Emotion-centric Image Captioning using a Self-Critical Mean Teacher Learning Approach

by

Aryan Yousefi

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science (MSc) in Computational Sciences

The Office of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Office of Graduate Studies/Bureau des études supérieures

Title of Thesis Titre de la thèse	Emotion-centric Image Captioning using a Self-Critical Mean Teacher Learning Approach
Name of Candidate Nom du candidat	Yousefi, Aryan
Degree Diplôme	Master of Science
Department/Program Département/Programme	Computational Science
Date of Defence Date de la soutenance	November 07, 2022

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Azam Asilion Bidgoli
(Committee member/Membre du comité)

Dr. Dipali Kasat
(External Examiner/Examinateur externe)

Approved for the Office of Graduate Studies
Approuvé pour le Bureau des études supérieures
Tammy Eger, PhD
Vice-President Research (Office of Graduate Studies)
Vice-rectrice à la recherche (Bureau des études supérieures)
Laurentian University / Université Laurentienne

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

Image Captioning is the multi-modal task of automatically generating natural language descriptions based on a visual input using various Deep Learning techniques. This research area is in the intersection of Computer Vision and Natural Language Processing fields, and it has gained an increasing popularity over the past few years. Image Captioning is an important part of scene understanding with various extensive applications, such as helping visually impaired people, recommendations in editing applications, and usage in virtual assistants. However, most of the previous work in this topic has been focused on purely objective content-based descriptions of the image scenes. The goal of this thesis is to generate more engaging captions by leveraging human-like emotional responses in the captioning process. To achieve this task, a Mean Teacher Learning-based method has been applied on the recently introduced ArtEmis dataset. This method includes a self distillation relationship between the memory-augmented language models with meshed connectivity, which will be first trained in a cross-entropy based phase, and then fine-tuned in a Self-Critical Sequence Training phase. In addition, we propose a novel classification module by decreasing texture bias and encouraging the model towards a shape-based classification. We also propose a method to utilize extra emotional supervision signals in the caption generation process, leveraging the image-to-emotion classifier. Comparing with the state-of-the-art results on ArtEmis dataset, our proposed model outperforms the current benchmark significantly in multiple popular evaluation metrics, such as BLEU, METEOR, ROUGE-L, and CIDEr.

Keywords: Image Captioning, Computer Vision, Natural Language Processing, Mean Teacher Learning, Self-Critical Sequence Training

Acknowledgments

I would like to express my deepest appreciation to my supervisor, professor Kalpdrum Passi, for the continued guidance and valuable insights that made this achievement possible for me. I am proud of and grateful for my time working with professor Passi.

Thanks to the Laurentian University of Sudbury for providing me with the financial aid that made this process easier for me. Also, many thanks to the Office of Graduate Studies for their great help and assistance throughout my studies.

Finally, I am truly grateful to my family, that endured this long process with me, and for the love, support, and motivation that they always provided me with. It was an absolute privilege to have them close by my side on this journey.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1 Introduction

Image captioning is an important step towards developing scene-understanding ability in deep learning models for plenty of purposes. This multi-modal task deals with both textual and visual modalities with the goal of generating fluent natural language descriptions according to the contents of a digital image [1]. The applications of image captioning include usage in virtual assistants, helping visually impaired people to get a better perception of their surroundings, and industrial quality control. This field is a combination of both Computer Vision and Natural Language Processing (NLP), which are described in the following sections.

1.1 Deep Learning

Deep learning is a subset of machine learning that attempts to imitate the human brain's behavior in the process of gaining knowledge from large amounts of data. It differs from machine learning with regards to the type of data it works with and its approach to learning. Deep learning does not necessarily require some of the data pre-processing needed in machine learning. These algorithms are also applying an automated feature extraction while being more robust to unstructured data, such as images, text, and videos. Deep learning is embedded in our everyday products, such as digital assistance or credit card fraud detection. Deep learning can be considered as a combination of statistics and predictive modeling, and in a sense, it is an automation of predictive modeling.

Artificial neural networks are the main components of the deep learning field, which were limited by the available computational power until recently. These are inspired by the human brain, where each neuron receives thousands of signals from other neurons. The neural networks consist of layers of nodes connected to the adjacent layers. The depth of the network is directly correlated with the number of layers. These nodes are assigned with their corresponding weights, which get adjusted during the training phase of the network. The last layer is responsible for compiling weighted inputs of the previous layers to generate a final output. A brief explanation of some deep learning concepts is mentioned below:

- **Neuron:** Similar to the human brain, artificial neural networks consist of neurons. They receive the information, process it, and generate an output that can be either passed to other layers for further processing or considered as final output.

- **Weights:** Each input to a neuron is multiplied by the weight associated with that neuron. The weights in the neural network are initialized randomly. As the training process goes on, the weights that are considered more important will increase, and the weights that are considered less important will decrease.

- **Bias:** The bias is a linear component applied to the input in addition to the weights, which is analogous to the constants in linear functions. Figure 1 is an illustration of a neuron receiving multiple inputs and applying the corresponding weights and bias to them.

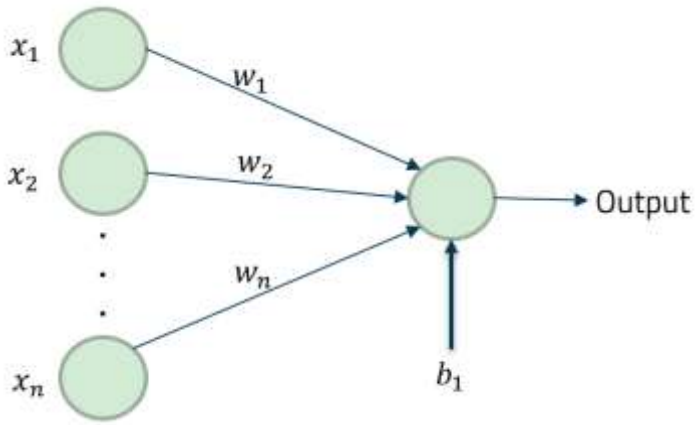


Figure 1. A neuron in a neural network receiving the inputs with their corresponding weights, along with the bias.

- **Activation Function:** The weights and biases are the linear portions of operations applied on neuron inputs, while the activation function is utilized to add non-linearity to this linear combination. Some commonly used activation functions are Sigmoid, Softmax, and ReLU.

- **Input / Hidden / Output layer:** As the names suggest, the input and output layers are the first and the last layers of a neural network, responsible for receiving the input and generating the output, respectively. On the other hand, hidden layers do the processing of their inputs and pass the result to the adjacent layers. Figure 2 is an illustration of these layers.

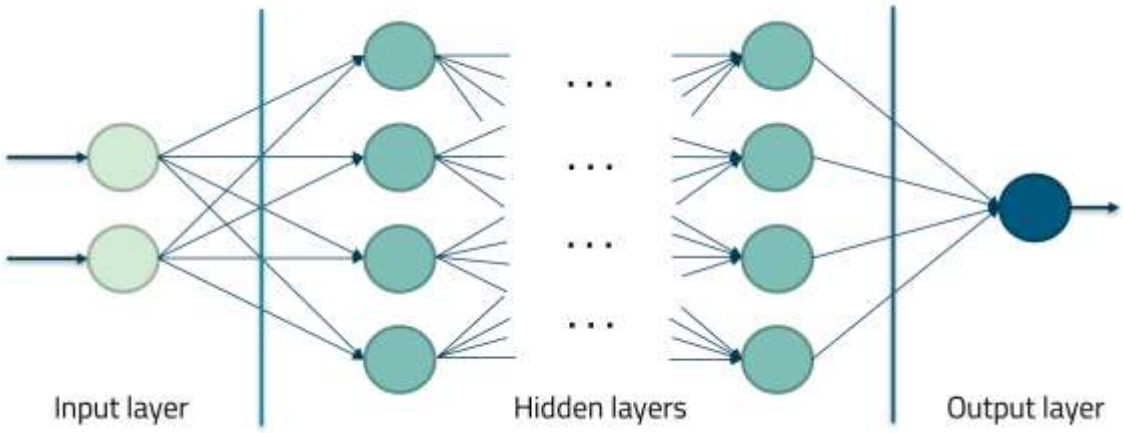


Figure 2. Input, hidden, and output layers of a neural network.

The flow of the information from the input layer to the hidden layers, which finally reaches the output layer, is called forward propagation.

- **Loss Function:** The goal of a neural network is usually to generate a prediction as accurately as possible. The loss function is utilized to measure the accuracy of the network, and it will penalize it in case of erroneous predictions. Therefore, the purpose of the neural network is to optimize its parameters to minimize the loss function.

- **Gradient Descent:** Knowing the objective of the neural networks, which is to minimize the loss function, we need an algorithm to reach that goal. Gradient descent is an iterative optimization algorithm to find the minimum of the loss function as a differentiable function. This algorithm takes steps proportional to the negative of the gradient of the function.

- **Learning Rate:** An indicator is required to determine the amount of minimization in the loss function in each optimization iteration. Therefore, the learning rate is utilized to control this matter. However, the value of the learning rate should be chosen carefully

since a high learning rate can lead to missing the global minimum, and a low learning rate causes the algorithm to take a very long time to converge. An illustration of the impact of choosing different values as the learning rate is provided in Figure 3.

