Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

Capturing the Narrative: Deep Learning Models for Comics Sequences

Project Report to fulfill the Master's degree in Informatics Engineering

Specialization in Intelligent Data Analysis

Author

Gonçalo Ventura Lourenço Marouvo

Supervisor

Francisco José Baptista Pereira

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, dezembro de 2024

Abstract

Comics represent the complex way humans can communicate and expose ideas, which pose additional challenges for image-to-text deep learning models. In this project, we investigate how multimodal deep learning architectures perform in describing a comics vignette. We investigate how current State-of-the-Art models (GIT and BLIP-2) are able to describe the narrative in 4-images comics sequence from a dataset we created.

We find that some prompting can produce acceptable results. We also assess how to propagate information across the sequence's images, by adding to prompts the previous outputs of the images from the same sequence. The results show limited improvements from this strategy.

While the overall meaning of the predicted descriptions is close to the semantic space of the real descriptions, they are still far away from human-level descriptions. Therefore we propose several future experiments, where we highlight reinforcement learning to train a large language model as a policy function for prompt generation.

Keywords: Comics, Computer Vision, Image Captioning, Multimodal Deep Learning Models, Prompt Engineering.

Acknowledgments

I would like to express my gratitude to my supervisor, Francisco Pereira, for his patience, encouragement, and expertise throughout this journey. Also, completing this task would not be possible without the comprehension which goes beyond a pure supervisor's job.

I am also profoundly thankful to my girlfriend, Andreia, for all the support. This Master degree has been quite a rollercoaster for me, and She always gave me support and showed comprehension. When I was too focused, she withstood the lack of attention from my side; when I was with no motivation, she helped me muddle through.

Finally, I want to extend my appreciation to my mother, whose endless sacrifices are an inspiration.

Index

Tables Index

Figures Index

Acronym Index

AI	Artificial Intelligence
BART	Bidirectional Auto-Regressive Transformer ([1])
BLEU	Bilingual Evaluation Understudy Score ([2])
BLIP-2	Bootstrapping Language-Image Pre-training-2 ([3])
CLIP	Contrastive Image Pre-Training ([4])
CNN	Convolutional Neural Networks
COCO	Common Objects in Context ([5])
FLAN	Scaling Instruction-Finetuned Language Models ([6])
GIT	Generative Image-to-Text Transformer ([7])
GPT-2	Generative Pre-trained Transformer 2 ([8])
GRU	Gated Recurrent Units
LLM	Large Language Model
LSTM	Long Short-Term Memory LSTM
MLP	Multi-Layer Perceptron
NLP	Natural Language Processing
OCR	Optical Character Recognition
OPT	Open Pre-trained Transformer Language Models ([9])
PPO	Proximal Policy Pptimization ([10])
RL	Reinforcement Learning
RNN	Recurrent Neural Networks
SOTA	State-of-the-Art

1 Introduction

1.1 Motivation

The story of humankind is closely related to the ability to create narratives and depict representations of reality. These representations do not limit themselves to the resemblance of what humans see, hear, feel, but also transcend those boundaries, by referring to general ideas and imaginary beings. The example can be as old as the cave drawings our ancestors created. Not only they contained "real" representations, but also some abstract symbols and supernatural beings. Back to the modern age, language systems allowed for more complex representations. Within this context, comics is one of the most complex multimodal examples. The combination of visual and textual elements in comics makes them a unique and powerful medium for storytelling, capable of conveying complex narratives, emotions and messages, in a visually engaging format. It is not by coincidence that comics is called the "ninth art".

The complexity of comics makes it challenging for a deep learning model to "understand" them ([11]) when compared to "real world" images. This work aims to make contributions to the literature on comics' understanding, by assessing the performance of image-to-text models in describing comics with no textual elements. In spite of the fact that there is ample literature on semantic segmentation of comics, the literature on deep learning applied to content generation in comics is scarce. There is ample quantity of models who seek to automatically identify which pixels of comics' vignettes belong to the scene, balloons, textual references, but just a few deal with content understanding and generation.

We also try to address the problem of how to capture the narrative in sequences of images. Labeling an action, for example, can sometimes be possible only when the information on several images is jointly taken into account. We propose to encompass that idea using prompting, by adding to next image's input a text input who concatenates previous captions. We assess how good this approach is in a newly created dataset, where we assess different prompts, as well as different ways to link the individual captions of each images, in order to arrive at a consistent description of the whole sequence.

The backbone of the architecture comprises multimodal models, who are trained to learn jointly the embedding space from both text and images. Initial analysis picks the best model among candidates to submit to the whole pipeline to capture the narrative.

We also try to provide valuable contribution by creating a dataset comprising several

sequences of four images. Raw comics can also be challenging by the unusual way the vignettes are distributed, as well as the presence of balloons. Additionally, removing textual references can put additional challenges, making images become irregular. Therefore we try to simplify by submitting our approach to a "cleaner" dataset, by creating one from scratch.

1.2 Objectives

There are relevant gaps to fill in the part related to understanding of the actions and the generation of image or text that prove to be satisfactory to humans. Additionally, there is scarcity of datasets able to assess image-to-text models. We highlight the following main challenges from a methodological standpoint:

- Having a model who can be precise in describing the content of the images, beyond identifying that they are of a type "cartoon";
- Flow information throughout the images of the sequence, so that the model is able to caption actions/ideas who are perceptible only by mixing information from several images;
- Having a "clean" dataset with comics' images and descriptions.

1.3 Findings

Our main findings are:

- Prompting can play a crucial role when reading all the images from the sequence at once;
- Improving prompts by adding previous image's outputs show a limited increment in performance scores;
- Prompts who include previous outputs have similar performance, so preceding information does not help the decoding process in this task;
- The preferred metric to assess our results relates to the proximity of the semantic meaning of sentences, instead of literal words in common between true and predicted narratives;
- The sentence similarity score of the narrative descriptions is decent, but there is margin for improvement, as predicted narratives are still far away from human-like ones.

1.4 Contributions

To sum-up, the main contributions of this work are:

- Build a comprehensive dataset on four-images comics sequences, which can be used for the assessment of a large variety of vision-language tasks;
- Assess how multimodal models (GIT [7] and BLIP-2 [3]) perform in captioning content of comics' images;
- Propose an architecture to provide automatic captioning of comics' sequences, focusing on the narrative of the whole sequence;
- Assess how to use prompt engineering to improve the caption of a comics' vignette.

1.5 Document structure

This work is organized as follows: chapter 2 shows an overview of the methodology; chapter 3 presents the dataset we created; chapter 4 dives into the state-of-the-art (SOTA) models on image captioning and takes a tour on the main deep learning approaches applied to the comics' context; chapter 5 sets and explains the main settings of our experiment; chapter 6 presents the results; chapter 7 proposes future investigation lines; 8 is the chapter who presents the conclusion of our project.

2 Computational Framework

This chapter aims to provide an overview of the methodology and essential steps of this work. It presents an high-level explanation to our approach to comics narrative description, while leaving a thorough description for the following chapters. We start with a broad summary of the key deep learning milestones crucial for the development of image-to-text models.

2.1 Deep Learning and Image to Text Models

There was a pivotal milestone who leads us to split history between two periods: before and after transformers took center stage.

The initial (and standard) deep learning model is the multi-layer perceptron (MLP) (see for example the overview in [12]), whose architecture consisted of several fully connected layers stacked together, where each node comprises some activation (non-linear) function on a linear combination of previous layer's nodes (each layer is composed of several nodes). The first layer comprises the input features and the output layer's size and activation function will depend on the prediction task. For example in classification tasks the final activation function is usually the softmax, who maps the previous layer into classes' probabilities.

However, this type of architecture exhibited some limitations in understanding both text and images, which led to different developments for the two types of data.

Starting with images, which are the main input of our work, the MLP lacked the idea of neighborhood: the architecture did not encompass a mechanism to identify some pattern from a group of pixels close to each other. Relative positions of pixels (which are a key variable to understand any image) are lost in MLP, as the first layer will flatten the 3-dimensional matrix of the RGB image. Convolutional neural networks (CNN) solved this problem by having layers which train filters who produce feature maps, serving as images' features identifiers ([13]). This idea is presented in Figure 2.1 and is the backbone of a CNN. As one goes deeper into the CNN, the feature maps usually become smaller and more complex. Imagine for example the CNN to identify whether some image has a cat in it. The feature maps in the first layers will represent simpler elements, like the whisker or head's contour. As one progresses in the layers, the final feature maps will represent crucial and more complex features of the cat: for example, the whiskers and mouth; the tail, or even some usual colors' combination with

silhouette ([14]).

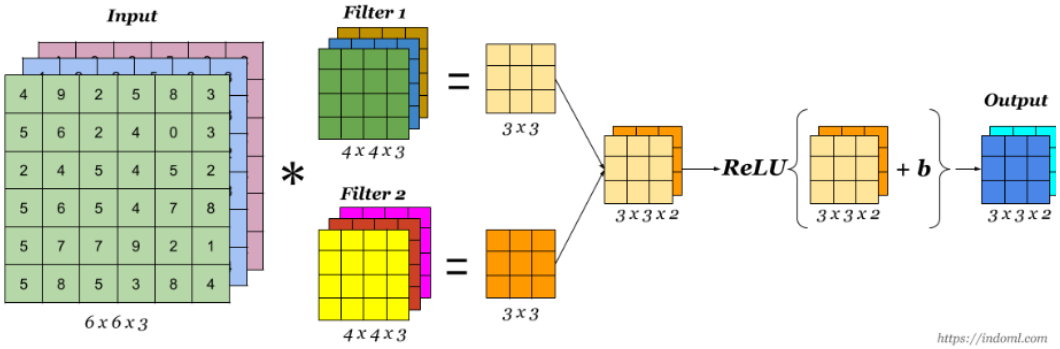


Figure 2.1: The convolution layer[1].

It is also worth emphasizing the translation invariance property, by which the CNN is able to identify an object regardless of its location in the image. Moreover, CNN is a more efficient set-up: the filters trained are the same irrespective of the position within the image (weight sharing). Finally, an image with a large number of pixels would have a MLP with much larger number of parameters to be trained. Therefore, not only the MLP lacks the mechanism to find patterns, but that can also lead to overfitting.

Regarding text comprehension, two essential elements are worth emphasizing: embeddings and neural networks for sequence understanding. Let us begin with the latter. Textual elements comprise sentences or paragraphs. This makes the words' order the crucial characteristic of any model who seeks to understand language. One sentence with the same words arranged in a different way can have a very different meaning (or no meaning). Therefore neural networks which aim at understanding and producing human language need to be able to interpret sentences as a sequence. Recurrent neural networks (RNN) ([15]) were the initial architecture to encompass this idea. Sentences are processed sequentially: any given word is an input to the model sequentially, who keeps an hidden state from previous words (Figure 2.2). The hidden state is the key variable which tries to propagate information across the whole sequence. Like the CNN for images, the weights are shared across each processing step, so RNN's not only encompass a better mechanism to understand sentences than the MLP, but they also are less parameter-heavy. While RNNs addressed these shortcomings of MLPs, they have limitations in capturing very long-term dependencies due to issues like vanishing gradients ([16]). These challenges led to the development of advanced variants like Long Short-Term Memory (LSTM) ([17]) and Gated Recurrent Units (GRU) ([18]), further enhancing RNN capabilities for text comprehension.

The second aspect of deep learning applied to text is embeddings. Their role is to compress the information in the vocabulary (all the words in the dataset) into a smaller dimension array, consistent with the semantic proximity. An extreme alternative was

[1] https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/.

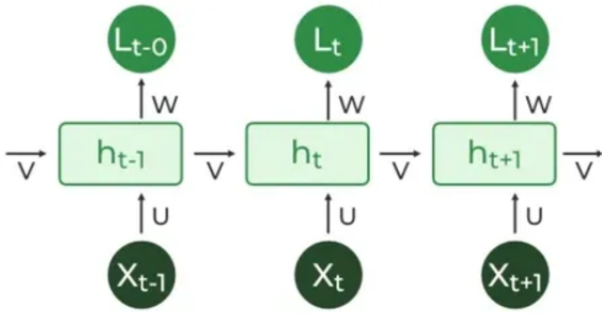


Figure 2.2: RNN's key element to process sequential data: x_t is the input, L_t the output and h_t the hidden state[2].

to do one-hot-encoding of words (they always need to be transformed into numbers for any model), but this option would render a very sparse matrix with some ones among a lot of zeros, whose dimension is too large (number of different words within the data). Initially, embeddings were static, meaning they were built from co-occurrence statistics from a corpus ([19]), regardless of context. Today embeddings are contextualized, usually learned within any language model, comprising the first layers of the neural network (for example [20]). Notice these embeddings are far away from static, implying they can be very different depending on the type of content the neural network is trained. For example, the words *pawn* and *queen* might be much closer in the embedding space if training occurs specifically in text where chess books were part of. Figure 2.3 shows one example. The embedding space has seven dimensions, and from the differences between words within each dimension, one can infer what the dimension is measuring. The representation in 2-dimensions space is to make it clear visually: *cat* and *kitten* are very close to each other; and *dog* near than *houses*; this is intuitive from the stand point of the closeness of words in their meaning.

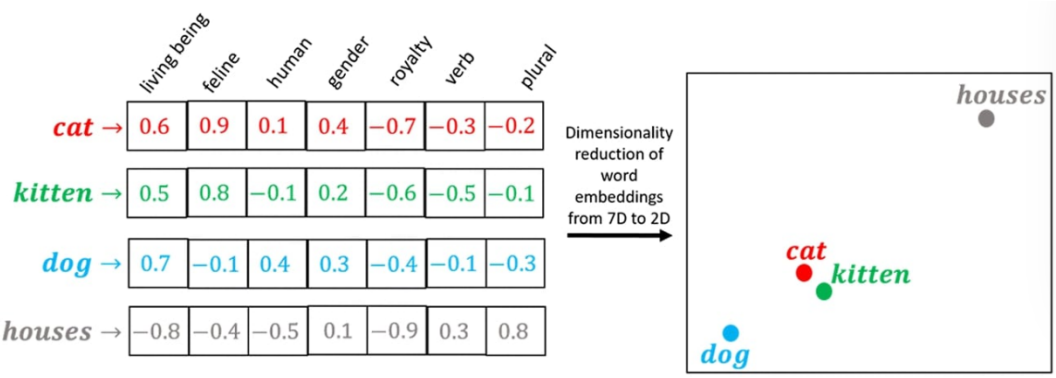


Figure 2.3: An example of word embeddings. Each element of the vector will capture a key dimension who defines the word[3].

Transformers revolutionized the deep learning space. They brought the idea that at-

[2] https://medium.com/@y.zhyliuk/what-is-recurrent-neural-networks-rnn-and-how-does-it-works-f3f9ad6ed471.
[3] https://coderzcolumn.com/tutorials/artificial-intelligence/how-to-use-glove-embeddings-with-pytorch.

tention mechanisms can be the main backbone for a language understanding model ([21]). Up until this discovery, attention mechanisms were already employed, mainly within RNN's. They allowed the RNN's output to focus on different parts of the input, depending on the word who is generated ([22]). Attention comprises a mathematical operation to distinguish which words have a stronger relationship and which associations give meaning to sentences. They are computed as internal products, and the training of the neural network involves finding the parameters who will define how strong is the relationship between the words of each sentence. Figure 2.4 shows the way attention is computed. It consists of three elements: query, key and value. When the model is "learning" how each word is relevant for own sentence's comprehension, the query matrix attends the same embedding vector as the key: this is called self-attention. When one is in the decoding part, the query matrix is attending the encoding part. Imagine the language translation task from english to french: the query is the english comprehension part, whereas the key relates to french's element ([21]).

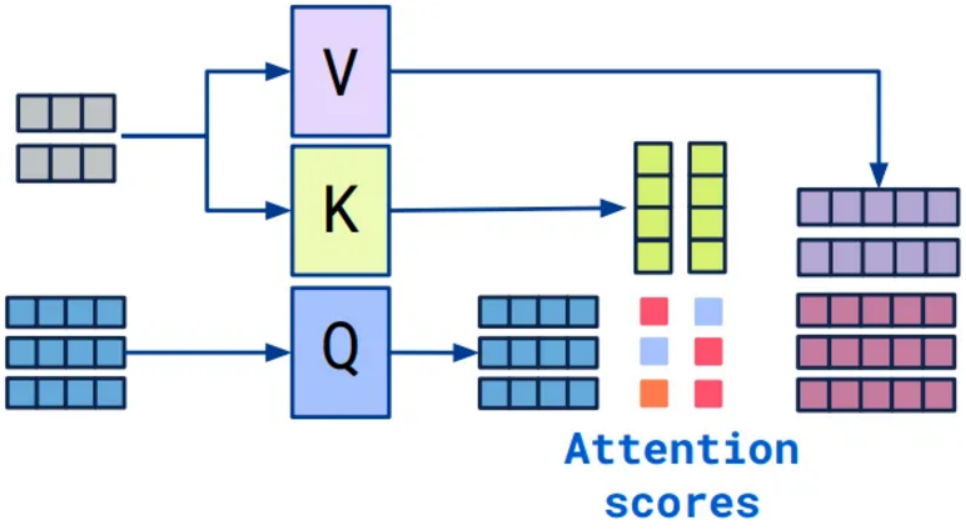


Figure 2.4: The essential element of the attention mechanism.[4].

The transformer revolution involved the realization that a mechanism previously considered secondary could become the primary and most crucial component of neural networks. There was no need to process data sequentially, sentences could be processed all at once. Therefore it also brought the ability to train models with parallelization ([21]). The attention mechanisms rely on the idea of computing internal products between the elements of the sentence. The training of this neural networks comprises the identification of whose words are more connected, both in backward and forward in the sentence.

It is also relevant to mention the two main elements of a transformer: the encoder and the decoder ([23]). The encoder is the component of a neural net who seeks input

[4] https://medium.com/@saba99/self-attention-0b21baad0a48.

understanding. It tries to identify the main elements of the input who solve the task, so it usually acts as a compressor of the information in the inputs. The decoder is usually designed for generative tasks. For example, producing a sentence. Figure 2.5 shows an example for the translation task we referred previously.

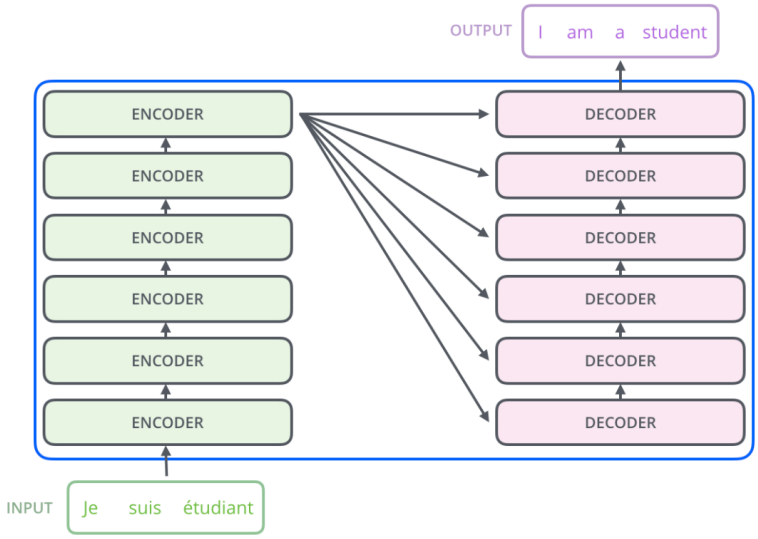


Figure 2.5: The encoder-decoder architecture[5].

Transformers initially did not compete in image understanding with CNNs. In fact, the attention mechanisms could not be computed on a pixel level basis, as the number of parameters to be trained grows quadratically with the dimension of the image. But looking to images in a more aggregate way, by using a group of pixels (patches) solved the large number of parameters problem, and transformers also became ubiquitous in image understanding ([24]).

So far, we talked about models which handle either text or images separately. There could be models who deal simultaneously with image and text, but the embedding space was learned separately. Even when the output or input are from different types (for example, an image input and a word output, like a caption), the encoder and decoder originally were built out of embedding spaces trained separately.

However soon it was shown that models who handle both image and text simultaneously, learning an embedding space which mixed both types of data, come with advantages (for example [4]). Getting back to the embeddings example, it would act as way to have in a diagram both words and images' patches (Figure 2.6). This unified embedding space allows for richer and more nuanced representations, as it inherently captures the relationships between text and visual information in a shared semantic context. By training on paired image-text data, these models align visual features with their corresponding textual descriptions, enabling tasks such as cross-modal retrieval,

[5] https://jalammar.github.io/illustrated-transformer/.
[6] https://weaviate.io/blog/multimodal-models.

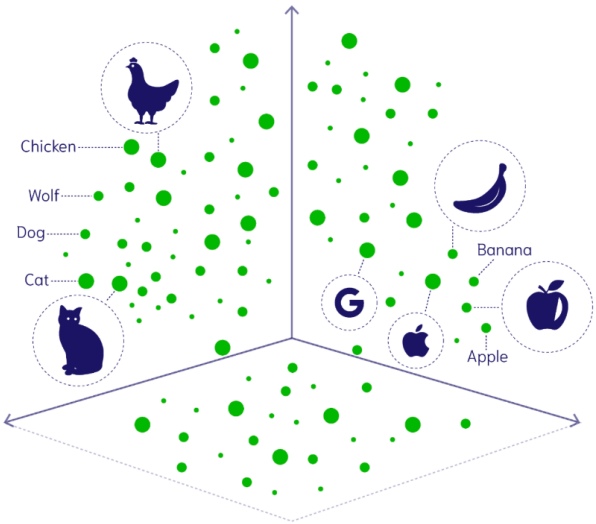


Figure 2.6: The multimodal space, where both text and image are mapped together[6].

where a text query can be used to find relevant images, and reciprocally. Additionally, such models can perform zero-shot learning more effectively, generalizing to tasks or concepts not explicitly seen during training ([3]). This seamless integration of modalities marks a significant step forward in creating systems that understand and interpret the world more holistically.

