which decoder is the best suitable for this task? How big in terms of capacity should be the vision models or the language models? How many layers should the encoder or the decoder have? Can we use pre-trained vision models or large language models and if we can, how to utilise them? Therefore, in this project, our main goal is to explore and investigate all the possible options, in order to have an efficient solution for any image captioning sub-task.

Many image captioning models have integrated large vision and language models with a total of millions of parameters. As a result, the training time of this procedure grows exponentially and the overall approach is already heavyweight. Therefore, in this project, we explored a lightweight solution for image captioning tasks, where not only the training, but also the inference time is much less than the previous approaches. In this way, we can offer an implementation, which can be trained very easily and deliver detailed captions for any user.

With respect to the Visual Question Answering task [2], the usual way is to handle it as a classification task. A main objective of this project is to alter this approach. We implemented a method, in which the decoder can generate outputs with respect to an image and a question. In this way, we have a transition from classification to the conventional encoder-decoder method, in which

the answer/outputs are generated in an auto-regressive manner. This has as a result to adapt the full encoder-decoder mechanism without any addition such as dense Neural Network layers for classification.

Consequently, as we have relative approaches both for image captioning and for visual question answering, we tried to attach both methods into a single unified approach, where the unified model can handle both image captioning and visual question answering tasks. We achieved that through multitask learning [5], utilising a unified loss function and adjusting the weights for both tasks. In this way, the model can be trained for both tasks and exploit the possibility to learn more representations through the relative assignments.

Lastly, in this project, we tried to employ the power of image generative models. The Diffusion models [26] can generate any image with respect to a caption. As we aim to achieve better performance for the image captioning models, we invoked the impressive stable diffusion models [24], to generate images so we can have additional training data. Moreover, we implemented different diffusion pipelines on how to select the best possible image via the image generation process.

1.3 Existing Work

The main model that we have used as a backbone, is based on the paper "Clip-cap: CLIP prefix for Image Captioning" [17]. The structure of this lightweight model consists of a frozen pre-trained vision model, a frozen pre trained language model and a trainable transformer in between. As an image encoder the authors have selected the CLIP model [21], which was trained in more than 400 million image-text pairs and its main goal is to understand and generate meaningful connections between images and their corresponding textual descriptions. As for the decoder, it is set to a pre-trained GPT2 Language model [22]. The main novelty of that particular paper lies on the trainable Transformer, which essentially is an encoder with eight encoder layers. The main goal of that Transformer is to map the embedding space of the CLIP with the respective space of the GPT2. This is why it is also called a "Mapping" network.The outcome , as the title suggests, is to generate a semantic prefix which is passed to the GPT2 Language model to produce the description.

In the last few months, a paper was announced by Microsoft with the title "GIT: A Generative Image-to-text Transformer for Vision and Language" [35]. This

architecture follows the conventional approach of encoder-decoder technique, in which the image encoder is set to the large variant of the CLIP model and the text decoder to the large variant of BERT [9]. The main GIT model variant has been trained roughly on 0.8 billion image-text pairs. The model can deliver both image captioning and visual question answering tasks. The main training is based on image captioning, but the fine-tuning applies on the visual question answering tasks with a modification on the attention mask.

To sum up, the GIT [35] as well as the Clipcap [17] are architectures completely different, both specialised for image captioning tasks. Any architecture has its own advantages and disadvantages and it can be adapted for any particular task or situation. For instance, the Clipcap architecture stands out in terms of agility and computational cost, whereas the GIT architecture excels in performance.

Nevertheless, a notable drawback with respect to previous architectures is that we do not have a single architecture that is mainly designed for visual question answering [2]. The GIT models are designed and trained mainly for image captioning tasks and finally fine tuned for visual questioning tasks. This has as a result to define visual-language models that are specialised for one relative task and partially trained for other tasks. In this way, we do not have the opportunity to combine those customised models and train them in parallel through multi task learning in order to leverage their performance.

1.4 Contributions

Having outlined some of the most significant goals of this project, overall our main contributions are as follow:

- We conduct extensive experiments with respect to our main model architecture on three different image-text datasets COCO [16], VizWiz [12] and TextCaps [29], applying all possible combinations of decoder capacities and distinct Diffusion pipelines.
- We introduce a modified architecture, in order to deliver visual question answering tasks. With small alterations of our main mechanism, but keeping the same approach, the model can provide an answer with respect to a specific visual content and question.
- We present a unified approach to deliver both image captioning and visual question answering tasks. With the utilisation of multi task learning, a model can provide a detailed description or an enriched answer.

- We design two distinct diffusion pipelines to generate an image with respect to a caption. The vanilla diffusion pipeline has integrated certain settings and prompt-engineering prefixes, whereas the "Clipscore" diffusion pipeline follows a rationale of generating multiple images and selecting an optimal image with respect to the CLIP embeddings [21].
- Finally, through all our experiments, we can verify that a certain derivative model that we introduce, which has adapted the large variant of a decoder (GPT2-XL) [22] as well as leveraged additional training images via "CLIPscore" diffusion pipeline, can outperform any model variation of its category and it is considered to be the most efficient model.

1.5 Structure

In this project, our goal is to identify the optimal combination to achieve high performance in both image captioning and visual question answering tasks, ensuring comparability among all models. The chapters of this work are organised as follows:

- Introduction, makes an introduction and definition of our tasks while also presenting the challenges, related work and contributions of this work.
- Background, presents the story of image captioning and visual question answering in detail, ranging from former methods to current state of the art solutions.
- Architecture, presents the key components of our approach.
- Implementation, highlights the modifications and advancements in our method.
- Experiments, presents details about our main datasets and our experimental setup.
- Results, demonstrates our findings based on our extensive experiments.
- Conclusion, outlines our findings.

Chapter 2

Background

2.1 Transformer

2.1.1 Core Concept

Several years back, language modelling in Natural Language Processing, was mainly based with architectures such as RNN and LSTM [28]. Although these architectures were considered state of the art, there were fundamental limitations in the sequence computation. Certain disadvantages were no parallel processing and a lot of training time. Additionally, RNNs and LSTM [28] architectures have a short reference window, which were incapable of using the entire context of the story while generating the text.

A major turning point in the deep learning research area was when the "Attention is all you need" paper [32] was published by Google in 2017. A new novel neural network called the Transformer, which is an attention based encoder-decoder type architecture, was introduced. This architecture was originally designed for Machine Translation tasks, outperforming all the previous architectures. The key element of Transformer [32] architecture, which makes that particular architecture superior in contrast to the previous, is the attention mechanism. The power of the attention mechanism is that it doesn't suffer from short term memory as RNNs and LSTM [28]. It has an infinite reference window, therefore a bigger capacity, which leads to capture the entire input context while decoding their final output. In this way, the original transformer architecture not only can handle parallel computation for sequential input data, where RNNs [28] need to process the input data in order, but also have significantly better performance. Finally, this state of the art architecture managed

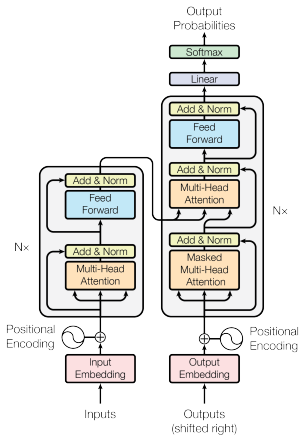


Figure 1: *The Transformer [32] architecture, it mainly consists of an Encoder (Left) and a Decoder (Right).*

to leverage the power of attention mechanism to make better predictions.

The main modules of this architecture are the encoder and the decoder layers. The authors of the paper proposed six stacked encoder as well as six stacked decoder blocks. Each layer has integrated the attention mechanism which is applied several times. At high level, the main goal of the Encoder [32] is to map an input sequence into an abstract continuous representation that holds all the learned information of that input, whereas the decoder utilises that continuous representation and generates autoregressively a single output while also taking into account its previous output.

The main pre-processing steps of the sequence input consist of the Word Embedding as well as the Positional Encoding [32] modules. The input of the architecture is a sequence of words and it is required to convert those words into vectors. This transition has as a result to have a continuous representation of each word. The conversion is developed in such a way that similar vectors can have similar representations. A standard limitation is that the embedding vector can handle up to 512 tokens per sequence.

For obvious reasons, the order of the words has a great impact on the meaning of a sentence. Therefore, a very important step is to capture the position of the tokens in a sequence. To achieve this, the authors of the paper have developed a process in which they add positional information of the respective words in

a sentence, simply by adding a new vector to the current embedding vector. Taking into consideration the current position p of the embedding as well as the index i of the position embedding dimension and finally applying cosine and sine functions, we can capture the word ordering. The positional encodings are calculated using the following equations:

$$\mathrm{PE}_{(\mathrm{p},2\mathrm{i})} = \sin\left(\frac{\mathrm{p}}{10000^{2i/d_{\mathrm{model}}}}\right) \tag{2.1}$$

$$\mathrm{PE}_{(\mathrm{p},2\mathrm{i}+1)} = \cos\left(\frac{\mathrm{p}}{10000^{2i/d_{\mathrm{model}}}}\right) \tag{2.2}$$

These equations represent the positional encodings for even and odd indices, respectively. Here, p represents the position of the token in the sequence, i represents the dimension, and d_{model} represents the dimensionality of the model's embeddings. By integrating these equations into the embedding vectors, the model gains valuable positional information, enhancing its understanding of word sequences.

2.1.2 Encoder Layer

The mechanics of an encoder block are composed of a Multi-Head Attention [32] module and a fully connected Feed Forward network layer, followed by a Normalisation Layer and its residual connections. The Multi-Head Attention [32] module is responsible for applying the attention mechanism with respect to the input data, in which the attention weights are calculated. In this way, the model can learn to associate each individual word in the input sequence to other words in the sequence. The residual connections help the network to train by allowing gradients to flow through the network directly, whereas the Normalisation layer helps to stabilise the network. The Feed Forward network layer has integrated several Linear layers with ReLu activation functions in between. Its main task is to further process the attention output. All the operations within an encoder block aim to encode the input to a continuous representation along with attention information.

The consecutive encoder blocks allow the model to capture different attention aspects, which has as a result a final enriched representation with respect to the input data. In this way, the transformer network potentially boosts its predictive power, while helps the decoder layers focus on the appropriate words in their input, during the decoding process.

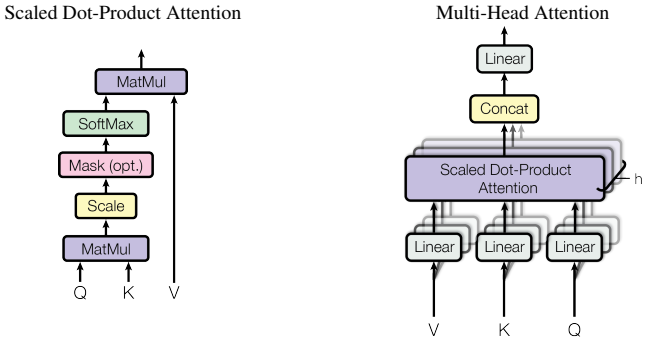


Figure 2: *The Multi-Head Attention mechanism that is applied in the Transformer [32] architecture, (Left) Scaled Dot-Product Attention. (Right) Multi-Head Attention consists of several attention layers running in parallel.*

2.1.3 Multi Head Attention

The sub-module of Multi Head Attention [32], which lies within the encoder layer, applies a specific attention mechanism commonly called self attention. Self-attention empowers the model to establish associations between each distinct word in the input and other words contained in the input. To achieve it, the input is fed to three distinct fully connected layers in order to create query, key and value vectors. The main idea is to map the query with the key vectors via a certain pipeline and finally connect the output with the respective value vectors.

