Universiteit Utrecht

From Image Captioning to Visual Storytelling

MSc Thesis

Author: *Admitos Rafail Passadakis*
Email: *r.a.passadakis@students.uu.nl*

Daily Supervisor: *Yingjin Song MSc*

Project Supervisor: *Dr. Albert Gatt*

Second examiner: *Dr. Denis Paperno*

MSc of *Artificial Intelligence*
Department of *Information and Computing Sciences*
Faculty of *Science*
Utrecht University

Utrecht, The Netherlands

November 2024

Acknowledgments

First of all, I am very thankful to my thesis project supervisor, Dr. Albert Gatt, for giving me the chance to work on this project and all his helpful advice and the detailed feedback when needed. I would also like to thank my second examiner, Dr. Denis Paperno and my daily supervisor, Yingjin Song, for her continuous support and guidance during the whole process. Furthermore, a great thank you is owned to my close friends, both in Greece and the Netherlands, who also aided me majorly by (literally) taking part on my project as evaluators. Without them, the obtained results wouldn't have been possible. Last but not least, I could not forget my family who supported me practically and mentally, throughout my Master's, and in their absence nothing of my today's accomplishments would have been achieved.

Abstract

This thesis explores the intersection of two key tasks in Vision & Language: Image Captioning and Visual Storytelling. While traditionally treated as separate problems, we investigate whether these two tasks can be combined into a single framework by viewing image captioning as a subset of visual storytelling. Our approach is threefold. First, we employ and fine-tune a transformer-based model, Clip-Cap, to generate individual captions for a sequence of images originating from VIST dataset. Then, these captions are transformed into coherent narratives using text-to-text Encoder-Decoder models, such as T5 or BART. The aim is to generate stories that capture the dynamic relationships between visual elements in the input image sequence. In the end, we unify our previous steps under an end-to-end architecture which will be capable of producing cohesive storylines given a sequence of correlated images in its input. The evaluation of the generated stories is conducted through multiple methods: 1) with automatic language metrics, 2) with human judgment for a more nuanced assessment, and 3) with GPT-4's artificial judge for comparisons against human annotators. Our results show that integrating image captioning and storytelling under our novel framework, has a positive impact on the quality of the generated stories. In fact, in many cases, our outcome surpasses even the human-level of written narratives, a conclusion supported by all evaluation methods employed. Consequently, the present work could contribute to the ongoing research in generative AI, particularly in bridging the gap between textual description and narrative coherence in multi-image sequences which are slightly correlated.

Contents

1 Introduction

1.1 Background

In the unfolding landscape of contemporary research, the intersection of Computer Vision (CV) and Natural Language Processing (NLP) has ignited a profound interest in the art of generating textual narratives from images and videos. This interdisciplinary pursuit has given rise to a plethora of consequential tasks in both of these fields such as image labeling, image and video description (or captioning) and visual question answering. Before the emergence of tasks that flow out from both of these fields, CV and NLP have to demonstrate substantial achievements over the last years individually. In particular, in the domain of CV, prominent results have been achieved in image description and classification with various deep neural network architectures such as the Convolutional Neural Networks (CNNs) [42, 73, 117, 123]. Simultaneously, on the NLP side, several tasks such as machine translation, summarization or text generation have become easier with new and pioneering models like Encoder-Decoder [17, 122] and most recently with Transformer architectures [11, 21, 105, 107, 130].

Yet, the urgency to forge more seamless connections between these twin pillars of Deep Learning has become more pronounced than ever before. To that end, the need of generating more narrative texts from images which will reflect temporal and sequential coherence, rather than just listing objects and their attributes or generating plain text without any incentive, has given rise to some novel challenges such as *Visual Storytelling* [54]. Herein, the narrative unfolds beyond a superficial representation, delving deep into the realms of experiences and feelings and providing a canvas for a profound exploration of a symbiosis between visual and textual narratives.

1.2 Introduction to Visual Storytelling

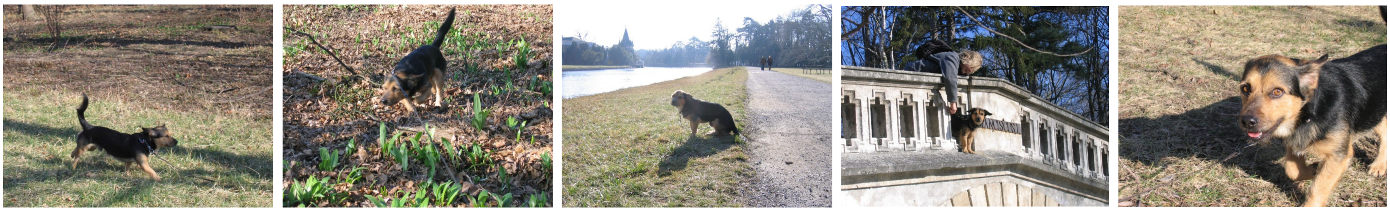
Visual storytelling [54], as an evolving field at the intersection of Computer Vision and Natural Language Processing, aspires to imbue machines with the ability to go beyond mere descriptive captions. Aiming to bridge the semantic gap between visual data and textual comprehension, this task transcends traditional image descriptions by delving into the nuanced and expressional realms of creating cohesive stories. It is usually considered as the descendant of the traditional image captioning, but unlike the latter, where the focus is often on simply detailing objects and their attributes, visual storytelling incorporates a narrative fashion by introducing coherence and a sense of temporal progression in the generated output.

For this precise reason we can trace the origins of visual storytelling back to the merging of image captioning and sequential image processing, where the goal now is not only to describe visual elements but to thread them together into a meaningful narrative. As technology evolves, so does the demand for multimedia storytelling, interactive interfaces, and immersive experiences. This entails the development of models capable of understanding the sequential relationships between images and crafting engaging and contextualized narratives that unfold over a series of visual inputs. Therefore, not only the interpretability of Artificial Intelligence (AI) systems is enriched but also doors to applications in areas such as content creation, multimedia understanding, and human-computer interaction are opened, fostering at the same time a deeper connection between AI and human expression [1].

A paradigm of a simple visual story that unfolds over five correlated images (a common feature is the dog) is shown on Fig. 1. The story is given in the second line of text. Along with it, five isolated captions for these images are also provided in the first line of textual descriptions of the image. It is obvious that storytelling engages emotions and situations that the protagonists of the image are part of, instead of simple depictions of those.

1.3 Motivation

In this landscape, our research embarks on the challenging task of Visual Storytelling. We aim to navigate in the intersection of CV and NLP, leveraging the foundational achievements in both of these fields. Our focus extends beyond individual language or vision tasks, striving to build a framework that will deal with the complicated quest of Visual Storytelling. However, unlike most of the current work on this field, which deals with Visual Storytelling as an undivided task, we will attempt to split it into two separate levels. This means that we will explore if its meaningful to work on Visual Storytelling by considering it as a super-set of Image captioning. But how exactly could this be achieved?



A small dog is running through the grass.

A small dog is walking through the leaves.

A dog is standing on the side of a road.

A woman and her dog on the edge of a bridge

A dog standing in the grass with it's tongue out

The black dog was ready to leave.

He had a great time on the hike.

And was very happy to be in the field.

His mom was so proud of him.

It was a beautiful day for him.

Figure 1. *An example of a sequential vision-to-language story, given also the isolated descriptions. The first line of captions, shows some isolated descriptions for these photos, whilst in the second line presents a story unfolded in five respective sentences.*

Taking as an example the visualised story of Fig. 1, someone could easily argue that the backbone of the unfolding story is given in the caption of the images. For instance, in the first image we can see that the caption describes a ”a dog in the grass” while the first sentence of the story talks about ”a walking dog”. Similarly, on the third picture the captions informs us that ”the dog in on the side of the road”, while the respective story sentence tells us that ”a dog is in the field”. More profoundly, the fourth caption tell us about ”the existence of a woman (with the dog)”, whilst the corresponding part of the story names her ”the dog's mom”. From these, it's evident that the captions provide, at least to some extent, the main information about what could be going on, on such a story sequence.

To that end, we intend to split the Visual Storytelling task in two levels. In the first level, we want to keep the simplicity of image captioning that will give us the core of what is happening in a series of images. Besides, we know already that image captioning is a task in which significant achievements have been made and over the last years major researches have showcased that we have the tools to generate appropriate descriptions for a variety of images. Furthermore, with the advancement of Deep Learning in NLP, many tasks including *text* or *story generation* have been simplified. For that purpose, our goal in the second level is to discover a suitable architecture that will transform these descriptions, blank of narrative meaning, to a sequential narrative story which will both reflect to visual context but also to temporal actions. In the ultimate phase of this work, we intend to merge these two levels to one and use a pipeline of selected models for narrative and coherent visual storytelling.

Through this exploration, we aim to contribute to the evolving narrative AI systems that not only observe and understand but also recount articulate storylines, close to human storytelling and thus bridging the semantic gap between vision and language.

1.4 Objective and Research Questions

As previously mentioned, despite the widespread belief that Visual Storytelling and Image Captioning are two distinct tasks, the objective of this Thesis project is to explore if we can work on these two simultaneously by considering image captioning as a total subset of visual storytelling.

To do so, we intend firstly to use a well known transformer architecture, named Clip-Cap [96] to produce isolated captions for images sequentially correlated (i.e. the images will have a degree of correlation like the ones shown on Fig. 1). Following this, we deploy some extra transformer architectures, one self-made from-scratch and two well-known language models which are BART [75] and T5 [107]. This time the nature of the task is text-to-text generation, which means that our models should reformulate the bare captions (produced previously) to a more narrative and coherent story which will express the dynamic interplay between visual elements. Ultimately, we are going to evaluate the generated stories compared to the original ones coming from the VIST dataset [54]. This evaluation contains comparison in terms of some well-known automatic metrics for image captioning like BLEU [99], METEOR [9], ROUGE-L [81], SPICE [5], CIDEr [131] and SPIDEr [86]. However, lately a lot of criticism has been raised around how credible and efficient are these automatic metrics on evaluating machine-generated descriptions [51, 137, 138]. For this reason, we aim to make our evaluation procedure more robust by using human evaluation and judge on the generated results.

We can now summarize our described objective more formally in the following research question:

RQ. *Can we use a transformer based framework where we firstly generate isolated captions for images and then we reformulate them to extract a cohesive and narrative story?*

Somebody would plausibly argue that this procedure can be divided into smaller sub-parts. Firstly, we need a sufficiently well trained captioning model such as ClipCap to give us the mere descriptions. At the same time, we fine-tune text-to-text architectures, utilizing a large amount of ground truth data, on story generation. Lastly, we can combine the best possible models from the previous two steps, to construct a framework that will be able to produce coherent stories by interpolating caption generation in between. Therefore, we have the following three derived sub-research questions where the first two serve as vestibule to the third:

Sub-RQ 1. *How accurate is Clip-Cap in image captioning on VIST dataset?*

Sub-RQ 2. *Can we create from-scratch text-to-text architectures or fine-tune sufficiently language models like T5 or BART, in order to make them able to reformulate plain captions to a meaningful narration?*

Sub-RQ 3. *In inference time, can we combine the text-to-text transformer-based models with our captioning system to efficiently produce narrative storylines?*

1.5 Structure of the Work

In this part, we present the structure of our exploration steps through several key sections. A focused Literature Review, is given in Section 2, where we provide general insights about Vision and Language tasks and the models. Our elaboration mainly attempts to focus around the task of Visual Storytelling, but also around other relative domains such as Image Captioning. Additionally, we give a brief outline of the relevant work on the text generation task, that we want succeed. Following this, Section 3 details our Methods and gives all the necessary background about the data that will be used throughout this work. Most importantly, this section offers a deep dive on the framework that will be applied, highlighting it's theoretical foundations and describing it's different ways of operation during the training and generation phase. There is also where we introduce our evaluation metrics and procedures.

Section 4 gives a general outline of the structure of our exterminations and underscores the technicalities behind them, such as the hyper-parameters of the models utilized. Then, Section 5 encompasses all the results of this work. In particular, we divide our results section to three main parts, where each of them essentially deals with one sub-research question. Therefore, firstly we present our findings after applying Clip-Cap on VIST for image captioning. Afterwards, we deploy a series of fine-tuned but also from-scratch constructed language models, to conduct reformulation on the mere descriptions of VIST in order to produce cohesive storylines. Finally, we combine our best performing architectures from our previous experiments, to construct an end-to-end framework that will actualize visual storytelling in two separate levels. In all of our phases we evaluate our results using the appropriate automatics metrics, whereas in the last part we recruit human evaluation as well. In the end, at Section 6 we discuss the aftermath of our investigation, returning to our initial research questions which we address them individually. There, we also include the limitations of our approach and some possible expansions of the present work.

2 Literature Review

As previously mentioned over the last decade major advancements have been done in the both the fields of Computer Vision (CV) and Natural Language Processing (NLP). The confluence of these two has brought birth to many new tasks such as Visual Question Answering (VQA) [7], Vision-Language Navigation (VLN) [6, 13, 59][1], Image Captioning [15], Video Captioning [132] and of course Visual Storytelling [54]. At the same time, these tasks necessitated the development of models which will be able to operate in both Vision and Language (V&L) domains.

V&L models are designed for dealing with these tasks and their architecture is inspired by the Encoder-Decoder [122] structure originally proposed for Machine Translation. Therefore, in their adaption to V&L tasks, these models consist of a visual encoder (instead of textual), a text encoder and sometimes, a cross-modal interaction module which maps the visual features to semantic embeddings. In the following sections, we will dive into the most significant V&L Tasks, explore some V&L Models which were applied in these tasks and in subsection 2.3 we will give the latest progressions in the field of V&L, exploring the state-of-the-art task of Visual Storytelling. Finally, in subsection 2.4, we will become acquainted with the technique of generating text (story) given some other text as input (caption).

2.1 Vision & Language Tasks

Beginning our journey, we will first delve into the landscape of V&L tasks, exploring their diversity and significance in bridging the gap between the two principal pillars of Deep Learning namely, Computer Vision and Natural Language Processing.

2.1.1 Visual Question Answering

The task of Visual Question Answering (VQA) is to provide the answer given an image and a related question. In VQA, an algorithm is presented with an image, and users pose questions related to the content of that image expecting by the model to provide answers. The VQA task requires a robust understanding of both image and language representations, making it a challenging problem in the intersection of computer vision and natural language understanding. Various methods have been introduced [31, 32, 61] working on a plethora of datasets such as VQA dataset [7], Visual Genome [72] and others [94, 128].

Some of the techniques leverage deep learning architectures, including Convolutional Neural Networks (CNNs) [73] for image processing and Recurrent Neural Networks (RNNs) [113] for processing textual information. On top of this, an Attention Mechanism [8] can be used to enhance the models performance both in comprehension and generation [56, 145]. Lately, Transformer models, like LXMERT [124] or UNITER [16] have been deployed boosting even further the machine understanding over visual and textual forms, achieving new state-of-the-art (sota) results on various benchmarks of VQA. The ultimate goal is to develop architectures that can perform reasoning and inference across different modalities, demonstrating a broader comprehension of visual scenes and matching the corresponding human ability of understanding.

2.1.2 Image Captioning

Image Captioning, is closely related to VQA and it aims at generating a natural language description of an image. Open-domain captioning is a very challenging task, as it requires a fine-grained understanding of the global and the local entities in an image, as well as their attributes and relationships. The objective is to equip machines with the ability to understand visual content and express it, in a human-like manner.

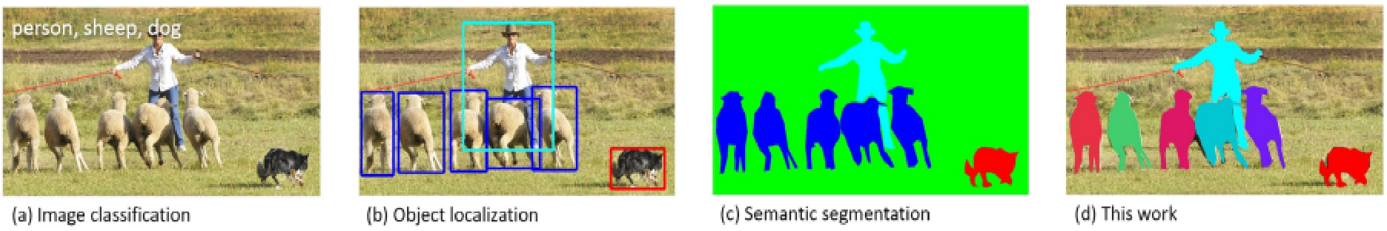
The origins of image captioning can be traced back to the pre-deep learning era when conventional methods based on manual engineering of features and on linguistic rules attempted to generate captions [27]. Some later approaches to image captioning relied on handcrafted features in conjunction with traditional

[1] A clarification need to be addressed here: Vision and Language tasks can be either Vision to Language or Language to Vision. The present work engages with the first category and since VLN is not clearly a Vision to Language task but it can also be a Language to Vision task, we are not going to analyze it here.

machine learning (ML) algorithms [22, 69]. Others, produced image descriptions by using visual dependency representations that capture existing relationships between image objects [25]. However, the landscape was totally transformed with the advent of deep learning, especially with CNNs [73] for image feature extraction and with RNNs [113] for sequential text generation. Once again, the rise of Transformers altered even more the scene on Image Captioning tasks over the last few years.

A very significant point for the evolution of Image Captioning but also for other V&L tasks, is considered to be the introduction of the Microsoft COCO (MSCOCO) dataset in 2014 [82], which provided a diverse and large-scale collection of images with associated captions, fostering a great variety of standardized evaluation metrics and enabling fair comparisons between models. It was one of the first datasets that were not directly intended for image classification or recognition but was constructed with the goal of broader context of scene understanding.

Fig. 2 showcases exactly this, by highlighting the differences between three well-established vision tasks and the case of *Segmenting individual instances of objects*, which is a novel task that MSCOCO introduces. Till today, numerous researches have worked on MSCOCO dataset and many advancements on all the abovementioned V&L tasks have been accomplished on this benchmark [6, 15, 110].



(a) Image classification

(b) Object localization

(c) Semantic segmentation

(d) This work

Figure 2. *Differentiations between Vision tasks: (a) image classification, (b) object bounding box localization, (c) semantic pixel-level segmentation and (d) segmenting individual object instances, on which MSCOCO focuses on. [82]*

However, image captioning was actualized in practice in the paper of *Vinyals et al.'s* [135]. In this work, the authors apply for the first time deep recurrent architectures for Image Captioning. The model comprises of a Convolutional Neural Network (CNN) to encode the image and a Long Short-term Memory (LSTM) [46] network to generate descriptive captions. Without a doubt, the foresaid endeavor was pivotal in popularizing the usage of neural networks for image captioning and for it's time it demonstrated state-of-the-art performance on the MSCOCO dataset.

This marked a departure from the utilization of handcrafted features, to the usage of neural models for image captioning. Similar kind of architecture was deployed by *Karpathy et al.'s* in [63] (this time they used a bidirectional RNN as a decoder), where they successfully generated image descriptions on images from several datasets such as Flickr8K [47], Flickr30K [150] and MSCOCO. A combination of a CNN-LSTM approach was used once more in [23] for producing accurate visual descriptions and in [62] where the authors introduced the dense captioning task, a generalization of object detection and image captioning.

The incorporation of attention mechanisms into image captioning models enhanced their capability to focus on relevant regions of an image while generating captions. Models like *Xu et al.'s* [142] demonstrated improved performance by learning a latent alignment from scratch when generating corresponding words. Later, *You et al.* [149] utilized high-level concepts and injected them into a neural-based approach as semantic attention to enhance image captioning and *Lu et al.*, [91] introduced an adaptive attention mechanism with a visual sentinel to determine when to generate and where to attend during captioning.

Lastly, *Anderson et al.* [6] proposes a model that combines bottom-up and top-down attention mechanisms that enables it to focus on prominent objects of the image and other salient regions. In this work, the authors use a Faster R-CNN to extract the most substantial image features (bottom-up network) and then they feed them to a series of Attention-LSTMs (top-down model) that generate language (captions). A more concrete illustration from the extraction of crucial features by the CNN and how these are fed to the generator model can be shown on Fig. 3

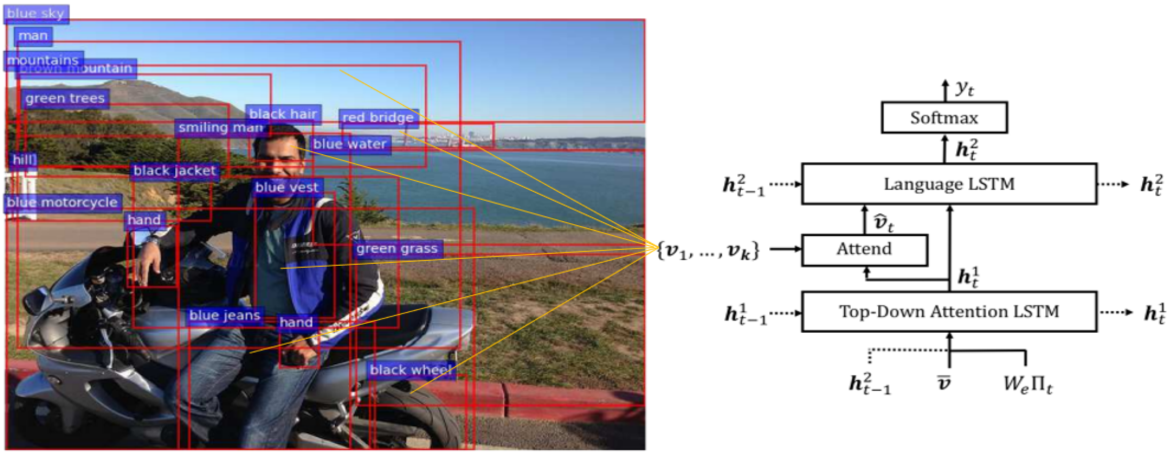


Figure 3. *Example output of how the Faster R-CNN bottom-up model feeds the top-down caption generator. Each bounding box is labeled with an attribute class and two LSTM layers are used to selectively attend these spatial image features [6]*

Making one step further from the attention mechanism, there are several outstanding works on image captioning. One of them is [111], where *Rennie et al.* used reinforcement learning to optimize image captioning systems on MSCOCO. More specifically, a new system was built with an optimization approach that is called self-critical sequence training (SCST) which is a form of the REINFORCE algorithm [140]. Another one, is *Yao et al.* [148], who developed a model which consists of a semantic and a spatial scene graph with the purpose to detect objects in the image, based on their spatial and semantic connections. The important features of each node in the scene graph are refined by leveraging Graph Convolutional Networks (GCN) and then are feed into the attention-LSTM decoder for sentence generation.

Similarly, *Yang et al.* [144] proposed a Scene Graph Auto-Encoder (SGAE) that incorporates the language inductive bias into the encoder-decoder image captioning framework. In a nutshell, the authors encode the graph structure of the sentence to learn a dictionary and then the semantic scene graph is encoded using the learnt dictionary. The last two approaches (*Yao et al.*,*Yang et al.*) are considered to be graph-based methods for image captioning. Transitioning to the Transformers era, the first attempt to adapt these models on image captioning was made by *Huang et al.* [53], when they introduced the Attention-on-Attention Network (AoANet) which extends the traditional attention mechanisms to determine the relevance between attention results and queries. Using *gated linear layers* [20], AoA generates an *information vector* and an *attention gate* using the simple attention result and the query and then applies element-wise multiplication to obtain the *attended information*.

Another interesting result described by *Herdade et al.* [44], is that the simple positional encoding (as proposed in the original transformer [130]), did not improve image captioning performance and thus a 2D encoding of position and size between detected objects is necessitated. Moreover, the entangled transformer [76] features a dual parallel transformer that encodes and refines visual and semantic information in the image, which is fused through a gated bilateral controller and eventually solves the semantic gap between vision and language that attention mechanisms and RNNs highly possess. *He et al.* [43], aimed to combine Transformers and graph-based methods by employing the spatial relationship detected between different but adjacent regions of the image. In their proposed model, each Transformer layer implements multiple subtransformers block, which aim to encode different kind of relations between regions. This encoding method combines both a visual semantic and a spatial graph.

2.1.3 Video Captioning

After the emerging of Image Captioning, the highly related task of Video Captioning was also nascent. However, unlike images that are static, working with videos requires modeling their dynamic temporal structure and then properly integrating that information for producing text in natural language. To that end, many works have came up with architectures that deal with both the spatial and temporal structure of videos and simultaneously produce descriptions.

The first time that Video Captioning and Deep Learning co-existed, was in the works of *Venugopalan et al.* [132, 133]. The principal of those works is the usage of stacked CNNs and stacked LSTMs for feature extraction from RGB frames of the video and description generation respectively. While the first study ([133]) uses *mean pooling* as the connection link between the two parts, the second ([132]) uses a more sequence-to-sequence approach exploiting also the optical flow of the images. Embodied with the an attention mechanism the model of *Yao et al.'s* [146] consists of a 3D Convnet that incorporates spatio-temporal motion features of videos. They also extract dense trajectory features like HoG [19] and concatenate them with the input alongside with attention mechanisms which learn to weight the frame features non-uniformly conditioned on the previous input-words.

Another innovative work is by *Wang et al.* [136], who propose a network with a novel encoder-decoder-reconstructor architecture (RecNet). In essence, they leverage both the forward optical flow as the backward one, for accomplishing both video captioning and reconstruction. A classical approach of a video-to-text framework, that was used in [132], is depicted on Fig. 4. From there, we can observe both the video frames and their optical flow, as well as the CNN feature extractor and the stacked LSTMs functioning as generators.

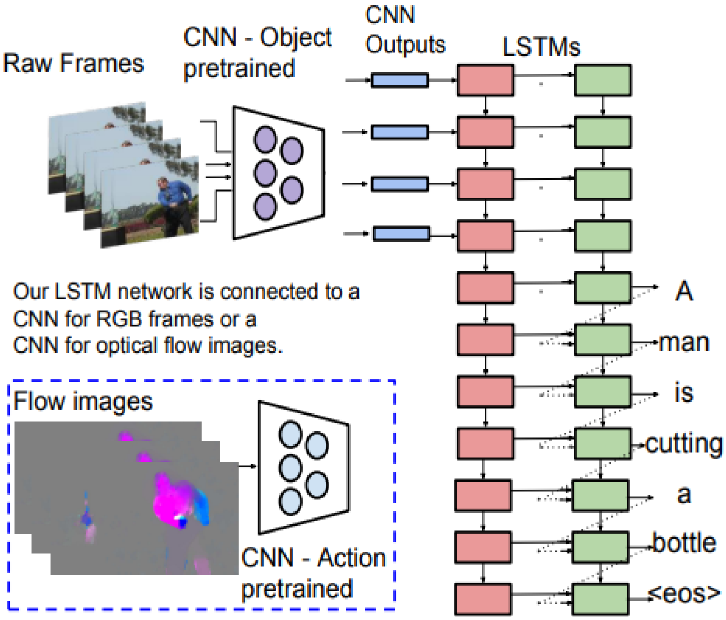


Figure 4. *An end-to-end sequence-to-sequence model to generate captions from videos. The architecture incorporates a CNN extractor of both RGB frames and optical flow and a stacked LSTM which reads these sequential inputs and generates a a corresponding sequence of words. [132]*

With the development of Transformers, video captioning met also radical changes. One of those is *Iashin et al's* [55], who presented a new dense video captioning approach that is able to utilize multiple number of modalities for event description from videos. They formulated the captioning task as a machine translation problem by utilizing the Transformer architecture and they showed that audio and speech modalities may improve a dense video captioning model when using automatic speech recognition (ASR) system. Last but not least, models of the BERT family were adjusted to video captioning task. Two important works are from *Sun et al.* and *Zhu et al.* [120, 157] who both altered the BERT architecture in order to learn joint embedding representations of video and language by applying pre-training and to focus on both global context and local details for video-text understanding.

Similarities with Visual Storytelling:

At this point, we should underline that Video Captioning is probably the closest V&L task to Visual Storytelling which is the end objective of this project. On the commons side, they are both multi-modal tasks since they engage images (or frames) and text, they both engage natural language understanding and in both cases the output is given sequentially. On the other hand, Visual Storytelling focuses on a sequence of isolated images while Video Captioning deals with video data which consists of many continual frames. Additionally, Visual Storytelling involves narrative structure that connects semantically all of the input images forming a story, whilst Video Captioning aims on describing specific events or actions within the video without necessarily adhering to a broader narrative structure.

2.2 Generalized Vision & Language Models

Having established some Vision and Language tasks and most importantly Image Captioning, the one of two main pillars of this work, now we focus our interest to some of the latest V&L models that have been developed over the last few years, especially in the region around Image captioning and Visual Storytelling. But what exactly is a V&L model? Before diving into some eminent such frameworks of the last quinquennium, we should first introduce the main conformation and the general idea behind V&L models and how do they operate in practice.

2.2.1 Introduction to V&L Models

With the development of V&L Tasks as the ones mentioned above, models that would be capable of learning simultaneously from images and texts became a necessity. Essentially, V&L models are a type of generative models that take image and text inputs, and generate text outputs. They are mostly descendants of various Transformer architectures that have been deployed to different NLP and CV tasks. Thus, many of these models are multi-tasking in the sense that they can complete a plethora of tasks, given the appropriate pre-training. As a result, many V&L models have good zero-shot capabilities, generalize well and can work with a variety of visual inputs spanning from images and videos to web pages and more.

The are several details on the structures and on the exact functionality of V&L models. The main idea is to unify the image and text representations and feed them to a text decoder for generation. Many prominent models often consist of an image encoder, an embedding projector to align image and text representations (often a dense neural network is used) and a text decoder, stacked in this order. In fact, this conformation has been adopted majorly as a common practice among V&L models. For the final generation *Attention mechanisms* play a crucial role. After the projection of the visual modality to the embedding space of the decoder, the latter applies attention across and within modalities (i.e both on individual text or visual features and for the combination of those). A typical representation of a V&L model is illustrated in Fig. 5. According to the authors, the specified model was pre-trained for Image Captioning and then fine-tuned for VQA [84].

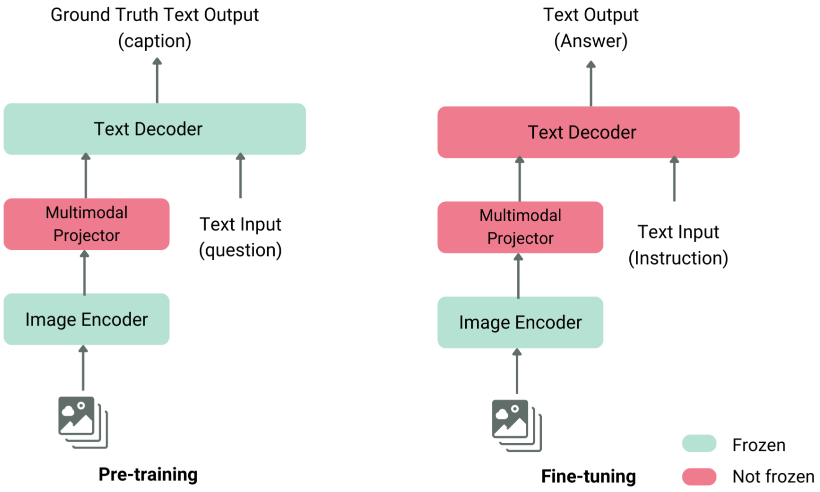


Figure 5. *Typical structure of a Vision & Language model [84]*

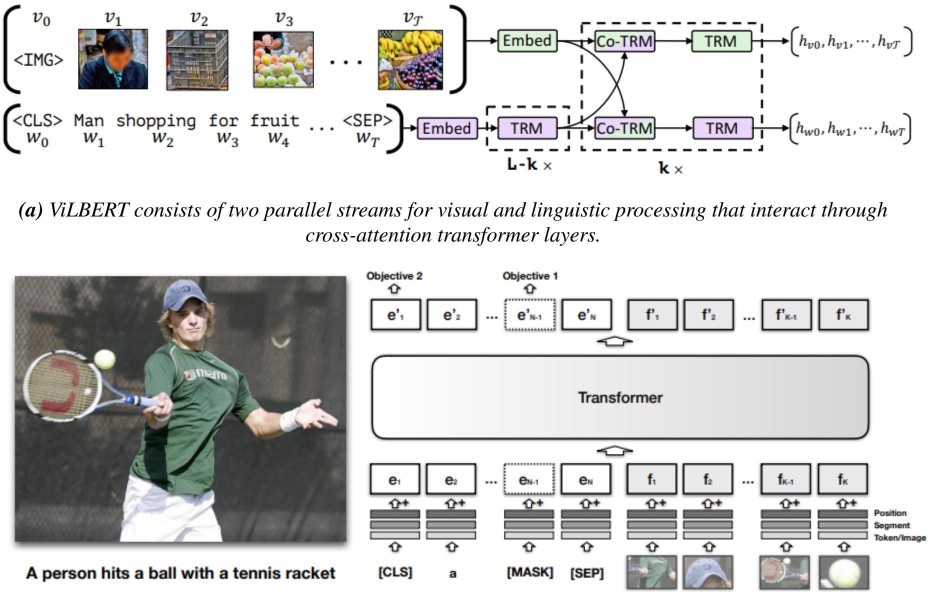
Pre-Training and Fine-Tuning of V&L Models:

Coming to the pre-training and fine-tuning procedures of V&L models, again there are several ways that these can be achieved. Regarding the former, one of the most common tricks is to keep frozen the image encoder and text decoder and only train the multimodal projector. Using concatenation or any other conflation strategy, the projected image and text features will be aligned and then fed to the decoder model for generation. Eventually, the output of this, will be compared to the ground truth text (captions). On the other hand, during fine-tuning, the idea is to keep (again) the image encoder frozen, but this time unfreeze the text decoder, and train it along with the multimodal projector, so to be able to adapt to the generating task which can be the same (captioning) or similar (question answering).

2.2.2 Former stages & the BERT family

To begin with, we will explore some such models for Image Captioning and VQA. As already seen, UNITER [16] and LXMERT [124] are two models like those. In addition, *Zhou et al.* [156] presents a unified Vision-Language Pre-training (VLP) transformer model, which can be fine-tuned for either vision-language generation (image captioning) or understanding (VQA). The unification comes after merging the Encoder and Decoder modules, unlike the majority of previous works where these parts were kept separated.

Some other important researches, concern the involve of the BERT family [21] in image captioning and generally in vision and language tasks. One of those, is named ViLBERT in [90], which is an extension of the classical BERT with the addition of a multi-modal two-stream model which can process both visual and textual inputs. ViLBERT is pre-trained through two proxy tasks and then transferred into multiple other vision-and-language tasks such as VQA and Image Captioning. Another alteration of BERT is VisualBERT [78], which feeds both text inputs and image regions into BERT aiming to discover the internal alignment between images and text with the self-attention mechanism. In this case, the pre-training of this model is done with both masked language modeling and sentence-image pair prediction. The architectures of these quite similar frameworks are pictured on Figs. 6a and 6b.


v_0 v_1 v_2 v_3 ... v_T
<IMG>
<CLS> Man shopping for fruit ... <SEP>
w_0 w_1 w_2 w_3 w_4 w_T
Embed
Co-TRM
TRM
L-k ×
k ×
$h_{v0}, h_{v1}, \cdots, h_{vT}$
$h_{w0}, h_{w1}, \cdots, h_{wT}$
Objective 2
Objective 1
e'₁ e'₂ ... e'N-1 e'N f'₁ f'₂ ... f'K-1 f'K
Transformer
e₁ e₂ ... eN-1 eN f₁ f₂ ... fK-1 fK
Position
Segment
Token/Image
A person hits a ball with a tennis racket
[CLS] a [MASK] [SEP]

(a) *ViLBERT consists of two parallel streams for visual and linguistic processing that interact through cross-attention transformer layers.*

(b) *On VisualBERT, image regions and language are fed jointly into the main Transformer, allowing the self-attention to discover implicit alignments between language and vision*

Figure 6. *Architectures of ViLBERT [90] and VisualBERT [78].*

On a similar note, ImageBERT [103] is another transformer model, pre-trained on a large-scale dataset containing weakly supervised image-text pairs (this time the pretraining included four tasks), which exhibited it's effectiveness on various vision-language tasks, including image captioning.

2.2.3 Joint image-text training

Very close to VisualBERT which was trained on image-text pairs, is SimVLM [139], a simple V&L model which reduces the training complexity by exploiting large-scale weak supervision and is trained end-to-end with *prefix language modeling* [79]. The prefix tokens consist of the patched encoded images that are processed by bidirectional attention such that the model can consume visual information and then to generate the associated text in an autoregressive manner. SimVLM was trained on image-text pairs from ALIGN [60] and text-only data from C4 dataset [107]. CM3 [2] is another autoregressive model which combines causal and masked language modeling and is particularly known for producing hypertext and thus it has been used on designing HTML web pages of noted sites like CC-NEWS and Wikipedia.

2.2.4 The CLIP family

On January 2021 OPEN AI (*Radford et al.*) published a paper entitled “Learning Transferable Visual Models From Natural Language Supervision” [104] and introduced a new type of V&L architecture, named CLIP (Contrastive Language-Image Pretraining). CLIP is a multimodal vision-language model that is trained using contrastive learning, which means that it associates similar image-text pairs on it’s input and differentiate between the dissimilar pairs. This enables the model to understand the relationships between images and their respective textual descriptions. It is designed to understand images and text in a unified manner, allowing it to perform a variety of tasks without task-specific training data. Given a dataset that contains image-text pairs CLIP is able to learn from it and then transfer this knowledge to various downstream tasks. In essence, CLIP links vision and language modalities into a unified embedding space, yielding tremendous potential for all V&L tasks including Visual Storytelling.

Since the introduction of CLIP, many works have consistently utilized it to improve the contextual results on various V&L tasks. One of the first deployments of CLIP in these tasks was made by *Shen et al.* [116]. In summary, the authors explore the advantages that the usage of CLIP, as a visual encoder (with and without pre-training), can bring in many V&L tasks such as VLN, VQA and Image Captioning. Likewise, *Portillo et al.* [102] deployed CLIP on Video Retrieval Captioning tasks, obtaining state-of-the-art results on several benchmarks. An additional CLIP-based model, targeted for image captioning is *Luo et al’s* VC-GPT [93], which avoids to place an object detector prior to the captioning model by connecting directly the visual encoder of CLIP to the used language model (LM), which is GPT-2 [106], using a self-ensemble cross-modal attention mechanism that handles both single- and cross-modal inputs. CoCa [151], is another model that uses contrastive learning and is designed as a pre-trained image-text encoder-decoder architecture which combines both contrastive loss and generation-caption loss during training. Interestingly, CoCa accomplished very high zero-shot transfer knowledge, on a variety of multi-modal evaluation tasks.

However, probably the most notable work that utilized CLIP specifically for image captioning is [96]. In this groundbreaking approach, *Mokady et al.* address the task of image captioning by using CLIP encoding as a prefix to captions [79]. This procedure involves employing a simple mapping network in order to connect aptly the CLIP encoder to the language decoder as well as fine-tuning the language model (LM) in case a simple mapping network is used. The authors show that without additional annotations or pre-training, CLIP-Cap (as they name the model) efficiently generates meaningful captions for large-scale datasets while simultaneously remains extremely light-weight for training.

Furthermore, CLIP-Cap became influential for some subsequent works such as *Tewel et al’s* [126] and *Ramos et al’s* [108]. In the former, the authors propose an entirely unsupervised approach (no training or tuning, named ZeroCap), by essentially using CLIP-Cap’s architecture to perform zero-shot image captioning. They observe the capability of a CLIP-based model to create reasonable captions, beyond the prefix prompted zero-shot learning that [96, 104] propose. In the latter, CLIP-Cap’s structure is exploited once again with the goal of reducing it’s trainable parameters even more. They key contribution is that this model (SMALLCAP as they name it) uses only interleaved cross-attention layers between the CLIP encoder and the language decoder, thus reducing the training time, and that it generates captions conditioned on the input image and some related captions retrieved from a datastore.

2.2.5 Other types of V&L models

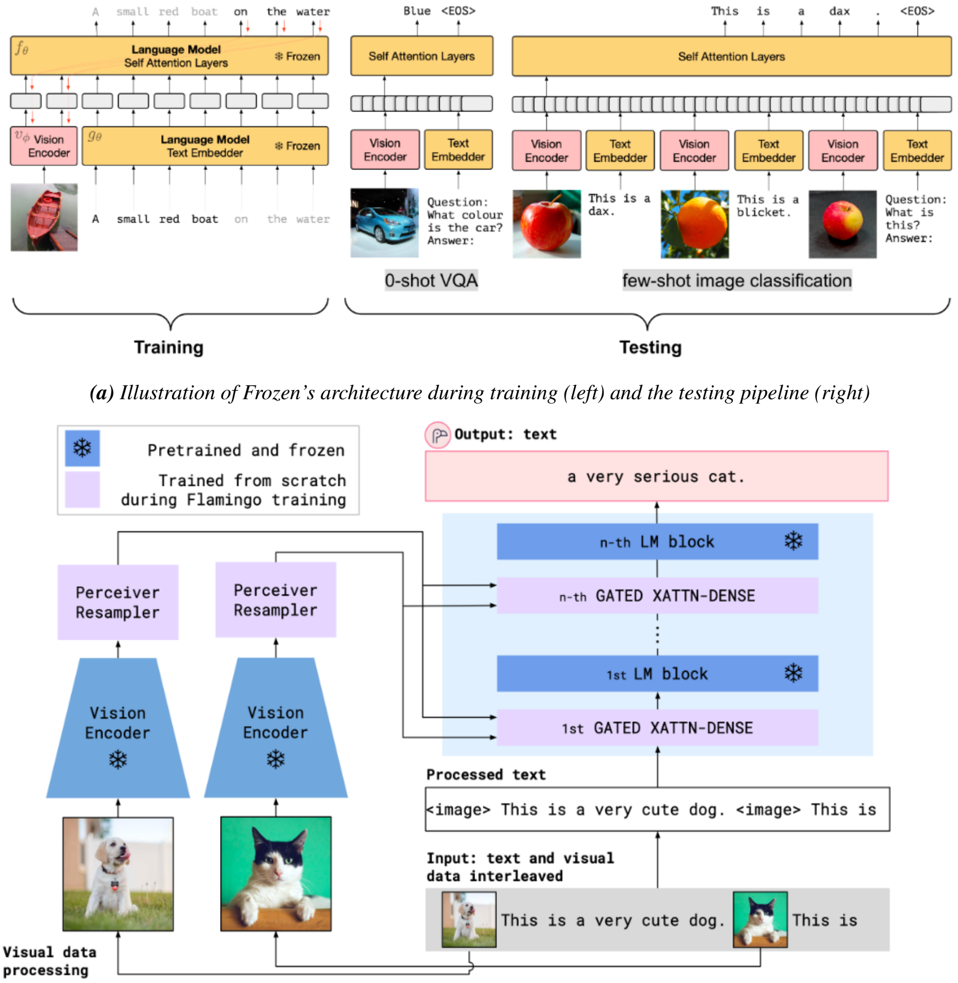
Outside of the limits of CLIP, there are other models that have been well established in the general area of V&L tasks in the last few years. Starting with, is OSCAR (Object-Semantics Aligned Pre-training) [80], a model which uses object tags detected in images, as points of notification that can be used as an alignment between visual and textual features. During pre-training OSCAR collocates *Contrastive Learning* along with *Mask Language Modeling* [21] since it incorporates both the contrastive loss and the mask language loss given by the input image-text pairs which in this case are represented as a triple (word tokens, object tags, region features). *Zhang et al’s* work [154] is considered to be another major object-centric model in which OSCAR is integrated as well. The main focus of the paper is to improve visual representations for V&L tasks by utilizing a much larger object-detector which contributes to richer semantics and visual concepts which eventually will generate more accurate responses.

Very similar to CLIP-Cap, *Tsimpoukelli et al's* model (Frozen) [129], extends the notion of static prefix and prompt tuning [74, 79] by making a dynamic prefix, in the sense that now it is not a constant bias that is added to the text embeddings, but an input-conditional extension produced by a neural network. In this whole process, the LM remains totally frozen. The resulting structure is a multimodal zero-shot or few-shot learner, with the ability of adaptation to a variety of relevant tasks. Fig. 7a, presents briefly how training and testing procedures are taking place in Frozen.

Working on a different way, *Li et al.* [77], involved bootstrapping in an attempt to create a system that jointly deals with understanding-based tasks and generation-based tasks. More precisely, a captioner is used for refining the training data by generating synthetic captions with a filtering step to mitigate the noise present in the given web-collected descriptions. Ultimately, Flamingo proposed by *Alayrac et al.* [4], is another framework for few-shot learning which interleaves cross-attention fusing layers into the text decoder and interposes a specific transformer-based schema called *Perceiver* [57], between the latter and the visual encoder. Once again, during this process the LM remains frozen. It is worth mentioning, that the training procedure in Flamingo uses *NLL loss* (instead of the most common *Cross Entropy*)[2], in an autoregressive manner. More precisely, while training the following actions are done alternately:

- The Perceiver receives spatio-temporal features from the vision encoder of image/video inputs to produce fixed-size visual tokens.
- The frozen LM is equipped with newly initialized cross-attention layers interleaved between the pre-trained LM layers. Thus the LM can generate text conditioned on the above visual tokens.

A visual depiction of these mechanisms is given on Fig. 7b. Due to these facts, Flamingo outperformed many task-specific models on numerous benchmarks, such as COCO [82] or VQAv2 [37] datasets.



***(a)** Illustration of Frozen's architecture during training (left) and the testing pipeline (right)*

***(b)** An Overview of the Flamingo model*

***Figure 7.** Architectures of Frozen [129] and Flamingo [4].*

[2] NLL loss in the Negative Log Likelihood loss and comes in handy when training for classification problems with C classes. More information about loss functions (including NLL & Cross Entropy loss) can be found on Appendix A.

2.3 Visual Storytelling

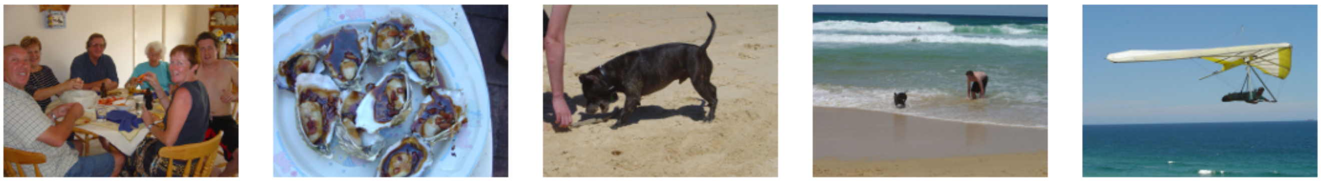
Thus far, we have explored various Vision and Language tasks and models, yet we have not delved into the primary objective of this study, which is Visual Storytelling. In the following paragraphs, we will break down some techniques that have been applied for Visual Storytelling over the last couple of years.

2.3.1 Preliminary Levels & Birth of Visual Storytelling

As already mentioned Video Captioning can be placed quite close to Visual Storytelling as a task. To that end, some works have approached Visual storytelling via very similar tasks such as video summarization [34] by using supervised probabilistic models. Others, like *Park and Kim* [100], came even closer to the traditional definition of Visual Storytelling, by applying a multimodal architecture called *Coherent Recurrent Convolutional Network (CRCN)* in order to get a sequence of natural sentences for a stream of images.

Nevertheless, Visual Storytelling was properly introduced in 2016 with the construction of a pioneering dataset named the VIsual StoryTelling (VIST) [54]. This work created the first dataset for sequential vision-to-language exploration. Initially the dataset included 81,743 unique photos in 20,211 sequences, aligned to both mere language descriptions (captions) and story language. In particular, the authors dub the captions as “Description of Images in Isolation - DII” and the stories as “Stories of Images in Sequence - SIS”.

Moreover, for producing both DII and SIS, the authors applied crowdsourcing, resulting all the annotations to be human-written. The main purpose of Visual Storytelling is to generate a sequence of sentences that collectively form a coherent story, a task that requires not only understanding the individual images but also composing a narrative structure. With that said, the authors clearly differentiate Visual Storytelling from Image Captioning. Last but not least, the paper proposes new evaluation metrics for assessing the quality of generated stories and based on those it gives some baselines on the novel task. Some examples of such constructed stories are demonstrated on Fig. 8[3].



+Viterbi	This is a picture of a family. This is a picture of a cake. This is a picture of a dog. This is a picture of a beach. This is a picture of a beach.
+Greedy	The family gathered together for a meal. The food was delicious. The dog was excited to be there. The dog was enjoying the water. The dog was happy to be in the water.
-Dups	The family gathered together for a meal. The food was delicious. The dog was excited to be there. The kids were playing in the water. The boat was a little too much to drink.
+Grounded	The family got together for a cookout. They had a lot of delicious food. The dog was happy to be there. They had a great time on the beach. They even had a swim in the water.

Figure 8. *Example of stories introduced in the paper of Visual Storytelling [54].*

With the introduction of VIST dataset, new opportunities arose for researchers to develop models and evaluate their performance specifically on Visual Storytelling tasks. As a consequence, this landmark dataset paved the way for creating pioneering systems that would be able to produce meaningful text and opened the path for innovative exploration in the common region of CV and NLP, sparking new avenues of research.

2.3.2 Post VIST era

The first attempts that were engaged with the exploration of the Visual Storytelling dataset included various studies that exploited techniques such as neural networks and deep learning. Others, went one step further by exploiting reinforcement learning-based strategies.

[3] It is worth mentioning that the authors in [54] opted for creating stories that were of length 5, meaning that each story will unfold over 5 images, exactly as the example on Fig. 8. The same schema was adopted in the present work.

Neural Networks Deployment:

Among the first works that experimented on VIST dataset was by *Agrawal et al.* [3], who retrieved jumbled images and their respective captions from the foresaid dataset and proposed a method of sorting them into a coherent story (something quite similar to our goal), by using two different type of networks: 1) A language-alone unary model that uses a Gated Recurrent Unit (GRU) [17] and 2) A language-vision binary model that embeds both the caption and the image using also CNNs. What is more, *Liu et al.* [88] also explored the generation of structured paragraphs from photo streams of the VIST dataset, but unlike [100], their model encompassed an additional attention mechanism to combat the large visual variance between the photo collection and to preserve the long-term language coherence among a multiple sentence text.

Gonzalez-Rico and *Fuentes-Pineda* in [35] proposed a neural visual storyteller that creates short stories from image sequences, extending the capability of the model by *Vinyals et al.* [135], which generated mere image descriptions. The extension relies on a series of encoder LSTMs (instead of only one in [135]), to compute a context vector of each story from the input sequence of images. Then, this context vector serves as the initial state for several separate decoder LSTMs, each of which generates the story segment for the corresponding image in the input sequence.

Kim et al's work [66] stands as a major turning point in the history of Visual Storytelling. This paper suggests a deep learning network model, designated as GLAC-Net (Global-Local Attention Cascading Network). This network is very alike to many previous models (CNN feature extractor, LSTM generators) but it adds two variations. Firstly, as its name indicates, it uses two types of attention mechanisms, one local (image feature level) and one global (the overall storyline encoding level). Secondly, it exploits a cascading mechanism which means that during generation the hidden state (context) of the previous sentence is conveyed to the next sentence. The analytical processes of this model are given in Fig. 9. From there, we can distinguish a ResNet image-feature extractor, the LSTMs, in the Encoder and Decoder and in between the global-local attention mechanism. As a result, GLAC-Net contributed to very competitive results on the VIST dataset and was instituted as a robust storytelling model.

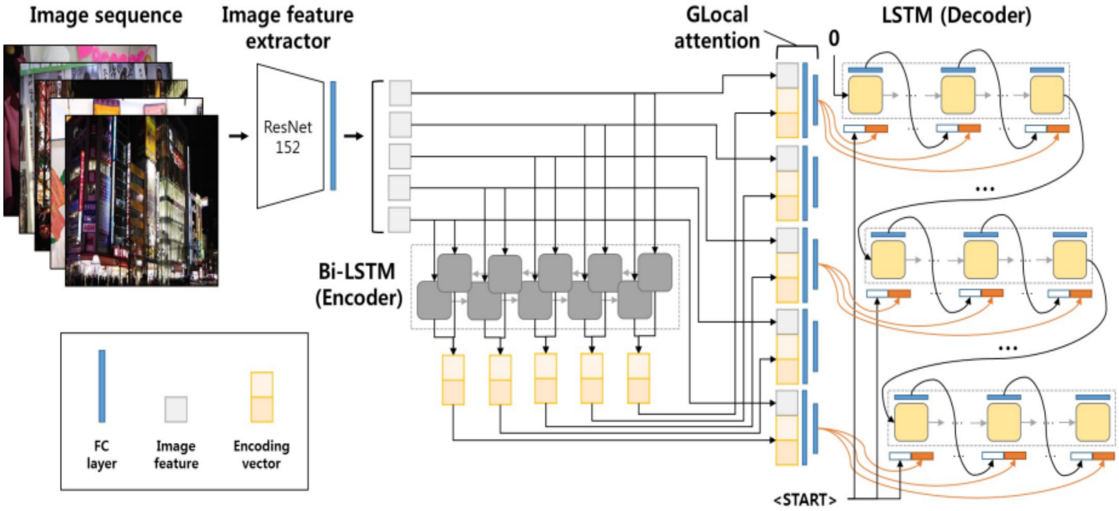


Figure 9. The global-local attention cascading (GLAC) network model for visual story generation [66].

Deep Hierarchical Approaches:

After the release of VIST dataset, two contemporary works proposed deep hierarchical approaches to Visual Storytelling which were by *Krause et al* [71] and by *Yu et al* [152]. These studies, extend the idea of image captioning and deal with the complicated task of visual storytelling by developing deep recurrent architectures with special modalities. More specifically, in [71] the model composes of an object-detector that projects image features (via pooling) to the space of two RNNs accountable for the coherence of the generated text, in word and sentence level respectively. Concurrently, the model of [152], consists of an image-features extractor (ResNet101 [42]) and three different RNNs, which encode the given album (as a whole), isolate the most relevant photos to the matched descriptions and generate the final story correspondingly.

A further hierarchical approach to our task, was given by *Nahian et al.* [97] who controversially used the sole image descriptions themselves (along with the images of course) of the VIST dataset, so to generate stories that maintain their context and coherence. Model-wise their architecture comprises of different levels having multiple encoders and decoders for both isolated and sequential images and/or descriptions.

Application of Reinforcement Learning:

Wang et al's work [138], was regarded as another significant milestone for the Visual Storytelling field. To begin with, the authors highlight the limitations of conventional metrics such as BLEU and METEOR in assessing the quality of generated stories and afterwords, to address this problem they propose an adversarial reward learning (AREL) framework, which includes a policy and a reward model. In a nutshell, the former takes an image sequence as input and attempts to form a narrative story while the latter, is optimized adversarialy and aims to bring the prediction closer to the respective human annotations. An overview of this framework is showcased in Fig. 10.