2.2 The Hugging Face platform

Hugging Face (`https://huggingface.co/`) is a widely used platform for people who seek to employ transformers based models. It can perform tasks in a wide variety of dimensions, like Natural Language Processing (NLP), computer vision or audio processing (Figure 2.7). It provides tools, pre-trained models, and libraries that simplify the process of building, training, and deploying machine learning models.

Its `Transformers` library is a key tool, which provides pre-trained implementations of models. These models can be used directly for a variety of tasks, such as text classification, machine translation, and text generation. Hugging Face also offers a `Datasets` library, which provides access to thousands of standardized datasets, making it easier to train models without the need for complex data preprocessing. In addition, the platform promotes an open source and collaborative environment in which users can share their models and resources.

A major feature of Hugging Face is its pipelines, which are high-level tools designed for common machine learning tasks. These pipelines handle the entire process of using a model, from loading the pre-trained model and tokenizer to generating predictions and processing outputs. This makes pipelines extremely easy to use, even for people

[7] `https://huggingface.co/models`.

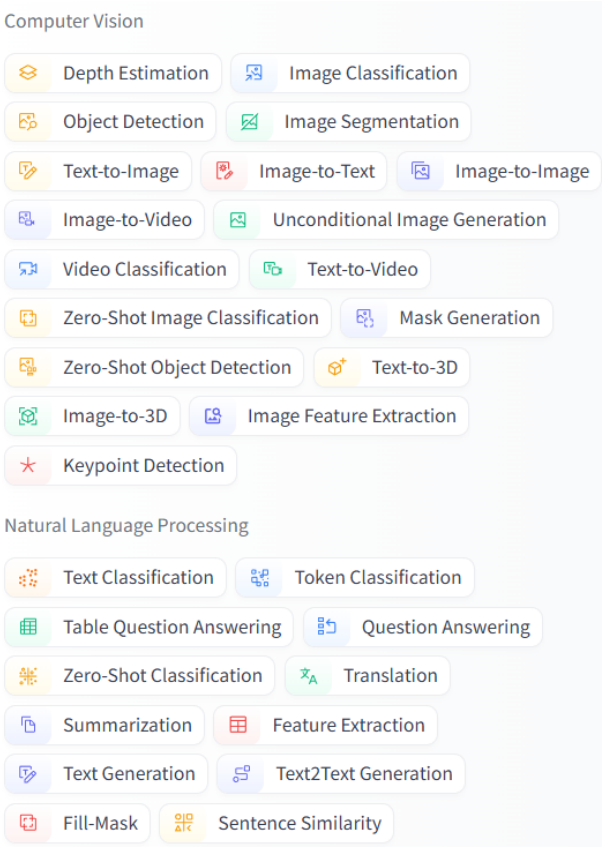


Figure 2.7: Examples of tasks in Hugging Face[7].

new to artificial intelligence (AI).

Pipelines are available for various tasks, including text classification, sentiment analysis, text generation, machine translation, named entity recognition, question answering, and summarization. Beyond text-based tasks, pipelines also support multimodal applications like image classification and speech-to-text processing. While pipelines are designed to work out of the box, advanced users can customize them by using their own pre-trained models or configurations. Figure 2.8 shows how a few lines of code can compute word embeddings, by using the library `SentenceTransformer`.

```
from sentence_transformers import SentenceTransformer
sentences = ["This is an example sentence", "Each sentence is converted"]

model = SentenceTransformer('sentence-transformers/all-MiniLM-L6-v2')
embeddings = model.encode(sentences)
print(embeddings)
```

Figure 2.8: Example of the implementation of a pipeline for the computation of sentence embeddings[8].

The benefits of Hugging Face and its pipelines are numerous. The platform makes advanced machine learning tools accessible to users without a deep background in AI,

[8] `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`.

reduces development time by simplifying implementation, and allows reusability of pre-trained models, saving computational resources. It supports both beginner-level and advanced use cases, making it highly flexible for different needs.

Finally, *Hugging Face* also has a direct API inference model. Figure 2.9 presents an example for Generative Pre-training Transformer 2 (GPT-2, [8]), directly in the platform. Users can also connect to the API, allowing for serverless inference.

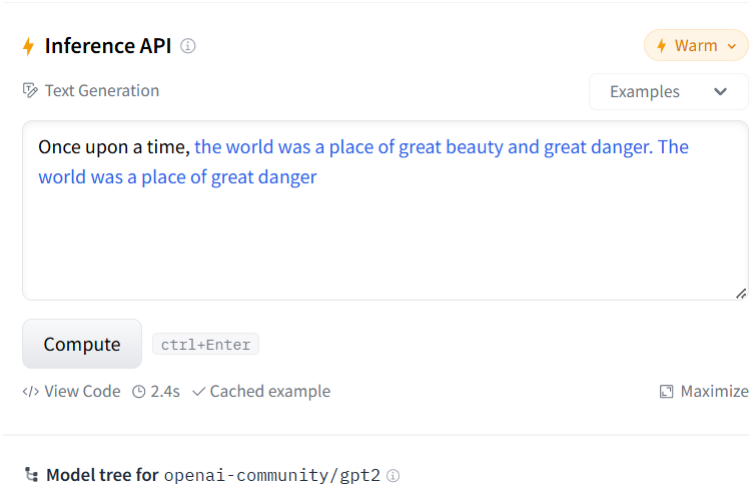


Figure 2.9: Example of direct inference in platform, using an LLM[9].

In summary, *Hugging Face* has transformed the AI landscape by making it easier for everyone to use state-of-the-art (SOTA) models. Its pipelines feature is a powerful tool that enables users to perform complex tasks efficiently and effectively, without needing to build models from scratch. Hugging Face is a vital resource for anyone working in NLP, computer vision, or other AI fields.

2.3 Framework

The goal we want to achieve is to have an architecture that is able to caption the narrative in a sequence of comic vignettes. Figure 2.10 presents the key parts of our work. The model receives 4 RGB images, which go through several steps to output a sentence aiming at describing the narrative.

We can sum our approach in the following steps:

1. Build a dataset having 4-image sequences of comics with no textual elements;
2. Create captions for both individual images and the whole sequence;
3. Choose candidate multimodal models;
4. Select the best metrics to check how close are the predicted captions to the "real" ones;

[9] `https://huggingface.co/openai-community/gpt2?text=Once+upon+a+time.`

Figure 2.10: The objective: input comics' images and output a caption.

5. Choose the best multimodal model;
6. Create several prompt strategies to investigate which one is better;
7. Propose different ways to link prompts in order to embed previous images' information into the current image;
8. Summarize the captions to have a concise, ideally one sentence predicted narrative;
9. Assess the performance across all prompts and connecting strategies.

2.3.1 Input

The goal is to input the model some comics' vignette and output a sentence depicting the narrative. Hence the model will process an RGB image. Two options will be assessed:

- Show each vignette one by one. The sequential strategy might be advantageous as prompting can be improved from previous images' output. Moreover, inputting the sequence at once might prove too much for the model to ingest.
- Show all 4 images at once, in a merged input.

The goal is to produce some sort of text output based on an RGB image input: an image-to-text model. The most suitable models are multimodal ones, once they are able to learn common image and word representations during the training process, facilitating the "translation" of images into words.

We then proceed to build a pipeline to caption the sequences of comics' images.

2.3.2 Prompting

A prompt is an input text or instruction given to the model to guide how it processes and decodes information. In the context of image-to-text models, a prompt can influence the way the model describes or interprets an image. The same model can produce very different outputs based on different prompts.

The prompt helps set the context and task for the model. This is especially important when a single model can perform multiple functions (e.g., description, inference, reasoning). A prompt tells the model what kind of information to prioritize. The wording of a prompt can tweak the tone or style of the output. Models are probabilistic, meaning they choose words based on learned likelihoods. A prompt can nudge these probabilities. By explicitly asking about relevant details, prompts can reduce ambiguity. Figure 2.11 shows how this mechanism usually works. The sequence of words comprising the prompt is converted into the embedding space, and the decoder of the language model will attend to it when it starts to generate text. Prompt therefore acts as a context vector from which the decoder will compute attention mechanisms to compute the first word, and proceed sequentially, up until the word given is the end of sentence/decoding.

We are particularly interested in starting to assess performance improvements by testing several prompts. Moreover, we can improve caption by prompting images with previous captions from the previous images of the same sequence. Therefore we also create several connectors to use prompting as the most straightforward way to "show" previous information. It is also worth mentioning that we sought to keep prompts as generic as possible, in order to avoid "nudging" the model to the results we want.

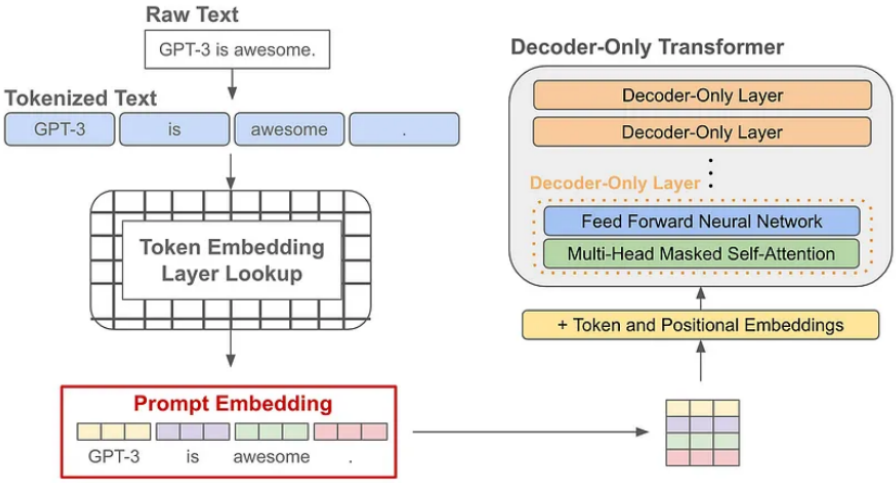


Figure 2.11: The mechanics of prompting[10].

2.3.3 Narrative assessment

We define two benchmarks to serve as baselines:

- Show the sequence all at once and use the set of prompts we define;
- Show each image individually, with no prompt.

These benchmarks will help in disentangling whether our results stem from individual

[10]Source: *https://towardsdatascience.com/advanced-prompt-engineering-f07f9e55fe01*

inputting the images or prompting. We assess zero-shot performance of models for 4 different approaches:

- Show each image individually, using a generic prompt;
- Show each image individually, while using a prompt who includes all previous outputs from previous images;
- Add a linking text, where context is given to "explain" all history of images.

We finalize by assessing how close predicted captions are to "real" ones. What metrics to use? We choose two metrics to assess the architecture's performance, who complement each other. On one hand, we have a metric which will check literally how many words does the predicted and true narrative descriptions have in common. However, some captions might be similar even if the words are not exactly the same, even if they belong to the same semantic space and are almost similar in describing the sequence's general idea. So we pick a metric of sentence similarity based on mapping both sentences into and embedding space and computing how similar they are.

This chapter aimed at presenting the an high-level overview of the methods related to our research question, which will be detailed in the next chapters.

3 Dataset

This chapter aims to present the dataset we built to evaluate our research question, as well as to detail the main criteria we use to pick the images' sequences.

There is no dataset suitable for our purposes. Although the number of datasets comprising comics' content is abundant (as it will be referred to in 4.1), no one fits our main criterion: no textual elements within the images. Additionally, albeit there are deep learning techniques who are able to remove those textual elements from comics' images, they leave a empty space within the image, and make it harder to depict the narrative.

Therefore, additional value of this work lies in the construction of a dataset having sequences of comics' figures without textual references. The data set is available on the link `https://www.kaggle.com/datasets/marouvo/comics-4-image-sequences/data`. Details of the sources are given in Annex B. The main focus of this work is to have a model depicting the narrative. As the dataset was created with no original narrative description (usually they come from dialogues, where the focus can deviate from some of the image's elements), we created the narrative's descriptions. Additionally, we produced individual captions, as this will help to select the best multimodal model.

To sum-up the elements of the dataset that were created:

- 60 sequences of 4 images: the input of the pipeline;
- The narrative description: essential to compare the proposed captions coming from the pipeline, and assess how close they are;
- Individual captions: raw description of each image, who serve the purpose of helping in the choice of the image-to-text deep learning model.

The images do not have textual elements, as our goal is to caption a narrative based on images. The presence of text within the image might make the model perform better because of some sort of embedded optical character recognition (OCR) and not because of "pure" image interpretation ([25]). Moreover, some comics which add textual elements attached to them, like dialogue balloons, might mislead the whole narrative when one is looking to the 4-images sequence.

As a result, there are no captions for the vignettes. We have no other option than to create our own description of the narrative. In order to avoid being influenced by the caption generated and have the incentive to propose a caption matching the generated ones, the captions were created before analyzing any output from the models.

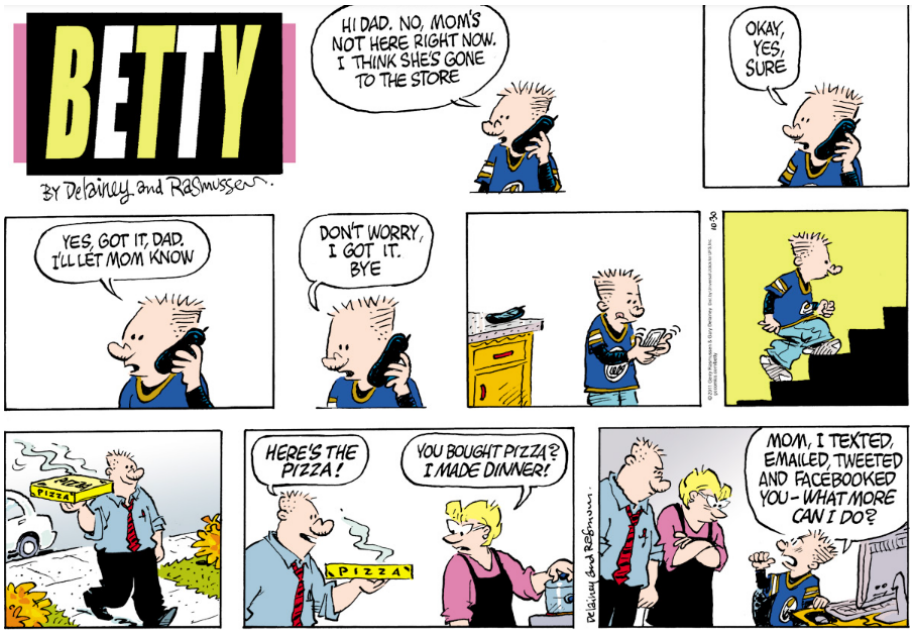
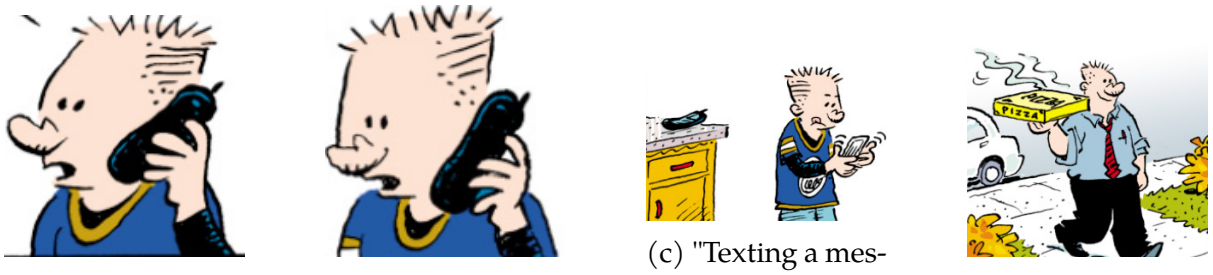


Figure 3.1: An example of comics vignette in original format[1]. Four images were cut to produce the sequence in Figure 3.2.

The proposed captions aimed at grasping the main idea stemming from the sequence, disregarding any details: our goal is to check how our pipeline can capture the main narrative. For the initial choice of the model, individual captions were also created, as they are relevant to pick the multimodal neural network that will be the backbone of the pipeline.



(a) "Talking on the phone"

(b) "Talking on the phone"

(c) "Texting a message on mobile phone"

(d) "Bringing a pizza"

Figure 3.2: The "cleaned" sequence of images. Proposed narrative: *"Ordering a pizza by phone"*. *Image_35* in the dataset.

We present some examples. Figure 3.1 shows the original vignette from which the four images presented in Figure 3.2 were extracted. Even when the figures were removed from vignette having balloons, the description of the sequence might be very different when one is digesting only the images. From the original comics, one understands that it depicts a situation were the son failed at informing the mother that the father would bring a pizza, so there was no need to cook dinner. When one looks into the

[1]Source: `https://www.gocomics.com/betty/2011/10/30`.

extracted images, the idea is not so clear. Although one could suspect that the man having the pizza was not a delivery employee due to his clothes, we defined the caption as *"Ordering a pizza by phone"*. Going beyond with more details or presumptions could be a subtler task and very demanding.

Figure 3.3: An example of comics vignette in original format[2].

(a) "I will try to use the kite"

(b) "The kite is not working"

(c) "I give up on the kite"

(d) "I will fly a balloon"

Figure 3.4: The cleaned sequence of images. Proposed narrative: *"He gave up on the kite to use the balloon"*

Figure 3.4 is presented as one example where the cleaned sequence is very close to the original vignette (Figure 3.3). However there are not a lot of Calvin and Hobbes (the name of the comics in 3.3) who fulfill our criteria, or some of them do not have a narrative who is easy to depict (they refer to some unrealistic/imaginary actions or situations). Hence Calvin and Hobbes ends-up having few sequences, even though one could expected them to be well represented, due to the fact that its vignette's form is very close to what we intend to have (very few textual elements, with few images).

Figure 3.6 presents another example on the difference between the original meaning of the sequence and the one we propose. The original comics in Figure 3.5 show a conversation where the wife wonders if the accidental shaving of half the husband's mustache might be a good surprise, because he might look better without it. The last image show deception, his face with no mustache made him look worse. This imaginary story seems to be too complex for one looking at the images without balloons

[2]Source: `https://i.pinimg.com/originals/87/eb/59/87eb594c959d1f71e5cf8c02c28477f6.gif`.

to grasp, so we propose the narrative description as *"shaving"*, expecting it to be sufficiently hard to be apprehended. The model needs to connect several references to shaving in each image: 3.6a mirror; 3.6b position of the hand, holding a small object; 3.6c towel; 3.6d the face without beard/mustache. We also emphasize the individual captions, which are raw and quite literal.

Figure 3.5: An example of comics vignette in original format[3].

(a) "Old couple looking at the mirror"

(b) "Old couple, man is shaving his face"

(c) "Old couple, man is drying face with towel"

(d) "Old couple talking"

Figure 3.6: The cleaned sequence of images. Proposed narrative: *"Shaving"*.

Below we present all criteria for the chosen images/sequences:

- Uniformity. Use same length of the images' sequence: four. This number was defined based on having the balance between having a sequence large enough to assess the model's ability to identify the narrative, but not demanding enough to show too complex set of actions.
- Purity. Pick images without textual references. Otherwise the model might perform better due to some kind of embedded optical character recognition, rather than "pure" understanding of the images. Removing textual references from the images largely reduces the array of candidate comics, as some images were designed in conjunction with textual elements (balloons). Hence it might be very hard to produce a narrative out of a images' sequence, even for a human who does not know any context. For example, one can imagine a simple dialogue, where the images are almost similar across the vignettes, and the change in information stems only from the content of the balloons: if one removes the balloons, one finds an uninformative sequence of similar images.

[3]Source: `https://www.gocomics.com/pickles/2024/06/11`.

Table 3.1: Number of sequences of the dataset per comic.

Comics Name	Count
Betty	15
Ben	8
Arlo and Janis	6
Calvin and Hobbes	5
Family Tree	5
Beardo	4
Pickles	3
Peanuts	2
The Barn	2
Barney & Clyde	2
Garfield	1
Big Nate	1
Adult Children	1
The Adventures of Business Cat	1
Banana Triangle	1
Buni	1
The Meaning of Lila	1
Next Door Neighbors	1
Total Sequences	**60**

- Variety. Do not rely in a small set of comics, as the model might perform well due to idiosyncratic issues. The goal is to have a captioning pipeline who works for several types of comics.
- Simplicity. Pick sequences whose figures can identify some action/idea which might be not too difficult to identify. This choice relied more on the intuition of human judgment. However, an aspect easy to illustrate is to avoid comics where there are tiny changes between the figures: the action/idea might become too subtle for a multimodal model to depict it.

We end-up with a dataset comprising 60 sequences of 4 comics' images, from different comics. Table 3.1 presents the count of sequences by source of the original comics they were from. We have a total of 18 different comics in our dataset. While 8 of them contribute with one sequence, there is one who contributes to a quarter of the sequences. What would be the ideal dispersion across comics? Taken into account the "variety" property we defined before, in the best scenario one would have no more than one sequence per comic. However, the comics we searched for do not fit the other three properties. Our search base was based on the page *GoComics* (`https://www.gocomics.com/`), which gathers a huge variety of comics, as well a large number of vignettes per comic. Unfortunately, the number of comics which fit the criteria we define is scarce. So we needed to repeat comics to have a dataset with some sample size. However, the Betty comics, which contribute to 15 sequences, are diverse enough not to

put in jeopardy the principles we define, and consequently not to hamper generalization ability. Betty is a very diverse comics, which depicts very heterogeneous scenes, performed by different characters, as we show in Figure 3.7. These three sequences show characters performing an heterogeneous set of tasks.