The query, value, and key vectors undergo a dot product matrix multiplication to produce a score matrix. The score matrix, denoted as Attention(Q, K, V), determines how much focus each word should have on other words. Each word is assigned a certain score corresponding to other words, calculated using the formula:

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

where Q, K, and V represent the query, key, and value vectors respectively. Here, $\sqrt{d_k}$ is the scaling factor, ensuring stable gradients and preventing exploding effects in the process. The softmax function scales the scores, producing attention weights ranging from zero to one. This scaling operation modifies the score values, inflating high values and depressing low values. Finally, the last step in the scaled dot-product attention involves a matrix multiplication

between the obtained attention weights and the value vector (V), refining the focus of the model on specific words.

Each self attention process is also called attention head. In order to get a fully Multi Head Attention [32], as its title suggests, we have to apply the same process several times. To achieve this, we split the query, key and value vectors into N vectors and employ the same self attention process individually. Each attention head in the process would learn something different, therefore enable the encoder to gain more representation power. After we applied N attention heads, we concatenate its output and as a last part of the Multi Head Attention process we apply a Linear Layer to shrink back the concatenated output back to its original size.

2.1.4 Decoder Layer

The goal of the decoder [32] is to generate text sequences with respect to the encoded representations. Although its layers are very similar to the encoder layers, there are main differences.

The components of the decoder are two Multi Head Attention [32] sub-modules and a fully connected Feed Forward neural network, followed by Normalisation layer and its residual connections, in the same way as in encoder's. However, its Multi Head Attention [32] modules are slightly different and each module serves a distinct task. The first Multi Head Attention of the decoder, also known as Masked Multi Head Attention, applies a certain attention mask not only to hide future tokens for the attention output but also to avoid introducing bias into the architecture. This act is mainly because the decoder operates in an autoregressive manner. Moreover, in the consecutive Multi Head Attention [32] the input of the decoder is the output of the final encoder block. In other words, the decoder utilises the encoder output as query and key vectors in conjunction with output of the first Multi Head Attention as value vector. This process matches the encoder output to the decoder input, allowing the decoder to decide which encoder input is relevant to put focus on. This mechanism is also known as Cross Attention [32] mechanism.

Finally, the Linear Layer acts as a classifier, which is as big as the number of classes/words, and applying a softmax layer, we can take the index of the highest word probability. Obviously, the model picks the word with the highest probability as the best candidate for the next token in the sequence. In the same way as in the encoder layers, the decoder layers can be stacked allowing

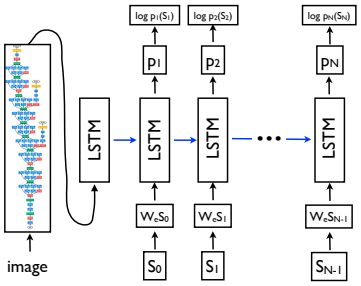


Figure 3: *The conventional architecture of "Show and Tell" [34] for image captioning. (Left) Extracting features from CNN, (Right) LSTM [28] decoder.*

the model to learn to extract and focus on different combinations of attention.

2.2 Image Captioning

2.2.1 Former Approaches

A very important and fundamental task in the domain of deep learning is image captioning. Image captioning is the process of generating a textual description for a given image. Several approaches have been proposed to solve this task. Almost a decade ago, a standard and verified approach, which was considered state of the art, was to utilise a Convolutional Neural Network to obtain the feature semantics of an image as well as to utilise a Recurrent Neural Network to generate the sequence of words. The key element of every image captioning process is to link the image feature vectors to its respective caption.

An official architecture, which followed the above technique, was proposed in a paper titled "Show and Tell: A Neural Image Caption Generator" [34] by Google in 2015. This architecture utilised LSTMs instead of plain RNN architecture in conjunction with Convolutional Neural Network. The union of a CNN and LSTM in its simplest terms takes the image as the input and outputs a description of what the image depicts. At high level, the CNN acts as an encoder and the LSTM [28] as a decoder. At the initial steps of the architecture the CNN creates the feature vector from the image, which is called an embedding. Usually, it is pre-trained Convolutional Neural Network, which is trained in a large number of images for other tasks such as object detection or classification. Then, the encoded image is transformed through a Linear Layer in order to pass as an input to the LSTM [28] decoder. With respect to the extracted features

from the CNN, the LSTM produces a sequence of words that describe what was in the image.

Although pre-trained CNNs can capture high-level semantic information in images and LSTMs can handle sequential data, there were certain limitations in generating contextually relevant captions for images. A main disadvantage is that designing and training a CNN-LSTM [28] architecture can be complex and computationally expensive. Additionally, due to the short reference window of an LSTM [28], the architecture can have low performance, with often errors in large captions and a requirement of large training time, consuming a lot of computational resources. This had as a result the research community to investigate and focus on alternative approaches for image captioning.

2.2.2 Encoder-Decoder

The innovative approach of encoder-decoder in the Transformer architecture can be remarked as revolutionary. The rise of attention mechanisms solved successfully a lot of NLP tasks such as sequence to sequence problems or machine translation. The research community suggested exploring this approved mechanism in other aspects of deep learning and not exclusive in text to text tasks. The initial questions were based on how to use an encoder that focuses only on images and mainly how can we link an image encoder with a textual decoder.

Image captioning is analogous to other sequence to sequence tasks except the only difference is that instead of translating from a language to language we are translating from an image to a language. The key element of the encoder decoder architecture [32] that allows the model to apply such a transition, is the cross attention layer. Via the cross attention mechanism, we can tie the embeddings of the vision encoder as well as the embeddings of the textual decoder. Therefore, in the cross-attention head, the encoder embeddings are used as key and value vectors and the decoder embeddings are used as query vectors. Using these inputs, the model is forced to generate a caption and also to understand the correlation between words in captions and objects in the input images.

There are a lot of combinations between the vision encoders and the textual decoders. The conventional way to build an image captioning system that follows an encoder-decoder style, is to train from scratch both the vision encoder and the textual decoder. Although there are techniques that can connect a pre-trained vision encoder like ViT [10] or CLIP [21] with a pre-trained textual decoder like BERT [9] or GPT2 [22], the main issue is that each pre-trained

element has been trained in a different high dimensional space. Therefore, the short path is to train from scratch the respective encoder and decoder, with their own model capacities.

In this way, the contemporary approach to build an image captioning system is via the encoder-decoder [32] architecture, which is a main element of the transformer [32] model family. In contrast to the previous image captioning approaches, the encoder-decoder style not only can be more agile and flexible in terms of model complexity and computation resources, but also more effective in terms of performance, outperforming all the former approaches.

2.2.3 ClipCap

Most image captioning systems are built with the conventional way of encoder-decoder architecture. The straightforward solution of encoder-decoder can be beneficial, but it has its own drawbacks, such as the training of both elements from scratch. The training of those elements can consume a lot of computational resources as well as be quite expensive. On the other hand, fine tuning an already pretrained Encoder/Decoder could save a lot of training time as well as take advantage of the enriched representations from the diverse datasets that have been trained on.

An approach, which took advantage of pre-trained elements in its architecture, was proposed in a paper titled "Clipcap: CLIP prefix for Image Captioning" [17] in 2021. The main idea is to utilise a pretrained image encoder like CLIP [21], which has been trained on millions of image text pairs, as well as a pre trained textual decoder like GPT2 [22], which also has been trained in huge publicly available training corpus from the internet, and combine them. In contrast to the conventional way, the combination of two already pre-trained elements can be challenging, as it should create a shared latent space of both vision and text. To achieve that, the authors of the paper suggested a small Transformer network or a tiny Multi-layer perceptron in order to bridge the gap, as the Encoder and the Decoder latent spaces are independent. Usually, the selection between a small trainable transformer or a tiny trainable MLP varies depending on the size of the training dataset and the respective task. However the solution that the authors proposed is a Transformer, which actually is an Encoder with eight blocks and has also the best predictive power among the other options.

The ClipCap [17] approach can be defined as a lightweight architecture, in which its encoder and decoder remain frozen. Therefore, it is an agile and flexible

solution for image captioning tasks, where the training time of a transformer is significantly less in contrast to the conventional image captioning systems. Finally, the alternative approach of Clipcap [17], can be quite competitive in terms of performance, as its predictive power can be leveraged depending on the respective model capacities and the size of the training data.

2.2.4 GIT

As we mentioned, a standard architecture for image captioning is based on the transformer[32] model family, and in the cooperation of the encoder and decoder. The mechanism of the encoder-decoder is heavily relied on the cross attention mechanism, which was defined on the previous sections. In this way, the architecture can manage to link a vision encoder with a textual decoder, without any modification in its base architecture. Although that alternative approaches exist such as the architecture of ClipCap[17], which utilise the power of pre-training process, the vanilla strategy of an encoder-decoder[32] has certain positive aspects.

Several months back, a paper was announced with the title "GIT: A Generative Image-to-text Transformer for Vision and Language" [35] by Microsoft. The GIT [35] architecture follows the conventional architecture of encoder and decoder [32] and the main novelty of the paper is that the GIT model can handle both image captioning tasks and visual question answering tasks as well. Another characteristic of the GIT model is the remarkable performance that achieved in the image captioning as well as in the visual question answering evaluation metrics. The main reason for that performance is the large amount of image-text pairs that the GIT model has been trained on. The vision encoder of GIT architecture is set to the large variant of the CLIP [21] model and the textual decoder is set to the large model variance of the BERT [9] model. Therefore, the large model capacities of the GIT [35] architecture allow the model to score better performance, as the encoder-decoder approach can be considered as resource-data hungry models, without reaching any limitation. The more training data you provide to the model, the better results the model can achieve. In this way, the GIT model has developed enriched and powerful embeddings that can boost its predictive power, outperforming all the previous models and particularly in the image captioning domain.

While the main strategy of the GIT [35] architecture is based on image captioning, the Visual Question Answering tasks can be delivered by GIT [35]. The main training phase of the GIT is set to understand image-text pairs, whereas

the process of visual question answering has been acquired during the fine-tuning phase. Moreover, the GIT [35] architecture doesn't offer a solid visual question answering solution, but only a partial solution mainly through image captioning.

Finally, there are a lot of approaches with respect to image captioning systems. Every approach, conventional or not, has its own advantages and disadvantages and their efficiency heavily depends on the requirements of an image captioning system.

2.3 Visual Question Answering

In the domain of vision language understanding, apart from image captioning systems there are a lot of advancements in the Visual Question Answering [2] as well. It is obvious that a system can be very beneficial if a user can also ask any question to a given image. A detailed caption of an image can be helpful, but what if a user can ask unlimited answers with respect to that image? It seems challenging, but the research community has made remarkable progress to define an approach to Visual Question Answering [2] systems.