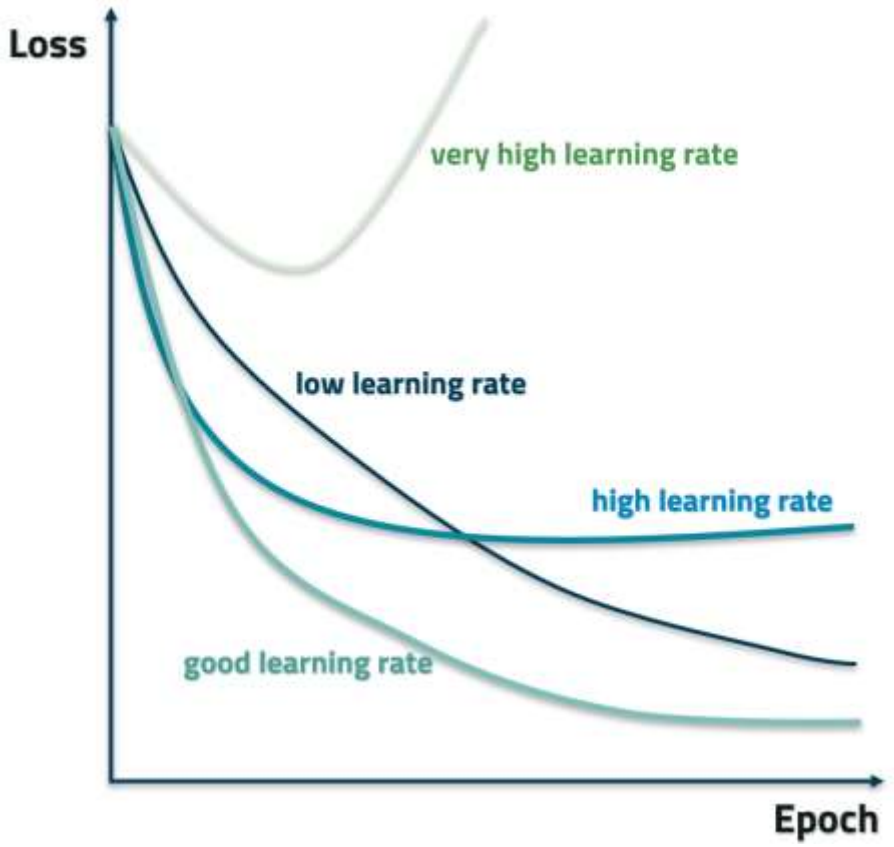


Figure 3. Impact of choosing different learning rate values on optimizing the loss function.

- **Back-Propagation:** As mentioned before, the weights and biases of the neural network are initialized randomly. After the completion of each forward propagation iteration, the back-propagation process computes the gradient of the loss function with respect to the weights based on an example input-output. In addition, the weights get updated based on the computed error values to reduce the inaccuracy. This operation is done starting from the output layer towards the input layer; hence, it is called back-propagation.

- **Batches:** It is not efficient to feed the entire data to the neural network at once for several reasons, such as memory limitation and generalization issues. Therefore, the data is fed to the network in randomly selected batches of equal size. Basically, the amount of data utilized in a single forward propagation and back-propagation iteration is called a batch of data.

- **Epochs:** A single iteration of passing all batches through the network is called an epoch, which is consisted of one forward propagation and one back-propagation. A higher number of epochs can lead to higher accuracy. However, this can also impact the generalization ability of the network, which is called the over-fitting problem.

- **Dropout:** There are multiple techniques to help overcome the over-fitting problem. One of these techniques is called dropout, where randomly selected neurons of the network are dropped. The reasonable range for dropout value is between 0.5 and 0.8.

1.2 Computer Vision

The focus of Computer Vision is to extract meaningful information from digital images, videos, and other visual inputs. Recent advances in artificial intelligence, especially neural networks and deep learning, have enabled researchers to tackle multiple Computer Vision problems more efficiently, which include object detection [2], action recognition [3], and motion tracking [4]. These deep learning models rely on huge volumes of labeled data to figure out the mathematical equation that enables the model to predict proper labels for unseen visual inputs. The human process of visual perception is extremely complicated, and the ultimate goal is to reach a human-level visual cognition in the models.

Let’s have a brief overview of the definitions of some computer vision applications:

- **Object Classification:** Based on the objects in the visual input, which category of entities do they belong to? (e.g., vehicles, animals)
- **Object Detection:** Locate different objects in the visual input.
- **Object Recognition:** Finding out what objects exist in the visual input and where they are.
- **Object Segmentation:** Which portions of the image are related to different objects.

Convolutional Neural Networks or CNNs are widely used in computer vision to extract features of an input image. In this process, a level of importance is assigned to different portions of an image through learnable weights and biases. In general, the spatial and temporal dependencies of an image are captured through the relative filters.

1.3 Natural Language Processing

Natural Language Processing or NLP is concerned with the interaction of computers and human language, with the task of understanding human language as it is written and spoken. Computational linguistics is combined with statistical, deep learning, and machine learning models to solve different problems like machine translation [5], speech recognition [6], and sentiment analysis [7]. A brief review of some of the most popular applications of NLP are mentioned below:

- **Machine Translation:** Automated translation from a particular language to a target language (e.g., German-to-English).

- **Speech Recognition:** Effectively transform spoken language to a format that can be processed by computers. (e.g., Siri and Alexa)

- **Sentiment Analysis:** Extracting different sentiments from textual data, such as opinions and emotions. (e.g., social media monitoring)

- **Text Classification:** The goal is to process, understand, and classify unstructured text, which includes sentiment analysis as part of its approach.

- **Text Summarization:** As the name suggests, the goal is to automatically summarize textual data according to the most important information conveyed by the text.

Similar to computer vision, the advances in NLP were boosted significantly as a consequence of the enhancements in deep learning and neural networks. Recurrent Neural Networks (RNNs) are one of the most popular architectures to deal with NLP problems since they are intuitively capable of handling sequential data, such as time-series and textual data. This capability is because, unlike feed-forward networks, an internal memory incorporates the previous computations in addition to the current input.

Long Short-Term Memory or LSTM is a frequently used variation of RNNs, utilized to overcome the vanishing gradient problem in these networks by introducing forget gates. The vanishing gradient problem happens when the network has more layers, and the back-propagated long-term partial derivatives of the loss function tend to zero. The internal gates control the flow of information by learning which portions of a sequence is important and which portions should be ignored. Figure 4 is an illustration of an LSTM cell, where the input vector, hidden state, and cell state vectors go through the sigmoid and hyperbolic tangent functions, along with the other

arithmetic operations. Another similar network is Gated Recurrent Unit or GRU, with the difference of using fewer gates to control the information flow.

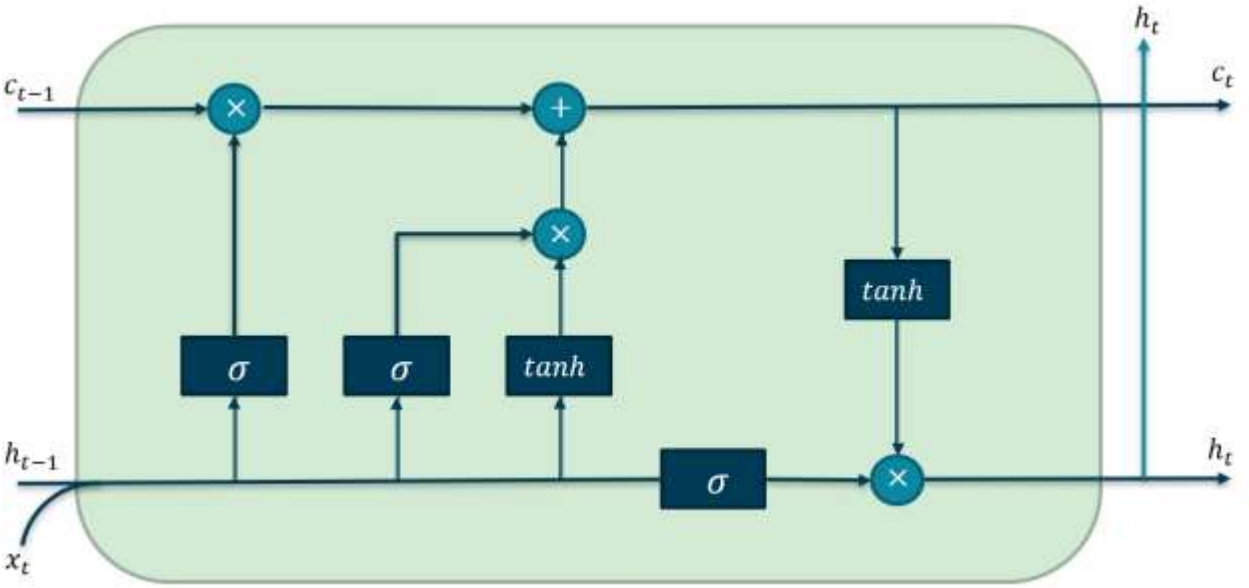


Figure 4. A Long Short-Term Memory or LSTM cell structure.

1.4 Multi-modal Deep Learning

The existing developments in uni-modal deep learning has led to a new area of research, which is called multi-modal deep learning. Modality is the format of storing representations of a particular type of information. Uni-modal approaches are insufficient to model the human behaviour in an accurate manner, since human's perceptions from outside world follows a multi-modal nature containing different senses like touch, smell, taste, hear, and sight. On a daily basis, people collect and process various information from numerous media sources, such as texts, images, and videos.