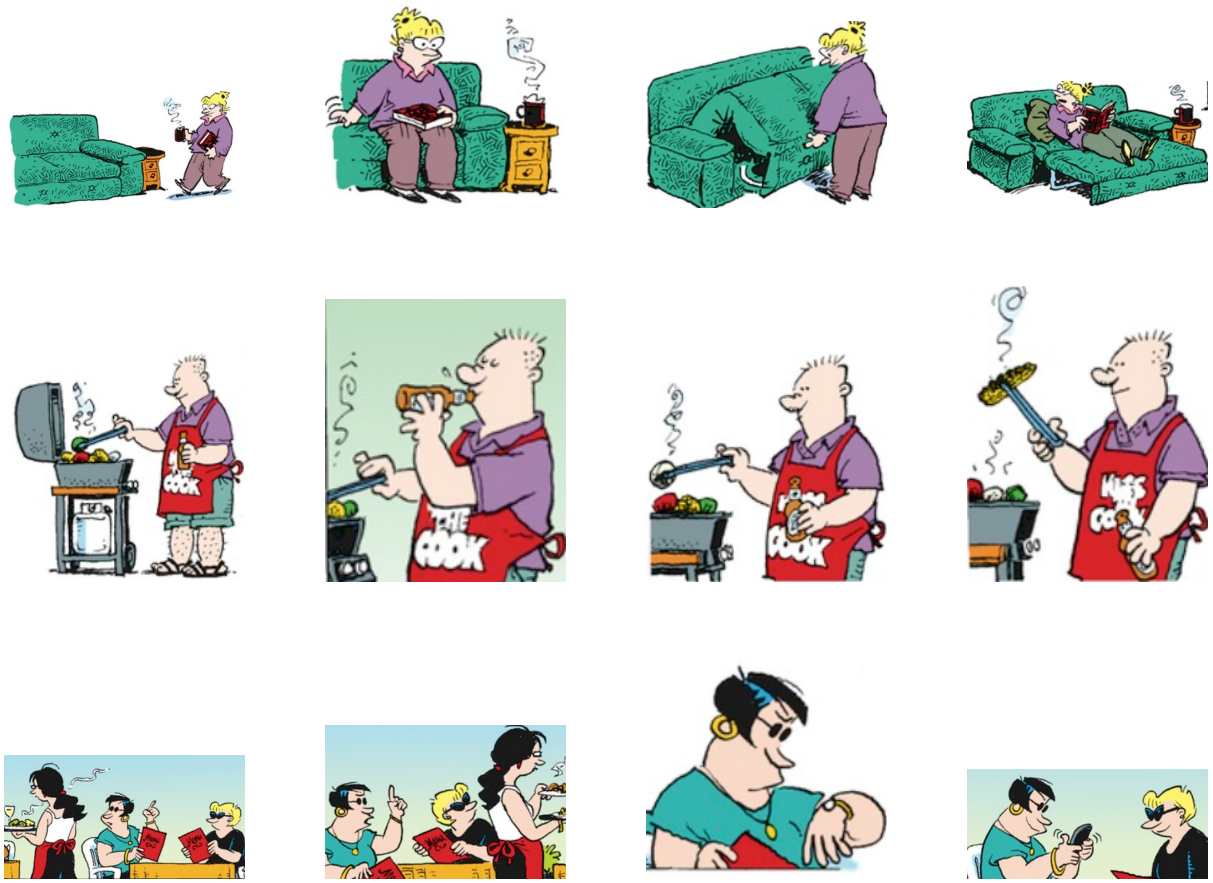


Figure 3.7: Three comics sequence highlight the variety of the comics they came from (Betty). *Image_33*, *Image_43* and *Image_45* in the dataset.

4 Literature Review

In the literature review, our objective is to give a brief overview on the work done related to ours. Firstly, we will present a review on deep learning methods applied to comics. Secondly, we revisit how deep learning image-to-text models has evolved, highlighting the main SOTA models throughout time. We also present the justification on the choices of the multimodal models who are candidates to be the backbone of our architecture, and detail the way they work.

4.1 Comics and Deep Learning

Deep learning applied to comics is essentially divided into two main areas: comics' elements identification and comics' scene understanding. The work in [25] offers an overview of the main deep learning methods applied to comics. We start by describing the former group.

The area where there is ample research is object detection, who relates to the identification of whether any pixel from the comics' image belongs to a speech balloon, caption or the image itself. In general terms, this task is named semantic segmentation. Figure 4.2 presents one example: the panel on the right is a mask where white areas relate to the prediction of the pixels who belong to balloons in the image on the left. In these type of tasks, the output is of the same dimension of the input, so models of the U-Net (pioneered in [26]) type are regularly used. The U-Net is composed of an CNN encoder who identifies the key feature maps for the problem, and a decoder which is the "reverse" of the encoder to map those pattern in the pixel location within the image. The model acts as a binary/multiclass identifier of whether a pixel belongs to a balloon or to the comics' image.

There are several papers who seek to train models to perform these tasks. For example, the work in [27] trains a model to perform semantic segmentation of speech balloons in 750 annotated pages of the Graphic Narrative corpus, while the authors in [28] aim to identify face and body detectors in cartoons. The paper [29] tries to do multitask learning for 3 tasks: panels and characters detection, balloons' segmentation and balloon-character association. Additionally, linking the balloons to the characters is also a relevant task and doing OCR to recognize text from the textual components of the image. This topic could be useful for our work to help in identifying a dataset suitable for our research questions. However, as it was mention in previous chapter, the

datasets coming from this deep learning pipelines are not suitable to answer our questions: the image becomes more noisy and the textual elements are dialogues and not narrative descriptors. There are other strategies to augment datasets too. For example, the paper in [28] addresses this problem by doing "cartoonization" of natural images' data, employing several types of style transfer techniques. Also, keeping textual elements might not be appropriate, as the model might perform better by having OCR abilities. The authors in [27] find that the evaluation on Japanese comics show worse results, which shows that the letters inside the balloons help the model in performing segmentation.

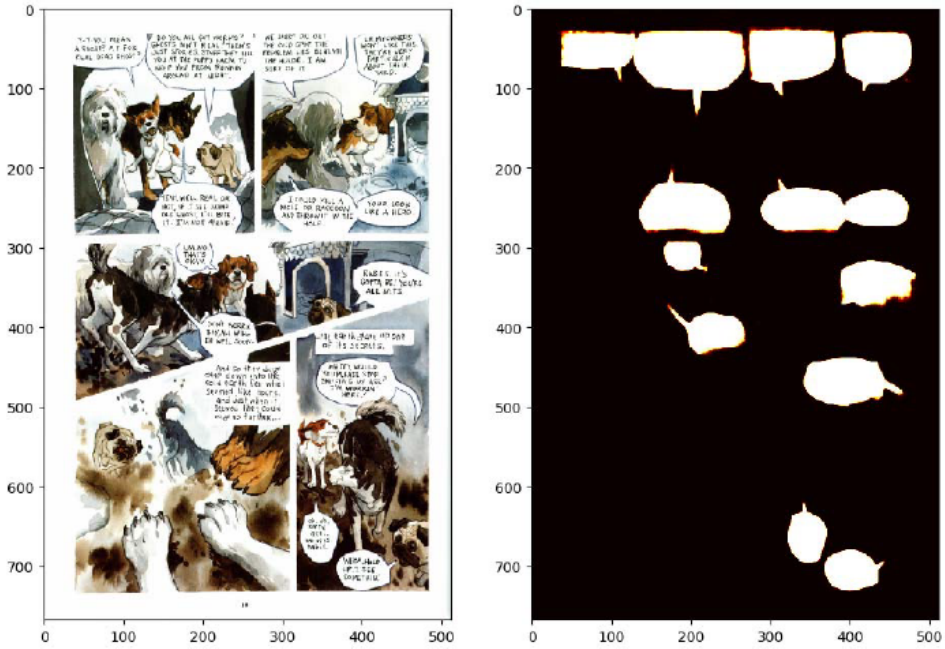


Figure 4.1: An example of segmentation of speech balloons. Source: [27].

We also highlight some work on the understanding of comics content. The authors in [30] employs two identification methods to perform anime style recognition on the dataset they created. The benchmark results they aim at building presents difficulties to achieve satisfactory results. This shows how more abstract features can pose additional challenges to SOTA architectures. There is also work who seeks to assess how different is to identify some patterns in comics and real images. The work in [11] addresses how emotions can be perceived by neural networks. They show that a CNN has some difficulties in identifying negative images in Japanese comics (manga). However the sample size is relatively small, which is an indication about the challenges we might face, because our dataset can also be considered small size (60 sequences of 4 images). This angle of understanding also motivated us to add a prompt in our pipeline where we ask for the emotions characters are experiencing. The paper in [31] also has some similarities with our work. Firstly, they have a dataset on 4-scene comics. Secondly, they seek to understand whether info from previous images helps in identi-

fying whether emotions are neutral in an image. They use a dataset with images from japanese manga, but translate the textual elements into english. Their results point to the ability of past information to improve the sentiment prediction of the model. Nevertheless the information from previous frame has almost the same performance as inputting the all 3 previous frames. In spite of some similarities, the last conclusion cannot be completely transposed into our framework, as our goal is to describe the whole narrative of the sequence, and not having previous information to help in understanding just one image.

Figure 4.2: Two comics sequences examples from [31].

Similar to some seminal papers on pre-trained models for benchmarking and context understanding, [32] seek to identify how to perceive the narrative from sequential comics' panels. They also build their own dataset on comics from the "golden age of comics" in the USA, isolate images and text. They perform panel and text-box segmentation by annotating random samples of the dataset and fine-tuning a type of CNN, but the main goal is to model the content from comics. They try to find how easy is for neural networks to learn the context of comics. They "show" the model a sequence of

images and text and give several options for the next element in the sequence. A couple of alternatives are tried: either to pick the right sentence that goes with the image, the image that goes next or the pairs image-text. The architecture who performs better is an hierarchical LSTM. The features of the images are extracted with several layers of an CNN type model. The taxonomy is also valuable: the intra-panel as well as the inter-panel dynamics is classified based on whether there is an action, or another character who appears. The results show how intricate and complex comics can be, as the neural network finds it difficult to pick the right continuation. The task becomes harder if the wrong options come from a nearby panel. The work in [33] extends this approach to the comparison comics and manga.

Regarding content generation, the literature is scarce. [34] uses a dataset from graphic novels from Japan. Manga follows a specific way of writing and image creation. They build a pipeline to isolate text from images, which includes semantic segmentation of the two elements and OCR of the words inside the balloons. The images are inputted into a generative adversarial network model for the image generation. The diversity of images led to convergence problems, so conditional training and transfer learning were employed. Regarding the text generation, a distilled version of GPT-2 is fine-tuned on text from comics and general literature. This is the first paper who tries to generate graphic novels from scratch, and a complete pipeline is built, where there is panel layouts and balloon speeches. Also worth mentioning several types of language to generate text, and the narration element was from a poetry type. Although the text and image are generated independently, it is advised that any future work must take into account the image and text consistency on a broader connection than just pure captioning. Text is not only a description of the image, but many times provides information to complement the understanding of the story.

Other relevant segment of research is the production of comics' images based on text descriptions. [35] build also on an adversarial network architecture. The initial idea was to use the text directly from the dialogues, but the dialogues proved to have no direct connection with the image, as there is no description of the characters and context. Moreover, the automatic creation of descriptions was also not suitable, as the models on description are trained on real photos. They needed to generate text description by themselves, with a direct semantic connection text-image: detailed character description and general context (like colour background and relative position). The investigation in [36] explores comics through the lens of multimodal conversation. Additional challenges: each character has his own style and persona; dialogues with more than two people; use of humor. It uses CLIP as image encoder, added to embedding of the previous dialogues and persona context. The model created (MPDialog) outperforms both models only using text from previous dialogues and adding persona description. Results hold on comics not used for training data. Figure 4.3 presents the architec-

ture. When compared to our work, there is more emphasis on encoding the textual part, with the context of the character. Our focus in on image comprehension, and the textual elements we use are one sentence prompts.

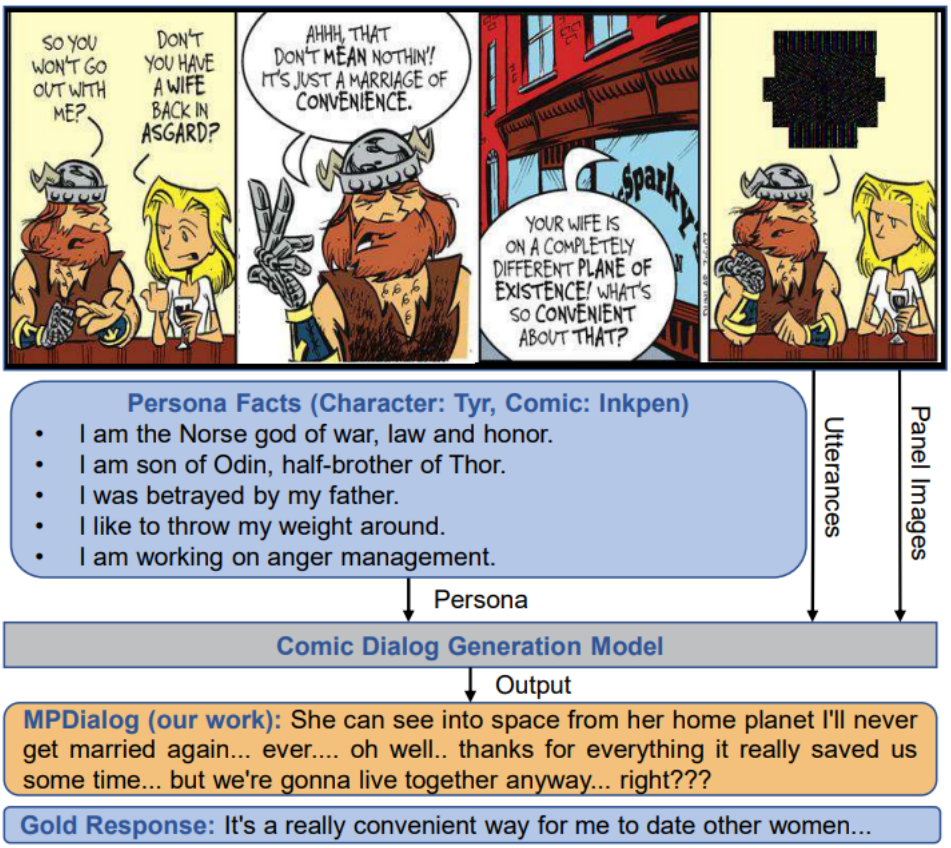


Figure 4.3: MPDialog architecture. One of the few papers about image-to-text models. Source: [36].

To sum up, the literature on comics comprehension is scarce. So far, we do not have any paper who proposes captioning sequence os comics' images without textual elements. When one narrows the search for image-to-text models, there is scarcity. So there is novelty in trying to inquiry whether deep learning models can as be as good at reading comics as they are at "real world" images.

4.2 Image Captioning and Deep Learning

Image-to-text models are at the core of the objectives of this work, so we provide an overview on how deep learning has evolved within this topic.

The last decade saw a staggering leap in image-to-text models. In fact, the dynamics mimics the same milestones from the natural language processing (NLP) field, with some lag. Is is not pure coincidence. In fact, image captioning can be interpreted as a kind of translation. But instead of translating a sequence of words into another sequence of words, one translates the image's "features" into a sequence of words.

Consequently, the encoder part of any image captioning architecture entails a computer vision model to identify the main attributes of the image; afterwards, the decoder part will connect to the words from the vocabulary. The encoder relates to the part of a neural network were inputs are compressed into a smaller dimension array of latent variables. The goal is to isolate inputs' crucial elements which are relevant for the task. In our context, the encoder part will be isolating image's relevant features for the task. For example, when identifying the action "playing baseball", one expects the model should be able to isolate the player, the special glove, or type of the ball. The ways to isolate this encoder part have changed throughout time, but we will leave this topic for the next paragraphs. If the task involves "just" describing the image in a word, for example to identify the race a cat, the job is done by the encoder part. However when the description of the image is intended to be more complex, the decoder element comes into play. The decoder part uses the latent variables as the building block to connect to a tokenized vocabulary of words. The decoder is therefore the generative part of the model, where text is produced.

The first image captioning models we highlight comprised a CNN as encoder and a RNN as decoder ([37]). The CNN part was the main architecture to image understanding before transformers. It acts as a sequential computation of feature maps from the image: each feature map seeks to isolate a relevant pattern of the image; as one progresses deeper into the CNN, feature maps usually become smaller and more complex. Going back to our baseball example, initial feature maps will identify the contours and the texture of the special glove; the feature map in the sequent hidden layers will join those two. However, this type of architectures (CNN+RNN) have problems in the decoding process, as image's feature maps are shown only at the beginning of the RNN "unrolling". Attention mechanisms had the first opportunity to show their key role, helping the RNN to focus on different elements of the image depending on the word being generated ([22]).

Resembling neural machine translation, attention mechanisms came from "just another feature" of the model to complete prevalence. The so called transformers' architecture took center stage also in this context, firstly focusing on the decoder part (for example [38]). Soon it was shown that the computer vision task could also be successfully addressed only with transformers ([39]). In fact, one of the most straightforward models based on transformers used a few years ago was the junction of the first encoder-only vision transformer ([24]) and the decoder generative pre-training transformer 2 (GPT-2, [8]), widely deployed by user of *Hugging Face*.

All models above learn have the embedding layer connected solely to an input which is either text or image, and do not blend the two. Sooner it started to become clear that there are advantages in having an embedding space trained jointly for both type of features. The embedding space is a way to compress the input information into

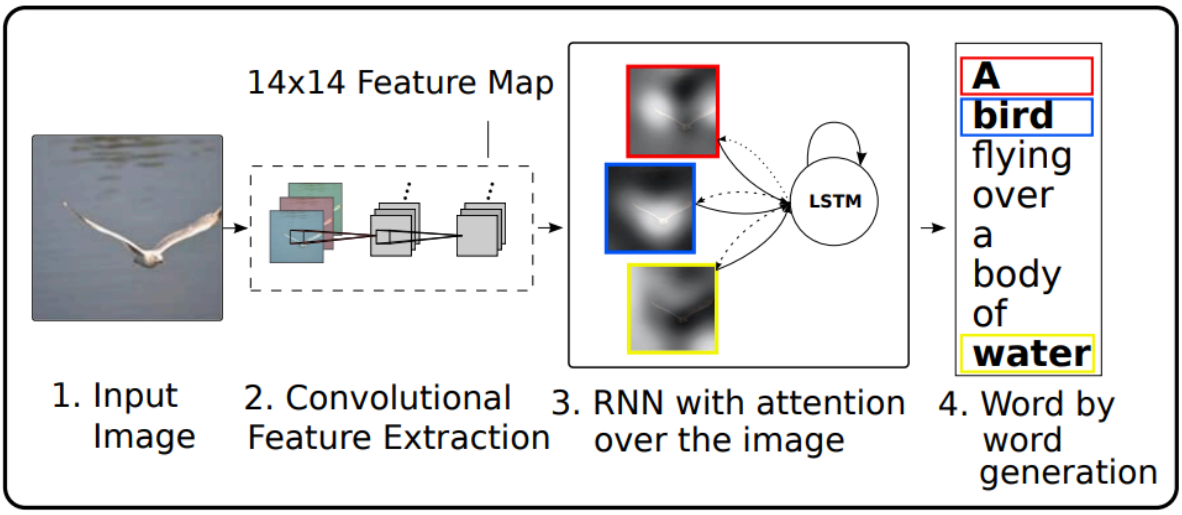


Figure 4.4: The architecture of the first seminal paper on image caption using attention. Source: [22].

a smaller dimension vector, before applying the encoder part of the neural network. The main purpose is to reduce dimensionality, as otherwise the number of parameters would be immensely larger: as attention mechanisms compute internal products, the number of parameters grows quadratically along the dimension of the vocabulary. The embedding space allows to compress information, implicitly decomposing any token into a set of vectors. When the input is an image, as in this work's case, images are split into patches, as splitting into pixels would make it much larger ([24]). The patches are afterwards input into the embedding layers.

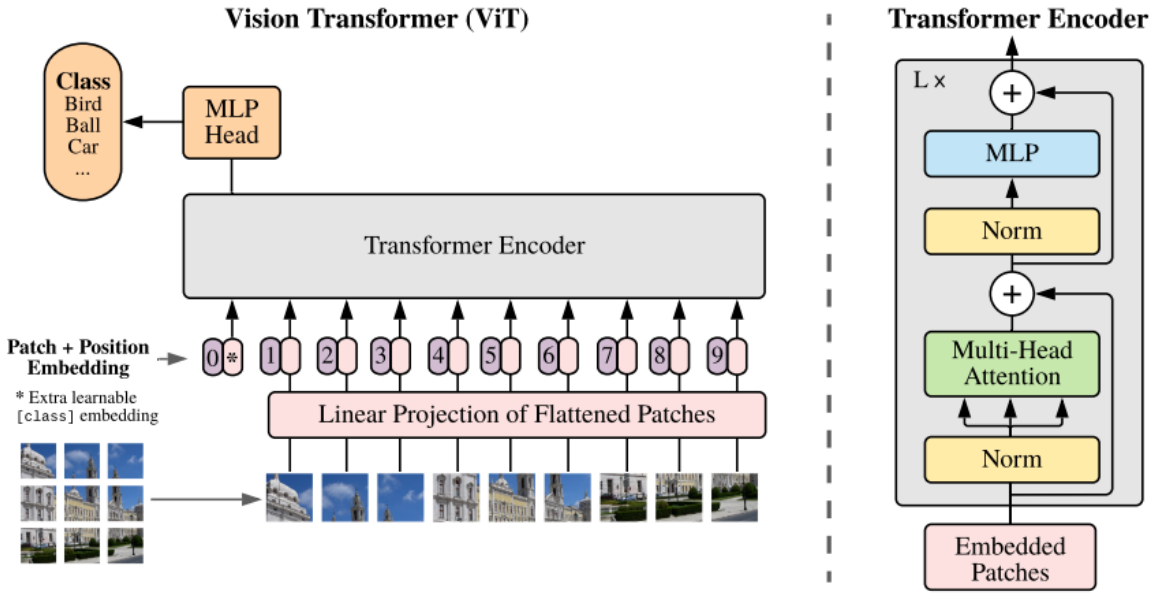


Figure 4.5: The architecture of the vision transformer. Source: [24].