Because of the large scope of question-answer pairs, most VQA [2] literatures adopt a certain constraint, in which they define a closed set of answers. Therefore, the majority of VQA systems work with a fixed answer set where exactly one of the possible answers is guaranteed to be correct. In other words, a Visual Question Answering task is transitioned to a multiple choice answer system, which can't generate any answer with respect to a question but only deliver the most likely answer.

In the same way as in the conventional image captioning approaches, the common Visual Question Answering architectures follow a standard technique of utilising a pretrained Image Encoder, extracting its feature vectors, and finally connecting to a LSTM or RNN [28] architecture. The LSTM architecture is often used to decode a question for any given image. The binding between the the image encoder and the LSTM [28] is established through a point wise multiplication layer, following a fully connected Neural Network with several layers and a softmax layer, to obtain probabilities for the final answer. The commonly used VQA architecture is very similar to the conventional Image Captioning architectures except that the VQA architectures should handle as input and a question apart from the image.

The limitation of the Visual Question Answering approaches is obvious and disappointing. A competitive VQA [2] system should generate any potential answer with no constraints as well as have a deeper understanding of an input image. The effect of the transformer[32] architecture has lately had a big impact in the VQA [2] systems, but the adaptation of a closed answer set is still a high standard. Certainly, there is a need for improvement in the Visual Question Answering architectures, overcoming limitations like those and making the whole task less challenging.

2.4 Image Generation

2.4.1 Diffusion Models

In the scope of Image Generation, the Diffusion Models [26] are one of the biggest developments in deep learning in the past several years. Essentially, those generative models can produce any image with respect to a textual description. Therefore, the combinations of descriptions of images can be endless. A synthetic image based on a user prompt can be very impressive and creative. The image generation process can be applied in several domains such as data augmentation, synthetic dataset or even art creation.

Until the rise of the Diffusions [26] models, the prevailing architecture in image synthesis was the Generative Adversarial Networks [6] also known as GANs. This type of network utilises a generator and a discriminator to produce a generative image. Its a iterative process, in which the generator creates an image until the discriminator allows that the produced image is valid and acceptable. However, the recent progressions in the generative domain showed that the diffusion models can beat GANs [6] for image synthesis. In the paper "GLIDE: Towards Photo realistic Image Generation and Editing with Text-Guided Diffusion Models" [18] the GLIDE diffusion model was first introduced. A main advantage of a diffusion process is that its generated images are much more photo realistic in contrast to the former approaches like GANs.

Fundamentally, Diffusion modelss [26] work by destroying training data through the progressive addition of Gaussian noise and then learning to recover the data by reversing this noising process. At inference, the Diffusion modelss [26] generate data by simply passing randomly sampled noise though the learned denoising process.

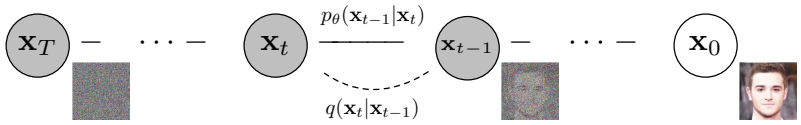


Figure 4: *The backward process of a diffusion model [26], in which we sequentially denoise the input image.*

In more detail, a Diffusions [26] model consists of a forward/diffusion process, in which an image is gradually noised by a sample noise from the Normal Distribution, and a backward/reverse diffusion process, in which the noise is gradually removed. Therefore, the model learns to fully predict the noise at each step and it needs several steps in order to get high quality images. The utilisation of the noising/denoising process is based on the fully convolutional Unet architecture [25], in which the input and the output image have the same dimensions. In the Unet architecture, we initially downsample the original image via convolution layers and then we upsample it back again. The backward diffusion process takes the textual description into account, which is encoded and modified as an embedding. Moreover, each attention layer of the Unet [25] is also attending to the respective embedding. In this way, the diffusion models can have a better guidance of the generated images. Finally, the diffusion models learn to predict the noise itself and the loss function is calculated between the predicted noise ($\epsilon_\theta(x_t, t)$) and the actual noise (ϵ) as follows:

$$\mathrm{L}_{\mathrm{DM}} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta\left(x_t, t\right)\|_2^2\right]. \tag{2.3}$$

To sum up, Diffusion models have impressive image generation capabilities and can easily outperform GANs [6], with more photo realism in its generations. Additionally, the Generative Adversarial Network requires a delicate balance between the discriminator and the generator and it is highly sensitive to even minor changes. However, the diffusion process may have less parameters than the GANs [6], its inference time period is much longer, which can be stated as a certain limitation of the process.

2.4.2 Latent Diffusion Models

The architecture of Diffusion models was considered state of the art for generative images, however the research community has successfully made certain improvements on the backbone of its process. Although the diffusion model was considered to be scalable in contrast to previous approaches, there were certain limitations in the process such as to have long inference time and to be computationally expensive. In the paper "High-Resolution Image Synthesis with Latent Diffusion Models" [24] a new version of diffusion model was introduced, utilizing the Latent Diffusion [24] approach. The main goal of that innovative approach was to handle those limitations and deliver a diffusion process that can be much faster and require less resources.

The main culprits of the high computational cost of the diffusion process were the Unet architecture in conjunction with the dimensions of an image. Because it is too costly to transform and retain an high resolution image in the Unet architecture [25], the authors of the Glide [18] diffusion model experimented on low dimensional images. That is the main reason why the authors of the "High-Resolution Image Synthesis with Latent Diffusion Models" [24], invoked the Variational AutoEncoders in the diffusion process. The core concept of the Latent Diffusion Models [24] is to surrender the idea of working in the image space and work on a latent space instead. In this way, the process does not utilise the original dimension of the image but a lower dimensional representation of it. The Variational AutoEncoder is essentially an Encoder-Decoder network that is trained to encode images in a lower-dimensional space and then decode them back to reconstruct the original high resolution image. In the Latent Diffusion architecture, the Variational AutoEncoder is trained separately before the diffusion process. Once it is trained, it remains frozen, while the diffusion model is trained in its lower dimensional latent space. Nevertheless, the remaining process persists intact and the diffusion model predicts the noise of the image with respect to its description. The loss function of the latent diffusion models is calculated as follows:

$$\mathrm{L}_{\mathrm{LDM}} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta\left(z_t, t\right)\|_2^2\right]. \tag{2.4}$$

To sum up, the latent diffusion approach allows the model to be faster at inference and not to require a lot of resources. Additionally, a major asset is that the latent diffusion models can process high resolution images and finally generate images with better quality. Lastly, the Stable Diffusion [24] is an open-source

generative model, which is based on the latent diffusion approach and can produce impressive synthetic images. All of our diffusion experiments in our work were set with a particular version of the Stable Diffusion [24] model.

Chapter 3

Architecture

3.1 Main Model

There are a lot of strategies to design an effective image captioning system. Either conventional or alternative approaches, all should deliver the same task. The main approach that we follow in order to conduct our experiments is based on the usage of pretrained models. The main idea behind the architecture is to employ pretrained models such as vision pretrained models and large language models, in order to combine them so they can generate meaningful captions to a visual content. A pre-trained language model such as GPT2 [22] along with a large vision encoder such as CLIP [21], can have a wide understanding of both visual and textual data.

A key element of our architecture is the binding between the vision and the language model is established through a Mapping Network, which is either a tiny multi layer perceptron or a small transformer [32]. We should mention that any key component can be finetuned or remain frozen, but the optimal solution is to keep the main elements frozen (CLIP [21] and GPT2 [22]) and to train only the mapping network, allowing a lighter architecture with less trainable parameters. Hence, our approach only requires rather quick training to produce a competitive model. Without additional annotations or pretraining, it efficiently generates enriched captions. In contrast to previous approaches that have been proposed for image captioning, which usually require object detection, large datasets or bigger architectures and eventually lead to extensive training time and a large number of parameters, our model can achieve comparable results to state-of-the-art methods on challenging datasets as Conceptual Captions and COCO [16], while it is simpler, faster and lighter. The agility of our proposed model in terms

of architecture, can be beneficial as any user can train from scratch a competitive image captioning model on any dataset or even from various datasets domains.

The vision encoder in our proposition is the base model variant of the CLIP [21] model. The CLIP [21] model is designed for image-text retrieval and it is trained in more than 400 million image-text pairs using a contrastive loss. CLIP [21] is developed to impose a shared representation for both images and text prompts. Hence, its visual and textual representations are well correlated and since CLIP [21] was trained over an extremely large number of images, we took the advantage of its rich embeddings.

As the textual decoder of our framework is a large language model, the textual encoder of CLIP [21] is not utilised, since there is no input text and the output text is generated by the language model. Thus, we make use of a powerful autoregressive language model such as GPT2 [22].

The input of the Mapping Network is actually the CLIP's embeddings and consecutively the output of the Mapping Network is the input of the text decoder. Essentially, the Mapping Network, or in other words the trainable Transformer [32], projects the CLIP embeddings along with learnable constants in order to produce a prefix for each caption. In more detail, these particular constants have a dual role not only to retrieve meaning information from the CLIP embedding through Multi Head Attention, but also to learn to adjust the fixed language model to the new data. The prefix, which is the output of the transformer, is basically a fixed size embedding sequence, which is then concatenated to the caption embeddings during the training phase. Finally, the result of the concatenation is the input to the language model in order to generate tokens. The main challenge of this particular approach is to translate or "map" the representations of CLIP with the representations of the language model. We should mention that both CLIP and GPT2 [22] have rich and diverse representations, but their latent spaces are independent, as they were not jointly trained. That is the main reason why the key component of this architecture is the Mapping Network, whose main goal is to translate the CLIP [21] embedding to the GPT-2 [22] space.

As we described above, there are several options that we propose for a Mapping Network, either a Multi Layer Perceptron or a Transformer [32]. Surprisingly, the most efficient approach, which also achieved the best results, is the one with a frozen GPT2 [22] and a trainable Transformer. The alternative approach with a MLP as a Mapping Network and a fine tuned GPT2 marked lower per-

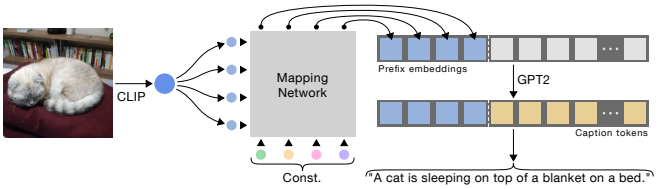


Figure 5: *Our plain architecture for image captioning as presented in Clipcap [17]. We train a lightweiht transformer [2]-based mapping network to generate prefix embeddings from CLIP [21] embeddings of the input image. As a last step, the frozen GPT2 [22] language model generates a caption with respect to the prefix.*

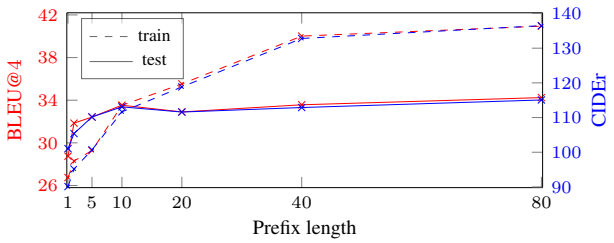


Figure 6: *The impact of the prefix length on the captioning performance in terms of BLEU[20] and CIDEr [33] scores, as shown in the Clipcap [17].*

formance. Additionally a derivative of that approach, where the language model is also fine-tuned, seems to be less effective. Mainly because of the fact that the underlying mapping is less challenging, as we easily control both networks. Therefore, the best possible option is a small Transformer with eight encoder blocks and a frozen GPT-2 model.