It becomes obvious that AREL is improving the quality of the sampled story with the constant interaction between it's components, namely the *policy model*, the *reward model* and the *objective function*. Availing both automatic metrics and human evaluation this system established state-of-the-art results in the VIST dataset and it was one of the first models to approach human-like stories. Similar to the case of GLAC-Net, AREL is also considered to be a mainstay for many newly developed frameworks for Visual Storytelling tasks, even till today.

Similarly to [138], *Hu et al* [52] proposed another storytelling framework that applies reinforcement learning by exploiting some specific reward functions. Working also on the VIST dataset, the paper conducts a study on what are the characteristics of a coherent story, emphasizing that there are three such key-criteria: *relevance, coherence and expressiveness*. Subsequently, the authors use the aforementioned model trying to capture the essence of these three characteristics when generating stories.

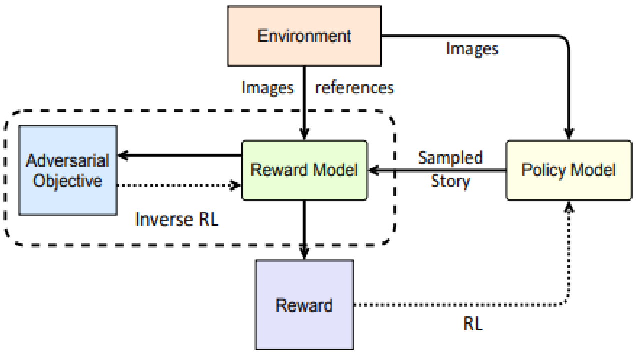


Figure 10. *The Adversarial Reward Learning (AREL) framework used for Visual Storytelling [138].*

2.3.3 Latest advancements

As the majority of fields in deep learning, most of the contemporary approaches are related to transformer-based models accompanied by several other boosting methods.

Transformer-based Approaches with external Knowledge Graphs:

With the new decade, noteworthy progress continued to emerge in the field of Visual Storytelling, some of which were marked by the integration of Transformer-based models. Perhaps one of the first works that implemented a versatile architecture consisting of multiple levels of transformer-based sub-parts, was by *Hsu et al.* [49]. In this work, the authors presented how the utilization of external *Knowledge Graphs*, can aid for story generation. To that end, a three-stage framework, named *KG-Story*, is developed. Collectively, this model applies three type of actions dubbed as *distill-enrich-generate*. Initially, the prompted images are distilled to word-terms in order to get descriptive representations. Following this, these word-terms are enriched to a linked conceptual path using the external graphs and finally, a simple transformer model [130], enhanced with positional encoding converts these paths to stories.

In a like manner, authors in [119] combine BERT [21] and hierarchical LSTMs to accomplish the generation of semantically complete stories. Conforming another hierarchical approach (from [71]), *Su et al.* attempted to approach Visual Storytelling, with a framework that models word-level and sentence-level semantics separately. Except of using a CNN image-feature extractor (VGG16 [117]) the paper deploys BERT to obtain textual embeddings from sentences/words and feeds them (along with image-features) to the hierarchical LSTM decoders.

The fragmentation, of Visual Storytelling as task, continued at *Hsu et al's* work [50], which introduces PR-VIST, a multi-stage model that generates human-relevant stories. Much like their previous work in [49], this model also takes advantage of external representation graphs dealing with storytelling in several stages. More specifically, PR-VIST depicts the input image as a visual graph in which the elements of the image stand as nodes and ultimately finds the optimal path that forms the best storyline. Subsequently, in the second stage of the framework, this storyline is reworked in a similar fashion that an image is generated by Generative Adversarial Networks (GANs) [36]. In particular, a story generator takes the storyline and produces the actual story, back-propagating the loss which is given by the evaluator during the classification of the generated story as good or bad. An overview of this proposal is highlighted on Fig. 11.

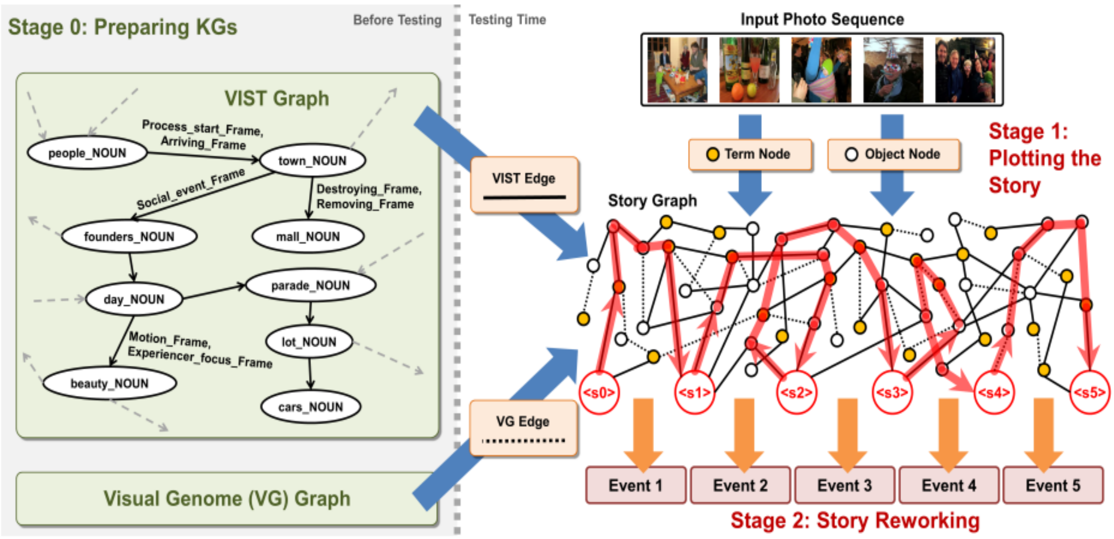


Figure 11. *In Stage 0 (Preparation): A collection of knowledge graphs encoding the relationships between elements is prepared. Stage 1 (Story Plotting): PR-VIST constructs a graph that captures the relationships between all elements in the input image sequence and identifies the optimal path in the graph to form the best storyline. In Stage 2 (Story Reworking): The model utilizes the identified path to generate the story, employing a story generator and a story evaluator to facilitate the process. [50].*

VIST Dataset Expansion & Criticism:

Taking into consideration the importance of GLAC-Net and AREL frameworks, *Hsu et al.* [51] utilized these two models in their process to expand the VIST dataset by adding human edits on machine-generated visual stories. The new dataset, entitled as *VIST-Edit*, includes 14,905 human-edited versions of 2,981 stories generated by AREL and GLAC-Net and underscores the weak correlation between the currently developed automatic evaluation metrics and the more thoughtful human ratings. Operating also in the dataset-level *Ravi et al.* [109], proposed *AESOP* (Abstract Encoding of Stories, Objects, and Pictures). This new dataset contains different themes and it's stories are highly narrative, coherent and follow a clear causal arc having even a moral at the end. Thus, *AESOP* fosters a more creative and causal reasoning taking Visual storytelling to it's next level. Lastly, the authors set some baselines for this novel dataset, creating ample opportunities for further exploration and feature research.

An additional study that criticized the traditional natural language generation metrics which are based on n-gram matching, such as BLEU or CIDEr, is by *Wang et al.* [137]. Claiming that these scores have weak correlation with corresponding human evaluation/judgment, the paper proposes and mathematically formulates three new scores: 1) visual grounding, 2) coherence, and 3) non-redundancy, which are eventually measured in reliability on the VIST dataset. Ultimately, *Liu and Keller* [83] inspired by [109], extended the notion of Visual Storytelling to a more character-centric point of view. Underlining, that a coherent story does not necessarily imply a narrative with protagonists and action, the paper introduced the *VIST-Character* dataset which provides rich character-centric annotations, including visual and textual co-reference chains and importance ratings for the characters.

2.4 Text to Text Generation

The purpose of this work is to examine, unlike most of the current literature review [54], the step of transitioning from Image Captioning to Visual Storytelling. As explained in 1.3 and 1.4, in order to do so, we will need to deploy a novel architecture which will be capable of reformulating captions to coherent and narrative storylines. For this purpose, it would be valuable to acquaint ourselves on how previous studies have dealt with the problem of text-to-text reformulation (generation).

The idea of storytelling is not new and even with the dawn of the new millennium, works developed multi-agent frameworks in virtual environments [33, 127] with the ability of generating natural language and undoubtedly till today, there are plenty of papers dedicated to story generation most of which concern pre-trained (Large) Language Models (LLMs) [1, 11, 21, 65, 107] or their variations which they are able of completing a large number of tasks including text-reformulation and text-generation. Nonetheless, the works that use specifically prompted sentences or keywords to construct a story are more limited and precisely, to the best of our knowledge, the methodology of "evolving" from Image Captioning to Visual Storytelling has not been applied yet.

2.4.1 RNN-based modules

Quite close to our purpose, *Jain et al.* [58] propose a sequence-to-sequence RNN [122] to address the task of coherent story generation from independent descriptions. In addition to that, they also deal with the task of text-generation as a Statistical Machine Translation (SMT) problem and they use two popular methods: Phrase-based-SMT and Syntax-based-SMT, which they compare it head-to-head with their deep learning-based approach.

Yao et al. [147] propounded a *plan-and-write* hierarchical generation framework that given a title as input, first plans a storyline and then generates a story based on that storyline. The hierarchical approach is composed by two levels: the dynamic and the static parts. The former (composed by a *Gated Recurrent Unit - GRU* [17]) is accountable for generating the next word in the storyline as well as the next sentence in the story and at each timestep uses the previously generated text as context (initialized with the title), whilst the latter, composed by an LSTM, generates a whole storyline straight away, giving the general realization of the story, and thus enhancing its coherence.

2.4.2 Transformer-based modules

Moving at the same pattern, *Guan et al.* [40] came up with a transformer-based architecture that firstly, utilizes commonsense insights from external knowledge bases and secondly, uses multi-task learning with the objective of distinguishing true and fake stories during fine-tuning, with the final aim of generating reasonable stories given the central context of the line. The model outperformed other state-of-the-art language models such as GPT-2 [106] on different evaluation metrics like Perplexity (PPL) [115] and BLEU.

Inspired by these previous works and based on [65], *Xu et al.* [143] proposed *MEGATRON-CNTRL*, a novel large-scale language model, that adds control to text generation by incorporating once more, external knowledge sources. This architecture is comprised of four main elements: a keyword predictor, a knowledge retriever, a contextual knowledge ranker and a conditional text generator which are respectively responsible for: setting the general theme of the story, receiving the external information, ranking this information and promoting only the useful parts and generating the final story. A summary of this algorithm can be seen on Fig. 12. It's worth emphasizing, that *MEGATRON-CNTRL* framework showcases remarkable controllability and simultaneously immense fluency, consistency and coherence when generating stories under both automatic metrics and human evaluation.

Likewise, *Fang et al.* [26] worked also in controllable story generation by devising another transformer-based framework which heavily depends on text-prompts. Specifically, the authors integrate latent representation learning on a pre-trained transformer model, for constructing a Conditional - Variational Autoencoder (CVAE) [155][4]. In order to incorporate correctly the latent representation vectors in CVAE, they use two

[4] Variational Autoencoders (VAEs) [68] are generative type of models that learn to map data in a latent space in a compressed representation, and then to produce new samples from this space. Unlike standard autoencoders, which encode information deterministically, VAEs introduced randomness in the procedure, learning a continuous, smooth latent space.

alternative injection mechanisms based on: 1) A simple addition of the input token to the positional embeddings, 2) Projection of encoded representation to a larger space through “pseudo-attention layers”. Eventually, this study achieves both controllability and state-of-the-art efficiency, over story generation under automatic metrics such as PPL and ROGUE (along with it’s variations).

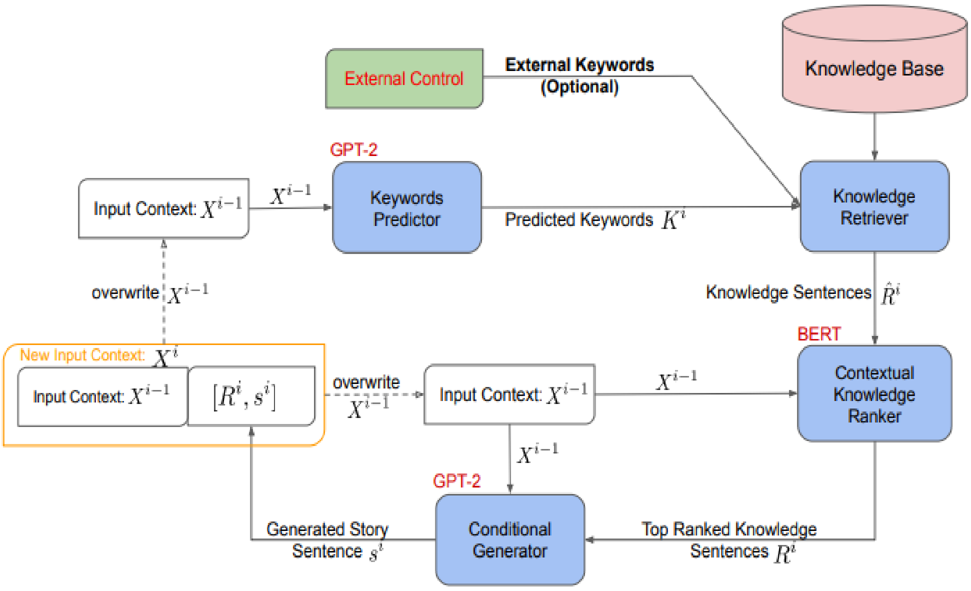


Figure 12. *Overview of the MEGATRON-CNTRL story generation method. Based on an input context, the knowledge predictor generate keywords for future context. Using these keywords, the retriever identifies relevant knowledge from an external knowledge-base and filters them based on their relevance to the context. Finally, the generation procedure is controlled by only focusing into the top scored knowledge sentences [143].*

3 Methods

Having established the necessary background in Vision & Language tasks and models, especially in Image Captioning and Visual Storytelling, and having also a slight acquaintanceship with text-to-text story-generation models, now we will give a brief overview of the data that will be used and the methods that will be deployed/implemented throughout this project.

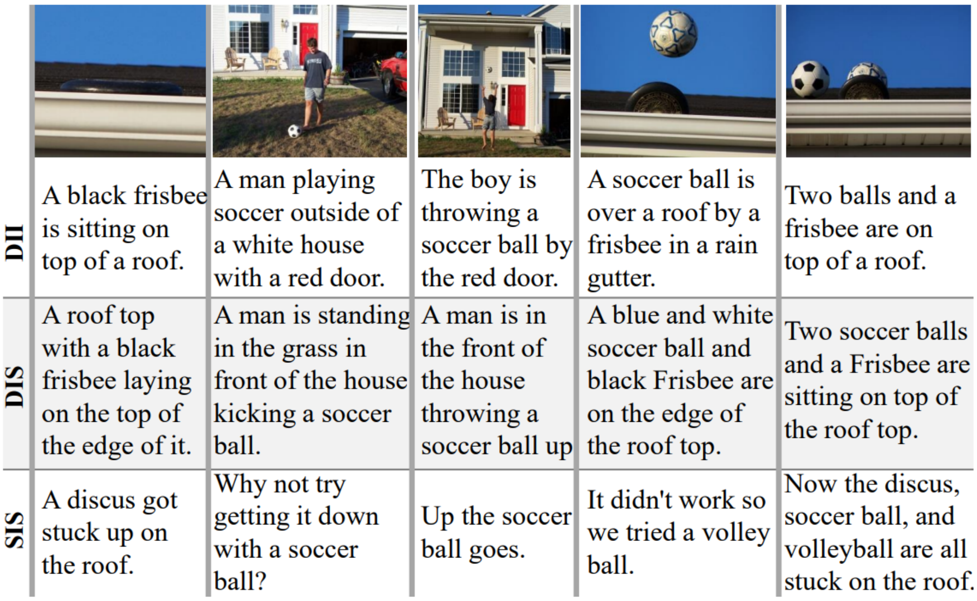
3.1 Dataset

Similarly to the majority of Visual Storytelling studies that were previously mentioned, this work will also make use of the VIST dateset [54]. To that end, in the following paragraphs we adduce some general information for the foresaid dataset.

3.1.1 General Information

As already mentioned on 2.3.1, VIST dataset is the first collection of sequential images with corresponding isolated descriptions and human-made stories and it introduces the task of *Visual Storytelling*. The dataset initially included 81,743 unique photos in 20,211 sequences and is divided in albums, for each of which, crowd-workers were selected to form five-image sequel by completing a linguistic story, consisting of five sentences as well. In total, for each of the stories, the dataset contains aligned three tiers of information. 1) *Descriptions of images-in-isolation (DII)*; 2) *Descriptions of images-in-sequence (DIS)*;[5] and 3) *Stories for images-in-sequence (SIS)*. An example, of a five-images story along with all of *DII*, *DIS* and *SIS* annotations is shown on Fig. 13.

To obtain more insights about the complicated structure of the VIST dataset, we conducted a research, from which we extracted the precise statistics of the dataset. Regarding albums the dataset consists of 8,031 albums, in the training set, 998 albums in the validation and 1,011 albums in the test set. In summary, from those albums, 50,200 stories are constructed. In particular, the stories in training, validation and test sets are respectively: 40,155, 4,990 and 5,055. At the same time, regarding the number of images, the dataset contains 167,528 unique images in the training set, 21,048 images in the validation set and lastly 21,075 images in the test set. These images, along with their respective annotations (story-sentences), are usually grouped by five in order to form a visual story.



DII	A black frisbee is sitting on top of a roof.	A man playing soccer outside of a white house with a red door.	The boy is throwing a soccer ball by the red door.	A soccer ball is over a roof by a frisbee in a rain gutter.	Two balls and a frisbee are on top of a roof.
DIS	A roof top with a black frisbee laying on the top of the edge of it.	A man is standing in the grass in front of the house kicking a soccer ball.	A man is in the front of the house throwing a soccer ball up	A blue and white soccer ball and black Frisbee are on the edge of the roof top.	Two soccer balls and a Frisbee are sitting on top of the roof top.
SIS	A discus got stuck up on the roof.	Why not try getting it down with a soccer ball?	Up the soccer ball goes.	It didn't work so we tried a volley ball.	Now the discus, soccer ball, and volleyball are all stuck on the roof.

Figure 13. *Example of a five-images story along with the descriptions of images in isolation (DII); descriptions of images in sequence (DIS); and stories of images in sequence (SIS) [54].*

[5] Note that in the release of the dataset, DIS annotations were not published.

However, it should be underlined that due to crowdsourcing the number of annotations in DII and SIS is not the same[6]. Recalling also the fact that our work focalizes on how we could use the image descriptions (captions) to produce an interesting story, it becomes plausible that we should turn our attention only on SIS annotations that are present on DII annotations and as well as they form a five-length story. Therefore, this procedure reduces the total number of training stories to 26,959, validation stories to 3,354 and test stories to 3,385. In Table 1, we depict the complete statistics of the VIST dataset with regard to the initial and subsequent annotations and stories before and after the disposal process. It also, shows the total number of albums, from where the Stories-in-Sequence were derived as well as the total number of images in each set.

VIST Dataset Statistics			
Data Type\Set	**Train**	**Validation**	**Test**
SIS Albums	8,031	998	1,011
Images	167,528	21,048	21,075
DII Annotations	120,465	14,970	15,165
SIS Annotations	200,775	24,950	25,275
SIS Stories (of length 5)[7]	40,155	4,990	5,055
SIS Annotations with photo_id in DII	159,899	19,977	20,080
DII-SIS Annotations repeated 5 times	134,795	16,770	16,925
DII-SIS Stories (of length 5)	26,959	3,354	3,385

Table 1. *Analytical statistics of the VIST dataset regarding the number of annotations, stories, albums and images*

3.1.2 Inner structure of the Dataset

On an implementation level, the dataset's annotations were provided as dictionaries, allowing for the correspondence between it's visual and linguistic components. Table 2 illustrates the structure of these dictionaries for both the description and the story tiers (DII & SIS).

DII		**SIS**	
Key	**Value**	**Key**	**Value**
original_text	Big old tree being photographed on a sunny day.	*original_text*	To see the final glimpse of the roots, extending out into the depths of the hill.
album_id	72157605930515606	*album_id*	72157605930515606
photo_flickr_id	2626983575	*photo_flickr_id*	2626982337
photo_order in_story	3	*setting*	first-2-pick-and-tell
worker_id	K4PTO2GKQ6BUJDA	*worker_id*	SY6QQXJCXXMNCYP
text	big old tree being photographed on a sunny day.	*story_id*	30355
tier	descriptions-in-isolation	*tier*	story-in-sequence
-	-	*worker_arranged photo_order*	2
-	-	*text*	to see the final glimpse of the roots, extending out into the depths of the hill.

Table 2. *Overview of DII and SIS Annotations*

[6] There are more annotations in SIS, because crowd-workers could repeatedly pick the same images from the given albums.

[7] Here we length of 5 we mean that the generated stories contain five images with five story descriptions. This is obvious for the initial SIS stories but after the truncation due to the demand of correspondence with the DII, all stories are not of length 5.

From Table 2, we can observe that while the two tiers share many similarities, they also exhibit some differences. The most crucial piece of information is the *”original_text”* field, which provides either the caption (in DII) or a story sentence (in SIS). We should highlight here, that for each image, VIST dataset included three possible captions, a trait that could serve well for data augmentation purposes. Moreover, the *”photo_flickr_id”* serves as a key identifier for correspondence between the two tiers, enabling us to track images that appear in both DII and SIS. Lastly, in the SIS tier, the *”story_id”* is particularly important for identifying and managing the stories to be retained or excluded. This will be our reference point for the second phase of the project.

Furthermore, since the project's purpose is to generate and evaluate stories that are part of the VIST, we would like to dive a bit deeper on exploring the linguistic characteristics of the dataset. We already know that in the second phase of the work a transformer-based language model (as we will see later on sub-section 3.2.3) will reformulate the produced captions (generated from the first phase) to stories. It becomes vivid that during the training process of that model we will have to deal with both the DII part of VIST (as the inputs-captions), as well as the SIS (as the target-stories).

In the context of training a language model, it is essential to determine the exact sizes of the vocabularies (number of unique tokens) used in both the input domain of captions (DII) and the output domain of stories (SIS), in order to make our second phase model capable of processing text information. To that end, a token-wise exploration took place regarding the respective train, validation and test sets of the input and output tiers. The overview of the sizes of the vocabularies for each of these sets are presented in Table 3.

VIST Dataset Vocabularies			
Tier	**Set**	**Number of Tokens**	
DII (Inputs)	Train	29330	32745
	Validation	10067	
	Test	10260	
SIS (Target)	Train	30981	34603
	Validation	10836	
	Test	11382	
Total VIST vocabulary			49535

Table 3. *Vocabulary size per Set and Tier for VIST dataset*

3.1.3 Pre-processing data for the second phase

Looking back at Table 1 we can recall that the total number of full stories (five in length) were 26,959, 3,354 and 3,385 for the train, validation and test sets respectively. Howbeit, due to a small minority of truncated images[8], when transitioning from phase 1 to phase 2, these numbers were diminished to 26,570, 3,319 and 3,338. Each of these stories, were arranged on tuples based on their *story id*, which was the identification key of the tuple. As values we set another tuple, which comprised of the 5×3 captions (since for every image, the dataset gave three possible captions) and the 5 sentences of the original story.

Nevertheless, we split the 5×3 captions to 3 sub-datasets where for each *story id* we had 3 alternatives of 5 captions and the 5 sentences of the original story[9]. For instance, you can take as an example, Fig. 19 from paragraph 3.3.1 below. From there, the "Input Captions" and the "Target Story", could be considered as a pair that consists one of the three alternative sub-datasets. Lastly, we should emphasize that thereafter, all mentioned models which were utilized during phase 2, were evenly trained (or/and fine-tuned) on all three alternative sub-datasets. This means that for a random number of epochs we give: $epochs/3$ to the 1^{st} sub-dataset, $epochs/3$ to the 2^{nd} alternative sub-dataset and the rest to the last sub-dataset.

[8] The images were given on a different folder than the annotations and some of those were found truncated when they were encoded by CLIP for phase 1. To that end, the whole story sequence that were part of, was discarded. As we will see later this would not cause a problem if we were to use the two parts of our main pipeline model from Fig. 14 individually, since we only care for the descriptions in the second stage. However, we decided to adhere to uniformity with our first phase, since the ultimate purpose of the project is to use the same images, with their produced captions to form a final story at once.

[9] This essentially augmented the data. Also for all three sub-datasets we had the same story as pair.

3.2 Framework

In this stage we are going to explore the framework that was utilized during our work. Starting from plain (but correlated) series of images, passing to caption generation for each single input images and finally transforming the isolated caption to a cohesive, complete and narrative story. As previously underlined in both sections 1.3 and 1.4, we will deal with these steps separately and thus, we will utilize two different architectures to accomplish our goals.

Firstly, comes the *Vision-to-Caption* architecture which practically consists of the known to us Clip-Cap model [96] and then is the *Caption-to-Story* architecture which will comprised of a transformer-based model. For the second stage, well-known language models could readily be deployed. However, our intention is also to built from-scratch an architecture, that will be able to generate text, given inputs in the same form (captions in text form). A more detailed overview of the proposed system is provided in the following paragraph 3.2.1. Subsequently, we expand our analysis on each individual component of the total framework.

3.2.1 Overview of the proposed Framework

At this point, having clear in our minds that we are dealing with the task of Visual Storytelling in two different steps we can now present the proposed framework which was used on this work. This is depicted in Fig. 14. From there, we can easily distinguish the vision-to-caption model which is comprised by Clip-Cap and the caption-to-story architecture (model represented by the orange oval contour). Regarding the latter, we came up with two distinctive alternatives:

First, is the self-created transformer-based model named **T4** (**t**ext-**t**o-**t**ext **T**ransformer)[10], which was built and trained from-scratch solely on the captions and stories of the VIST dataset. Secondly, we have two quite popular Encoder-Decoder Transformer architectures[11], namely *T5* [107] and *BART* [75] which have achieved state-of-the-art results on plenty of Natural Language Generation (NLG) tasks. In addition, it should be noted that both of them are pre-trained on a massive amount of textual data for different NLP & NLG tasks, such as *General Language Understanding Evaluation (GLUE)* tasks, *Question-Answering* tasks and other *Generation* tasks.

Nonetheless, it's important to emphasize that the proposed framework of Fig. 14 is applied in a serial manner only in the inference time, whilst in training (fine-tuning) time the two sup-parts can be considered as totally independent. In practice, this means that while training (fine-tuning) Clip-Cap for generating captions for the Train set of VIST dataset, we don't feed these captions to the Text-to-Text models for their own fine-tuning. Instead, we use again the reference stories provided by VIST in order to maximize the generative ability of our second-phase models as well. So, only during test-time the completely fine-tuned pipeline will be leveraged to generate stories for the series of images originating from the Test set of VIST dataset.

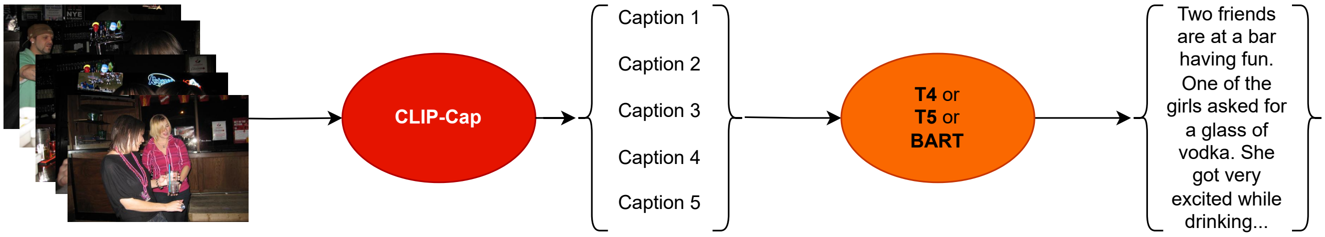

CLIP-Cap
Caption 1
Caption 2
Caption 3
Caption 4
Caption 5
T4 or
T5 or
BART
Two friends are at a bar having fun. One of the girls asked for a glass of vodka. She got very excited while drinking...

Figure 14. *An overview of the proposed Framework*

3.2.2 The Vision-to-Caption architecture

In this section, we will describe the main architecture that we are going to use in the first phase, which is Clip-Cap [96] by *Mokady et al.* In a nutshell, Clip-Cap is comprised by three sub-parts; the CLIP model, a language generator model and a mapping network that projects the CLIP embeddings in a way that are

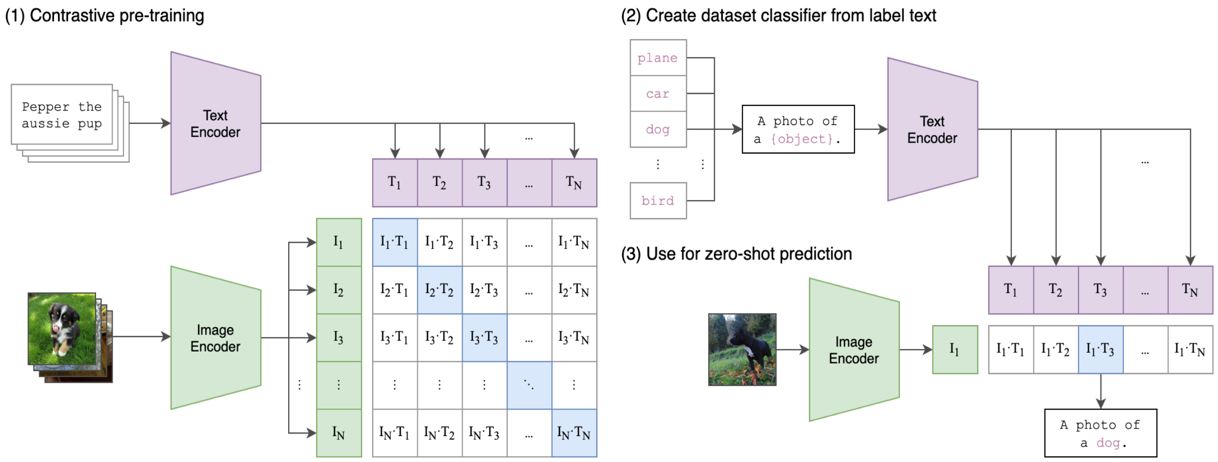
[10] The name comes from the fact that in the second step of our work we are dealing only with text data, meaning that we have text-input & text-output. A similar way of designating the model was used on Google's *T5* [107].

[11] We will see more about this type of models later in paragraph 3.2.3

feedable to the language model. In the original paper, the authors created two variants of the Clip-Cap model depending on whether the language generator was fine-tuned or not. It's worth noting that in both of these versions, the CLIP model was frozen (not fine-tuned). In the following paragraphs, we will elaborate more on these three sub-parts of Clip-Cap.

Image Encoder:

The image encoder is comprised by CLIP (Contrastive Language-Image Pretraining), which (as we saw) is a multimodal vision-language model that is trained using a contrastive learning approach. It is designed to understand images and text and judge if these can fit together in the form of captioning. During training, it tries to maximize the "cosine similarity" between correct image-caption vector pairs, and minimize the similarity scores between all incorrect pairs. During testing, it calculates the similarity scores between the vector of a single image with a bunch of possible caption vectors, and picks the caption with the highest cosine similarity. It should be underlined that CLIP is not a caption generation model; rather, it's primary capability lies in determining the compatibility between existing textual captions and corresponding images. The basic architecture of CLIP is shown in Fig. 15 and as you can observe it comprises of both image and text encoders for projecting these two modalities in the same embedding space (a $2-$dimensional matrix).



(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
T_1 T_2 T_3 ... T_N
Image Encoder
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
T_1 T_2 T_3 ... T_N
(3) Use for zero-shot prediction
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

Figure 15. Architecture of the CLIP model [104]

Language Decoder (Generator):

The text generator model can be any language model (LM) that is capable of generating text. In the original work the authors used the Generative Pretrained Transformer-2 or GPT-2 [106]. GPT-2 is an LM by OPEN AI and is the second in their GPT series of models. This model is pre-trained on BookCorpus, a dataset of over 7,000 self-published fiction books from a variety of genres. After this, GPT-2 was trained also on a dataset of 8 million web pages. In its final and largest form the model contains about 1.5 billion parameters. In short, the model processes input sequences in parallel through layers of self-attention mechanisms, capturing contextual relationships between tokens (words). In addition, it uses a mask-mechanism to make sure that the prediction for the token i, only uses the inputs from 1 to i but not the future tokens. This method of masking is called causal attention mechanism. During training, GPT-2 learns to predict the next token in a sequence based on the preceding context. In generation tasks, the model utilizes its learned knowledge to generate coherent language autoregressively.

Mapping Network:

The main functionality of the mapping network, is to translate the embeddings generated by CLIP and a learned constant to the GPT-2 input space. As previously mentioned according to [96], the two versions of Clip-Cap depend on fine-tuning or not of the language model (GPT-2). As a result, the authors used a different mapping network when GPT-2 was frozen and when it was not. In the first case, they deployed a simple Multi-Layer Perceptron (MLP), whilst in the second they utilized a more expressive transformer [130] architecture. The transformer enables global and self attention between input tokens while reducing the number of parameters for long sequences. In their work, the transformer is fed with two inputs, the visual encoding of CLIP and a learned constant input. This will enable the model to retrieve meaningful information from CLIP embeddings as well as to learn to adjust the fixed LM to the new data. An illustration of the mapping network, as a black-box, embedded on the whole of Clip-Cap's architecture is shown in Fig. 16, where we can see both CLIP (encoder), GPT-2 (decoder/generator) and the mapping network. Regarding the latter, it receives the images features extracted by CLIP, along with the constant term and projects them into the GPT-2 embedding space. The size in which CLIP vectorized the input image is called *prefix_size* while the size of the projected embedding is named *prefix_length*[12]. After this, the prefix embeddings are concatenated with language (input) embeddings and lastly are fed to GPT-2 for caption generation.

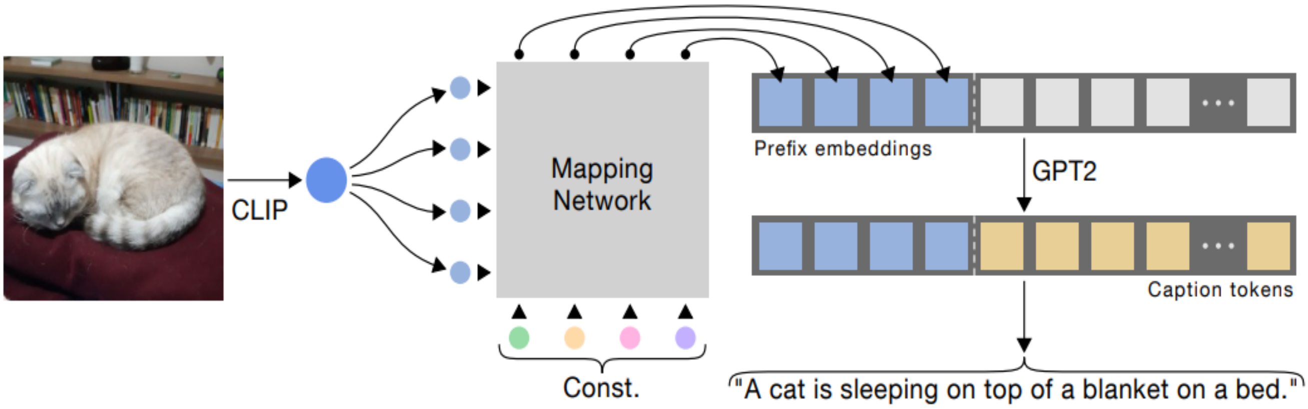


Figure 16. *Architecture of the Clip-Cap model [96]*

3.2.3 The Caption-to-Story architecture

Coming to the second part of our framework we will need a text generation model, that will be able to construct narrative stories from simple, isolated captions [58][13]. Our first and simpler choice, was be to reuse a pre-trained generative transformer model such as those introduced earlier in sub-section 2.4. Specifically, we opted to utilize two well-established transformer architectures, specialized on general text-to-text processing. These were *BART* [75] and the *T5* [107] models.

Bidirectional Auto-Regressive Transformer - BART:

Architecture-wise, *BART* is a denoising autoencoder transformer that maps a corrupted document to the original document it was derived from. It is implemented as a sequence-to-sequence model with a bidirectional encoder over corrupted text and a left-to-right autoregressive decoder. From that, it is easily concluded that the encoder of BART is similar to BERT's architecture [21] while it's generative type decoder shares high resemblance with GPT [105]. Coming to the training process of the model, BART as many other language models is pre-trained using several tricks and then can be fine-tuned for a wide range of tasks. Pretraining in BART is divided into two stages:

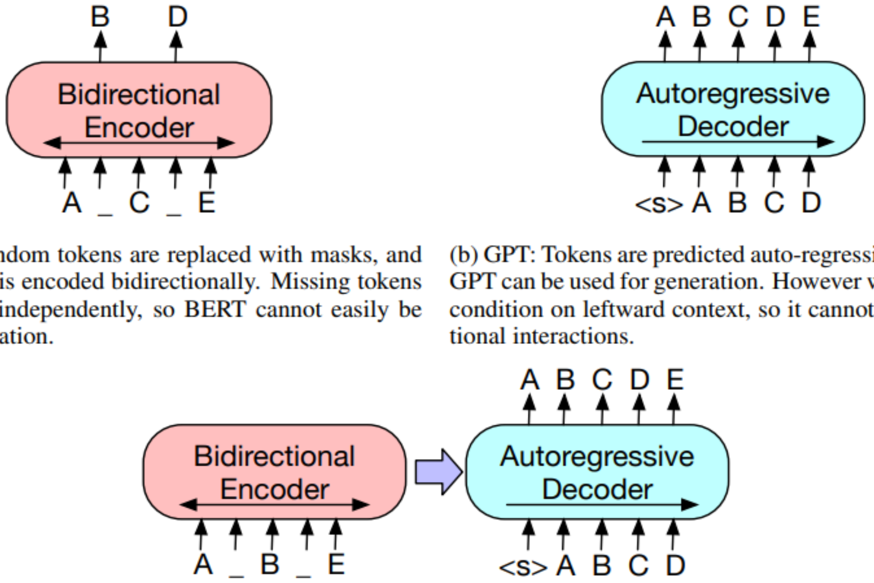
- Initially, text is corrupted with an arbitrary noising function. Methods of corruption include: Token Masking, Token Deletion, Document Rotation, Sentence Permutation and Text Infilling.
- Then, the sequence-to-sequence model (decoder) is learned to reconstruct the corrupted encoded input back to the original text. For this procedure, optimization of the negative log likelihood of the original document takes place.

Lastly for fine-tuning BART has been utilized for both classification tasks ("BERT-like tasks"), such as Sequence Classification & Token Classification, but also for generation tasks ("GPT-like tasks"), like Sequence

[12] In the case of the transformer mapper we have an additional parameter, *clip_length*, that is used for projection, in the ultimate linear layer in the top of the transformer.

[13] In contrast to *Jain et al.* [58] we are going to use a transformer-based model instead of sequence-to-sequence RNNs.

Generation or Machine Translation. An overview of how BART handles inputs sequences, during encoding and decoding, can be found in Fig. 17. As *Lewis et al.* emphasize, BART leverages both the advantages of bidirectional encoding and autoregressive generation techniques and as a result it can be fine-tuned for a greater variety of tasks than BERT or GPT. This comes quite handy in our case since we would like to utilize a model that has the capability of encoding the inputs (captions), but also of generating text.



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.

(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.

(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

Figure 17. *Schematic comparison between BART, BERT and GPT models [75]*

Text-to-Text Transfer Transformer - T5:

The *T5*, is another Encoder-Decoder model which closely follows the original transformer architecture by *Vaswani et al.* [130]. T5 is a versatile and powerful model designed to handle a wide range of natural language processing tasks by converting them into a text-to-text format. This means that all tasks, such as translation, summarization, and classification, are framed as text generation problems. Similarly, to BART, the T5 architecture uses an encoder that reads and processes the input text bidirectionally, and a decoder which generates the output text in an autoregressive manner, much like GPT. This dual approach enables T5 to handle diverse tasks by unifying them under a single framework, leveraging the strengths of both BERT-like bidirectional encoding and GPT-like generative decoding.

Probably the most crucial point on T5's pre-training is that it converts all text-based language problems into a text-to-text format. This procedure involves pre-training on a massive corpus (Colossal Clean Crawled Corpus - C4) of text using a unsupervised objective. As in the case of BART, this objective includes the corruption of the input text using different methods and the training of the model, under these circumstances, to reconstruct the original text. The corruption methods include:

- Span Corruption: Random spans of text are replaced with a unique sentinel token.
- Sentence Shuffling: The order of sentences in the input text is shuffled.
- Token Deletion: Random tokens are deleted from the input text.

The pre-training objectives are designed to enable the model to handle a wide variety of text-to-text tasks effectively. Span corruption, which is essentially a generalization of Mask Language Modelling, helps the model learn context-aware representations by focusing on the relationships between different (local) parts of the text. In addition, sentence-level manipulation such as shuffling or deletion, improve the model's ability not only understanding the local, token-level dependencies but also to handle tasks that require the broader context of the input sequence.

Finally, after pre-training, the T5 model can be fine-tuned on specific downstream tasks by continuing the training process on task-specific datasets. This includes both classification tasks and generation tasks. It is worth noting that the fine-tuning process for any of these tasks, should follow the same text-to-text framework in which T5 was exposed to, during pre-training. This is quite important, since it allows the model to switch between tasks without the need for any modification to it's own architecture, but only expanding the input with a prefix that determines the task that the model should make.

From-scratch Transformer:

Despite that *BART* and *T5* are considerably strong, the purpose of this project is also to explore in-depth the realm of transformer models and to gain more hands-on experience on how language models capable of generating meaningful text, operate. Therefore, during our experimentation, we attempted to build a model from-scratch that could ponder and would be able to create narrative storytellings. There are three main types of transformer architectures that can be used for natural language processing and specifically for language-to-language generation, like in our case:

- **Encoder-Only or Autoencoding Transformer:** This part of the transformer aims to capture meaningful representations of input data and squeeze them on an encoded form, which usually is a dense vector. Models of such type are particularly strong towards language understanding and classification tasks. The training process of these networks often occurs on a bidirectional fashion, which means that they consider both the forward and backward context of the input sequence and thus it captures dependencies within the text more effectively. Additionally, autoencoders employ masking techniques to deliberately hide or corrupt certain parts of the input during training. This procedure forces the model to acquire insights about robust features, rendering it with the ability of retrieving any missing information. This kind of models, include the *BERT family* [21], which became a mainstays in a plethora of classification and summarization-oriented tasks.

- **Decoder-Only or Autoregressive Transformer:** Autoregressive models, that rely solely on the decoder of the transformer, leverage probabilistic inference to predict the next token iteratively by depending also at one prior token. Unlike sequence-to-sequence models, autoregressive models don't require an explicit input sequence and are suitable for text generation tasks. Among the most popular autoregressive architectures, is the *GPT-series* [105], which have been immensely associated with tasks such as text generation, sequence-to-sequence modeling or even time series forecasting.

- **Encoder-Decoder Transformer:** Combines some strengths from both of the former categories. The encoder processes the input text, capturing any contextual information and the decoder generates sequential output based on the encoded information. This architecture is suitable for tasks, when understanding the input context and generation of a coherent sequence is needed and thus it's applicable for tasks like text-to-text generation. Two Encoder-Decoder transformers of high profile, are both the aforementioned *BART* [75] and *T5* models [107].

Due to the demand of generating a narrative story from a given input text (captions), we opted for the complete Encoder-Decoder architecture. In particular, in the *encoding phase*, the transformer's encoder will processes each caption independently, creating a high-dimensional representation for each of those. Using a *Self-Attention* mechanism we want to weigh the importance of different words in each caption, hoping that the model will capture the relationships between words and phrases.

In the *decoding stage*, the purpose is to generate the output sequence (the narrative story) autoregressively, one word at a time. As it has been employed numerously in prior works, the decoder should consider both the encoded captions and the words it has generated so far, to produce the next word in the sequence. A high-level illustration of an Encoder-Decoder Transformer that could be deployed for our mission is given in Fig. 18. It's worth noting that both *BART* and *T5* models follow closely the illustrated architecture.

Let's now brake down what is happening inside the transformer block. Regarding the Encoder, the input sequence is first passed through an embedding layer which converts each token into a dense vector representation. Assuming that $\mathbf{X}$[14] is the input sequence and $\mathbf{E}$ is the embedding matrix then embedded input can be expressed as:

$$\mathbf{X}_{\text{emb}} = \mathbf{X} \cdot \mathbf{E} \tag{1}$$

Since the Transformer doesn't have any recurrence, positional encodings are added to the embeddings to provide information about the position of each token in the sequence. Let $\mathbf{PE}$ be the positional encoding matrix, then the input to the first encoder layer will be:

$$\mathbf{H_0} = \mathbf{X}_{\text{emb}} + \mathbf{PE} \tag{2}$$

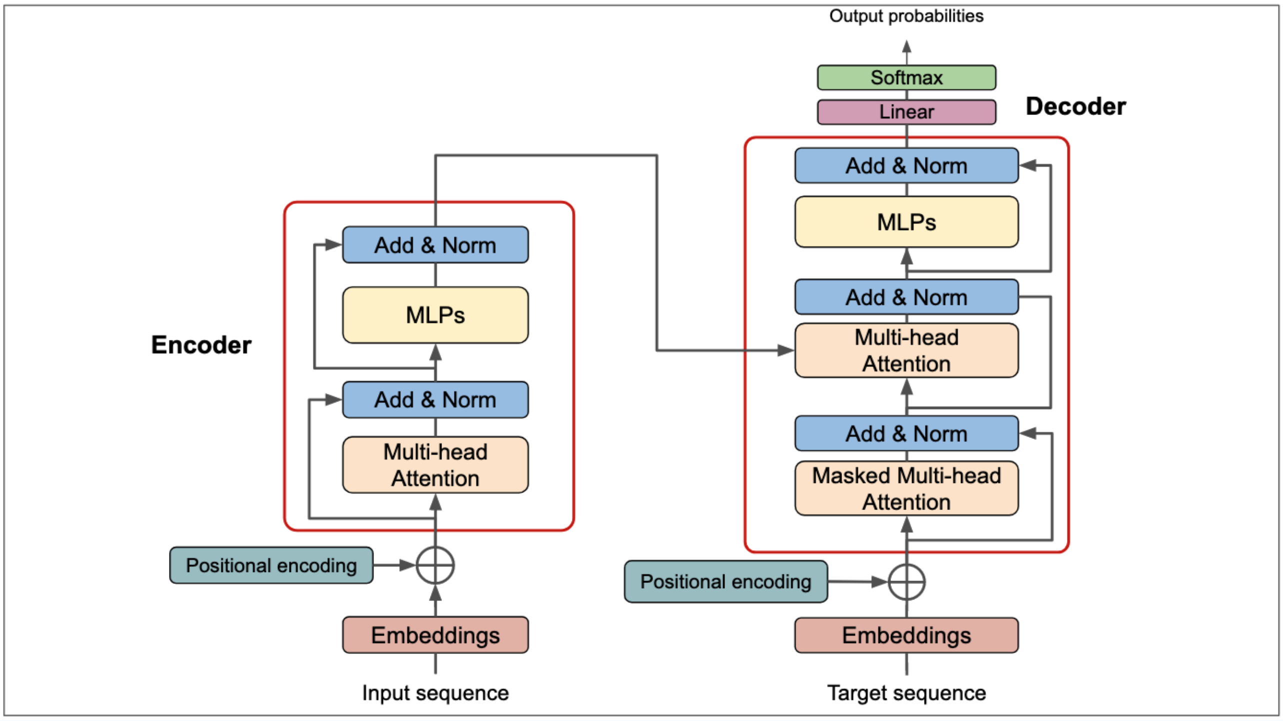


Figure 18. *The Transformer Architecture [98] (adapted from [130])*

At this step, the enhanced input sequences (with the positional encodings), passes through a multi-head self-attention mechanism. For each head, the attention scores are calculated as follows:

$$\text{Attention}_i(\mathbf{Q_i}, \mathbf{K_i}, \mathbf{V_i}) = \text{softmax}\left(\frac{\mathbf{Q_i}\mathbf{K_i}^T}{\sqrt{d_k}}\right)\mathbf{V_i} \tag{3}$$

Here: $\mathbf{Q_i} = \mathbf{H_0}\mathbf{W_i}^Q$, $\mathbf{K_i} = \mathbf{H_0}\mathbf{W_i}^K$ and $\mathbf{V_i} = \mathbf{H_0}\mathbf{W_i}^V$ are the queries, keys, and values for the i^{th} head and d_k is the dimension of the keys. Moreover, $\mathbf{W_i}^Q$, $\mathbf{W_i}^K$, and $\mathbf{W_i}^V$ are learned weight matrices used to transform the input embeddings into queries, keys, and values respectively. In the end of the attention block, the outputs of all the attention heads are concatenated via:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)\mathbf{W^O} \tag{4}$$

where $\mathbf{W}^O$ is another learned weight matrix used to project the concatenated outputs back to the original dimension. After this, the current output is added with the attention layer's input through a residual connection and then the result is normalized:

$$\mathbf{H}_1 = \text{LayerNorm}(\mathbf{H}_0 + \text{MultiHeadAttention}(\mathbf{H}_0)) \tag{5}$$

[14] We can assume that the input sequence has size of $(bs, |\mathbf{X}|)$, where bs is the batch size and $|\mathbf{X}|$ is the maximum length of the input sequence in the batch.

Now, each position (x) of the sequence is processed by a feed-forward network (FFN), consisting of two linear transformations with a ReLU as activation function. Lastly, the FFN output is also added with its own input through a residual connection and then passes a normalization layer. These are formulated below:

$$FFN(x) = \max(0, x\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2 \tag{6a}$$

$$\mathbf{H_2} = \text{LayerNorm}(\mathbf{H_1} + FFN(\mathbf{H_1})) \tag{6b}$$

$\mathbf{H_2}$ symbolizes the encoded input that it will be given to the next encoder blocks (in case of many blocks) or it will consist the *memory* input of the Decoder (in case of one block).

Coming to the latter, it's pretty much regulated by similar equations. Initially, the decoder embeddings are drawn out from the target sequence ($\mathbf{Y}$)[15] with the usage of the embedding layers ($\mathbf{E}$) and the positional encodings ($\mathbf{PE}$) are added to these target embeddings, equally to the Encoder:

$$\mathbf{Y}_{\text{emb}} = \mathbf{Y} \cdot \mathbf{E} \tag{7a}$$

$$\mathbf{H'_0} = \mathbf{Y}_{\text{emb}} + \mathbf{PE} \tag{7b}$$

In the attention part, the decoder uses masked multi-head self-attention to prevent attending to future tokens. The mask ensures that the prediction for a token only depends on known-previous outputs:

$$\text{MaskedAttention}_i(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = \text{softmax}\left(\frac{\mathbf{Q}_i\mathbf{K}_i^T}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V}_i \tag{8}$$

Here, the mask matrix $\mathbf{M}$ is applied to enforce the autoregressive property. Like in the case of the encoder, the output of the masked multi-head attention, after the concatenation of the heads, is followed by a residual connection for addition and layer normalization:

$$\mathbf{H'_1} = \text{LayerNorm}(\mathbf{H'_0} + \text{MaskedMultiHeadAttention}(\mathbf{H'_0})) \tag{9}$$

Subsequently, the decoder performs multi-head attention over the encoder's output (*memory*) and its own previous modules' output. This allows the decoder to focus on relevant parts of the input sequence and it's mathematically performed as below:

$$\text{Attention}_i(\mathbf{Q_i}, \mathbf{K_i}, \mathbf{V_i}) = \text{softmax}\left(\frac{\mathbf{Q_i}\mathbf{K_i}^T}{\sqrt{d_k}}\right)\mathbf{V_i} \tag{10}$$

Howbeit, this time, only the query is obtained by the previous decoder's outputs as: $\mathbf{Q_i} = \mathbf{H'_1}\mathbf{W_i}^Q$, whilst the key and the value comprise the incoming information (memory) coming from the encoder as: $\mathbf{K_i} = \mathbf{H_2}\mathbf{W_i}^K$ and $\mathbf{V_i} = \mathbf{H_2}\mathbf{W_i}^V$. Following the pattern of the encoder, this output is also followed by a residual connection and layer normalization, which in turn are followed by an FFN that processes each position of the current sequence. Finally, another residual connection and layer normalization take place:

$$\mathbf{H'_2} = \text{LayerNorm}(\mathbf{H'_1} + \text{MultiHeadAttention}(\mathbf{H'_1}, \mathbf{H_2}, \mathbf{H_2})) \tag{11a}$$

$$\mathbf{H_3} = \text{LayerNorm}(\mathbf{H'_2} + FFN(\mathbf{H'_2})) \tag{11b}$$

The very last part of the transformer block is the so-called output layer which essentially is a linear transformation followed by a softmax activation to produce the probabilities given by the model ($\mathbf{P}$) over the vocabulary (V) for the next token prediction as:

$$\mathbf{P} = \text{softmax}(\mathbf{H'_3}\mathbf{W_o} + b_o) \tag{12}$$

Considering the two kind of inputs and the above transformer-pass, from equation (12) we can derive that $\mathbf{P} \in \mathbb{R}^{bs \times |\mathbf{Y}| \times |V|}$, where $|V|$ is the size of the output vocabulary. Our ultimate purpose is that during generation, the model should pick the most probable token out of the total $|V|$ tokens.

[15] We can assume that the target sequence has size of $(bs, |\mathbf{Y}|)$, where bs is the batch size and $|\mathbf{Y}|$ is the maximum length of the target sequence in the batch.

3.3 Training Techniques for the second phase

In our earlier discussion, we mentioned that for the second phase of our experiment we used different *text-to-text* (transformer) models, including an architecture that was rigged from-scratch. As shown in the overview of Fig. 14, two out of the three models used for this phase were already pre-trained (T5 & BART). To that end, these models should only be fine-tuned to our task using normal supervised training techniques. Conversely, for our from-scratch model utilizing only supervised training would result to a tremendously under-trained in comparison to the previous two models, since in general a language model needs vast amount of training data and dozens of hours of pre-training to reciprocate decently on text generation tasks. For instance, as we already saw, both T5 and BART were pre-trained, by using integrated processes of document corruption. Inspired by these pre-training steps, we applied a small subset of those for our T4 model as well, staying of course, within the boarders of VIST dataset. However, the ultimate fine-tuning step for generating stories, was always the normal *supervised training* for all three of the models.

3.3.1 Supervised Training

In our case, *supervised training* is essentially comprised by a *direct sequence-to-sequence training* where the pairs of captions and stories (input & target sequences) were fed to the model which in turn gives out the predicted output sequence at once.