The work in [24] presents the Vision Transformer (ViT). This work showed for the first time that one could have a vision model without convolutional elements embedded, only recurring to transformers. The patches of the image are processed within the ViT just like words are.

4.3 Multimodal Models

Multimodal models have their embedding space learned in the same semantic space for both images' patches and textual elements. Contrastive Image Pre-Training (CLIP, [4]) is one of the main references for multimodal models. CLIP is trained on 400 million image-text pairs, and its loss function not only seeks to maximize the similarities between the correct pairs, but also minimize it across text who do not matches with images, as can be seen in Figure 4.6. By understanding the context and content of the image, CLIP can be effective in performing zero-shot image classification and at helping defining which description best matches the image.

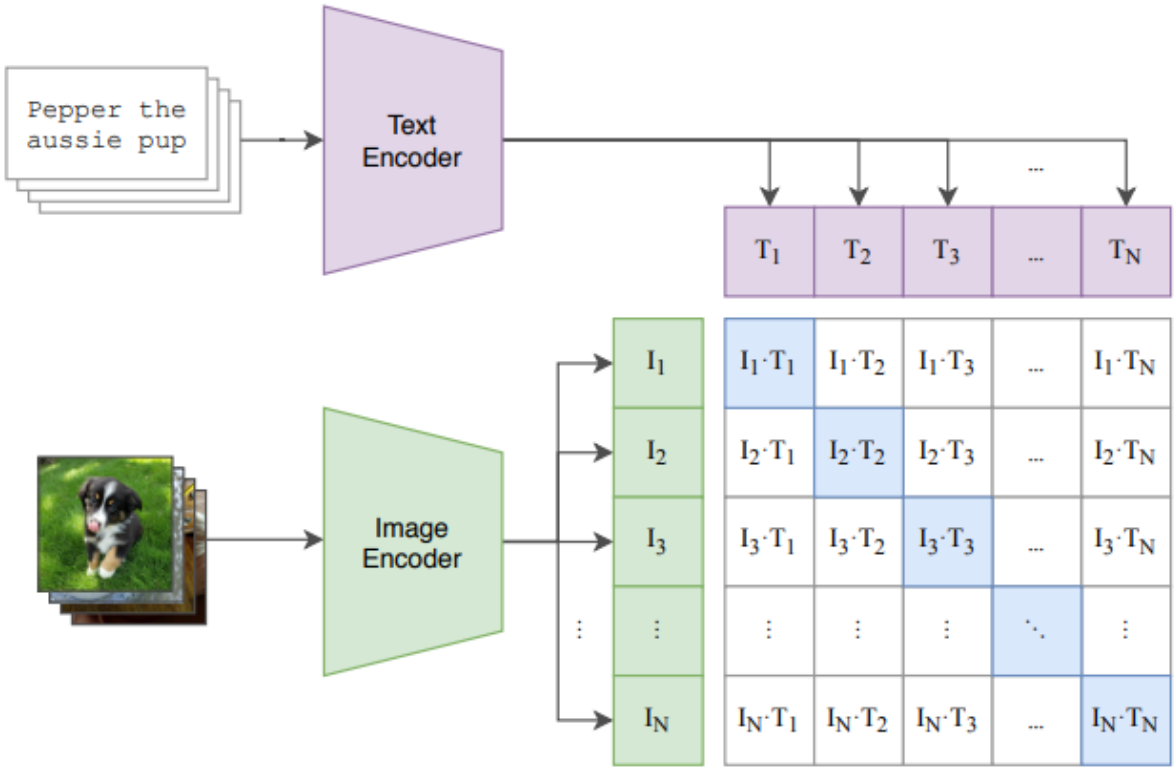


Figure 4.6: The pre-training objective of CLIP. Source: [4].

However, CLIP is not equipped for generative tasks, it is a image-text encoder. So a text decoder must be added to perform image-to-text generation. CLIPCap ([40]) is one architecture who seeks to accomplish a more detailed description of images. GPT-2 is the text decoder. A mapping network is used to connect the CLIP encoder with the GPT-2 decoder. GPT-2 starts to generate text upon the prefix embeddings. Two type

of versions are assessed for the mapping network: one in which both the mapping parameters and GPT-2 are trained; in the second the GPT-2 is also frozen. As the latter has the decoder frozen, the mapping network needs to be more complex. Therefore a transformer is used for the latter and a simple multi-layer perceptron is employed for the former. The main architectures of this type have similar building blocks as the one we have detailed.

When dealing with comics captioning, the type of language might require the language generation to flexibly focus on some details of the image that "sticking" to a specific architecture is not able to do. Therefore models who accomplish a more detailed description of the image might also prove useful. FuseCap ([41]) achieves it by creating a more detailed description of the image, through different vision encoders for different tasks (OCR, detection, attributes extraction). The paper [42] does something similar by merging the outputs from SOTA image captioning models. Showing a large language model (LLM) decoder a diverse input of word summary of photos is expected to be particularly relevant for comics.

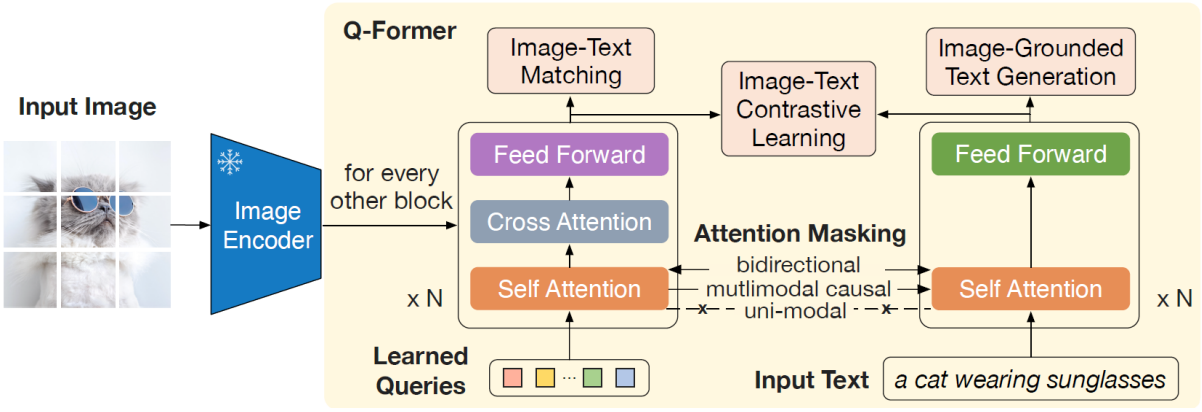


Figure 4.7: BLIP-2 architecture (Source: [3]).

In order to optimize computational resources, the ability of a model to do correct characterization of images not used during training is becoming relevant. The usually called zero-shot image captioning models allow for more efficiency in applying image-to-text models: they are able to perform tasks in a context they have not been trained on. Bootstrapping language-image pre-training (BLIP, [43]) is one of the current SOTA ones. The loss function comprises three components, whose goal is to align the image and text encoders and the text decoder. The "bootstrap" part comes from leveraging images from the internet to use more data to train the model. The noisy data from the web implies a filter to exclude the more noisy image-text pairs. BLIP2 ([3]) was later introduced to improve training efficiency and mitigate catastrophic forgetting. The architecture aims at efficiency in training, by keeping the parameters of the encoder and decoder model frozen. The trainable parameters are part of a "bridge", the query transformer (Q-former, Figure 4.7). The Q-former is trained in several steps.

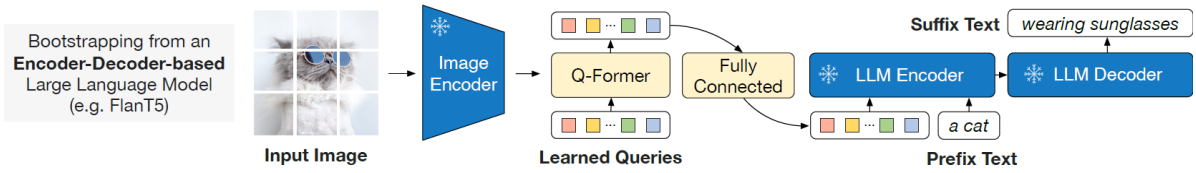


Figure 4.8: BLIP-2 decoding process. Source: [3].

BLIP-2 therefore improves on BLIP, by suggesting an architecture where a "bridge" is the only trainable part of the model. The bridge (know as "Q-former") tries to optimize the interaction between the model which "looks" into the figure and extracts the main features, and the language model which seeks to output text. This approach is proposed in order not to re-train the whole set of weights of the model. By training only the parameters of the Q-former and not the parameter from both the image extractor and text generator, the goal is to train faster, as a smaller set of parameter is optimized. The LLM encoder-decoder connects to the image encoder through the Q-Former (Figure 4.8).

Due to its SOTA performance and zero-shot capabilities, BLIP-2 will be one of the candidate's model for the main backbone of the pipeline for comics' description. We now present the other one: Generative Image-to-Text Transformer (GIT, [7]). Figure 4.9 presents its architecture. GIT is known for its "simple" structure, reaching SOTA performance on several tasks related to image captioning, visual-question answering and video comprehension. The model is composed of one image encoder and one text decoder.

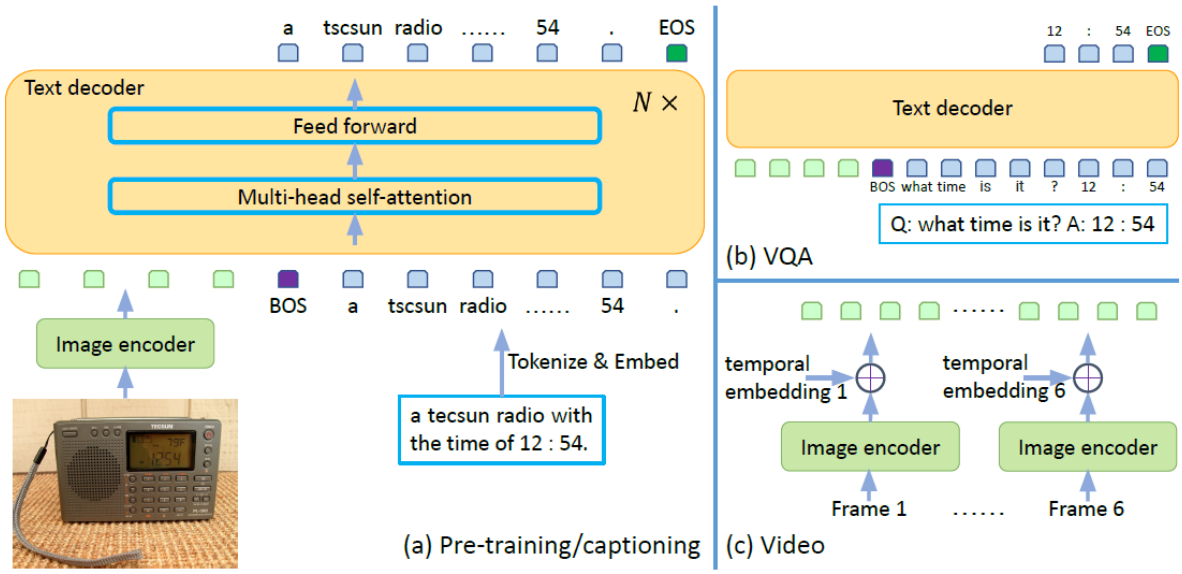


Figure 4.9: Overview of GIT ([7]) architecture.

The image encoder is based on a multimodal contrastive pre-trained model in [44], called Florence. This image encoder is trained similarly to CLIP, by contrastive learn-

ing, but it aims to add more fine-grained image representations. Also, there are some differences on the image encoder part of Florence when compared to CLIP. The decoder part of GIT is a standard transformer, were self-attention layers and feed forward are the standard elements. The type of masking is `seq2seq`, which means that a given word is decoded based on the feature maps of the image and previous words: the standard autorregressive decoding we see in GPT-2 for example (except in the latter there are no images). Overall, GIT has the advantage of using standard transformer layers and leverage an image encoder to reach good results on image and video comprehension.

The first stage of this process is to choose the multimodal model to caption the images' sequences. Here we have chosen two candidate models: GIT ([7]) and BLIP-2 ([3]). These two models were chosen as they fulfill two criteria. First, they are SOTA models with top performances; second, they are available on Hugging Face, which allows better usability. BLIP-2 is available in `https://huggingface.co/docs/transformers/model_doc/blip-2`, where one can find several components. GIT is in `https://huggingface.co/docs/transformers/model_doc/git`, both under the "multimodal models" tab.

Rank	Model	BLEU-4	CIDER	METEOR	SPICE	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	CLIPScore	Extra Training Data	Paper	Code	Result	Year	Tag
1	**mPLUG**	46.5	155.1	32.0	26.0						×	mPLUG: Effective and Efficient Vision-Language Learning by Cross-modal Skip-connections			2022	
2	**OFA**	44.9	154.9	32.5	26.6						×	OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework			2022	
3	**GIT**	44.1	151.1	32.2	26.3						×	GIT: A Generative Image-to-text Transformer for Vision and Language			2022	
4	**BLIP-2 ViT-G OPT 2.7B** (zero-shot)	43.7	145.8								×	BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models			2023	

Figure 4.10: The ranking of the best models in captioning the COCO ([5]) dataset.[1].

[1] `https://paperswithcode.com/sota/image-captioning-on-coco-captions`

In order to prove the SOTA performance of these two models, we refer to the most standard task in terms of image understanding: image captioning in a widely used dataset. Common objects in context (COCO, [5]) captions dataset is probably the main resource for anyone seeking to train and assess an image captioning model, comprising more than a million and a half descriptions of 330 thousand images. Figure 4.10 highlights the top four models with best BLEU-4 metric for the captioning the COCO dataset, aggregate in by the *Papers With Code* project (`https://paperswithcode.com/`). GIT and BLIP-2 stand in the thrid and forth place, respectively. Additionally, BLIP-2 achieves this result zero-shot: without being trained on the dataset.

We therefore conclude that GIT and BLIP-2 are two models with SOTA performance, employing SOTA architectures. As they are open source, we will use them as candidates for the backbone of our pipeline: we will prompt them to get a text description of the images. We will assess performance in captioning all 240 individual images, in a more "raw" task. We then pick the best for narrative description. It is also worth emphasizing that we do not use any of the two best models because they are not open source and linked in *Hugging Face*.

5 The Experiment: designing prompts, summarization and metrics

In this chapter we delve into the main steps and experimental choices of our work. We will detail the versions and hyperparameters of all deep learning models we employ within the whole pipeline. Moreover, we define and explain the metrics we use to assess how close are predicted and true narratives. Finally, we present the different prompts we employ, as well as the three strategies to flow the information through the images of the sequence.

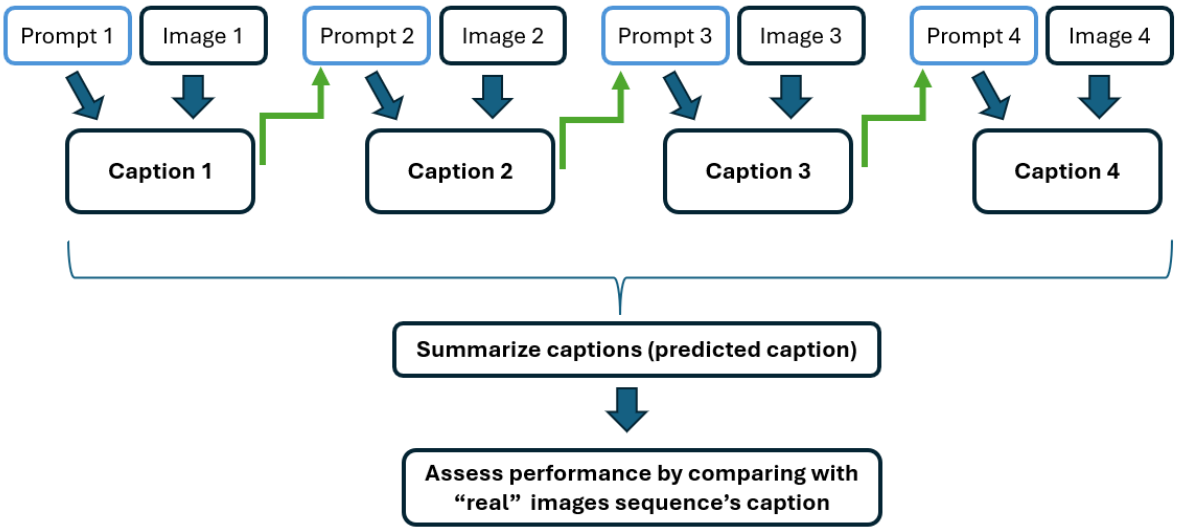


Figure 5.1: Schema of our approach to caption the comics' sequence.

Figure 5.1 summarizes our approach. In this section we will detail every connector of the narrative description pipeline:

- Prompting. The choices of the textual elements in the light blue box of the Figure 5.1:
  - The definition of the ten prompts we use;
  - How Prompts 2, 3 and 4 can embed element from previous captions, to help contextualizing (green arrow).
- Arrow who connects Prompts+Images to Caption: the multimodal deep learning model, which will be BLIP-2 or GIT. In this chapter we present the version and hyperparameters of these models.

- The summarizer which will aggregate captions 1, 2 3 and 4 into a consistent sentence to describe the narrative.
- The metrics which will assess how the final narrative predictor (the output of the summarizer) meaning is close to the real narrative description.

5.1 Multimodal Models

In this part we will refer which versions of the multimodal models we have used. The multimodal models are the first element of the pipeline, they receive the image and the prompt as inputs to output the predicted description of the image. As referred to in section 4.3, BLIP-2 and GIT are the two candidate models to be the image-to-text model. Figure highlights where in our pipeline is this located.

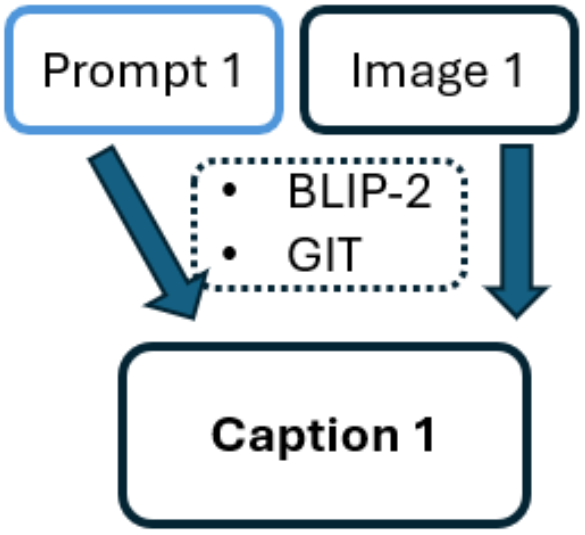


Figure 5.2: The role of the image-to-text model in our pipeline. Example for first image, but it works in an equal way for the other three images.

BLIP-2 uses two types of models for the text part. The decoder OPT (Open Pre-trained Transformer Language Models, [9]), or the encoder-decoder FLAN (Scaling Instruction-Finetuned Language Models, [6]). Both models exhibit good results in the paper. FLAN performs better in zero shot visual question answering (VQA), while OPT presents better results in zero shot image captioning and fine-tuned VQA. Therefore there no direct conclusion on the version to pick. Hence we opt for the model with the lowest number of parameters, to make inference faster. So the OPT version was chosen, which overall has 3.1 billion parameters. It is also relevant to add in this context the main advantage of BLIP-2, by using the trainable Q-former as the only part which is not frozen: only 104 million parameters were trained.

Regarding GIT, we refer to the baseline model , with 0.7 billion parameters. It is worth mentionig the large diffence in overall parameters between the version of BLIP-2 and

GIT. CPU based inference takes a long time to run, so we have used GPU withing *Google Colab* for inference.

5.2 Prompting

Prompting plays a key role in our analysis and finding the best prompt to get an output closer to reality is a relevant contribution of our work.

Table 5.1 presents the ten sentences we defined as prompt. They are already is the format *Question? Answer:*, as is suggested by the authors of BLIP-2 ([3]) to use as template for any prompt to the model version with the OPT decoder. The main criteria to define the prompts:

- Use questions whose semantic space relates to capturing the narrative
- Be as general as possible in the prompts, to avoid any additional "nudge" to the model and loss of generality
- Reference to the word "cartoon" was used
- Try to address different angles in the questions, to check which type of perspective might lead to better performance

Table 5.1: Prompts used to caption images.