As the Mapping Network produces a certain prefix with respect to its CLIP embeddings [21], its respective prefix length is negotiable. The prefix length can take various length values and each setting will have a different impact in the performance of our model. The best recommendation for the prefix length is to set it to 10, as it marks the highest evaluation metrics and also avoids overfitting issues. In order we to make such conclusions, we provided a thorough analysis of the required prefix lengths and the effects that they produce in the language model.

To sum up, having outlined all the core elements of our framework, its main advantage is agility. A lightweight solution with significantly less parameters

in contrast to other approaches. How quickly can be trained, how easily can adapt to new input data and how efficient and competitive it can be with such a limited training interval.

Chapter 4

Implementation

4.1 Key Adjustment

We have reached a certain point, where we can propose an efficient image captioning system. However our initial motivation was to implement a unified image-to-text model, which can deliver not only descriptive captions, but also can handle any question that user can provide with respect to an input image. In this way, we can have a complete and effective vision-language model that can be integrated in any framework.

In more detail, our image captioning approach takes the relative CLIP's embedding [21] of an image, produces a prefix and concatenates it with the original caption. In order to generate a caption, the attention mask of the GPT2 [22] language model should attend to the whole concatenated tensor of its prefix and its tokens. The padding technique is a helpful method to retain a global length of the tokens. In an attempt to apply the padding technique we should define a maximum length of the sequence tokens. The length of the tensor is defined by the minimum between the average sequence length along with an inflated standard deviation and the maximum sequence length of the captions.

The transition from image captioning to visual question answering [2] has not a straightforward solution. A vital adaptation that we applied was to extend and modify the attention mask so that it can encompass both the question and the answer of any training example. In this way, we concatenate the prefix, the question and the answer along with the special token id of the GPT2 [22] tokenizer that defines the end of sentence. With that delicate modification we accomplished to retain the "schema" between our image captioning system and

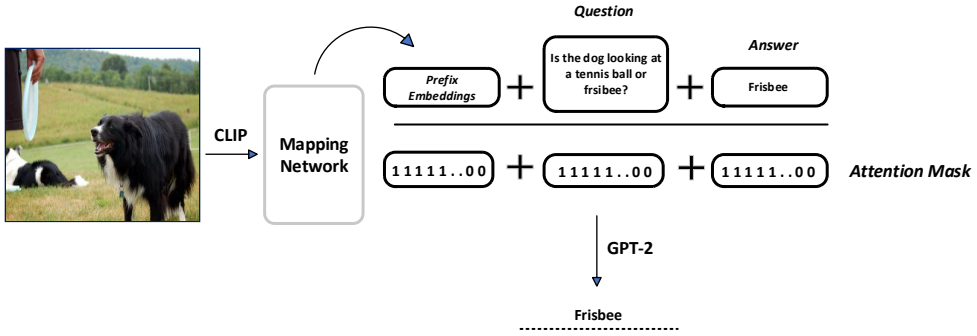


Figure 7: *Our proposed modified architecture for VQA tasks. We refine the attention masks specifically for question-answer pairs, not captions. This strategic adjustment enables the model to generate answers more effectively and accurately.*

our potential visual question answering approach. After the adjustment of the GPT2's attention mask, the next steps remain intact. We calculate the cross entropy loss function in each step, from the predicted logits of the GPT2 [22] and the respective target tokens.

As a result, we altered our image captioning approach so that it can handle visual question answering tasks. The key point of this VQA [2] proposal is that it can generate its output and not classify it. Thus, having a Visual Question Answering architecture that can generate its answer word by word in a autoregressive manner can be considered as an innovative solution, that exceeds the standards of any VQA system.

4.2 Multitask Learning

Our initial milestone was to define two distinct approaches for image captioning and visual question answering with respect to the same model structure. Hence, our next goal is to establish a unified model that integrates our two main approaches. To achieve that we have to employ the multi task learning technique [5], in which a model can be trained on several tasks in parallel. It may seems a challenging task, but the actual key points of multi task learning are relatively easy to implement.

Typically, a model can be trained to do a single task which is convenient, but we

want to use a single model to solve multiple problems for various reasons, such as efficiency and better generalisation. The main constraint of multi task learning is that the selected tasks of any model should be relative. It is common to notice in Computer vision literature, where vision models are trained in parallel via multi task learning for tasks such as image classification, object detection and image segmentation. An important step in order to implement multi task learning is to share some of the learned parameters or even layers of the model between the respective tasks. In this way, we can define a unified multi task model, where we not only can improve its efficiency, but also save in general memory, computations and power. Intuitively, the closer the tasks are the more features it can share [31]. In our case, the image captioning and visual question answering tasks are very similar. Therefore, our unified model can learn different semantics of its respective task, thus boosting its general performance.

Another requirement of multi-task learning is the need to define an overall loss function, which combines multiple loss functions corresponding to different tasks. To maintain the scalability of the model, it is essential to assign weights to the relative tasks. If one loss function significantly outweighs the others, it could dominate the training process, even though certain loss functions might be more crucial or converge faster. Therefore, it is crucial to adjust the weights on each loss function through trial and error. The overall loss function (L_{Total}) is calculated as a weighted sum of individual losses (L_1 and L_2) with the constraint that the weights (w_1 and w_2) must sum up to 1, as shown in the equation below:

$$L_{Total} = w_1 \cdot L_1 + w_2 \cdot L_2 \text{ , where } w_1 + w_2 = 1 \tag{4.1}$$

In order to avoid any confusion in our work, we designed two distinct data loaders, each one designed for its task. Therefore, in our multi task implementation we have an image captioning and a visual question answering data loader, in which all the required transformations are being made. Iteratively, we fetch batches from the data loaders so that our model can learn its tasks alternately. In this point we have to mention that in our case only the Mapping Network can share its learnable layers in the scope of multi task learning. The other main elements of our architecture remain frozen (GPT2). As a final step we calculate the cross entropy loss of every task and we adjust them with weights so that we can have the overall loss function. We conducted several experiments in our multi task approach in order to define the optimal weights of its tasks. The main criterion was the combined performance of our unified model in its relative tasks.

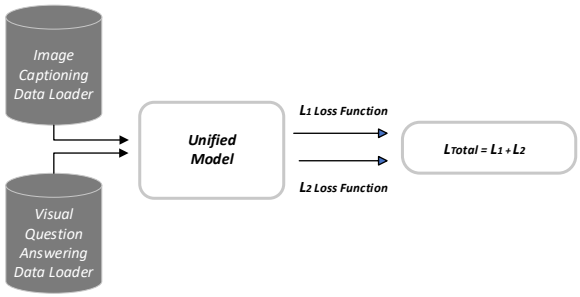


Figure 8: *A high level overview of our multi task learning architecture. In the training phase, the data loaders for image captioning and visual question answering operate in tandem, delivering batches to the model alternately. This approach allows the model to derive its final loss by aggregating the individual losses.*

In conclusion, integrating two distinct models—image captioning and visual question answering—into one can be highly advantageous. A unified model that not only can handle two different tasks, but also can have better generalisation is impressive. As our tasks are different but relative, we had to try to define a model so that it can learn different representations of its respective task, enhancing its predictive power.

4.3 Synthetic Datasets via Diffusion

4.3.1 Reproduction of the Datasets

A main aspect of our work is the extensive usage of the generative diffusion models. In order to boost the predictive capabilities of our image captioning - visual question answering framework, we employed a pre-trained Stable diffusion model [24] so that we generate images by our training descriptions. In this way we can recreate from scratch our datasets, so that we can provide more training observations to our models. Unfortunately, this concept applies only for our image captioning training examples as the question answer pairs can not facilitate the diffusion process for obvious reasons.

In many cases, a generative image may not be as realistic as it should be.

Therefore, we should find ways to enhance our diffusion process in order to produce the best possible results. Hence, we can apply a certain prefix to our descriptions such as "High photo realistic" so that we can force the respective diffusion model to generate more realistic outputs. Alternatively, there can be a finetuning in the defaults settings of the stable diffusion pipeline. The main diffusion model that we employed in our process is the "Compvis/stable-diffusion-v1-4" [24], which as its name suggests is a Stable diffusion model.

Moreover, we utilised our fine tuned diffusion process for all of our training datasets. As a result, we have synthetic datasets with newly generated images based on their captions, from the original datasets of COCO [16], VizWiz [12] and Text Caps [29]. In more technical detail, with respect to our generated images, we calculated its Clip embeddings and then we concatenated those generated features with the original features from the respective dataset. We should mention that the process of generating images is computationally expensive and time consuming. For this reason, for large datasets like the COCO dataset we applied a threshold of generating 100 thousand images, whereas for smaller datasets such as VizWiz and Text Caps we reproduced the whole dataset. The synthetic datasets based on the smaller datasets have approximately 100 thousand generated images, therefore we can assume that we retain homogeneity in our diffusion approach. We present in Figure 9,a sample of generated images from the COCO dataset,with respect to their original images and captions.

Enriching our arsenal in the scope of training efficiency is more than significant. Populating our training data with newly generated images based on the defined captions, can be considered as a valuable asset in the whole process.

4.3.2 Clipscore

Although we have created and fine tuned our diffusion pipeline, we tried to optimise it even further. The main reason is that a one shot generated image may not be so sufficient to be integrated as a training example. Many generated images can be so noisy that it can be considered as a distrurbed image. Additionally, a lot of generated images can be not so realistic after all. The more complex is the caption of the image, the more noisy the generated image will be. In order to prevent this issue we advanced our process utilising the resources of CLIP [21].
To counteract the sensitivity of diffusion models [26] to noise, we populated the generated images. Instead of one shot generated image per caption, we generated five potential images per caption. Therefore, we have five candidate images


"A closeup of a red fire hydrant including the chains."

"A little dog is laying on a man's arm."


"A bathroom with a border of butterflies and blue paint on the walls above it."