Direct Sequence-to-Sequence Training:

Sequence-to-sequence training for our task can be achieved by simply passing the input sequence (captions) and the target sequence (story) to the model at every batch iteration. More specifically, as it can be distinguished back on Fig. 14, the five captions[16] are concatenated in one mini-paragraph by using the *separation* or *end-of-sequence* token of the model's tokenizer as the connection link of the sentences. This concatenated paragraph is then fed to the encoder of the models. One the other hand, the five-sentenced story remains intact and is passed to the decoder of each transformer.

A visualization of the form of these two inputs and how the feeding to a transformer-based model is done, is given on Fig. 19. Since our model (T4) used a BERT-like tokenizer the end-of-sequence token is "[SEP]"[17], while for BART and T5 this token becomes "<\s>". Ultimately, when the *Output Sequence* is retrieved from the last block of the decoder, it is compared to the *Target Story*, using the appropriate *Loss Function*, which in all cases was *Cross-Entropy Loss*.

An algorithmic pseudo-code of the direct sequence-to-sequence training is shown on Algorithm 1. For this purpose, we consider a train loader $\mathcal{TL}$, a model $\mathcal{M}$, a hypothetical function that untangles the captions and stories from each batch named "unpack_batch" and the Cross-Entropy loss function $\mathcal{LF}$. Moreover, with $O_s \in \mathbb{R}^{bs \times |\mathbf{Y}| \times |V|}$, where $|V|$ is the vocabulary size, bs the batch size and $|\mathbf{Y}|$ the length of target sequence, we symbolize the output of the model at each batch iteration. Note, that in the loss function the whole output sequence and the whole stories are compared. The notation "rest of optimization steps", essentially involves the gradient updates and loss accumulation.

Algorithm 1 Sequence-to-Sequence Training Pseudo-code

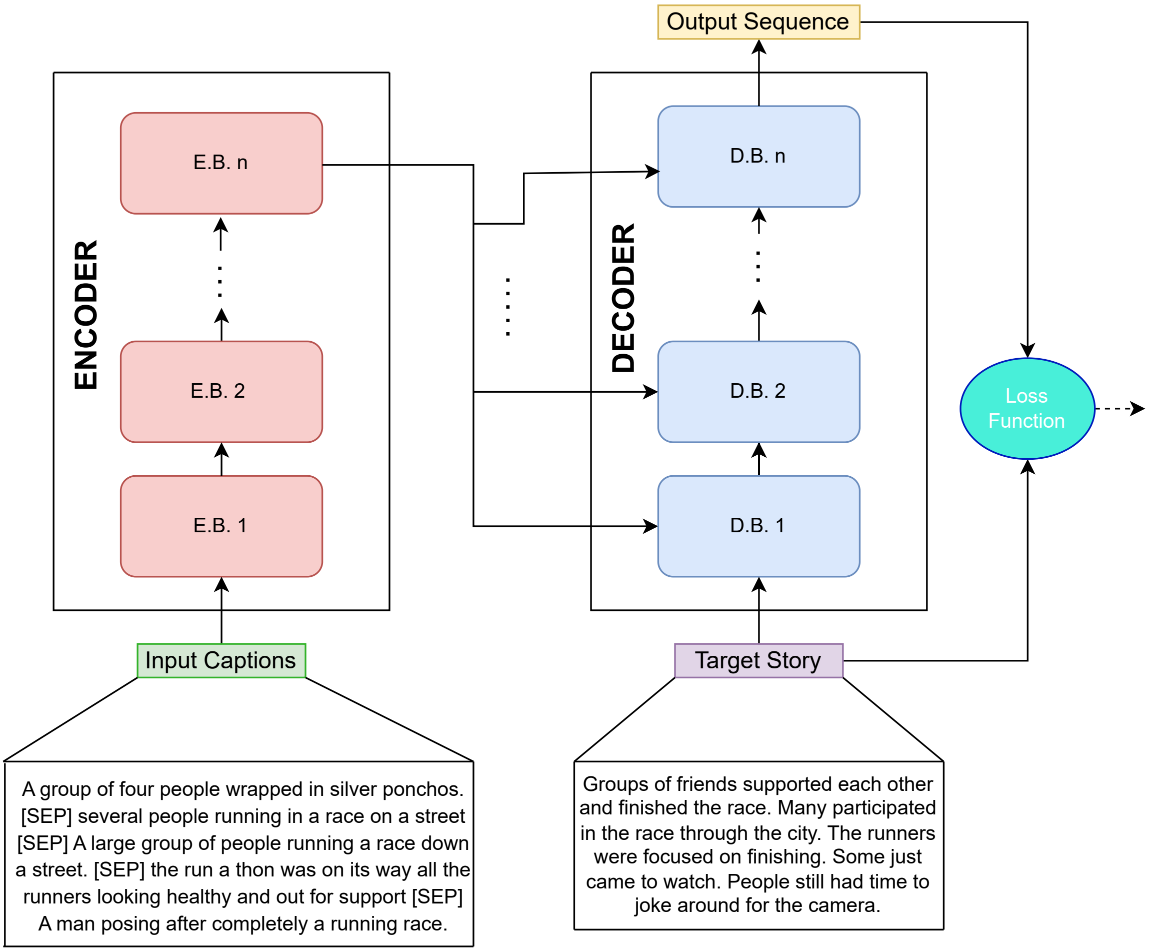
```
for $epoch = 1$ to $num\_epochs$ do
    for $batch$ in $\mathcal{TL}$ do
        $caps, stories \leftarrow unpack\_batch(batch)$
        $O_s \leftarrow \mathcal{M}(caps, stories)$
        $loss \leftarrow \mathcal{LF}(O_s, stories)$
        "rest of optimization steps"
    end for
end for
```

[16] coming either from the first stage, if we are on the prediction phase or from the dataset itself if we are in fine-tuning phase

[17] We will see later that since we used BERT tokenizer to feed our model, it's input vocabulary would be of size: $|V| = 30522$.

3.3.2 Training Methods Using Document Corruption

From all the pre-training procedures of BART and T5 involving document corruption, we opted for *Mask Language Modeling (MLM)* and *Sentence Permutation (SP)* to enhance T4's ability to grasp semantic notions and syntactic structures met in natural language. Nonetheless, it was necessitated due to different nature of visual storytelling task (from isolated captions), that these techniques are properly adjusted to our needs with some minor changes. Definitively, after these stages, we performed *Supervised Training* to lead the model towards our specific task of story generation given the isolated captions.



A group of four people wrapped in silver ponchos. [SEP] several people running in a race on a street [SEP] A large group of people running a race down a street. [SEP] the run a thon was on its way all the runners looking healthy and out for support [SEP] A man posing after completely a running race.

Groups of friends supported each other and finished the race. Many participated in the race through the city. The runners were focused on finishing. Some just came to watch. People still had time to joke around for the camera.

Figure 19. *An overview of how our models are fed with the (captions, stories) pairs during sequence-to-sequence training*

Mask Language Modeling:

Mask Language Modeling (MLM) is a technique where some percentage of the input tokens are masked at random, and the model is trained to predict these masked tokens. It is used particularly for pre-training on some BERT-like architectures. MLM owns numerous advantages that can boost a model's capabilities both in Natural Language Understanding (NLU) and Natural Language Generation (NLG). Some of those are reported below:

- **Contextual Understanding**: When training with MLM, models attempt to predict missing words in a sentence based on the surrounding context. This encourages the model to develop a deep comprehension of the context in which words appear and the order that they should have in order to form meaningful sentences. This leads to better performance on various language tasks such as question answering, sentiment analysis or named entity recognition.

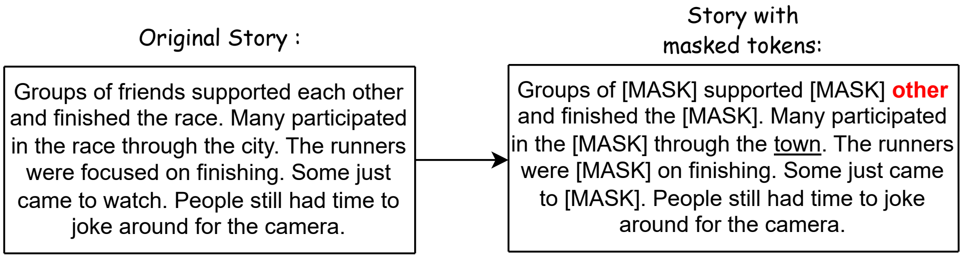
- **Bidirectional Training**: Unlike the traditional language models which are only left-to-right or right-to-left trained, MLM allows for processing bidirectional context. This means the model takes into account both the words before and after the masked token, leading to more accurate predictions and richer contextual representations. Consequently, this helps the model to learn abstracted textual features that capture the meanings of words in various contexts, leading to better performance on tasks that require nuanced understanding of language.

- **Robustness & Flexibility**: By randomly masking words and training the model to predict them, MLM introduces variability and forces the model to learn more complete representations of language that generalize well across different contexts and tasks. Along with it's bidirectional nature, MLM allows the model to be fine-tuned for both classification and generation tasks, making it versatile for various real-time applications.

For our training pipeline, MLM was the first technique to be applied on the T4 model. It's worth underlining that this method was only used on the target stories (the decoder inputs from Fig. 19), as an adaptation to our visual storytelling objective. On those sequences, we select some tokens to be masked based on a *Bernoulli* distribution with probability of 0.15. Notwithstanding, not all selected tokens are replaced with the "[MASK]" token. More precisely, only 80% are turned to actual masked tokens, while the rest 10% is remained unchanged and the last 10% is substituted with other random tokens.

In this way, we eager to boost the generalization abilities of the model as well as to increase it's robustness. Avoiding 100% replacement by the "[MASK]", significantly reduces overfitting. On the contrary, introducing a slight noise, by using some random tokens, and keeping a fraction of the original context with some unchanged tokens, both can aid the model to learn more contextually rich and diverse representations but at the same time not to lose all the original information around the masked tokens. Last but not least, we have to note that during the loss calculation with MLM the actual target didn't change. This means that the noisy[18] model's output is contrasted with the original stories, like in the case of the supervised training. A schematic depiction of how the target story is altered for this purpose is rendered on Fig. 20.

Despite having numerous advantages, MLM is considered to be a complex and a computationally intensive technique. MLM training requires masking a portion of the input tokens and then predicting them, something that adds extra complexity and computational load to the training pipeline compared to a simpler and unidirectional training of language models. Things turn to be even more ominous, when we attempt to incorporate it, into the already complicated task of visual storytelling.



Groups of friends supported each other and finished the race. Many participated in the race through the city. The runners were focused on finishing. Some just came to watch. People still had time to joke around for the camera.

Groups of [MASK] supported [MASK] **other** and finished the [MASK]. Many participated in the [MASK] through the town. The runners were [MASK] on finishing. Some just came to [MASK]. People still had time to joke around for the camera.

Figure 20. *A representation of how stories are masked for the purpose of MLM. On the left, the original story. On the right, the story with 7 candidate masked tokens. In this case 5 tokens: "friends", "each", "race", "focused" and "watch" are replaced with the actual "[MASK]" token, while the red font word "other" is the one that remains unaltered. The underlined word "town" is the one that substituted the original token "city".*

Sentence Permutation:

Sentence permutation is a document corruption technique where the sentences within a document are shuffled randomly. This method is primarily used in pre-training language models to improve their apprehension of document-level coherence and sentence-level context. Like in the case of MLM, sentence permutation is an important methodology in pre-training language models, that can lead to the development of models with enhanced contextual understanding and robustness. Some advantages of sentence permutation encapsulate:

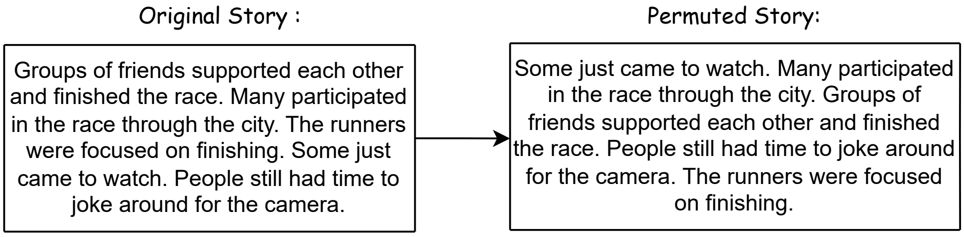
- **Enhanced Understanding of Context**: Sentence permutation helps models learn better context management and understanding. By encountering sentences in a shuffled order, the model is trained to infer the correct sequence and relationships between sentences. This for example, assists for capturing more long-range dependencies and the overall document structure.

- **Coherence, Cohesion & Robustness**: It enhances the model's ability to recognize coherent and cohesive text by learning to reorder sentences logically. Additionally, models become more resilient to variations in text and as result more robust to linguistic differentiations.

[18] due to the noisy/masked input on the decoder

- **Versatility**: Pre-training with sentence permutation can make models more adaptable to a variety of NLP tasks that require understanding the flow and structure of text, such as summarization, question-answering and translation.

For our specific training occasion, sentence permutation was applied quite similarly to MLM. Therefore, the input captions remained untouched once again, and all diversifications were made upon the target stories (decoder's input). What is more, optimization and loss calculation were granted precisely as in the cases of MLM and supervised training. To that end, we had a comparison between the model's noisy output[19] and the ground true stories anew. As we have already mentioned, in the final version of the dataset the stories comprised of five sentences. Hence, for permuting the stories, we simply changed randomly the order of those sentences. This transformation of the stories can be seen on Fig. 21.



Groups of friends supported each other and finished the race. Many participated in the race through the city. The runners were focused on finishing. Some just came to watch. People still had time to joke around for the camera.

Some just came to watch. Many participated in the race through the city. Groups of friends supported each other and finished the race. People still had time to joke around for the camera. The runners were focused on finishing.

Figure 21. *A representation of how stories are altered for the purpose of Sentence Permutation technique. On the left, the original five-sentences story. On the right, the permuted story, where all five sentences have been randomly shuffled.*

Sentence permutation is a very promising method, but still raises the prospect of introducing high levels of undetermined noise. This is not always beneficial for all tasks, particularly for those that rely heavily on the original sequence of sentences and can result to the underfitting of the model and eventually poorer results. For these reasons, it needs to be managed with great caution during the training process. Despite this phenomenal drawback, it would be intriguing to check if it could help our models for visual storytelling.

3.4 Decoding strategies

In the terms of Natural Language Processing, machine learning systems like *Neural Networks* such as LSTMs, or even transformer-based language models, in order to generate text, use a method named *Autoregressive Generation*. In broad terms, this method of generating text imposes to the model, $\mathcal{M}$, to produce, in an iterative manner, one token at a time, till the generation reaches the *end-of-sentence* token ($[eos]$). The model is initialized with the token that indicates the *start-of-sentence* ($[sos]$).

More formally, assuming an input a sequence $\mathbf{X}$ and ground true target sequence $\mathbf{Y}$, we would like to approach $\mathbf{Y}$ with the generated sequence by $\mathcal{M}$, $\hat{y}$. Furthermore, we symbolize $\mathcal{M}$'s output sequence with $O_s \in \mathbb{R}^{bs \times |\mathbf{Y}| \times |V|}$, where $|V|$ is the size of the utilized vocabulary, bs the batch size and $|\mathbf{Y}|$ the length of the target sequence. Algorithm 2, describes how autoregressive generation works. The main idea is to initialize the generated sequence, $\hat{y}_l$ and iteratively pass it (along with the input $\mathbf{X}$) through the model, where $l-1$ is the final length of the generated sequence.

At this point, using *Softmax* and a *Decoding Strategy*, $\mathcal{DS}$, the algorithm generates the next token (t_i) and concatenates it with existing predicted sequence. Additionally, the algorithm accepts an input integer max_length, which determines the maximum number of tokens (words) that we allow the model, $\mathcal{M}$, to generate[20]. Lastly, we should notice that in line 6 of the algorithm the logits are stored only for the last generated token, whilst the returned sequence (in line 14) excludes the $[sos]$ token.

While revising Algorithm 2, someone will plausibly wonder what exactly is the decoding strategy $\mathcal{DS}$ that is reported. In the following, paragraphs we will answer this, by introducing four well-known decoding strategies which are used across the Bibliography. These are *Greedy decoding*, *Multinomial sampling*, *Nucleus sampling* or *p-sampling* and *Beam search*.

[19] since the permuted stories to the input of the decoder cause noise

[20] Alternatively, the max_length could also set in relation to the input sequence $\mathbf{X}$ as $max\_length \leftarrow \alpha \times |\mathbf{X}|$, where α is a positive constant or even according to the target sequence as $max\_length \leftarrow |\mathbf{Y}|$.

Algorithm 2 Sequence Generation Algorithm

```
Input: $\mathbf{X}$, $[sos]$, $[eos]$ , $max\_length$
Output: Generated sequence $\hat{y}_l$ by $\mathcal{M}$
Initialize $\hat{y}_0 \leftarrow [sos]$
for $i = 1$ to $max\_length$ do
    Forward pass through the model: $O_{s_i} \leftarrow \mathcal{M}(\mathbf{X}, \hat{y}_{i-1})$
    Get logits for the last generated token: $L_t^i \leftarrow O_{s_{-1}}$
    Compute probabilities for the token over $V$: $P_t^i \leftarrow softmax(L_t^i)$
    Get the most likely next token: $t_i \leftarrow \mathcal{DS}(P_t^i)$
    if $t_i = [eos]$ then
        break
    end if
    Append the predicted token to the sequence: $\hat{y}_i \leftarrow \hat{y}_{i-1} \parallel t_i$
end for
return $\widehat{y_{l_{1:}}}$
```

3.4.1 Greedy decoding

Greedy decoding or *greedy search* [122] is a simple and commonly used strategy for generating sequences from probabilistic models, such as neural networks used in NLP. In this approach, at each time step i, the token with the highest conditional probability is selected as the next token in the sequence. This process is repeated until a stopping criterion is met, typically when the $[eos]$ token is generated or the max_length of the sequence is reached. Formally, let $P(t_i|t_{<i}, x)$ represent the probability of token t_i at time step i given the sequence of previously generated tokens $t_{<i}$ and the input sequence x. In greedy decoding, the strategy $\mathcal{DS}$ for choosing the next token t_i can be defined as:

$$\mathcal{DS}(P_{t_i}) : t_i = \arg\max_{v \in V} P(t_i = v \mid t_{<i}, x), \tag{13}$$

where V is the vocabulary of possible tokens. This selection maximizes the probability of the next token given the current context. Greedy decoding is an efficient and straightforward way but may not always produce the most optimal sequence, because it is not considering future potential tokens and their probabilities. It is prone to getting stuck in local optima, which can result in suboptimal sequences compared to more sophisticated decoding strategies like beam search which we will see later.

3.4.2 Multinomial sampling

Multinomial sampling [8,134] is another probabilistic decoding strategy used in text generation. Instead of always selecting the token with the highest probability (like in greedy search), multinomial sampling selects the next token based on a specific probability distribution over the vocabulary V. In this case, given the conditional probability distribution $P(t_i|t_{<i}, x)$ at each time step i, the next token t_i is sampled according the decoding strategy, $\mathcal{DS}$, below:

- Compute the probability distribution over the vocabulary for the next token:

$$P(t_i = v \mid t_{<i}, x), \quad \forall v \in V \tag{14}$$

- Sample the next token t_i following a multinomial distribution[21]:

$$t_i \sim Multinomial(P(t_i \mid t_{<i}, x)) \tag{15}$$

[21] The multinomial distribution gives the probability of observing a particular combination of outcomes over n trials. Each of these trials can result in one of k possible outcomes. If we denote the number of times each outcome occurs by $x_1, x_2, \ldots, x_k$ and and the corresponding probabilities of each outcome by $p_1, p_2, \ldots, p_k$, where $\sum_{i=1}^{k} p_i = 1$ and $\sum_{i=1}^{k} x_i = n$, then the probability mass function of the multinomial distribution is: $P(X_1 = x_1, \ldots, X_k = x_k) = \frac{n!}{x_1! \cdots x_k!} \prod_{i=1}^{k} p_i^{x_i}$

This means that for each possible token $v \in V$, the probability that the token t_i takes the value v is given by $P(t_i = v \mid t_{<i}, x)$. This probabilistic approach allows for more diverse and varied text generation compared to greedy decoding, as it introduces randomness into the process by sampling from the entire probability distribution rather than always choosing the most likely token. On the other hand, due to its randomness it can account for less coherent or syntactically structured text.

3.4.3 Nucleus sampling (p-sampling)

Nucleus sampling, also known as *top-p sampling* [48], is a decoding strategy that selects tokens from the smallest possible subset of the vocabulary, such that the cumulative probability of this subset reaches at least p. This method ensures that the sampling process focuses on the most likely tokens while still allowing for diversity in the generated sentences.

We can formulate nucleus sampling decoding strategy ($\mathcal{DS}$) by firstly sorting the token probabilities $P(t_i \mid t_{<i}, x)$ in descending order at each time step i. If v_k is the token with the k^{th} highest probability then we can define the cumulative probability C_k as :

$$C_k = \sum_{j=1}^{k} P(v_j|t_{<i}, x) \tag{16}$$

We can opt for the smallest integer m, so as $C_m \geq p$. In this occasion the nucleus N_p is the set of tokens $\{v_1, v_2, ..., v_m\}$. The $\mathcal{DS}$ now imposes that he next token t_i is sampled from the truncated distribution over the nucleus, which essentially is translated that now we sample by a multinomial distribution given a conditional probability according to the formula below:

$$t_i \sim Multinomial(P(t_i \mid t_{<i}, x) \mid t_i \in N_p) \tag{17}$$

Focusing on the top-p subset, nucleus sampling balances between deterministic and stochastic generation, offering a flexible trade-off between syntactic quality and linguistic diversity.

3.4.4 Beam search

Beam search [30, 38, 122] is an heuristic search algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is a modification of *best-first search* algorithm that reduces its memory requirements and the total execution time. It is commonly used in tasks, such as text generation and machine translation, to find the most likely sequence of words. Unlike greedy search, which selects the best candidate at each step, beam search keeps track of multiple hypotheses (or beams) simultaneously, which allows it to explore a larger search space.

Formally, recalling that $\mathbf{X}$ is the input sequence and $\hat{y}_l$ is the target sequence to be generated ($l - 1$ is the final length of the sequence), then the goal of beam search is to find the best sequence $\hat{y}_l$ that maximizes the conditional probability $P(\hat{y}_l|\mathbf{X})$. During the expansion of the graph, at each time step i, the algorithm maintains only a set of the top k hypotheses (beams), $\hat{y}_i$, based on their scores. Here, k is a positive integer called *beam width*. By denoting with $\hat{y}_{i-1}^{j}$ the currently generated sequence of the j^{th} beam at time step $i - 1$ and with $P(\hat{y}_i^j|\mathcal{M}(\mathbf{X}, \hat{y}_{i-1}^{j}))$ the corresponding probability for the expansion of the j^{th} beam, we can alter Algorithm 2 so to include all possible expansions of the k beams as shown in Algorithm 3.

The latter, shares numerous similarities with the former, yet it exhibits increased complexity. Initially, we create a set of beams, B, consisting only of the $y_0 = [sos]$ and a starting score of 0. The main difference comes when for each time step i, we also traverse all the k beams in B by firstly initializing the candidate sequences ($cand$). Taking the already stored hypotheses y_{i-1}^{j} in B, we compute the probabilities for each of those by using the input sequence $\mathbf{X}$ and the model $\mathcal{M}$.

Now, the algorithm goes one step further by computing the scores for each single token t_i in the vocabulary V and by creating a total of $|V|$ new hypotheses, y_i^j. After that, these hypotheses are stored as candidates in the $cand$ list. By the end of loop at line 15, $k \times |V|$ candidates sequences have been created[22].

[22] except of the case $i = 1$, where the number of beams are 1 due to its initialization in line 3.

Out of those, the algorithm chooses only k candidates based on their accumulative score (given by $\mathcal{M}$) as the potential sequences that proceed to the next generation step. Ultimately, the algorithm returns the hypothesis in B with the highest final score, excluding the start-of-sequence token.

To sum up, this algorithm ensures that we can keep track of multiple hypotheses at each time step, allowing for a more thorough exploration of the search space compared to greedy search and thus contributing to a more diverse text generation. Nevertheless, while beam search can be computationally feasible for low values of the beam width (e.g 2 to 5), it is obvious that by increasing k, the algorithm shows greater complexity and becomes extremely computationally expensive.

Algorithm 3 Beam Search Algorithm

```
**Input:** $\mathbf{X}$, $[sos]$, $[eos]$, $max\_length$, $k$
**Output:** Generated sequence $\hat{y}_l$ by $\mathcal{M}$
Initialize the set of beams as (sequence= $\hat{y}_0$, score): $B \leftarrow [([sos], 0)]$
**for** $i = 1$ to $max\_length$ **do**
    Initialize candidates: $cand \leftarrow []$
    **for** each $(\hat{y}^j_{i-1}$, score) in $B$ **do**
        Forward pass through the model: $O_{s_i} \leftarrow \mathcal{M}(\mathbf{X}, \hat{y}^j_{i-1})$
        Get logits for the last generated token: $L^i_t \leftarrow O_{s_{-1}}$
        Compute probabilities for the token over $V$: $P^i_t \leftarrow softmax(L^i_t)$
        **for** each token $t_i \in V$ **do**
            Compute new score: new_score $\leftarrow$ score $+ \log P^i_t[t_i]$
            Form a new hypothesis: $\hat{y}^j_i \leftarrow \hat{y}^j_{i-1} \parallel t_i$
            Add new hypothesis to candidates: $cand \leftarrow cand \cup (\hat{y}^j_i$, new_score)
        **end for**
    **end for**
    Select the top $k$ hypotheses from candidates: $B \leftarrow \text{top}_k(cand)$
    **if** any $\hat{y}^j_i \in B$ ends with $[eos]$ **then**
        **break**
    **end if**
**end for**
**return** $\hat{y}_l = \arg\max_{y^j \in B} \{\text{score}(y^j)\}_{1:}$
```

3.5 Evaluation Metrics

The last step of this project, as already mentioned during our research questions (sub-section 1.4), will be the evaluation of the results. The assessment will take place in three phases and with two different ways. To begin with, we will evaluate the results of the model(s) in both the caption generation stage (phase 1) and in the story generation part (phase 2), via some automatic metrics like METEOR [9], BLUE [99], SPICE [5], CIDEr [131] and SPIDEr [88] (1^{st} way of evaluation).

Combining the best performances from these two phases, will drive us to the third and *ultimate phase*, where the whole pipeline of Fig. 14 is going to be evaluated end-to-end, but only during test time. For this experiment, evaluation with automatic metrics will be recruited once more. Albeit, in order to ensure more robustness and pluralism in our final results and conclusions, human evaluation will also take place in the form of crowdsourcing (2^{nd} way of evaluation).

3.5.1 Automatic metrics

For phase 1, our goal is to automatically evaluate for an image I_i the quality of a candidate caption c_i given a set of reference captions $Si = (s_{i1}, ..., s_{im}) \in S$[23]. For VIST dataset $m = 3$, which means that per image we have three different captions. The caption sentences are represented using sets called *n-grams*.

[23] With S we denote the total set of reference captions present in the dataset.

Each *n-gram*, ω_k, is defined as a set of one or more ordered words and $\omega_k \in \Omega$, where the latter is the vocabulary of all *n-grams*. As many other works, we explore *n-grams* with length from one to four words. No stemming or lemmatization is performed on the sentences. The number of times an n-gram ω_k occurs in a reference sentence s_{ij} is denoted as $h_k(s_{ij})$, while for the candidate sentence $c_i \in C$[24] is denoted with $h_k(c_i)$.

BLEU:

BLEU [99] is a popular machine translation metric that analyzes the co-occurrences of *n-grams* between the candidate and reference sentences. It computes a corpus-level clipped n-gram precision between the two kind of sentences as follows:

$$CP_n(C,S) = \frac{\sum_i \sum_k \min\left(h_k(c_i), \max_{j \in m} h_k(s_{ij})\right)}{\sum_i \sum_k h_k(c_i)}, \tag{18}$$

where k indexes the set of possible *n-grams* of length n in a sentence. The clipped precision metric limits the number of times an *n-gram* may be counted to the maximum number of times it is observed in a single reference sentence. In the way that CP_n precision score is defined, it favors short sentences. For that reason, a brevity penalty is also used:

$$b(C,S) = \begin{cases} 1 & \text{if } l_C > l_S \\ e^{1-\frac{l_S}{l_C}} & \text{if } l_C \leq l_S \end{cases} \tag{19}$$

where l_C is the total length of candidate sentences c_i's and l_S is the length of the corpus-level effective reference length. When there are multiple references for a candidate sentence, we choose to use the *closest* reference length for the brevity penalty. Therefore, the overall BLEU score is computed using a weighted geometric mean of the individual *n-gram* precision:

$$BLEU_N(C,S) = b(C,S)exp\left(\sum_{i=1}^{N} w_n log\{CP_n(C,S)\}\right), \tag{20}$$

where $N = 1, 2, 3, 4$ and w_n is a weight that typically is held constant for all n.

METEOR:

METEOR [9] is calculated by generating an alignment between the words in the candidate and reference sentences, with the aim of $1:1$ correspondence. This alignment is computed while minimizing the number of chunks, ch, of contiguous and identically ordered tokens in the sentence pair. The alignment is based on exact token matching, followed by *WordNet* synonyms [95], stemmed tokens and paraphrases. Given a set of alignments, m, the METEOR score is the harmonic mean of precision P_m and recall R_m between the best scoring reference and candidate:

$$P_{en} = \gamma \left(\frac{ch}{m}\right)^{\theta} \tag{21a}$$

$$F_{mean} = \frac{P_m R_m}{\alpha P_m + (1-\alpha) R_m} \tag{21b}$$

$$P_m = \frac{|m|}{\sum_k h_k(c_i)} \tag{21c}$$

$$R_m = \frac{|m|}{\sum_k h_k(s_{ij})} \tag{21d}$$

$$\text{METEOR} = (1 - P_{en}) F_{mean} \tag{21e}$$

Thus, the final METEOR score includes a penalty P_{en} based on the chunkiness of resolved matches and an harmonic mean term that gives the quality of the resolved matches. The default parameters α, γ and θ are used for this evaluation.

[24] With C we denote the total set of machine generated captions.

ROUGE-L:

ROUGE [81] is a set of evaluation metrics designed to evaluate text summarization algorithms. ROUGE has different variants. The one that we are using here is ROUGE-L. This metric uses a measure based on the *Longest Common Sub-sequence* (LCS). A LCS is a set of words that is shared by two sentences and which occur in the same order. However, unlike *n-grams* there may be words in between the words that create the LCS. Given the length $l(c_i, s_{ij})$ of the LCS between a pair of sentences, ROUGE-L is found by computing an F-measure as following:

$$R_l = \max_j \frac{l(c_i, s_{ij})}{|s_{ij}|} \tag{22a}$$

$$P_l = \max_j \frac{l(c_i, s_{ij})}{|c_i|} \tag{22b}$$

$$ROUGE_L(c_i, S_i) = \frac{(1+\beta^2)R_l P_l}{R_l + \beta^2 P_l} \tag{22c}$$

In equations (22) R_l and P_l are the recall and precision of the LCS, while β is a constant that is usually set to favor recall ($\beta = 1.2$). Due to the use of the LCS, *n-grams* are implicit in this measure and so they do not need to be explicitly identified during the calculation of the score.

CIDE-r:

The CIDE-r metric [131] measures consensus in machine generated captions for images by performing a *Term Frequency - Inverse Document Frequency* (TF-IDF) weighting for each *n-gram*. Recalling that the number of times an *n-gram*, ω_k, occurs in a reference sentence s_{ij} is denoted by $h_k(s_{ij})$ and the number of times it occurs in the candidate sentence c_i is denoted by $h_k(c_i)$, then CIDE-r computes the TF-IDF weighting $g_k(s_{ij})$ for each *n-gram*, ω_k, utilizing the formula below:

$$g_k(s_{ij}) = \frac{h_k(s_{ij})}{\sum_{\omega_l \in \Omega} h_l(s_{ij})} \log\left(\frac{|I|}{\sum_{I_p \in I} \min(1, \sum_q h_k(s_{pq}))}\right), \tag{23}$$

where Ω is the vocabulary of all *n-grams* and I is the set of all images in the dataset. The first term measures the TF of each *n-gram*, ω_k, and the second term measures the rarity of ω_k using it's IDF. Intuitively, TF places higher weight on *n-grams* that frequently occur in the reference sentences describing an image, while IDF reduces the weight of *n-grams* that commonly occur across all descriptions. That is, the IDF provides a measure of word saliency by discounting popular words that are likely to be less visually informative (for example tokens like: “and”, “the” etc). The IDF is computed using the logarithm of the number of images in the dataset $|I|$ divided by the number of images for which ω_k occurs in any of it's reference sentences.

The $CIDEr_n$ score for *n-grams* of length n is computed using an average cosine similarity between the candidate sentence and the reference sentences:

$$CIDEr_n(c_i, S_i) = \frac{1}{m}\sum_j \frac{\mathbf{g^n}(c_i) \cdot \mathbf{g^n}(s_{ij})}{||\mathbf{g^n}(c_i)||\,||\mathbf{g^n}(s_{ij})||}, \tag{24}$$

where $\mathbf{g^n}(c_i)$ is a vector formed by $g_k(c_i)$ corresponding to all *n-grams* of length n and $||\mathbf{g^n}(\mathbf{c_i})||$ is the magnitude of the vector $\mathbf{g^n}(c_i)$. Similarly for $\mathbf{g^n}(s_{ij})$. Higher order (longer) *n-grams* are used to capture grammatical properties as well as richer semantics. The scores from *n-grams* of varying lengths can be combined as follows:

$$CIDEr(c_i, S_i) = \sum_{i=1}^{N} w_n CIDEr_n(c_i, S_i) \tag{25}$$

Here, w_n is a weight of importance given to each family of *n-grams*. Usually $w_n = 1/N$, whilst $N = 4$, which means that there is an equal distribution of weights for each form of $CIDER_n$.

SPICE:

All of the previously presented evaluation metrics are highly sensitive to *n-gram* overlap. However, mere *n-gram* overlap is neither necessary nor sufficient for two sentences to effectively convey the same meaning. Due to this limitation, the authors in [5] proposed a more robust metric that attempts to better capture the underlying semantic meaning between two sets of sentences. This innovative metric is called SPICE. In order to compute the SPICE score, both the candidate and reference sentences are initially parsed using a *dependency parser*, which establishes syntactic dependencies between the words. These syntactic dependencies can then be mapped through a *scene graph*. By employing this method, it becomes possible to detect complex relationships between words, such as identifying the main subject, determining which words function as action verbs, and understanding how the other words are syntactically and semantically related to them.

In the final graph, some vital post-processing steps for facilitating the extraction of semantics are necessitated. These include: the *Simplification of Quantificational Modifiers* (e.g ”three cats” are depicted as a single object with the attribute of ”three” and not separately), *Resolution of Pronouns* (pronouns like ”he”, ”she” etc, are replaced with the object that they refer to) and *Handling of Plural Nouns* (there are no multiple nodes for a plural noun but instead, the single instance is getting the attribute of plurality). An example, of such graph is displayed on Fig. 22 indicating the previously-mentioned characteristics. We can perceive the initial image with its respective parsed caption, showing all the intrinsic/syntactic dependencies and of course the produced scene graph.

What is more, the semantic relations in the generated scene graph are viewed as logical propositions or tuples. Each of these tuples contain elements that can represent an *Object* ($O(c)$), an *Attribute* ($K(c)$) or a *Relation* ($E(c)$), where c is a given caption either true or generated. Additionally, given the scene graph of a caption, $G(c)$, a function T that returns logical tuples from $G(c)$ can be defined as:

$$T(G(c)) \triangleq O(c) \cup K(c) \cup E(c) \tag{26}$$

For instance, for the graph depicted on Fig. 22, the tuples O, K and E could be:

$$\{O = \{(\text{girl}), (\text{court})\}, K = \{(\text{girl, young}), (\text{girl, standing}) (\text{court, tennis})\}, E = \{(\text{girl, on-top-of, court})\}\}$$

At this point, the comparison between the candidate and reference captions has been converted a comparison between their corresponding graphs. To complete this task, the authors in [5] introduced a binary matching operator $\otimes$, as the function that returns matching tuples in two scene graphs. With that in mind, the Precision P, Recall R and the final $SPICE$ score can be defined as:

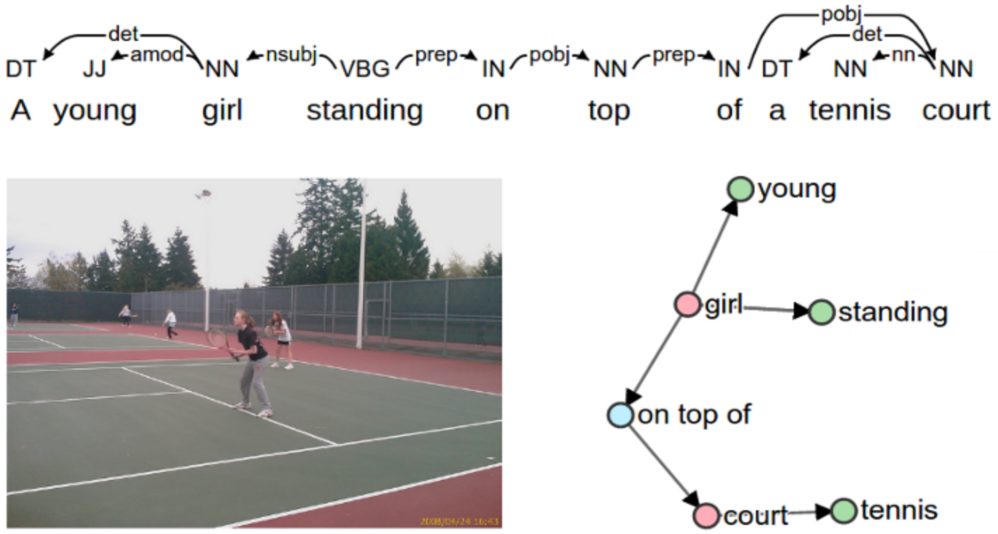


Figure 22. *A schematic representation of how SPICE works. In essence, it uses semantic propositional content to assess the quality of image captions. Reference and candidate captions are mapped through dependency parse trees (top) to semantic scene graphs (right) — encoding the objects (red), attributes (green), and relations (blue) present. Caption quality is determined using an F-score calculated over tuples in the candidate and reference scene graphs [5].*

$$P(c,S) = \frac{|T(G(c)) \otimes T(G(S))|}{|T(G(c))|} \tag{27a}$$

$$R(c,S) = \frac{|T(G(c)) \otimes T(G(S))|}{|T(G(S))|} \tag{27b}$$

$$SPICE(c,S) = F_1(c,S) = \frac{2 \cdot P(c,S) \cdot R(c,S)}{P(c,S) + R(c,S)} \tag{27c}$$

In equations (27), the Precision (P) can be defined as the ratio of the number of matching tuples between the candidate and reference scene graphs to the total number of tuples in the candidate scene graph, while the Recall (R) can be defined as the ratio of the number of matching tuples between the candidate and reference scene graphs to the total number of tuples in the reference scene graph. Lastly, the $SPICE$ is simply the harmonic mean ($F1$ score) of the previous precision and recall scores.

SPIDEr:

Probably one of the most accurate of the automatic metrics to capture both the semantic background and the syntactic morphology of a natural language sentence, was proposed by *Liu et al.* in [86]. This metric is dubbed as *SPIDEr* and it is a linear combination of SPICE and CIDEr. This metric is based on the scene graph of SPICE to detect semantic similarities but also uses the TF-IDF weights for each *n-gram*, ω_k (a component of CIDEr), for identifying the major syntactical factors of a sentence.

Moreover, the authors use a *Policy Gradient* (PG) method, which improves on the prior MIXER approach, by using Monte Carlo rollouts in order to optimize this metric (among others), for achieving more semantically plenteous and descriptively precise captions for images. Their results illustrate that SPIDEr emulated better the human judgment and therefore comes closer to human evaluation of natural language scripts, than any of the other automatic metrics including SPICE and CIDEr.

3.5.2 Human Evaluation

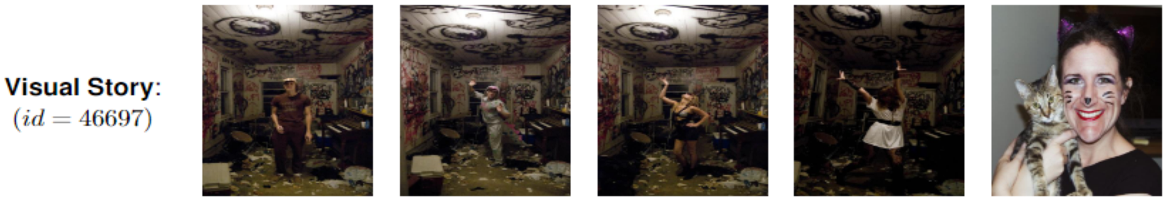
In the second half of this work, as it has been numerously highlighted, will be dedicated to the story generation. Besides, this is the final goal of this project as well. We divide our story generation procedure in to two mains sub-categories. Firstly, we have the machine generated stories which come after feeding our chosen language models (T4, T5 & BART), with isolated captions that originate from VIST dataset. Secondly, we will have the case of testing end-to-end the whole framework of Fig. 14, feeding it images from VIST, expecting to produce recitative stories (that will correspond only to the images - i.e. no intermediate reference point to the captions). At this point, it becomes clear, that first sub-category will be used as the anteroom for the second, in the sense that the models which generate the best *stories from captions*, will be combined with the top models that *produce accurate captions* from phase 1, in order to result to the best possible narrative generators of visual stories.

In all respects, we would like somehow, to evaluate this final product[25] with credibility and validity. While automatic metrics such as BLEU, METEOR, and SPICE provide valuable quantitative assessments of generated text, they inherently fall short in capturing the nuanced qualities essential for coherent and imaginative storytelling. These metrics primarily focus on surface-level similarities to reference texts and fail to account for the deeper aspects of narrative structure, creativity, and contextual coherence. Therefore, to achieve a comprehensive evaluation of generated stories, it is imperative to incorporate human judgment, which can discern and appreciate the subtleties and complexities that automatic metrics overlook.

Our human evaluation procedure took place in the form of a questionnaire where participants had to answer questions related to the machine generated visual stories according to their judge based on criteria. More specifically, in every visual story sequence we have five given stories, which we need to evaluate. An example of such a visual storyline along with the five accompanying stories is provided on Fig. 23. We need

[25] i.e. the final stories generated during the ultimate phase

to point out, that from these five stories one is written by humans, while the rest four are machine generated. In the last parts of our work (see subsection 5.5), we will get to know that these four machine generated stories were derived by variants of the framework illustrated on Fig. 14, as we just described previously.



Story 1 (S1):	I hope they like it.	He is not like an artist.	But he did a good job.	It's not everyday that you see people spray painting on walls.	I'm not sure if she likes the cat face or not.
Story 2 (S2):	My friends and I decided to take pictures posing in our room that we made turn out to look of nothing but cats.	We are all making silly poses to take.	My friend here decided to dance to her music all while making different poses for the camera.	Our friend here decided to just stand there and not show any emotion at all.	I really love kittens. Aren't they beautiful.
Story 3 (S3):	The walls were covered in graffiti.	And the artist was very proud of his work.	The lady was also very impressed with the finished product.	The girl was so happy with her work.	She even had her face painted.
Story 4 (S4):	Jason tries to get this shot.	Jason tries to take a picture of the wall art.	This is Marissa trying to get a good pose for her picture.	Marissa just had to take another picture.	Finally, Marissa takes her picture.
Story 5 (S5):	The walls were covered in graffiti.	The artist was very proud of his work.	The lady was very excited to show off her artwork.	She posed in front of the stairs.	The woman then dressed up as a cat.

Figure 23. *A depiction of a visual story expanding over five images. Along with it, we show five stories four of which are machine generated by our models and one is human written.*

This evaluation procedure was intended to bring out any semantic traits of our generated stories. In addition, we would like to directly compare them with their human written counterpart stories. Therefore, we adopted two different types of evaluation and these are the following:

- Rank stories based on certain criteria
- Guessing the human written story

The criteria we seek in our stories are both lexical and semantic and they are listed below:

1. **Relevance to the input images** (How well does the story correspond to the sequence of images? Does it accurately describe or interpret the visual content?)
2. **Coherence and Flow** (How coherent is the given story? Do the sentences flow logically from one to the very next?)
3. **Narrative Depth** (Does the story provide enough depth and detail to create a vivid narrative experience for the reader?)
4. **Imagination and Creativity** (How imaginative and creative is the generated from the model, story?)
5. **Engagement and Interest** (How engaging is the story? Does it capture your attention and interest?)
6. **Language Quality and Style** (How would you rate the quality of the language and style of the story?)

Our evaluation procedure is divided in to two main sections. The first section is devoted in ranking our five generated stories based on the aforementioned criteria, while in the latter section we ask from our participants to guess which among the five given stories is the human written one.

4 Experiments - Set up

As previously discussed, the first phase of this project is dedicated to evaluate the capabilities of Clip-Cap model in generating descriptions for images of the VIST dataset. In the second phase, we are going to fine-tune other Encoder-Decoder architectures to be able to produce complete stories from golden isolated captions, meaning that they also originate from the DII tier of VIST. Finally, we aim to combine these two phases and leverage the framework of Fig. 14 to craft narrative visual stories, that reveal underlying subplots or background details on the initial images, that mere captioning alone cannot convey.

4.1 Phase 1 - Evaluation of Clip-Cap in image captioning

In the first phase of our project we needed to extract the necessary images from the train, validation and test parts of VIST with their corresponding captions in the DII tier. However, due to the fact that the captions and the images are provided separately, we needed first to find the part of images that had a corresponding caption on the DII domain. Therefore, the total amount of train, validation and test image-caption pairs were reduced[26]. The final number of such pairs were 39,696, 4,949 and 4,992 for the train, validation and test sets of the DII tier, respectively. We also need to underline, that since every image of VIST is given with three captions, the aforementioned image-caption pairs for all sets, consist of three caption per image as well. Additionally, due to the immense resolution of images taken from VIST (reaching up to 4096×4096 size and more), we first needed to resize them into a standard size of 512×360, if the width of the image exceeded the height or into 360×512 in an opposite case.

In the subsequent phase, we used three different versions of the Clip-Cap model and evaluated them on the test set of the image-caption pairs. These were: a Zero-shot Clip-Cap, taken directly pre-trained from MSCOCO and used for generating captions in the VIST dataset, the pre-trained Clip-Cap model from MSCOCO but also fine-tuned on VIST and lastly the Clip-Cap model trained totally from scratch on the train part of the previously mentioned pairs and validated in the corresponding validation set. From the Clip-Cap variants that were indeed fine-tuned, we need to underline that we opted a training of 10 $epochs$. As far as the loss function is concerned we chose *Cross-Entropy*, whereas for optimization we utilized *AdamW*. Finally, all three families of models were trained/fine-tuned on an RTX-2080 GPU.

4.1.1 Zero-shot Models (Trained on MSCOCO)

As already explained Clip-Cap model (pre-trained) comes with two different variants, depending weather we use a transformer-based or an MLP mapping network. Additionally, during inference time the model outputs probabilities for all vocabulary tokens[27], which are used to determine the next token by employing a greedy approach or beam search. This, bring us to four different variants of possible zero-shots models: 1) MLP mapper with greedy search and 2) beam search and 3) transformer mapper with greedy search and 4) beam search. It's worth reminding that the MLP-based networks had fine-tuned GPT-2 while the transformer-based had the language model frozen.

Furthermore, we should mention that the MLP-based networks used ViT_{Huge}(with 32 processing layers) [24] as the image encoder of CLIP and thus *prefix_size=512*, whilst the transformer-based models had as CLIP image encoder a *ResNet-50* [42] network with *prefix_size=640*. Ultimately, for the MLP based network the *prefix_length* was chosen to be 10, whereas for the Transformer based Clip-Cap both the *prefix_length* and *clip_length* were set at 40.

4.1.2 Pre-trained Model from MSCOCO & Fine-tuned on VIST dataset

The second family of Clip-Cap models that we evaluated were also pre-trained on MSCOCO but later fine-tuned on the VIST dataset as well. Since the initial models originated from the work of *Mokady et al.*, both the MLP and Transformer based networks had the same features and parameters like in the case of the Zero-shot models. That means that the MLP-based Clip-Cap had ViT_{Huge} as CLIP encoder and the

[26] compared to the original number of train, validation and test annotations and the respective numbers of images.

[27] It's remarkable to note that the vocabulary of Clip-Cap is the vocabulary of the language generator, i.e GPT-2

transformer-based Clip-Cap had a *ResNet-50*, with the same size of parameters as in the previous case. Albeit, unlike the Zero-shot occasion, this time during the fine-tuning of the models, for both type of mapping networks we fine-tuned & left frozen the language model. Finally, we decided once more to use both the greedy and explorative (beam search) approaches during generation.

4.1.3 Models Trained from-scratch solely on VIST dataset

The last family of models, that were deployed for our first sub-research question, were those which were set and train from-scratch exclusively on the VIST dataset. Considering the versatility of this case, we attempted to train and test several models depending on various of parameters. These were:

- Mapping Network used: MLP or Transformer.
- Train or not the language model (GPT-2).
- CLIP Encoder used: *ResNet-50* or ViT_{Huge}.
- Use of greedy search or beam search during generation.

Therefore, all of the above-mentioned possible combinations were utilized.

4.2 Phase 2 - Evaluation of text-to-text models on story generation

As we have mentioned plenty of times, throughout this report, the second part of project was dedicated to the transformation of plain captions to meaningful and coherent stories using some language models. Our original framework from Fig. 14 dictates that for the second phase we are going to use T4, T5 and BART generative models which were earlier introduced. Having said that, in this section we will get to know all the architectural details of the aforementioned models. We should already underscore that all models reported further down were trained/fine-tuned on an RTX-2080 GPU.

4.2.1 From-scratch Model (T4)

As discussed earlier, in subsection 3.3, we came up with several pre-training steps for our T4 model including *Mask Language Modeling (MLM)* and *Sentence Permutation (SP)*. The model was also fine-tuned precisely to our specific task by utilizing a *direct sequence-to-sequence (seq-to-seq)* training. However, for the wholeness of our study we used two variants of T4. The first, was the one that we just described where the T4 model had been pre-trained with both MLM & SP and then been subjected to normal seq-to-seq training. Hereafter, this model will be named as $T4_{MLM+SP}$. In the second case, we had a T4 model exclusively trained with the direct seq-to-seq manner and as a result this model will be dubbed as $T4_{base}$ from now on.

Moreover, as previously underlined, both of these versions were an authentic replica of the original transformer architecture (from Fig. 18) and for both we used the BERT tokenizer in order to tokenize all the input sequences (captions & stories). The structural details of these two models are given on Table 4 below ($|V|$ is the Vocabulary Size while FF dim. is the the FeedForward dimension that is used in MLP modules of the Transformer blocks and LR rate is the Learning rate):

Model	$\|V\|$	Emb. Size	Heads	Enc. Layers	Dec. Layers	Dropout[28]	FF dim.	LR rate
$T4_{base}$	30522	512	8	6	6	0.1	2048	0.0001
$T4_{MLM+SP}$	30522	512	8	6	6	0.1	2048	0.0001

Table 4. *Configuration details for the two versions of the T4 model*

As it can easily be inferred from Table 4, the two variants of T4 model that we used, were identical in their parameterization. As a result, the exact number of parameters for both models stood at 75,425,594. The

[28] The dropout rate concerns the whole architecture universally.

loss function and the optimizer were *Cross-Entropy* and *AdamW* correspondingly[29]. It is also noteworthy, that for both models we used a batch size of $bs = 10$ and a linear scheduler for adjusting the learning rate, with 10000 warm-up steps and $epochs \times \left\lfloor \frac{|D_{tr}|+bs-1}{bs} \right\rfloor$ training steps[30], where $|D_{tr}|$ is the length of the train dataset and the symbol $\lfloor \quad \rfloor$ indicates the arithmetic rounding down.

The only concrete difference between these two models were the number of training epochs. While for T4$_{base}$ we opted for 30 $epochs$ (all for seq-to-seq training and 10 to each sub-dataset), for T4$_{MLM+SP}$ model we chose 45, allocating 15 $epochs$ for MLM, 15 $epochs$ for SP and the rest 15 for the fine-tuning (seq-to-seq training, 5 for each sub-dataset). Last of all, both T4$_{base}$ and T4$_{MLM+SP}$ were tested under all the possible decoding methods which were presented on subsection 3.4 and then compared between each other but also against the T5 & BART contenders.

4.2.2 T5 model

Our T5 model was directly retrieved from Hugging Face and the used version was *"T5-base"*. It was developed by Google along with it's other variants. The configuration characteristics of this model, hereafter called T5$_{base}$, are provided on Table 5. This table, showcases that T5$_{base}$ outclasses the T4 alternatives in all configurational parameters, with larger vocabulary and greater dimensionality in both the embeddings and the feedforward units. In addition, both heads and encoder & decoder layers are more, than in the case of T4. This resulted that T5$_{base}$ had a total of 222,903,552 parameters. Moreover, as a loss function T5 uses the integrated to it's module *Cross-Entropy* function, whilst for optimizer we used once again *AdamW*, but this time with half the learning rate (than in the case of T4). The scheduler applied was exactly the same as in T4 models and the batch size was also $bs = 10$. Finally, due to it's bigger size, we fine-tuned this model for 24 $epochs$ giving 8 $epochs$ to each of the three alternatives regarding the input captions. T5$_{base}$ was tested under *greedy search*, *nucleus sampling* & *beam search* decoding strategies.

Model	$\|V\|$	Emb. Size	Heads	Enc. Layers	Dec. Layers	Dropout	FF dim.	LR rate
T5$_{base}$	32100	768	12	12	12	0.1	3072	0.00005

Table 5. *Configuration details for the T5 model*

4.2.3 BART model

The BART model used, was also taken from Hugging Face and was developed by Facebook/Meta. The version that was deployed was *"BART-large"* and it's parameterization is granted on Table 6. Hereinafter, this model is named BART$_{large}$. Someone, by looking on Table 6 will quickly realize that BART$_{large}$ is by far superior to our T4 models and even greater in size than T5$_{base}$. This enormity is mainly justified by the immense Vocab size that the BART model incorporates (if we contrast it to the size of vocabularies of T4 & T5), but also due to the increment in the size of embeddings and in the dimensionality of the feedforward components. Consequently, BART$_{large}$ is comprised in total by 406,291,456 parameters.