Our ultimate goal in deep learning is to properly emulate human behaviour in the designed models, which cannot be achieved using only uni-modal approaches. Hence, researchers are tending to use

multi-modal deep learning to overcome these shortcomings. Some of the most important multi-modal deep learning tasks are mentioned below:

- **Image Captioning:** Generating textual descriptions for an input image. (Modalities: text + image)
- **Video Captioning:** Similar to image captioning, video captioning is the task of generating textual descriptions based on the contents of an input video. (Modalities: text + video)
- **Visual Question Answering:** Providing answers in a few words or short phrases based on questions related to the contents of an input image or video. (Modalities: text + image, text + video)
- **Multi-modal Speech Synthesis:** Developing systems capable of generating human-like speeches in a waveform format through Text To Speech (TTS) systems, after generating the textual modality.

1.5 Image Captioning

As mentioned earlier, the image captioning field is a hybrid field including both computer vision and NLP techniques. Recent advances in these deep learning fields have led to an increasing attention towards image captioning. The goal is to automatically generate accurate natural language descriptions according to the contents of visual input. Image captioning is concerned with textual and visual modalities; hence, it requires models capable of handling both modalities and their fusion. Figure 5 shows examples of purely objective content-based automatically generated captions.

Some of the applications of image captioning are mentioned below:

- **Medical Field:** Assisting visually impaired persons by helping them to understand the environment and providing textual descriptions and interpretations about medical images.

- **Social Media:** Generating captions for images in social media platforms, which are sometimes associated with the tags provided by users. This process can include eliminating erroneous and incorrect tags as well.

- **Industrial Applications:** Assisting in quality control process by manufacturers and decreasing human involvement in various procedures. The robots employed can leverage the generated captions to make informed decisions.

- **Agricultural Applications:** Providing descriptive data about plant requirements and fruit detection. In addition, image captioning can be utilized to assist fruit segmentation in order to decrease the need for human labor.

Figure 5. Examples of purely objective content-based captions generated by the model proposed by [13].

There are some important problems in the image captioning task, such as the parallax error which can happen with human eyes as well. This happens when objects are not detectable from specific angles. As another example, the scene clutter issue is another obstacle for a proper scene understanding which leads to erroneous captions to be generated. This problem happen when the scene is too chaotic, and it is difficult for the model to detect the objects.

Our proposed model is capable of generating affective utterances describing the triggered human-like emotional responses stemming from visual stimulus. The generated utterances are according to both the visual features and emotion-based supervision signals. The proposed pipeline consists of:

- auxiliary text-to-emotion and image-to-emotion classification tasks.
- A visual encoder to extract visual features from the input image.
- Two interconnected language models following a transformer-based encoder/decoder architecture.

1.6 Contributions

In this thesis, we introduce three main contributions:

(1) A novel approach for the image-to-emotion classification task by decreasing the texture bias of the classifier and encouraging the model towards a shape-based classification. This is because of the differing local textures in our input images (artworks) in comparison to the real world.

(2) A transformer-based architecture composed of stacks of memory-augmented encoders and meshed decoders to process the visual features and generate one word at each time-step accordingly.

(3) Achieving state-of-the-art performance using the Nemesis on the ArtEmis dataset. We prove that our proposed pipeline, which consists of two language models interacting in a self-critical mean teacher learning-based approach, is a promising path toward generating more human-like, emotionally rich captions. We also propose a variation of Nemesis, which is supervised by extra emotional signals leveraging the proposed image-to-emotion classifier, called the Emotionally Grounded Nemesis or EGNemesis.

1.7 Organization of the Thesis

This thesis is organized as follows:

- Chapter 1 introduces the topic of image captioning and the contributions of this research.
- Chapter 2 is a comprehensive literature review of the previous studies in the image captioning field.
- Chapter 3 provides the definition of the different techniques and methods utilized in the proposed model.
- Chapter 4 includes the architecture, pipeline, and training strategies of the proposed model.
- Chapter 5 elaborates on the experimental settings, including the details about the utilized dataset, visual encoding modes, and implementation details.
- Chapter 6 is about the evaluation process and the obtained results, which includes details about the evaluation metrics, ablation studies, and comparison with the state-of-the-art.
- Chapter 7 provides a conclusion of the research and discusses the potential future direction of emotion-centric image captioning.

Chapter 2
Literature Review

2 Literature Review

Over the past years, increasing efforts have been made in the field of image captioning to improve different aspects of the process. These endeavors are related to both the computer vision and NLP parts of the problem. The enhancement in the processing of the visual input provides the opportunity for a better understanding of the scene. At the same time, the improvement in the linguistic generation leads to a more articulated description of the extracted visual features. The two main components of an image captioning pipeline are the visual encoder and the language model. The visual encoder is responsible for extracting the visual features and passing them to the language model to generate the corresponding linguistic description.

In this chapter, we will have a comprehensive literature review of the previous image captioning studies. First, we will discuss some early studies. Then, we will review the more recent studies with respect to their approach to the visual encoding process and the proposed language models separately. Next, we will review some of the most popular image captioning datasets and image captioning variants, followed by the performance comparison between the mentioned studies. Finally, the motivation of this thesis will be discussed.

2.1 Retrieval-based and Template-based Image Captioning

The early studies were based on retrieval-based [8], [9], and template-based methods [10], [11], [12], where the captions are directly retrieved from an existing database causing the captions to be repetitive and not completely specific to the input image.

Pan *et al.* [8] introduced a retrieval-based method in the proposed image captioning model, where the correlations between image features and keywords are discovered first, then the relative keywords according to the new images are retrieved from the pre-existing database.

Yang *et al.* [10] proposed a template-based technique where the image description process is based on predicting proper nouns, verbs, scenes, and propositions that will eventually be put into an existing template for the sentence structure.

2.2 Visual Encoding

As an early stage in an image captioning pipeline, the spatial information and structure are extracted from our input image to achieve a proper visual representation.

There are different approaches to tackle the visual encoding task; we divide these approaches into the following four categories based on their approach to processing the visual input: 1) non-attentive CNNs to extract the global features. 2) Grid-based and region-based feature extraction using additive attention. 3) Graph-based attention. 4) Feature extraction using self-attention.

2.2.1 Non-attentive CNNs to Extract the Global Features

Non-attentive CNNs to extract the Global Features utilize the output of the last layers of the CNNs, which includes the high-level attributes of the image. The advantage of using global features is the

simplicity and the inclusion of the entire visual input. However, there are some drawbacks as well, such as leading to the extraction of over-generalized features, which causes the generated captions to be imprecise and not detailed enough. The general process is illustrated in Figure 6, where a non-attentive CNN receives the whole image and extracts the global features.

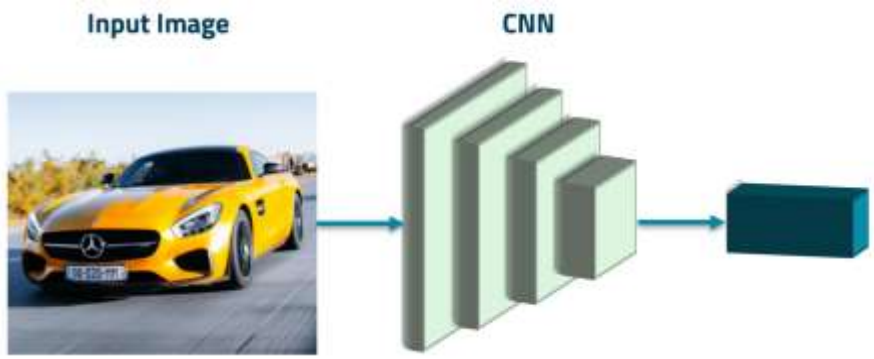


Figure 6. Non-attentive CNNs to extract global features.

Vinyals *et al.* [13] proposed "Show and tell", which utilizes a CNN for image representation. The utilized CNN applies a batch normalization technique to each training mini-batch. The batch normalization process fixes the means and variances of the layer inputs, it can act as a regularizer, and it also provides the ability to choose higher learning rates. Karpathy *et al.* [14] used a CNN pre-trained and fine-tuned on ImageNet [15]. The image representation considers the top 19 detected locations along with the whole image, which gives the advantage of learning a joint embedding space leading to a better scene-understanding capability. This process can be formally defined by the equation below:

$$I = W_m[CNN_\theta(P_b)] + b_m,$$

Where P_b represents the pixels in each bounding box, θ is the parameters of the network, and $CNN(.)$ function transforms the dimensions of the pixels to a desired value.

Mao *et al.* [16] utilized a pre-trained AlexNet [17] and VggNet [18] on the ImageNet dataset, and the CNN gets updated based on the gradient backpropagated from the multimodal layer. However, the main problem is that the model must process the whole global features at each time-step. You *et al.* [19] proposed a model to extract both top-down and bottom-up features from a visual input, gaining a global visual description of the input image. In addition, the model attempts to predict a set of concepts or visual attributes that are most likely to appear in the visual input. The utilization of these attributes can be considered the main advantage of the proposed model.