Number	Prompt
0	Question: Identify all elements in image. Answer:
1	Question: What actions are performed in the cartoon? Answer:
2	Question: Describe the content of the cartoon. Answer:
3	Question: Caption this cartoon. Answer:
4	Question: Use one sentence to caption the cartoon. Answer:
5	Question: Describe the scene. What are the characters doing? Answer:
6	Question: What emotions are shown in the characters' expressions? Answer:
7	Question: What is happening in this scene? Answer:
8	Question: What might the characters be saying or thinking? Answer:
9	Question: Create a funny caption for this cartoon. Answer:

One can add previous images' outputs to current prompt. The idea is that adding previous captions might improve the semantic space of the prompt, allowing for a more aligned output of current image. Figure 5.3 presents the three options we employ to connect prompts with previous outputs:

1. Leave that task to the summarizer at the end to gather all captions and try to build a summary consistent with the narrative (no green arrow in figure 5.1);
2. Just concatenate the previous output to the prompt who is in use;
3. Add some context, referring to the number of the image.

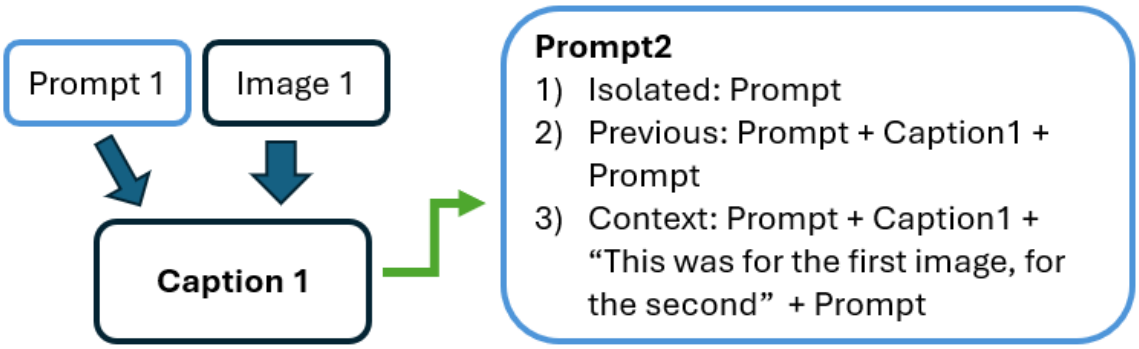


Figure 5.3: The three strategies employed to link current prompt to previous outputs/prompts.

5.3 Summarization

We use the *Hugging Face* pipeline on summarization (`https://huggingface.co/docs/transformers/tasks/summarization`). The summarizer is the element which receives the outputs of all images from the same sequence, and summarizes them into one sentence (preferably). It refer to the part we highlight in Figure 5.4.

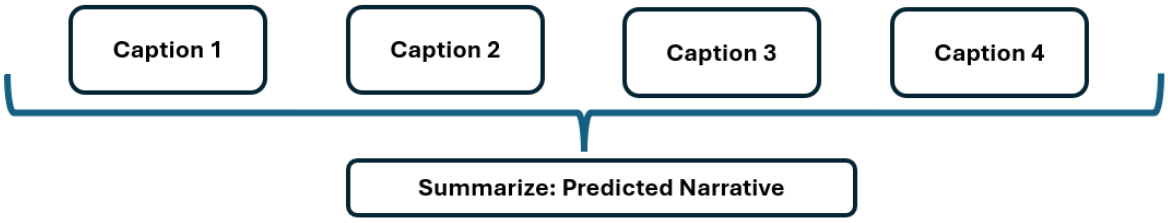


Figure 5.4: The role of the summarizer in our pipeline.

We now move into describing how the summarizer works and how it was trained. The summarization pipeline we employ is the fine tuning of the Bidirection Autoregressive Transformer (BART, [1]) model on the samsung dataset ([45]), available on `https://huggingface.co/philschmid/bart-large-cnn-samsum`. The model is a language model who is a denoising autocoder, trained by Meta (previously facebook AI). An autoencoder is a model where some text is corrupted, by several strategies, and the goal is to have a model who retrieves an output similar to the initial text before corruption. The transformer architecture is used to try to retrieve and output which is similar to the text before the corruption. By using different types of noising transformations, the model will learn to perform in a complete way, as shown in Figure 5.5.

The pipeline we use fine-tuned the BART specifically for a text summarization task, having SAMSum ([45]) as dataset (Figure 5.6 presents an example of the type of text it relates to). The dataset comprises dialogues and summaries. We choose this fine-tuned version because it resembles the problem we have in hands. Like in dialogues, we might find captions which are quite different between the images of the sequence.

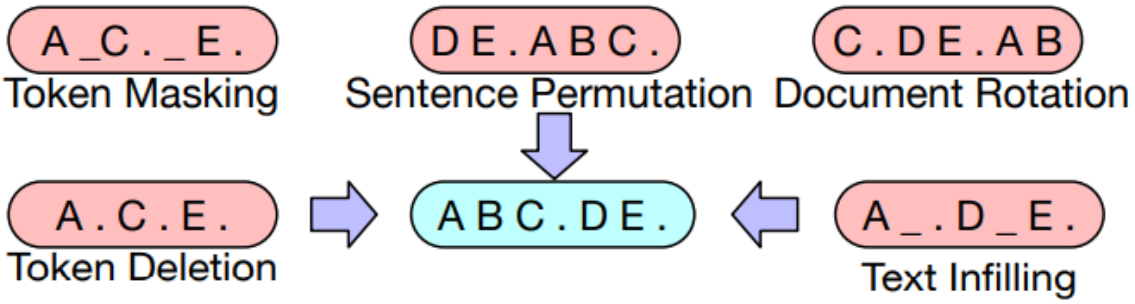


Figure 5.5: The several text transformations in BART model. Source: [1].

Aggregate the information from several captions to build a sentence describing the narrative seems to be a task close to summarizing the main ideas of a dialogue.

Dialogue
Blair: Remember we are seeing the wedding planner after work Chuck: Sure, where are we meeting her? Blair: At Nonna Rita's Chuck: Can I order their seafood tagliatelle or are we just having coffee with her? I've been dreaming about it since we went there last month Blair: Haha sure why not Chuck: Well we both remmber the spaghetti pomodoro disaster from our last meeting with Diane Blair: Omg hahaha it was all over her white blouse Chuck: :D Blair: :P
Summary
Blair and Chuck are going to meet the wedding planner after work at Nonna Rita's. The tagliatelle served at Nonna Rita's are very good.

Figure 5.6: Example of one data point from SAMSum dataset. Dialogues are inputs and summary is the target output. Source: [45].

The maximum length was set to be 15 words and the minimum 3. The goal is to have the narrative summarized in one sentence.

5.4 Metrics

There are several metrics to assess the performance of captioning models. The widely used bilingual evaluation understudy (BLEU, [2]) measures how many sequences of

N words are common between the "true" caption and the generated one. The metric for evaluation of translation with explicit ordering (METEOR, [46]) takes into account stemming and synonyms when comparing how the two sentences are similar. Consensus-based image description evaluation (CIDEr, [47]) is specifically designed for image captioning, putting emphasis on the most informative facts in common with several possible captions. In fact, some models use CIDEr loss in a second stage, to help fine tune the description not only on a "raw" word-to-word basis but also giving relevance to overall meaning. Semantic propositional image caption evaluation (SPICE, [48]) goes beyond the simple overlap of words and tries to identify several propositions that describe the "truths" in the image. The score then measures how similar is the semantic propositional context.

We choose BLEU to use as metric for assess the predicted captions of the work. BLEU is the simplest metric, and can be a measure to directly assess how many words the predicted and real captions have in common. We use a single gram to evaluate captions. If the models prove to be successful by having very high BLEU 1-gram, we can move into higher grams.

BLEU is computed in the following way:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{5.1}$$

- BP (Brevity Penalty): A penalty factor to discourage translations that are too short, defined as:
$$BP = \begin{cases} 1 & \text{if } c > r, \\ \exp(1 - \frac{r}{c}) & \text{if } c \leq r, \end{cases}$$
where c is the length of the candidate translation and r is the length of the reference translation.
- p_n: The precision for n-grams, calculated as:
$$p_n = \frac{\text{Number of matching } n\text{-grams}}{\text{Total } n\text{-grams in candidate}}$$
- w_n: Weight assigned to the n-gram precision scores, typically $w_n = \frac{1}{N}$ for uniform weights.
- N: The maximum n-gram size considered in the computation (commonly 4, i.e., up to 4-grams).
- $\exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$: The geometric mean of the n-gram precisions.

In summary, the BLEU score combines the precision of matching n-grams with a brevity penalty to measure the similarity of the candidate translation to one or more reference

translations. In our initial approach, we will use only 1-gram, which makes the formula simpler.

But we need to complement this "literally literal" measure. We also seek to take into account that there are captions whose sense is close, even if they do not have any word in common. We decide to use the hugging face sentence similarity space (`https://huggingface.co/tasks/sentence-similarity`) to encompass this evaluation idea. Sentence similarity builds on the embedding space of the words comprising the sentences to assess how close is the meaning of two sentences.

We refer to the model `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2` used.

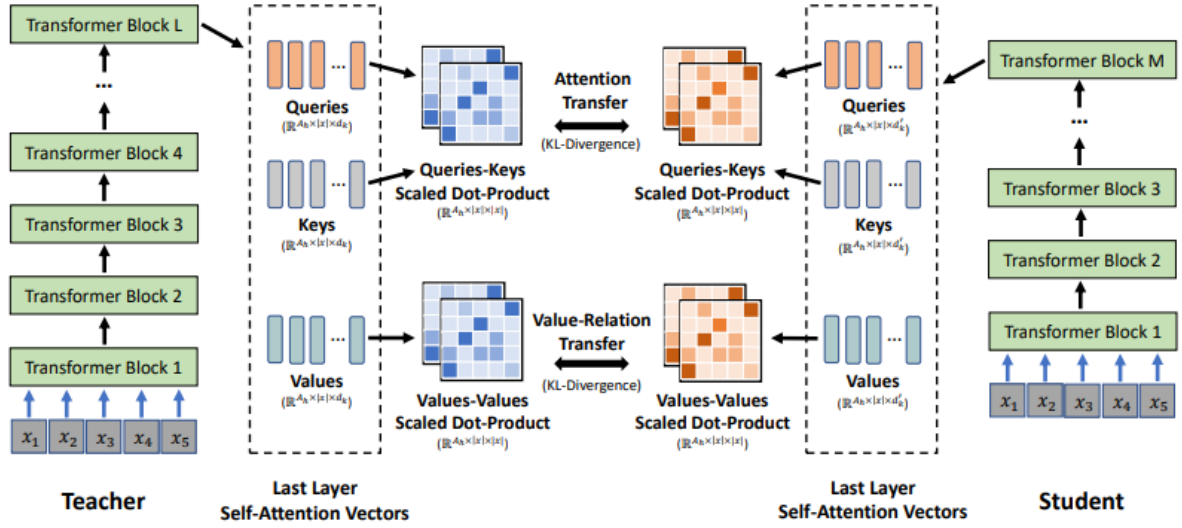


Figure 5.7: The architecture of the model which computes embeddings for the sentence similarity score. Source: [49].

Sentence similarity comprises two essential steps. The first one relates to compute the embeddings of the sentences. Embeddings are a way to sum the sematic relationships among words in several numbers. In some sense, each embedding dimension will try to sum how close are the words in, as we previously mentioned. There are several approaches to have an embedding model. Nowadays the most common one is to use the one from the training of an LLM. Usually embeddings are the first layer in the transformer models, having also elements which take into account the position of the word. Embeddings are crucial to make transformers able to be trained, as they compress the relationships between words in a dimensional set which is small than the length of the vocabulary.

After having the sentences transformed into a tensor (array) where embedding numbers are, there needs to be a measure of how close are two vectors. Cosine similarity is the metric computed. Some models are also trained having cosine similarity as objective function, for example CLIP.

We define the computation of the sentence similarity metric.

We use a generic function `Represent` with a language model (`LLM`) to compute the embedding for a sentence:

$$\mathbf{E}_s = \text{Represent}(\texttt{LLM}, s) \tag{5.2}$$

where:

- $\mathbf{E}_s$ is the embedding array for the sentence s.
- `LLM` is the pre-trained language model (in our case [49]).
- Represent is the embedding extraction function.

The cosine similarity between the embeddings of a predicted sentence $\mathbf{E}_{\text{pred}}$ and a true sentence $\mathbf{E}_{\text{true}}$ is computed as:

$$\text{Similarity}(\mathbf{E}_{\text{pred}}, \mathbf{E}_{\text{true}}) = \frac{\mathbf{E}_{\text{pred}} \cdot \mathbf{E}_{\text{true}}}{\|\mathbf{E}_{\text{pred}}\| \|\mathbf{E}_{\text{true}}\|} \tag{5.3}$$

where:

- $\mathbf{E}_{\text{pred}} \cdot \mathbf{E}_{\text{true}}$ is the dot product of the two embedding arrays.
- $\|\mathbf{E}_{\text{pred}}\|$ and $\|\mathbf{E}_{\text{true}}\|$ are the Euclidean norms of the predicted and true embeddings, respectively.
- The resulting similarity score ranges from -1 to 1, where 1 indicates identical embeddings and -1 indicates completely opposite embeddings.

This approach provides a numerical score to quantify the semantic similarity between two sentences, leveraging pre-trained language models and cosine similarity.

6 Results

This chapter presents the results. We start with the performance of BLIP-2 and GIT in predicting the individual captions, and pick the one with best metrics. Afterwards we introduce the benchmark results for narrative prediction, stemming from showing the merged sequence as one image and the 4 images individually but without prompt. Finally, the pipeline is assessed across several prompts, assessing the performance both in absolute terms and relative to benchmarks.

6.1 Multimodal model

How to choose the right multimodal model? Our goal is to start with zero shot performance assessment, but one wants to have the option to go beyond and perform some optimization in the parameters, so an open-source model is preferable. Therefore we have used the *Hugging Face* platform, widely known platform for transformers' based models, to pick candidate SOTA architecture whose parameters are available. We got to two main architectures within the multimodality space: GIT and BLIP-2. In the literature review chapter, we also showed both have SOTA performance in image captioning tasks.

We will pick the one who performs better in captioning the individual images of the dataset (and not the sequences). We decided to assess at the individual caption level rather than at the sequence (narrative) level because:

- It allows for the quadruple number of instances to assess performance: 240 images *versus* 60 sequences;
- For the multimodal model to be assessed at the sequence level, the whole pipeline needs to be run (which makes it heavier);
- The subtask the models are evaluated is very similar to the overall task (describing an image); they differ on the viewpoint of the image(s) one is assessing. It would be unexpected to have the best model at individual level worse than at the sequence level.

Table 6.1 shows the results of the descriptive statistics of the BLEU-1 (except identified otherwise, when we refer to BLEU, we mean the BLEU-1 metric) score for both GIT and BLIP-2. Beyond the mean, we compute some quantiles of the distribution: the median (measures central tendency but less sensitive to outliers than the mean) and

the first and third quartiles, to compare different parts of the distribution. Individual captions are compared, so the descriptive statistics have 240 instances (60 sequences times 4 figures per sequence).

Table 6.1: Comparison of the BLEU score for individual caption performance.

	25th Percentile	Mean	Median	75th Percentile
GIT	0.00	0.08	0.08	0.13
BLIP2	0.00	0.12	0.10	0.20

The BLEU score distribution is higher for the BLIP-2 model over all metrics, except the first quantile, where it is 0 for both. Hence there are more than a quarter of the captions where both models cannot have one word in common with the "real" caption. The advantage of BLIP-2 over GIT is larger in the mean than the median (4 against 2 percentage points difference), so there should be some captions where the differences are larger. In fact, there seems to be larger distinctions on performance in the best ones, as the 75th percentile for GIT is 0.13 and BLIP-2 is 0.20.

Table 6.2 computes the model assessment for the sentence similarity metric.

Table 6.2: Comparison of the sentence similarity score for individual caption performance.

Model	25th Percentile	Mean	Median	75th Percentile
GIT	0.14	0.29	0.27	0.41
BLIP-2	0.30	0.41	0.41	0.55

The differences between GIT and BLIP-2 is more even across the whole distribution, as BLIP-2 overcomes GIT by 12 to 16 percentage point for all 4 statistics. Figure 6.1 presents some examples of captions where BLIP-2 performs much better than GIT. From these 4 anecdotal evidences (which then the metrics confirm) one can infer that GIT seems to have some problems in identifying some of cartoons' elements: it labels the sun as lemon (Figure 6.1a); the man's hand as a cat (Figure 6.1b); the chess as a shoulder (Figure 6.1c); and a man as a cat (Figure 6.1d).

One remark to the fact that the results so far look into the distributions of each metrics within each model. There could be a chance that individually there can be some captions where GIT is better than BLIP-2, but the picture might be blurred by the distributions. Hence we computed the difference in score for each individual caption, and again calculated the same metrics. Table 6.3 shows theses results.

Table 6.3: Comparison of the individual captions' differences in scores.

	25th Percentile	Mean	Median	75th Percentile
BLEU	-0.10	-0.04	0.00	0.00
Sent. Sim.	-0.25	-0.12	-0.10	0.02

(a) True caption: *Rabbit with sunglasses is hot on the sun*.
BLIP-2 caption: *a cartoon rabbit with sunglasses and a sun in the sky*. BLEU: 36%; Sent. Similarity: 83%.
GIT caption: *a cartoon of a rabbit with a yellow lemon in his mouth*. BLEU: 8%; Sent. Similarity: 41%.

(b) True caption: *A man playing chess*.
BLIP-2 caption: *a cartoon of a man playing chess*. BLEU: 43%; Sent. Similarity: 82%.
GIT caption: *a cartoon of a man with a cat in his mouth*. BLEU: 9%; Sent. Similarity: 11%.

(c) True caption: *Old woman with spider on the background*.
BLIP-2 caption: *a cartoon of an old woman with a spider on her head*. BLEU: 33%; Sent. Similarity: 80%.
GIT caption: *a cartoon man with a hand over his shoulder.*. BLEU: 11%; Sent. Similarity: 13%.

(d) True caption: *Cat with suit running*.
BLIP-2 caption: *a cartoon cat in a suit is running down the hallway*. BLEU: 18%; Sent. Similarity: 78%.
GIT caption: *cartoon of a man running to the door*. BLEU: 13%; Sent. Similarity: 29%.

Figure 6.1: Some examples of individual captions, where the true and captions predicted by BLIP-2 and GIT are presented.

We conclude that using individual captions scores' differences confirm our initial conclusions. A negative (positive) value comprises a score where BLIP-2 (GIT) has better performance. The metrics keep magnitudes similar to what we reported before. The average and mean difference for the sentence similarity score decreases somewhat, from 14-16 to 10-12 percentage points. This aspect highlights there are some individual captions where GIT is better than BLIP-2, as the positive third quantile shows (0.02). However, the magnitude is low. The BLEU metric change less, as GIT cannot outperform BLIP-2 in at least three quarters of the captions (the 75th percentile is zero).

We conclude that BLIP-2 performs better than GIT in the majority of the captions, and by a large margin on average. Therefore BLIP-2 will be the chosen model to be the multimodal backbone of our pipeline to caption the 4-image sequences.

6.2 Benchmarks

We define benchmarks to assess how the different prompt strategies generate better descriptions. Firstly, comparing the results of our pipeline with outputs from the 4 images input as a merged figure will show the (possible) value added of input the images sequentially. At first sight, this might seem odd when compared to the whole transformer revolution, which brought the evidence that showing all at once (instead of using a RNN) can produce better results. However, here we are talking about a different stage of the architecture, a higher layer where sequential comes from sequential image inputs, and not sequential words.

The second benchmark relates to input the image sequentially with no prompt, and summarize the four captions into a sentence describing the story. Here our goal is to assess the value added of prompting, and how metrics improve from these two baselines.

6.2.1 All at Once

Summarize the captions of each image of the sequence might be unnecessary, so we check how BLIP-2 performs if images are shown all at once. We define all as merging the images of the same sequence into one (with no additional adjustments) and a caption is output, both with and without prompting. This caption will have no additional post processing, and will be compared directly to the true narrative description. An example of the merged image with the whole sequence is presented in Figure 6.2.

Table 6.4 comprises the results without any prompt. This strategy could act as a third and possibly the easiest benchmark to outperform, as the model needs to grasp the idea of a large image and no prompting is given to help in the decoding process. The results are poor, as one would intuitively expect. The best quarter of sequences have a

Figure 6.2: The example of the image input all at once, whose label in the dataset is *image_1*.
Predicted narrative: *garfield the cat and his friends.*
True narrative: *It was not me who steal the steak.*
BLEU: 12%; sentence similarity: 21%.

low sentence similarity score, starting at around 30%. This is a low value, as the score accounts for some similarities in the semantic space that a human would not account to. For example, mentioning again Figure 6.2, the predicted caption receives a 21% score, even if for a human the sentences *"garfield the cat and his friends"* and *"It was not me who steal the steak"* seem to gave nothing in common. The predicted caption of this sequence also highlights how prompting can be effective: without any prompt, BLIP-2 will refer to the name of the cartoon. One can conclude that at least the model was trained in some Garfield cartoon, which might be a good indication for it to be able to grasp the narrative once proper prompting is given.

Table 6.4: Metrics for the benchmarks where the sequence is input all at once, with no prompting.

Method	Metric	25th Percentile	Mean	Median	75th Percentile
All at once	BLEU	0.00	0.06	0.03	0.11
All at once	Sentence Similarity	0.05	0.17	0.13	0.30

Figure 6.3 presents another example, where this strategy had the best sentence similarity score (61%). The term *baseball* was key for the good result, and the object being a key element of the narrative also provided additional performance. Although the model does not capture the idea that the glove was slipping, identifying the type of sport and the key object granted it a high score in this metric. In the remaining of this chapter, we will not use this benchmark to compare the results, as they set a very low bar and so comparing in relative or absolute terms would be almost the same.