As the T5 model, BART module has it's own built-in *Cross-Entropy* as loss function. Both the optimizer and the scheduler were the same as in the previous two cases (*AdamW* as an optimizer and a linear scheduler with the same configuration). A small alteration, due to the extent of this model, was made in the batch size and the number of $epochs$. Regarding the former, we reduced it to $bs = 8$, while regarding the latter we picked 12 learning $epochs$ (that is 4 $epochs$ per variant of the input captions). Lastly, following the example of T5$_{base}$, BART$_{large}$ was also tested under *greedy search*, *nucleus sampling* & *beam search* generation strategies.

Model	$\|V\|$	Emb. Size	Heads	Enc. Layers	Dec. Layers	Dropout	FF dim.	LR rate
BART$_{large}$	50265	1024	16	12	12	0.1	4096	0.00005

Table 6. *Configuration details for the BART model*

[29] For more information about the optimizer and the loss function you can check Appendix A.

[30] Essentially the quantity $\left\lfloor \frac{|D_{tr}|+bs-1}{bs} \right\rfloor$ is the length of the train data-loader.

4.3 Models summary (phase-2) & Experimentations planning

At this point, we would like to summarize the total number of parameters of the four models from paragraphs 4.2.1, 4.2.2 and 4.2.3 into Table 7, as later in Chapter 5 it will be a point of comparison between our candidate architectures. From that depiction, the difference in size is readily observable. T5 model is 3 times as large as our from-scratch models and the BART candidate, in turn, is twice as big as the T5, making it almost 6 times larger than the T4 counterparts.

Model	**Total number of parameters**
T4$_{base}$	75,425,594
T4$_{MLM+SP}$	75,425,594
T5$_{base}$	222,903,552
BART$_{large}$	406,291,456

Table 7. *Total number of parameters for each of the models used during the second phase*

On top of that, the experiments in which the models (or families of models) from subsections 4.1 and 4.2 took place, can be shaped as further down:

- Experiments on *Image Captioning* by utilizing all the families of models from paragraphs 4.1.1, 4.1.2 and 4.1.3. These Clip-Cap variants were trained and validated respectively on the train and validation parts of VIST and ultimately tested on the test section of the dataset. (Phase 1)

- Experiments on *Story Generation given only language descriptions (captions)*. Obviously, the captions emanated from the DII part of VIST dataset. The architectures involved were those appearing on Table 7 (each with the selected aforementioned decoding techniques) and for their training, validation and testing we focused on the corresponding pieces of VIST/DII. (Phase 2)

- In the end, our final experimentation engaged the direct deployment of the framework from Fig. 14 on the *images of VIST/Test* (and only on this part), with the purpose of *Visual Story Generation*. This framework, would emerge (as Fig. 14 illustrates) by combining the top two performing models of the previous two occasions (Simple cases of Image captioning & Story generation from captions). It's worth noting that no training procedure is contained on this stage. (Ultimate Phase)

5 Results - Discussion

Coming to the results section of this project, we can divide them into three different topics, following an one-to-one correspondence with the segregation of tasks from subsection 4.3. Recalling also our central architecture (see Fig. 14) and the research questions (see section 1.4) that we seek to answer, these topics could be formulated as follows:

- Results from Phase 1 for Image Captioning by evaluating different versions of Clip-Cap.
- Results from Phase 2 for Storytelling by evaluating our different types of our *text-to-text* models, when these are fed with the ground truth labels from the VIST dataset, and
- Results from the evaluation of the whole framework of Fig. 14, in the ultimate test time by feeding it the test images from VIST and expecting it to produce visually relevant, semantically coherent and linguistically complete storylines. (Results from Ultimate Phase)

5.1 Phase 1 results

In the following sub-chapters we present the analytical results from all versions of Clip-Cap that we discussed earlier in the part 4.1. The total number of models tested in this phase reached the 28, originating by combining different components of Clip-Cap. The evaluation metrics that we used, where presented in paragraph 3.5.1[31]. We need to underscore that B-1, B-2, B-3, B-4 stand for BLEU-1, BLEU-2, BLEU-3 and BLEU-4 respectively, while M and R_L stand for METEOR and ROUGE-L correspondingly. Lastly, it's important to say that the models appearing in parts 5.1.1 and 5.1.2 are all taken directly from the repository of *Mokady et al.*[32], and thus they have already pre-set CLIP Encoder. More specifically, all models comprised by an MLP mapper had ViT$_{Huge}$ as their image encoder, while the Transformer based networks had *ResNet-50* for encoding images. In sub-subsection 5.1.3, we had the flexibility to set the entire model from-scratch and as a result, we tested both of these CLIP Encoders for both mapping networks.

5.1.1 Results for zero-shots Models

Table 8 summarizes the results for the four zero-shot models that we tested during our experimentation on the test set of the DII tier of VIST. In addition to the abbreviations that we have already mentioned, here we should add that the "TRASNF." and "MLP" denote the mapping network (Transformer and MLP respectively), while the "with beam" & "no-beam", indicate the usage or no of beam search strategy during the decoding stage. Scores in bold, denote the best performance per metric within this family of models (i.e zero-shot models). Finally, there is an identification number for each model ($\mathcal{M}_i$), which will come in handy in our upcoming evaluation procedure.

Method / Metric		**B-1**	**B-2**	**B-3**	**B-4**	**M**	**R_L**	**Cider**	**Spice**	**Spider**
Zero-shot	TRANSF. no-beam ($\mathcal{M}_1$)	32.57	12.87	5.75	2.89	7.38	23.92	4.27	2.79	3.53
	TRANSF. with beam ($\mathcal{M}_2$)	32.12	**13.13**	**6.3**	**3.31**	7.64	24.08	**4.82**	**3.08**	**3.96**
	MLP no-beam ($\mathcal{M}_3$)	**33.67**	13.06	5.47	2.6	**7.66**	**24.35**	3.9	2.3	3.1
	MLP with beam ($\mathcal{M}_4$)	32.72	12.6	5.11	2.46	7.32	23.77	3.64	2.25	2.95

Table 8. *Evaluation Results for Zero-shot Models. The purpose of colored cells is described on paragraph 5.2.1*

[31] In phase 1 we did not use human judgment as an evaluation method.

[32] The repository can be found HERE.

5.1.2 Results for fine-tuned from MSCOCO Models

Table 9 shows the results we obtained from the different variants of models that we used, which they were already pre-trained on the MSCOCO dataset. It's notable to emphasize that here the postfix "prefix-only" and "prefix-GPT2" mean that during fine-tuning only the mapping network was trained (in the first case) and both the mapping network and the language model (GPT-2) were trained (in the second case). Once more, scores in bold denote the best performance per metric within this family of models.

Method / Metric		B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
Fine-tuned on MSCOCO	MLP prefix-only no-beam ($\mathcal{M}_5$)	**33.11**	13.17	5.99	3.08	7.76	24.16	4.46	3.01	3.74
	TRANSF. prefix-only no-beam ($\mathcal{M}_6$)	31.95	12.9	5.9	3.02	7.71	24.19	4.37	2.92	3.65
	MLP prefix-only with-beam ($\mathcal{M}_7$)	32.34	**13.35**	**6.49**	**3.52**	**8.1**	**24.4**	5.25	**3.41**	4.33
	TRANSF. prefix-only with-beam ($\mathcal{M}_8$)	30.56	12.57	6.15	3.34	7.95	24.12	5.16	3.34	4.25
	MLP prefix-GPT2 no-beam ($\mathcal{M}_9$)	32.64	12.88	6.03	3.16	7.73	23.71	4.75	3	3.88
	TRANSF. prefix-GPT2 no-beam ($\mathcal{M}_{10}$)	32.52	13.01	6	3.08	7.74	23.97	4.62	2.9	3.76
	MLP prefix-GPT2 with-beam ($\mathcal{M}_{11}$)	30.86	12.44	6.11	3.32	7.86	23.08	**5.63**	3.37	**4.5**
	TRANSF. prefix-GPT2 with-beam ($\mathcal{M}_{12}$)	30.81	12.51	6.13	3.29	7.89	23.24	5.5	3.24	4.36

Table 9. *Evaluation Results for Fine-tuned on MSCOCO Models. The purpose of colored cells is described on paragraph 5.2.1*

5.1.3 Results for Models trained from-scratch on VIST

Table 10 summarizes the results from all the different variants of Clip-Cap that we created and trained from-scratch. As we mentioned earlier, in this occasion we were also able to alter the CLIP encoder opting between *ResNet-50* and ViT$_{Huge}$. The rest of the abbreviated parameters are kept the same as in the case of sub-subsections 5.1.1 and 5.1.2. Lastly, once again, results in bold indicate the best performing model per metric within the family of the from-scratch trained models.

5.2 Phase 1 discussion

In this section we will analyze the results shown in paragraphs 5.1.1, 5.1.2 and 5.1.3. This analysis will be unfolded in two main directions. Firstly, we are going assess the results with respect to the models and the three greater belonging families of those, that we introduced in sub-subsections 4.1.1, 4.1.2 and 4.1.3. The second direction that we moved, was the investigation of results regarding the different components of Clip-Cap itself. This involves comparisons between the kind of CLIP encoder used on Clip-Cap, the kind of the mapping network used and the preservation of GPT-2 generator as frozen or not, during fine-tuning.

5.2.1 Family-wise, Model-wise Comparison

Looking back on the results of phase 1 collectively from Tables 8, 9 and 10, we can observe that per metric there are three cells/scores that are highlighted with green, yellow and red colors. These correspond to the top three scores that any of the evaluated models accomplished in a metric. More precisely, the best score is highlighted with green, while the second best with yellow and the third with red. In order to quantify the results, we will define a simple yet useful function of scoring points by a certain model, $\mathcal{M}$, on a specific metric, m, in the following manner:

Models Trained from-scratch

	Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
ResNet CLIP Encoder	MLP Scratch prefix-only no-beam ($\mathcal{M}_{13}$)	30.82	12.53	5.85	3.05	7.57	23.81	4.39	2.89	3.64
	TRANSF. Scratch prefix-only no-beam ($\mathcal{M}_{14}$)	31.85	12.97	5.95	3.06	7.76	**24.2**	4.41	2.97	3.69
	MLP Scratch prefix-only with-beam ($\mathcal{M}_{15}$)	26.41	10.9	5.38	2.92	7.78	23.64	4.84	3.29	4.06
	TRANSF. Scratch prefix-only with-beam ($\mathcal{M}_{16}$)	29.31	11.94	5.7	3.01	**7.93**	24.09	4.93	3.38	4.16
	MLP Scratch prefix-GPT2 no-beam ($\mathcal{M}_{17}$)	31.43	12.31	5.52	2.85	7.61	23.47	4.33	2.79	3.56
	TRANSF. Scratch prefix-GPT2 no-beam ($\mathcal{M}_{18}$)	**32.05**	12.41	5.43	2.68	7.68	23.66	4.3	2.83	3.56
	MLP Scratch prefix-GPT2 with-beam ($\mathcal{M}_{19}$)	29.7	11.98	5.89	3.21	7.85	22.72	5.42	3.21	4.31
	TRANSF. Scratch prefix -GPT2 with-beam ($\mathcal{M}_{20}$)	30.22	11.99	5.71	3.09	7.79	23.02	5.26	3.16	4.21
ViT CLIP Encoder	MLP Scratch prefix-only no-beam($\mathcal{M}_{21}$)	31.83	12.53	5.91	3.08	7.52	23.75	4.42	2.81	3.61
	TRANSF. Scratch prefix-only no-beam ($\mathcal{M}_{22}$)	31.39	12.34	5.65	2.88	7.6	23.66	4.15	2.93	3.54
	MLP Scratch prefix-only with-beam ($\mathcal{M}_{23}$)	25.74	10.38	4.87	2.5	7.64	23.17	4.71	3.19	3.95
	TRANS. Scratch prefix-only with-beam ($\mathcal{M}_{24}$)	29.5	11.87	5.83	3.17	7.92	23.82	5.09	**3.39**	4.24
	MLP Scratch prefix-GPT2 no-beam ($\mathcal{M}_{25}$)	31.8	12.49	5.69	2.86	7.72	23.54	4.61	2.89	3.75
	TRANSF. Scratch prefix-GPT2 no-beam ($\mathcal{M}_{26}$)	31.93	**12.54**	5.62	2.84	7.7	23.55	4.44	2.88	3.66
	MLP Scratch prefix-GPT2 with-beam ($\mathcal{M}_{27}$)	30.12	12.09	**5.96**	**3.18**	7.84	22.81	**5.46**	3.22	**4.34**
	TRANS. Scratch prefix-GPT2 with-beam ($\mathcal{M}_{28}$)	30.3	12.03	5.89	3.22	7.8	22.85	5.33	3.16	4.25

Table 10. *Evaluation Results for Trained from-scratch Models with ResNet-50 and ViT$_{Huge}$ CLIP Encoders. The purpose of colored cells is described on paragraph 5.2.1*

$$f(\mathcal{M}(m)) = \begin{cases} 3, & \text{if } \mathcal{M}(m) \text{ is the best score } \forall \mathcal{M}, \forall \mathbb{M} \text{ on metric } m \\ 2, & \text{if } \mathcal{M}(m) \text{ is the } 2^{nd} \text{ best score } \forall \mathcal{M}, \forall \mathbb{M} \text{ on metric } m \\ 1, & \text{if } \mathcal{M}(m) \text{ is the } 3^{rd} \text{ best score } \forall \mathcal{M}, \forall \mathbb{M} \text{ on metric } m \\ 0, & \text{otherwise,} \end{cases} \tag{28}$$

where $\mathcal{M}(m)$ denotes the score of a model $\mathcal{M}$ on metric m. With $\mathbb{M}$ we symbolize all possible sets of models, which in practice are the three families of models presented on Tables 8, 9 and 10. Of course, for each model $\mathcal{M}$, we have that $\mathcal{M} \in \mathbb{M}$ for some family (set) $\mathbb{M}$.

Comparison between families of models:

Having that in mind, we can see that the overwhelming majority of best scores per metric is achieved by the fine-tuned on MSCOCO models. More specifically, $8/9$ of top-1 scores are achieved by this family of models on metrics B-2, B-3, B-4, M, R_L, Cider, Spice and Spider. In addition, $6/9$ second best scores occur within this family as well, on metrics B-1, B-2, B-4, M, Cider and Spider. Regarding the zero-shot models, they have the best score on B-1 and the second best scores in B-3 and R_L. Quite counter-intuitively, trained from-scratch models, despite the fact that were more numerous, under-perform hitting only one top-2 result on Spice metric. To recapitulate, in terms of our defined function from equation (28) we can see the dominance of fine-tuned models, by the fact that:

$$f(\mathbb{M}_{zero\_shot}(m)) = 9, f(\mathbb{M}_{fine\_tuned}(m)) = 38 \text{ and } f(\mathbb{M}_{from\_scratch}(m)) = 7,$$

where $\mathbb{M}_{zero\_shot}$, $\mathbb{M}_{fine\_tuned}$ and $\mathbb{M}_{from\_scratch}$ match to the three families of models respectively. What is more, Table 11 presents the average results per metric attained by each family of models (best results per metric are on bold). This table allows us to observe that zero-shot models perform better in simpler metrics (such as B-1 & B-2), whereas the $\mathbb{M}_{fine\_tuned}$ family excels in more complex metrics (such as Cider, Spice & Spider). An exception is noted in the M metric, where the from-scratch models demonstrate superior performance. Overall, Table 11 provides an additional evidence that $\mathbb{M}_{fine\_tuned}$ constitutes the most accurate family of models for the task of image captioning on VIST dataset.

Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
$\mathbb{M}_{zero\_shot}$	**32.77**	**12.92**	5.66	2.82	7.5	24.01	4.16	2.61	3.39
$\mathbb{M}_{fine\_tuned}$	31.85	12.85	**6.1**	**3.23**	7.64	**24.03**	**4.97**	**3.15**	**4.06**
$\mathbb{M}_{from\_scratch}$	30.28	11.7	5.63	2.98	**7.73**	23.39	4.75	3.05	3.91

Table 11. *Evaluation Results per Family of Models*

Individual Comparison of models:

Going to an individual model-wise comparison we can identify again that fine-tuned on MSCOCO model, without tuning GPT-2 and with MLP as a mapping network which uses beam search ($\mathcal{M}_7$), is by far the most powerful captioning model, accomplishing 6 out of 9 best scores across all the evaluated metrics. Another notable mention is the respective model where GPT-2 was tuned ($\mathcal{M}_{11}$), which attains $2/9$ of the best performances, the zero-shot model with MLP mapping network and beam search ($\mathcal{M}_3$), which accounts for the best score on B-1 and finally the fine-tuned model with Transformer mapper without training GPT-2 but with the utilization of beam search ($\mathcal{M}_8$).

Table 12 summarizes all the models, $\mathcal{M}_i$, for which $f(\mathcal{M}_i) > 0$ at least for one metric m, in descending ranking. Here, i is the identification number of the models appearing in Tables 8, 9 and 10, and $i \in [1, 28]$. From there, we verify the analysis previously discussed with the dominance of $\mathcal{M}_7$ which accounted for more than 2.5 times greater $f(\mathcal{M}_i)$ score than any other model. Also, as we can see the total number of models that achieved a top-3 performance were 12.

Model id	Score $f(\mathcal{M}_i)$	Model id	Score $f(\mathcal{M}_i)$
$\mathcal{M}_7$	18	$\mathcal{M}_2$	3
$\mathcal{M}_{11}$	7	$\mathcal{M}_{24}$	2
$\mathcal{M}_3$	5	$\mathcal{M}_{16}$	2
$\mathcal{M}_8$	5	$\mathcal{M}_{27}$	2
$\mathcal{M}_5$	4	$\mathcal{M}_{14}$	1
$\mathcal{M}_{12}$	4	$\mathcal{M}_4$	1

Table 12. *Ranking of models based on their $f(\mathcal{M}_i)$ score*

Examples of caption generation:

Table 13 shows some examples of generated captions by some of the models presented on Table 12 and for another model that didn't achieve $f(\mathcal{M}_i) > 0$, for contrast. Specifically, the models that we chose were $\mathcal{M}_7, \mathcal{M}_{11}, \mathcal{M}_3$ and $\mathcal{M}_{22}$. In the first column of the table the image is depicted, in the middle column the three given ground truth labels (originating from VIST) are shown, while the last column has the generated captions by the aforementioned models.

This table indicates that the automatic evaluation of models is not in symphony with our intuition. For instance, as we saw, the zero-shot model ($\mathcal{M}_3$), has superior scores in automatic metrics than the scratch model ($\mathcal{M}_{22}$), but the captions that it generates are far less informative and far more inaccurate than the latter model. In general, according to human judge we could argue that $\mathcal{M}_3$ performs very poorly, repeating some captions for several examples such as "A man is sitting on the ground.". On the contrary, both fine-tuned models ($\mathcal{M}_7$ & $\mathcal{M}_{11}$) seem to be quite descriptive and accurate by producing captions that are quite complete and close to the ground truth captions or even contain world knowledge, like "A view of the Golden Gate Bridge in San Francisco.". It is noteworthy, that for the third image (arguably the most difficult from those displayed) the only model that came close to the actual description is the model $\mathcal{M}_{11}$.

Summing up, it is safe to dispute that the good performance of $\mathcal{M}_7$ & $\mathcal{M}_{11}$ on the automatic metrics is also depicted in the intuitive results of Table 13, something which is not in valid for the zero-shot model, $\mathcal{M}_3$. Last but not least, on the third column of the table we indicate inappropriate generations (or hallucinations [112][33]) with respect to the input image. These "false" generations are highlighted with red hue and are in italic style. Again, we confirm that model $\mathcal{M}_3$ hallucinates the most while for the third image all models give incorrect (or highly incorrect) descriptions.

Nevertheless, despite these hallucinations, our models and especially those fine-tuned, such as $\mathcal{M}_7$ and $\mathcal{M}_{11}$, can still produce highly descriptive captions that imprint the central essence of the input images (Sub-RQ 1). These captions, could in turn constitute the conceptional basis for another generative model that will operate as a narrative storyteller.

5.2.2 ClipCap's component-wise Comparison

Besides comparing the family of tested models or the models individually, we also experimented with the components of Clip-Clap itself. More precisely, the evaluation included comparisons between the mapping network that was used (MLP vs Transformer), if the training procedure encompassed the fine-tuning of the language model or not and the type of CLIP encoder that was exploited (ViT$_{Huge}$ vs *ResNet-50*).

ViT$_{Huge}$ vs *ResNet-50*:

Our first comparison came with the models trained from-scratch, where we had the ability to alter the CLIP encoder of Clip-Cap. Fig. 24 visualizes the average results per metric for the eight models of Table 10 that used ViT$_{Huge}$ ($\mathcal{M}_{13}$ to $\mathcal{M}_{20}$) and *ResNet-50* ($\mathcal{M}_{20}$ to $\mathcal{M}_{28}$) image encoders correspondingly. From there, we could argue that alternating the encoding method of images does not play any significant role on the final evaluation of the captions since both families of models perform quite evenly.

[33] Hallucination in image captioning is a phenomenon where a captioning model, doesn't learn the complete representation of the visual scene due to several reasons such as overfitting to the loss function. The outcome, is the generation of inconsistent descriptions with respect to the visual input.

Frozen GPT-2 vs Fine-tuned GPT-2:

Secondly, we would like to investigate the performances per metric by averaging the results between the models that kept *frozen* GPT-2 and those in which we *fine-tuned* GPT-2. The averaging of results concerns both the models that were fine-tuned on MSCOCO ($\mathcal{M}_5$ to $\mathcal{M}_{12}$) and those that were trained from-scratch ($\mathcal{M}_{13}$ to $\mathcal{M}_{28}$). A depiction of these results is offered on Fig. 25. Once more, there is no vivid distinction between the two tested families as both Frozen GPT-2 and Fine-tuned GPT-2 models showcase very close/similar scores. It is evident from the graph, that neither family consistently outperforms the other across all metrics. For instance, the Fine-tuned GPT-2 models achieve higher B-1, while the Frozen GPT-2 models get the edge in R_L. This observation indicates that the performance varies across the metrics, with each family occasionally outperforming the other.

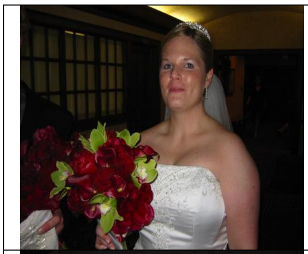
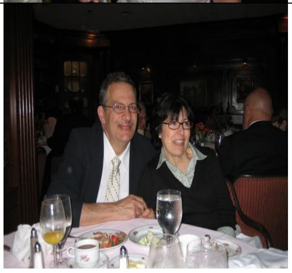
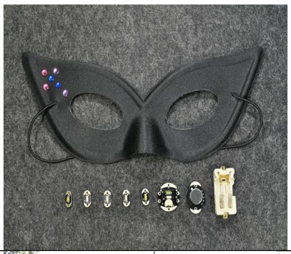
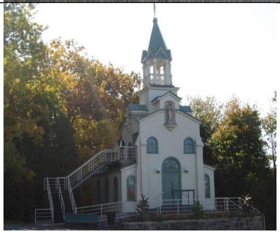
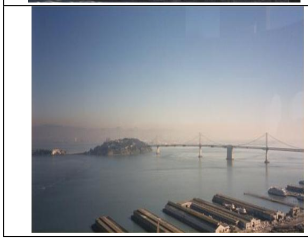
Images	Ground Truth Captions	Generated Captions
	1. A large woman in a wedding dress holding a bouquet of flowers. 2. A bride with a huge bust looking at the camera. 3. A happy bride holds a bouquet and smiles.	• $\mathcal{M}_3$: A woman is standing *next to a clock*. • $\mathcal{M}_7$: A woman in a wedding dress holding a bouquet of flowers. • $\mathcal{M}_{11}$: A woman in a wedding dress poses with a bouquet of flowers. • $\mathcal{M}_{22}$: A woman in a white dress is holding a bouquet of flowers.
	1. A man and a woman are sitting at a table in a restaurant. 2. Man and woman sitting at a table with glasses. 3.A man and woman are at a table posing for a picture.	• $\mathcal{M}_3$: A man is sitting *on the ground*. • $\mathcal{M}_7$: Two men and a woman sitting at a table in a restaurant. • $\mathcal{M}_{11}$: A group of people that are sitting next to each other. • $\mathcal{M}_{22}$: A man and woman sitting at a table *with a woman and man*.
	1. Black mask layer out on a black carpet with objects. 2. A black cat-eye mask with purple, blue, and pink rhinestones has been placed on a gray rug, just above a row of eight small mechanical components. 3. A black mask with several electronic chips underneath it.	• $\mathcal{M}_3$: *A person is standing in front of a clock*. • $\mathcal{M}_7$: *A pair of black gloves, a pair of black gloves, and a pair of black and gold scissors*. • $\mathcal{M}_{11}$: *A pair of black gloves* and a black mask are on display. • $\mathcal{M}_{22}$: *A pair of black and white shoes* are on a black wall.
	1. A white church sits nestled against a bank of trees 2. A two story white church located near trees. 3. This chapel in the country has a side entrance to the second floor as well as one in the front to the main floor.	• $\mathcal{M}_3$: *A man is sitting on the ground*. • $\mathcal{M}_7$: A small church with a clock tower on top of it. • $\mathcal{M}_{11}$: A white church *with a cross on the front of it*. • $\mathcal{M}_{22}$: A church with *a white building in the background*.
	1. A boat harbor sits primarily empty during dusk. 2. The bridge crossed the water during a sunny day. 3. A bridge over a a river at sunset.	• $\mathcal{M}_3$: *A street sign next to a building*. • $\mathcal{M}_7$: The view of the city from the bridge over the water. • $\mathcal{M}_{11}$: A view of the Golden Gate Bridge in San Francisco. • $\mathcal{M}_{22}$: A view of a city from a bridge over a bay.

Table 13. *Example of captions from different models on images from VIST test set. Highlighted red shows visual hallucinations*

MLP vs Transformer:

The last category of comparison between two sub-parts of Clip-Cap was related to the used mapping network. The two possible selections here were, the *Transformer* & the *MLP* mapping networks. Fig. 26 illustrates the average performance of all those models ($\mathcal{M}_1$ to $\mathcal{M}_{28}$) that either used MLP or Transformer mappers during their evaluation procedure. As we can observe in this instance, there is a slight but constant precedence of the Transformer-based family of networks in all metrics over their MLP counterparts. It is plausible that this could seem as counter-intuitive since (as we saw earlier) most of the top-3 scores happen under the usage of an MLP mapping network. Notwithstanding, the outcome of the automatic metrics here, indicates that in general, Transformer-based Clip-Cap variants, are proven to be slightly more credible on generating image descriptions of higher accuracy.

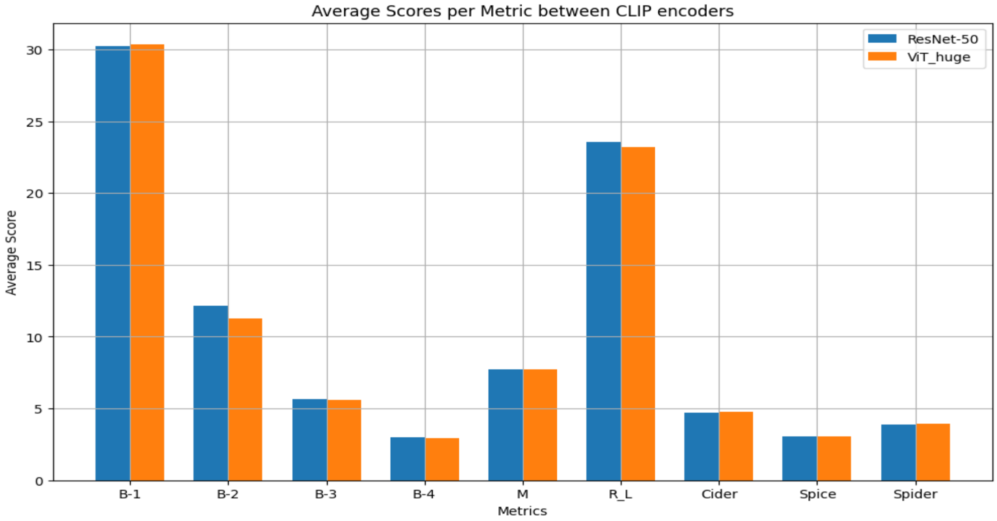


Figure 24. *Average results per metric between models that used ViT$_{Huge}$ & ResNet-50 as image encoders on Clip-Cap*

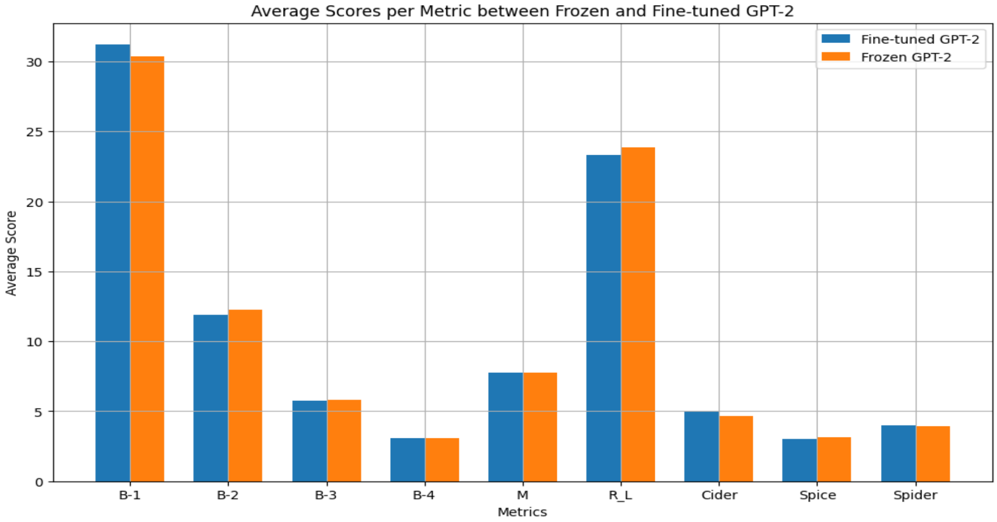


Figure 25. *Average results per metric between Clip-Cap models that use frozen GPT-2 & fine-tuned GPT-2.*

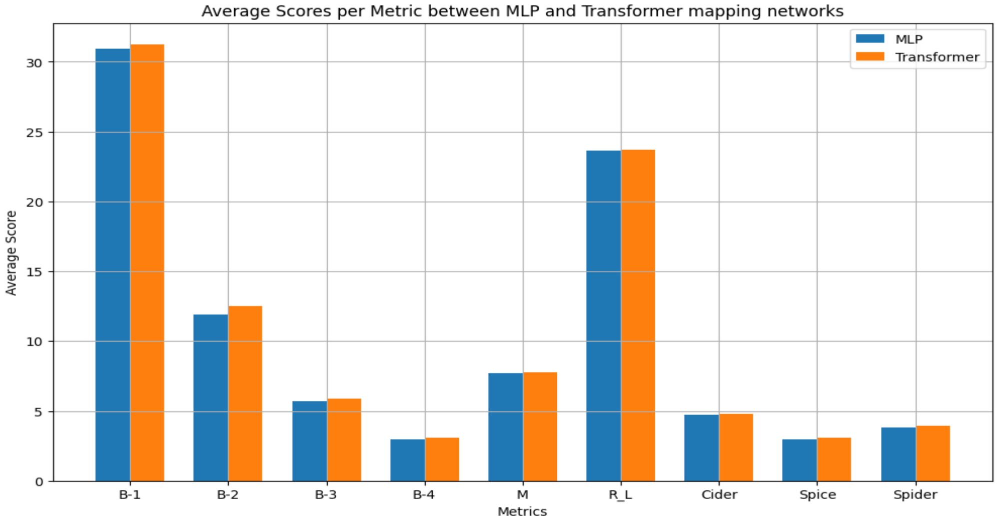


Figure 26. *Average results per metric between Clip-Cap models that used MLP & Transformer based mapping networks.*

5.3 Phase 2 results

Likewise to Phase 1, Phase 2 was unrolled in a multifaceted comparison between the contestant-models. The total number of the generative models exploited during this phase was 14, originating from the combination of different decoding strategies on the families of language models that were presented back in subsection 4.2. The evaluation process in this stage was once again completed with the usage of the automatic metrics, presented earlier in subsection 3.5.

Note that all the involved models were trained on *caption-to-story* generation, with captions/stories coming from VIST/DII/Train. In a same manner, the upcoming results originated from the appliance of the models on VIST/DII/Test. All models, were evaluated upon 3,338 generated storylines. For the automatic evaluation we compared these stories with respective ground truth from the dataset.

5.3.1 Collation within the from-scratch models (T4-family)

Our first level of comparisons, was within the *from-scratch models (T4-family)*. As it was introduced in sub-subsection 4.2.1, this family, consisted of T4$_{base}$ and T4$_{MLM+SP}$ models. For each of those we used the four decoding strategies that were earlier presented: 1) *greedy search (GS)*, 2) *multinomial sampling (MS)*, 3) *nucleus sampling or p-sampling (NS)* and 4) *beam search (BS)*. For the last two, we had to choose for the parameters, *p* and *beam width (k)* respectively. As a result, for all model configurations, we set $p = 0.9$ and $k = 3$. Table 14 outlines the results on the automatic metrics for the eight tested models. With bold marks we state the best score for the metric in the present table, while the increasing number on the parentheses are the abbreviations of the models with their identification number[34].

Method / Metric		B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
T4 Family	T4$_{base}$-GS ($\mathcal{M}_{1'}$)	14.67	4.38	1.73	0.77	6.12	13.62	2.81	4.07	3.44
	T4$_{base}$-MS ($\mathcal{M}_{2'}$)	19.55	5.99	1.97	0.75	7.27	12.94	3.55	4.11	3.83
	T4$_{base}$-NS ($\mathcal{M}_{3'}$)	**19.59**	**6.43**	2.33	0.97	**7.33**	13.52	**4.43**	4.49	4.46
	T4$_{base}$-BS ($\mathcal{M}_{4'}$)	13.51	5.18	**2.43**	**1.29**	5.66	12.46	4.13	**5.33**	**4.73**
	T4$_{MLM+SP}$-GS ($\mathcal{M}_{5'}$)	15.62	4.9	1.9	0.86	6.34	**13.97**	2.65	4.28	3.51
	T4$_{MLM+SP}$-MS ($\mathcal{M}_{6'}$)	19.36	5.78	1.9	0.75	7.11	13	2.86	3.53	3.2
	T4$_{MLM+SP}$-NS ($\mathcal{M}_{7'}$)	18.71	5.86	2.05	0.85	7.01	13.27	3.17	3.76	3.46
	T4$_{MLM+SP}$-BS ($\mathcal{M}_{8'}$)	10.94	3.9	1.77	0.92	4.75	11.29	2.71	3.84	3.27

Table 14. *Evaluation Results for the T4-family (from-scratch models)*

5.3.2 Results from the pre-trained models (T5 & BART)

In contrast to the T4-family's experiments, in this occasion we availed only three decoding techniques. These were *greedy search (GS), nucleus sampling (NS)* with $p = 0.9$ and *beam search (BS)* with beam width $k = 3$. The results are shown on Table 15. Once again, you can see the that best scores per metric are declared on bold as well as the models' identification number is continuing increasing from the previous table.

5.4 Phase 2 discussion

Similarly to Phase 1, in this subsection we will discuss the results from paragraphs 5.3.1 and 5.3.2. This analysis will be unfolded in two directions as well. Firstly, we are briefly going discuss our findings from Tables 14 and 15. Next, we attempt to move the evaluation of models which perform storytelling, outside the canonical comparisons with automatic metrics, and thus we assess their outcome based on the linguistic

[34] We use the "prime" symbol to distinct the identification numbers of models mentioned here, from those reported in the previous section (during phase 1).

	Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
Pre-trained Models	T5$_{base}$-GS ($\mathcal{M}_{9'}$)	18.6	7.7	3.53	1.74	8.45	16.95	10.32	10.14	10.23
	T5$_{base}$-NS ($\mathcal{M}_{10'}$)	21.58	8.24	3.46	1.6	8.83	15.68	9.98	8.89	9.43
	T5$_{base}$-BS ($\mathcal{M}_{11'}$)	19.53	8.4	4.09	2.16	8.44	16.8	11.88	10.28	11.08
	BART$_{large}$-GS ($\mathcal{M}_{12'}$)	20.18	8.48	3.88	1.87	8.88	16.92	11.41	10.22	10.82
	BART$_{large}$-NS ($\mathcal{M}_{13'}$)	**23.5**	**10.19**	**4.81**	**2.44**	**9.71**	**17.25**	**13.72**	**10.55**	**12.14**
	BART$_{large}$-BS ($\mathcal{M}_{14'}$)	22.35	9.65	4.57	2.32	9.44	17.02	13.47	10.5	11.98

Table 15. *Evaluation Results for the pre-trained models*

traits that they showcase. Additionally, we introduce a new metric, named *ideality*, which encloses all these lexical traits and outputs a general score of how well a language model performed during story generation in contrast to a golden reference. Lastly, examples of stories generated from mere captions are also provided.

5.4.1 Discussion on the automatic evaluation

Looking back on Tables 14 & 15, some interesting conclusions can be derived. Firstly, it seems that the enhancement of our T4 model with *MLM* and *SP* methods didn't really render any assistance in our storytelling cause, according the automatic metrics, except for the case of *Rouge_L*. It's easily distinguishable that the majority of the top scores occur within the T4$_{base}$ family and in particular $8/9$ are shared between models $\mathcal{M}_{3'}$ and $\mathcal{M}_{4'}$. Later, we will also verify the generative weakness of T4$_{MLM+SP}$ from Table 16 below, where a family-wise comparison is provided.

Secondly, it's evident that both T5 and BART families, outperform by far our from-scratch models, since there is a significant increase in efficacy for all metrics (again see Table 16). Individually speaking, the BART model with *p-sampling* decoding technique ($\mathcal{M}_{13'}$), stands as the most decorated model, accomplishing the peak scores in all 9 metrics both on the single Table 15 and Tables 14, 15 jointly. In order our comparison to become more tangible, we present the average results per metric for the 4 families of models (as they were given back in Table 7) on Table 16.

This table corroborates the above-mentioned analysis between the pre-trained and the from-scratch model families and how BART models, in general, dominate the scores in every metric. It's remarkable to highlight, that for the more intricate metrics such as *Cider*, *Spice* and *Spider* the gap between the scores of the T4-family and the pre-trained models (T5 & BART) engorges even more to a 3-fold (or 4-fold) difference. On the other hand, as we noted earlier, T4$_{MLM+SP}$ family is affirmed to be the worst performing group of trained models almost across all the listed metrics (not for *R_L*).

Furthermore, Table 16 illustrates the average results that each of the four families of models scored in a specific metric. BART$_{large}$ family is prevailing in all recorded metrics. In addition, watching more carefully the table, we get a nice hint for correlating the size of the language models (LMs) with their performance on the automatic metrics. More precisely, combining its information with the one from Table 7, we can assert that increment on model's parameters can highly guarantee a more accurate *story generation from isolated captions*. This precise interrelation is depicted on Fig. 27, where metrics performance is contrasted with the models parameters, from where we can readily discern the three distinct LM-families along the horizontal axis. This figure, confirms the ascending order in the scores for all metrics, when we go from the most simple to the most complicated LM-families (regarding their trainable parameters).

Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
T4$_{base}$	16.83	5.5	2.12	0.95	6.6	12.88	3.73	4.49	4.12
T4$_{MLM+SP}$	16.16	5.11	1.93	0.85	6.3	12.9	2.95	3.86	3.43
T5$_{base}$	19.67	7.98	3.69	1.86	8.28	16.39	10.39	9.8	10.15
BART$_{large}$	**22.1**	**9.44**	**4.42**	**2.21**	**9.31**	**17.06**	**13.53**	**10.74**	**11.65**

Table 16. *Average Evaluation Results for the four families of models*

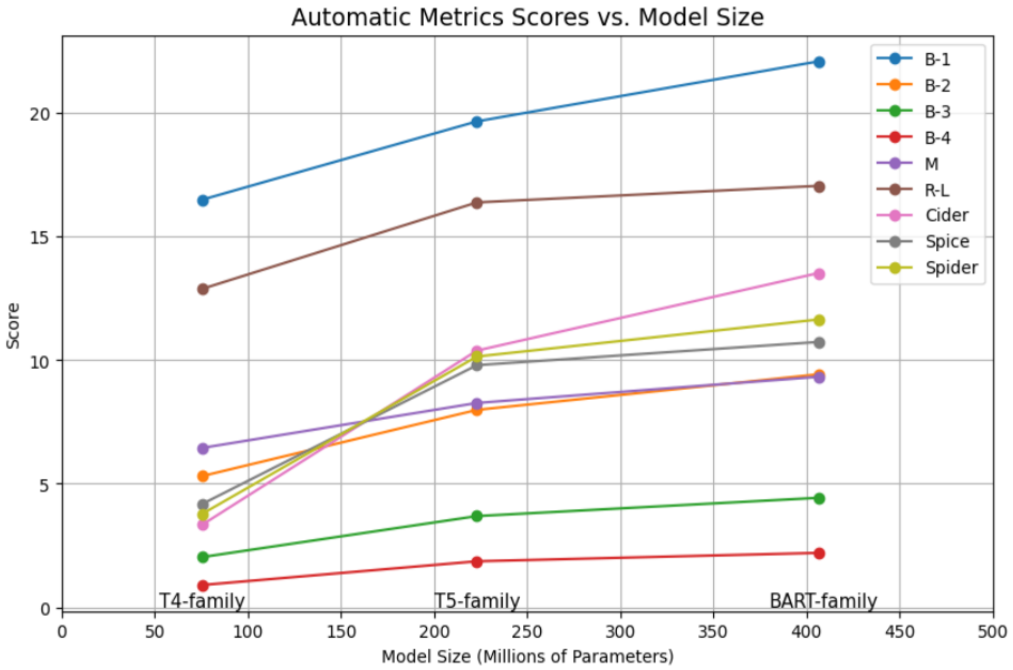


Figure 27. *Average results per metric for the three different kinds of LM families used for story generation from captions. The results per metric, are plotted as functions of the trainable parameters of the three types of LMs*

5.4.2 Looking out of the box

Besides keeping track of the accuracy of a machine generated story, given some automatic metrics, another way to measure completeness of generative system, is to analyze the statistics of the generated stories, such as the length of the stories or sentences and the *part-of-speech (pos)* distribution. Moreover, a particularly intriguing feature is the so-called "lexical creativity" of a model, which essentially corresponds to its capability to produce tokens (words) that it has not seen during training.

Table 17 shows the aggregated results regarding the structural statistics of the generated stories from all methods. In these statistics, we include the *average length of the stories* (in tokens/words), the *average length of sentences* (within the stories) and the percentages of *Nouns*, *Verbs*, *Pronouns* and *Adjectives (ADJ)* in the stories. Additionally, we record the *size of the vocabulary* ($|V|$)[35], the *total number of generated tokens* (N_{tok}) and a quantification of *lexical diversity*. According to [29], a simple way to define diversity is to get the ratio of the vocabulary size over the total number of tokens[36]. Formally, this is equivalent to:

$$Diversity = \alpha \times \frac{|V|}{N_{tok}}, \tag{29}$$

where α is usually an integer constant, which for our case was set to 100. Basically, this metric of diversity calculates the number of distinct unigrams in the generated sentences and normalizes them by the total number of tokens. A higher score indicates greater diversity, as it reflects to a richer variety of unique words.

On the top of the table, with blue hue, we report all these metrics for the *Original Stories* which were obviously written by humans (from VIST dataset) and we utilize their results in order quantify how far from the "ideal" are the machine generated stories in such a case. To that end, for each of the four families of systems, we underline the score that minimizes the absolute distance from the respective human score. The score that minimizes this divergence the most, in the entire column, is given in bold. We can formulate this *closer-to-humans score (ch)*, by using the following equation (for an individual metric):

$$ch = \min_{ms \in \mathbb{M}} |ms - hs| \tag{30}$$

[35] With this we mean the number of unique tokens that each model managed to generate.

[36] Another way to quantify *diversity* is to use the inverse CIDEr score [71, 125]. Since CIDEr computes a weighted n-gram similarity between a generated sentence and a set of reference sentences, then a lower CIDEr score might indicate higher diversity in some contexts because the generated text diverges more from standard references. This divergence could mean less repetition of common phrases or patterns.

In equation (30), we denote with hs the score that the human written stories get on any of the above-mentioned metrics, with ms the corresponding scores that are given to the machine generated stories and finally with $\mathbb{M}$ the total set of methods that are tested (all methods/models appearing on Table 17).

Likewise, we can keep track of how close to humans scores, are on average, the individual model's scores (scores for the entire row). This novel metric is called "ideality" and indicates how far is on average, the machine generated story from the respective human one, on the linguistic features that are presented on Table 17. Formally, we can expound model's *ideality* as:

$$ideality = \sum_{ms,hs \in \mathbb{S}} \frac{1}{\sigma(|ms - hs|)}, \tag{31}$$

where $\mathbb{S}$ is the total set of metrics displayed on Table 17 and σ is the *sigmoid* function. It becomes vivid that for the purposes of our assessment, we would like minimum distance between ms and hs, so to increase the model's "ideality", and thus obtain stories with more similar lexical characteristics to human produced text[37].

	Average Story Length	Average Sentence Length	Nouns (%)	Verbs (%)	Pro-nouns (%)	ADJ (%)	|V|	N$_{\text{tok}}$	Diversity
Original Stories (Humans)	50.57	10.11	19.78	12.65	10.16	6.53	10235	191056	5.38
T4$_{\text{base}}$-GS ($\mathcal{M}_{1'}$)	39.3	7.9	21.68	12.12	6.68	7.53	767	147768	0.52
T4$_{\text{base}}$-MS ($\mathcal{M}_{2'}$)	49.04	**9.99**	19.12	12.91	11.5	7.3	**8187**	184037	**4.45**
T4$_{\text{base}}$-NS ($\mathcal{M}_{3'}$)	45.76	9.41	19.02	**12.63**	11.75	7.38	5121	171813	2.98
T4$_{\text{base}}$-BS ($\mathcal{M}_{4'}$)	37.1	7.43	22.11	12.99	13.87	5.5	412	140616	0.29
T4$_{\text{MLM+SP}}$-GS ($\mathcal{M}_{5'}$)	40.1	8.42	22.92	12.52	4.65	7.89	754	149706	0.5
T4$_{\text{MLM+SP}}$-MS ($\mathcal{M}_{6'}$)	**49.35**	9.96	18.94	12.97	11.96	7.23	6432	**185046**	3.49
T4$_{\text{MLM+SP}}$-NS ($\mathcal{M}_{7'}$)	45.78	9.32	19.19	12.58	11.90	7.36	4468	171807	2.6
T4$_{\text{MLM+SP}}$-BS ($\mathcal{M}_{8'}$)	33.84	6.77	22.61	13.00	13.15	8.63	112	129655	0.09
T5$_{\text{base}}$-GS ($\mathcal{M}_{9'}$)	35.28	7.06	21.24	11.16	7.35	7.23	2402	134682	1.78
T5$_{\text{base}}$-NS ($\mathcal{M}_{10'}$)	42.2	8.51	**20.32**	12.51	10.91	**6.58**	5538	159470	3.47
T5$_{\text{base}}$-BS ($\mathcal{M}_{11'}$)	38.63	7.7	23.23	11.46	8.83	5.27	2371	146187	1.62
BART$_{\text{large}}$-GS ($\mathcal{M}_{12'}$)	36.87	7.37	21.22	11.68	8.58	7.17	2981	141429	2.11
BART$_{\text{large}}$-NS ($\mathcal{M}_{13'}$)	42.31	8.45	21.61	11.94	9.62	6.27	4767	159920	2.98
BART$_{\text{large}}$-BS ($\mathcal{M}_{14'}$)	40.77	8.17	21.44	11.95	**9.66**	6.35	3116	154525	2.02

Table 17. *Lexical characteristics of the human & model generated stories during phase 2*

Analyzing Table 17, we can obtain several conclusions. Firstly, the from-scratch models (T4-family) come closer to the numbers of the Original Stories, in a plethora of metrics including the average story & sentence length, the percentage amount of used verbs, the size of the vocabulary ($|V|$) and the number of generated tokens (N_{tok}). Intriguingly, T4$_{base}$-MS approaches the most human diversity, by hitting a 4.45 score (as diversity was defined on equation (29)), when at the same time humans deliver at almost 5.4.

Secondly, observing the results from the decoding strategies perspective, causes no surprise. In general, more random generative methodologies like *multinomial & nucleus sampling* end up in a greater vocabulary, more lengthy stories and bigger amount of generated tokens, than the more "strict" methods of *greedy & beam search* approaches. Furthermore, we can see that these greedy techniques result in a lower percentage of pronouns, while simultaneously the keep high the amount of nouns[38]. On the contrary, random methods, seek for a more even distribution of the *pos*, fact which contributes to the far greater diversity scores, of those systems, compared to their greedy counterparts.

[37] More details are provided in the following paragraph.

[38] Logical, as these methods attempt to create solid sentences around main *pos* such as nouns, while random sampling methods introduce more variability, increasing the likelihood of pronouns being used for cohesion.

Lastly, opposing the results from Table 17, with those from Tables 14, 15 and 16, we can safely verify that automatic metrics can be a very good indicator of how near we are to a golden reference, but they cannot measure other characteristics that the human text embodies, which can also be very important, such as the lexical diversity or length of passage. In other words, by setting as unit, the absolute distance between the scores of models and the scores of human created stories, for the specific metrics appearing on Table 17, then the T5 and BART families (the previously best performing models), are not the ones that walk up to human stories the most. This reveals that according to the selected lexical features, our from-scratch models showcase a decent level of flexibility and produce more diverse text with also reacher vocabulary.

Besides, we should keep in mind, that not all natural language tasks are the same. For instance, in machine translation, the most important criteria is to produce an accurate, high quality translation of the input; generating a variety of alternative translations is also useful, but not if it comes at the cost of correctness. Meanwhile, on tasks such as ours (story generation), the goal is to create narrative and semantically coherent stories that encompasses even elements of imagination that will make it enjoyable to the reader. Of course, this can lead eventually, to highly differentiated stories than the original ones.

5.4.3 But how ideal can we be?

Earlier in equation (31), we introduced the notion of "ideality" for a model, but we did not elaborate more on this. As that equation imposes, we can use it to measure how far from certain "standard" scores are our models results, in various metrics, for instance those appearing on Table 17. In mathematical terms, we can think of *ideality* as a function of the scores that a model gets on a set of metrics. We can define this function as follows: $I(ms) : \mathbb{R} \rightarrow (n, 2n]$ where n is the norm of set $\mathbb{S}$ (i.e $||\mathbb{S}|| = n$)[39]. In simpler words, n represents the number of elements in $\mathbb{S}$. For our case, we have 9 evaluation criteria, so $||\mathbb{S}|| = n = 9$, and therefore I can get values in the interval $(9, 18]$.

Fig. 28 portrays the ideality scores as a function of all the models which were presented on Tables 14 & 15. For the x-axis we use the identification numbers of the models, while the y-axis gives their ideality scores. For reasons of contrast, we also provide the *human* level of ideality at $2n = 18$ with a red dashed line and the theoretical worst result that a model can get, dubbed as *zero-system* just above $n = 9$, at 9.0000001 with a yellow dashed line. From the foresaid figure, we can see that only 3 models surpass the threshold of $I = 12$ and these are $\mathcal{M}_{2'}$, $\mathcal{M}_{6'}$ and $\mathcal{M}_{10'}$. Given this, it becomes obvious that the more random technique of generation we use (firstly *multinomial*, then *p-sampling*), the more our systems will approach the human levels of *ideality* for the specific lexical characteristics (criteria) presented on Table 17.

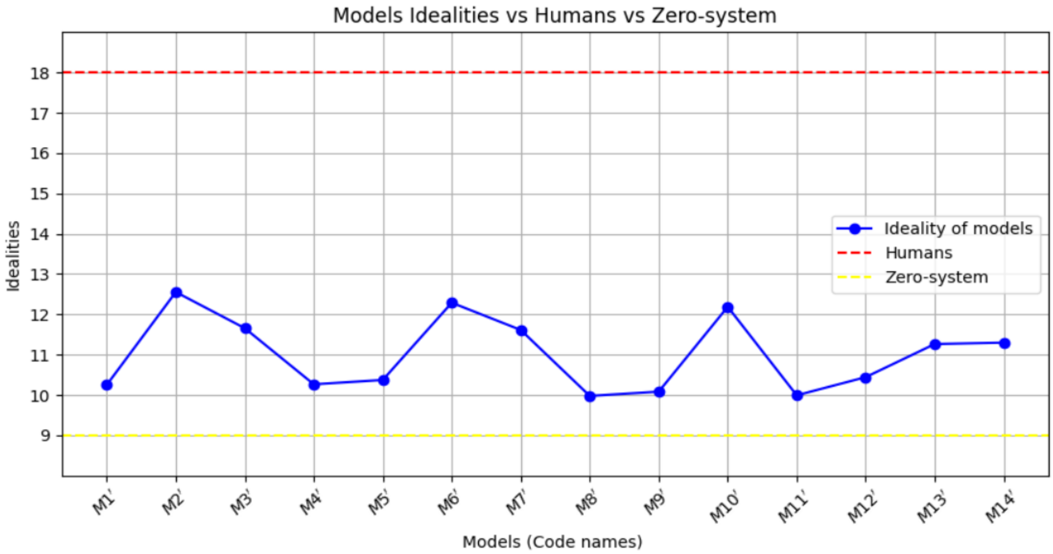


Figure 28. *Idealities of all systems used in phase 2 as function over the models. Also the human-level & the zero-system level of ideality is provided for visual comparison.*

[39] The domain value of I is between n and $2n$ by the definition of the function at eq.31. In the worst case, all of the model's results will be 0 (practically impossible), so the *sigmoid* will take only the human indicators as inputs, and thus we get: $\sum_n \frac{1}{\sigma(|hs|)} = n \times \frac{1}{\sigma(|hs|)} \approx n$, if we always have $|hs| >> 1$ for all human scores. Conversely, if we have a perfect model that succeeds exactly at human level (also impossible), then the inputs of *sigmoid* will be $|hs - hs| = 0$ and thus we get: $\sum_n \frac{1}{\sigma(0)} = n \times \frac{1}{0.5} = 2n$.

Going one step further in our analysis, we investigated which family of models generated the most *out-of-vocabulary tokens* ($tok \notin V$). This term stands for the number tokens that a model generated during test time which were not encountered during training time. Table 18 displays precisely this information, as a percentage of the *total size of the corpus* that each family of models outputted collectively during test time ($N_{tok_{total\_test}}$). We additionally mention here, that for all models the training vocabulary comprised of $44,375$ tokens originating both from the isolated captions (the encoder's input during training) and the target stories (the decoder's input during training).