Figure 9: *A sample of generated images from COCO [16], with captions shown underneath. (Left) Original image; (Right) Generated image.*

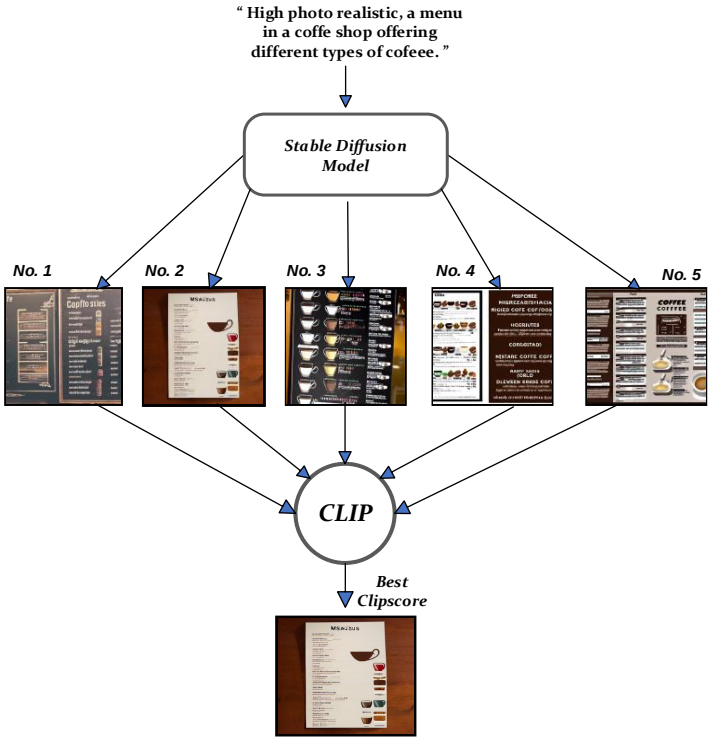


Figure 10: *An illustration of our Clipscore pipeline. Employing a stable diffusion model, we generate five candidate images corresponding to a textual input. The Clipscore for each generated image is computed as the cosine similarity between its CLIP [21] embeddings and the original caption. Ultimately, the image with the highest Clipscore value is chosen.*

and we should pick the more representative one. To achieve this, we employed the CLIP model, whose main attribute is to calculate the cosine similarity of an image-text pair. This process is also known as ClipScore and it measures how similar an image is with a text prompt, that actually describes the respective image.Thus, calculating the Clipscore values of the five candidate images we can easily pick the best possible generated image.

To sum up, we strengthen our diffusion pipeline in order to have more representative and reliable training observations in our datasets. Our ClipScore approach is a lightweight and optimal solution that can be applied in many cases that require the usage of generated images. It is an efficient and effective

technique that produces not only realistic but also qualitative generated images.

Chapter 5

Experiments

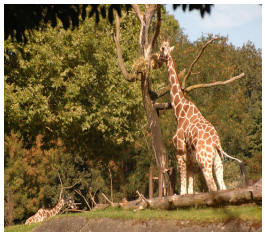
5.1 Datasets

5.1.1 COCO

The COCO [16] dataset is a large-scale object detection, segmentation and captioning dataset. For obvious reasons, we utilised only the captioning sub dataset. In more detail, the COCO dataset has several versions, we selected the COCO - 2014 challenge, which was presented in the CVPR conference. Essentially, this is a collection of image-text pairs, where each caption describes the respective image in a unique way. The training dataset consists of 82.783 images and on average there are about five distinct captions per image, with a total of 414.113 training observations. In the same way, the validation dataset consists of 40.504 images, with an overall of 202.654 observations. In the same way as in the ClipCap [17],we didn't utilise the test dataset. Thus, we performed our final evaluation metrics only to the validation dataset.

Additionally, the COCO dataset can provide a Visual Question Answering dataset, the VQA challenge. The VQA [2] challenge has also several versions, we picked the most recent the VQA-v2 challenge [2]. The majority of this collection relies on the COCO images. The training and the validation datasets are almost identical with 82.783 and 40.504 images respectively. The annotations of this dataset provide pairwise a certain question and several answers with respect to an image. We picked the only one answer from the annotations, as the rest answers can be considered as duplicate answers. However, the VQAv2 dataset provides a test dataset, where we evaluate our VQA trained models. The test dataset consists of 81.434 images with a certain question per image.

GT: a dog is looking at a blue frisbee.
Q: Is the dog looking at a tennis ball or frisbee?
A: frisbee



GT: a giraffe eating food from top of the tree
Q: How many animals are in this photo?
A: 2

GT: a man is doing a trick on
a skateboard
Q: Does the guy have a tattoo?
A: yes

GT: A person on skis in the snow with trees in back.
Q: What is she holding?
A: poles

GT: A painting of an array of flowers in a vase.
Q: What color are the flowers?
A: yellow and white

GT: A waterfront walkway and garden area next to
a river.
Q: Is the water calm?
A: yes

Figure 11: *A sample from COCO [16] dataset, along with its captions and question-pairs.*

5.1.2 TextCaps

The official dataset of TextCaps [29], as its title suggests, is designed especially for Image Captioning. We focused on the latest version of Textcaps v0.1, which was presented in the 2021 TextCaps challenge. The dataset consists of train, validation and test sets. However, only the training and the validation datasets were utilised. The training set entails 21.953 distinct images, whereas the val-

idation set entails 3.166 distinct images. There are exactly 5 different captions per image, where each caption describes the respective visual content with a different perspective. Totally, there are 109.765 image-text pairs in the training dataset and 15.830 pairs in the validation dataset.

As we described, the evaluation of our models was applied on the validation dataset. However, all the evaluation metrics were obtained by a popular platform, where all the Text Caps challenges are hosted. EvalAI [36], is an open-source framework that facilitates the evaluation of machine learning models by providing an interface for hosting and participating in various AI challenges, thereby promoting collaborations and benchmarking in the research community. Therefore, all the respective teams that participate in the 2021 TextCapts challenge, should submit their predictions on the validation dataset with a certain format, in order to receive their corresponding evaluation metrics.

5.1.3 VizWiz

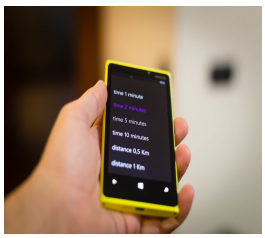
The VizWiz [12] dataset is a multidimensional dataset with a lot of domains, such as Image Classification, Object Detection, Image captioning and Visual Question Answering. Therefore, we make our experiments both in Image Captioning and Visual Question Answering subdatasets. The original dataset split into three categories of train, validation and test dataset. In this case, as usual the training dataset is set for training the respective models, the validation dataset for evaluating across the epochs and the test dataset is utilised to evaluate the performance of the models. The training dataset consists of 23.431 distinct images with on average 5 captions per image and a total of 117.155 training observations, whereas the validation dataset consists of 7.750 distinct images and a total of 38.750 observations. Finally, the test dataset consists of 8.000 distinct images and 40.000 captions. However, the VizWiz dataset can be considered as a “noisy” dataset, not only because of the errors in its descriptions, but also because there are a lot of blurry photos with not semantic meaning at all. In many observations, there is a boolean flag of ”is-reject”, where you can reject that particular observation, as we did in our experiments.

In the same way as in the TextCaps [29] dataset, we participated in the VizWiz-VQA as well as in the VizWiz-Caption challenge through the EvalAI [36] platform. As it is mandatory, if a team aims to publish its evaluation scores for any particular challenge, it should provide the predictions with respect to its dataset.

GT: Man wearing a blue, Blue Jays jersey out on the grass

GT: he library is a brick building with a blue banner on it.

GT: A sign in a foreign language the sign says no fishing



GT: A smartphone with times and distances on the screen

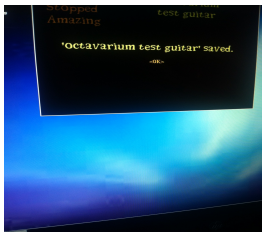
GT: A watch with a brown leather band shows 10:38.

GT: A child at a table with three Starbucks beverages.

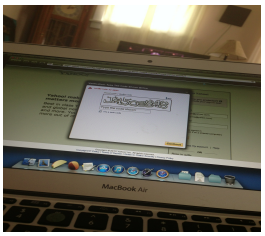
Figure 12: *A sample from TextCaps [29] dataset, along with its captions and question-pairs.*

Therefore, we followed all the necessary steps to provide our results of the test datasets for the VizWiz-VQA and VizWiz-Caption challenges respectively.

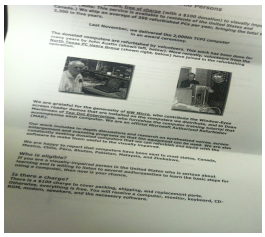
GT: Outdoor US Postal Service letter collection box
Q: What is this?
A: Post office box drop



GT: A computer screen with a black window
Q: What does the text say?
A: octavarium



GT: An advertising message appears on a computer screen.
Q: Is there a button to push?
A: No



GT: Piece of white paper with black font lettering and two pictures on it
Q: What does it say?
A: About giving away refurbished computers

GT: A person is sitting in achar facing the stage area.
Q: What is picture of?
A: Person sitting in front stage

GT: A page of written characters with a pair of metal rim glasses on top.
Q: What is this?
A: Arabic writing.

Figure 13: *A sample from VizWiz [12] dataset, along with its captions and question-pairs.*

5.2 Evaluation Metrics

A required step to monitor and validate the performance of our models is to apply evaluation metrics on every training phase. In this way we evaluate the quality of our generated text outputs as well as monitor the general improvement of our models in the scope of training efficiency. In our work, our primary output with respect to the models is textual, as we have defined various image to text models both in image captioning and visual question answering tasks. In general, our textual outputs are either descriptive captions or straight answers with respect to image-question pairs. The most common evaluation metrics that are applied on textual data, are based from NLP applications such as text summarization and machine translation.

Thus, all the metrics that we used for validation are widely used. Our main metrics are Rouge-L [15], Bleu [20], METEOR [3] and CIDEr [33], which we will describe extensively in the following submodules. As we have discussed above a lot of research works and publications employ a certain framework, in which they provide their outputs in a specific format and internally many evaluation metrics are calculated. In this way, the research community not only can retain transparency in their final results, but also can avoid any marginal error through the manual calculation of the metrics. The EvalAI [36] framework is a trustworthy environment in which many challenges with respect to datasets and tasks are being hosted. Therefore, a public challenge of a task is required for any participant to upload its results. The majority of our tasks in our work are represented as an EvalAI challenge. Except for COCO [16] image captioning which is not represented, all the other respective tasks are being hosted in the EvalAI framework. From our side we provided our final results in the EvalAI infrastructure in order to be consistent and also have clarity in our achievements.

As we have outlined in the previous sub-modules, the generated outputs of the VQA [2] task are computed autoregressively. However it is not commonly used to apply evaluation metrics such as Rouge-L and Bleu. It is feasible to calculate all these metrics, but we aligned with the concept of any VQA challenge in the EvalAI framework to handle these tasks as classification tasks. Accuracy is the main metric of any VQA challenge [2] and it is calculated internally via a specific formula as well as the assistance of human annotations.

$$\text{Accuracy } = \min\left(\frac{\#\text{ humans that provided that answer}}{3}, 1\right) \tag{5.1}$$

5.2.1 Bleu

For many NLP tasks, we can use common metrics like accuracy or F1-score, but what do you want to measure the quality of a generated output? Essentially those common metrics fail to capture the quality of a text output. Bleu [20] is probably the most common metric with respect to text evaluation.Its extensive usage belongs to the machine translation domain, but it is applicable for image captioning tasks as well.