Wu *et al.* [20] employed a CNN to solve a multi-class classification problem by minimizing an element-wise logistic loss function. The advantage of this model is the utilization of an intermediate image-to-attribute layer; however, the problem is that it requires an extra attribute vocabulary to be determined. Chen *et al.* [21] proposed training a visual parsing tree after extracting the global features using a CNN, which is responsible for extracting entities and their relationships in the visual input. The main contribution of this model is the group-based approach to model the similarities and differences between a group of images to obtain a collaborative captioning process. Rennie *et al.* [22] proposed an FC model which includes a CNN encoding the visual input and then embedding the output through a linear projection. Although this model offers compactness in obtaining the visual representations, it lacks granularity.

2.2.2 Grid-based Feature Extraction using Additive Attention

Additive attention has been utilized to extract grid-based and region-based features in order to overcome the limitations of extracting global features using non-attentive CNNs. In additive attention, the feed-forward network includes a single hidden layer to compute the attention weights. A hyperbolic tangent function is utilized to add non-linearity to the additive attention

equation. The additive attention between two members of two sets of vectors is formally defined as below:

$$f_{att}(a_i, b_j) = W_3^T \tanh(W_1 a_i + W_2 b_j),$$

where $a_i \in \{a_1, \dots, a_n\}$, and $b_j \in \{b_1, \dots, b_n\}$. On the other hand, W_1 and W_2 are learnable vectors, and W_3 is the linear combination. The result is referred to as the alignment score, which will eventually get passed through a Softmax function to compute the alignment score indicating how relevant a_i is to b_i.

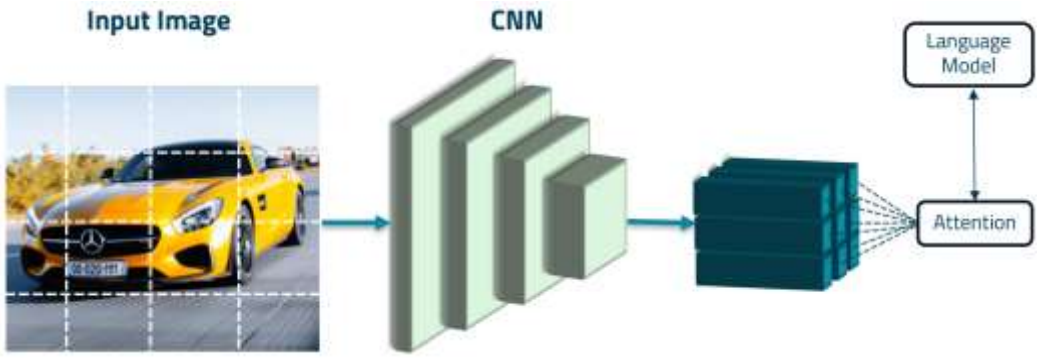


Figure 7. Grid-based feature extraction using additive attention.

Xu *et al.* [23] utilized a CNN to extract D-dimensional feature vectors. This extractor produces N feature vectors corresponding to each grid of the input image.

$$v = \{v_1, \dots, v_N\}, \qquad v_i \in \mathbb{R}^D$$

In addition, features are extracted from the lower convolutional layer to indicate the correspondence between the feature vectors and grids of the input image. Next, additive attention is used to compute the weight for each grid.

Li *et al.* [24] considered the output of the last convolutional layer of ResNet as the spatial features. The dimensionality of this layer is $2048 \times 7 \times 7$, which calculates N grid features that will finally produce the global image feature as defined below:

$$v_{global} = \frac{1}{N}\sum_{i=1}^{N} v_i,$$

where $v_i \in \mathbb{R}^{2048}$ is the extracted feature from the i^{th} grid. Next, these feature vectors are transformed to new feature vectors of dimension D through the below equation:

$$t_i = ReLU(W_v v_i),$$

$$t_{global} = ReLU\left(W_{global} v_{global}\right),$$

where W_v and W_{global} are the weight parameters, and the transformed feature vectors form $T = [t_1, \ldots, t_N]$.

Yang *et al.* [25] introduced a CNN-based VggNet review network that calculates thought vectors by performing several review steps over the encoder hidden states with an attention mechanism. On the other hand, Chen *et al.* [26] suggested not using spatial features as it does not comply perfectly with the attention mechanism. Hence, they proposed a CNN-based architecture that combines spatial and channel-wise attentions. Jiang *et al.* [27] introduced a Recurrent Fusion Network (RFNet) to utilize complementary information from multiple visual encoders. It processes interactions between the image encoder outputs to produce compact information to be used by the decoder in later stages.

2.2.3 Region-based Feature Extraction using Additive Attention

In region-based feature extraction using additive attention, the language model predicts the next word by attending to different regions of the visual input. In this top-down approach, the regions are selected based on the contents of the image, unlike the grid-based approach, where the grid placement is not relevant to the image content. In this technique, an object detector is deployed to propose the image regions. The detected image regions are either employed one at a time or multiple regions can be utilized simultaneously. Figure 8 illustrates region-based feature extraction using additive attention, where the object detector detects different image regions; then, the attention mechanism is applied to the corresponding extracted features while interacting with the language model.

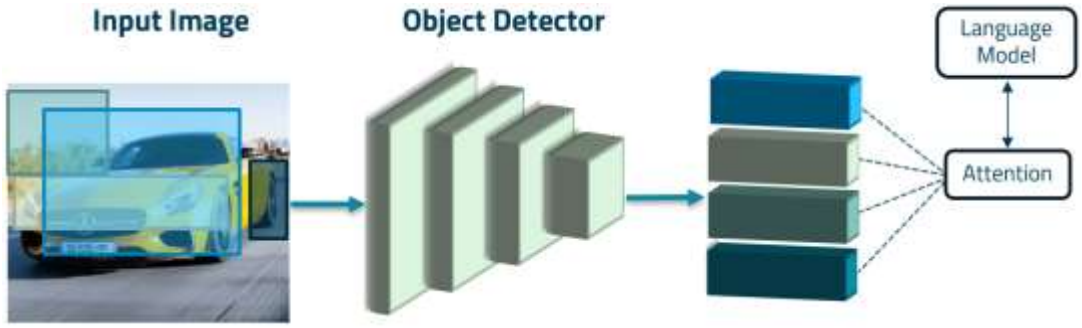


Figure 8. Region-based feature extraction using additive attention.

Anderson *et al.* [28] proposed a bottom-up and top-down attention mechanism that calculates the attention scores with respect to different objects and image regions. The architecture of the visual encoder includes a bottom-up mechanism based on a Faster R-CNN [29] to propose the image regions, while the feature weightings are calculated by a top-down mechanism. The Faster R-CNN detects the image regions in two stages: 1) Image regions are predicted by the Region Proposal Network (RPN), where a non-maximum suppression for each object class using an Intersection-

over-Union (IoU) threshold is applied on the final output of the Faster R-CNN. 2) Small feature maps are extracted based on a Region of Interest (RoI) pooling, which will be passed to the final CNN layers. A Softmax distribution over the object class labels for each bounding box proposal is considered the final output.

Zha *et al.* [30] introduced the Context-Aware Visual Policy network (CAVP), which includes a variation of the attention mechanism that attends to complex compositions of visual regions instead of attending to a single image region at each time-step. This model uses an LSTM to incorporate the historical visual actions, and it is optimized by an actor-critic policy gradient method.

2.2.4 Graph-based Attention

Due to the ability to represent relations in graphs, they can be associated with the image regions to improve the region encodings. The utilized graphs include semantics and spatial relationships. The semantic relationships in the graph are concerned with the actions and reactions between the object pairs. On the other hand, spatial relationships are concerned with the geometrical information related to the bounding boxes of detected objects. Figure 9 is an illustration of graph-based attention, where the object detector selects different image regions; then, after extracting the corresponding features, each of them will be represented by a node, while the edges represent their relationships.

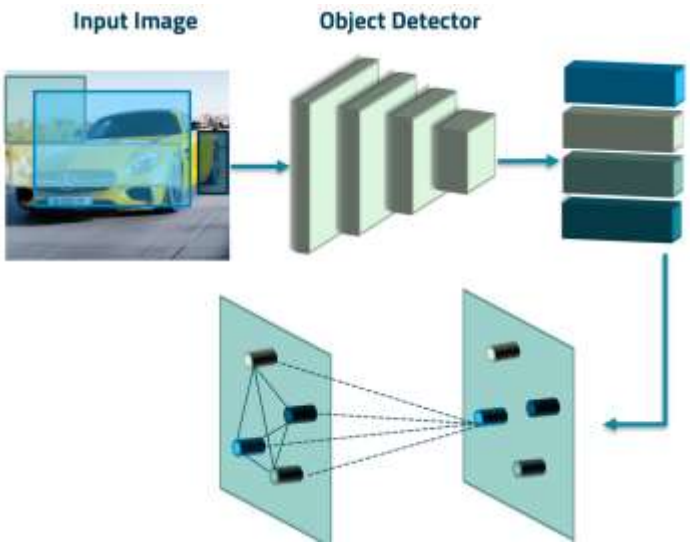


Figure 9. Graph-based attention in visual encoding.

Guo *et al.* [31] proposed using a Graph Convolutional Network (GCN) [32] to model the geometry and semantics of object interactions of the visual input. A geometry graph and a semantic graph are created according to each visual semantic unit, where the semantic and geometrical context-aware embeddings are obtained via the corresponding GCNs. The geometrical relationship graph is constructed based on spatial measures between the object bounding boxes, while the semantic relationship graph is obtained by utilizing a Visual Genome-based [33] pre-trained classifier.