Model	$\mathbf{N_{tok_{total\_test}}}$	**tok ∉ V (%)**
T4$_{base}$	**644,234**	12.52
T4$_{MLM+SP}$	636,214	12.44
T5$_{base}$	440,339	**13.42**
BART$_{large}$	455,874	12.93

Table 18. *Total size of the collective corpuses produced during test time and percentage of out-of-vocabulary tokens for the four families of models utilized in phase 2*

For one more time, Table 18 showcases that our T4 models produced collectively more tokens than the pre-trained systems[40]. However, the family that attains the highest rank on producing out-of-vocabulary tokens is the T5$_{base}$ family outreaching the barrier of 13%. The only other family that approaches this threshold is the BART$_{large}$ family which stands slightly above 12.9%. At the same time, our T4$_{base}$ and T4$_{MLM+SP}$ models stand around at 12.5% of generated out-of-vocabulary tokens. This is considered to be intuitive, since these two families were the ones which performed the poorest according to automatic metrics (see Table 16).

5.4.4 Examples of generated stories from captions

In this paragraph we are providing some examples of generated stories (from captions), in order to obtain a more tactile view of the abilities our developed systems possess. These stories emanate from all four introduced families of models. Table 19 visualizes five selected of these output stories which were created by four models. These were the models which accomplished the highest scores, if we take into consideration only the automatic metrics (Tables 15 & 16), within their families. In particular, we are talking about $\mathcal{M}_{3'}$, $\mathcal{M}_{7'}$, $\mathcal{M}_{11'}$ and $\mathcal{M}_{13'}$ models, as their identification number appears on Tables 15 & 16. Remarkably, for $3/4$ families, the most superior decoding strategy is the *nucleus technique*.

Noticing Table 19, the first comment that someone could make is that, despite under-performing according to the automatic metrics, our from-scratch models ($\mathcal{M}_{3'}$, $\mathcal{M}_{7'}$) show versatility and produce lengthy and generally meaningful stories equivalent in quality (if not better) to those of the pre-trained models ($\mathcal{M}_{11'}$, $\mathcal{M}_{13'}$). This is an indicator that our T4 models are qualified enough, so to create narrative stories that posses an adequate level of syntactic uniformity and semantic rationality. For this reason, we have underlined some indicative linking words (which boost up the logical flow of a text) that these models generate and quite surprisingly the pre-trained methods generally do not.

The problem seems to arise when we attempt to reflect the generated story back to input basis, i.e the captions. Indeed, in many cases there is not high, or even no correlation, with the input captions and thus the output story diverges significantly from the original one. Naturally, as an aftereffect we have a large registered difference in scores between the competitor models in the automatic metrics. To make this phenomenon more comprehensive, we have highlighted with blue color some key words (from the captions) that the pre-trained models reproduce on their story, while our methods do not.

A special reference needs to be made for the model $\mathcal{M}_{13'}$, which seems to combine both lexical plurality and attentive precision (to the inputs), and as a result, achieves cohesive and accurate stories, something that is of course illustrated also by it's individual scores back in the automatic metrics. Last but not least, the influence of *beam search* is clear, since we can discern high resemblance with respect to the inputs, but in general T5's ($\mathcal{M}_{11'}$) stories are shown to be the shortest and most simplistic compared to the rest.

[40] Nonetheless, keep in mind that the plurality of tokens for T5 and BART models emerged from the results taken by three variants and not four like in the case of the T4 family.

Inputs Captions & Original Stories	Generated Stories
Input Captions: [A man and woman wearing black with painted school faces.], [A man in a hospital gown and bracelet clapping his hands.], [A patient in his hospital gown outside getting some air.], [A man that is wearing a weird outfit in the room.], [All of the costumers stand together to show their creativity.] **Original Story:** Some of our costumes are really amazing today. We are having a party, and everyone has arrived. It is Halloween again, that is not an escaped medical patient. Check out all the cool costumes. We decided to take the party outside since it is so nice.	• $\mathcal{M}_{3'}$: Finally, we got the slide in on the upper level. First, I got the actual shirtless white photo. Then my sister got a little tired. I walked around with all the doughnuts. After that, it was time for some new grand finale fun. • $\mathcal{M}_{7'}$: It is always a great day to celebrate. We even have a small family room to sit together and talk about strangers. My mother got me a big card from dad as well. Even the kids have a chance to take pictures and their little girl taught me how to paint. It was my birthday and I hope my parents turn to join in the fun! my parents decided to take pictures at the end of the day and we had an awesome party. • $\mathcal{M}_{11'}$: Some of the costumes were very unique. I had a great time at the costume party last week. There were a lot of people there. I had a great time there. There were a ton of people there. • $\mathcal{M}_{13'}$: There were a lot of people dressed up for Halloween at the hospital. They had some crazy costumes. Some of them had music playing. It was a very unique event. It took a while for everyone to finish dressing up.
Input Captions: [The view of a long stretch of straight road from inside a vehicle.], [A view from a bus, of a corn or wheat field.], [Cars and trucks are going by cones at a construction site.], [A road that is filled with some cars on the tar.], [The vehicle with fringe on it is driving down a foggy road.] **Original Story:** It was a great feeling being on the open road. It was a great trip and we got to see many places. Well, most of the time anyways! Nothing compares to having a whole road to yourself! We took a drive across the country last summer.	• $\mathcal{M}_{3'}$: Before going to the shop I see a lot of cat perched on it when it was gone. I had to drive the train to town to get a little closer to the subway. I was driving on a bus ride home that I just got to see that I was driving up to it. I ended up being in a seat to get there and take some pictures. • $\mathcal{M}_{7'}$: We had a great time looking at all the local monuments. Afterwards, we went to the local cafe. The weather wasn't perfect. I wanted to take a photograph of all of the beautiful things. I got to take a ride in a nearby town. • $\mathcal{M}_{11'}$: The road was long. The corn field was empty. The construction site was empty. There was a lot of tar on the road. There was a lot of fog on the road. • $\mathcal{M}_{13'}$: This was the beginning of our highway trip. We were heading south. There were some construction going on near our destination. After seeing the construction we decided to follow the road for a while. It was a bit foggy as we drove along.
Input Captions: [A happy looking man in a brown coat with several people in the background.], [A man in a red outfit making a sign with his hand.], [The young man is the someone's wing man for the night.], [The two men are dressed up in Halloween costumes for the party.], [The man is wearing a suit and a fake mustache.] **Original Story:** They were great sports. The thug look was pretty popular. As was the capes and hats. The winner was the thug though. There were a lot of men at the party.	• $\mathcal{M}_{3'}$: He then took some candies of the work, looking back on his cake. He took a picture with his new camera, which surprised them with an amazing look. David took a picture of himself, taking shots of them kissing his camera. He was amazed by the appetizers. • $\mathcal{M}_{7'}$: Here are some of the best assorted stuff. I'm heading to the state park without a hitch. No one was around because they had such a good time. We had a great time, can't wait to do it again. We had many different pictures taken of each other. • $\mathcal{M}_{11'}$: I had a great time at the costume party last night. I had a lot of people show up. Some of them were very creative. There were a lot of people there. I had a great time over there. • $\mathcal{M}_{13'}$:We had a great time and hope to do it again soon. My brother dressed up as a hooker for Halloween. My friends dressed up like their favorite guys from college. My husband and I went as a cowboy and a construction worker, respectively. This is my best friend's boyfriend, he was dressed up too.
Input Captions: [Two workers on an assembly line, preparing some kind of product.], [Rows and rows of oak barrels are lined up in a building.], [The bottles are ready to be consumed on the bar.], [An oak barrel bears the logo of the whiskey that is aging inside of it.], [Eight empty semi trucks with open backs in front of a large brick buildings.] **Original Story:** Two people are seen working on a factory line. The barrels line the walls of this dark space. Bottles of Whiskey are on full display. The barrel displays the name of the factory. A group of shipping containers sits in the parking lot of the factory.	• $\mathcal{M}_{3'}$: I wondered if I would make a difference of things in there's directions. It's easy to pick your shoes and come out to make a difference. We need new material. I just can't believe how high it is! We've got a new worker and it's time to go home. • $\mathcal{M}_{7'}$: There was a huge waterfall today so I decided to travel. The roads were covered with brilliance and various goods. Some of the yards were scary, but overall, it was fun to watch! I went to the quits and technology convention. There was a lot of them there! • $\mathcal{M}_{11'}$: The barrels are lined up and ready to be poured. The barrels are lined up and ready to be poured. The finished product is ready to be poured. The barrels are lined up and ready to be poured. The warehouse is full of barrels ready to be poured. • $\mathcal{M}_{13'}$: The people who work on the barrels are very dedicated. There are many different kinds of barrels that are used in the making of whiskey. The finished product is a very interesting looking bottle. Each barrel is designed to look like it has a specific chemical working on it. A warehouse warehouse is where the whiskey is produced.
Input Captions: [A couple of women are on some stairs.], [Four girls and one man posing for a picture.], [A woman is standing in front of a small window.], [Two teens pose for a photo inside a colorful play center.], [The men are all sitting down in a circle.] **Original Story:** They had a good time spending the day together. The family got together for a day out. They went to a fun museum. There were lots of interesting exhibits that everyone enjoyed. Everyone was worn out by the end of the day!	• $\mathcal{M}_{3'}$: Time for a shower. The boys decided to take a break and have a good time. They played lots of games. The whole gang gathered for a holiday party. They all took a photo together in the end. • $\mathcal{M}_{7'}$: Afterwards, we got together for pictures. I had a lot of fun at the meeting last week. The entire family was there. There were a lot of people there. We had really a lot of fun. • $\mathcal{M}_{11'}$: The girls posing for a picture on the stairs. The group of girls posing for a picture. The girl posing for a picture in the window. The girls posing for a picture inside the play center. The men posing for a picture in the circle. • $\mathcal{M}_{13'}$: We had a great time and can't wait to do it again next year. We got to hang out with some new friends. We spent the weekend at a play center. There were a ton of kids there. The center was full of activities.

Table 19. *Examples of generated stories with captions as inputs from different language models. The underlined parts are linking words or punctuated phrases generated by the from-scratch models that enhance the coherence/narration of a story. Colored words indicate some keywords produced by the pre-trained methods, which reflect back to the input captions, increasing their robustness.*

5.5 Ultimate phase Results - Automatic Evaluation

As we have pointed out numerously, the final purpose of this project is to test if semantically accurate and narrative visual stories can be generated, when we assume the intermediate step of producing isolated captions that will consist the backbone of the final story. To that goal, we introduced our "pipeline-model" of Fig. 14. However, up to this moment, we haven't seen that framework in action. After all, in this subsection, we show the obtained results after deploying the model of Fig. 14 for the task of *visual story generation*. As we have described earlier on subsection 4.3, our "pipeline-model"[41] was utilized only during test time and it was composed of the two best performing models from phase 1 & phase 2. On aggregate, Tables 8, 9 and 10 (or Table 12) along with Tables 14 and 15 indicate that the best individual models from phases 1 & 2, according to the automatic metrics, are models $\mathcal{M}_7$ (for image captioning) and $\mathcal{M}_{13'}$ (for reformulating isolated captions to storylines)[42] correspondingly[43].

Combining the identification numbers of these models and their actual type (the first is a variant of ClipCap and the second a BART variant), hereinafter we will name this model as: *ClipCap*$_7$-*BART*$_{13'}$. The importance of the foresaid model is quite critical, since it is considered to be our best candidate for giving a positive answer to our initial research question (RQ from subsection 1.4). Additionally, continuing on this reasoning, we can verify that the second best captioner from in phase 1 is the model $\mathcal{M}_{11}$ and the runner-up reformulator (from phase 2) is the model $\mathcal{M}_{14'}$. As a result, by combining all the possible top-2 captioners and storytellers we are able to construct three more "pipeline-models" such as the one from Fig. 14. These are: *ClipCap*$_7$-*BART*$_{14'}$, *ClipCap*$_{11}$-*BART*$_{13'}$ and *ClipCap*$_{11}$-*BART*$_{14'}$.

In the following paragraphs, we are taking into consideration, the results given by the four preceding "pipeline-models", based on automatic metrics in order to find out, to what extent our initial research question can be answered positively. In a following subsection (5.7), we additionally consult the more reliable human judgment, towards this goal.

5.5.1 Results from the automatic evaluation of the whole framework

Succeeding the example of all of our previous automatic evaluation procedures, the assessment of the four abovementioned *ClipCap-BART* combinations, was granted after using the same automatic metrics as in phases 1 & 2. It is reminded, that the test stories were 3,338 and as a result the generated visual stories from all models were of the same number. For the evaluation we used the corresponding human written stories given from VIST/Test set. Table 20 elucidates the results according to the automatic metrics, for the visual storylines generated by *ClipCap*$_7$-*BART*$_{13'}$, *ClipCap*$_7$-*BART*$_{14'}$, *ClipCap*$_{11}$-*BART*$_{13'}$ and *ClipCap*$_{11}$-*BART*$_{14'}$. Furthermore, under the "Method" column an identification number (symbol) of each framework is provided in parentheses, whereas with bold the highest scores per metric are emphasized.

Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
ClipCap$_7$-*BART*$_{13'}$ ($\mathcal{F}_1$)	21.52	8.02	3.24	1.44	8.6	14.82	8.99	8.1	8.49
ClipCap$_7$-*BART*$_{14'}$ ($\mathcal{F}_2$)	**21.55**	**9.05**	4.19	2.13	**8.88**	**15.78**	11.52	9.39	10.46
ClipCap$_{11}$-*BART*$_{13'}$ ($\mathcal{F}_3$)	21.31	7.92	3.22	1.37	8.57	14.8	9.2	8.12	8.65
ClipCap$_{11}$-*BART*$_{14'}$ ($\mathcal{F}_4$)	21.18	8.98	**4.21**	**2.16**	8.83	15.74	**11.83**	**9.48**	**10.66**

Table 20. *Evaluation Results from the four (top-performing) combinations of ClipCap-BART models during the ultimate phase. The generated storylines correspond to VIST/Test set actual stories.*

5.5.2 Discussion on the automatic metrics results

Filtering out the outcome from Table 20, a very intriguing deduction can be drawn. Theoretically, the best performing model on the isolated phases 1 & 2, *ClipCap*$_7$-*BART*$_{13'}$, is not the one that generates the most complete stories in the ultimate phase, at least when evaluating under the automatic metrics. Not

[41] The serialization of models from phase 1 & 2 and the deployment of the full architecture only for inference time.

[42] Here the models are mentioned with their identification number provided in the respective tables.

[43] Henceforward, models from phase 1, for image captioning, will be called "captioners", whilst models from phase 2, for storytelling (by reformulating the captions), will be named as "storytellers" or "reformulators".

only this, but the foresaid framework gives the lowest scores on several metrics such as Cider & Spider. The problem seems to lie on the storyteller (BART$_{13'}$) which is a generative model that utilizes *p-sampling* decoding strategy. On the contrary, we can observe that “pipeline-architectures” that exploit a *beam-search* based storyteller (BART$_{14'}$), consistently perform higher than their *p-sampling* counterparts.

Overall, despite being implausible, the framework that consists of the second best captioner (ClipCap$_{11}$) and the second best reformulator (BART$_{14'}$) is the one that generates visual stories, closer to the golden ones, since it gets $5/9$ of the highest scores, including the peaks in all of the most intricate metrics (B-4, Cider, Spice & Spider). Last but not least, we are not able to extract any drastic association between the usage of the 1^{st} (ClipCap$_7$) and 2^{nd} (ClipCap$_{11}$) best captioners, with the final effectiveness of the frameworks (that appear on Table 20) used for visual storytelling.

5.5.3 Lexical traits of the visual generated stories

Mimicking sub-subsections 5.4.2 and 5.4.3, where we found some linguistic traits for the generated stories from isolated captions, here we work on a similar direction for the visual generated stories from the four models present on Table 20. As a result, we measure the same metrics of Table 17 for the four “pipeline-models”. The analytical results are shown on Table 21. Once again, we include the corresponding human written stories for a reference point, for which the scores are the same with Table 17. Per metric (column), scores in bold denote the model that minimized the distance between the machine generated score and the respective human one, according to equation (30). Finally, in the last column of the table, we encompass the ideality score of each model, $I(ms)$, as this is derived by equation (31), calculated for all $ms \in \mathbb{S}$.

	Average Story Length	Average Sentence Length	Nouns (%)	Verbs (%)	Pronouns (%)	ADJ (%)	|V|	N$_{\text{tok}}$	Diversity	I(ms)
Original Stories (Humans)	50.57	10.11	19.78	12.65	10.16	6.53	10235	191056	5.38	18
ClipCap$_7$-BART$_{13'}$ ($\mathcal{F}_1$)	**43.94**	**8.73**	19.88	12.71	11.07	6.15	5007	**166686**	3.01	12.28
ClipCap$_7$-BART$_{14'}$ ($\mathcal{F}_2$)	41.27	8.25	21.47	11.99	**10.31**	5.89	2374	155897	1.53	11.27
ClipCap$_{11}$-BART$_{13'}$ ($\mathcal{F}_3$)	43.63	8.7	**19.76**	**12.65**	11.34	6.29	**5045**	165630	**3.05**	**12.36**
ClipCap$_{11}$-BART$_{14'}$ ($\mathcal{F}_4$)	40.26	7.43	21.17	11.67	10.54	**6.32**	2312	152257	1.52	11.21

Table 21. *Lexical characteristics of the human & model generated stories during the ultimate phase*

Table 21 provides us with another quantitative comparison between the four models utilized during the ultimate phase. However, unlike the case of automatic metrics comparison, where we saw that the storyteller BART$_{13'}$ underperforms, here the frameworks that included this model (*ClipCap$_7$-BART$_{13'}$*, *ClipCap$_{11}$-BART$_{13'}$*) seem to generate stories with lexical characteristics closer to the human ones and as a result, they achieve higher scores of ideality.

Moreover, it is interesting to contrast the above results, with those from phase 2 and Table 17. As we have already mentioned, models in phase 2 were trained with the ground truth captions taken from VIST dataset, while in the ultimate phase the captioning models were given the correlated images from the dataset, generating their own captions, which in turn were given to reformulators for producing the final stories. Therefore, the main difference stands on the fact that now the storytellers, received machine generated captions whilst in phase 2 the had been given the golden captions.

Despite this phenomenon, BART$_{13'}$ storyteller here, seems to produce more varied stories accomplishing higher scores than the respective $\mathcal{M}_{13'}$ model, in ideality (12.32 on average vs 11.29) as well as in other metrics such as diversity (3.03 on average vs 2.98), the size of vocabulary (5026 on average vs 4767) and others. Thus, BART$_{13'}$ shows larger linguistic plurality even when fed with machine generated captions.

On the other hand, this is not the case for the storyteller BART$_{14'}$, the results of which seem to be affected when given machine generated captions in its input. Consequently, this model attains lower scores in ideality (11.24 on average vs 11.31), diversity (1.525 on average vs 2.02), the size of vocabulary (2343 on average vs 3116) and other metrics than its counterpart model $\mathcal{M}_{14'}$.

Similarly to phase 2 analysis, as a final step we examined the *out-of-vocabulary tokens* ($tok \notin V$) that each of the tested frameworks generated, during the ultimate phase. The results are concisely presented on Table 22. Again we indicate the *total size of the corpus* that each framework produced ($N_{tok_{total}}$)[44], since the out-of-vocabulary tokens are expressed as a fraction of this number.

Model	$\mathbf{N_{tok_{total}}}$	**tok** $\notin$ **V** (%)
ClipCap$_7$-BART$_{13'}$	**166,686**	12.47
ClipCap$_7$-BART$_{14'}$	155,897	12.51
ClipCap$_{11}$-BART$_{13'}$	165,630	12.49
ClipCap$_{11}$-BART$_{14'}$	152,257	**12.75**

Table 22. *Total number of generated tokens (total corpus size) per framework and percentage of out-of-vocabulary tokens generated per architecture during the ultimate phase*

Confirmed also from Table 21, *ClipCap$_7$-BART$_{13'}$* is the framework that generates the largest number of tokens. Howbeit, it is noteworthy that *ClipCap$_{11}$-BART$_{14'}$* is the framework that produces slightly the most out-of-vocabulary tokens, despite the fact that both the captioner and the storyteller are not top-1 models (when looking back to phases 1 & 2). In particular, this architecture hits a 12.75% out-of-vocabulary tokens, while all three rest of models stand at around 12.5%.

5.5.4 Examples of generated visual stories

Following yet again the pattern of subsection 5.3 from phase 2, the current paragraph furnishes us with some paradigms of generated visual stories by the four contestant frameworks. These visual stories are illustrated on Figs. 29 & 30, where we have picked some random *story_ids* from the Test set of VIST and we depict the unfolding story over the five given images. On each figure, three visual stories are presented, having also their index (raw) number (ind) from the test set of VIST (out of the total 3338 stories)[45].

What is more, the frameworks now are presented with their identification symbol, provided back on Tables 20 & 21. Along with the machine generated visual story, the original (human written story), is also provided for comparisons. It is important to underscore that all the showed stories unfold over five sentences[46] since the input images are also five. In this way, each sentence of the story semantically corresponds to the respective image in the input. As a result, all the illustrated stories are read horizontally.

Lastly, we emphasize with highlighted red italics, some generated inconsistencies[47] by our frameworks with respect to the input images that form the visual storyline. Examining these two figures, brings us to central topic of this work. This is none else, but to evaluate and analyze the machine generated storylines, after applying the pipeline, that was delivered back on Fig. 14, on a series of correlated images (all of the times the images are five), which conceptually are forming a story.

[44] This time we omit the "test" indicator on the name of the variable, as the ultimate phase is by definition a testing procedure for the frameworks-contenders.

[45] Note that the variables *story_id* and ind are not the same. While the former is a unique id that the creators of VIST gave to each story (e.g *story_id* = 49514), the latter is the plain raw number that these stories have when appearing on the test dataset, and ranges from 0 to 3338 (the length of test-stories set).

[46] This is also the way that the VIST dataset was structured.

[47] With the term "inconsistencies" in a visual story, we mean a clear mismatch between the sayings of the story and the visual inputs. Here, we do not address semantic inconsistencies that may arise, something that the human evaluation could reveal more readily. For example, on the first visual story and the generation from $\mathcal{F}_1$, the last sentence mentions "a sunny day", something that clearly is not the case by looking to the respective image. Contrariwise, in the same story generated by framework $\mathcal{F}_2$, we see in both the second and fourth sentence, the generation of the word "everyone", which is a very vague term. It is more probable that "not everyone was out on the streets", since this would be a major conceptional generalization and for this reason we don't include such occasions of inconsistencies in our highlighted examples.

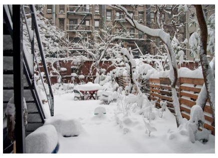
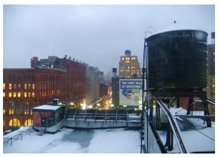
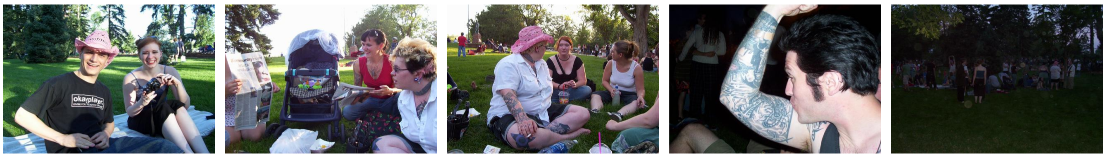
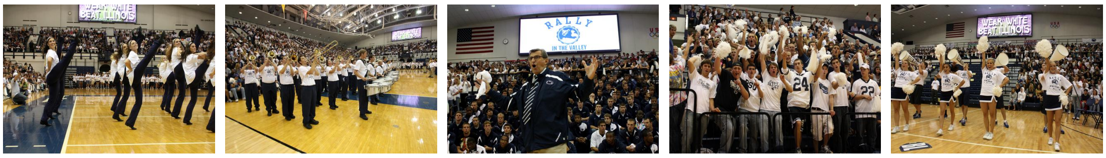
Visual Story: $(ind = 1880)$					
$\mathcal{F}_1$:	I saw that a tree had branches broken off from the weight of the snow.	There was a lot of snow in my area yesterday.	Some people were complaining that the roads were slippery.	When I got to the other side of town I saw *that my car had been buried in snow*.	It was a beautiful *sunny* day out.
$\mathcal{F}_2$:	It was a cold and snowy day in the city.	The streets were covered in snow and everyone was out on the streets.	Even the pets were out playing in the snow.	Everyone was bundled up and ready for the cold weather.	The sky was clear and *the sun was shining*.
$\mathcal{F}_3$:	I had a great time at work yesterday.	There were a lot of cars that were blocked off because of the snow.	There *was* a lot of people that were walking around the city.	Some of the people *were faster than others*.	When I finished my shift I decided to go back home and see what it was about.
$\mathcal{F}_4$:	It was a cold and snowy day in the city.	There were a lot of cars that were stuck in the snow.	We decided to go out for a walk.	It was a lot colder than I thought it would be.	The view from the top of the building was amazing.
Humans:	It is a picture of roof tops.	The street is covered in snow.	A man is shoveling the snow.	People are walking in the snow.	The ground and fence are covered in snow.
Visual Story: $(ind = 2800)$					
$\mathcal{F}_1$:	Then they saw a car that was blue *and white*.	The spectators watched as the cars went by.	The group waited for the cars to pass by.	They saw a car first.	Then another car passed by as well.
$\mathcal{F}_2$:	The winner of the race *waves to the crowd* as he drives past.	A man *stands in front of his car* as he waits for the race to start.	People stand on the side of the road to watch the important race.	The race begins and the cars are going fast.	The blue car is in the lead as the race continues.
$\mathcal{F}_3$:	My friend led most of the way, but he was passing everyone.	The second place car was a Subaru that I had never seen before.	I took my daughters to see their first race this weekend.	My neighbor was third place but it didn't matter much, because he had a good time.	A car drove by me and stopped suddenly.
$\mathcal{F}_4$:	The *white* car took the lead at the start.	The blue car was leading most of the way.	The crowd watched as the cars went by.	The *white* car came around the bend first.	Then the blue car caught up to him.
Humans:	People were standing out in front of the cars putting themselves in danger.	Crossing the finish line, the fans went crazy and the race was over.	The race started and it was dangerous, as always.	The head car rounded the final corner.	The cars took no time flying and weaving around corners.
Visual Story: $(ind = 1352)$					
$\mathcal{F}_1$:	A lot of important people in the military attended this meeting.	There were a lot of pictures taken during the meeting.	Afterwards, everyone got together to talk.	They discussed all the important things they had to do.	They took turns in talking and they had their say.
$\mathcal{F}_2$:	They discussed the drills and how they could be better prepared for the next drill.	The military men gathered for a drill.	They *took a group photo* to remember the day.	They discussed how well they were prepared and how much they were going to be learning.	After the meeting, they had some drinks and talked about what they had learned.
$\mathcal{F}_3$:	It was tense and everyone was there for the same thing.	Today was the last day of our military careers.	And it was hard to take it all in.	But we dealt with it and we shared this moment with each other as best as we could.	We had a meeting to talk about what we would do on the last night.
$\mathcal{F}_4$:	They discussed the drills and how they could be better.	They worked on tying each other's uniforms.	The military men gathered for a drill.	They discussed how they would be better on guard in the future.	After that they discussed how to improve their leading team.
Humans:	One soldier adjusts another's formal fatigues. Appearance is important, especially when going before the committee.	Deckard salutes the esteemed members, showing his respect for them, even though they're all desk jockeys, while he saw real action in the Arabian theatre.	The members of the committee also needed to prepare and they are feeling relaxed and attentive.	Deckard testified to the committee for hours, exhaustively explaining why putting decapitated POW heads on stakes was necessary for morale.	After the deliberation, the committee agreed unanimously to recommend to the Commander that new SOPs should specify decapitation for all.

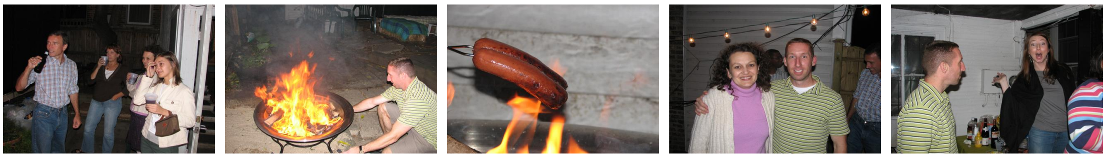
Figure 29. *Examples of visual generated stories by the four tested frameworks along with the golden (human written) storylines. With highlighted red, we note vivid inconsistencies on the frameworks output with regard to the visual input (images).*



Visual Story: $(ind = 1791)$					
$\mathcal{F}_1$:	My husband was happy to sit down and enjoy the day with us.	We all sat and enjoyed the park with good food.	These three sisters got together for a picnic.	We heard a man singing an awesome song.	At night, we all gathered *around the fire* and talked.
$\mathcal{F}_2$:	A couple takes a break from all the excitement of the day to enjoy a *nice lunch*.	A group of friends gather for a day in the park full of fun.	Some of the friends decide to take a break and just enjoy the day.	A man announces to the group that he is going to perform a song for everyone.	At the end of the night, the friends gather together to say goodbye to each other.
$\mathcal{F}_3$:	My boyfriend and I attended a local fair.	There were many people there who had different kinds of tattoos.	It was a fun time and I am glad I went to it.	I had a tattoo of my father on my arm.	The fairgrounds were filled with different types of people.
$\mathcal{F}_4$:	We had a lot of fun sitting around talking.	I went to the park yesterday.	I had a great time there.	I got some tattoos while I was there.	There were a ton of people there.
Humans:	I think I need a new tattoo to commemorate the occasion.	We had the babies there.	At dusk we all did a prayer circle.	We spent the day at the park.	We had the grannies there.



Visual Story: $(ind = 1667)$					
$\mathcal{F}_1$:	Afterwards, we all got up to dance.	There was a lot of singing and people were getting up.	I went to a nice concert last night.	There were a lot of people *sitting* there.	The band played for a long time.
$\mathcal{F}_2$:	There was a lot of dancing.	There were many speakers.	I went to a very big concert last night which had a maestro.	I had a great time there.	They were very good.
$\mathcal{F}_3$:	At the end of the competition, the cheerleaders were cheering on.	There was a large band in attendance that played very well.	The cheerleaders gathered at the school's stadium for a competition.	The *officials* watched as the cheer squad ran on the field.	Then, they went onto the field for another competition.
$\mathcal{F}_4$:	The cheerleaders did a fantastic job during the competition.	The marching band did a wonderful job with their routine.	Everyone was excited for the game to start.	The crowd was very big and supportive.	The cheerleaders were ready to perform.
Humans:	Even the cheerleaders took part.	So they decided to engage in a tournament of group martial arts.	They trained night and day.	In the end, they broke records and won the heart of everyone.	Wilford High had never won a game.



Visual Story: $(ind = 1569)$					
$\mathcal{F}_1$:	We spent time together with my family.	We then roasted *marshmallows* by the fire.	He taught me how to make *hushpuppies*.	I hung out with my boyfriend yesterday.	When we got back from the hike we shared stories.
$\mathcal{F}_2$:	The whole family was there.	We had a cookout.	We also barbecued some hot dogs.	We all had a great time.	It was a very nice party at my friend's house last week.
$\mathcal{F}_3$:	All the family was there to celebrate their loved ones.	Last night's cookout was a lot of fun.	They barbecued tons of meat on the grill.	We invited all of our friends over.	It was a very happy occasion for everyone.
$\mathcal{F}_4$:	All of the family got together to have a cookout.	At the end of the day, we all sat around the fire and enjoyed each others company.	We barbecued hot-dogs and hamburgers.	We had a great time together.	We laughed and told many stories.
Humans:	My niece and Jim seem to be having some creative differences before the show.	The family having a couple of drinks before our annual cookout, which always makes good memories.	Thanks to Jim everyone can now relax by the fire and roast some hot-dogs.	My sister with the pyro-maniac Jim, getting ready to entertain us with some Karaoke.	Jim is always trying to blow something up and looks like that he was definitely the man for this job.

Figure 30. *Examples of visual generated stories by the four tested frameworks along with the golden (human written) storylines. With highlighted red, we note vivid inconsistencies on the frameworks output with regard to the visual input (images).*

From a brief look on Figs. 29 and 30, we can see that for almost all of the depicted storylines, the machine generated stories are on a similar size to the human written ones (the exception is the third visual story from Fig. 29). At the same time, we can also discern that the language and the vocabulary used in the machine generated stories is of the same level as on their human counterparts, again with very few exceptions. A chance for a more extensive, qualitative and illustrative comparison of these stories will be sorted-out in our next subsection, 5.6. Additionally, further analysis on the context of the produced stories will be possible with the results from *human evaluation* (see subsection 5.7). Lastly, more examples of visual stories generated such as those on Figs. 29 and 30, are provided on Appendix B.

5.6 Elaboration upon the generated Visual Stories

The purpose of this subsection is to dig further and expand our analysis on the results that we acquired from paragraph 5.5.4, mainly regarding the visual stories from Figs. 29 and 30, which essentially consist the central product of this project. The breakdown here follows a bottom-up spirit, meaning that we are first looking at the main component of the resulted stories (i.e. their sentences), and then we review them as a complete narrative. This approach will aid us on focusing more readily on the strengths and weaknesses of our constructed frameworks.

5.6.1 Inconsistencies & Within-sentences Analysis

Regarding the visual inconsistencies, with a first look we could argue that it's highly probable that if a model miss-generates a detail in an image then another model may follow, possibly due to the difficulty of the image itself. Some instances of this phenomenon are: Frameworks $\mathcal{F}_1$ and $\mathcal{F}_3$ in the fourth image of the first visual story (from Fig. 29), $\mathcal{F}_1$ and $\mathcal{F}_2$ in the last image of the same story and models $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ on the opening image of the second storyline (from Fig. 29). Staying in the same context, we can observe that these disjunctions are quite similar between the frameworks. For example, $\mathcal{F}_1$ and $\mathcal{F}_3$ both interpret (falsely) that the day was sunny in the last image of the first visual story, while $\mathcal{F}_1$ and $\mathcal{F}_4$ both characterise the displayed car in the first image of the second storyline, as white (instead of blue).

An interesting inconsistency occurs in the first visual story from Fig. 29, where the framework $\mathcal{F}_3$ generated the word "was" instead of "were" (in the third sentence), making a *syntactic* mistake. Nonetheless, these visual inconsistencies fortunately don't spoil neither the narration nor the central message of the sentences. Besides, compared to the amount of generated stories (and the total number of sentences) they are also quite few. To that end, we could argue that in general all frameworks seem to accomplish a decent level of relevance (match) with the provided visual input.

What is more, all models seem to have similar capabilities of *individual storytelling*[48]. In a nutshell, the majority of generated sentences, from all of our tested methods, show a high rate of narration by using words and grammatical structures that us (humans) apply, when we are storytelling and not when we are just describing. Concurrently, these narrations correctly correspond to the visual content of the respective images. For instance, on Fig. 30 and the first visual story, both $\mathcal{F}_1$ and $\mathcal{F}_3$ use the concept of "husband/boyfriend" respectively (for the first image of the story), which is a clear indication of imagination that our models could possess. In the same sentences, it seems that the one recounting the story is a girl, proving that both of these models have captured the existence of two people in the respective image.

In order to make the notion of individual storytelling more comprehensible we created a *graph of relevance* for the generated storyline by the model $\mathcal{F}_1$ for the visual story with $ind = 1791$, which is presented on Fig. 31. In this graph, the color of the directed arrows from the story sentences towards the images point the strength of relevance to the visual input as well as the amount of narrative style that the generated sentence possesses individually[49]. Forsooth, we can confirm that the recitative competence and the attention that the foresaid framework pays to each individual input image is quite *strong*, since all of the arrows indicate links above *moderate* level on average.

[48] With this term, we mean the ability of a model to generate a narrative and also cohesive sentence that will reflect back to the visual input but simultaneously will not be a mere description.

[49] Keep in mind that, this analysis is according to a human evaluator for whom the task was to annotate the color of the arrows, showing the relevance and the narrative of the sentence of each image-sentence pair individually.

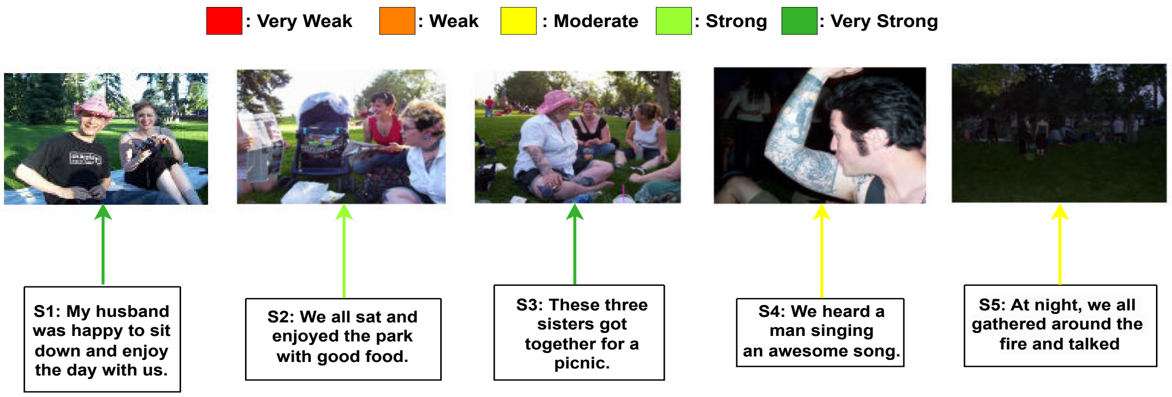

🟥 : Very Weak
🟧 : Weak
🟨 : Moderate
🟩 : Strong
🟩 : Very Strong
S1: My husband was happy to sit down and enjoy the day with us.
S2: We all sat and enjoyed the park with good food.
S3: These three sisters got together for a picnic.
S4: We heard a man singing an awesome song.
S5: At night, we all gathered around the fire and talked

Figure 31. *Relevance graph for the storyline generated by $\mathcal{F}_1$ for the visual storyline with $ind = 1791$. They directed colored arrows indicate the strength of individual narrative capabilities of the tested model as well as the relevance of the generated sentence with the corresponding image.*

5.6.2 Between-sentences Analysis & GPT-4o Evaluation

On the previous paragraph we saw, through an example, that our frameworks succeeded on producing narrative and rational sentences with hidden underground intricacies that any intellectual being could think when observing an image. Hence, we can postulate that our frameworks show *within-sentences* coherence and their generative ability is significantly robust (at least to some extent), considering always a sentence-level evaluation. But what exactly is happening when we turn our attention to the semantic analysis on a story-level and therefore, we examine the *between-sentences* coherence?

In contrast, there are many occasions, while someone that is carefully reading the generated stories, will notice no strict temporal frame that the hypothetical actions are unfolding. The main cause for this could be the absence of linking words (e.g ”After this”, ”Moreover” etc), which can permeate a temporal nature in the evolving story. As a result, judging the stories quite strenuously we could argue that despite the recount texture of each sentence isolated, we can not find human equivalent coherence in the whole story sequence as a total, at least not in all of our tested frameworks.

Let’s take as an example the generated by framework $\mathcal{F}_3$ storyline for the first visual story from Fig. 29 and elaborate more on this. Below, Fig. 32 illustrates the five sentences given by $\mathcal{F}_3$ for the visual story with $ind = 1880$ in a *graph of coherence*. The aim of this graph is to show the logical connectivity and continuity that the sentences of the depicted story have. From each sentence an arrow starts indicating the “logical flow” towards the next sentence. If a sentence is somehow semantically connected to any other (preceding) sentence, then another arrow may be initiated from that specific sentence-box. For instance, we can see a common ground between **Sentence 5 (S5)** and **Sentence 1 (S1)**, since both of them mention or imply the term “work”, although the latter isn’t following the former inside the story. Lastly, the color of the arrow demonstrates the strength of coherence that the two interconnected sentences share[50].

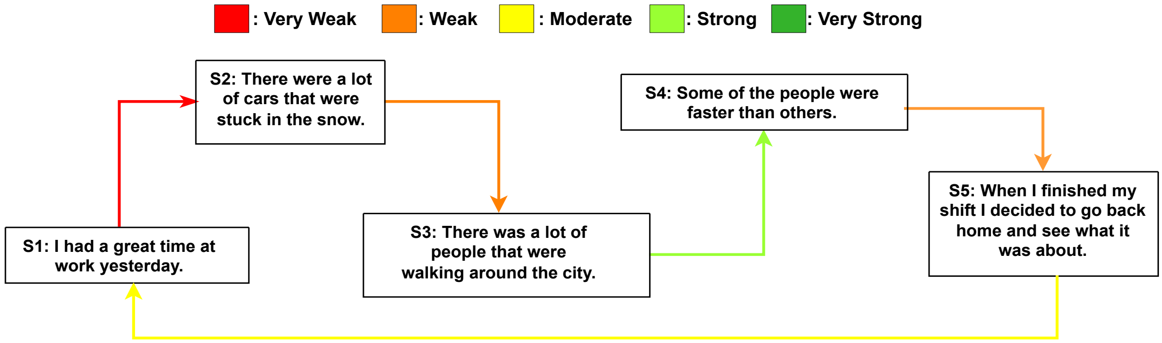

🟥 : Very Weak
🟧 : Weak
🟨 : Moderate
🟩 : Strong
🟩 : Very Strong
S2: There were a lot of cars that were stuck in the snow.
S4: Some of the people were faster than others.
S5: When I finished my shift I decided to go back home and see what it was about.
S1: I had a great time at work yesterday.
S3: There was a lot of people that were walking around the city.

Figure 32. *Coherence graph for the story generated by $\mathcal{F}_3$ for the visual storyline with $ind = 1880$. They directed arrows indicate the logical flow from one sentence to the very next one. If a sentence displays logical connectivity with any other previous sentence then another arrow may arise. The color of the arrows show the strength of coherence that the two sentences share.*

[50] Once again the analysis is according to a human evaluator for whom the task was to annotate the color of the arrows, showing the strength of “logical flow” between the sentences.

Fig. 32 renders vivid that the flow of coherence on the unfolding story is quite *weak*, as most of the sentences are neither semantically connected to their previous ones nor they consist a logical continuation of those. In particular, let's analyze the sentence pairs one by one and justify the weak logical flow of the story:

- **S1 - S2:** While the hypothetical person recounting the story mentions the term "work" in S1, S2 is irrelevant with this concept and highlights the the terms "snow" & "cars". We notice no logical continuity and the two sentences could be easily parts of totally different stories.
- **S2 - S3:** Again the central protagonists of S2 ("snow" & "cars") are absent from S3 which talks about "people walking in the city". However, the close style of the two sentences in terms of describing existing entities ("There were cars", "There was people"), shows a dim piece of semantic similarity.
- **S3 - S4:** They same concepts are present in both sentences ("people walking") and the comparison involving the "the pace of walking", creates a strong semantic bond between the two sentences and gives recitative continuity.
- **S4 - S5:** While the previous sentences (S3, S4) were referring to the concept of "people walking", the thematic topic of S5 changes entirely, and essentially involves a personal narration/action ("finishing the shift") of the hypothetical declaimer of the story. The usage of the linking word "When", slightly connects temporally, the current sentences with the preceding ones.
- **S5 - S1:** The common reference of the concept "work" builds a logical link between S1 and S5, but in practice, their far way distance renders this connection moderate[51]. Also, this common link is absent in all sentences-mediators and thus, the cohesive power of the concept "I was at work" fades away.

The abovementioned analysis on the graph of coherence, showcases that the model which generated the story has a limited skill on synthesising a coherent large text (i.e. story), although each individual piece shows major narrative and attentive (to the respective image) traits.

In order to generalize more our findings, we utilized the objectively acceptable judge of an LLM such as the GPT-4o model [1], for evaluating the transition between the generated sentences by our frameworks in terms of coherence, temporal continuity and logical flow. All these criteria were assessed at once by a weighted judge from the LLM, which for each transition (from a sentence to the very next one), gave a score out of 10. The evaluation process encompassed all the produced storylines from the visual stories appearing on Figs. 29 and 30, including the corresponding human written stories.

On Fig. 33 we have impressed on a *box and whisker plot*, the distribution of the transition scores that GPT-4o gave after appraising the coherence, temporal continuity and logical flow from sentence to sentence for each generated story. Besides the scores distribution, the *median* and *mean* values are also evident. The foresaid figure caters us with a very significant deduction. According to GPT-4o, all of our tested frameworks showcase higher level of coherence, continuity and flow than the respective human stories from the dataset. In particular, the five *mean values* of the transition scores stand at 6.88, 6.79, 6.83, 7.33 and 6.63 for the frameworks $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$, $\mathcal{F}_4$ and `Humans` correspondingly. This proves the slight but existing edge that our models have evenly compared to the human written stories. Likewise, the *median* for $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$ and `Humans` is 7, whereas for $\mathcal{F}_4$ is recorded at 7.5.

On top of that, it's vivid that $\mathcal{F}_4$ is the most robust framework since it shows the highest consistency by achieving coherent transitions between the sentences. This is illustrated by the fact, that the *interquartile range (IQR)*[52] of this model is the only one lying strictly between 7 and 8. In addition, framework $\mathcal{F}_1$ seems susceptible to variability due to the greater range of *whiskers*[53] that is showing, stretching from 4 to 9, while the rest of the models (including humans) have greater consistency in generating more cohesive transitions. Finally, we can notice that for frameworks $\mathcal{F}_2$ and $\mathcal{F}_4$, upper and lower outliers exist respectively which are marked as empty dots. More results and analysis, for this evaluation technique is given on Appendix C.

[51] While someone is normally reading the story, S1 and S5 are separated by the three intermediate sentences, which is considered a long distance dependency in terms of natural language.

[52] The interquartile range (IQR) represents the middle 50% of the data, from the first quartile (Q1 - lower quartile) to the third quartile (Q3 - upper quartile). Essentially, in our figure is the red colored area (box).

[53] The whiskers show the range of the data which are within 1.5 times the IQR from Q1 and Q3. In other words, the lower whisker will extend to the smallest datapoint that is within 1.5 × IQR below Q1 ($Q1 - 1.5 \times$ IQR) and the upper whisker will extend to the largest datapoint that is within 1.5 × IQR above Q3 ($Q3 + 1.5 \times$ IQR).

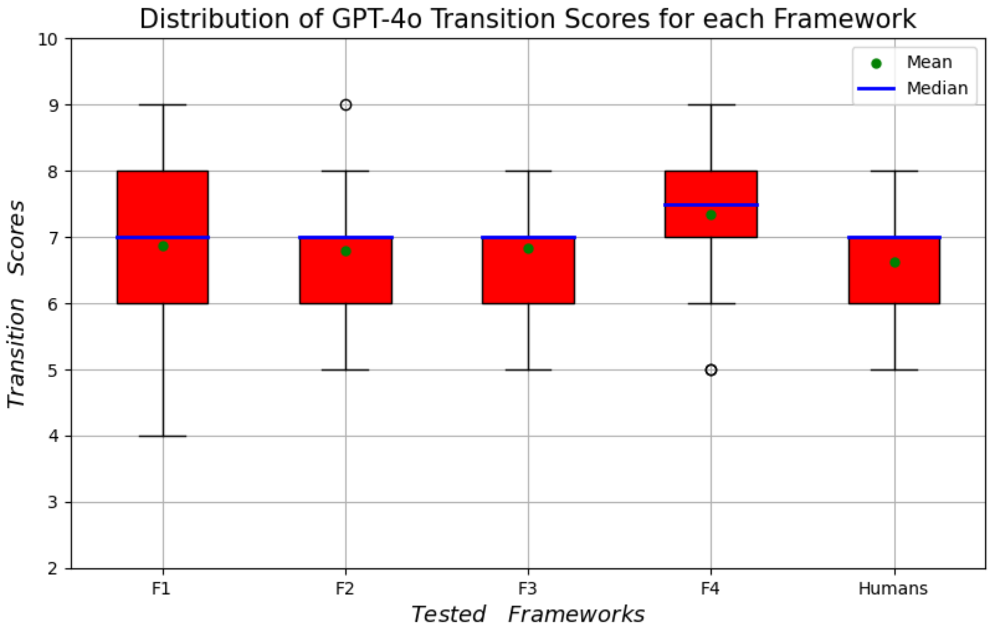


Figure 33. *Distribution of transition scores from sentence to sentence given by GPT-4o for all stories appearing on Figs. 29 & 30. The criteria of assessment on each sentence transition were coherence, temporal continuity and logical flow. The ultimate judge of the LLM was a unified, out of 10 mark for all three criteria.*

5.7 Ultimate phase Results - Human Evaluation

Retracing back in sub-subsection 3.5.2, we introduced the procedure followed when we conducted the human evaluation. In this section, we are going to present our findings. Since the evaluation was split into two sections, the results will be divided accordingly. Firstly, the results from ranking stories based on specific criteria are provided and then, the outcomes from the part of guessing the human written story. We need to note that the total number of participants were $N = 25$.

5.7.1 Results from ranking stories based on specific criteria

As we explained earlier (paragraph 3.5.2) the criteria that we looked upon when the participants ranked the stories were six. These were: 1) *Relevance with the input images (R)*, 2) *Coherence and Flow (C)*, 3) *Narrative Depth (N)*, 4) *Imagination and Creativity (I)*, 5) *Engagement and Interest (E)* and 6) *Language Quality and Style (L)*. Along with that, it should be emphasized that for each of the criteria, two visual stories, such as the one shown on Fig. 23, were utilized[54]. Consequently, this brought up the total number of story-rankings per criterion to 50. The answer (rank of stories) of each participant for one specific criterion had the following form: *1) S3, 2) S4, 3) S2, 4) S5, 5) S1*[55]. Based on this type of response (see footnote 55) and on the fact that the human written story was always "S5", we decided to assess our results on three directions:

- Based on the position that each story (i.e the "model-producer" of the story) got, we award it with a standard number of points. In particular, *Position 1* is awarded with 5 *points*, *Position 2* is awarded with 4 *points*, *Position 3* is awarded with 3 *points*, *Position 4* is awarded with 2 *points* and finally, *Position 5* is awarded with 1 *point*. Our ultimate purpose, is to create per criterion, a unified standings board with the scores of the five contestant models (the four used frameworks + the human story).
- Among all the 50 judges/answers per criterion, we are interested in the absolute number of top-1 spots that each story (i.e the "model-producer" of the story) attained.

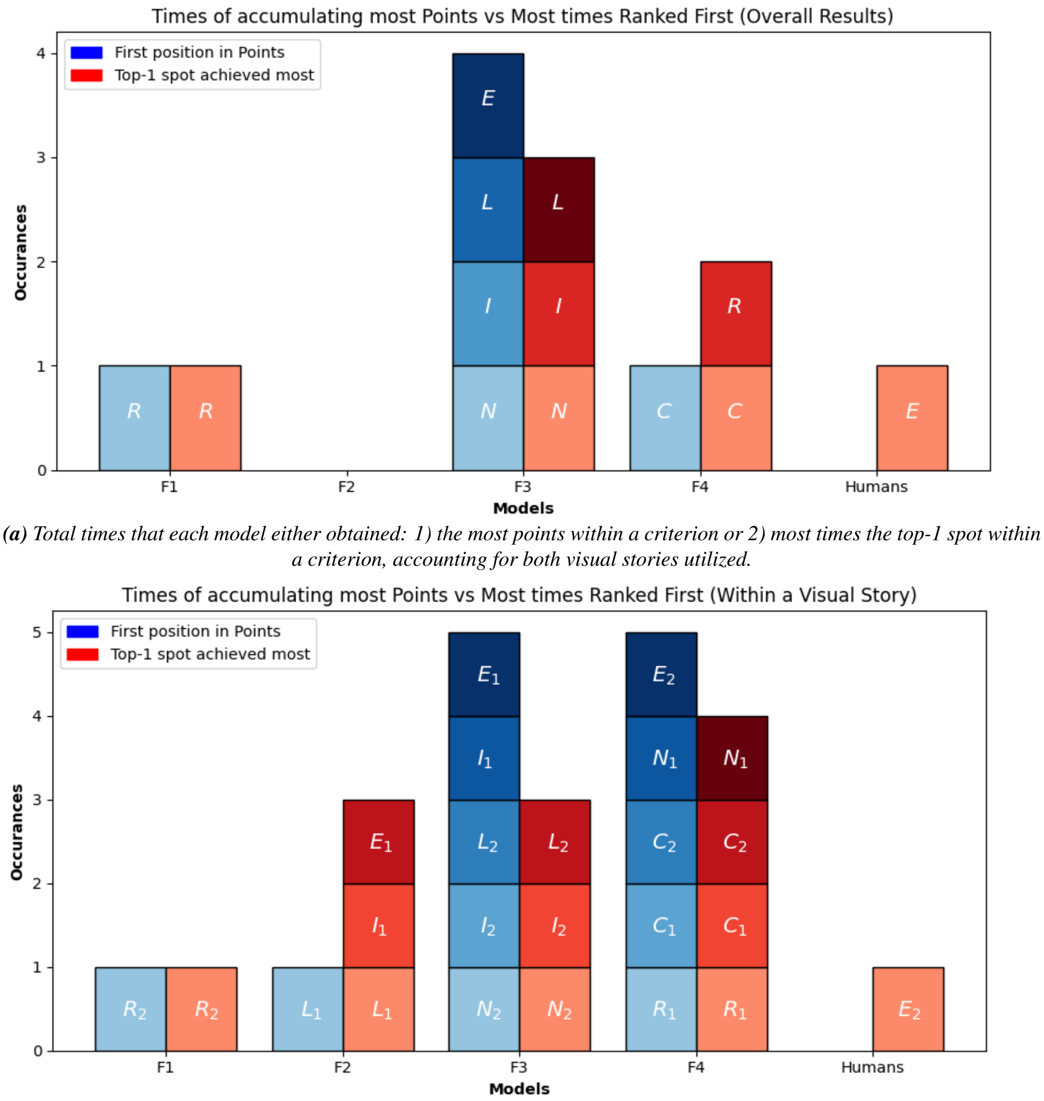
[54] It's important to note here the difference in appearance between Fig. 23 and Figs. 29 & 30. While in the latters, we indicate the "model-producer" of each story along with the human written story, in the former we denote each generated story with the name "Story 1 (S1)", "Story 2 (S2)" etc. Moreover, on Fig. 23, the order of the stories is shuffled making the human written story not to appear always in the last position, as we see it on Figs. 29 & 30.

[55] During human evaluation, for the first part (ranking stories according to specific criteria), in order to preserve the uniformity of the text answers, we did not shuffled the order of stories, but we always used the correspondence: Story 1 (S1) $\rightarrow \mathcal{F}_1$, Story 2 (S2) $\rightarrow \mathcal{F}_2$, Story 3 (S3) $\rightarrow \mathcal{F}_3$, Story 4 (S4) $\rightarrow \mathcal{F}_4$, Story 5 (S5) $\rightarrow$ `Humans`. Meanwhile, in the second part (guessing the human written story), in addition of using the "story-based" names, we also shuffled the order of stories, perplexing the task of participants to detect the human story.