The main concept of Bleu [20] metric is to compare the n-grams of the generated captions with the n-grams of the referenced captions. A n-gram is just a set of ordered words in a sequence. With respect to the length of a n-gram, we can define a unigram, bigram, trigram or even a 4-gram. As their titles suggest, the prefix defines the length of a n-gram. The computation of Bleu is based essentially on the n-gram precision. A n-gram precision is the ratio of the number of words that match between the generated and referenced caption to the number of words of the generated caption. Originally, the precision metric ranges from zero to 1, so higher precision means more qualitative captions. The unigram precision formula is calculated as follows:

$$\text{Uni-gram Precision} = \frac{\text{Number of words that match}}{\text{Number of words in generated caption}} \tag{5.2}$$

In many cases we can notice over-regenation of outputs or in other words repeated words. To handle this issue, Bleu metric has integrated a modified n-gram precision that limits the number of times that it counts a specific word. Mainly because we should take into account the order of a sentence, we should apply all of the variations of n-grams. Last step of Bleu calculation is to apply the geometric mean of the respective n-grams. Therefore we can also have variations of the Bleu metric depending on the range of the n-gram precisions.Consecutively, with respect to the n-gram precisions we can define Bleu-1, Bleu-2, Bleu-3 and Bleu-4 evaluation metrics as follows:

$$\text{BLEU-1} = p_1 \tag{5.3}$$

$$\text{BLEU-2} = \sqrt[2]{p_1 \cdot p_2} \tag{5.4}$$

$$\text{BLEU-3} = \sqrt[3]{p_1 \cdot p_2 \cdot p_3} \tag{5.5}$$

$$\text{BLEU-4} = \sqrt[4]{p_1 \cdot p_2 \cdot p_3 \cdot p_4} \tag{5.6}$$

To sum up, the Bleu metric has its own advantages and disadvantages. It is very effective, commonly used by the research community and straightforward to compute it. However, it doesn't incorporate the semantics of a sentence, which is very important especially for image captioning and also it doesn't consider the recall of the n-grams.

5.2.2 Rouge

In the same scope as Bleu's, the calculation of Rouge metric is based on the comparison of the n-grams between the generated and the referenced captions.Apart from machine translation tasks, it is widely used also in text summarizations tasks. In contrast to Bleu metric, Rouge [15] takes into account both the precision and recall of the n-grams within a sentence.

The recall n-gram is just the ratio of the number of words that match between the generated and the referenced output to the number of words in the referenced output. To combine the recall and the precision of n-grams we can utilise the F1-score, which is just the harmonic mean of those two elements. Therefore the Rouge-1 metric is actually the F1-score of the uni-grams. In the same way, we can employ the bi-grams so that we can define the Rouge-2 metric. However, in our work we utilise neither Rouge-1 nor Rouge-2 metrics. We applied the enhanced Rouge-L [15] metric, which does not compare n-grams, but instead treats each summary as a sequence of words. Essentially, in the calculation of recall and precision, it looks at the longest common sub-sequence ,also known as LCS, within a sentence.The longest common sub-sequence is a sequence that appears in the same relative order, but not necessarily contiguous. Thus, we can calculate the Rouge-L recall and precision equations as follows:

$$\text{ROUGE-L Recall} = \frac{\text{LCS(gen,ref)}}{\text{Number of words in referenced captions}} \tag{5.7}$$

$$\text{ROUGE-L Precision} = \frac{\text{LCS(gen,ref)}}{\text{Number of words in generated captions}} \tag{5.8}$$

The main advantage of Rouge-L over Rouge-1 and Rouge-2, is that it doesn't depend on consecutive n-grams matches, so it can capture the sentence structure more accurately.

5.2.3 Meteor

In the same rationale with the calculations of BLEU and Rouge, the METEOR [3] evaluation metric is based on the comparisons of the uni-grams between the referenced and generated caption.It combines both precision and recall uni-grams, computing in its final step the respective Fmean score.

Its main attribute is that it can handle semantic similarity, as it has its own synonymy database that accounts for synonyms or even for paraphrases.Furthermore, it calculates the root of every word in the captions, invoking the stemming technique.The main difference of the METEOR [3] metric in contrast to the rest metrics is that it adds more weight to the recall of an uni-gram.In this way, the Fmean score is altered by certain coefficients and its formula is displayed below:

$$\mathrm{F}_{mean} = \frac{10PR}{R + 9P} \tag{5.9}$$

Additionally, the METEOR evaluation metric incorporates a penalty function to account for the ordering within a sequence. The penalty function p is calculated as:

$$\mathrm{p} = 0.5 \left(\frac{c}{u_m} \right)^3 \tag{5.10}$$

where c represents the number of matching uni-grams, and u_m is the total number of uni-grams. The final METEOR score is obtained by multiplying the Fmean-score with $1 - p$, where $\mathrm{METEOR} = \mathrm{F}_{\mathrm{mean}}(1 - p)$.

5.2.4 CIDEr

In contrast to all previous evaluation metrics, the CIDEr [33] metric is the most interesting and important measurement. It is especially designed for image captioning tasks and all the research community put an extra focus on it, to validate the performance of their models.It utilises all the n-grams variations, ranging from uni-gram to 4-gram.

CIDEr [33] metric is an automatic metric used to evaluate the quality of machine generated captions. Mainly, it measures the similarity between a generated caption and a set of reference captions. By the term "automatic", we mean that it doesn't require human evaluation at all, as it can handle context similarity itself.Its main attribute is that it is consensus based, meaning that it is designed to capture the consensus view of the reference texts. In other words, it gives higher scores to generated captions that are similar to multiple reference

captions rather than just one reference caption.

The CIDEr [33] metric integrates stemming, computing the root word in both generated and reference captions. CIDEr [33] represents each sentence using its set of n-grams and computes the overlap between the n-gram sets of the generated and reference sentence. It employs the TF-IDF [30] approach to assign weights to the n-grams based on their importance. The cosine similarity of each n-gram of length n adjusted with its respective TF-IDF [30] weights is calculated using the formula:

$$\mathrm{CIDEr}_n\left(c_i, S_i\right) = \frac{1}{m} \sum_j \frac{\boldsymbol{g}^n\left(c_i\right) \cdot \boldsymbol{g}^n\left(s_{ij}\right)}{\left\|\boldsymbol{g}^n\left(c_i\right)\right\| \left\|\boldsymbol{g}^n\left(s_{ij}\right)\right\|} \tag{5.11}$$

where c_i is the generated caption, S_i is the set of reference captions, m is the number of reference captions, and n represents the length of n-grams. The CIDEr metric for a caption c_i is then calculated as:

$$\mathrm{CIDEr}\left(c_i, S_i\right) = \sum_{n=1}^{N} w_n \mathrm{CIDEr}_n\left(c_i, S_i\right) \tag{5.12}$$

5.3 Experiments Setup

In all of our experiments we tried to be consistent and to maintain a homogeneity. Therefore, we followed a global approach to define settings and parameters in our implementations. However, with respect to our tasks and to our underlying model capacities, certain settings may be modified.

The majority of our implementations utilised the powerful NVIDIA A100 32GB RAM as its main GPU. The internal default settings of the Mapping Network are an integrated Transformer [32] with 8 Encoder blocks and the Clip prefix of the Mapping Network output to have a length of 10. The main criterion of our tasks, either image captioning or visual question answering, is set to cross entropy, whereas their training phase lasts for about 10 epochs. For optimization, we use AdamW with weight decay fix, with a learning rate of 2e-5 and 5000 warm up steps. The batch size of the processes is based on which GPT-2 [22] decoder variation we will use. Therefore, the GPT-2 decoder has a batch size of 23, whereas the GPT-2 XL decoder has a batch size of 8.

In our diffusion generating process, as we have outlined, our main diffusion model is the "Compvis/stable-diffusion-v1-4" [24]. Nevertheless, the number

of inference steps in our stable diffusion pipeline have been modified. We have altered the argument from 50 to 100. In general, results are better the more steps you use, however the more steps, the longer the generation takes. Furthermore, we have kept the default guidance scale parameter of the pipeline. Guidance scale is a parameter that can increase the adherence to input textual prompt of the user. Thus, following our adjusted settings, we have accepted a trade off in our diffusion process, between long inference time periods and quality in our generated images.

Chapter 6

Results

Our main goal in all of our experiments was to explore all the aspects of our proposed approaches. We tried to execute all the possible combinations to validate which is the optimal approach in terms of performance. Through this extensive investigation process we confirmed the advantages and limitations of each approach, trying to consolidate the most effective solution for both image captioning and visual question answering tasks.

As we have discussed in the previous sections, our straightforward experiments follow our proposed vanilla architecture for image captioning and visual question answering. Optionally, these two tasks can be combined and trained in parallel via multi task learning [5]. In this way, the multi task learning technique can be applied in the respective architectures and it can offer a significant dimension in our investigation process. Furthermore, the diffusion process can be very useful for generating additional training data for our experiments. We provide 2 main variations for the diffusion process, the simple diffusion process, , also known as 'One-shot generation', which generates a single image with respect to the input desciprtion and the Clipscore diffusion process (Section 4.3.2), which generates five potential images and picks the highest scored image. All of our techniques can contribute, but also can affect the performance of a model.

A plain modification in our architecture is to alter the pretrained model variations in our architecture. Although our vision model Clip [21] has been remained intact through all of our experiments, our decoder GPT2 [22] has been adjusted. Obviously, our expectations were confirmed that the XL model variant of GPT2 [22], would have more predictive power in their textual generations as it has bigger capacity than our default decoder (GPT2-Small). In this way,

we have added another option in our experiments, to leverage the power of GPT2 [22] language model.

6.1 Comparison

As we have discussed in our previous sections, many approaches have been proposed both for image captioning and visual question answering. The architectures may vary significantly, with different vision encoders and language models as well as with internal techniques in the architecture, which an affect the model's performance. However, the evaluation metrics of a model can be considered as the main criterion, to conclude which approach should be adapted for our respective tasks.

At a quick glance from the Table 6.1, we can notice that many models that considered to be state-of-the-art have lower performance than our baseline approaches for image captioning. Recent suggested models, such as KOSMOS-1[13] model, which has also the same number of parameters as in our models, marked a lower CIDEr[33] score in the COCO[16] validation dataset. Therefore, we can notice a significant gap in terms of performance, between prior SOA models and our proposed models. Our main models MTL_DF and MTL_CS, which were trained though multi task learning both for image captioning and visual question answering outperform all the former approaches. The main difference of those models is that the former has been trained with 'one-shot generation' diffusion process, whereas the latter with the diffusion clipscore process that we described in our previous sections. In more detail, the MTL_CS model achieved a slightly better CIDEr score of 99.67% in contrast to MTL_DF which achieved a CIDEr score of 98.56%.

Table 6.1: *Comparison with prior SOA models on image captioning on COCO [16]. Our model MTL_CS, trained through multi-task-learning with additional generated images, obtained the maximum CIDEr and Bleu-4 scores.*

Method	Vision_Model	Language_Model	Params	B4	CIDEr
VLKD[7]	CLIP ViT-B/16	BART-L	0.5B	16.7	58.3
KOSMOS-1[13]	CLIP ViT-L/14	Transformer	1.6B	-	84.7
CAPDEC[19]	CLIP RN50	GPT2-L	1.1B	26.4	91.8
VIRTEX[8]	RN101	Transformer	-	-	94
Baseline	CLIP ViT-B/32	GPT2	0.2B	27.98	90.91
Baseline	CLIP ViT-B/32	GPT2-XL	1.6B	28.16	91.87
MTL_DF	CLIP ViT-B/32	GPT2-XL	1.6B	29.66	98.56
MTL_CS	CLIP ViT-B/32	GPT2-XL	1.6B	**29.87**	**99.67**

In the same manner, in the scope of VQAv2 [2] dataset, the MTL_CS model achieved a competitive performance in the visual question answering as well. As we can see from the Table 6.2, our main model has successfully gained the ability to capture the semantics of an image with respect to a question. Moreover, the model marked a sufficient Accuracy score of 60% in contrast to other models. We have to mention that other architectures such as the multi modal Flamingo[1], which was introduced by Deep Mind and has a heavyweight architecture with more than 80 billion parameters, achieved a lower Accuracy score in the VQAv2 [2] challenge.