Yang *et al.* [34] utilized a directed scene graph where its object nodes are connected to other nodes representing the corresponding adjectives and relationships. This procedure integrates the priori information obtained from textual data with the visual input. Similarly, Shi *et al.* [35] proposed a scene graph named Caption-Guided Visual Relationship Graph (CGVRG) based on a weakly supervised learning approach. The CGVRG construction process is initialized by a scene graph parser extracting relationship triplets from the ground-truth captions rather than external datasets. Furthermore, the object detection is done by a Faster R-CNN, which returns the corresponding image regions.

Yao *et al.* [36] proposed HIerarchy Parsing (HIP) architecture that integrates hierarchical structure with visual encoding. The image understanding is based on a hierarchical tree including different types of features, where the leaves represent instance-level features, intermediate nodes represent region-level features, and the root represents global features. These features are enhanced by utilizing a tree-structured Long Short-Term Memory (tree-LSTM) network acting as a feature refiner that outputs multi-level representations of the visual input.

2.2.5 Feature Extraction using Self-attention

In self-attention, the elements of the input set interact with each other while producing the output. This process includes five steps mentioned below:

Step-1. During the training process, three weight matrices (W_Q, W_V, W_K) are calculated and multiplied by each of the input vectors. This multiplication will result in three vectors (query vector, value vector, key vector) for each input vector.

Step-2. The current input's query vector is multiplied by the key vectors of other inputs.

Step-3. The result of the multiplication in the second step can lead to a large value which can cause extremely small values after a Softmax function is applied to it. Hence, the result is divided by the square root of the dimension of the key vectors (d_k) as a scale factor.

Step-4. The Softmax function is applied to the self-attention scores calculated according to each query word. Next, the result is multiplied by the value vector.

Step-5. The self-attention output for the given word is calculated by summing up the weighted value vectors generated in the previous step.

The above steps are applied to all input sequences; then the results are concatenated to generate the Q, V, K matrices for queries, values, and keys, respectively. This is formally expressed as the mathematical equation below:

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

The outcome of this equation is the self-attention matrix. Figure 10 is an illustration of self-attention based visual encoding, where self-attention is applied to the extracted features from the detected image regions. These extracted features act as the input sequences described in the mentioned steps.

Li *et al.* [37] proposed a transformer-based architecture named EnTangled Attention (ETA), which enables the transformer to utilize the semantic and visual information simultaneously. This information is extracted by encoders based on self-attention and feed-forward layers. The visual information is extracted by an encoder for the image regions, and the semantic information is obtained by an encoder exploiting knowledge from an external tagger.

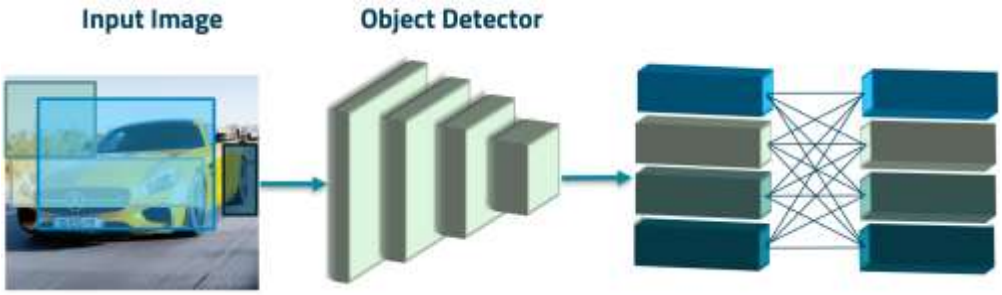


Figure 10. Self-attention based visual encoding.

Herdade *et al.* [38] introduced the Object Relation Transformer, which utilizes feature vectors obtained from the information about the spatial relationship between the detected objects. This

information is extracted by an object detector using geometric self-attention, in which these geometric weights are used to scale the self-attention score.

Guo *et al.* [39] proposed a variation of self-attention called Normalized Self-Attention (NSA), which applies the normalization process to the hidden activations inside self-attention. They also introduced Geometry-aware Self-Attention (GSA) to overcome the transformer-based architecture's limitation in modeling the geometry structure of detected objects in the input image. This is achieved by utilizing the relative geometric relationship between the input objects.

Huang *et al.* [40] proposed Attention-on-Attention (AoA), which utilizes attention scores and current context to calculate "information vectors" and "attention gates." Next, "attended information" is obtained via another layer of attention by applying an element-wise multiplication to both mentioned results of AoA. An illustration of this visual encoder is provided in Figure 11.

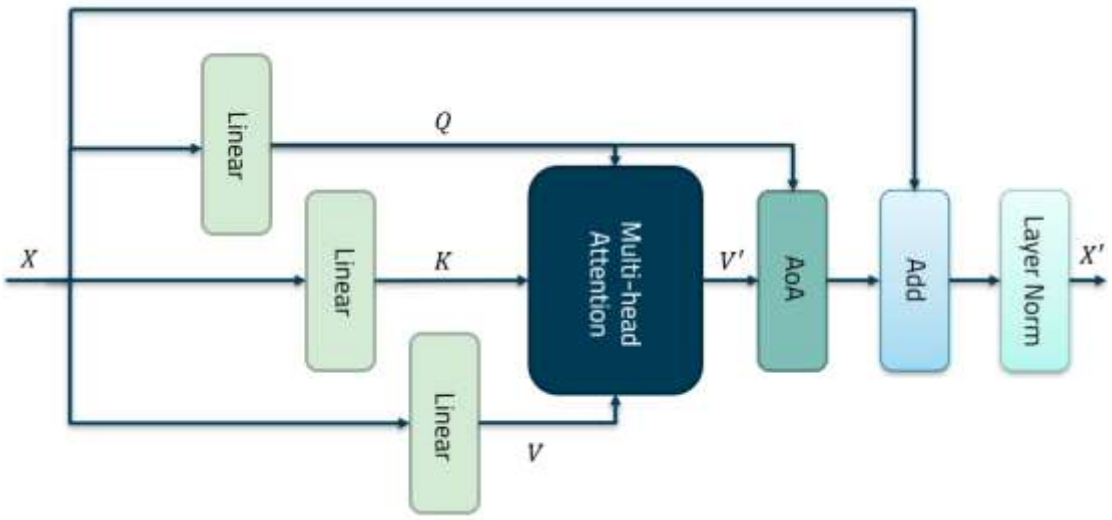


Figure 11. Attention-on-Attention mechanism.

Pan *et al.* [41] proposed the X-Linear attention block, which captures the second-order unimodal and multi-modal interactions by employing both channel-wise and spatial bi-linear attention

simultaneously. These X-linear attentions blocks are integrated together to form the X-Linear Attention Network (X-LAN), which encodes the region-level and higher-order interactions to extract the corresponding features.

Cornia *et al.* [42] introduced the Meshed-Memory transformer (M^2), which utilizes priori knowledge in the image encoding process through additional memory vectors in the self-attention. The proposed model learns multi-level representations for the input image, and it can extract both low- and high-level features via a meshed connection between the encoders and decoders, which is shown Figure 12.

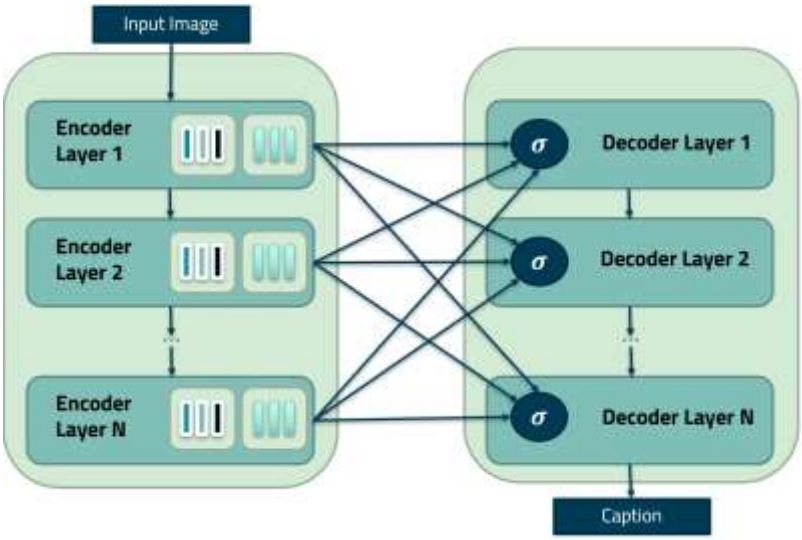


Figure 12. The meshed connectivity between the encoders and decoders in Meshed-Memory architecture.

Dosovitskiy *et al.* [43] proposed Vision Transformer (ViT), which is a pure transformer-based architecture applied to a sequence of image patches. They proved that the utilization of CNNs in image processing is not necessary and causes inductive biases. ViT splits the input image into several patches and calculates their linear embeddings after they are flattened. Next, the positional embedding is added to preserve the actual positions of the patches in the original image. Finally,

the transformer encoder is fed with the sequence, which is usually pre-trained on a huge dataset and fine-tuned for a downstream task afterward. As it is shown in Figure 13, the image is divided into different patches, then the patches, along with their positional encodings, are passed to the encoder module to extract the features. Next, the Multi-Layer Perceptron Head (MLP Head) classifies the extracted features.