Table 6.5 depicts the results from showing all images of the same sequence at once with the prompts identified in Table 5.1. Theses results will be crucial to assess the value added of sequential captioning with prompting, as well as having the first conclusions on what is the type of prompt which is better at capturing the narrative. Table 6.5 presents the results regarding the same statistical metrics for the distribution of the score. We remind that there are 60 sequences, hence for example the first quartile (or twenty fifth percentile) comprises the 15th worst score (or the 45th best). The results highlight that only some prompts are affective at improving the similarity score.

Figure 6.3: The example of the image input all-at-once, whose label in the dataset is *image_4*.
Predicted narrative: *a cartoon of a baseball player with a glove.*
True narrative: *The glove slip while trying to catch baseball ball.*
BLEU: 22%; sentence similarity: 61%.

Prompt 7 is the one who is best by a large margin, whereas the second is Prompt 3. The remaining prompts exhibit a score very similar to the one where there is no prompt, and there are some where prompt outputs worse results.

Table 6.5: Sentence similarity score for showing the image all at once, with prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.05	0.17	0.12	0.21
1	0.05	0.17	0.11	0.24
2	0.05	0.17	0.12	0.25
3	0.03	0.19	0.17	0.35
4	0.06	0.20	0.13	0.27
5	0.06	0.19	0.12	0.30
6	0.04	0.11	0.08	0.14
7	0.05	0.24	0.21	0.40
8	0.08	0.21	0.14	0.32
9	0.06	0.12	0.10	0.16

To finish the benchmarks, we would like to add that there are images where sentence similarities scores are negative. The narrative descriptions predicted by the model is so different from the truth that the meaning might be the opposite. However as we mention previously, a very low sentence similarity score usually means that there is no apparent relationship between the sentences. Therefore a negative score but above -10% just means there is no semantic proximity. In Figure 6.4 the sequence with worst sentence similarity score for both no prompts, prompt 3 and prompt 7. Even the own prompt predictions are very different across each other, so this is not a case where BLIP-2 consistently goes in the wrong direction, but goes wrong in very different ways (the similarity score across the 3 predictions is roughly 10%). This evidence might be an indication that the model was not trained on a sufficient amount of these type of images. In fact, this cartoon is different from the others: it shows no colors and the contours are more idiosyncratic.

The results highlight how relevant it is to have a sentence similarity measure. Although

Figure 6.4: Comparison of scores for image *image_13* (merged images).
True narrative: *Deciding on the shampoo in the shower.*
No prompt: *a drawing of a group of people standing in a line.* BLEU: 9%; sent. similarity: -9%.
Prompt 7: *The characters are fighting over a piece of land.* BLEU: 0%; sent. similarity: -6%.
Prompt 3: *The cartoon shows a man and woman in a car.* BLEU: 9%; sent. similarity: -6%.

all at once and sequential have close BLEU scores, the differences when one uses a semantic space metric become larger. When BLEU is low, the ability to identify whether some models are better than others by pure randomness is in jeopardy, as small and widely used vocabulary's words might become a relevant factor.

6.2.2 No Prompting, Sequential

We move into the first results where the images are shown sequentially. We define sequentially as inputting to BLIP-2 each image of the sequence individually. When the prompt does not concatenate with outputs from previous images of the same sequence, call this "sequentially" is a little misconception. When prompts do not depend on previous images, the images could be input each one in parallel to BLIP-2. However, it is relevant to keep the order of the output once they are concatenated to summarize. A different sentence order might have completely different output once the deep learning summarizer condensates the 4 output in one sentence to depict the narrative.

Table 6.6: Metrics for the benchmarks with no prompting.

Method	Metric	25th Percentile	Mean	Median	75th Percentile
Sequential	BLEU	0.00	0.07	0.08	0.10
Sequential	Sentence Similarity	0.09	0.24	0.21	0.38

We can compare Table 6.6 with Table 6.4 to assess how inputting the image individually instead of the sequence as a whole can generate better descriptions when no prompting is used. The results highlight how relevant it is to have a sentence similarity measure. Although all at once and sequential have close BLEU scores, the differences when one uses a semantic space metric become larger. When BLEU is low, the ability to identify whether some models are better than others by pure randomness is in jeopardy, as

small and widely used vocabulary's words might become a relevant factor. Showing the images sequentially is better than all at once with no prompting, as the sentence similarity score is higher across the whole distribution.

(a) "a cartoon of a woman taking a shower"

(b) "a cartoon of a man with a towel on his head"

(c) "a cartoon of a woman washing her hair"

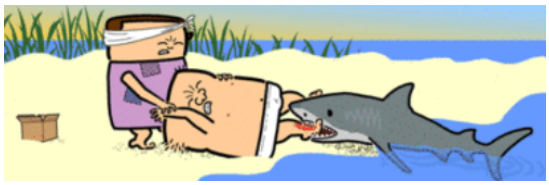
(d) "a cartoon of a man in a bathrobe looking at a computer"

Figure 6.5: Individual predictions and narrative prediction when no prompt is given. Proposed narrative: *"There is a cartoon of a woman taking a shower"*. BLEU: 10%; sent. similarity: 42%. *Image_13* in the dataset.

We also present one example where individual images perform better than all at once, and present the example of *image_13*, who previously was shown that the all at once models had negative sentence similarity scores. Figure 6.5 present both the individual captions and the prediction. The prediction of the narrative is the result of the application of the text summarizer pipeline from *Hugging Face* into the concatenated individual sentences. BLIP-2 was able to identify the semantic environment of the sequence, even if it had 42% score.

We conclude with the main summary of the benchmark section. The benchmark results present a baseline to assess the future results from the whole pipeline with prompting and sequential image input. The median values of the score distribution point to sentence similarities around 20%, which is a low value. Showing the images sequentially is better than all-at-once with no prompting, as the sentence similarity score is higher across the whole distribution. However, the best prompt (7) already exhibits an acceptable value at the forth quartile, around 40%, which is quite similar to the model who uses the whole sequence as image input. These two models have in fact quite close scores, but this hides a lot of heterogeneity across sequences. We computed the rank correlation between these two, and it is 31%. It is a low value, meaning that although they have distributions who have close metrics, the sequences at which they perform better are quite close. In Figure 6.5 we already show a sequence where individual input is much better than all at once output. We now present the opposite example, in Figure 6.6. Here the whole narrative relates to someone who was bitten by a shark. The individual caption with summarizer renders the output *a cartoon character in a purple dress is standing next to* (-6% sentence similarity) which means the summarizer, when faced with individual captions with apparent no connection, chooses to focus on the

sentence from the Figure 6.6b: *a cartoon character in a purple dress standing next to rocks*. Conversely, when BLIP-2 receives all images at once, it is able to focus on the idea of the shark bite as main event (albeit in a different version), which results in a 64% sentence similarity score.



(a) Predicted caption: *"a cartoon man and a shark on the beach"*

(b) Predicted caption: *"a cartoon character in a purple dress standing next to rocks"*

(c) Predicted caption: *"a cartoon of two people on the beach"*

(d) Predicted caption: *"a cartoon of a man and woman on the beach"*

Figure 6.6: Comparison of individual predictions and narrative prediction when no prompt is given for *Image_16*.
True narrative: *Hurt by a shark bite*.
Prediction from summarizer of individual captions, no prompt: *"A cartoon character in a purple dress is standing next to"*. BLEU: 0%; sent. similarity: -6%.
Prediction from prompt 7, input sequence in one merged image: *"The man is trying to get the shark to bite the other man"*. BLEU: 15%; sent. similarity: 64%.

6.3 Pipeline assessment

We move into the part where we assess the results of our prompting techniques, and how they improve over the benchmarks. We assess 6 prompting strategies, which we summarize in Table 6.7.

Figure 6.7 shows the results for the 60 sequences' median BLEU score across the several strategies. We put the benchmarks in bars and line, to make them an easier visual reference. The blue bars comprise the "all-at-once" caption with prompting, whereas the green line refers to the narrative built of individual images' captioning with no prompting. The green line is constant across the prompts, as no prompt was given in this approach. As it was concluded in the previous section, the green line is almost always above the blue bars, except for prompt 7 in some parts of the distribution. In

Table 6.7: Prompting techniques across different combinations. Names in the cells refer to legends in the figures of this section.

	Summarize the 4 images' output	**Summarize only the last output**
Same prompt	Isolated prompts, sequential	Isolated prompts, fourth caption
Prompt + Previous output	Previous captions	Previous captions, fourth caption
Prompt + Previous + Connector	Previous captions with context	Previous captions with context, fourth caption

the median, by using prompt 7 and showing the sequence all at once, it is the prompt who makes the sentence similarity score the closer to the narrative out of the individual captions with no prompt (the x value where the blue bar is closer to the green line). This conclusion is common both for BLEU and sentence similarity scores (Figures 6.7, 6.9).

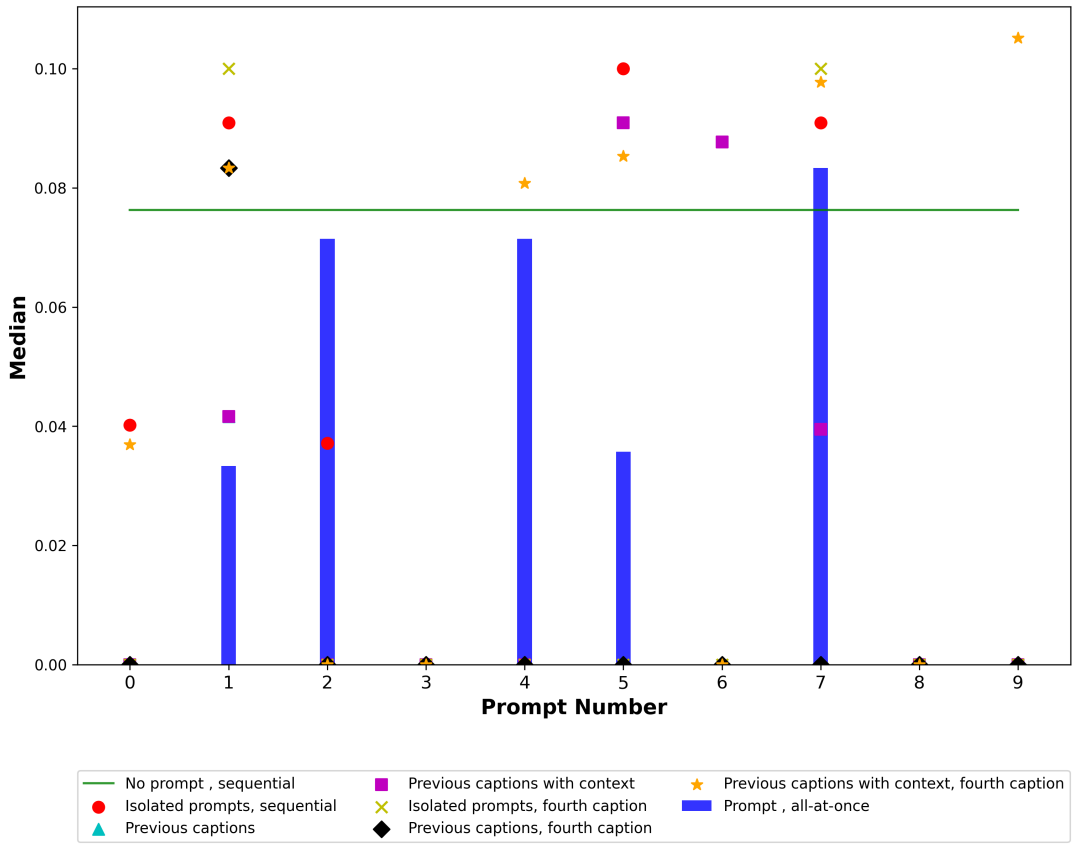


Figure 6.7: Median BLEU score.

However, again the BLEU score might be inconclusive, as almost all models lead to scores smaller than 0.1: on the median images' sequence, the predicted captions have 10% of the words in common with the true ones (Figures 6.7). This is a very low number which again leads us to rely more on the sentence similarity score. When one looks

into the third quartile (Figure 6.8 the BLEU score improves (it could not be lower, as the third quartile is always larger or equal than the median, by definition), but continues to show relatively low values, on average on the 10% to 15% range. Therefore even the one quarter top narrative descriptions created by the model have a small number of words in common with true narratives. When BLEU is low, its variability across different strategies might be led by differences in common words across narratives, and not the "key" words who describe the story of the sequence.

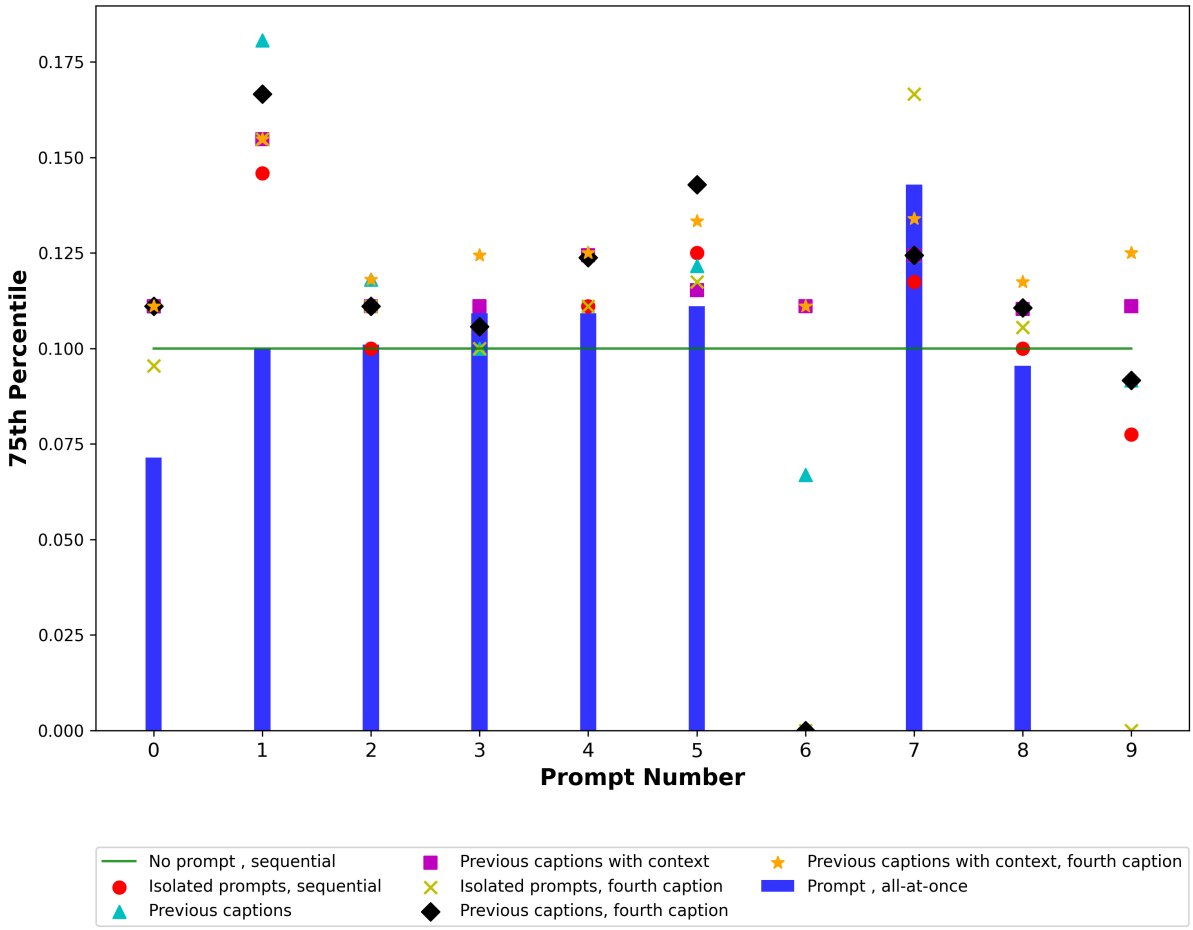


Figure 6.8: 75th percentile's BLEU score.

Figure 6.9 changes the metric to the median similarity score across the sixty sequences. Prompt 7 consolidates its indication as best prompt. Additionally, adding sequential prompting improves the performance, making it reach a sentence similarity score around 30 %. However, the disadvantage of some prompts is reduced by a large margin when each image is input individually. This evidence might mean that prompting can generate more value added when the image input is more complex (the model is ingesting 4 images merged into one). Prompts 0 to 2 and 4 to 5 exhibit the largest improvements (markers are more distant from the blue bar). Nevertheless, they struggle in making better than no prompting (markers slightly above the green line, on average), so one can conclude that the better sentence similarity scores come from inputting

the images one by one, rather than using prompts 0 to 5. In fact, prompts 6 and 9 deviate BLIP-2 from the right path to describe the narrative, as they present poor results. Prompt 6 is indeed the worst consistently, and adding sequential image input even worsens the similarity with original captions. It is the only prompt where all marker are below the blue bar, meaning asking for comics' emotions image by image might blur even more the ability to depict the narrative. In sum, asking for the model to identify emotions seems not to be the right path to achieve a good caption. This might be because it is harder to identify emotions in comics (or at least the narrative of the comics is not built upon the emotions of the characters). A brief comment on the absolute value of the median sentence similarity score: it does not seem very high, highlighting the difficulties in having a model able to capture the narrative.

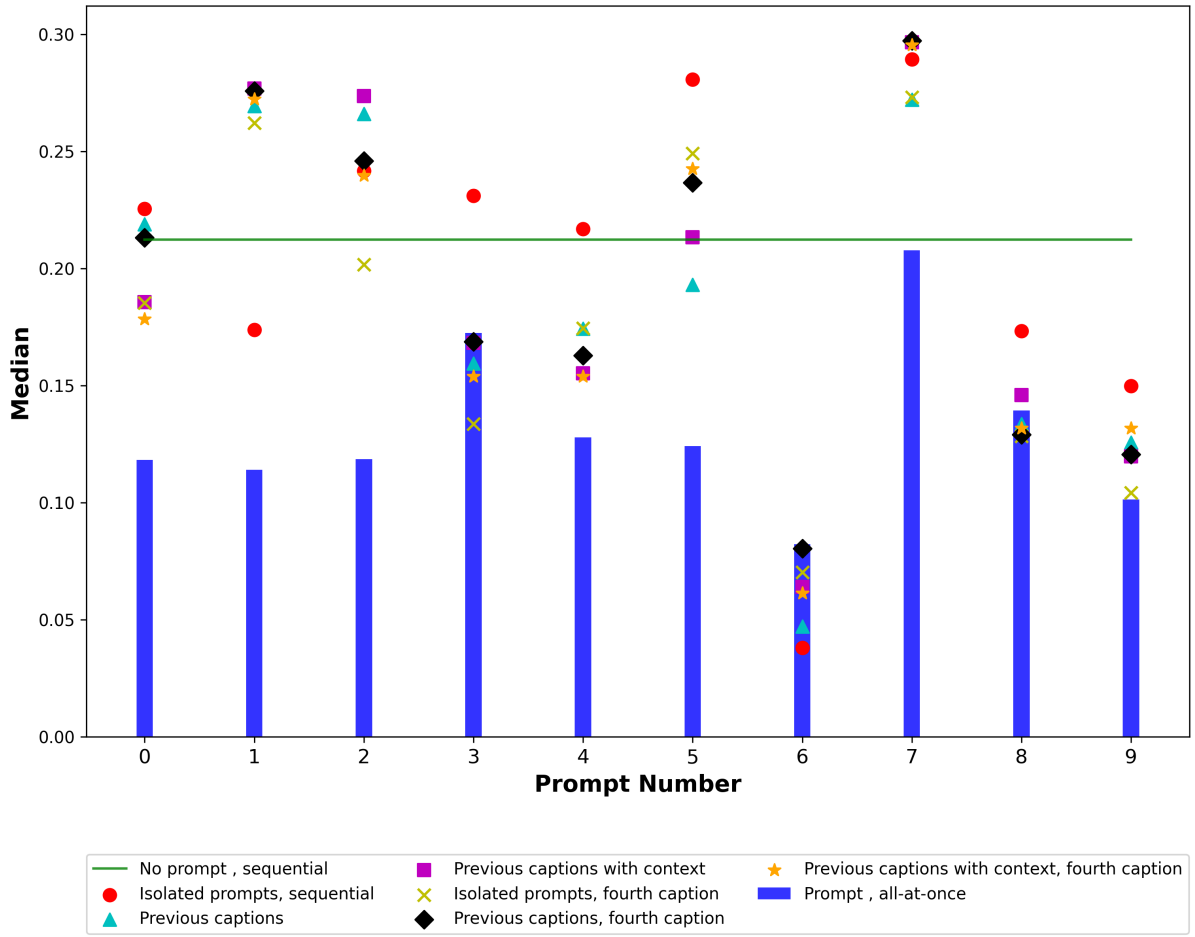


Figure 6.9: Median sentence similarity score.

Some prompts improve by a large margin when they are employed sequentially, compared to showing the whole sequence (the markers are very distant from the blue bars). Prompts 0, 1, 2 and 5 exhibit this pattern. BLIP-2 shows problems in extracting the narrative of the whole sequence at once, but once each image is input individually, the narrative comes closer to reality. All these four prompts rely on questions related to prompting the model into describing, identifying actions or elements. It is worth

emphasizing how different strategies perform within these prompts. In prompt 1, do isolated prompting image by image produces worse results, so it seems that prompting related to actions has large gains when the previous outputs add to current prompts. However, the output from the forth image whose prompt does not add previous elements

Prompt 2 is the only one where the most complex prompt strategy performs best (the violet square is above all other markers). Asking for the model to describe the content seems to connect well with previous outputs to help the model in capturing the narrative. This can also help the summarizer building a consistent narrative in one sentence.