Table 6.2: *Comparison with prior SOA models on visual question answering on VQAv2 [2]. Our model MTL_CS, trained though multi-task-learning with additional generated images, achieved a remarkable Accuracy.*

Method	Vision_Model	Language_Model	Params	Accuracy
KOSMOS-1[13]	CLIP ViT-L/14	Transformer	1.6B	51.0
FLAMINGO[1]	NFNet-F6	Chinchilla	80B	56.3
BLIP-2[14] [1]	ViT-L	FlanT5 XL	3.3B	62.3
MCB[11]	RN152	LSTM	-	**64.7**
Baseline	CLIP ViT-B/32	GPT2	0.2B	51.7
MTL_DF	CLIP ViT-B/32	GPT2-XL	1.6B	59.24
MTL_CS	CLIP ViT-B/32	GPT2-XL	1.6B	**60.39**

6.2 Qualitative Results

Let us provide you, a sample of our results for our image captioning and visual question answering tasks. Based on random samples from the validation COCO[16] dataset and the test VQAv2[2] dataset, our MTL_CS model generated its own captions and answers respectively. From the Figures 14,15 we can confirm that our model has gained a wide generalisation ability so that it can interpret correctly not only the main elements of an image but also the relationship between them. In the context of the visual question answering, we can validate that it can address a certain question based on a image and consequently generate a meaningful answer.

[1] Zero-shot Evaluation

Naturally, there are some cases, in which our model fails to capture main features of an image. Certain images may be too complex for the model to generate proper captions or answers. In the Figure 16, we can notice certain examples which the MTL_CS model misinterpreted the content of an image.

GT: A person walking in the rain on the sidewalk.
P: A person walking down a street with an umbrella.

GT: The telephone has a banana where the receiver should be.
P: A banana is sitting on a table next to a phone.

GT: A dirt bike rider doing a stunt jump in the air.
P: A person on a motorcycle doing a trick in the air.

GT: Sunrise on an airport tarmac with a plane at ramp.
P: A plane parked at an airport tarmac.

GT: An old green car parked on the side of the street.
P: A vintage car parked on the side of the road.

GT: Two people sitting on dock looking at the ocean.
P: A couple of people sitting on a bench.

Figure 14: *Captioning results generated by our model MTL_CS, on random samples from COCO [16] validation dataset.*

Q: What is on the elephant's head?
A: Scarf.
P: Scarf.

Q: Is the boy playing baseball?
A: Yes.
P: Yes.

Q: How many giraffes can been seen?
A: 2
P: 2

Q: What color is the man's jacket?
A: Red.
P: Red.

Q: Is it still snowing in the picture?
A: Yes.
P: Yes.

Q: What is this animal?
A: Zebra.
P: Zebra.

Figure 15: *VQA results generated by our model MTL_CS, on random sample from VQAv2 [2] test dataset.*

GT: A man sitting on a toilet in front of the computer.
P: A man sitting at a desk using a laptop.

GT: A giraffe with 3 birds on his fur.
P: A group of giraffes are eating leaves from a tree.

GT: A bright room with green walls.
P: A kitchen with a green table and a table.

GT: A metal statue of two women sitting on a bench.
P: Two people sitting on a bench next to a statue of a statue in a street.

GT: A young girl sits in the bottom bunk of her bunkbed.
P: A person is laying in a bathroom with a small cribs in a bathroom.

GT: A doorway leading into a delapitated wall in a room.
P: A broken bathroom with a damaged window and a broken window.

Figure 16: *Captioning examples that our model MTL_CS failed to generate effectively.*

6.3 Ablation

In order to better understand the importance of different components in our proposed architectures, we performed ablation studies. Through our ablation process, we tried to analyse the contribution and impact of our main modules, elements or techniques that take part in our architecture. By disabling main parts of the model or inflating/deflating certain parameters of our approach we can validate the effects in the model's performance.

With respect to the Tables 6.3,6.7,6.8 we should mention that we try to implement all possible combinations within our architecture. In other words, we adjust our decoder model variant as well as apply multi task learning. In this point we should outline that our multi task learning technique requires both an image captioning and visual question asnwering dataset. Therefore, it can be applied only for the COCO and VizWiz dataset and not for the TextCaps. Furthermore, our diffusion strategies can only be applied for image captioning as a description and not a question-answer pair, is required to generate an image. In the Tables 6.3,6.7,6.8 our diffusion approach is defined with 'O', which stands for 'One-shot generation', and 'CS', which stands for 'Clipscore'.

Table 6.3: *Effect of multi task learning (MTL) and diffusion (DF) for image captioning on COCO [16] along with different GPT2 [22] variants. 'O' : 'One-shot generation'; 'CS': 'Clipscore'; 'DF' : 'Diffusion'; 'TR' : 'using the original training set'.*

GPT2	MTL	DF	TR	B1	B2	B3	B4	R_L	M	CIDEr
S	_	_	✓	**71.72**	**54.6**	**39.48**	**27.98**	**52.61**	**24.89**	**90.91**
S	_	O	✓	71.91	54.63	39.29	27.65	52.45	24.73	90.57
S	_	CS	✓	70.25	52.48	37.44	25.02	51.59	24.4	87.24
S	_	CS	_	63.55	44.00	28.63	18.04	45.70	20.04	59.25
XL	_	_	✓	**72.11**	**55.1**	**40.14**	**28.16**	**53.12**	**25.53**	**91.87**
XL	_	O	✓	71.07	52.86	37.25	25.62	51.14	23.99	86.42
XL	_	CS	✓	71.24	55.07	40.39	28.70	52.66	24.19	85.84
XL	_	CS	_	65.13	45.97	30.69	20.00	46.58	21.50	68.48
S	✓	_	✓	68.92	51.54	36.53	25.4	50.76	23.52	83.15
XL	✓	_	✓	70.94	53.06	37.67	26.17	51.7	24.29	89.75
S	✓	O	✓	70.61	53.34	38.12	26.68	51.83	24.20	87.54
XL	✓	O	✓	73.77	56.62	41.36	29.66	53.65	25.67	98.56
S	✓	CS	✓	71.65	54.36	39.00	27.32	52.07	24.28	88.36
XL	✓	CS	✓	**73.53**	**56.57**	**41.45**	**29.87**	**53.82**	**25.88**	**99.67**

From Tables 6.3 and 6.7, we can confirm that all the experiments, which are based solely on the synthetic datasets, have very low performance. Probably, it is required to provide at least a partition of the original training data, as the model should obtain and learn from certain features from the source data. Gen-

erally, we can confirm from all the Tables that the diffusion Clipscore is more effective and efficient than the plain Diffusion process. Intuitively, we expected that outcome, because choosing among five potential generated images is better and safer than processing a zero shot generated image. With respect to the ablation studies of COCO dataset [16] (Table 6.3), we can confirm that although the XL decoder improves the performance, the inclusion of the MTL technique boosts even further. Therefore, the model which incorporates both MTL, diffusion Clipscore and the XL decoder, marks the best score of the CIDEr [33] metric (99.61%) outperforming all the rest model variants. Nevertheless, the accuracy of the MTL_CS model in the VQA [2] task is also the greatest (60.39% in Table 6.2).The achievements of the MTL_CS model are remarkable, because it is a single model and has gained more predictive power than any other model. Consecutively, it is obvious that the Multi task learning approach is more than productive in the COCO dataset, meaning that the relative tasks of image captioning and visual question answering should be trained in parallel as the model shares the additional features of its relative tasks.

In the same manner, we can confirm the above conclusions with respect to the Textcaps [29] dataset (Table 6.8). The combination of a XL decoder with the diffusion process of Clipscore significantly improves the model's performance. The same applies for the VizWiz [12] dataset (Table 6.7), where the inclusion of XL decoder and Clipscore facilitates the model. However, we should point out that the MTL approach has not so positive results with respect to the VizWiz [12] dataset. As we have outlined in the previous sections, the VizWiz [12] dataset is not as representative and reliable as the COCO, because of its low quality image-text pairs. Nevertheless, the MTL_CS model variant of the VizWiz marked the best score in Accuracy (42.93% in Table 6.7) in its VQA task.

Apart from our main ablation studies, in which we implement all the variations of our proposed architecture, we also investigated other aspects of our approach. In our next ablation experiments, we focused on the number of additional generated images via diffusion, the impact of weight loss as well as the impact of the batch size in the multi task learning process. The main dataset, in which those ablation process were applied, is the COCO dataset. As we have already discussed in the previous sections, the COCO dataset is the most reliable and representative dataset in contrast to the VizWiz and TextCaps dataset. In all the below ablation experiments, the decoder of our architecture was set to its smallest model variant.

As we have outlined in the Section 4.3.1, the upper threshold of generated images in COCO dataset is 100 thousand images based on their respective captions. With respect to that limit we tried to decompose the impact of the diffusion process. As can be seen in Table 6.4, we split our generated images into four batches of 20, 40, 60 and 80 thousand images. We can validate that the more training images we obtain through the diffusion process, the better the results will be. Intuitively, this behaviour is expected as the number of qualitative training images plays an important role, almost in any neural network process.

Table 6.4: *Impact of generated images for image captioning (CIDEr) on COCO [16].*

Gen_Images	CIDEr
20K	83.54
40K	84.29
60K	85.15
80K	86.5
100K	88.36

In the same manner, we have discussed in the Section 4.2 that the weight losses between the relative processes should be adjusted. The default setting in our approach is 0.5 for each task respectively. Inflating the weight with respect to one task and consecutively deflating the remaining weight, should assign more emphasis on the inflated task. Therefore in Table 6.5, we can confirm that assigning more weight to image captioning will have as a result the model to perform better in image captioning and vice versa.

Table 6.5: *Effect of MTL weight sizes for image captioning (CIDEr) and visual question answering (Accuracy) on COCO [16].*

IC_Weight	VQA_Weight	CIDEr	Accuracy
0.2	0.8	72.91	50.54
0.4	0.6	85.00	50.41
0.6	0.4	86.80	49.34
0.8	0.2	87.20	48.15
0.5	0.5	83.15	48.00

Additionally, we described in our proposed multi task learning technique that we utilised data loaders with respect to each task. As a result, we experiment

with the batch sizes that the respective designed data loaders can provide. Generally, in our experiments the default setting of the batch size was 32 in each task respectively. In the Table 6.6, we can notice that the best performance is achieved by a model, in which the batch size of the designed image captioning data loader was 32, whereas the batch size of the designed visual question answering data loader was 16. Thus, we can confirm that overall the model requires not only a bigger amount of image captioning observations in contrast to VQA [2], but also in the ratio of 2 to 1.