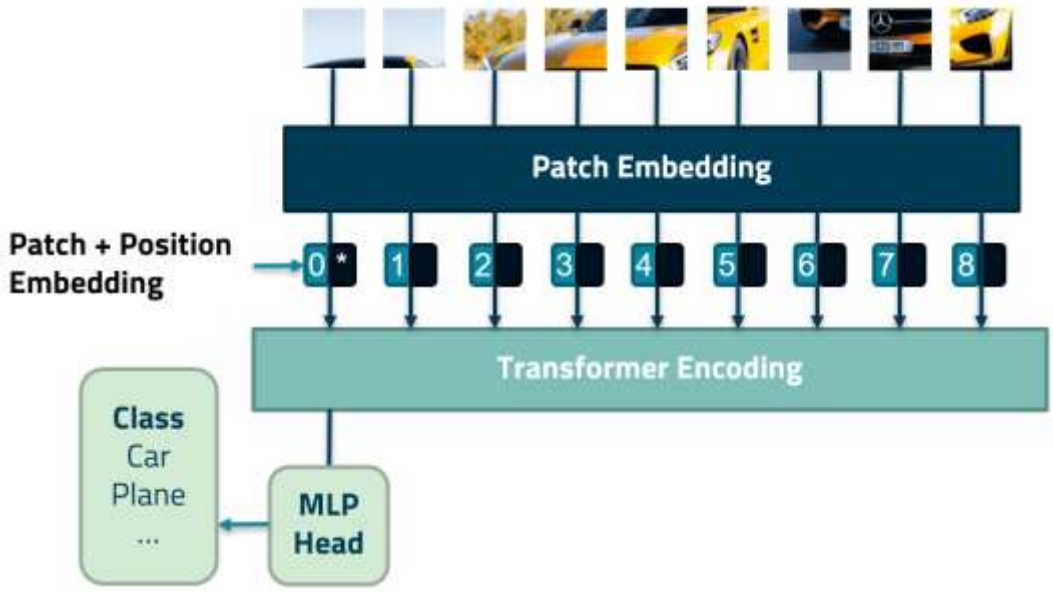


Figure 13. Vision Transformer (ViT) pipeline proposed by [43].

Another method of tackling the image captioning task is by utilizing early vision-language fusion. Zhou *et al.* [44] proposed a unified Vision Language Pretraining (VLP) model, which utilizes a shared transformer for both encoding and decoding. Therefore, it can be fine-tuned for either vision-language understanding or generation. The VLP model can perform both bidirectional and sequence-to-sequence predictions after a pretraining phase over a huge dataset of image-text pairs.

2.3 Language Models

The second major stage of an image captioning process is generating the linguistic descriptions, which is done by language models. The task of these language models is to predict the probability of a sequence of words occurring in a sentence. The generation process stops when the language model outputs a special end-of-sequence (EOS) token. Language models can be categorized into the following four groups based on their approaches: 1) LSTM-based. 2) CNN-based. 3) Transformer-based. 4) Vision-language early fusion.

2.3.1 LSTM-based Language Models

Recurrent Neural Networks (RNNs) have always been a popular choice for the generation of sentences in image captioning because of their ability to deal with sequential data. One of the most dominant variants of these networks is the Long Short-Term Memory or LSTM.

Vinyals *et al.* [13] utilized a single-layer LSTM as their language model, in which a copy of the LSTM memory is created for each word in the generated sentence. The output of the visual encoder is fed to the LSTM as the initial hidden state, which leads to the prediction of the next word at each time-step by applying a Softmax activation function. This prediction is also based on the previous words in the ground-truth caption during training time and previously generated words in the testing time. Figure 14 is an illustration of the general architecture of a single-layer LSTM-based language model.

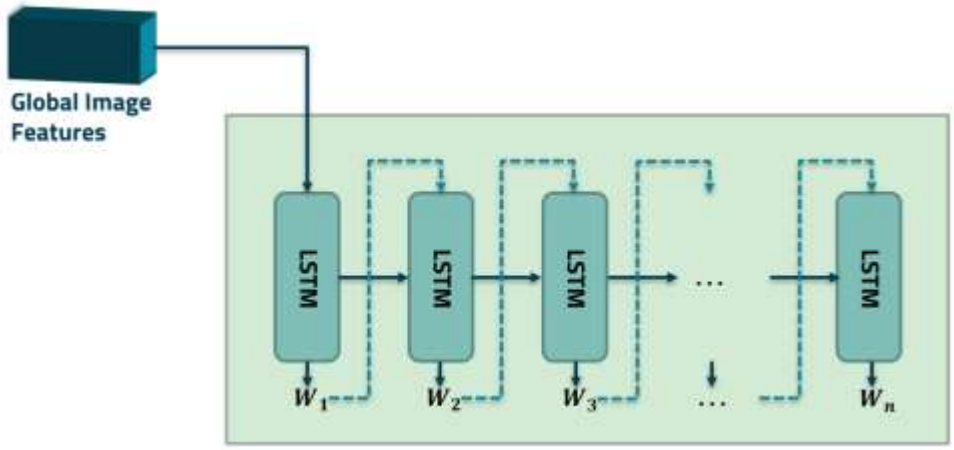


Figure 14. Single-layer LSTM-based language model. (W_i represents the i^{th} generated word in the sequence)

Xu *et al.* [23] improved the model introduced by [13] via utilizing additive attention. The proposed model weights the importance of features extracted from different locations of the input picture. This weighting process is achieved by assigning a coefficient α calculated by the attention mechanism. Therefore, the language model receives the weighted feature vectors (context vectors) to predict the next word. As it is shown in Figure 15, the LSTM layer receives the previous hidden state (h_{t-1}), the previous generated word (w_{t-1}), and the weighted feature vector from the attention module. The MLP head receives the current hidden state, the weighted feature vectors, and the output of the LSTM layer to generate the next word.

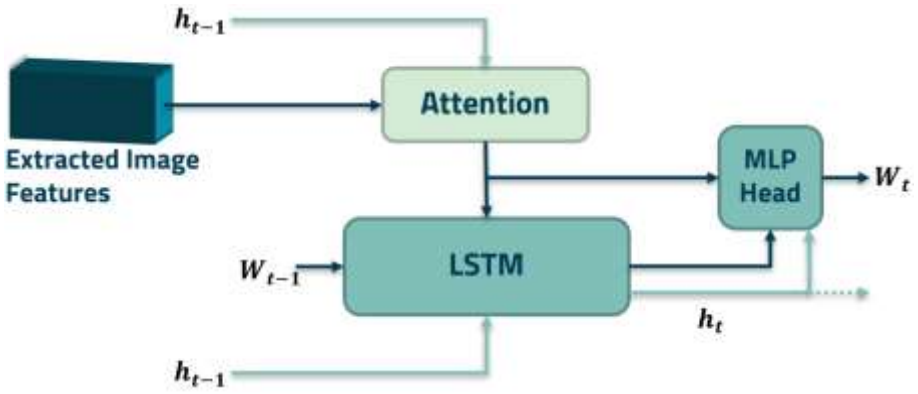


Figure 15. Single-layer LSTM with attention.

The previous studies have shown that utilizing multiple layers of deep networks can lead to improvements in their performances. Donahue *et al.* [45] proposed Long-term Recurrent Convolutional Network (LRCN), which uses two layers of LSTM networks in the language model. At each time-step, the bottom-most LSTM receives the word embedding from the previous time-step, which are encoded as one-hot vectors. It also receives the output of the CNN and passes the result to the second LSTM layer, which generates the words one by one at each time-step. This process is illustrated in Figure 16.

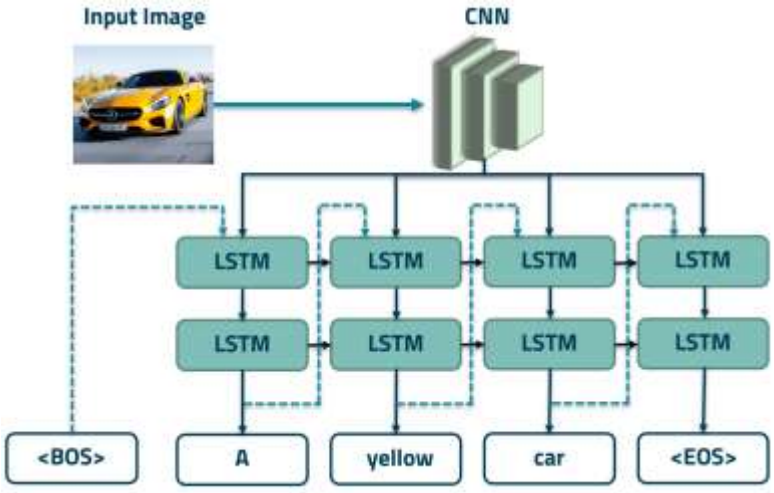


Figure 16. Two-layer LSTM language model. (<BOS> is the beginning-of-sentence token, and <EOS> is the end-of-sentence token.)