- Among all the 50 judges/answers per criterion, we would like to check the number of times that a machine generated story (i.e S1, S2, S3 or S4) was placed above the respective human story, S5.

Results from position-based rankings & top-1 spot gained per framework:
Table 23 summarizes the results regarding the first two out of the three abovementioned directions. Each sub-table refers to one of the six criteria according to which, our human evaluators judged the stories. We can discern that for all criteria our models (both the frameworks and the human story), were assayed on two visual stories. For each of them, we note the points and the top-1 position that each model managed to fetch. Similarly, the column of "Overall results" simply exhibits the summation of points and top-1 spots taken per contender-model, in the two reported visual stories. Furthermore, for each of the visual-stories columns, the best scores are underlined, while for the overall results the top performance is given in bold.

Probably the most prominent inference that Table 23 offers us, is the fact that all of our frameworks produce equally or most of the times better stories than those written by humans and this is valid for all six of the examined criteria. More specifically, by seeing the "Overall Results" columns, we can become ascertained that $\mathcal{F}_3$ generates the most complete stories in terms of points, on $4/6$ criteria. These are: N, I, E and L. The rest two criteria are shared by frameworks $\mathcal{F}_1$ (R) and $\mathcal{F}_4$ (C). At the same time, in terms of the total times that each framework was placed as first in the overall rank, we have $\mathcal{F}_3$ taking the spot on N, I and L, whilst $\mathcal{F}_4$ on R[56] and C. Lastly, `Humans` manage to barely win the top-1 spot most times, only on E. All this information is provided more illustratively on the bar-plot of Fig. 34a.



(a) Total times that each model either obtained: 1) the most points within a criterion or 2) most times the top-1 spot within a criterion, accounting for both visual stories utilized.

(b) Times that each model, within one of the two tested visual stories, obtained either: 1) the most total points within a criterion or 2) most times the top-1 spot within a criterion.

Figure 34. *Times that each model obtained the most points and most times the top-1 spot across criteria, both in overall account (upper graph) and individually on one of the two visual stories (down graph).*

[56] In this criterion $\mathcal{F}_4$ and $\mathcal{F}_1$ have the same number of times of obtaining the first position.

Relevance with the input images

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	72	2	106	16	**178**	**18**
$\mathcal{F}_2$	87	5	82	4	169	9
$\mathcal{F}_3$	50	1	57	1	107	2
$\mathcal{F}_4$	111	15	62	3	173	**18**
Humans	55	2	68	1	123	3

Coherence and Flow

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	64	0	81	7	145	7
$\mathcal{F}_2$	94	7	67	3	161	10
$\mathcal{F}_3$	64	4	78	3	142	7
$\mathcal{F}_4$	104	12	93	11	**197**	**23**
Humans	49	2	56	1	105	3

Narrative Depth

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	87	2	52	8	139	10
$\mathcal{F}_2$	80	4	73	2	153	6
$\mathcal{F}_3$	81	2	110	16	**191**	**18**
$\mathcal{F}_4$	91	11	60	1	151	12
Humans	36	0	80	4	116	4

Imagination and Creativity

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	85	2	43	0	128	2
$\mathcal{F}_2$	90	13	48	2	138	15
$\mathcal{F}_3$	95	8	100	11	**195**	**19**
$\mathcal{F}_4$	46	0	94	5	140	5
Humans	59	2	90	7	149	9

Engagement and Interest

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	81	6	46	1	127	7
$\mathcal{F}_2$	90	9	16	1	136	10
$\mathcal{F}_3$	96	5	92	7	**188**	12
$\mathcal{F}_4$	44	1	99	7	143	8
Humans	64	4	92	9	156	**13**

Language Quality and Style

Models	Visual Story 1		Visual Story 2		Overall Results	
	Points	*Top-1*	*Points*	*Top-1*	*Points*	*Top-1*
$\mathcal{F}_1$	83	3	39	0	122	3
$\mathcal{F}_2$	103	14	49	1	152	15
$\mathcal{F}_3$	93	5	108	14	**201**	**19**
$\mathcal{F}_4$	40	2	99	8	139	10
Humans	56	1	80	2	136	3

Table 23. *Frameworks comparison across the six criteria judged during our human evaluation procedure. The quantifying parameters were the total number of points that a framework gather based on it's position in each of the human judges and the number of times that it managed to gain the top-1 position in the rank.*

By observing at Fig. 34a, we can verify how many times each model secured the first position in points (bars are given on blue hues for this) or was ranked in the first place the most times (bars are on red hues for this), across the six criteria (*R, C, N, I, E, L*). In a like manner, Fig. 34b yields the same kind of information, but this time when we are looking at each of the two visual stories on which each criteria were evaluated on. To that end, the notation X_i in that figure, where $X \in \{R, C, N, I, E, L\}$ and $i \in \{1, 2\}$, can be translated as: "the criterion X on the first (or second) visual story".

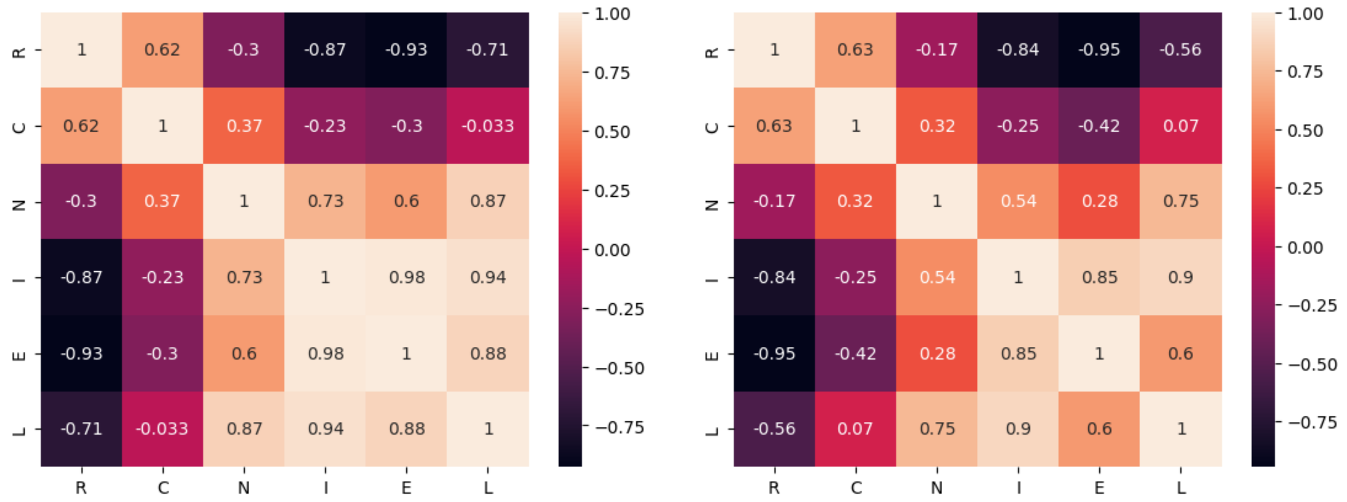
It's significant to pay attention on the main differentiation of Figs. 34a and 34b. From the latter, we could claim that the stories of frameworks $\mathcal{F}_3$ and $\mathcal{F}_4$ are those that our human evaluators have chosen to be the most complete based on the six examined criteria, followed by the stories of framework $\mathcal{F}_2$. However, from the former figure this impression fades out, since on aggregate $\mathcal{F}_2$ is not decorated at all and $\mathcal{F}_4$ accomplishes drastically less best performances than $\mathcal{F}_3$. This lead us to our first comment, i.e that $\mathcal{F}_3$ is overall the most comprehensive framework on generating stories, when judged under our six selected criteria.

Correlation between our measured criteria:

Except of enabling us to see on which criterion our frameworks make it better, Table 23 gives us also the opportunity to examine the correlation between the tested criteria. As randomness may govern the judges of our human annotators, this is a nice chance to reveal any intricate patterns and subsurface relationships that the six criteria may hide. This in-between criteria correlation, which also named *Pearson's correlation coefficient* (r)[57], is given by the following formula:

$$r = \frac{\sum(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum(X_i - \overline{X})^2 \sum(Y_i - \overline{Y})^2}}, \tag{32}$$

where X_i & Y_i are observations of the random variables X and Y, whereas $\overline{X}$ & $\overline{Y}$ are the average values of those random variables respectively. For our case, X_i & Y_i are the points attained by each of the models under two different criteria and $\overline{X}$ & $\overline{Y}$ are the mean points for the models under those criteria. Note that here $i \in \{1, 2, 3, 4, 5\}$, since we have 5 contender-models. Fig. 35a gives the detailed *Pearson's r* correlation matrix, when X and Y account for the total gathered points per framework. If we change our preference, and we also like to include the correlation between the distributed (among models) top-1 spots, then the ultimate matrix is altered substantially (see Fig. 35b).



(a) *Correlation matrix for the six criteria based on the total points.*

(b) *Correlation matrix for the six criteria based on total points and top-1 spots.*

Figure 35. *Pearson's r correlation matrix for the six criteria based: 1) only on the total points gathered per framework (left), 2) on both the total earned points per framework and the total number of times that each framework got the top-1 spot (right).*

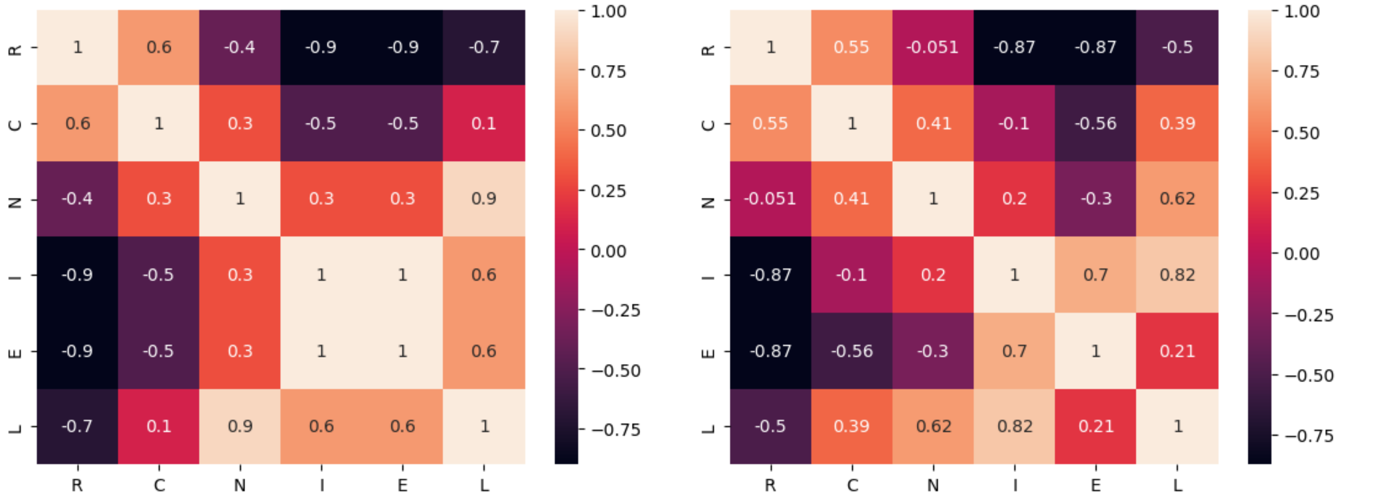
[57] *Pearson's correlation coefficient or simply Pearson's* r [101], is a measure of the linear relationship between two variables. It assesses how well the change in one variable can predict the change in another. The coefficient value ranges between -1 and 1. Values close to $r = 1$ indicate high positive correlation, meaning that as one variable increases, the other increases as well in a linear manner. Values close to $r = -1$ indicate high negative correlation, meaning that as one variable increases, the other decreases in a linear manner. Lastly, values around $r = 0$ indicate no linear correlation between the two variables, meaning changes in one variable do not predict changes in the other. Many similar works to ours, that have included human evaluation in their research, have used Pearson's correlation and other types of correlation metrics. Such works are [12, 137].

Examining Figs. 35 we can distinct two clear patterns. Firstly, the criteria *R* and *C* are correlated strongly on both occasions with scores 0.62 and 0.63 correspondingly. Synchronously, the rest four criteria *N, I, E,* and *L* showcase strong to very strong correlation, when accounting only for the total points, with their scores ranging from 0.6 to even 0.98. This landscape changes significantly when we pay attention to the top-1 spots as well, since we have a decrease of positive correlation for all of these four criteria (now their scores range from 0.28 to 0.9). Last but not least, it's worth mentioning that *R* shows very strong opposite trends with *I* and *E* with an average score of approximately -0.9 on both matrices.

Nevertheless, Pearson's r assumes a linear relationship between the random variables, but this is not always the case. *Spearman's rank correlation coefficient* or simply *Spearman's* ρ [118] is another non-parametric measure[58] of rank correlation, which focuses on how well the relationship between two variables can be described using a monotonic function. It's particularly handful when dealing with ordinal data which show non-linear relations. For the random variables X and Y with n pairs of ranks (observations), Spearman's ρ can be formulated mathematically, as follows:

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}, \tag{33}$$

where d_i is the difference between the ranks of each observation X_i and Y_i[59]. Once again, in our case the random variables X and Y, can represent the total points of each model under one of the six criteria. Analogously to Fig. 35, Fig. 36 pictures the Spearman's ρ correlation matrices when X and Y account only for the total gathered points per framework (Fig. 36a), and when X and Y collectively embody both the total points and the top-1 spots won per framework (Fig. 36b).



(a) Correlation matrix for the six criteria based on the total points. *(b) Correlation matrix for the six criteria based on total points and top-1 spots.*

Figure 36. *Spearman's ρ correlation matrix for the six criteria based: 1) only on the total points gathered per framework (left), 2) on both the total earned points per framework and the total number of times that each framework got the top-1 spot (right).*

The main difference between the two kinds of correlation matrices lies on the fact, that now the criteria *N, I, E* and *L* are not so strongly correlated between each other, especially the pairs: $\{I, N\}$, $\{E, N\}$, $\{L, I\}$ and $\{L, E\}$ marking a significant reduction from 30% to more than 50%. This change is even more profound when we take into consideration the top-1 rankings as extra. In particular, the previously strong correlation between *I* & *N* and *L* & *E* now becomes weak (previously at 0.57 on average, now at 0.2), while *E* & *N* now showcase negative correlation (previously at 0.28, now at -0.3). Nevertheless, the very strong negative relation of *R* & *I* and *R* & *E* remains on both figures. Lastly, from Fig. 36a we notice that according to Spearman's coefficient, now *I* & *E* have a perfect match.

[58] Non-parametric measures are those which make minimal assumptions about the underlying distribution of the studied data. Often these are infinite-dimensional, rather than finite dimensional, as the parametric measures.

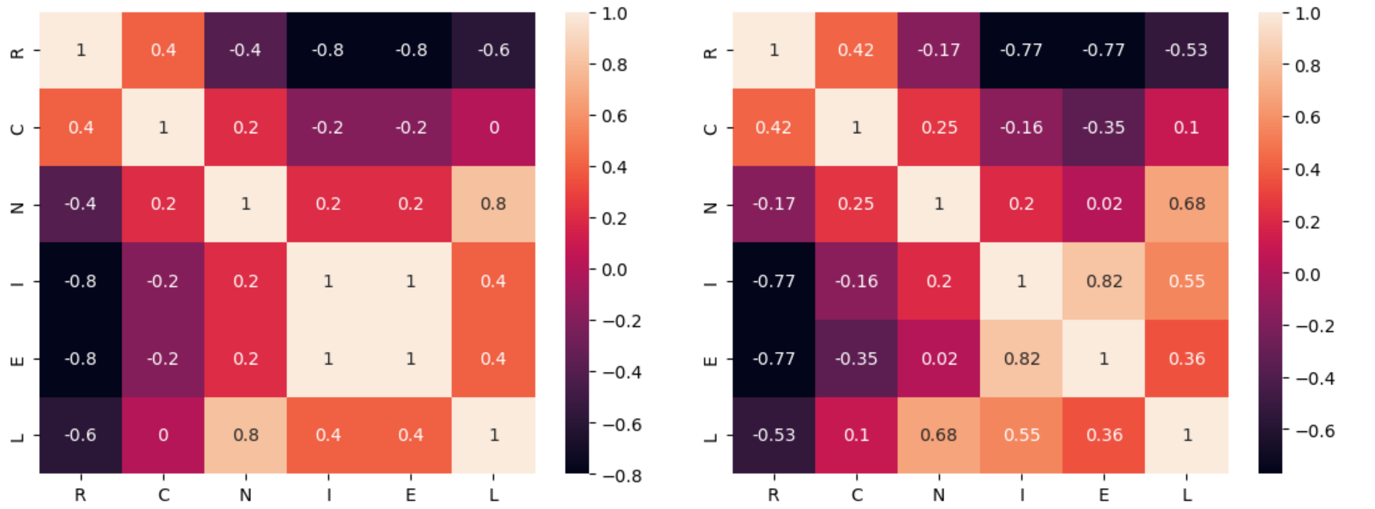
[59] Similar to Pearson's r, Spearman's ρ also ranges between -1 and 1. Values close to $\rho = 1$ indicate high positive monotonic correlation (both variables increase together). Values close to $\rho = -1$ indicate high negative monotonic correlation (while one variables increases the other decreases). Finally, values around $\rho = 0$ suggest no monotonic relationship between the variables.

As a third and final type of correlation, we calculated the so-called *Kendall's rank correlation coefficient* or just *Kendall's τ coefficient* [64]. This is a statistical metric that is used to measure the ordinal association between two quantities. More specifically, this non-parametric hypothesis measures the rank correlation, taking into account the number of concordant and discordant pairs. In terms of interpretation, Kendall's τ is identical to Pearson's r and Spearman's ρ, with values close to 1 indicating high positive correlation, values close to -1 high negative correlation and values around 0 no correlation.

More formally, let $(x_1, y_1), \dots, (x_n, y_n)$ be a set of observations of the random variables X and Y, such that all the values of x_i and y_i are unique. Any pair of observations (x_i, y_i) and (x_j, y_j), where $i < j$, is said to be *concordant* if the sort order of (x_i, x_j) and (y_i, y_j) agrees that: either both $x_i > x_j$ and $y_i > y_j$, or both $x_i < x_j$ and $y_i < y_j$. Otherwise, they are said to be *discordant*. Accordingly, the Kendall's τ correlation coefficient for n data-points, can be defined as:

$$\tau = \frac{(\text{\# of concordant pairs}) - (\text{\# of discordant pairs})}{\text{\# of pairs}} = 1 - \frac{2 \cdot (\text{\# of discordant pairs})}{\binom{n}{2}} \tag{34}$$

Once more, for our example, the coordinates (x_i, y_i) are formed as the combination of the total points that each framework gathered in two random criteria (of the six in total). Similarly to the previous figures of the other correlation coefficients, Fig. 37a portrays Kendall's τ when the x_i and y_i represent only the total points per model, while Fig. 37b, outlines the same information when we also take into consideration the top-1 positions obtained per model.



(a) Correlation matrix for the six criteria based on the total points.

(b) Correlation matrix for the six criteria based on total points and top-1 spots.

Figure 37. *Kendall's τ correlation matrix for the six criteria based: 1) only on the total points gathered per framework (left), 2) on both the total earned points per framework and the total number of times that each framework got the top-1 spot (right).*

Fig. 37a showcases that the correlation between criteria, according to Kendall's τ coefficient is weakened even less than Spearman's case when focusing only on the achieved total points, and this refers especially to the pairs: $\{$*R, C*$\}$ $\{$*I, N*$\}$, $\{$*E, N*$\}$, $\{$*L, I*$\}$, $\{$*L, E*$\}$ and $\{$*L, N*$\}$. For instance, the correlation for *R* & *C* diminishes from 0.6 to 0.4, whereas for *E* & *N* from 0.6 to 0.3. Howbeit, on some of these pairs, such as $\{$*L, N*$\}$, $\{$*E, I*$\}$ and $\{$*L, E*$\}$, the correlation is increased when we take into account also the first positions that were secured by each framework (for example, the correlation for $\{$*L, E*$\}$ raises from 0.21 to 0.36). Simultaneously, the previous strong negative correlation of $\{$*R, I*$\}$ and $\{$*R, E*$\}$ is somehow emasculated, raising from -0.9 and -0.87 formerly, to -0.8 and -0.77 now, respectively for the two matrices. It's worth noting once again, that based only on the total points the correlation of *I* & *E* is perfect as it stands exactly at 1.0.

In-between annotators agreement during human evaluation:

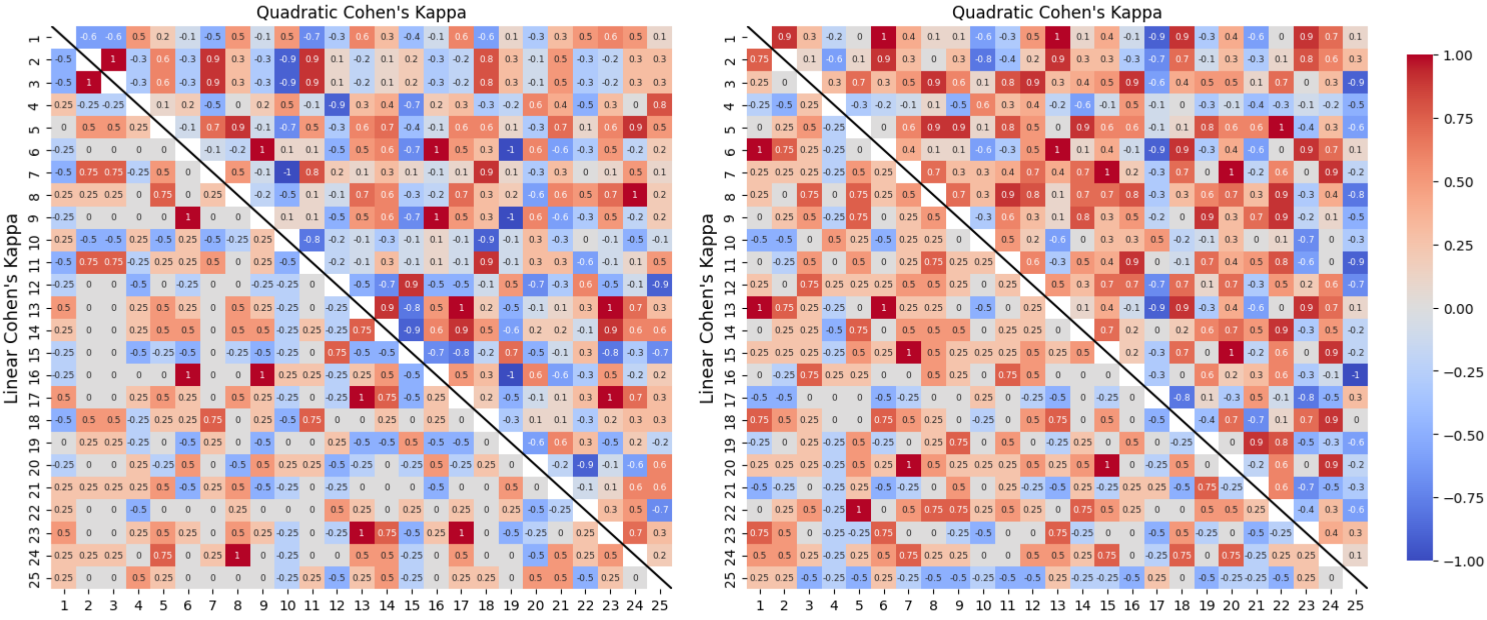
As we already mentioned the randomness that we meet on our human evaluators answers can be remarkably high. For this reason, it's necessitated to check the agreement between various evaluations. This encompasses the agreement between the annotators responses within a visual story during the human evaluation. For quantifying this agreement, since our human rankings of stories are essentially ordinal data, we are going to utilize the *Weighted Cohen's Kappa* statistical measure [70]. Like it's simpler form, *Cohen's Kappa*[60], this metric takes into account the possibility of agreement occurring by chance, which makes it a more robust measure for understanding the true agreement between evaluators. What is more, we would like somehow to capture the degree of disagreement between the human annotators, something which is also dealt by the Weighted Cohen's Kappa (κ_w), as it uses appropriate, both linear and non-linear (quadratic), weights which penalize larger disagreements. This can be formalized with the following way:

$$\kappa_w = 1 - \frac{\sum_{i,j} w_{ij} O_{ij}}{\sum_{i,j} w_{ij} E_{ij}}, \tag{35}$$

where w_{ij} is the weight assigned to the disagreement between the ratings i and j and can be linear or quadratic. In addition, O_{ij} is the observed frequency of the annotators' ratings, E_{ij} is the expected frequency if the annotators rated randomly and finally $\sum_{i,j}$ is the summation over all possible pairs of ratings i and j.

We have seen already that the answers of each annotator came as an ordinal rank, like: *1) S1, 2) S4, 3) S2, 4) S5, 5) S3* and that the total number of annotators per visual story was $N = 25$. As a consequence, in order to find the in-between annotators agreement, we should look at each visual story, within each criterion, separately. This lead us to 25 ordinal comparisons between the ranks for our evaluators and 50 comparisons for one criterion, since each of the latter is comprised by two visual stories.

Figs. 38 jointly visualize the agreement of our evaluators in such 50 ranks, as it shows the weighted Cohen's Kappa coefficient for the two storylines that we used for *R* criterion (the 25 pairs of the first visual story are on the left, and the 25 pairs of the second visual story are on the right). It should be noted, that each correlation matrix depicts the level of agreement between our annotators, using both linear (in its lower triangular part) and quadratic weights (in its upper triangular part). Similarly, Figs. 39, ..., 43 illustrate the same kind of agreement between the annotators for the two visual stories of all other criteria. In particular, the order of the upcoming figures relate to the criteria: *C, N, I, E* and *L*, correspondingly.

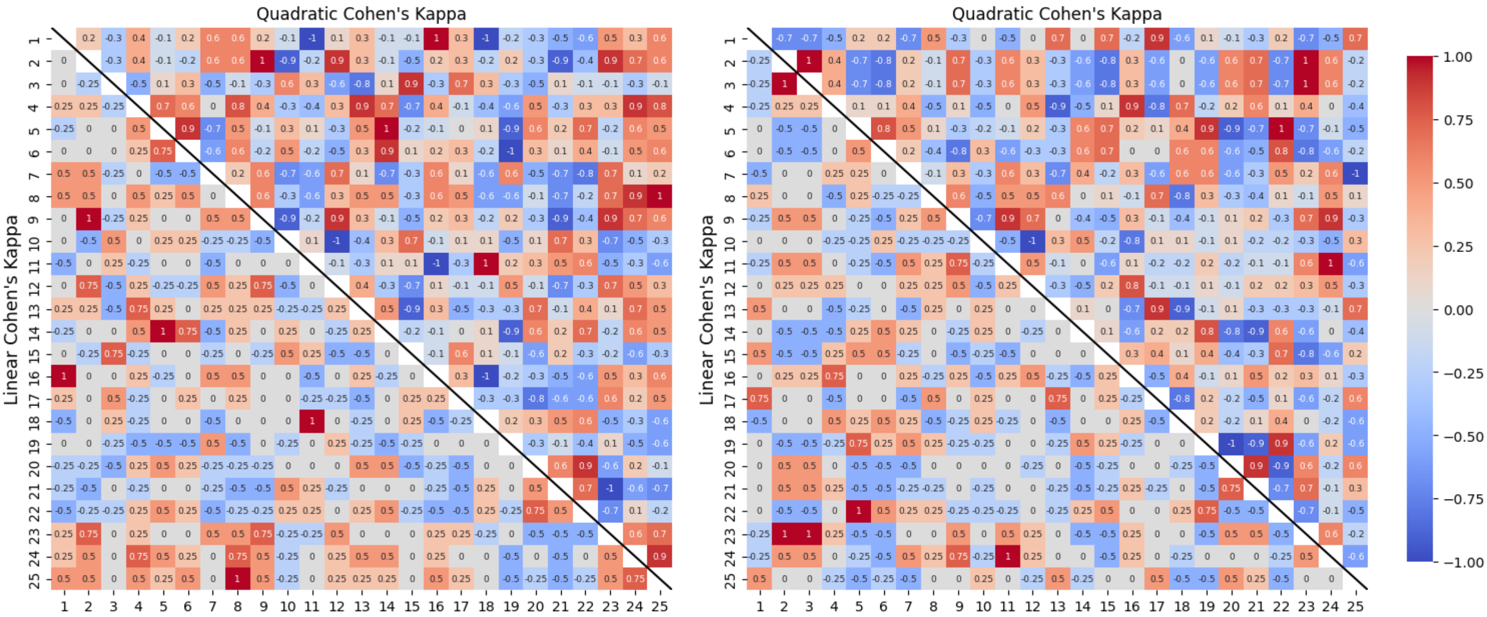


*(a) Correlation matrix of Cohen's Kappa from **visual story 1** of the **"Relevance with the input images"** criterion.*

*(b) Correlation matrix of Cohen's Kappa from **visual story 2** of the **"Relevance with the input images"** criterion.*

***Figure 38.** Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the **"Relevance with the input images"** criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*

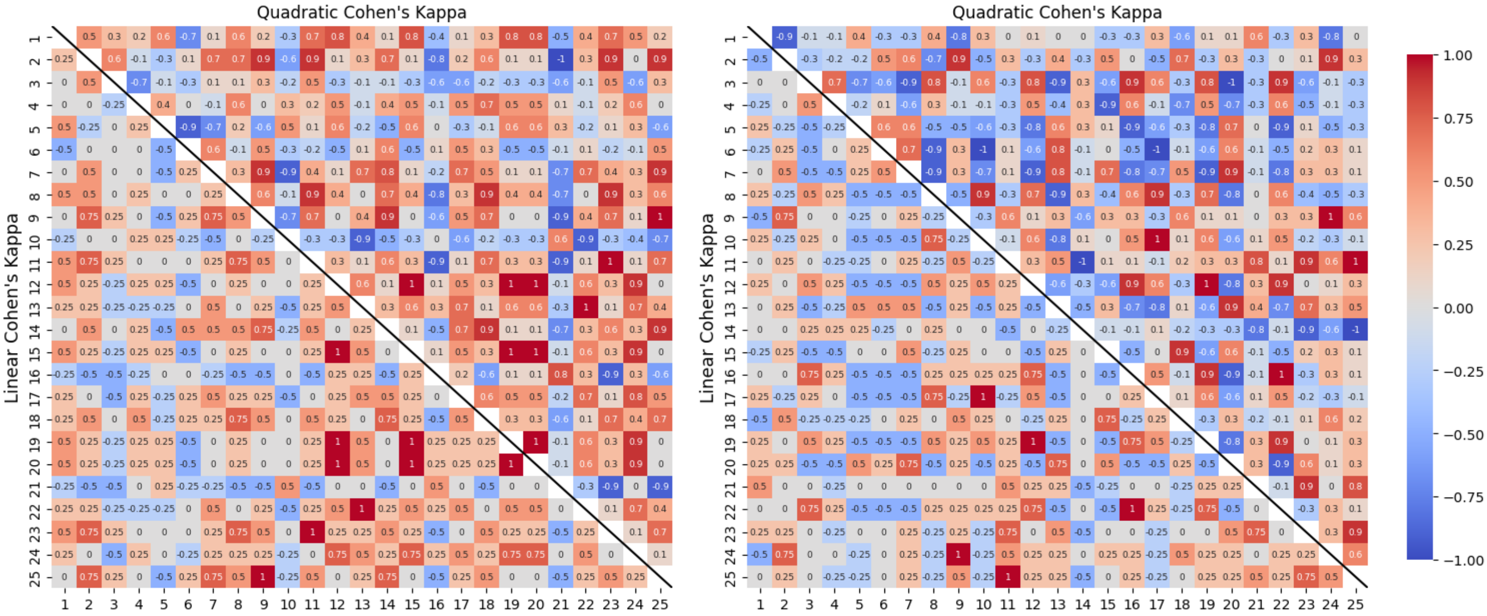
[60] Note that Cohen's Kappa only deals with categorical data annotated by two raters and always uses binary weights (agree/disagree) for the disagreement between the two evaluators. Both variants of Cohen's Kappa (weighted and no) take values from -1 to 1. Values close to 1 indicate strong agreement between the raters, whereas values close to 0 show no agreement beyond what would be expected by chance. Negative values indicate worse than chance agreement. Analytically, the positive values of this metric show: $0 - 0.20$: Slight agreement, $0.21 - 0.40$: Fair agreement, $0.41 - 0.60$: Moderate agreement, $0.61 - 0.80$: Substantial agreement and $0.81 - 1.00$: Almost perfect agreement.



***(a)** Correlation matrix of Cohen's Kappa from **visual story 1** of the **"Coherence and Flow"** criterion.*

***(b)** Correlation matrix of Cohen's Kappa from **visual story 2** of the **"Coherence and Flow"** criterion.*

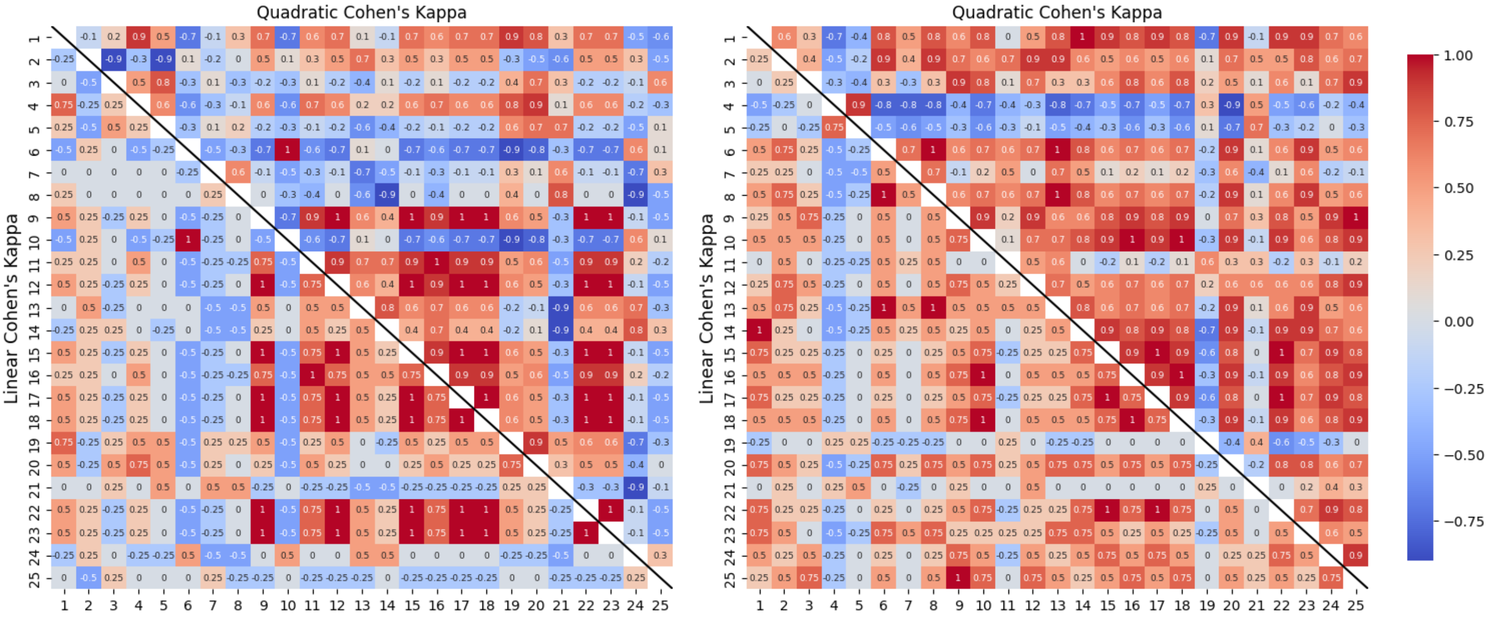
***Figure 39.** Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the **"Coherence and Flow"** criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*



***(a)** Correlation matrix of Cohen's Kappa from **visual story 1** of the **"Narrative Depth"** criterion.*

***(b)** Correlation matrix of Cohen's Kappa from **visual story 2** of the **"Narrative Depth"** criterion.*

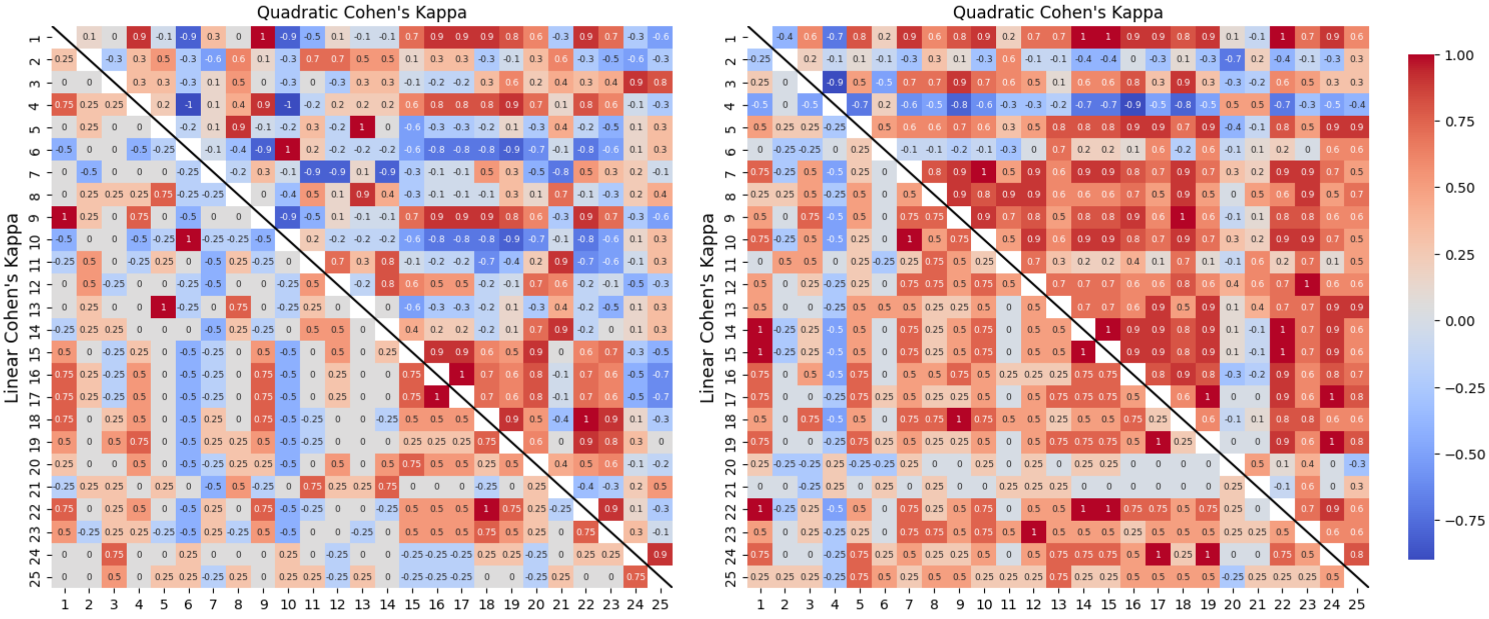
***Figure 40.** Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the **"Narrative Depth"** criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*



(a) Correlation matrix of Cohen's Kappa from ***visual story 1*** *of the* ***"Imagination and Creativity"*** *criterion.*

(b) Correlation matrix of Cohen's Kappa from ***visual story 2*** *of the* ***"Imagination and Creativity"*** *criterion.*

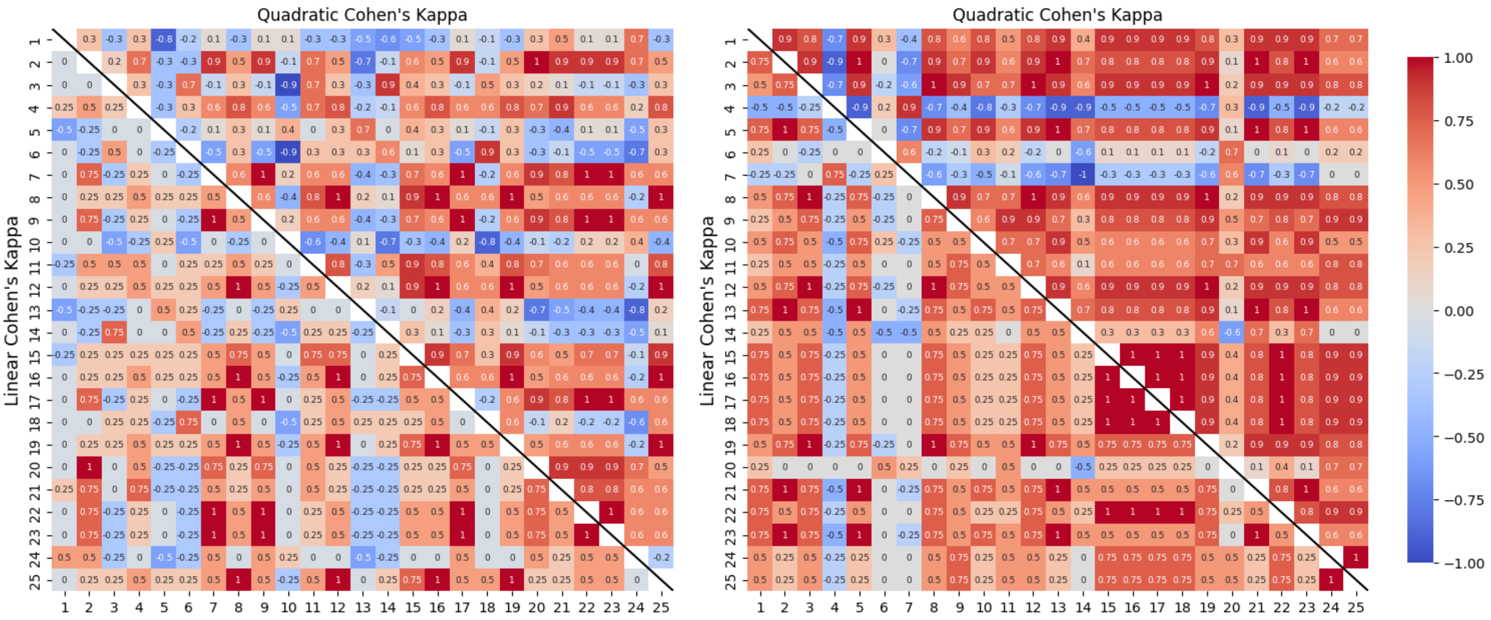
Figure 41. *Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the* ***"Imagination and Creativity"*** *criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*



(a) Correlation matrix of Cohen's Kappa from ***visual story 1*** *of the* ***"Engagement and Interest"*** *criterion.*

(b) Correlation matrix of Cohen's Kappa from ***visual story 2*** *of the* ***"Engagement and Interest"*** *criterion.*

Figure 42. *Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the* ***"Engagement and Interest"*** *criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*



*(a) Correlation matrix of Cohen's Kappa from **visual story 1** of the **"Language Quality and Style"** criterion.*

*(b) Correlation matrix of Cohen's Kappa from **visual story 2** of the **"Language Quality and Style"** criterion.*

***Figure 43.** Linear & Quadratic Cohen's Kappa agreement between the 25 annotators for the two visual stories used for the **"Language Quality and Style"** criterion. The lower triangular matrices show the pair agreement between the evaluators when we use linear weights, whilst the upper triangular matrix show the same information when we use quadratic weights.*

As an overall aftermath from Figs. 38 to 43, we could postulate that indeed quadratic weights capture better the agreement/disagreement between annotators, "painting" the respective triangular matrices more with red and blue colors and not with a neutral hue towards white. Mostly, we can observe that in the quadratic part of the matrices the dominating hue is red, something which intuitively makes sense, since its very likely that our human annotators would agree/disagree on which stories are better/worse per criterion, in most of the cases. More precisely, per individual criterion we could comment the following:

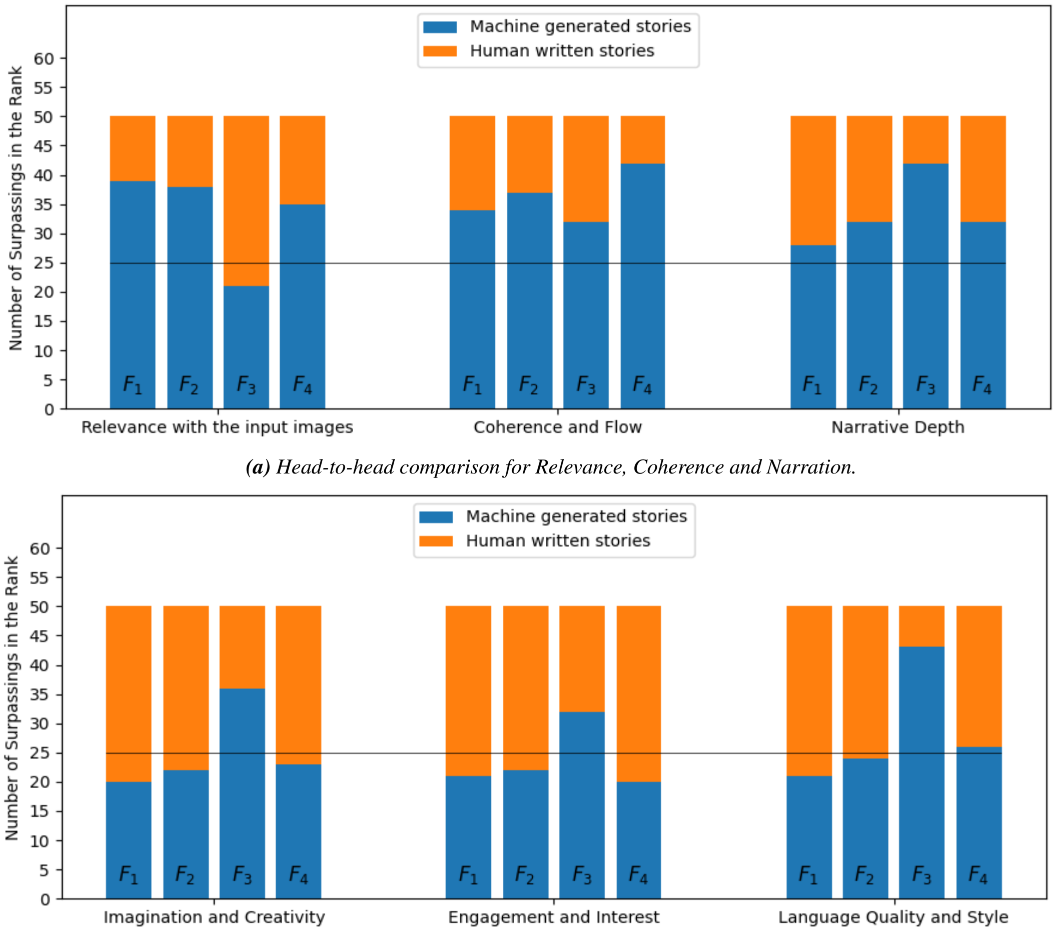
- Figs. 38 indicate that regarding R, our evaluators agreed more in visual story 2 than in visual story 1, since the right figure contains a more reddish shade.
- Regarding C, we notice from Fig. 39a, a quite high divergence in the agreement level of our annotators, something that is enhanced even more in visual story 2 on Fig. 39b.
- Going to Figs. 40 and N, we see that the patter remains similar, as in visual story 1 there is moderate agreement (mostly shown in the quadratic case), while in visual story 2 we could probably say that the one which prevails, is disagreement.
- Fig. 41a shows that for I, visual story 1 features considerable level of disagreement. However, there is a specific group of evaluators that tend to agree in a very high degree. At the same time, in visual story 2 the level of agreement is generally eminent with a very few exceptions (two to three annotators).
- Figs. 42 contrast each other regarding agreement on E, since in visual story 1 blue shade seems to predominate by a few against the red, whilst on the right plot the latter, is far more extensive, setting the agreement level very high. Of course, there are two to three minor exceptions as well.
- Lastly, concerning L, we discern a clear agreement between judges in visual story 1 in Fig. 43a. At once, Fig. 43b records the highest level of agreement from all criteria, as the red color completely overwhelms the blue, with only two annotators indicating otherwise.

Direct comparison of machine generated and human written stories per criterion:

The last direction in which we moved on, encompassed the direct comparison of the stories generated by our frameworks with the human stories provided from VIST dataset. This comparison became possible by recording, per criterion, the times that our produced stories were placed above their human counterparts, while accounting the rankings of all annotators. We know that for each criterion the given ranks were 50, thus we would like to see if our frameworks ($\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$, $\mathcal{F}_4$) and their respective stories (*S1, S2, S3, S4*) were positioned above `Humans` and their story, *S5*, more than 25 times.

This information is precisely visualized on Figs. 44. Namely, on Fig. 44a we see the comparison of machine generated and human written stories in *R, C* and *N* criteria, whereas on Fig. 44b on *I, E* and *L*. On these figures, each individual bar indicates the amount of times that the machine and human stories surpassed each other, on a total comparison of 50 ranks. Additionally, we have annotated a conceptual line at 25, in order to make effortless this contrast. On each bar, whenever the blue hue "pusses" the orange hue above that line, means that the specific model outperformed the human stories, since it was placed more times above them in the 50 ranks. The opposite stands when the orange color "pusses" blue below the conceptual line.

The pattern on these figures is quite distinct. In *R, C* and *N* our frameworks largely outperform the respective human made stories, while in *I, E* and *L* the latters are prevalent. Notwithstanding, there are two points of differentiation here: firstly, on Fig. 44b, there is always the framework $\mathcal{F}_3$ which evidently overcomes all the human stories and secondly, the average margin of ascendance of our frameworks on Fig. 44a is substantially greater than the corresponding average margin of human win on Fig. 44b. Turning our attention to a more semantic interpretation, we could reason that the vocabulary and the ways of expressing a story that our models adopt, are not yet equivalent to those of humans, but in terms of hardened relevance and coherence the accomplished level is considerably high.



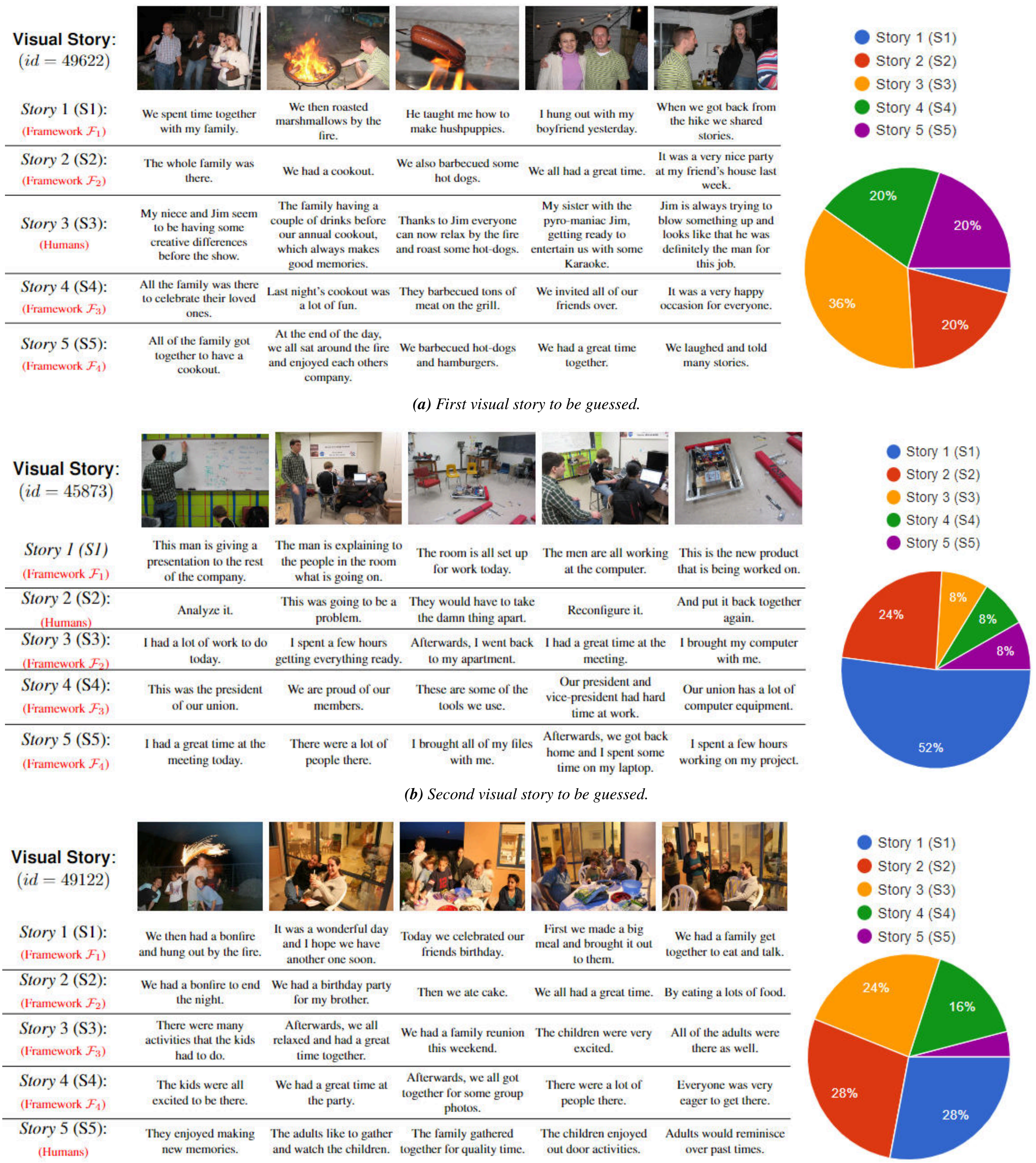
***(a)** Head-to-head comparison for Relevance, Coherence and Narration.*

***(b)** Head-to-head comparison for Imagination, Engagement and Language.*

***Figure 44.** Head-to-head comparison between the machine generated and human written stories based on how many times (over 50 ranks) one was placed above the other by an evaluator. The comparison was held for all six criteria.*

5.7.2 Results from guessing the human written story

For the part of guessing the human written story the participants were given six examples such as the one of Fig. 23. As it was earlier highlighted, during this part, the story-based names of the producing models were used as well and in addition all the stories were shuffled. Ergo, the human story index changed in every visual storyline. Among the five (4 from our models + 1 human) given stories per visual example, the participants should choose, which they believed was written by people. The results for those six visual stories are presented on Figs. 45 and 46 respectively.