Table 6.6: *Effect of MTL batch sizes for image captioning (CIDEr) and visual question answering (Accuracy) on COCO [16].*

IC_Batch	VQA_Batch	CIDEr	Accuracy
16	32	87.50	49.74
16	64	88.48	50.81
32	16	**89.52**	**51.04**
64	16	87.61	50.40
32	32	83.15	48.00

Consequently, we tried to explore all the aspects of our above propositions. Via our ablation processes, our goal was to deeply understand the advantages as well the limitations of each element. Of course, we took into account the randomness in our experiments, but our general objective is to define the optimal configurations in our architecture.

Table 6.7: *Effect of multi task learning (MTL) and diffusion (DF) for image captioning and visual question answering on Vizwiz [12] along with different GPT2 [22] variants. 'O' : 'One-shot generation'; 'CS' : 'Clipscore'; 'DF' : 'Diffusion'; 'TR' : 'using the original training set'.*

GPT2	MTL	DF	TR	B4	R_L	M	CIDEr	ACC
S	_	_	✓	14.33	39.22	16.2	38.34	_
S	_	O	✓	16.28	41.15	17.37	44.46	_
S	_	CS	✓	17.97	42.6	17.92	45.55	_
S	_	CS	_	10.81	37.88	14.86	28.93	_
XL	_	_	✓	18.51	41.96	18.09	50.74	_
XL	_	O	✓	18.51	43.22	19.1	52.38	_
XL	_	CS	✓	19.41	43.67	19.29	53.63	_
XL	_	CS	_	14.81	39.58	16.32	35.91	_
S	✓	_	✓	11.25	36.86	13.99	32.27	39.86
XL	✓	_	✓	11.02	35.92	13.56	21.56	36.16
S	✓	O	✓	10.8	36.51	14.11	29.13	37.42
XL	✓	O	✓	12.9	37.72	15.32	33.73	41.13
S	✓	CS	✓	11.05	36.32	13.83	28.38	38.54
XL	✓	CS	✓	**13.98**	**38.23**	**15.54**	**35.69**	**42.93**

Table 6.8: *Impact of diffusion (DF) for image captioning on TextCaps [29] along with different GPT2 [22] variants. 'O' : 'One-shot generation'; 'CS' : 'Clipscore'; 'DF' : 'Diffusion'; 'TR' : 'using the original training set'.*

GPT2	DF	TR	B4	R_L	M	CIDEr
S	_	✓	14.41	39.44	17.35	32.32
S	O	✓	13.91	38.98	17.57	33.95
S	CS	✓	**14.01**	**39.52**	**17.97**	**34.66**
S	CS	_	11.41	37.13	16.37	29.35
XL	_	✓	15.13	39.74	18.02	37.23
XL	O	✓	15.33	39.74	18.07	37.81
XL	CS	✓	**15.5**	**39.84**	**18.22**	**38.73**
XL	CS	_	13.61	38.35	17.45	34.24

Chapter 7

Conclusion

In our project we demonstrated several approaches and techniques regarding image captioning and visual question answering tasks. We provided a thorough analysis of our propositions and we performed extensive experiments to investigate all the aspects of our solutions. Through our experiments we applied not only our plain architecture, but also an enhanced version of it, where we introduced a unified solution integrating both image captioning and visual question answering.

In addition, we demonstrated how beneficial and effective is the multi task learning technique, where we achieved better performances in training a single model to handle two relative tasks. In more detail, we showed that our MTL_CS model not only outperformed all the prior SOA models in image captioning, but also achieved a competitive performance in visual question answering. Additionally, we introduced an efficient diffusion approach, in which we can generate qualitative and descriptive images to enrich our original datasets. We validated that the generated images can contribute significantly to the model's performance.

To sum up, our vital diffusion strategy in conjuction with the multi task learnring technique can offer a unified model that has more predictive power than a model which has been solely trained for one task. This proposal is actual and the main novelty of this project, where our initial motivation was to offer a single unified model that could be competitive in both tasks.

Concerning future work and future directions, we propose:

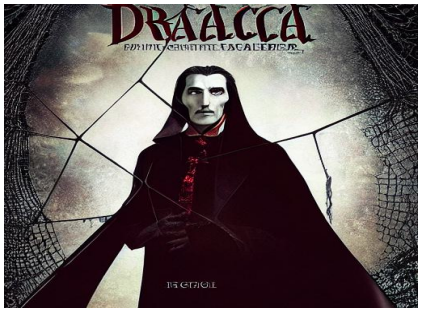
- The conduction of extensive experiments using larger datasets such as LAION-5B [27]. Large-scale datasets can improve significantly the model's performance, as the model can learn different features from billion high quality image-text pairs.
- The utilization of our clipscore diffusion strategy with respect to visual question-answer pairs. Related works [4] have been implemented, where we can construct a new question-answer pair based on a given caption. Nevertheless, the inverse process would be much more beneficial. In other words, to create a descriptive caption based on a question-answer pair so that we can generate an image through our diffusion proposal.
- The usage of large-scale pretrained dialogue models. Large language models that have been trained on conversations, would give us the ability to have a conversation with respect to an image and consecutively answer any questions that may arise.
- An optimization of our proposed diffusion strategy. Given that, a descriptive caption is required for our diffusion process, it would be beneficial if we can employ large language models, which have been trained for paraphrase generation [23], to rephrase input captions In this way, our solution would offer us another perspective on image generation process.

Appendix A

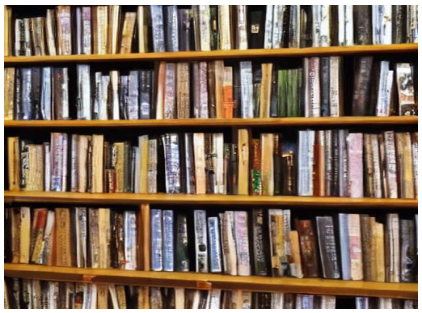
Appendix

A.1 Sample of Generated Images

Let us present below, certain visual representations Figures 17,18 of our generated images, from our innovative diffusion Clipscore process (Section 4.3.2), in the context of VizWiz [12] and TextCaps [29] dataset. With respect to a descriptive caption of an image, we generate five potential images so that the image with the highest Clipscore value is selected. We should mention that the original image is displayed on the left, while the generated image is displayed on the right, so we can evaluate the quality of the generated images.



"A poster showing Dracula with his hand lifted behind a spider web."



"A stack of identical books by author Chinua Achebe."

"A baseball game being played with a Coca-Cola billboard in the background."

Figure 17: *A sample of generated images from TextCaps [29], with captions shown underneath. (Left) Original image; (Right) Generated image.*


"A person uses a pair of brown hiking boots."


"A can of Coca Cola on a counter is shown."


"Two quarters, a nickel, and a dime sitting on a wooden surface."

Figure 18: *A sample of generated images from VizWiz [12], with captions shown underneath. (Left) Original image; (Right) Generated image.*

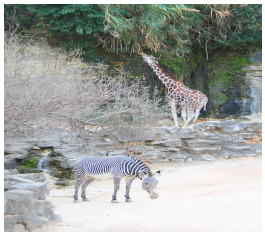
A.2 Additional Results

As we have demonstrated certain qualitative results in Section 6.2, let us provide below additional outputs from our MTL_CS model, which was trained via multi task learning and in the context of COCO dataset [16]. The partition of generated images via diffusion clipscore pipeline is integrated in the scope of its training process. We select randomly images from the validation dataset of COCO [16] and the test dataset of VQAv2 [2] respectively. The generated outputs of our model are displayed in Figures 19, 20, 21, 22.

GT: Many sheep graze in a grassy pasture in a valley.
P: A herd of sheep grazing in a field.

GT: An airplane flying away in a cloudy sky.
P: A plane flying through a cloudy sky.

GT: A orange cat taking a nap on top of a car.
P: A cat sleeping on top of a car.



GT: Wild animals grazing near a lagoon surrounded by trees.
P: A zebra and a baby giraffe standing next to a tree.

GT: A baby holding a spoon looking at a cupcake and candle.
P: A young child sitting in front of a chocolate cake.

GT: Several people riding on the back of elephants.
P: A group of elephants walking down a road.

Figure 19: *Captioning results generated by our model MTL_CS, on random sample from COCO [16] validation dataset.*

GT: Two young men walking down a runway toward an airplane.
P: Two men are walking towards a plane.

GT: A chef carrying a large pan inside of a kitchen.
P: A woman in a kitchen cooking something.

GT: An old military jet fighter climbing in altitude.
P: A fighter jet flying through the air in the blue sky.

GT: A city street with people walking and vehicles on the road.
P: A street with people walking.

GT: Creamy cheesecake dessert with whip cream and caramel.
P: A piece of cake on a plate with a fork.

GT: Dog in parking lot and on a leash.
P: A dog is standing in front of a truck of a parking lot.

Figure 20: *Captioning results generated by our model MTL_CS, on random sample from COCO [16] validation dataset.*

Q: Is that a fork or knife ?
A: Fork.
P: It's a fork.

Q: What are they doing?
A: Playing Wii.
P: Playing Wii.

Q: What season is shown here?
A: Winter.
P: Winter.

Q: Whose shadow is on the ground?
A: Batter's.
P: Batter's shadow.

Q: What color is the traffic light?
A: Red.
P: Red.

Q: What kind of game is this?
A: Baseball.
P: Baseball.

Figure 21: *VQA results generated by our model MTL_CS, on random sample from VQAv2 [2] test dataset.*

Q: What kind of animal is this?
A: Dog.
P: Dog.

Q: What fashion accessory is on top ?
A: Tie.
P: Tie.

Q: What room is this a picture of?
A: Bathroom.
P: Bathroom.

Q: What is behind the elephants?
A: Trees.
P: Elephant.

Q: Is the man wearing a helmet?
A: Yes.
P: Yes.

Q: Who is the sponsor on the wall?
A: Usta.
P: The tennis player.

Figure 22: *VQA results generated by our model MTL_CS, on random sample from VQAv2 [2] test dataset.*

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[3] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[4] Soravit Changpinyo, Doron Kukliansky, Idan Szpektor, Xi Chen, Nan Ding, and Radu Soricut. All you may need for vqa are image captions. *arXiv preprint arXiv:2205.01883*, 2022.

[5] Michael Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

[6] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[7] Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. Enabling multimodal generation on clip via vision-language knowledge distillation. *arXiv preprint arXiv:2203.06386*, 2022.

[8] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11162–11173, 2021.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

[12] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

[13] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Qiang Liu, et al. Language is not all you need: Aligning perception with language models. *arXiv preprint arXiv:2302.14045*, 2023.

[14] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[15] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[17] Ron Mokady, Amir Hertz, and Amit H Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021.

[18] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards

photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[19] David Nukrai, Ron Mokady, and Amir Globerson. Text-only training for image captioning using noise-injected clip. *arXiv preprint arXiv:2211.00575*, 2022.

[20] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[23] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[26] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[27] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[28] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[29] Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. Textcaps: a dataset for image captioning with reading comprehension. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 742–758. Springer, 2020.

[30] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

[31] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pages 9120–9132. PMLR, 2020.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[33] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[34] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[35] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022.

[36] Deshraj Yadav, Rishabh Jain, Harsh Agrawal, Prithvijit Chattopadhyay, Taranjeet Singh, Akash Jain, Shiv Baran Singh, Stefan Lee, and Dhruv

Batra. Evalai: Towards better evaluation systems for ai agents. *arXiv preprint arXiv:1902.03570*, 2019.