Anderson *et al.* [28] proposed a bottom-up and top-down image captioning model, where the language generation is done by a two-layer LSTM structure. The first LSTM layer is characterized as a visual attention model to attend to spatial image features, and the second LSTM layer is considered the language model. The input to the first LSTM layer is the previously generated word, the previous hidden state, and the mean-pooled image features. On the other hand, the second LSTM layer's input is the output of the attention LSTM, concatenated with the attended image features.

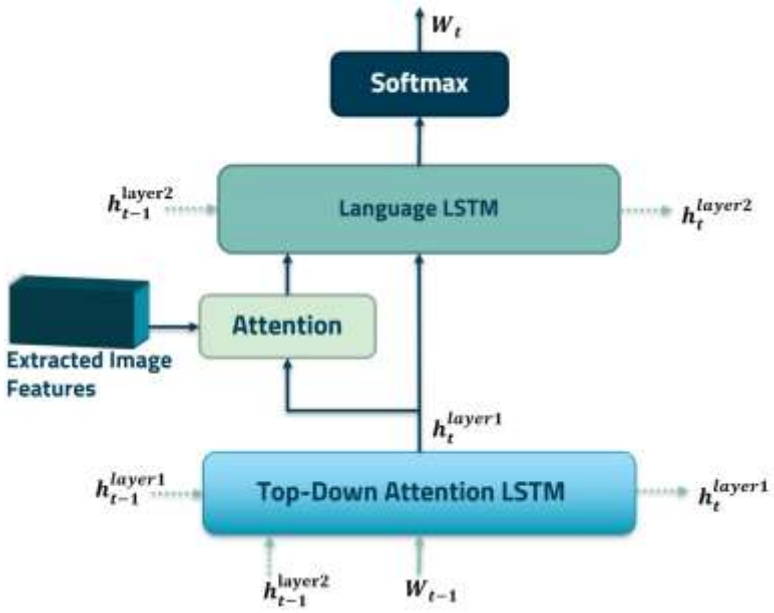


Figure 17. Two-layer LSTM language model with attention.

The general architecture of two-layer LSTM language models with attention is illustrated in Figure 17, where the first LSTM layer receives the previously generated word (w_{t-1}), the previous hidden states from the first LSTM layer (h_{t-1}^{layer1}), and the previous hidden state from the second LSTM layer (h_{t-1}^{layer2}). Next, the second LSTM layer receives the attention result along with the current hidden state to generate the next word.

2.3.2 Transformer-based and BERT-like Language Models

The image captioning task can be considered as a sequence-to-sequence process, where a sequence of image features are utilized to generate a sequence of textual linguistic descriptions.

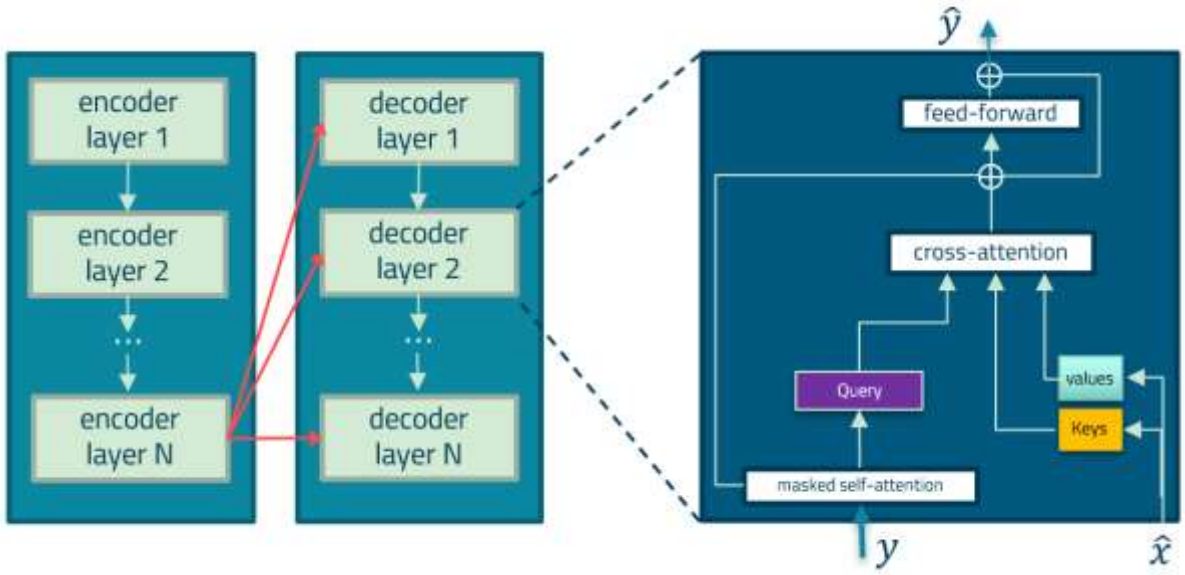


Figure 18. Transformer-based architecture for language models.

Vaswani *et al.* [46] proposed a transformer-based fully-attentive paradigm for the language generation task, which was adopted by numerous studies later on. The general pipeline of the transformer decoder which generates the textual description includes 1) a masked self-attention process. 2) a cross-attention mechanism where the words act as queries, while keys and values are the outputs of the last encoder layer. 3) a feed-forward network.

In addition, BERT-like architectures are also gaining popularity in image captioning tasks, especially in studies that utilize pretraining techniques. An early fusion occurs between the textual and visual modalities in these models, which are also capable of leveraging pre-trained parameters

in their initialization stage. These parameters are calculated based on a pretraining process over huge textual corpora.

2.4 Image Captioning Variants

Image captioning techniques can be categorized based on the type of captions they generate, which can be either factual or stylized. Factual image captioning is focused on generating purely objective content-based descriptions. Over time, these descriptions have gotten increasingly accurate, but they lack personality, emotion, and other human-like attributes. However, since factual captions are more directly reporting the contents of a visual input, they are more suitable for medical and assistive technologies.

On the other hand, stylized image captioning is an emerging topic utilizing linguistic techniques to convey specific feelings to the person reading the generated caption. These styles can be categorized based on both their linguistic style (e.g., humorous or romantic) and their sentiment (e.g., positive or negative). The main goal is to generate more appealing captions by incorporating human-like attributes in the generated text.

The absence of a large-scale dataset containing stylized (image, text) pairs has also led to adopting semi-supervised approaches, usually exploiting an unpaired stylized corpus to extract different styles [47], [48], [49], [50]. Gan *et al.* [47] proposed StyleNet framework, in which the target style for generated captions can be controlled utilizing a proposed module called "factored LSTM." This module factors parameters of the traditional LSTM (W_x) into three matrices (U_x, S_x, V_x) to memorize the linguistic patterns. It can be defined as follows:

$$W_x = U_x S_x V_x,$$

where U_x and V_x are learnable matrices intended to model the factual style in the textual data, while S_x is a learnable matrix to model the different style-specific factors. The StyleNet framework incorporates both a dataset containing factual (image, text) pairs, and a stylized monolingual corpus.

Mathews *et al.* [48] proposed SemStyle, which similarly generates stylized captions utilizing a paired dataset along with an unpaired stylized corpus with the core idea of separating style from semantics. Their proposed language model chooses the generated words from different vocabularies depending on the target style. The language model first utilizes the term generator to translate the visual input into appropriate semantic term representations; then the final stylized caption is generated using the language generator.

Factual: a man riding skis down a snow covered slope.

Romance: a man in a black jacket is jumping over a snow covered mountain to experience the thrill of life.

Humor: a lonely skier goes down an snowy hill thinking of cute lady skiers.

Positive: a amazing people stand on his skis on a snowy hill.

Negative: a man stands on his broken skis in the dirty snow.

Factual: two giraffes are standing outdoors near a building.

Romance: two giraffes are walking through the filed, exploring the woods.

Humor: two hungry giraffes standing in the filed looking for things to eat.

Positive: two giraffes in a pleasant park are against beautiful trees.

Negative: two giraffes are against a broken tree and dead grass.

Figure 19. Stylized vs. Factual captions generated by MSCap model [49].

Guo *et al.* [49] proposed Multi-Style Image Captioning (MSCap), which follows an adversarial learning-based paradigm. First, the captions are generated by a style-dependent caption generator, and then a caption discriminator is used to distinguish whether the generated caption is real or not. These two components are employed in an adversarial manner to increase accuracy and generate more human-like captions. Next, a classifier is utilized to determine the style class of the caption. Moreover, a back-translation module is incorporated to guarantee the consistency of the generated captions with the corresponding visual inputs. Figure 19 shows the MSCap's generated captions for two input images, where the captions describe each image in the following styles: factual, romance, humor, positive, and negative.

2.5 Image Captioning Datasets

There are several image captioning datasets publicly available, and each one offers different unique properties. Some of these datasets are mentioned below:

- **Microsoft Common Objects in Context** or **MS COCO** [51] is a popular large-scale dataset consisting of 328K images, 91 object classes, and 5 captions corresponding to each image. The MS COCO dataset contains annotations useful for various tasks, such as object detection, image captioning, and key-points detection.

- **Visual Genome** [33] is a unique dataset including different captions corresponding to different regions of the same image. It contains over 101K images from the MS COCO dataset, accompanied by information about attributes, relationships, scene graphs, question-answer pairs, and region descriptions.