Visual Story: ($id = 49622$)					
Story 1 (S1): (Framework $\mathcal{F}_1$)	We spent time together with my family.	We then roasted marshmallows by the fire.	He taught me how to make hushpuppies.	I hung out with my boyfriend yesterday.	When we got back from the hike we shared stories.
Story 2 (S2): (Framework $\mathcal{F}_2$)	The whole family was there.	We had a cookout.	We also barbecued some hot dogs.	We all had a great time.	It was a very nice party at my friend's house last week.
Story 3 (S3): (Humans)	My niece and Jim seem to be having some creative differences before the show.	The family having a couple of drinks before our annual cookout, which always makes good memories.	Thanks to Jim everyone can now relax by the fire and roast some hot-dogs.	My sister with the pyro-maniac Jim, getting ready to entertain us with some Karaoke.	Jim is always trying to blow something up and looks like that he was definitely the man for this job.
Story 4 (S4): (Framework $\mathcal{F}_3$)	All the family was there to celebrate their loved ones.	Last night's cookout was a lot of fun.	They barbecued tons of meat on the grill.	We invited all of our friends over.	It was a very happy occasion for everyone.
Story 5 (S5): (Framework $\mathcal{F}_4$)	All of the family got together to have a cookout.	At the end of the day, we all sat around the fire and enjoyed each others company.	We barbecued hot-dogs and hamburgers.	We had a great time together.	We laughed and told many stories.

(a) First visual story to be guessed.

Visual Story: ($id = 45873$)					
Story 1 (S1) (Framework $\mathcal{F}_1$)	This man is giving a presentation to the rest of the company.	The man is explaining to the people in the room what is going on.	The room is all set up for work today.	The men are all working at the computer.	This is the new product that is being worked on.
Story 2 (S2): (Humans)	Analyze it.	This was going to be a problem.	They would have to take the damn thing apart.	Reconfigure it.	And put it back together again.
Story 3 (S3): (Framework $\mathcal{F}_2$)	I had a lot of work to do today.	I spent a few hours getting everything ready.	Afterwards, I went back to my apartment.	I had a great time at the meeting.	I brought my computer with me.
Story 4 (S4): (Framework $\mathcal{F}_3$)	This was the president of our union.	We are proud of our members.	These are some of the tools we use.	Our president and vice-president had hard time at work.	Our union has a lot of computer equipment.
Story 5 (S5): (Framework $\mathcal{F}_4$)	I had a great time at the meeting today.	There were a lot of people there.	I brought all of my files with me.	Afterwards, we got back home and I spent some time on my laptop.	I spent a few hours working on my project.

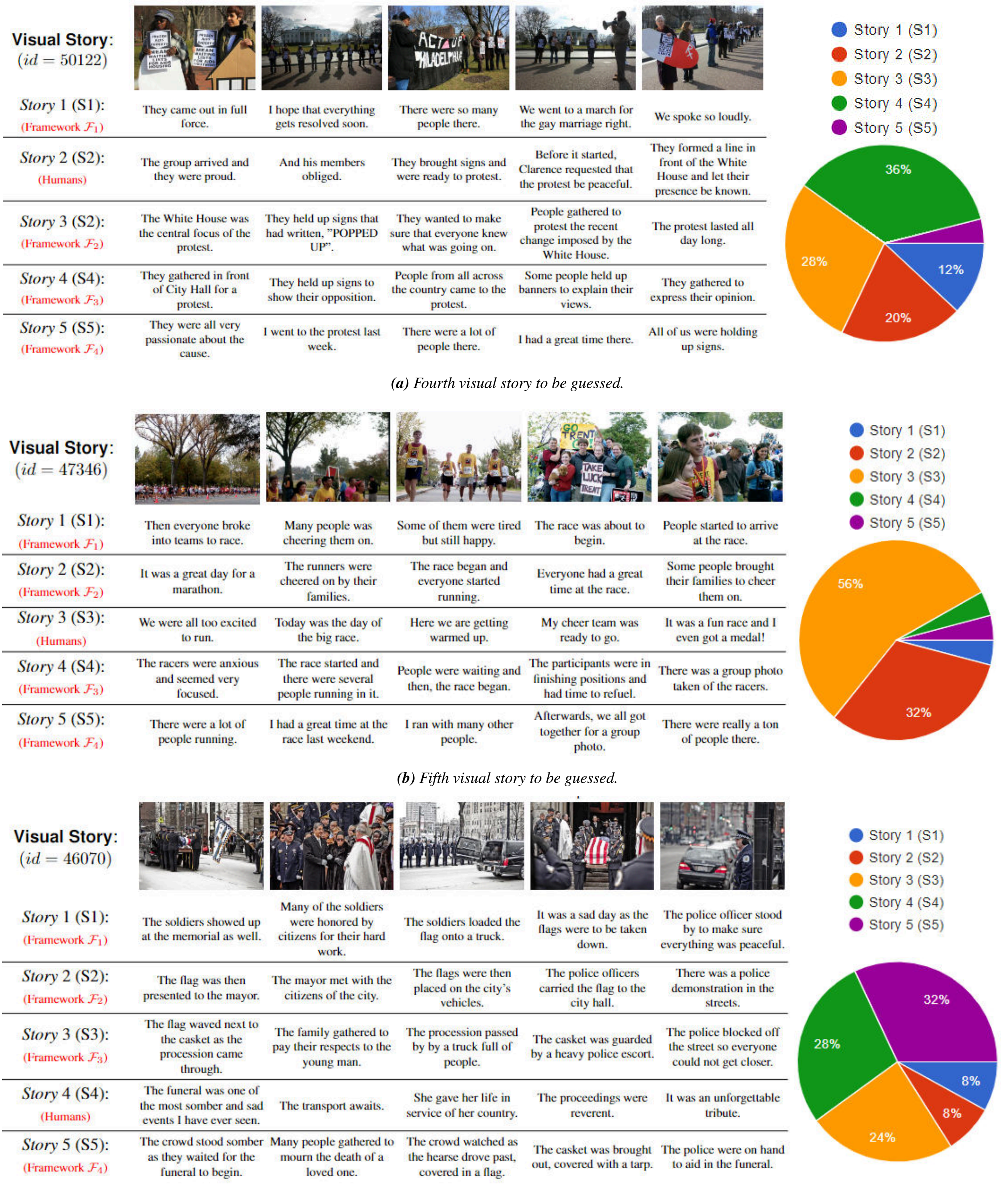
(b) Second visual story to be guessed.

Visual Story: ($id = 49122$)					
Story 1 (S1): (Framework $\mathcal{F}_1$)	We then had a bonfire and hung out by the fire.	It was a wonderful day and I hope we have another one soon.	Today we celebrated our friends birthday.	First we made a big meal and brought it out to them.	We had a family get together to eat and talk.
Story 2 (S2): (Framework $\mathcal{F}_2$)	We had a bonfire to end the night.	We had a birthday party for my brother.	Then we ate cake.	We all had a great time.	By eating a lots of food.
Story 3 (S3): (Framework $\mathcal{F}_3$)	There were many activities that the kids had to do.	Afterwards, we all relaxed and had a great time together.	We had a family reunion this weekend.	The children were very excited.	All of the adults were there as well.
Story 4 (S4): (Framework $\mathcal{F}_4$)	The kids were all excited to be there.	We had a great time at the party.	Afterwards, we all got together for some group photos.	There were a lot of people there.	Everyone was very eager to get there.
Story 5 (S5): (Humans)	They enjoyed making new memories.	The adults like to gather and watch the children.	The family gathered together for quality time.	The children enjoyed out door activities.	Adults would reminisce over past times.

(c) Third visual story to be guessed.

Figure 45. *The first triplet visual stories on which our participants, during human evaluation, had to guess the human written storyline. On each sub-figure, on the column which shows the story index, we also indicate the "producer" of the story with red font. On the right of each sub-figure, the pie-charts show the percentage of annotators who voted each respective story to be the human one (if the percentage is not shown, then it's less than* 5%*).*

Each of these figures presents three visual stories from those used for the guessing task on our human evaluation process. On the most left side column, where the story-number lies, we have also included the "producer" (our frameworks or humans) of the story with red colored letters. Along with it, on the right of each visual story, a pie-chart appears, indicating the percentage of evaluators who believed that each respective story was the one that was written by humans. If the percentage is not listed, then it's less than 5%. From these figures, we could summarize our findings as below:



Visual Story: ($id = 50122$)					
Story 1 (S1): (Framework $\mathcal{F}_1$)	They came out in full force.	I hope that everything gets resolved soon.	There were so many people there.	We went to a march for the gay marriage right.	We spoke so loudly.
Story 2 (S2): (Humans)	The group arrived and they were proud.	And his members obliged.	They brought signs and were ready to protest.	Before it started, Clarence requested that the protest be peaceful.	They formed a line in front of the White House and let their presence be known.
Story 3 (S2): (Framework $\mathcal{F}_2$)	The White House was the central focus of the protest.	They held up signs that had written, "POPPED UP".	They wanted to make sure that everyone knew what was going on.	People gathered to protest the recent change imposed by the White House.	The protest lasted all day long.
Story 4 (S4): (Framework $\mathcal{F}_3$)	They gathered in front of City Hall for a protest.	They held up signs to show their opposition.	People from all across the country came to the protest.	Some people held up banners to explain their views.	They gathered to express their opinion.
Story 5 (S5): (Framework $\mathcal{F}_4$)	They were all very passionate about the cause.	I went to the protest last week.	There were a lot of people there.	I had a great time there.	All of us were holding up signs.

(a) *Fourth visual story to be guessed.*

Visual Story: ($id = 47346$)					
Story 1 (S1): (Framework $\mathcal{F}_1$)	Then everyone broke into teams to race.	Many people was cheering them on.	Some of them were tired but still happy.	The race was about to begin.	People started to arrive at the race.
Story 2 (S2): (Framework $\mathcal{F}_2$)	It was a great day for a marathon.	The runners were cheered on by their families.	The race began and everyone started running.	Everyone had a great time at the race.	Some people brought their families to cheer them on.
Story 3 (S3): (Humans)	We were all too excited to run.	Today was the day of the big race.	Here we are getting warmed up.	My cheer team was ready to go.	It was a fun race and I even got a medal!
Story 4 (S4): (Framework $\mathcal{F}_3$)	The racers were anxious and seemed very focused.	The race started and there were several people running in it.	People were waiting and then, the race began.	The participants were in finishing positions and had time to refuel.	There was a group photo taken of the racers.
Story 5 (S5): (Framework $\mathcal{F}_4$)	There were a lot of people running.	I had a great time at the race last weekend.	I ran with many other people.	Afterwards, we all got together for a group photo.	There were really a ton of people there.

(b) *Fifth visual story to be guessed.*

Visual Story: ($id = 46070$)					
Story 1 (S1): (Framework $\mathcal{F}_1$)	The soldiers showed up at the memorial as well.	Many of the soldiers were honored by citizens for their hard work.	The soldiers loaded the flag onto a truck.	It was a sad day as the flags were to be taken down.	The police officer stood by to make sure everything was peaceful.
Story 2 (S2): (Framework $\mathcal{F}_2$)	The flag was then presented to the mayor.	The mayor met with the citizens of the city.	The flags were then placed on the city's vehicles.	The police officers carried the flag to the city hall.	There was a police demonstration in the streets.
Story 3 (S3): (Framework $\mathcal{F}_3$)	The flag waved next to the casket as the procession came through.	The family gathered to pay their respects to the young man.	The procession passed by by a truck full of people.	The casket was guarded by a heavy police escort.	The police blocked off the street so everyone could not get closer.
Story 4 (S4): (Humans)	The funeral was one of the most somber and sad events I have ever seen.	The transport awaits.	She gave her life in service of her country.	The proceedings were reverent.	It was an unforgettable tribute.
Story 5 (S5): (Framework $\mathcal{F}_4$)	The crowd stood somber as they waited for the funeral to begin.	Many people gathered to mourn the death of a loved one.	The crowd watched as the hearse drove past, covered in a flag.	The casket was brought out, covered with a tarp.	The police were on hand to aid in the funeral.

(c) *Sixth visual story to be guessed.*

Figure 46. *The second triplet visual stories on which our participants, during human evaluation, had to guess the human written storyline. On each sub-figure, on the column which shows the story index, we also indicate the "producer" of the story with red font. On the right of each sub-figure, the pie-charts show the percentage of annotators who voted each respective story to be the human one (if the percentage is not shown, then it's less than 5%).*

- Fig. 45a shows us that the preponderance of our participants indeed guessed correctly the human written story with a percentage of 36%. But even in this case, all frameworks $\mathcal{F}_2$, $\mathcal{F}_3$ and $\mathcal{F}_4$ gathered a significant proportion of votes which stood at 20%.

- On Fig. 45b the absolute majority of votes went for framework $\mathcal{F}_1$ which accounted for a total 52%.

Very far behind on the second place, it's the actual human story which collected 24% of all the votes, while all the rest of stories get lower than 10%

- Fig. 45c is quite interesting where all of our models gathered a greater proportion of votes than the real human written story. In particular, first we have frameworks $\mathcal{F}_1$ and $\mathcal{F}_2$ which both assembled 28% of the total poll, whilst frameworks $\mathcal{F}_3$ and $\mathcal{F}_4$ come third and fourth with 24% and 16% correspondingly. Counter-intuitively the human made story accounted only for 4% of the votes.
- From Fig. 46a, we can affirm once more that the actual human story was not voted as the highest. In fact, the stories generated from frameworks $\mathcal{F}_2$ and $\mathcal{F}_3$ are the prevalent ones, contributing for 36% and 28% of the pie respectively. At the same time the human percentage stands only at 20%.
- Fig. 46b is the only case, where we see that the voting percentage of the human made story is completely dominant, with a total 56%. Besides this, only the generated story from framework $\mathcal{F}_2$ achieves a quantifiable percentage with 32%, whereas all other stories gather negligible percentages.
- Lastly, on Fig. 46c, we observe a close tie between the real human written story and the story produced by framework $\mathcal{F}_4$. From those, the latter accomplishes 32% while the former 28% of the total votes. Moreover, pretty close in the third spot, is the percentage of model $\mathcal{F}_3$ with 24%.

Going to a more qualitative analysis of these results, we could argue that the stories generated from our frameworks pose a serious challenge to our human evaluators, when they attempt to distinguish them from human written text. In this small sample of six visual stories, all of our models raise doubts to our annotators, when guessing, at least once. More specifically, each of our four frameworks accounts for a higher proportion of votes than the human story exactly two times, something which indicates that all of them cause substantial confusion to our evaluators equivalently.

A recapitulation of the results from this procedure can be seen on Fig. 47. On this figure, we depict the average percentage that each “model-producer” got over the six tested stories which we used for this part. In other words, Fig. 47 betokens the mean percentage of participants who believed that the respective story from each “model-producer” was the human written one. Indeed, we verify that the human story itself was the one that most annotators thought to be, not machine generated with a percentage of 28%[61]. Nonetheless, all three frameworks of $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ have percentages around 20%, which correspond to random-baseline since we had five contender-stories.

On the other hand, model $\mathcal{F}_4$ seems to be the least favourable to produce human-like stories, scoring an average percentage of only 14%. This comes in contrast with our findings from Fig. 34b, where we saw that $\mathcal{F}_4$ obtained the most top-1 positions based on our measured criteria. In any case, despite this fact and the tenuous precedence of human family of stories, Fig. 47 declares that it's still quite difficult for a human evaluator to distinct our machine generated stories from the actual manlike stories, a factor which in turn, indicates that our frameworks can produce narratives with human-like characteristics.

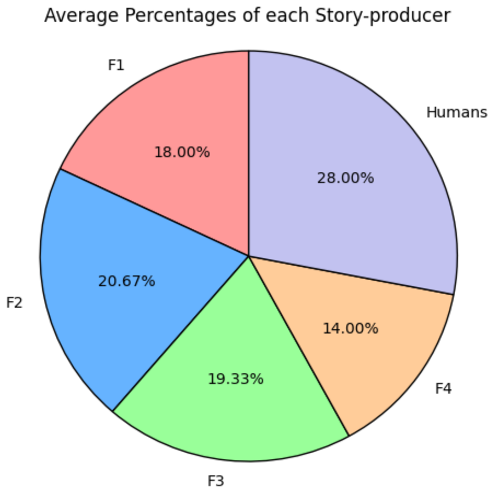


Figure 47. *An aggregated pie-chart of the average percentages of each "model-producer" over the six tested visual stories which were used int the "guess of the human story" part of the human evaluation.*

[61] But this highly attributed to the fact that on the fifth visual story the guesses for the candidate human story reached 56%.

5.8 Overall discussion for the end-to-end architecture

In this final part of our result chapter, our aim is to combine our findings from the three types of evaluation processes that we deployed and make an overall statement about the performance of our frameworks. To that end, we are going to compare the human judgment of our evaluators on the selected visual stories that were given against the results both from the automatic evaluation and the artificial assessment of GPT-4o based on sentences transitions for the visual stories appearing on Figs. 29 & 30.

5.8.1 Combining automatic and human evaluation

In order to compare the results obtained during the automatic and human evaluations, we should first place them under the same basis, meaning that our frameworks should be evaluated on precisely the same visual stories. Consequently, we re-evaluated the automatic metrics score for each of framework (as we had done for Table 20), only for those visual stories that were used during our human evaluation procedure. The summarized results for those 10 visual stories, are demonstrated on Table 24.

Method / Metric	B-1	B-2	B-3	B-4	M	R_L	Cider	Spice	Spider
ClipCap$_7$-*BART*$_{13'}$ ($\mathcal{F}_1$)	21.59	7.43	2.36	0.0	8.84	15.85	4.79	7.84	6.32
ClipCap$_7$-*BART*$_{14'}$ ($\mathcal{F}_2$)	23.36	**10.92**	**5.21**	0.0	**10.56**	**17.88**	17.21	8.92	13.06
ClipCap$_{11}$-*BART*$_{13'}$ ($\mathcal{F}_3$)	20.12	7.41	2.27	0.0	8.72	16.27	12.6	9.87	11.24
ClipCap$_{11}$-*BART*$_{14'}$ ($\mathcal{F}_4$)	**24.18**	9.02	3.42	0.0	9.2	17.02	**21.8**	**10.34**	**16.07**

Table 24. *Evaluation Results from same combinations of ClipCap-BART used on Table 20, but this time only for the portion of visual stories that was used during the human evaluation procedure.*

It's readily distinguishable from Table 24, that the top performing models for these 10 visual stories according to the automatic metrics are frameworks $\mathcal{F}_2$ and $\mathcal{F}_4$ (as they were also proven to be the best, back on Table 20). In addition, it's worth underlining that in this small sample of stories, no model produced a common *4-gram* with the golden stories. At the same time, from human evaluation we conclude that regarding our measured criteria, Figs. 34 showcase that framework $\mathcal{F}_3$ and secondly $\mathcal{F}_4$ are those which attain peak efficiency. Lastly, the part of guessing the human written story, indicates that both frameworks $\mathcal{F}_2$ and $\mathcal{F}_3$ produce the most human-like stories.

Taking into consideration all the above, we could argue that our automatic and human evaluation results are in moderate symphony, since both agree that our top models for story generation are $\mathcal{F}_2$ and $\mathcal{F}_4$ (+ $\mathcal{F}_3$ only from human evaluation), whereas both exclude framework $\mathcal{F}_1$. This consensus can be verified from Table 25, which summarizes the Pearson's (r), Spearman's (ρ) the Kendall's (τ) correlation coefficients, where most of the values appear to be positive. In particular, out of the 144 computed coefficients, 73 (i.e more than 50%) are positive, 52 are negative and lastly 19 are exactly zero. The coefficients were computed column-wise for each of the metrics appearing on Table 24, and the respective columns from the total points that each framework assembled per criterion during the human evaluation (see Table 23).

Furthermore, it's worth to pay attention on the fact that the majority of these positive coefficients originate from the last four automatic metrics namely, ROUGE_L, CIDER, SPICE and SPIDER. This is not an accidental incident, since these metrics are more complicated than BLEU and METEOR and therefore, they capture more underlying meanings in the output stories, something which is of course, seen by the human judgment as well. As a result, these more complex metrics and humans scores are more related.

Finally, taking the comparison to criteria-level, we could remark that by far the criterion for which the automatic metrics and human judges are in agreement the most, is *Coherence and Flow (C)*, followed by *Relevance with the input images (R)*. Especially for the former, we can see that for all metrics the coefficients are positive, while for the latter some negative values only emerge on the last and more intricate metrics. This is another thing that is not surprising, since *Coherence* and *Relevance* are the most easily perceivable and tangible criteria to common logic, compared for example, to *Imagination (I)* or *Engagement (E)*. Hence, the "simplistic" (comparatively to human judge) automatic metrics tend to agree more with these criteria.

	Relevance (R)			Coherence (C)			Narration (N)			Imagination (I)			Engagement (E)			Language (L)		
Metrics	ρ	r	τ	ρ	r	τ	ρ	r	τ	ρ	r	τ	ρ	r	τ	ρ	r	τ
B-1	0.75	0.4	0.33	0.87	1.0	1.0	-0.63	-0.4	-0.33	-0.69	-0.2	0.0	-0.64	-0.2	0.0	-0.59	-0.4	-0.33
B-2	0.42	0.2	0.0	0.45	0.8	0.67	-0.28	-0.2	0.0	-0.4	-0.4	-0.33	-0.39	-0.4	-0.33	-0.17	-0.2	0.0
B-3	0.41	0.2	0.0	0.38	0.8	0.67	-0.29	-0.2	0.0	-0.4	-0.4	-0.33	-0.4	-0.4	-0.33	-0.17	-0.2	0.0
B-4[62]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
M	0.39	0.2	0.0	0.23	0.8	0.67	-0.28	-0.2	0.0	-0.39	-0.4	-0.33	-0.41	-0.4	-0.33	-0.16	-0.2	0.0
R_L	0.26	-0.4	-0.33	0.49	0.6	0.33	-0.1	0.4	0.33	-0.22	0.2	0.0	-0.21	0.2	0.0	0.0	0.4	0.33
Cider	0.06	-0.2	0.0	0.83	0.8	0.67	0.12	0.2	0.0	0.03	0.4	0.33	0.1	0.4	0.33	0.14	0.2	0.0
Spice	-0.43	-0.4	-0.33	0.6	0.4	0.33	0.55	0.4	0.33	0.52	0.8	0.67	0.59	0.8	0.67	0.52	0.4	0.33
Spider	-0.01	-0.2	0.0	0.81	0.8	0.67	0.18	0.2	0.0	0.1	0.4	0.33	0.17	0.4	0.33	0.2	0.2	0.0

Table 25. *Correlation coefficients between automatic metrics and the criteria from human evaluation for the common visual stories.*

5.8.2 Combining human and GPT-4o evaluation

Despite the fact that a head-to-head comparison between human and GPT-4o judgment, on exactly the same topic, is conducted on Appendix C, here we would like to review the overall outcome of the transition-based assessment by GPT-4o on the sentences of stories appearing on Figs. 29 and 30, which were also part of our main human evaluation process. Looking back on paragraph 5.6.2, we attempted to utilize an LLM, in order to grade the flow and continuity on sentences transitions that each of our frameworks display. Of course, this highly resembles to the *Coherence and Flow* criterion[63], which was also assessed during the human evaluation. Therefore, our precise goal, is to check the symphony of those results.

Fig. 33 appoints that only framework $\mathcal{F}_4$ statistically shines outs from the rest of the models, as it attains an average mark of more than 7, in producing more naturally flowing stories. Concurrently, based on the second sub-table from Table 23, framework $\mathcal{F}_4$ stands as first both for most points gathered (197) and for the most top-1 spots taken (23) for *Coherence and Flow*. In fact, this is the only criterion of the six, in which the same model wins both categories in both visual stories on which the criterion was assessed. Thence, the distinct advantage of $\mathcal{F}_4$ against all other frameworks on Fig. 33, is confirmed by human judge as well.

Even if we expand our analysis to the *Narrative Depth* criterion, which can be considered as the second closest form, according to which GPT-4o evaluated the sentences transitions of models, we can validate from Table 23 that framework $\mathcal{F}_4$ has a leading role here as well, since it gets both top spots in visual story 1 and the second place in the overall rankings for the foresaid criterion. Having said all these, it becomes obvious that our GPT-4o and human evaluation procedures are in substantial agreement, since both of them consider $\mathcal{F}_4$ as the best framework, that in general, produces stories which are strongly governed by coherence, temporal continuity and logical flow.

Formally, we can prove our claim, by calculating, once again, the correlation coefficients (Pearson's r, Spearman's ρ, Kendall's τ) between the mean values from GPT-4o ratings and the total number of points that each model collected per criterion. Verily, *Coherence and Flow (C)* is by far the criterion where these two types of evaluation agree the most, whereas *Narrative Depth (N)* comes third. Surprisingly, despite the fact that *Relevance with the input image (R)* criterion, is not directly interrelated with what we asked our LLM to judge upon the generated stories, it raises as the second biggest interface between the two kinds of evaluation.

Type of Weights / Criterion	R	C	N	I	E	L
Pearson's (r)	0.51	**0.91**	0.23	-0.19	0.0	-0.31
Spearman's (ρ)	0.6	**0.7**	0.2	-0.9	0.15	-0.6
Kendall's (τ)	0.4	**0.6**	0.2	-0.9	0.11	-0.4

Table 26. *Correlation coefficients between GPT-4o and human evaluation on the six different criteria*

[62] Since the scores for all frameworks for BLEU-4 were zero, the correlation coefficients between this metric and the total gathered points from human evaluation could not be computed.

[63] and secondly perhaps to *Narrative Depth*.

6 Conclusion & Future Work

Reaching the conclusion of this work, we will discuss the most important inferences drawn from the results presented during Chapter 5. Additionally, we will connect these conclusions with the very beginning of this project and our initial (sub-)research questions. Aiming to maintain correspondence in this link, we divide the current section into three main parts, one for each of the three phases of our experiment (where each phase technically represents one sub-research question). Ultimately, we provide some restrictions that we came up against during the project implementation, but we also give some possible future directions, which we believe would combat these limitations and would make our work even more complete.

6.1 Main Takeaways

As it was already mentioned the main takeaways of this work are given according to the phases of our experimentation, following the structure of our sub-research questions from subsection 1.4. Firstly, we talk about Phase 1 and the eventual accuracy of Clip-Cap in image captioning. Afterwards, our discussion passes to Phase 2 where we provide the central deduction from our attempt to both fine-tune and create from-scratch text-to-text architectures that will be able to reformulate plain captions to meaningful stories. Finally and most importantly, we recapitulate all the major conclusions drawn from the main experiment of this project, where we deployed our end-to-end framework, in the Ultimate Phase, for the fulfillment of visual storytelling.

6.1.1 Deductions from Phase 1

Retracing back to subsections 5.1 and 5.2, we saw three families of Clip-Cap variants: 1) Zero-shot models, 2) Fine-tuned models on MSCOCO and 3) From-scratch models, which were utilized for caption generation on the images of VIST Test set. The second and third family of models were fine-tuned/trained from-scratch on the respective Train part of the dataset. Our results verify that all variants of Clip-Cap can produce moderate to decent captions for images of the VIST dataset. Albeit, the family that was proven to be more accurate specifically for our data, was the Fine-tuned models on MSCOCO dataset.

This is not a surprising result, since the process of fine-tuning language models can be quite beneficial for adapting these architectures to slightly different datasets or even other downstream tasks. In particular, the fine-tuned models with MLP mapper that used beam search as decoding technique ($\mathcal{M}_7$ and $\mathcal{M}_{11}$) performed strongly, notably in the more complicated automatic metrics such as CIDEr, SPICE and SPIDEr. Besides, this efficiency on generating descriptive captions is further established from Fig. 13, where these two models gives us the most accurate and admittedly closer to reality descriptions.

Regarding the Clip-Cap's component-wise comparison, we did not find any statistically significant divergence on the efficacy of the different variants, with an inappreciable exception occurring between the usage of Transformer and MLP mapping networks on Clip-Cap (see Fig. 26). Even in this case, our general conclusion is that *Mokady et al's.* framework is proven to be equally robust on producing accurate image captions, under numerous alterations in its architecture.

6.1.2 Deductions from Phase 2

Phase 2 was probably the unsung corner-stone of this work, since it was the part where we performed actual storytelling by following our main idea of reformulating the provided captions from Phase 1 (or the VIST dataset itself) and also implementing our own from-scratch transformer model to complete the process[64]. The results from subsections 5.3 and 5.4 gifted us with some remarkable conclusions regarding our second sub-research question, which quite logically can be divided to two even smaller pieces: 1) The part of creating our own from-scratch language model with pre-training steps, that would perform text reformulation and eventually purposeful story generation and 2) The part of fine-tuning appropriately existing language models, specifically T5 and BART, in order for them to complete our intricate task of storytelling. In the following lines, we address these two separate parts of Sub-RQ2 one by one.

[64] In addition to fine-tuning some well-known and powerful existing language models.

With regard to the first of these questions, it is evident that our from-scratch T4 models (both the MLM+SP and the base type) cannot compete against large pre-trained models such as T5 and BART, since the latters are clearly superior, at least in the aspect of automatic evaluation and this is confirmed from Tables 14, 15 and 16. Despite this fact, we should not forget that our models are 3 times smaller than T5 and almost 6 times lighter than BART, something that plausibly makes them less versatile in terms of capacity. Moreover, even within our T4 family, the pre-training steps of MLM and SP did not seem to aid majorly, as the base type of T4 surpassed its pre-trained counterpart most of the times. But still, pre-training steps such as MLM and SP (or others), require big external datasets and computational resources that were not available for this project and thus, the application of those techniques was not exploited in the maximum potential.

Though, when looking out of the box (as literally our paragraph 5.4.2 indicates), we cannot disregard the fact that our T4 models outperform T5 and BART on many important metrics such as *the average story length*, *the average sentence length* and even *diversity*. This paradoxical phenomenon, continues even to our self-created *ideality* metric, according to which $4/5$ greatest scores are achieved by T4 variants (see Fig. 28). Taking all these into account and by looking more thoroughly on the example outputs of stories from those models on Fig. 19, we could summarize our inferences regarding T4 models, as follows:

> Despite the extensive difference between pre-trained (T5 & BART) and from-scratch (T4 variants) models in the automatic metrics, the latters do not fall short on the quality and diversity of language that they generate, nor in the general narrative traits of the produced text. Instead, it seems that they lose their correspondence with the input sequence, resulting to a very low relevance with it, something that could justify the eventual under-performance in the automatic metrics. This low relevance with regard to the input, could possibly be attributed to the lack of pre-training methods, the absence of large scale linguitic data, the confinements in terms of size (trainable parameters) and other technical limitations of our set-up and our approach.

Coming to the second of the previously postulated questions concerning Sub-RQ2, pre-trained language models like T5 and BART, can unmistakably be fine-tuned appropriately and become capable of generating coherent and relevant to their input, stories. As we mentioned, our results showcase that these type of architectures due to their involute structure, their size and their vast pre-training can be adapted to text reformulation quite readily and efficiently. They demonstrate equivalent quality of language, but unlike to our from-scratch T4 family, they also keep high levels of attention to the initial input sequence. The confirmation for this comes, can be traced back, to the generated story examples of Fig. 19 (especially for BART), and of course, by their large margin win against T4 models, particularly in the more complicated automatic metrics.

6.1.3 Conclusions from the Ultimate Phase

Eventually, the Ultimate Phase constituted the principal product of the present work and its results answer to Sub-RQ3, but also to our initial research question. By combining the best two performing systems from the preceding phases (image captioning & storytelling from isolated descriptions) and aiming to generate knowingly stories, this part incarnates all the essence of our dissertation. Due to the fact that the evaluation was multifaceted (automatic, AI and human evaluation), we will attempt to split our takeaways accordingly and later to merge them, as we did with our findings in subsection 5.8.

Recalling our results from paragraphs 5.5.1 and 5.5.3, firstly we should comment that the top four reformulators comprise only of BART models. Furthermore, from the automatic evaluation, it's unambiguous that frameworks that use $\text{BART}_{14'}$ as a storyteller ($\mathcal{F}_2$ & $\mathcal{F}_4$), accomplish higher scores in automatic metrics, particularly in those which introduce more complexity. Nonetheless, likewise in the case of Phase 2[65], these frameworks were not those designated with the top scores regarding lexical characteristics. On the contrary, systems with the storyteller $\text{BART}_{13'}$ ($\mathcal{F}_1$ & $\mathcal{F}_3$), showcased peak performance having larger *story* and *sentence sizes*, richer *vocabulary* and therefore, greater *diversity* and *ideality*. This, can be reasonably attributed to the fact that $\text{BART}_{13'}$ is a model with *nucleus sampling* as a generation strategy, whilst $\text{BART}_{14'}$ utilizes *beam search* during generation. As a result the former introduces more randomness while generating and has a bigger range of words, whereas the latter, operates more greedily and opts only from a restricted pool of tokens, something that still brings higher precision in automatic evaluation.

[65] where the models that topped the automatic evaluation, did not get the highest spots on our analysis about linguistic traits.

Our exposition in subsections 5.6 and 5.7 enhanced even more our findings drawn from automatic evaluation. The elaborating breakdown in paragraphs 5.6.1 and 5.6.2 indicated that our frameworks have powerful sense of relevance to their input and can generate creative and grammatically structured individual sentences. Contemporaneously, the semantic links between sentences are not as strong, resulting to a lower level of total coherence. After deploying GPT-4o for assessing the logical flow of our stories, we saw that only model $\mathcal{F}_4$, was clearly above the human scores. Supplementarily, this claim is further reinforced on Appendix C were for the exact same purpose, human judgment was capitalized.

Coming to the main human evaluation process and its first part[66], we can notice similar results to those from automatic evaluation. As we pointed out on our discussion of sub-subsection 5.8.1, the two kinds of results lie on symphony, as both establish framework $\mathcal{F}_4$ (and secondly $\mathcal{F}_3$), as the most complete option for creating high end stories. The collective performance of our models against the human written stories (Figs. 44a & 44b), gifts us with a valuable deduction. All of our frameworks perform remarkably well on criteria which are more specific and "measurable" such as *Relevance* or *Coherence* and not so abstract like for example *Imagination* or *Engagement*. This is to be expected, because for instance, we can far more easily train a model to follow the course of the input images than to engage each individual human annotator.

As a last point on this, our latter observation can be related (and actually also be empowered) with/by the agreement of our evaluators as well. By carefully looking on Figs. 38 to 43, we could allege that the average agreement on the first three criteria (*R, C, N*) is lower than the last three (*I, E, L*), and since there are four different machine generated stories and only one golden story, it would make more sense that in the criteria where the agreement is high, the performance of that lone golden story is better.

The second half of our human evaluation[67], confirmed further our assertions that our frameworks pose a serious challenge to our human assessors, for understanding if the provided stories are machine generated or not. In spite of the fact that human stories indeed obtain the largest guessing percentage on average, this is a weak majority and leaves back a lot of uncertainty, when we attempt to distinguish human written stories from those generated by our frameworks.

To recapitulate, human evaluation was of critical importance for deeply comprehending the outcomes of our project. This procedure ascertained that indeed, our main framework is competent of creating cohesive and narrative stories, at least to a comparable level to the human ones from the dataset. Both the ranking based on criteria and the part of guessing the human written story, showed that our annotators considered all the stories of equal overall quality and could not easily distinguish which was human/machine produced. At the same time, quite similar conclusions, regarding the models which performed better/worse, are drawn both from the automatic and artificial (with GPT-4o) assessments.

Therefore, as an ultimate consequence, we firmly believe that architectures such as the one depicted on Fig. 14, can be utilized fruitfully for visual storytelling, especially the frameworks $\mathcal{F}_2$, $\mathcal{F}_3$ and $\mathcal{F}_4$. However, we should highlight here, that VIST is an old dataset and the overall structure and quality of its language could be far more superior, than the one that it actually has. To that end, we would not like to declare the story generation level of our models equivalent to the general human skill of narrating.

6.2 Limitations & Future Work

As most of the academic researches, ours, had its own limitations and obstacles while unfolding. Synchronously, the extremely interesting and multifarious topic of Visual Storytelling, leaves plenty of leeway to expand, deepen and improve our research. Having said these, in this subsection, we reveal some limitations of our study and we provide possible future directions for dilatation.

6.2.1 Limitations

When describing the results from Phase 2 and the Ultimate Phase we underlined some limitations of our approaches. One of them was the under-performance of our from-scratch made model T4, compared to the pre-trained models T5 and BART. As we already mentioned, this can be attributed to several reasons,

[66] where the generated stories were ranked according to criteria.

[67] where our evaluators had to guess the human written stories.

from which the most major would be: the pretraining procedures, the different levels of exposure to external linguistic corpuses, the model size and lastly, some possible low-level intricate handlings within the architectures of T5 and BART. What is more, the selection of MLM and SP as methods of extra tuning, did not had a positive effect on the efficacy of the model, possibly because they are not enough to act on their own, meaning that for example on T5 and BART their were utilized in conjunction with other techniques like *document rotation* or *text infilling*. We should not omit that the absence of alternative sources of data, hindered our attempt to apply MLM and SP outside of the context of just storytelling.

Framework $\mathcal{F}_1$ was the one which was composed by the best captioner (ClipCap$_7$) and the best reformulator (BART$_{13'}$), but paradoxically, turned up to be the least efficient during the Ultimate phase, unanimously by all ways of evaluation. As we have underscored before, the problem partially lies on the *p-sampling*-based generator BART$_{13'}$, but even compared to $\mathcal{F}_3$ (the other framework that uses BART$_{13'}$), we can see an inferiority from the former, particularly when taking into account the human evaluation. This lead us to conclude, that the specific couple of captioner-reformulator does not align as expected, perhaps because this BART variant combines better with a transformer-based captioner than with an MLP-based captioner.

Another limitation was identified in the low in-between sentences coherence. This is a problem that springs from our data, where the linguistic quality is low, something that reflects to the moderate ranks of the human stories themselves, during evaluation. In turn, our models are under-trained and for example, they lack the capacity of using linking words during generation. As a result, despite that within our dataset the outcome is satisfactory, we cannot declare that our framework, in its current phase, can achieve state-of-the-art results in *generalized Visual Storytelling*, compared for instance, to an LLM. Naturally, but to match such a network we would need far greater computational resources, amount of available data and even labour force.

6.2.2 Future Directions

Most of the aforementioned limitations point to possible research paths that our present work could be expanded. Firstly, the efficiency of our T4 model can be improved after taking into consideration several directions. Obviously, increasing the amount of similarly structured data offered to the model, and thus the training, could help, but in our opinion a very promising attempt, would be to apply methods of document corruption outside of the sphere of visual storytelling. In the case of abundant external datasets (not necessarily of the same kind), this encloses MLM and other such techniques only for predicting missing tokens or span of tokens, so to empower the contextual understanding abilities of the model. All the same, unlike our occasion, this time we should keep in isolation the encoder and decoder respectively and they would be treated and pre-trained as different structures separately.

On top of that, since our system seems to lose the correspondence with its input captions, alternative objective functions should be considered. Among them, the usage of contrastive loss could force the alignment of the generated stories with the inputs, while penalizing outputs that are irrelevant or fine-tuning the model with reinforcement learning where a reward is given based on coherence with the input captions and not only the similarity with the original story.

Continuing on the total proposed pipeline, our main suggestion for improvement is quite evident. The framework of Fig. 14 is not trained end-to-end and only its individual parts are fine-tuned. A plausible expansion would be to unify Clip-Cap and our storytellers under one single network which will be fine-tuned directly with visual content (i.e the images of VIST), giving it the chance to be adapted to visual storytelling during training time. Of course, this would come to the cost of resources and time since combining such models could bring up the number of parameters to even half a billion. Notwithstanding, given the necessary hardware, we deem that following this practice would tremendously improve the final result, since both of the already fine-tuned language models would now focus, all together only, on how to adjust their weights for the common goal of story generation.

Our whole project is based on the consensus that image captioning can be useful for visual storytelling, when applied in a serial manner as in the following schema: *images* $\rightarrow$ *captions* $\rightarrow$ *story* (likewise Fig. 14). An intriguing idea is to convert this serialization to a more “parallel format”, in the sense that we exploit concurrently the visual & textual information from the series of images and the captions respectively, and we pass them to a modified model for producing the final story. The high-level schema now would look like:

images + *captions* → *story*, and our network could be reduced down to something close to Clip-Cap but with two diversifications: 1) The capability of accepting a series of images in parallel, encode them and pass their representations to a language model for story generation along with the encoded, by a textual encoder, representations from the captions and 2) The new model, obviously, should be trained on storytelling and not on generating mere descriptions. We believe that with the right adjustments, this novel kind of V&L model, as generalized version of Clip-Cap, could capture more deeply the context of the correlated images and create more diverse stories, having in its weaponry the accompanying information from the captions at once.

As the last note of this journey, we could see this work to become a real-world application, especially after implementing some of the previously mentioned expansions. Going one step further the idea of our "generalized Clip-Cap", we could replace the series of correlated images, with frames from videos and the sentences from stories, with subtitles. In such a scenario, that system could be utilized for creating narrative subtitles for movies and documentaries or for turning photo albums into expository travel stories for the purpose of automated travel journals. Furthermore, with the uprising of social media and internet platforms this type of systems, could be availed for creating content, by automatically generating narratives for photo series or even videos. The list could go far longer here, but our main point is that Vision & Language universe, it's undeniably a limitless area to explore, with numerous daily life applications which for sure is going to keep busy the scientific community in the upcoming years.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Armen Aghajanyan, Bernie Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, et al. Cm3: A causal masked multimodal model of the internet. *arXiv preprint arXiv:2201.07520*, 2022.

[3] Harsh Agrawal, Arjun Chandrasekaran, Dhruv Batra, Devi Parikh, and Mohit Bansal. Sort story: Sorting jumbled images and captions into stories. *arXiv preprint arXiv:1606.07493*, 2016.

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

[5] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pages 382–398. Springer, 2016.

[6] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[7] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

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

[10] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[12] Hong Chen, Yifei Huang, Hiroya Takamura, and Hideki Nakayama. Commonsense knowledge aware concept selection for diverse and informative visual storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 999–1008, 2021.

[13] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[14] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

[15] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[16] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. *open review*, 2019.

[17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[18] Dami Choi, Christopher J. Shallue, Zachary Nado, Jaehoon Lee, Chris J. Maddison, and George E. Dahl. On empirical comparisons of optimizers for deep learning, 2020.

[19] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[20] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR, 2017.

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

[22] Jacob Devlin, Saurabh Gupta, Ross Girshick, Margaret Mitchell, and C Lawrence Zitnick. Exploring nearest neighbor approaches for image captioning. *arXiv preprint arXiv:1505.04467*, 2015.

[23] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

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

[25] Desmond Elliott and Frank Keller. Image description using visual dependency representations. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1292–1302, 2013.

[26] Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*, 2021.

[27] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 15–29. Springer, 2010.

[28] William Feller. *An introduction to probability theory and its applications, Volume 2*, volume 81. John Wiley & Sons, 1991.

[29] Mauajama Firdaus, Arunav Pratap Shandeelya, and Asif Ekbal. More to diverse: Generating diversified responses in a task oriented multimodal dialog system. *PloS one*, 15(11):e0241271, 2020.

[30] Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. *arXiv preprint arXiv:1702.01806*, 2017.

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

[32] Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6639–6648, 2019.

[33] Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. Story plot generation based on cbr. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 33–46. Springer, 2004.

[34] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. *Advances in neural information processing systems*, 27, 2014.

[35] Diana Gonzalez-Rico and Gibran Fuentes-Pineda. Contextualize, show and tell: A neural visual storyteller. *arXiv preprint arXiv:1806.00738*, 2018.

[36] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[37] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[38] Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

[39] Alex Graves. Generating sequences with recurrent neural networks, 2014.

[40] Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108, 2020.

[41] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[43] Sen He, Wentong Liao, Hamed R Tavakoli, Michael Yang, Bodo Rosenhahn, and Nicolas Pugeault. Image captioning through image transformer. In *Proceedings of the Asian conference on computer vision*, 2020.

[44] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words. *Advances in neural information processing systems*, 32, 2019.

[45] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

[47] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.

[48] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[49] Chao-Chun Hsu, Zi-Yuan Chen, Chi-Yang Hsu, Chih-Chia Li, Tzu-Yuan Lin, Ting-Hao Huang, and Lun-Wei Ku. Knowledge-enriched visual storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7952–7960, 2020.

[50] Chi-Yang Hsu, Yun-Wei Chu, Ting-Hao'Kenneth' Huang, and Lun-Wei Ku. Plot and rework: Modeling storylines for visual storytelling. *arXiv preprint arXiv:2105.06950*, 2021.

[51] Ting-Yao Hsu, Chieh-Yang Huang, Yen-Chia Hsu, and Ting-Hao'Kenneth' Huang. Visual story post-editing. *arXiv preprint arXiv:1906.01764*, 2019.

[52] Junjie Hu, Yu Cheng, Zhe Gan, Jingjing Liu, Jianfeng Gao, and Graham Neubig. What makes a good story? designing composite rewards for visual storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7969–7976, 2020.

[53] Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. Attention on attention for image captioning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4634–4643, 2019.

[54] Ting-Hao K. Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Aishwarya Agrawal, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. Visual storytelling. In *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*, 2016.

[55] Vladimir Iashin and Esa Rahtu. Multi-modal dense video captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 958–959, 2020.

[56] Ilija Ilievski, Shuicheng Yan, and Jiashi Feng. A focused dynamic attention model for visual question answering. *arXiv preprint arXiv:1604.01485*, 2016.

[57] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[58] Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. Story generation from sequence of independent short descriptions. *arXiv preprint arXiv:1707.05501*, 2017.

[59] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*, 2019.

[60] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

[61] Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10267–10276, 2020.

[62] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4565–4574, 2016.

[63] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[64] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[65] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[66] Taehyeong Kim, Min-Oh Heo, Seonil Son, Kyoung-Wha Park, and Byoung-Tak Zhang. Glac net: Glocal attention cascading networks for multi-image cued story generation. *arXiv preprint arXiv:1805.10973*, 2018.

[67] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[68] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[69] Benjamin Klein, Guy Lev, Gil Sadeh, and Lior Wolf. Fisher vectors derived from hybrid gaussian-laplacian mixture models for image annotation. *arXiv preprint arXiv:1411.7399*, 2014.

[70] Jacob Kohen. A coefficient of agreement for nominal scale. *Educ Psychol Meas*, 20:37–46, 1960.

[71] Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. A hierarchical approach for generating descriptive image paragraphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 317–325, 2017.

[72] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017.

[73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[74] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[75] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[76] Guang Li, Linchao Zhu, Ping Liu, and Yi Yang. Entangled transformer for image captioning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8928–8937, 2019.

[77] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[78] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[79] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[80] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16*, pages 121–137. Springer, 2020.

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

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

[83] Danyang Liu and Frank Keller. Detecting and grounding important characters in visual stories. *arXiv preprint arXiv:2303.17647*, 2023.

[84] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023.

[85] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023.

[86] Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Optimization of image description metrics using policy gradient methods. *CoRR*, abs/1612.00370, 2016.

[87] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[88] Yu Liu, Jianlong Fu, Tao Mei, and Chang Wen Chen. Let your photos talk: Generating narrative paragraph for photo stream via bidirectional attention recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[89] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[90] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[91] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 375–383, 2017.

[92] Ruotian Luo. A better variant of self-critical sequence training. *arXiv preprint arXiv:2003.09971*, 2020.

[93] Ziyang Luo, Yadong Xi, Rongsheng Zhang, and Jing Ma. A frustratingly simple approach for end-to-end image captioning. *arXiv preprint arXiv:2201.12723*, 2022.

[94] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. *Advances in neural information processing systems*, 27, 2014.

[95] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[96] Ron Mokady, Amir Hertz, and Amit H. Bermano. Clipcap: Clip prefix for image captioning, 2021. 2.

[97] Md Sultan Al Nahian, Tasmia Tasrin, Sagar Gandhi, Ryan Gaines, and Brent Harrison. A hierarchical approach for visual storytelling using image description. In *Interactive Storytelling: 12th International Conference on Interactive Digital Storytelling, ICIDS 2019, Little Cottonwood Canyon, UT, USA, November 19–22, 2019, Proceedings 12*, pages 304–317. Springer, 2019.

[98] Jean Nyandwi. The transformer blueprint, 2023. Accessed: March 1, 2024.

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

[100] Cesc C Park and Gunhee Kim. Expressing an image stream with a sequence of natural sentences. *Advances in neural information processing systems*, 28, 2015.

[101] Karl Pearson. Vii. note on regression and inheritance in the case of two parents. *proceedings of the royal society of London*, 58(347-352):240–242, 1895.

[102] Jesús Andrés Portillo-Quintero, José Carlos Ortiz-Bayliss, and Hugo Terashima-Marín. A straightforward framework for video retrieval using clip. In *Mexican Conference on Pattern Recognition*, pages 3–12. Springer, 2021.

[103] Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sacheti. Imagebert: Cross-modal pre-training with large-scale weak-supervised image-text data. *arXiv preprint arXiv:2001.07966*, 2020.

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

[105] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *mike captain*, 2018.

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

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

[108] Rita Ramos, Bruno Martins, Desmond Elliott, and Yova Kementchedjhieva. Smallcap: lightweight image captioning prompted with retrieval augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2840–2849, 2023.

[109] Hareesh Ravi, Kushal Kafle, Scott Cohen, Jonathan Brandt, and Mubbasir Kapadia. Aesop: Abstract encoding of stories, objects, and pictures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2052–2063, 2021.

[110] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[111] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7008–7024, 2017.

[112] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. *arXiv preprint arXiv:1809.02156*, 2018.

[113] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[114] Robin M Schmidt, Frank Schneider, and Philipp Hennig. Descending through a crowded valley-benchmarking deep learning optimizers. In *International Conference on Machine Learning*, pages 9367–9376. PMLR, 2021.

[115] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[116] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can clip benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021.

[117] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[118] Charles Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 1961.

[119] Jing Su, Qingyun Dai, Frank Guerin, and Mian Zhou. Bert-hlstms: Bert and hierarchical lstms for visual storytelling. *Computer Speech & Language*, 67:101169, 2021.

[120] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7464–7473, 2019.

[121] Aditya K Surikuchi, Raquel Fernández, and Sandro Pezzelle. Not (yet) the whole story: Evaluating visual storytelling requires more than measuring coherence, grounding, and repetition. *arXiv preprint arXiv:2407.04559*, 2024.

[122] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[123] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[124] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

[125] Guy Tevet and Jonathan Berant. Evaluating the evaluation of diversity in natural language generation, 2021.

[126] Yoad Tewel, Yoav Shalev, Idan Schwartz, and Lior Wolf. Zerocap: Zero-shot image-to-text generation for visual-semantic arithmetic. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17918–17928, 2022.

[127] Mariët Theune, Sander Faas, Anton Nijholt, and Dirk Heylen. The virtual storyteller: Story creation by intelligent agents. In *Proceedings of the Technologies for Interactive Digital Storytelling and Entertainment (TIDSE) Conference*, volume 204215, page 116, 2003.

[128] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

[129] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

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

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

[132] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE international conference on computer vision*, pages 4534–4542, 2015.

[133] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[134] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

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

[136] Bairui Wang, Lin Ma, Wei Zhang, and Wei Liu. Reconstruction network for video captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7622–7631, 2018.

[137] Eileen Wang, Caren Han, and Josiah Poon. Rovist: Learning robust metrics for visual storytelling. *arXiv preprint arXiv:2205.03774*, 2022.

[138] Xin Wang, Wenhu Chen, Yuan-Fang Wang, and William Yang Wang. No metrics are perfect: Adversarial reward learning for visual storytelling. *arXiv preprint arXiv:1804.09160*, 2018.

[139] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*, 2021.

[140] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

[141] Qi Wu, Peng Wang, Chunhua Shen, Ian Reid, and Anton van den Hengel. Are you talking to me? reasoned visual dialog generation through adversarial learning, 2017.

[142] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[143] Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. Megatron-cntrl: Controllable story generation with external knowledge using large-scale language models. *arXiv preprint arXiv:2010.00840*, 2020.

[144] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10685–10694, 2019.

[145] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29, 2016.

[146] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015.

[147] Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385, 2019.

[148] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 684–699, 2018.

[149] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659, 2016.

[150] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

[151] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

[152] Licheng Yu, Mohit Bansal, and Tamara L Berg. Hierarchically-attentive rnn for album summarization and storytelling. *arXiv preprint arXiv:1708.02977*, 2017.

[153] Youngjae Yu, Jiwan Chung, Heeseung Yun, Jongseok Kim, and Gunhee Kim. Transitional adaptation of pretrained models for visual storytelling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12658–12668, 2021.

[154] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5579–5588, 2021.

[155] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *arXiv preprint arXiv:1703.10960*, 2017.

[156] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI conference on artificial intelligence*, pages 13041–13049, 2020.

[157] Linchao Zhu and Yi Yang. Actbert: Learning global-local video-text representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8746–8755, 2020.

A Appendix A: Loss Functions & Optimizers

During both our Literature Review (Chapter 2) and our Experimental set-up (Chapter 4), we mentioned several kind of loss functions and optimization methods without really explaining in depth their functionality and how they can assist in deep learning applications, especially tasks such as narrative text generation.

A.1 Loss Functions

For V&L tasks such as Image Captioning and Visual Storytelling there is a variety of *loss functions* that can be applied depending each time on the nature of the application and the model. Some well-known loss functions that are being used for these tasks include:

- *Cross-Entropy Loss (CE)*
- *Reinforcement Learning-based Loss*
- *Contrastive Loss*
- *Negative Log-Likelihood (NLL) Loss*

A.1.1 Cross-Entropy Loss

Cross-Entropy loss (CE) was firstly introduced by Claude Shannon in 1948 [115], in his book about information theory. It is a function that is widely used both in classification problems and sequence generation tasks, including image captioning and visual storytelling. In particular, in text generation tasks, it is used to compare the predicted sequence, which can be a series of tokens, with the ground truth sequence. To understand even further, for each unit in the sequence (i.e token), the model outputs a probability distribution over the vocabulary (V) at each time step. Cross-entropy loss measures the difference between the predicted distribution and the encoded form of the ground truth token. It penalizes incorrect predictions and encourages the model to maximize the probability of the correct token.

More formally, if we denote with y_t the ground truth token at time step t, with $y_{1:t-1}$ the preceding ground truth sequence up to step $t-1$ and with $\mathbf{X}$ any other kind of additional input[68], then we can define the expression $p(y_t|y_{1:t-1}, \mathbf{X})$, which is the model's prediction of the probability distribution over the next token y_t, given the ground truth sequence up to the previous time step ($y_{1:t-1}$) and any other input context. From this point, the *Cross-Entropy* loss, $\mathcal{L}_{\text{CE}}$, is simply given as the negative log of this expression (we call this *likelihood*) over all times steps t, according to the formula below:

$$\mathcal{L}_{\text{CE}} = -\sum_{t=2}^{T} \log p(y_t|y_{1:t-1}, \mathbf{X}) \tag{36}$$

In equation (36), T is the length of the target sequence. Cross-Entropy is perhaps, the most common loss function used in sequence generation tasks and since visual storytelling involves sequence generation as well, it was also universally used (i.e. in all our models and phases) in this project[69].