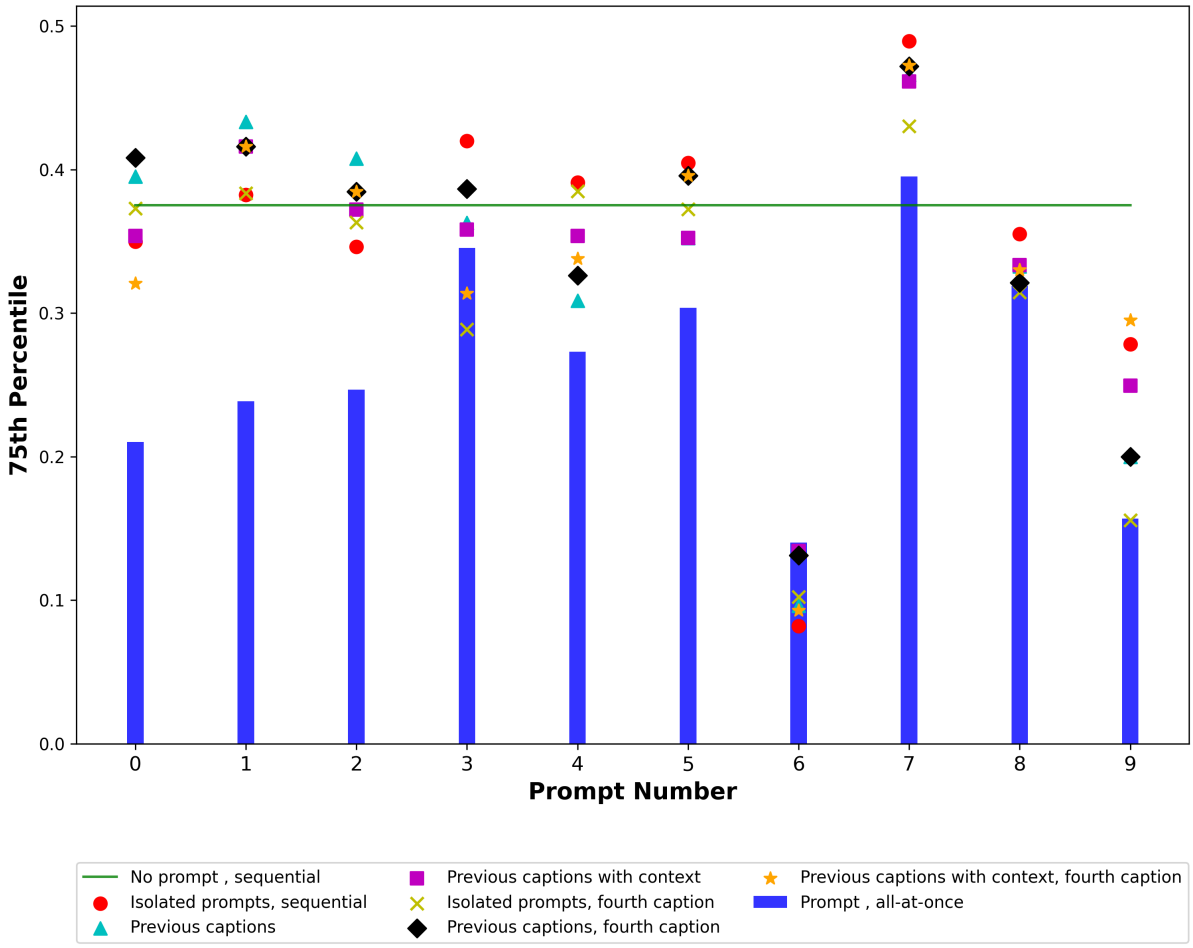


Figure 6.10: 75th percentile sentence similarity score.

Is is also relevant to stress several prompts where isolated prompting performs better than other ways to create prompting (there are several x values where the red circles are above the remaining of the markers. For these prompts (3, 4, 5, 8 and 9), adding previous outputs to the prompts might be producing more noise for the language decoder to perform its task in a successful way. Finally, prompt 5 with isolated prompts comes very close to the best prompt (7).

Figure 6.10 show the results for the 75% percentile of the similarity score. The strategy were no prompts are given but images are input sequentially shows roughly an 15 percentage point improvement in similarity score, highlighting that there are a group of sequences were there is still a significant improvement from the group around the median. Moreover, the markers are on average closer to the green line, which may indicate that the sequences around this percentile are easier to describe by BLIP-2, and prompting does not make a larger impact. However prompt 7 continues to be the best, and the only one that guarantees that showing the sequence all at once can have better descriptions of the narrative that sequential with no prompting. Prompting and showing images sequentially leads to improvements, with the best strategy having a similarity score very close to 50%. The best strategy is the red circle, which is the prompting with no information from previous images of the same sequence: for the best prompt, adding previous images' output seems to make the description process noisier.

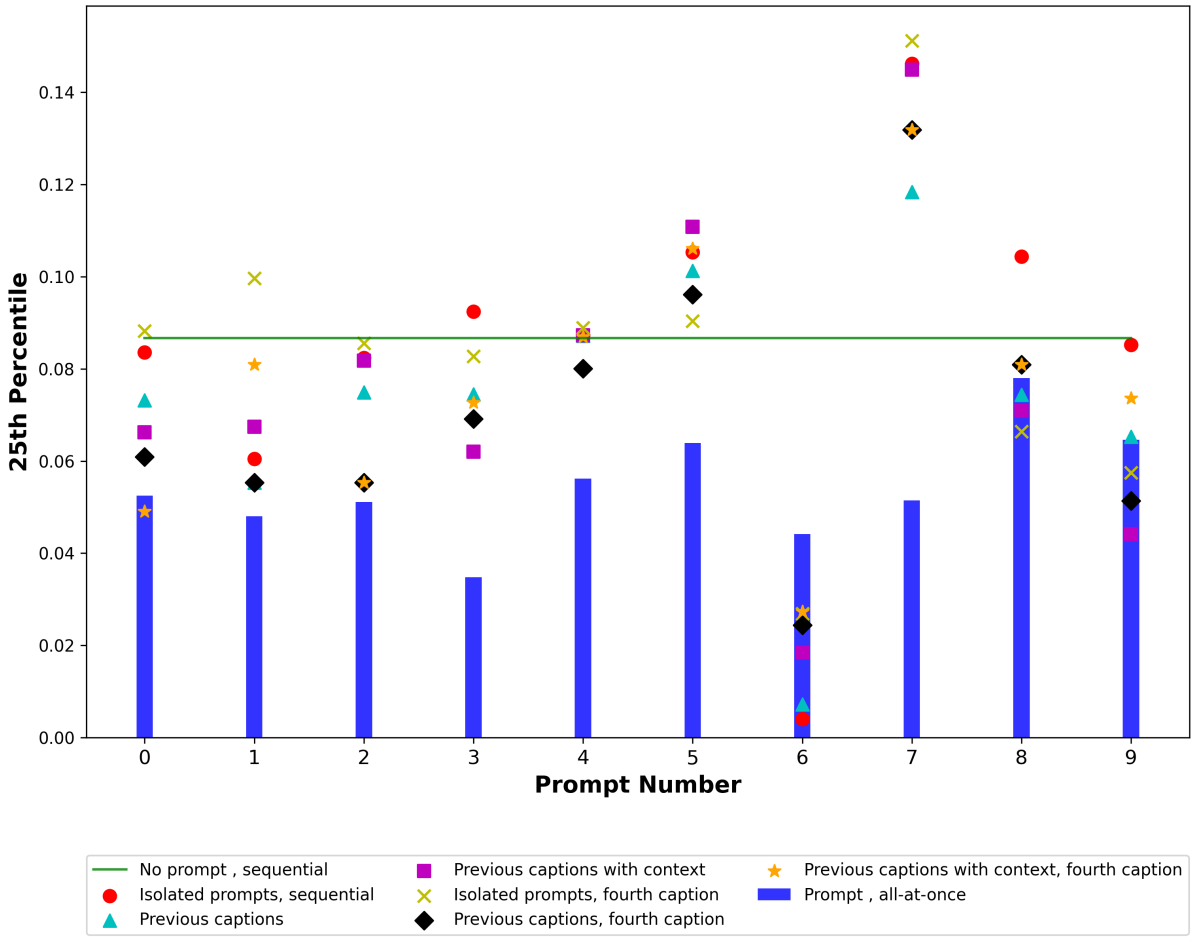


Figure 6.11: 25th percentile sentence similarity score.

There are sequences which the pipeline finds very hard to capture the narrative. Figure 6.11 confirms this assertion, by showing that around the quarter of sequences with worst sentence similarities score there are scores rarely higher than 10%. Prompt 7 continues to be the best for the first quartile, with sequential prompting leading to large

gains (scores around 14%). This is also the only prompt who leads to scores higher than no prompting at all. Additionally, showing the sequence all-at-once renders poor results.

Figure 6.12: An example where the description of the last image is good at predicting the sequence's narrative. True narrative: *"Taking a photograph"*.
Proposed narrative by using prompt 7 with forth caption: *"The man is taking a picture of the woman."*. BLEU: 11%; sent. similarity: 56%. *Image_23* in the dataset.

It is also worth mentioning the fact that the prompt strategy who performs better (less badly) around the first quartile is the last output when the prompt does not embed previous outputs ("isolated prompt, forth caption"). This evidence is odd, as theoretically one would expect this prompt to be the worst: is is the output of the last figure of the sentence, where the prompt is not connected to other outputs. Hence, it is the output of a model where information from all previous 3 images of the sequence is completely ignored! Having this strategy perform better might mean how hard it is to grasp the narrative of the sequence, that it is better to just grasp the idea from one image. Figure 6.12 present one example on this idea, where the idea in the last image is representative of the whole sequence (taking a photo).

7 Discussion and Improvements

The results we find can be summarized as:

- Prompting can produce better results;
- Prompts who include previous captions have similar performance, so previous information does not help the decoding process in this task;
- Summarize all outputs and not just the output from last image produces better results;
- The BLEU score has limited interpretation, as it exhibits low values;
- The sentence similarity score of the narrative descriptions closer to reality is good, but not spectacular.

The approach we shown so far was a zero-shot based. It comprised the pure junction of several already trained models to assess how close to reality was the predicted caption. Moreover, the prompts we defined were based on a manual choice. Therefore, no optimization tasks were employed. We could look at the several prompts and linking strategies as some sort of grid search, but it is clearly not a very clever one. How could we make the machine improve on our approach?

Several ways to improve our captions:

1. Fine-tune the current pipeline to our task: with some elements of the architecture frozen, apply back-propagation to reduce the cross entropy between predicted and true captions.
2. Try to improve the prompting mechanism, to assess how far the current pipeline can be better as narrative predictor by changing the initial embedding space upon which the model starts decoding the information.
3. Change the pipeline, by:
    - Adding some elements or changing the type of deep learning model;
    - Replace some of the hyperparameters or versions of the deep learning models employed.

    In the next sections, we detail each of these options.

7.1 Fine Tuning

Fine-tuning involves training some parameters of the pipeline in order to minimize some loss function related to the goal we have at hands. In our case, the most standard loss function would be the cross entropy, which tries to make the predicted probabilities of the words as close as possible to the words in the true description of the narratives. One would train in a subset of the 60 sequences and assess performance in other subset. In our case, we can highlight two challenges to this approach: BLIP-2 is a model with a lot of parameters, so fine tuning can be costly in terms of computer resources; one alternative is to freeze the majority of the parameters and train just a part of the model (in fact, the idea within BLIP-2 is to do that with the Q-former acting as a trainable "bridge" between the frozen encoder and decoder), but notice that our architecture is more complex than BLIP-2. It has a summarizer on top, so back-propagation needs to compute gradients that take into account the whole pipeline and not just BLIP-2. Additionally, if the type of prompting strategy has prompts depending on previous outputs, there needs to be a type of "unrolling" of the pipeline: gradients would be based not on one BLIP-2 model, but on the succession of 4 BLIP-2 models, which makes it much more complex; and the trainable parameters of the BLIP-2 model would be present in 4 different parts of the pipeline. If one were to use the prompt who embeds previous outputs, one would be computing gradients of the following function:

$$\begin{aligned}\text{SUMM}_\beta\Big(&\text{BLIP}_\theta(\text{image}_0, \text{prompt}_0) \oplus \text{BLIP}_\theta(\text{image}_1, \text{prompt}_1) \oplus \\ &\text{BLIP}_\theta(\text{image}_2, \text{prompt}_2) \oplus \text{BLIP}_\theta(\text{image}_3, \text{prompt}_3)\Big)\end{aligned} \tag{7.1}$$

with:

$$\text{prompt}_i = \text{BLIP}_\theta(\text{image}_{i-1}, \text{prompt}_{i-1}) \oplus \text{prompt}_0, \quad i = 1, 2, 3 \tag{7.2}$$

Here, $\oplus$ denotes concatenation. SUMM is the summarizer model (with parameters β) and BLIP is BLIP-2 model (with parameters θ).

As one can conclude, computing gradients over this complex structure is cumbersome. It involves 4 BLIP-2 functions and one summarizer (SUMM). Even with the majority of the parameters frozen this is not straightforward. Therefore we leave this task as a research question for future investigation.

7.2 Prompt Engineering with Reinforcement Learning

In this section, we present some discussion on future developments that could be made on the prompting of our pipeline. Adjusting the prompts seems to be most direct and straightforward way to try to improve the description of narratives of the pipeline. In fact, prompt engineering has become a relevant field of deep learning investigation, allowing for more efficient task adaption of large deep learning models ([50]).

7.2.1 Reinforcement Learning

The route we suggest starts by framing this problem as a reinforcement learning (RL) one. We suggest presenting briefly what is RL and which are the main ways to create an optimization strategy for an agent to learn to behave to maximize positive returns (see for example [51]). RL is a branch of machine learning concerned with training agents to make sequential decisions in an environment.

A RL model can be characterized by a markov decision process, which is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where:

- $\mathcal{S}$: The state space, representing all possible configurations of the environment;
- $\mathcal{A}$: The action space, representing all possible actions the agent can take;
- $P(s'|s, a)$: The transition probability function, denoting the probability of transitioning to state s' from state s after taking action a;
- $R(s, a)$: The reward function, providing the scalar feedback received by the agent after executing action a in state s;
- $\gamma \in [0, 1)$: The discount factor, determining the importance of future rewards relative to immediate rewards.

The goal of an RL agent is to learn a policy $\pi(a|s)$, which defines the probability of selecting action a in state s, that maximizes the expected cumulative reward over time.

There are several ways to try to find the optimal decision process for the agent. We proceed to highlight from a generic point of view three approaches:

- Value function maximization. The value function $V_\pi(s)$ quantifies the expected return when starting in state s and following policy π. It is defined as:

$$V_\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s \right]. \tag{7.3}$$

The optimal value function, $V^*(s)$, satisfies the Bellman optimality equation:

$$V^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right]. \tag{7.4}$$

- Policy optimization. The policy function $\pi_\theta(a|s)$, parameterized by θ, can be optimized directly by maximizing the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{7.5}$$

Using the policy gradient theorem, the gradient of $J(\theta)$ is:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \log \pi_\theta(a|s) Q_\pi(s,a) \right], \tag{7.6}$$

where $Q_\pi(s,a)$ is the action-value function.

- Actor-critic model. The actor-critic architecture combines policy optimization (actor) and value function estimation (critic). The actor updates the policy $\pi_\theta(a|s)$, while the critic evaluates the policy using an estimate of the value function $V_w(s)$, parameterized by w. The loss functions for the actor and critic are defined as:

$$\mathcal{L}_{\text{critic}}(w) = \mathbb{E}_{\pi_\theta} \left[\left(R(s,a) + \gamma V_w(s') - V_w(s) \right)^2 \right], \tag{7.7}$$

$$\mathcal{L}_{\text{actor}}(\theta) = -\mathbb{E}_{\pi_\theta} \left[\log \pi_\theta(a|s) \left(R(s,a) + \gamma V_w(s') - V_w(s) \right) \right]. \tag{7.8}$$

The parameters θ and w are updated iteratively to optimize their respective objectives, enabling the actor-critic model to learn an effective policy.

7.2.2 RL Application to Prompting

How can we interpret our prompt design as a RL problem? The sentence similarity score is the reward from a pipeline which acts as an exogenous black box. The actions are the prompts given to the pipeline and the state is the previous prompts. We can then adjust an external language model to output prompts which try to improve rewards. This external LLM acts as the policy function.

There are several papers which follow this approach. RLPrompt ([52]) trains an MLP layer (Figure 7.1) added to an LLM in order to get the trainable parameters to reach the better rewards. The main advantages of this approach is that an LLM is directly trained and not just the embedding space (which is usually called soft prompt), which might make it more intuitive for a human to understand prompt (as one looks directly into the best sequence of words and not a theoretical embedding array). However the

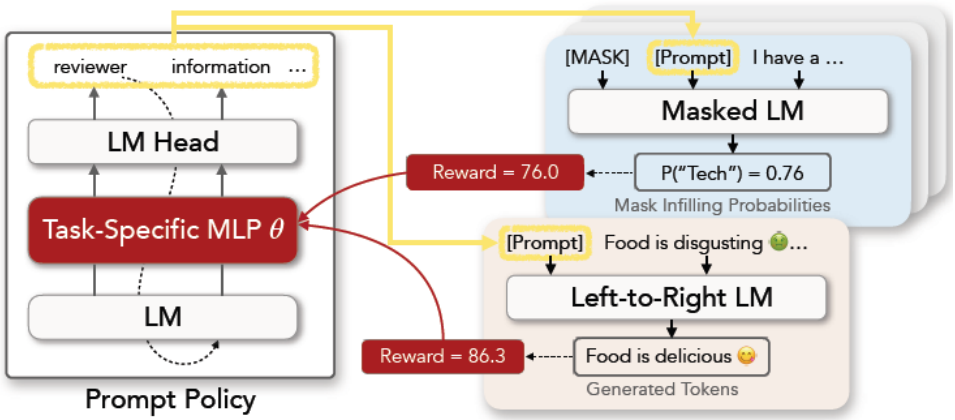


Figure 7.1: The architecture of RLPrompt. Source: [52].

sequence of sentences RLPrompt arrives to is not completely a consistent sentence or questions. PreWrite ([53] aims to circumvent this problem, by training a LLM policy where the whole prompt is trained as a whole, which reduces the likelihood of having a sequence of words with no meaning on its own for a human (Figure 7.2).

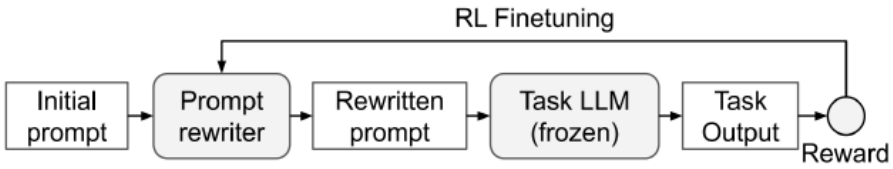


Figure 7.2: The architecture of PreWrite. Source: [53].

There are also several ways to train a RL. RLPrompt uses soft Q-learning ([54]), whereas PreWrite employs proximal policy optimization (PPO, [10]). This last one is the one we would seek to apply in our work. PPO improves on [55] by introducing a clipped objective function, whose goal is to limit the magnitude of parameters changes. Usually RL training can be very erratic, so an algorithm who allows for a large update on parameters might be responding to noise rather than signal. The solution is to force the model to move in smaller steps, and this is achieved by creating an optimization function which is clipped (Figure 7.3). The clipped objective is a simpler way to limit the possible erratic movements of parameters. The previous approach ([55]) acted in a different way: it proposed a penalty in the objective function, consisting of a measure of how current policy is different from previous one. By assessing and adding a penalty on the divergence between the distributions of current and previous policy, TRPO limits the change in parameters within each learning step.

PPO became ubiquitous as a RL tool to allow alignment between LMM outputs and human preferences. RL from human feedback (RLHF, see for example [56]) builds upon this framework. Humans are shown several outputs from an LLM and they rank them based on several dimensions. Then a dataset is built out of the rankings and out-

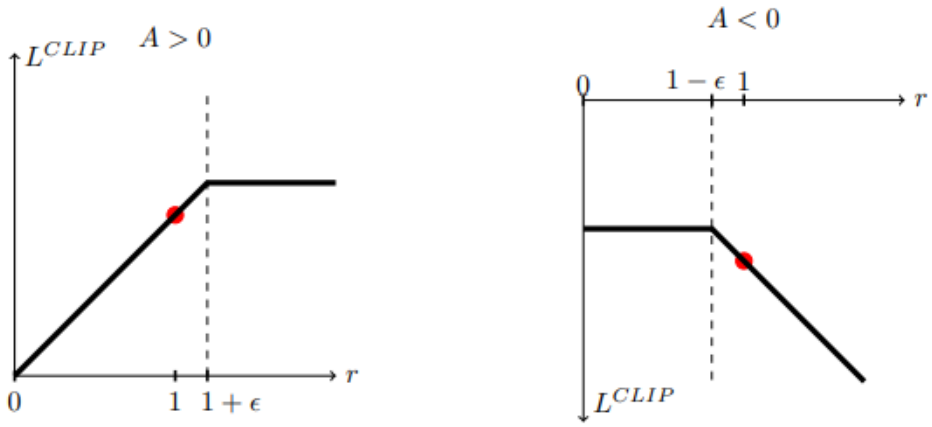


Figure 7.3: The clipped objective function. Source: [10].

puts, and a reward function is trained to mimic human preferences. Once the rewards function has been trained, all elements have been gathered to train a RL framework. Is is relevant to emulate RHLF in our work? RHLF is particularly useful when there is not by definition a reward function who can assemble automatically human preferences. But in our work we already have a sentence similarity metric, which can directly compare how close is the meaning of two narratives (unless we did not trust the metric and preferred to have an human classifying how good the narrative descriptions are).

We finalize this section with our proposal to materialize the use of RL to our work:

- Choose an LLM to act as policy: we would start by GPT-2;
- Define one word to act as start to the decoding process. For example "Describe";
- Generate several prompts from decoding GPT-2 with high temperature (to generate a more diverse set of outputs/prompts to the pipeline);
- Define a set of comics sequences to form the training dataset and other for the test set;
- Use the prompts who output from GPT-2 to input as prompt to one version of our narrative description pipeline;
- For each prompt, store the average sentence similarity score on the training set as the reward;
- With the several prompt-reward pairs, train the GPT-2 in order to minimize the PPO loss. The trainable parameters could comprise one of the last layers of GPT-2. The PPO loss might not need to include the value function; the value function is usually included to ensure additional stability, but in our context we know the rewards are bounded between -1 and 1 (the cosine similarity of the embedding space).

7.3 Changing the Pipeline's Elements

The first option to change the pipeline would be to tweak some of the models who are employed. For example, in BLIP-2, try to use the version where FLAN is the decoder: this is an encoder-decoder type of LLM, so maybe it can be more able to encode the more complex prompts, who embed outputs from previous images. The second option would be to change the summarizer, by trying to use a version specially trained for story and narrative creation.

We finalize with the suggestion of a more extreme change, to the type of deep learning model. The way we suggest to pass through information from previous images to help in describing the narrative is using prompting. However, one could argue that the compression of information might be too large for the pipeline to grasp the whole idea of the narrative. Additionally, even when the prompt does no embed previous captions, compressing the information in a sequence of text in hope for the summarizer to built a narrative out of four individual descriptions of four images might lead to loss of information which is crucial to build a narrative. What solution could one test? One where narrative is built directly from information out of all images. But we have done this already, when we showed the image all at once to BLIP-2. What is missing from this approach? The idea of sequence. How could one mix the sequence and a decoder which embed the information from all images? By using a video-to-text model. The models who caption videos can be able to capture the narrative. In fact, the 4-image sequence might be interpreted as a video with 4 frames.

8 Conclusion

Comics has its own design and language. The majority of deep learning models are trained on "real" images, which might it challenging to do zero-shot description and understanding of comics' scenes. In this work, we sought to contribute to literature on deep learning applied to comics understanding, in several dimensions.

The first one is by the creation of a dataset comprising 60 sequences of 4-image's comics, as well as proposed individual captions and narrative descriptions. Although datasets on comics are abundant, the literature lacks images with no textual references. Hence we are contributing to future research focused on the understanding of comics images, allowing for a "cleaner" inspection of narratives from comics vignettes.

We have assessed the performance of a pipeline to do zero-shot description of the comics' narrative. We define two benchmarks, to disentangle whether the results of our approach come from prompting or sequential inputting the images of the sequence. We also inquiry the role prompting can play in grasping the narrative. We aim to understand whether adding previous images' outputs to prompts can improve to narrative identification. The results are mixed: in some images, prompting without references from other images can improve the narrative identification, but this is not a general pattern.

We conclude that prompting generates value, and the prompting with better performance relates to one where it is asked for what is happening in the scene. Overall, the majority of the predicted narratives show some similarity with the real narratives, but they are still far away from perfect proximity of meaning.

For future investigation, we propose the assessment of video-to-text models, which embed directly the temporal dimension of an image's sequence. Moreover, we propose an optimization strategy to improve prompts by training an external LLM as policy, who outputs the prompts for the image's description pipeline.

References

[1] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. rahman Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Annual Meeting of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:204960716

[2] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. USA: Association for Computational Linguistics, 2002, p. 311–318. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[3] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models," in *Proceedings of the 40th International Conference on Machine Learning*, ser. ICML'23. JMLR.org, 2023.

[4] A. Radford, J. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 02 2021.

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Springer International Publishing, 2014, pp. 740–755.

[6] H. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. dehghani, S. Brahma, A. Webson, S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, and J. Wei, "Scaling instruction-finetuned language models," 10 2022.

[7] J. Wang, Z. Yang, X. Hu, L. Li, K. Lin, Z. Gan, Z. Liu, C. Liu, and L. Wang, "Git: A generative image-to-text transformer for vision and language," *ArXiv*, vol. abs/2205.14100, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:249152323

[8] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[9] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. T. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer, "Opt: Open pre-trained transformer language models," *ArXiv*, vol. abs/2205.01068, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248496292

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:28695052

[11] Y. Sato, K. Mineshima, and K. Ueda, "Visual representation of negation: Real world data analysis on comic image design," 05 2021.

[12] F. Murtagh, "Multilayer perceptrons for classification and regression," *Neurocomputing*, vol. 2, no. 5, pp. 183–197, 1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0925231291900235

[13] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, 2004, pp. II–104 Vol.2.

[14] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2146–2153.

[15] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278919305974

[16] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, pp. 107–116, 04 1998.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[18] K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 06 2014.

[19] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://aclanthology.org/D14-1162

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:52967399

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[22] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 2048–2057.

[23] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, p. 3104–3112.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 10 2020.

[25] J. Laubrock and A. Dunst, "Computational approaches to comics analysis," *Topics in Cognitive Science*, vol. 12, 11 2019.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *ArXiv*, vol. abs/1505.04597, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:3719281

[27] D. Dubray and J. Laubrock, "Deep cnn-based speech balloon detection and segmentation for comic books," 09 2019, pp. 1237–1243.

[28] B. Topal, D. Yuret, and T. Sezgin, "Domain-adaptive self-supervised face body detection in drawings," 08 2023, pp. 1432–1439.

[29] N.-V. Nguyen, C. Rigaud, and J.-C. Burie, "Comic mtl: optimized multi-task learning for comic book image analysis," vol. 22, no. 3, p. 265–284, sep 2019. [Online]. Available: https://doi.org/10.1007/s10032-019-00330-3

[30] H. Li, S. Guo, K. Lyu, X. Yang, T. Chen, J. Zhu, and H. Zeng, "A challenging benchmark of anime style recognition," 06 2022, pp. 4720–4729.

[31] J. Chen, R. Iwasaki, N. Mori, M. Okada, and M. Ueno, "Understanding multilingual four-scene comics with deep learning methods," in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, vol. 1, 2019, pp. 32–37.

[32] M. Iyyer, V. Manjunatha, A. Guha, Y. Vyas, J. Boyd-Graber, H. Daume, and L. Davis, "The amazing mysteries of the gutter: Drawing inferences between pan-

els in comic book narratives," 07 2017, pp. 6478–6487.

[33] Y.-C. Chen and A. Jhala, "A computational model of comprehension in manga style visual narratives," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 43, no. 43, 2021.

[34] I. Manouach, T. Melistas, Y. Siglidis, and F. Kalogiannis, "A deep learning pipeline for the synthesis of graphic novels," in *International Conference of Computational Creativity (ICCC 21)*, 2021.

[35] B. Proven-Bessel, Z. Zhao, and L. Chen, "Comicgan: Text-to-comic generative adversarial network," 09 2021.

[36] H. Agrawal, A. Mishra, M. Gupta, and Mausam, "Multimodal persona based generation of comic dialogs," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 14 150–14 164. [Online]. Available: https://aclanthology.org/2023.acl-long.791

[37] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3156–3164.

[38] J. Jiayi, Y. Luo, X. Sun, F. Chen, G. Luo, W. Yongjian, Y. Gao, and R. Ji, "Improving image captioning by leveraging intra- and inter-layer global representation in transformer network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 1655–1663, 05 2021.

[39] Y. Wang, J. Xu, and Y. Sun, "End-to-end transformer based model for image captioning," in *AAAI Conference on Artificial Intelligence*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247778466

[40] R. Mokady and A. Hertz, "Clipcap: Clip prefix for image captioning," *ArXiv*, vol. abs/2111.09734, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:244346239

[41] N. Rotstein, D. Bensaid, S. Brody, R. Ganz, and R. Kimmel, "Fusecap: Leveraging large language models for enriched fused image captions," *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5677–5688, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258960254

[42] S. Bianco, L. Celona, M. Donzella, and P. Napoletano, "Improving image captioning descriptiveness by ranking and llm-based fusion," 06 2023.

[43] J. Li, D. Li, C. Xiong, and S. C. H. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation,"

in *International Conference on Machine Learning*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246411402

[44] L. Yuan, D. Chen, Y.-L. Chen, N. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li, C. Liu, M. Liu, Z. Liu, Y. Lu, Y. Shi, L. Wang, J. Wang, B. Xiao, Z. Xiao, and P. Zhang, "Florence: A new foundation model for computer vision," 11 2021.

[45] B. Gliwa, I. Mochol, M. Biesek, and A. Wawer, "Samsum corpus: A human-annotated dialogue dataset for abstractive summarization," 11 2019.

[46] A. Lavie and A. Agarwal, "Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments," in *Proceedings of the Second Workshop on Statistical Machine Translation*, ser. StatMT '07. USA: Association for Computational Linguistics, 2007, p. 228–231.

[47] R. Vedantam, C. L. Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4566–4575.

[48] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 382–398.

[49] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "Minilm: deep self-attention distillation for task-agnostic compression of pre-trained transformers," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[50] P. Sahoo, S. Saha, V. Jain, S. Mondal, and A. Chadha, "A systematic survey of prompt engineering in large language models: Techniques and applications," 02 2024.

[51] M. Naeem, S. Rizvi, and A. Coronato, "A gentle introduction to reinforcement learning and its application in different fields," *IEEE Access*, vol. 8, pp. 209 320–209 344, 01 2020.

[52] M. Deng, J. Wang, C.-P. Hsieh, Y. Wang, H. Guo, T. Shu, M. Song, E. Xing, and Z. Hu, "RLPrompt: Optimizing discrete text prompts with reinforcement learning," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 3369–3391. [Online]. Available: https://aclanthology.org/2022.emnlp-main.222

[53] W. Kong, S. Hombaiah, M. Zhang, Q. Mei, and M. Bendersky, "PRewrite: Prompt rewriting with reinforcement learning," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*,

L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 594–601. [Online]. Available: https://aclanthology.org/2024.acl-short.54

[54] H. Guo, Z. Liu, E. Xing, and Z. Hu, "Text generation with efficient (soft) q-learning," 06 2021.

[55] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1889–1897. [Online]. Available: https://proceedings.mlr.press/v37/schulman15.html

[56] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. E. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. J. Lowe, "Training language models to follow instructions with human feedback," *ArXiv*, vol. abs/2203.02155, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246426909

Annexes

Annex A - Additional Results

Table A1: Sentence similarity score for showing the images sequentially, with isolated prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.08	0.24	0.23	0.35
1	0.06	0.23	0.17	0.38
2	0.08	0.24	0.24	0.35
3	0.09	0.27	0.23	0.42
4	0.09	0.25	0.22	0.39
5	0.11	0.28	0.28	0.40
6	0.00	0.06	0.04	0.08
7	0.15	0.31	0.29	0.49
8	0.10	0.24	0.17	0.36
9	0.09	0.20	0.15	0.28

Table A2: Sentence similarity score for showing the images sequentially, while adding previous prompts to prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.07	0.24	0.22	0.40
1	0.06	0.27	0.27	0.43
2	0.07	0.27	0.27	0.41
3	0.07	0.23	0.16	0.36
4	0.09	0.22	0.17	0.31
5	0.10	0.24	0.19	0.35
6	0.01	0.06	0.05	0.10
7	0.12	0.30	0.27	0.47
8	0.07	0.20	0.13	0.33
9	0.07	0.18	0.13	0.20

Table A3: Sentence similarity score for showing the images sequentially, while adding previous prompts and some connectors to prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.07	0.23	0.19	0.35
1	0.07	0.27	0.28	0.42
2	0.08	0.26	0.27	0.37
3	0.06	0.22	0.17	0.36
4	0.09	0.22	0.16	0.35
5	0.11	0.25	0.21	0.35
6	0.02	0.08	0.06	0.13
7	0.14	0.31	0.30	0.46
8	0.07	0.21	0.15	0.33
9	0.04	0.18	0.12	0.25

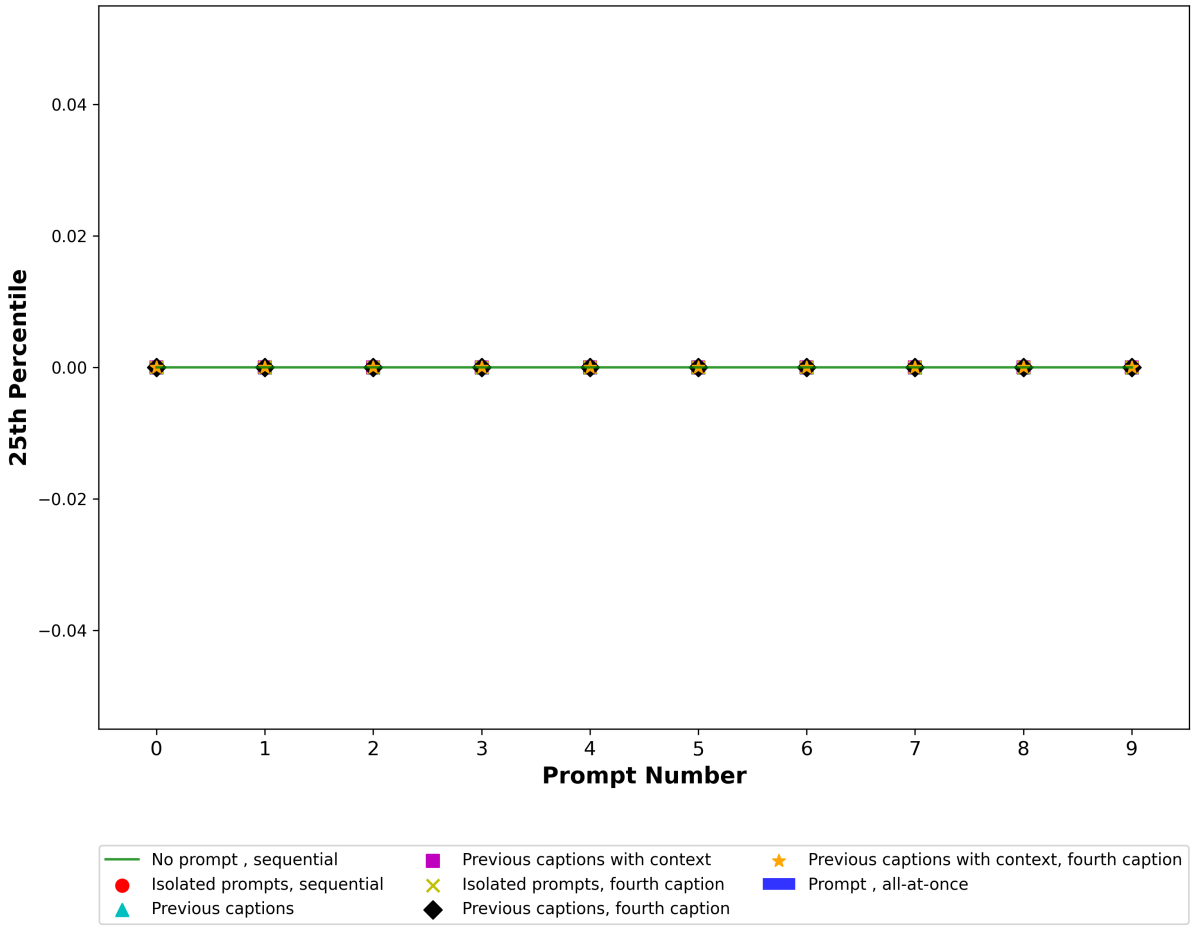


Figure A1: 25th percentile's BLEU score.

Table A4: BLEU score for showing the images sequentially, with isolated prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.00	0.08	0.04	0.11
1	0.00	0.09	0.09	0.15
2	0.00	0.07	0.04	0.10
3	0.00	0.07	0.00	0.11
4	0.00	0.07	0.00	0.11
5	0.00	0.10	0.10	0.13
6	0.00	0.01	0.00	0.00
7	0.00	0.09	0.09	0.12
8	0.00	0.05	0.00	0.10
9	0.00	0.05	0.00	0.08

Table A5: BLEU score for showing the images sequentially, while adding previous prompts to prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.00	0.06	0.00	0.11
1	0.00	0.09	0.04	0.18
2	0.00	0.07	0.00	0.12
3	0.00	0.05	0.00	0.10
4	0.00	0.07	0.00	0.13
5	0.00	0.08	0.00	0.12
6	0.00	0.03	0.00	0.07
7	0.00	0.08	0.00	0.12
8	0.00	0.05	0.00	0.11
9	0.00	0.04	0.00	0.09

Table A6: BLEU score for showing the images sequentially, while adding previous prompts and some connectors to prompting.

Prompt	25th Percentile	Mean	Median	75th Percentile
0	0.00	0.06	0.00	0.11
1	0.00	0.09	0.04	0.15
2	0.00	0.08	0.00	0.11
3	0.00	0.07	0.00	0.11
4	0.00	0.06	0.00	0.12
5	0.00	0.08	0.09	0.12
6	0.00	0.08	0.09	0.11
7	0.00	0.08	0.04	0.12
8	0.00	0.05	0.00	0.11
9	0.00	0.07	0.00	0.11

Annex B - Dataset's Sources

Name	Source
*image*_0	https://i.pinimg.com/originals/87/eb/59/87eb594c959d1f71e5cf8c02c28477f6.gif
*image*_1	https://www.mezzacotta.net/garfield/?comic=2658
*image*_2	https://www.gocomics.com/calvinandhobbes/2024/03/31
*image*_3	https://www.gocomics.com/peanuts/2024/02/25
*image*_4	https://www.gocomics.com/bignate/2012/08/05
*image*_5	https://www.gocomics.com/calvinandhobbes/2016/06/08
*image*_6	https://www.gocomics.com/adult-children/2022/09/23
*image*_7	https://www.gocomics.com/the-adventures-of-business-cat/2019/02/21
*image*_8	https://www.gocomics.com/arloandjanis/2020/10/09
*image*_9	https://www.gocomics.com/arloandjanis/2014/08/11
*image*_10	https://www.gocomics.com/arloandjanis/2014/01/03
*image*_11	https://www.gocomics.com/arloandjanis/2001/07/02
*image*_12	https://www.gocomics.com/arloandjanis/2018/08/29
*image*_13	https://www.gocomics.com/arloandjanis/2002/01/16
*image*_14	https://www.gocomics.com/calvinandhobbes/2024/03/29
*image*_15	https://www.gocomics.com/peanuts/1960/05/03
*image*_16	https://www.gocomics.com/banana-triangle/2015/05/13
*image*_17	https://www.gocomics.com/thebarn/2014/05/10
*image*_18	https://www.gocomics.com/beardo/2017/03/07
*image*_19	https://www.gocomics.com/ben/2017/01/02
*image*_20	https://www.gocomics.com/ben/2014/10/20
*image*_21	https://www.gocomics.com/thebarn/2009/11/04
*image*_22	https://www.gocomics.com/barneyandclyde/2017/06/25
*image*_23	https://www.gocomics.com/barneyandclyde/2013/05/27
*image*_24	https://www.gocomics.com/ben/2022/04/08
*image*_25	https://www.gocomics.com/ben/2024/02/21
*image*_26	https://www.gocomics.com/ben/2016/06/01
*image*_27	https://www.gocomics.com/ben/2019/03/02
*image*_28	https://www.gocomics.com/ben/2014/01/10
*image*_29	https://www.gocomics.com/ben/2023/03/20

Name	Source
*image*_30	https://www.gocomics.com/beardo/2023/06/30
*image*_31	https://www.gocomics.com/beardo/2016/12/11
*image*_32	https://www.gocomics.com/beardo/2013/10/30
*image*_33	https://www.gocomics.com/betty/1999/03/21
*image*_34	https://www.gocomics.com/betty/2021/01/10
*image*_35	https://www.gocomics.com/betty/2011/10/30
*image*_36	https://www.gocomics.com/betty/2003/06/22
*image*_37	https://www.gocomics.com/betty/2009/03/15
*image*_38	https://www.gocomics.com/betty/2021/04/25
*image*_39	https://www.gocomics.com/betty/2021/04/25
*image*_40	https://www.gocomics.com/betty/2016/12/25
*image*_41	https://www.gocomics.com/betty/2019/08/04
*image*_42	https://www.gocomics.com/betty/1998/12/13
*image*_43	https://www.gocomics.com/betty/2002/05/19
*image*_44	https://www.gocomics.com/betty/1996/11/03
*image*_45	https://www.gocomics.com/betty/2005/05/22
*image*_46	https://www.gocomics.com/betty/1999/03/07
*image*_47	https://www.gocomics.com/betty/2020/09/20
*image*_48	https://www.gocomics.com/buni/2018/08/31
*image*_49	https://www.gocomics.com/calvinandhobbesespanol/2014/08/30
*image*_50	https://www.gocomics.com/familytree/2010/05/14
*image*_51	https://www.gocomics.com/familytree/2011/04/01
*image*_52	https://www.gocomics.com/meaningoflila/2020/04/10
*image*_53	https://www.gocomics.com/next-door-neighbors/2024/01/03
*image*_54	https://www.gocomics.com/pickles/2024/06/11
*image*_55	https://www.gocomics.com/pickles/2023/07/19
*image*_56	https://www.gocomics.com/familytree/2014/01/05
*image*_57	https://www.gocomics.com/familytree/2018/12/09
*image*_58	https://www.gocomics.com/familytree/2021/06/04
*image*_59	https://www.gocomics.com/pickles/2015/09/16

Instituto Superior de Engenharia
Politécnico de Coimbra