A.1.2 Reinforcement Learning-based Loss

As we have already been acquainted from paragraph 2.3.2, many V&L tasks including visual storytelling use Reinforcement Learning (RL) approaches in order to construct a solid framework for text generation [52, 138]. In such occasions, RL-based loss functions are utilized to address the limitations of cross-entropy loss, particularly when models are evaluated using non-differentiable metrics like BLEU, ROUGE_L, CIDEr, or SPICE. RL-based methods aim to directly optimize these evaluation metrics using

[68] for example, we can have images, other text etc. Of course, $\mathbf{X}$ here can also be omitted.

[69] Keep in mind that in phase 1 the internal contrastive loss of CLIP for associating pairs of captions and images is different than the final loss, that we use during fine-tuning or training which is calculated with the cross-entropy function.

different methodologies such as the REINFORCE algorithm [140] and more precisely an application of it, the *Self-Critical Sequence Training* (SCST) [111].

Roughly, SCST is a technique that fine-tunes a model by directly optimizing the automatic evaluation metrics using RL. The model's own greedy predictions operate as a baseline for the rewards. At every generation step, SCST refines the produced sequence by rewarding the model, based on how much better its sampled output is compared to the baseline. The key idea here, is to optimize for evaluation metrics (e.g., CIDEr, BLEU) by treating the generation process as a reinforcement learning problem.

In more detail, initially the model generates two kind of sequences, a sampled and a greedy sequence. The former corresponds to a sequence stochastically sampled from its own final probability distribution over V, whereas the latter is a sequence generated using greedy decoding. The reward is a score computed using a task-specific metric and it measures how well the sampled and the greedy sequence match the ground truth. In each iteration (generation step), the model's reward for the sampled sequence is compared to the reward of the greedy sequence which in the beginning serves as the baseline. This can be formulated as follows:

$$\mathcal{L}_{SCST} = -\left(r(\hat{y}) - r(\hat{y}_{\text{greedy}})\right) \sum_{t=2}^{T} \log p(\hat{y}_t|\hat{y}_{1:t-1}) \tag{37}$$

On equation (37) we symbolize with $\hat{y}$ the stochastically sampled sequence, with $\hat{y}_{\text{greedy}}$ the greedy sequence, while $r(\hat{y})$ is the reward per evaluation metric for the stochastically sampled sequence and $r(\hat{y}_{\text{greedy}})$ is the reward for the greedy sequence which is set as baseline. Ultimately, the model is updated by maximizing the difference between the rewards of the sampled and the greedy sequence respectively.

Among its benefits, SCST optimizes non-differentiable metrics which otherwise remain intact through cross-entropy, alleviates the contained bias in the generated sequence by combining both the sampled and greedy sequence in the loss and improves long-term dependencies. Hence, a plethora of works have exploited the SCST approach (or in general RL based approaches) for better tuning their architectures on V&L tasks such as generation of visual dialogues [92, 141].

A.1.3 Contrastive Loss

A quite enterprising way of computing the loss in generative tasks is the so called *Contrastive Loss*[70]. This type of loss was introduced by *Hadsell et al.* in [41] and the basic idea is that it promotes the model to learn similarities between input pairs, for instance, visual and text inputs. For this reason, it is employed in scenarios where models learn joint embeddings for both visual and textual information. Contrastive loss attempts to bring semantically related image-sentence pairs closer by mapping them in a new embedding space, while pushing unrelated pairs farther apart (in that same space). The close in similarity pairs are dubbed as *positive* samples while the dissimilar ones as *negative* pairs.

While being fine-tuned the model learns by itself to minimize the distance between the correct (positive) representations of the two kind of inputs, while maximizing the distance between incorrect (negative) pairs. Hence, *Contrastive Learning* is considered to be a self-supervised learning method. In the general form, the contrastive loss, $\mathcal{L}_{\text{contrastive}}$, can be expressed as follows:

$$\mathcal{L}_{\text{contrastive}} = \sum_{(x,x^+),(x,x^-)} \left[\lambda \cdot d(f(x), f(x^+)) + (1-\lambda) \cdot \max(0, \alpha - d(f(x), f(x^-)))\right], \tag{38}$$

where λ is a binary number (0 or 1), that indicates if we have positive or negative pairs, x is the fixed input (e.g., an image), x^+ is the positive sample (e.g., a matching caption), x^- is the negative sample (e.g., a non-matching caption), $d(f(x), f(x^+))$ and $d(f(x), f(x^-))$ are the distances between the representations[71] of x & x^+ and x & x^- respectively. Finally, α is a margin ensuring that the negative pairs is sufficiently far away from the positive ones. It becomes vivid, that contrastive loss comes in handy when we try to match or align image-text pairs like in the cases of image captioning tasks [104]. Moreover, as a self-supervised learning technique it can be useful when our models have to learn representations without labeled data by using as positive samples different parts of the augmented initial data [14].

[70] The technique that uses contrastive loss is called *Contrastive Learning*.

[71] potentially after mapping the inputs into a common embedding space.

A.1.4 Negative Log-Likelihood Loss

Negative Log-Likelihood (NLL) loss is one of the most classical loss function used in all of machine learning (ML) models. It roots back to *Maximum Likelihood Estimation (MLE)* [28] and it is widely known for many classification applications but also for a variety of sequence prediction problems [10]. For this kind of tasks, such as visual storytelling, NLL loss measures how well the predicted probability distribution matches the target sequence and essentially it becomes identical to CE[72]. As a result, we can formalize the expression for NLL loss as below:

$$\mathcal{L}_{\text{NLL}} = -\sum_{t=2}^{T} \log p(y_t | y_{1:t-1}, \mathbf{X}) \tag{39}$$

More elaborately, NLL loss is typically used in conjunction with a softmax layer (existing in an assumed language model) that outputs a probability distribution over the vocabulary V. The resulted loss measures how well the model's predicted probabilities for the next token match the actual words in the ground truth story. More precisely and as equation (39) indicates, at each time step t, the loss is computed for the predicted word given the previously generated words and any other contextual input $\mathbf{X}$.

Because of it's "step-by-step" nature, NLL loss ensures that on each time step the model generates the correct token, thus leading to a higher accuracy in word-level predictions. Furthermore, it allows an easy integration into a wide variety of model architectures such RNNs, LSTMs, GRUs or Transformers, and therefore shows great versatility and flexibility.

A.2 Optimizers

Optimization algorithms are now an integrated part of deep learning and are the best way to improve the weights of a neural network by minimizing some error function or by maximizing some objective function [18,114]. They are divided into 2 major categories: a) gradient descent algorithms and b) adaptive algorithms. Despite the fact that gradient descent algorithms are vital in many ML applications, adaptive optimizers have proven more effective, particularly in tasks involving sequence generation, like visual storytelling. Hence the are deployed more excessively in complex architectures such as transformer-based language models. For this reason, this is the family that we will elaborate more in the present subsection. In particular, we discuss in more detail the following algorithmos of optimization:

- *Root Mean Square Propagation - RMSprop*
- *Adaptive Moment Estimation - Adam*
- *Adaptive Moment Estimation with Weight Decay - AdamW*

A.2.1 Root Mean Square Propagation (RMSprop)

The *Root Mean Square Propagation* or simply *RMSprop* [45] is an optimizer that uses adaptive learning rate, popular for it's ability to handle non-stationary loss functions, which can often arise in sequence generation tasks, such as image captioning or visual storytelling. It was developed in order to address the basic problem of the *Adagrad* algorithm, concerning the annihilation of the learning rate during optimization. The core idea behind RMSprop is to maintain a moving average of the square of the gradients and normalize the current gradient by this moving average.

On operation level, RMSprop at each time step t, computes the gradient of the loss function $L(\theta)$ with respect to the model parameters θ. Mathematically this is expressed as:

$$g_t = \nabla_\theta L_t(\theta) \tag{40}$$

[72] The two differ mostly in contexts like classification, where Cross-Entropy loss explicitly accounts for the full distribution over classes while NLL is just used when we already have the probability distribution.

After this, RMSprop keeps an exponentially decaying average of the squared gradients (those calculated in equation (40)) as:

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho)g_t^2, \tag{41}$$

where ρ is a decay term that determines the refresh rate of gradient and $E[g^2]_t$ is the moving average of the squared gradient at time step t. In the end, the algorithm updates the parameters θ of the model as below:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \tag{42}$$

Here, ϵ is a smoothing parameter to avoid division by zero and η is the learning rate. According to the proposer of the algorithm, Geoffrey Hinton, a good choice of the parameter ρ is 0.9, while a good initialization of the learning rate η is 0.001.

Generally, RMSprop brings numerous advantages during training since it gives an adaptive learning rate, prevent large oscillations of the loss and also possesses memory in the sense that it focuses on recent gradients that is probable to be more important[73]. However, most notably in sequence generation, the foresaid algorithm helps on the stabilization of training, where the loss can be volatile due to dependencies between tokens [39]. Additionally, in some sequence generation tasks, gradients can be sparse or change abruptly. RMSprop is efficient at handling such scenarios by adjusting the step size accordingly.

A.2.2 Adaptive Moment Estimation (Adam)

On several experiments we used the *Adaptive Moment Estimation (Adam)* as an optimizer, which is another adaptive algorithm proposed in [67]. Beyond storing the exponentially decreasing mean of past squares of the gradients (as RMSprop does), it also stores the exponentially decreasing average of the previous slopes m_t (referring to the instantaneous slope g_t). Given the gradients g_t at any time step t, mathematically, we can express this as follows:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t \tag{43a}$$

$$u_t = \beta_2 u_{t-1} + (1-\beta_2)g_t^2 \tag{43b}$$

In this case, m_t and u_t are called bias-corrected first (mean) and second (uncentered deviation) momentum of the gradients respectively, while β_1 and β_2 are exponential decay rates for the moving averages. Since m_t, u_t are initialized as vectors with zeros, they are observed to be biased around the region of zero and especially during the early stages of training. To deal with this difficulty, *Kingma and Ba* propose to re-evaluate the first and second momentums at time step t, as follows:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{44a}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{44b}$$

We then use the new above momentum estimates for updating the parameters θ of the model, in a similar way to RMSprop:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{45}$$

where η is the learning rate and ϵ is a small constant for avoiding division by zero. According to the bibliography some good values for the parameters β_1 and β_2 are 0.9 and 0.999 respectively whilst for ϵ is 10^{-8}. That is, the values ofβ_1, β_2 should be close to 1 which slows down the rate of exponential decay of the slopes and as a result the variation of course oscillations towards the minimum.

Besides reducing the amount of oscillations, Adam also shows adaptability with the automatically adjustable learning rate but also gives reliable estimates during the early stages of training due to the bias-correction at each time step. For tasks like language modeling, text generation and other sequence-to-sequence tasks, Adam allows faster convergence than the traditional gradient descent methods. In the cases

[73] This happens because of the influence of ρ on equation (41).

of these tasks where the sequences are long, Adam reduces the learning rate where needed (through v_t) and increases it when the gradients are small, something that improves overall optimization. For these reasons, the foresaid algorithm shows robustness and thus it is commonly used in transformers-based and other deep generative models [106, 130, 133].

A.2.3 Adaptive Moment Estimation with Weight Decay (AdamW)

Adaptive Moment Estimation with Weight Decay or just *(AdamW)* is an optimization algorithm that introduces a small modification to the Adam optimizer by applying weight decay separately, which leads to better regularization [89]. It was the optimizer that was used in the majority of our experiments. In practice, AdamW keeps the same equations for calculating the first moment (mean) and second moment (uncentered deviation), as well as the bias correction for the moments as in the case of simple Adam (equations (43) & (44)). The differentiation comes on the way of updating the parameters, where a decoupled weight decay factor λ is added as well:

$$\theta_t = \theta_{t-1} - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_{t-1} \right), \tag{46}$$

The term $\lambda\theta_{t-1}$ applies the weight decay directly to the model parameter updates. In general, this decoupling often has the effect that AdamW tends to generalize better than Adam. This is significant for sequence generation models, because we need them to generalize well to unseen inputs. Simultaneously, AdamW retains the fast convergence of Adam and also demonstrates greater stability in tasks with long sequences, such as image captioning or visual storytelling, since the decoupled weight decay ensures more robustness during training. Having these merits, this algorithm has become a mainstay in numerous well-known transformer models which are applied both for language modelling or sequence generation [11, 21, 87].

B Appendix B: Additional Visual Stories

In this chapter we are going to give more examples of visual stories along with the respective generated storylines that we got during the ultimate phase of our experimentation, since they consist the main product of our work and most of qualitative analysis was conducted around them.

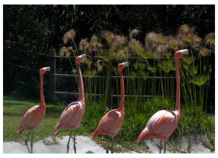
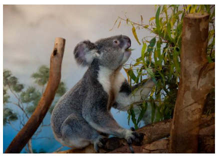
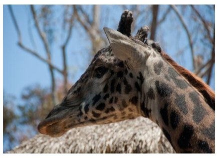
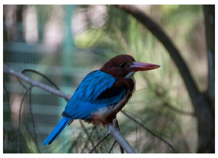
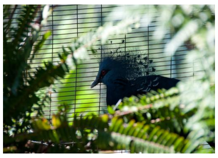
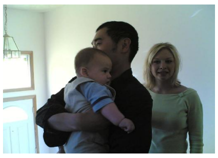
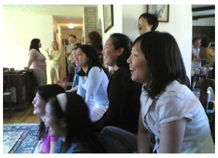
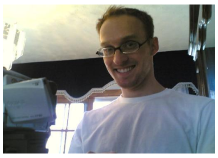
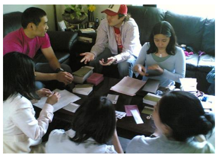
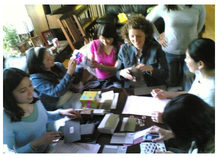
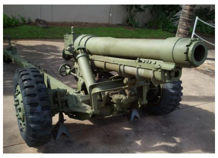
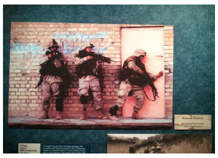
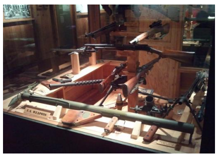
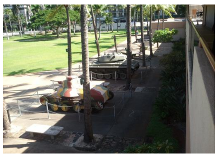
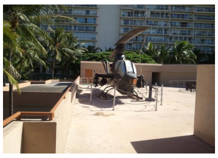
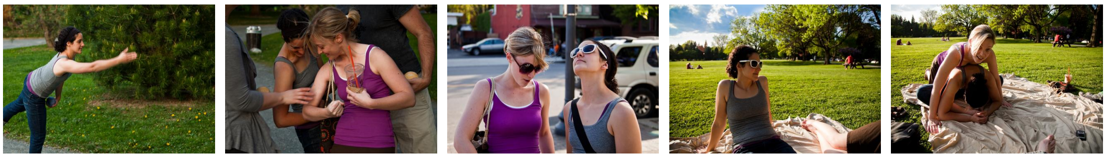
Visual Story: $(ind = 2563)$					
$\mathcal{F}_1$:	They saw flamingos.	And a kitty!	The giraffe was sleeping.	The birds were tweeting.	The ferret was on the fence.
$\mathcal{F}_2$:	The flamingos were at the zoo.	The kitty was looking for food.	The giraffe was relaxing.	The bird was watching the other birds.	The ferret was watching them too.
$\mathcal{F}_3$:	There was also a family of flamingos.	There was a kangaroo hanging out.	I went to the zoo and saw a lot of animals.	The giraffe was relaxing.	I saw a bird that was tweeting.
$\mathcal{F}_4$:	The flamingos were very beautiful.	The kangaroo was hanging out in the trees.	The giraffe was very lazy.	The bird was very pretty.	There was a bird at the zoo.
Humans:	The animal species at the zoo were so diverse.	It was a very unique place.	My friend almost passed out from excitement after seeing her favorite animal, the giraffe.	We had so much fun on our trip to the zoo!	We saw so many animals.
Visual Story: $(ind = 80)$					
$\mathcal{F}_1$:	My brother and his wife were so happy to see each other for the first time in over a year.	We had a family get together for my brother's birthday.	My brother even brought his camera to capture the memories.	We all ate and drank and had a great time.	We all talked and told stories and laughed.
$\mathcal{F}_2$:	This couple is also from my town and is celebrating the end of their marriage.	Family and friends gathered together for a happy event.	This guy was the photographer and he took lots of photos.	This was the group shot at the end and everyone was happy.	The bride and groom are waiting to start their wedding.
$\mathcal{F}_3$:	They were all very happy to see me.	I had a great time at the party last week.	There were a lot of pictures being taken.	Everyone was very eager to get to know me.	Afterward we all got together to talk.
$\mathcal{F}_4$:	He was a little surprised when I hugged him.	I took photos of every one.	I went to my friend's house.	They were all very happy to see me.	We all sat down and talked for a while.
Humans:	The man is holding the baby while the woman is smiling.	The family is making big plans at the table.	The man is smiling widely at the camera.	The whole family is watching television.	The family is getting organized at the table.
Visual Story: $(ind = 60)$					
$\mathcal{F}_1$:	The military had tanks and airplanes.	We took a tour around the military museum.	We saw cool artifacts.	It was a great day and ended with a walk under the palm trees.	We stopped and saw a seagull fighter jet at the hotel.
$\mathcal{F}_2$:	A tank that was used in Vietnam.	A group of men standing in front of a wall.	A museum with lots of war instruments.	A view of the tanks with palm trees.	An helicopter sitting on the roof of a building.
$\mathcal{F}_3$:	There was a tank on display for people to see.	We visited the museum in the town center.	There was a room with a lot of war instruments on display.	We drove around the town to look at all the interesting sights of the museum.	We saw an helicopter on the roof of the building.
$\mathcal{F}_4$:	There were also canons.	We went to a war museum.	There were a lot of instruments.	Afterward we went back home.	We had a great time.
Humans:	Many guns were on display.	The war museum was neat.	The outdoor area was nicely designed.	They had exhibits.	The helicopter is the highlight of the museum.

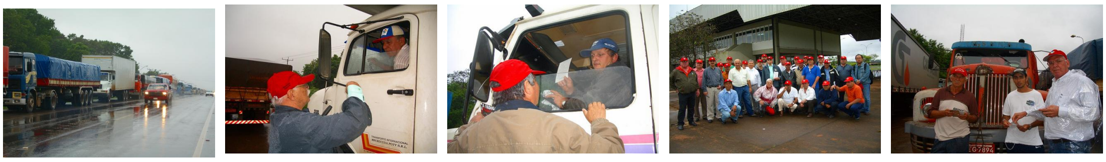
Figure 48. *Additional examples of visual generated stories by the four tested frameworks along with the golden (human written) storylines.*

Visual Story: ($ind = 208$)



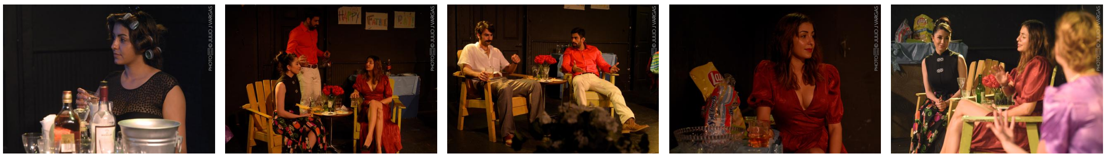
$\mathcal{F}_1$:	The games we played included volleyball, but also lawn games.	Our family got together at the park for a fun day.	Later, we went for a walk around the park.	My sister got some shade and I watched the kids play.	We had a few beers and relaxed.
$\mathcal{F}_2$:	The kids had fun playing with a ball.	The whole family had a picnic at the park.	The family went to the park for a picnic.	Some of the kids got tired and needed a break.	Others just sat around and relaxed.
$\mathcal{F}_3$:	Jane thought she was a model.	She posed in front of her friends.	Jane thought that she was the model in the local play.	Her best friend agreed on that.	But then she said that she wasn't.
$\mathcal{F}_4$:	We went to the park today.	There were a lot of people there.	Afterwards we went back home.	We had a great time in the park.	We took some pictures while we were there.
`Humans`:	The girls haven't seen each other in a few months.	So they went to the park.	And compared muscles with each other.	It was a fun day.	They played lawn bowling.

Visual Story: ($ind = 200$)



$\mathcal{F}_1$:	The company truck full of retirees heading to their destination.	The guy is putting something in the back of the truck.	The man is putting a number in the truck for a job promotion.	The men are meeting to discuss the upcoming company event.	The company representatives are in the parking lot greeting employees.
$\mathcal{F}_2$:	They were all very excited.	I had a great time at the swap meet last weekend.	I got to ride in a lot of different trucks.	There were many people there.	Afterwards, we all got together for a group photo.
$\mathcal{F}_3$:	They were all very tired after all the driving.	There were a lot of them that needed to do a lot.	I had a great time at the meeting last weekend.	I got to meet a lot new and interesting people over there.	They were all standing around talking.
$\mathcal{F}_4$:	We all got in the truck and hit the road.	We got to work right away.	We had a lot of work to do.	I had a great time at the swap meet last week.	Everyone was very nice.
`Humans`:	He told me he was sick of his company that he worked for because his truck kept breaking down.	I drove by and saw a truck pulled over on the highway.	He then quit his company and joined mine.	I gave him my contact info for my trucking company.	He quickly became acquainted with the fellow workers of the new company.

Visual Story: ($ind = 500$)



$\mathcal{F}_1$:	The girls are ready to celebrate another successful year.	The working period ends on a happy note and the focus goes to the holidays.	One last picture of the two on top of the stage for a memorable season.	There's nothing like celebrating another successful season on the last working week.	The police officer stood by to make sure everything was peaceful.
$\mathcal{F}_2$:	The girls were excited to celebrate their anniversary.	Everyone had a great time at the dinner party.	The guys sang karaoke on top of the stage.	There was plenty of food and drinks.	They caught up with each other's lives.
$\mathcal{F}_3$:	These two girls decided to hang out all night.	They made a toast to their new found freedom.	These guys were bored of the TV.	These two seem to be the life of the party.	The night was young when these two decided to meet up.
$\mathcal{F}_4$:	We had a few friends over for dinner last night.	We talked about all kinds of stuff.	Afterwards, we all relaxed on the couch together.	And we had a lot of drinks there.	Everyone was really friendly.
`Humans`:	The guys, Randy and Matt were okay. I would have chose different but they were perfect assholes as they should have been.	I was so styling in my curlers. ”Don't look at me I'm hideous!” I cried my lines.	My pal Erica was so good. She screamed and flung herself all over.	”No way! We don't eat meat here!” We turned up our noses. The wiener monologues were so much fun!	The play I was in last night was fabulous. This is Mandy, Erica and I talking on stage.

Figure 49. *Additional examples of visual generated stories by the four tested frameworks along with the golden (human written) storylines.*

C Appendix C: Using an LLM for evaluation

On section 5.6 and more specifically on paragraph 5.6.2, we utilized an LLM, GPT-4o, for evaluating the results of visual stories produced by our frameworks. That was fulfilled by taking into consideration the coherence, temporal continuity and logical flow that the models exhibited in each transition from one generated sentence to the very next one. Even though a condense visualization of GPT-4o's scores was presented on Fig. 33, here we will give a more elaborating analysis of this evaluation procedure. Moreover, we are going to sustain this evaluating process with the respective results obtained after employing two human annotators to carry out the exact same assessment on those stories that GPT-4o also judged, and eventually we will compare the aftermath of the two proceedings.

C.1 Human vs Artificial judgment

For that comparison, we need Figs. 51 & 52 which depict the detailed scores given by GPT-4o and our human annotators for the transitions from sentence to sentence (each sentence is abbreviated as S$_i$, where $i \in \{1, 2, 3, 4, 5\}$) on the generated stories provided on Figs. 29 & 30. For GPT-4o's grades we use red colored numbers, while the human scores are in green. On the right of each sub-figure, the average transition score achieved per framework in each visual story is displayed as well. Lastly, we need to accentuate that the visual stories rendered on Figs. 51a, 52b and 52c were assessed by the first annotator, while the rest from our second participant and that the judgment scores were based on the same criteria as in the LLM case.

From a brief look on those figures, we can observe that GPT-4o evaluates the coherence and logical flow from sentence to sentence per framework, approximately as our human annotators do, but in the case of the latters a lot more fluctuation is emerging[74]. To better illustrate these discrepancies, we concatenated all the average transition scores per framework created by the LLM and by our human annotators on Figs. 51 & 52 and we plotted them as function of the total number of evaluated models ($5 \times 6 = 30$) on Fig. 50. From there, the landscape is obvious. Both GPT-4o and our participants indeed have comparable grading means with 6.9 and 6.84 respectively, but humans demonstrate fivefold variance with 2 against 0.4. This results, to a more than double *1-std interval* for humans, having 2.82 compared to GPT-4o which accounts for 1.24[75].

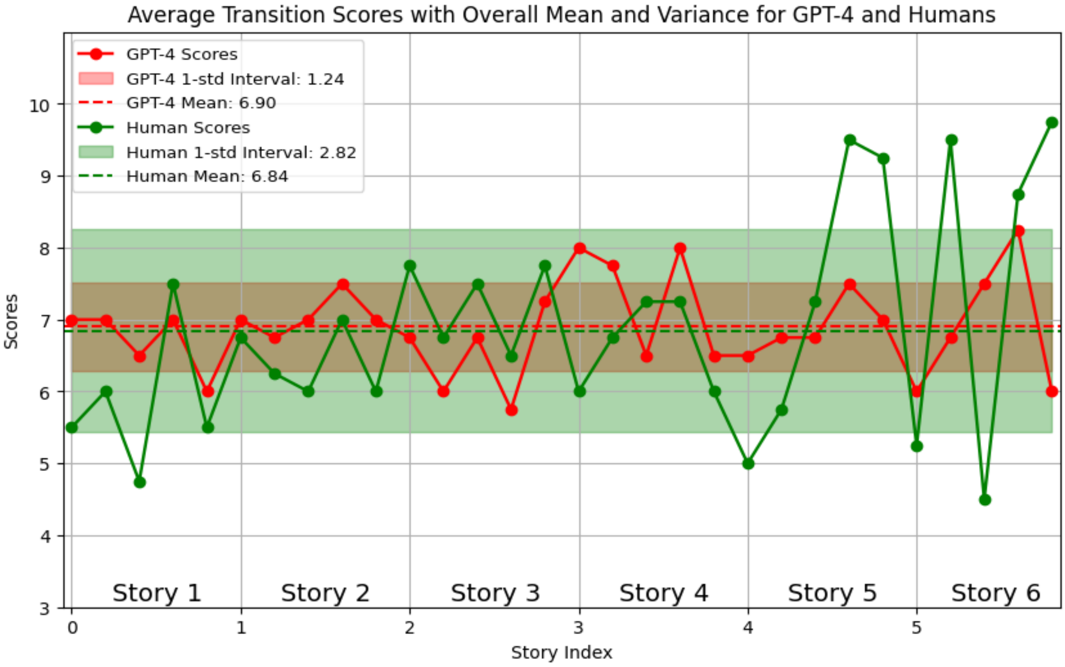
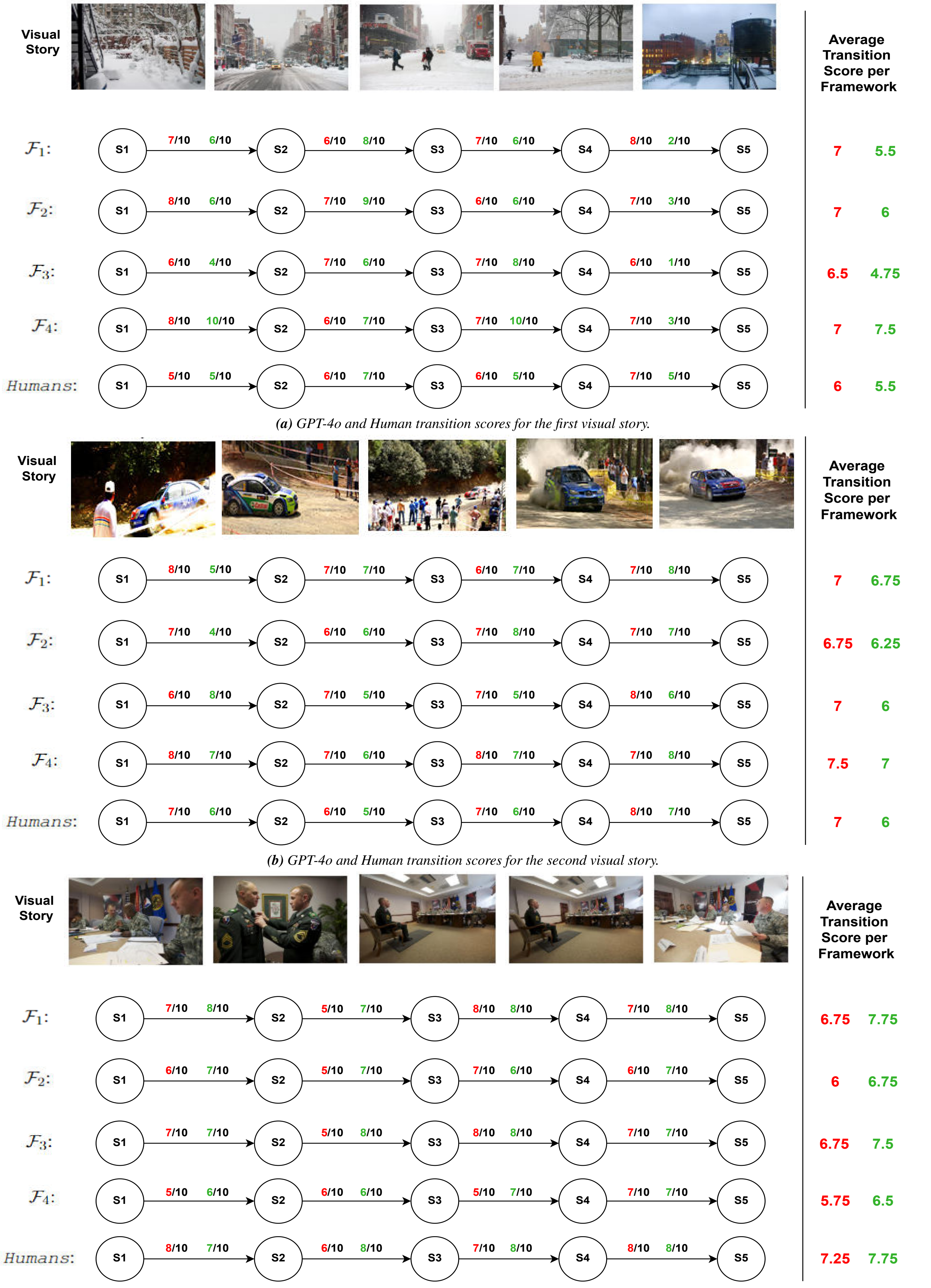


Figure 50. *Average transition scores ascribed by GPT-4o (red) and humans (green), as a function of the total number of evaluated frameworks for the six given visual stories. The plot resulted, after concatenating all the mean values that appear on the right side of Figs. 51 & 52. The respective in color shaded areas indicate the 1-std interval that each kind of judgment showcased whilst the dashed line, their overall mean grading value. On the x-axis, the vertical grid divides the plot into six areas, each of which represents one of the six stories on Figs. 51 & 52. Story 1 corresponds to the first story, Story 2 to the second etc.*

[74] Although this also depends on the individual person that completed the evaluation.

[75] The *1-std interval* or *1-sigma interval* is the area which is one standard deviation above and below the mean. Mathematically, is formulated as the interval $(\mu - \sigma, \mu + \sigma)$, with μ being the mean value, σ the standard deviation and the variance $var = \sigma^2$.


Visual Story
Average Transition Score per Framework
$\mathcal{F}_1$: S1 7/10 6/10 S2 6/10 8/10 S3 7/10 6/10 S4 8/10 2/10 S5 7 5.5
$\mathcal{F}_2$: S1 8/10 6/10 S2 7/10 9/10 S3 6/10 6/10 S4 7/10 3/10 S5 7 6
$\mathcal{F}_3$: S1 6/10 4/10 S2 7/10 6/10 S3 7/10 8/10 S4 6/10 1/10 S5 6.5 4.75
$\mathcal{F}_4$: S1 8/10 10/10 S2 6/10 7/10 S3 7/10 10/10 S4 7/10 3/10 S5 7 7.5
Humans: S1 5/10 5/10 S2 6/10 7/10 S3 6/10 5/10 S4 7/10 5/10 S5 6 5.5

(a) GPT-4o and Human transition scores for the first visual story.

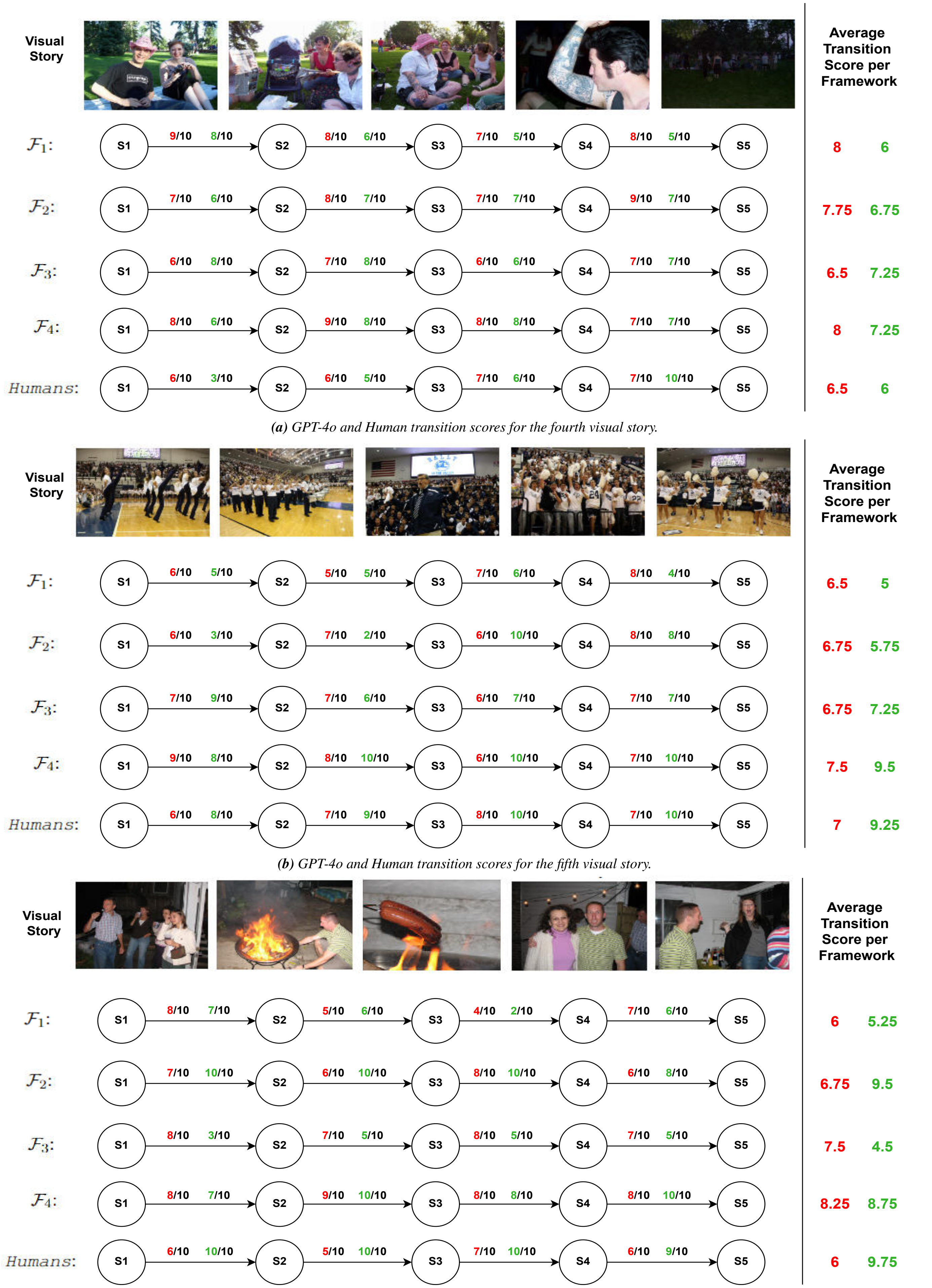
Visual Story
Average Transition Score per Framework
$\mathcal{F}_1$: S1 8/10 5/10 S2 7/10 7/10 S3 6/10 7/10 S4 7/10 8/10 S5 7 6.75
$\mathcal{F}_2$: S1 7/10 4/10 S2 6/10 6/10 S3 7/10 8/10 S4 7/10 7/10 S5 6.75 6.25
$\mathcal{F}_3$: S1 6/10 8/10 S2 7/10 5/10 S3 7/10 5/10 S4 8/10 6/10 S5 7 6
$\mathcal{F}_4$: S1 8/10 7/10 S2 7/10 6/10 S3 8/10 7/10 S4 7/10 8/10 S5 7.5 7
Humans: S1 7/10 6/10 S2 6/10 5/10 S3 7/10 6/10 S4 8/10 7/10 S5 7 6

(b) GPT-4o and Human transition scores for the second visual story.

Visual Story
Average Transition Score per Framework
$\mathcal{F}_1$: S1 7/10 8/10 S2 5/10 7/10 S3 8/10 8/10 S4 7/10 8/10 S5 6.75 7.75
$\mathcal{F}_2$: S1 6/10 7/10 S2 5/10 7/10 S3 7/10 6/10 S4 6/10 7/10 S5 6 6.75
$\mathcal{F}_3$: S1 7/10 7/10 S2 5/10 8/10 S3 8/10 8/10 S4 7/10 7/10 S5 6.75 7.5
$\mathcal{F}_4$: S1 5/10 6/10 S2 6/10 6/10 S3 5/10 7/10 S4 7/10 7/10 S5 5.75 6.5
Humans: S1 8/10 7/10 S2 6/10 8/10 S3 7/10 8/10 S4 8/10 8/10 S5 7.25 7.75

(c) GPT-4o and Human transition scores for the third visual story.

Figure 51. *Transitions scores given by GPT-4o (on red font) and human annotators (on green font) for the first triplet of visual stories from Figs. 29 and 30. On the right side of each figure the average transition scores per framework are presented for a given story. The criteria of assessment on each sentence transition were coherence, temporal continuity and logical flow.*



(a) GPT-4o and Human transition scores for the fourth visual story.

(b) GPT-4o and Human transition scores for the fifth visual story.

(c) GPT-4o and Human transition scores for the sixth visual story.

Figure 52. *Transitions scores given by GPT-4o (on red font) and human annotators (on green font) for the second triplet of visual stories from Figs. 29 and 30. On the right side of each figure the average transition scores per framework are presented for a given story. The criteria of assessment on each sentence transition were coherence, temporal continuity and logical flow.*

Qualitatively, from the transition graphs on Fig. 50, we can argue that GPT-4o showcases moderate similarity with the human participants, particularly with our second annotator on Stories 2, 3 and 4, meaning that their varying is of the same trend in many occasions (increasing/decreasing simultaneously). Besides this, in the same span of stories, the magnitude of the given scores from both kinds of evaluation is quite close. On the contrary, on Stories 1, 5 and 6 we can notice greater dissimilarity between the ranks of the LLM and those from our first evaluator (although some symphony is still present there as well, e.g the first and penultimate generations in Story 6). Finally, it's worth underlining that the very same annotator delivers multiple times both very low (beneath 5) and very high (above 9), average scores to the produced stories.

C.2 Frameworks efficiency according to Human judgment

In addition, we would like to know how our four frameworks behaved during the evaluation from our participants, similarly to what we had seen on sub-subsection 5.6.1 from GPT-4o. To that end, we created an alike to Fig. 33, *box and whisker plot*, but this time regarding the human distribution of the transition scores given to our models. This visualization for the four frameworks, is imprinted on Fig. 53 with the human written stories also shown. Firstly, the far greater variability of the human scores is clear, as the smaller *IQR* here has a range of 2 (for frameworks $\mathcal{F}_1$, $\mathcal{F}_2$), whereas for the LLM that was the largest. At the same time, the greatest *IQR* here, stands at 3.5 for the human category of stories. This behavior is translated to the *whiskers* of each framework as well, with a heyday on $\mathcal{F}_3$ where they span from 1 to 9.

Coming to model performance, we can see that both types of evaluation agree that framework $\mathcal{F}_4$ is generating the most logically continual stories, both in terms of *IQR* with a range from 7 to 10, but also in terms of average score with 7.75. In contrast to GPT-4o's results, here the human stories get the second spot accomplishing a mean value of 7.37 with an *IQR* spanning from 5.75 to 9.25. What is more, it is vivid that the worst performing models are $\mathcal{F}_1$ and $\mathcal{F}_3$ obtaining means of slightly above 6 and 6.2 respectively, an outcome that aligns with what we saw during the automatic evaluation. Due to this reason, $\mathcal{F}_2$ stands above the recently mentioned frameworks, with an average score of 6.84, being the only model where an extreme outlier can be detected (empty dot).

In total, we can claim that both the two human annotators and the utilized LLM give prominence to $\mathcal{F}_4$ of being the framework that produces the most cohesive and semantically compact stories, a fact that was also verified by our main human evaluation process. This is not the case for the rest of the models, where they tend to disagree at least to some extend. Thus, the similarity between the artificial and human assessment, could be characterized at most, as moderate, but still such a conclusion remains vague due to the significant variance observed on the human rankings (Fig. 50). Lastly, for plenitude reasons on Fig. 54, we include the exact prompt that we gave to GPT-4o for grading the sentences transition on the generated stories.

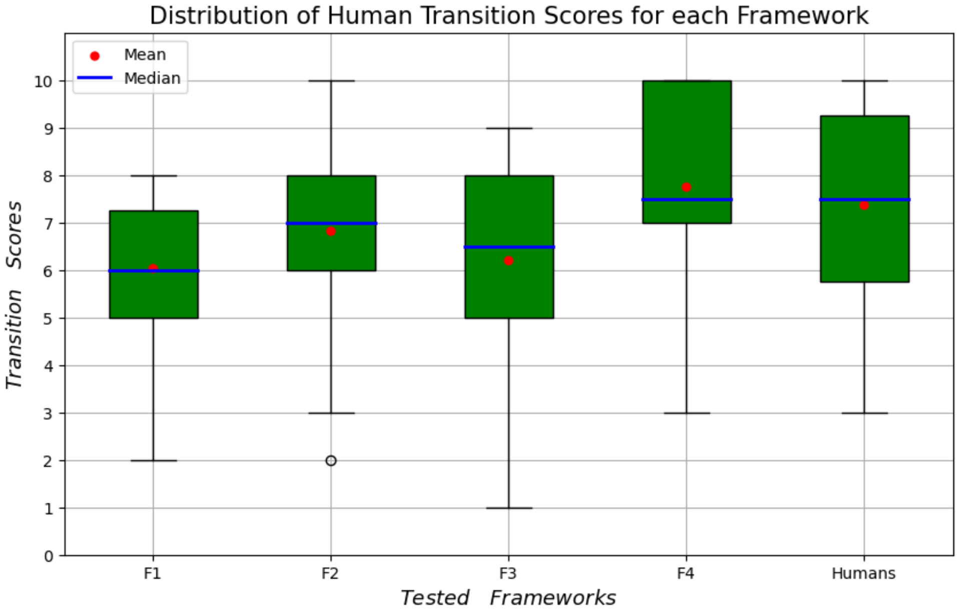
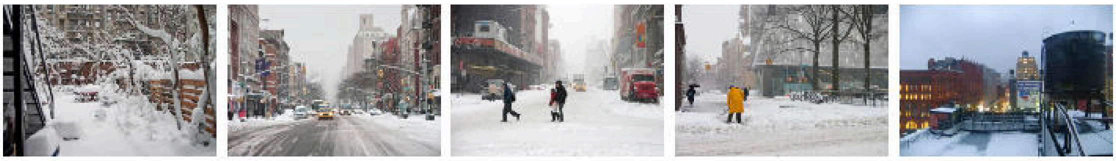
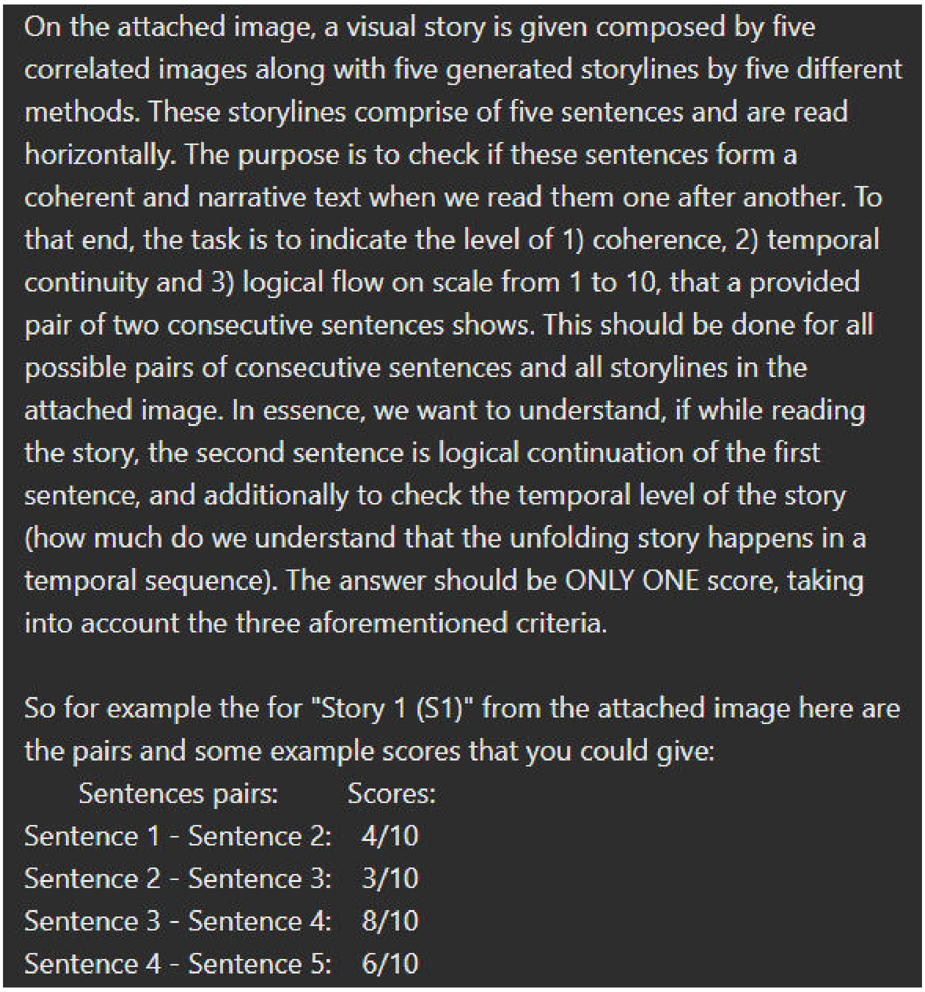


Figure 53. *Distribution of transition scores from sentence to sentence given by human participants for all stories appearing on Figs. 29 & 30. The criteria of assessment on each sentence transition were coherence, temporal continuity and logical flow. The ultimate judge by the participants was a unified, out of 10 mark for all three criteria.*

Visual Story: $(id = 48298)$



Story 1 (S1):	I saw that a tree had branches broken off from the weight of the snow.	There was a lot of snow in my area yesterday.	Some people were complaining that the roads were slippery.	When I got to the other side of town I saw that my car had been buried in snow.	It was a beautiful sunny day out.
Story 2 (S2):	It was a cold and snowy day in the city.	The streets were covered in snow and everyone was out on the streets.	Even the pets were out playing in the snow.	Everyone was bundled up and ready for the cold weather.	The sky was clear and the sun was shining.
Story 3 (S3):	I had a great time at work yesterday.	There were a lot of cars that were blocked off because of the snow.	There was a lot of people that were walking around the city.	Some of the people were faster than others.	When I finished my shift I decided to go back home and see what it was about.
Story 4 (S4):	It was a cold and snowy day in the city.	There were a lot of cars that were stuck in the snow.	We decided to go out for a walk.	It was a lot colder than I thought it would be.	The view from the top of the building was amazing.
Story 5 (S5):	It is a picture of roof tops.	The street is covered in snow.	A man is shoveling the snow.	People are walking in the snow.	The ground and fence are covered in snow.



On the attached image, a visual story is given composed by five correlated images along with five generated storylines by five different methods. These storylines comprise of five sentences and are read horizontally. The purpose is to check if these sentences form a coherent and narrative text when we read them one after another. To that end, the task is to indicate the level of 1) coherence, 2) temporal continuity and 3) logical flow on scale from 1 to 10, that a provided pair of two consecutive sentences shows. This should be done for all possible pairs of consecutive sentences and all storylines in the attached image. In essence, we want to understand, if while reading the story, the second sentence is logical continuation of the first sentence, and additionally to check the temporal level of the story (how much do we understand that the unfolding story happens in a temporal sequence). The answer should be ONLY ONE score, taking into account the three aforementioned criteria.

So for example the for "Story 1 (S1)" from the attached image here are the pairs and some example scores that you could give:

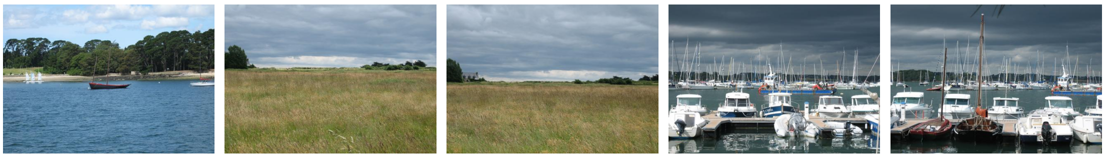
Sentences pairs:	Scores:
Sentence 1 - Sentence 2:	4/10
Sentence 2 - Sentence 3:	3/10
Sentence 3 - Sentence 4:	8/10
Sentence 4 - Sentence 5:	6/10

Figure 54. *An example visual story (and the five generated storylines) given to GPT-4o for evaluation. Along with that, the prompt that we used in order to instruct the LLM to grade our stories based on the desired criteria.*

D Appendix D: Comparison with other works

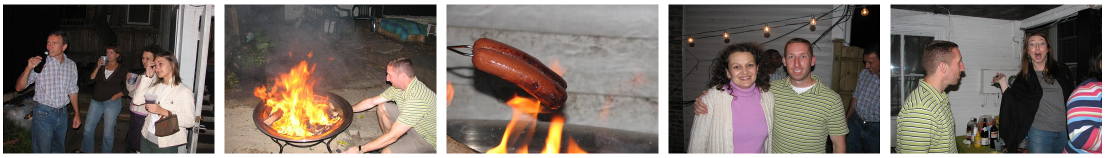
Throughout this project we gave a scrutinizing analysis on the content of our generated stories, by deploying an extensive evaluation procedure with three types of assessment. Nonetheless, we didn't compared our outcome with external studies that have worked on VIST dataset over the last years and achieved state-of-the-art results. To that end, Fig. 55 visualizes 2 examples of generated stories from several state-of-the-art visual storytelling models along with the produced storylines from 3 of our frameworks. Namely, these external models are: AREL [138], GLACNet [66], KG-Story [49], MCSM+BART [12], TAMP [153] and LLaVA [85,121]. In addition, we need to note that the provided results are intended only for intuitive contrast, as we didn't proceed to any further in-depth qualitative/numerical comparisons. This is left for future work.

Visual Story: *Comparison*



AREL:	I went on a boat trip to the lake.	This is a picture of a lake.	This is a picture of a field.	The water was clear and the water was calm.	The boats were docked in the water.
GLACNet:	The cruise ship was getting ready to go.	They were off to a great spot.	The beach was beautiful.	There was a lot of boats.	It was a very nice day.
KG-Story:	We had a great time at the lake.	There were so many boats.	It was very beautiful.	I spent all day out on this field.	And even saw one boat.
MCSM+BART:	We had a nice summer in [location].	The fields were absolutely gorgeous.	We also had a farm that looked like a real farm all around.	There were even boat campsites.	Some of the boats were out at night to camp.
Ours ($\mathcal{F}_2$):	After that, they went down to the river and looked on the sights.	Then, they went to the field to take some pictures of the scenery.	They ended the day by taking pictures of some of the fields.	They stopped by the marina to take pictures.	The family was vacationing in a new city.
Ours ($\mathcal{F}_3$):	My dad is a great boater and an amazing shot.	My parents grew up here and I love the grassy fields.	I was a little scared of walking across the grass-covered fields.	I finally got a chance to go boating the other day.	Dad showed me how fun it was to drive all the way out to the ocean with all these boats.
Ours ($\mathcal{F}_4$):	We saw a boat in the water and decided to go for a ride with it.	Then, we went to the field to look at some things.	Then, we drove to the other side of town.	After that, we saw a marina on the other end of the dock.	There were a lot of people already parked on that side of the road.

Visual Story: *Comparison*



AREL:	The friends were having a great time at the party.	The fire was <UNK> and <UNK>.	The fire was <UNK> and <UNK>.	The guys were having a great time.	We all had a great time and had a great time.
GLACNet:	The family was having a party.	They played some fire.	Then they had a big bonfire.	Everyone was happy.	It was a great day.
TAPM:	The group of friends got together for a bonfire.	We had a lot of fun cooking.	The barbecue was delicious.	We took a lot of pictures.	The night ended with a few drinks.
Ours ($\mathcal{F}_2$):	The whole family was there.	We had a cookout.	We also barbecued some hot dogs.	We all had a great time.	It was a very nice party at my friend's house last week.
Ours ($\mathcal{F}_3$):	All the family was there to celebrate their loved ones.	Last night's cookout was a lot of fun.	They barbecued tons of meat on the grill.	We invited all of our friends over.	It was a very happy occasion for everyone.
Ours ($\mathcal{F}_4$):	All of the family got together to have a cookout.	At the end of the day, we all sat around the fire and enjoyed each others company.	We barbecued hot-dogs and hamburgers.	We had a great time together.	We laughed and told many stories.
LLaVA:	In the dark, a group of friends huddled around a fire, their faces lit up with the warmth of the flames.	The fire crackled and roared, casting dancing shadows on their faces.	One friend, a bit too eager, accidentally dropped a hot dog into the fire, causing a burst of flames and laughter.	The friends cheered and clapped, their joy infectious.	As the night wore on, they shared stories and laughter, the fire slowly dying down, leaving behind only the memories of their fun-filled evening.

Figure 55. *Two examples of visual generated stories by some state-of-the-art Visual Storytelling models, along with the respective generated stories from three variants of our proposed framework. Highlighted blue words visually relate to the input images.*