- **Google's Conceptual Captions** [52] contains approximately 3.3 million images paired with natural language captions. These image-text pairs include a wider variety of styles since they have been directly collected from the internet. To be specific, the captions are collected from the alt-text HTML attribute associated with the images, and they have been utilized after a filtering process. The pipeline is illustrated in Figure 20, which consists of the following stages: image filtering, text filtering, image/text filtering, and text transformation. In the image filtering process, the images are filtered based on their size, aspect ratio, inappropriate content, and encoding format. In the text filtering stage, the alt-text is filtered according to the analysis of sentiment and part-of-speech (POS) annotations. The next step is image/text filtering, in which the consistency between the text and the image is examined and evaluated. Finally, the final caption is generated based on the original alt-text in the text transformation process.

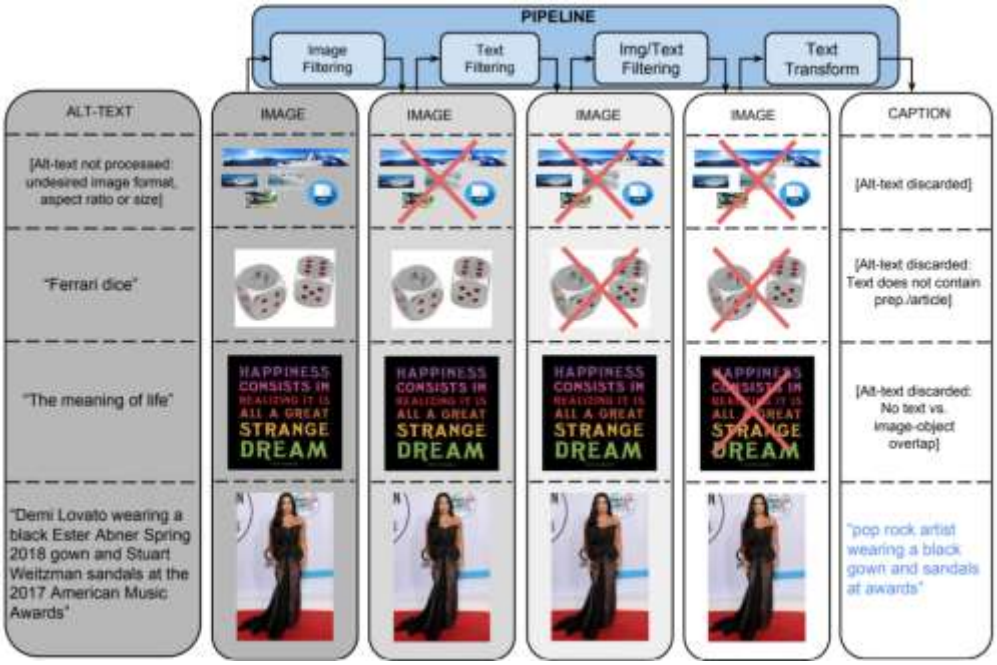


Figure 20. The pipeline used for collecting Google's Conceptual Captions dataset by Sharma *et al.* [52].

- **Flickr30K** [53] consists of 31K images collected from Flickr and 5 ground-truth captions written by human annotators for each image.

- **FlickrStyle10K** [47] has been collected based on the Flickr30K dataset, while, unlike the original dataset, it contains stylized captions for its 10K images. The styles include humorous, romantic, poetic, and neutral (factual).

2.6 Performance Comparison

In this section, the performances of the mentioned studies are compared based on various standard evaluation metrics in the image captioning field. Table 1 includes the results of the non-stylized image captioning models with respect to the BLEU [54], METEOR [55], ROUGE [56], and CIDEr [57] metrics. These results are obtained by performing the evaluation on the MS COCO [51]

dataset. On the other hand, Table 2 contains the results of the stylized image captioning models with respect to the BLEU, METEOR, and CIDEr metrics. The utilized dataset to evaluate these studies is the FlickrStyle10K [47]. These results are also categorized based on the following: 1) whether the training process is a multi-style paradigm or not, in which a single model learns to generate the captions in various styles. 2) The styles of the generated captions. (i.e., positive, negative, humorous, and romantic)

It is worth mentioning that for all these metrics, the higher values indicate better performance. In addition, these metrics are discussed elaborately in chapter 6.

Table 1: Results of the non-stylized image captioning models.

	BLEU-1	BLEU-4	METEOR	ROUGE-L	CIDEr
Vinyals *et al.* [13]	71.3	30.9	25.4	53.0	94.3
Karpathy *et al.* [14]	62.5	23.0	19.5	-	66.0
Mao *et al.* [16]	68.5	27.9	22.9	50.4	81.9
You *et al.* [19]	73.1	31.6	25.0	53.5	94.2
Wu *et al.* [20]	73.0	31.0	25.0	53.0	92.0
Chen *et al.* [21]	74.4	33.8	26.2	-	94.0
Rennie *et al.* [22]	78.0	35.3	27.1	56.7	117.4
Xu *et al.* [23]	70.5	27.7	24.1	51.6	86.5
Li *et al.* [24]	74.6	33.5	26.4	55.0	103.7
Chen *et al.* [26]	71.2	30.2	24.4	52.4	91.2

Table 1: (continued).

Jiang *et al.* [27]	76.4	37.0	27.4	-	112.5
Anderson *et al.* [28]	79.4	36.7	27.9	57.6	122.7
Zha *et al.* [30]	80.1	37.9	28.1	58.2	121.6
Guo *et al.* [31]	79.9	37.4	28.2	57.9	123.1
Herdade *et al.* [38]	80.5	38.6	28.7	58.4	128.3
Guo *et al.* [39]	80.8	38.8	29.0	58.7	126.3
Huang *et al.* [40]	80.2	38.9	29.2	58.8	129.8
Pan *et al.* [41]	80.8	39.5	29.5	59.2	132.0
Cornia *et al.* [42]	80.8	39.1	29.2	58.6	131.2
Zhou *et al.* [44]	80.9	39.5	29.3	59.6	129.3
Donahue *et al.* [45]	71.8	30.6	24.7	52.8	92.1
Yang *et al.* [34]	81.0	39.0	28.4	58.9	129.1
Shi *et al.* [35]	80.8	38.9	28.8	58.7	129.6
Yao *et al.* [36]	81.6	39.3	28.8	59.0	127.9
Li *et al.* [37]	81.2	38.9	28.6	58.6	122.1

Table 2: Results of the stylized image captioning models.

	Multi-style	Style	BLEU-1	BLEU-3	METEOR	CIDEr
Gan *et al.* [47]	No	Pos	45.3	12.1	12.1	36.3
		Neg	43.7	10.6	10.9	36.6
		Roman	13.3	1.5	4.5	7.2
		Humor	13.4	0.9	4.3	11.3
Guo *et al.* [49]	Yes	Pos	46.9	16.2	16.8	55.3
		Neg	45.5	15.4	16.2	51.6
		Roman	17.0	2.0	5.4	10.1
		Humor	16.3	1.9	5.3	15.2
Zhao *et al.* [50]	Yes	Pos	51.1	17.0	16.6	52.8
		Neg	49.2	18.1	15.7	59.4
		Roman	19.7	4.0	7.7	19.7
		Humor	19.8	4.0	7.2	18.5

2.7 Motivation

As it is inferable based on the literature review, most of the previous studies are focused on generating purely objective content-based descriptions. Their proposed models tend to describe the visual input in a factual approach and a neutral tone. On the other hand, more recently, some studies have focused on generating stylized captions to improve linguistic appeal. These studies aim to make the generated captions more attractive and diverse. However, these models still describe a visual input in an objective and content-based paradigm, with a difference in the generated descriptions' wordings.

This thesis is focused on an emotional-based approach to the image captioning task. We aim to develop a model capable of generating emotion-centric utterances based on various emotions, such as anger, sadness, contentment, and excitement. Another goal is to incorporate supervisory emotional signals by utilizing an emotional encoder in addition to the visual encoder and the language model in our image captioning pipeline. This causes a deeper connection between the reader and the generated captions on an emotional level. The ultimate goal is to propose a neural speaker capable of describing images based on the feelings conveyed by the shapes, colors, and other visual characteristics and not solely based on the objects in an input image.

Chapter 3
Methods

3 Methods

In this chapter, we provide a definition of the different methods and techniques that are utilized in this thesis. However, the exact role of these methods in the proposed model and the interactions between them will be discussed in the following chapters.

3.1 Mean Teacher Learning

Mean teacher learning is a semi-supervised paradigm based on the interaction between two models referred to as the teacher model and the student model. In the first place, Samuli *et al.* [58] proposed a novel architecture in which the temporal ensembling maintains an exponential moving average of the target predictions, while the inconsistent predictions get penalized by taking the mean squared error between the predictions of both models. However, temporal ensembling had a slow pace in utilizing the learned information in the training process since the targets are updated only once per epoch. Hence, it was not efficient when applied to large-scale datasets.

Tarvainen *et al.* [59] proposed mean teacher learning, in which the teacher model maintains an average of the student model's weights consecutively during the training steps instead of the label predictions. Figure 21 shows an illustration of the interaction between the two models in the mean teacher learning technique, where the teacher model utilizes the Exponential Moving Average (EMA) of weights of the student model. The important part of the method is the consistency cost between these two models, which is defined formally as below:

$$J(\theta) = E_{x,\eta',\eta}[\|f(x,\theta',\eta') - f(x,\theta,\eta)\|^2],$$

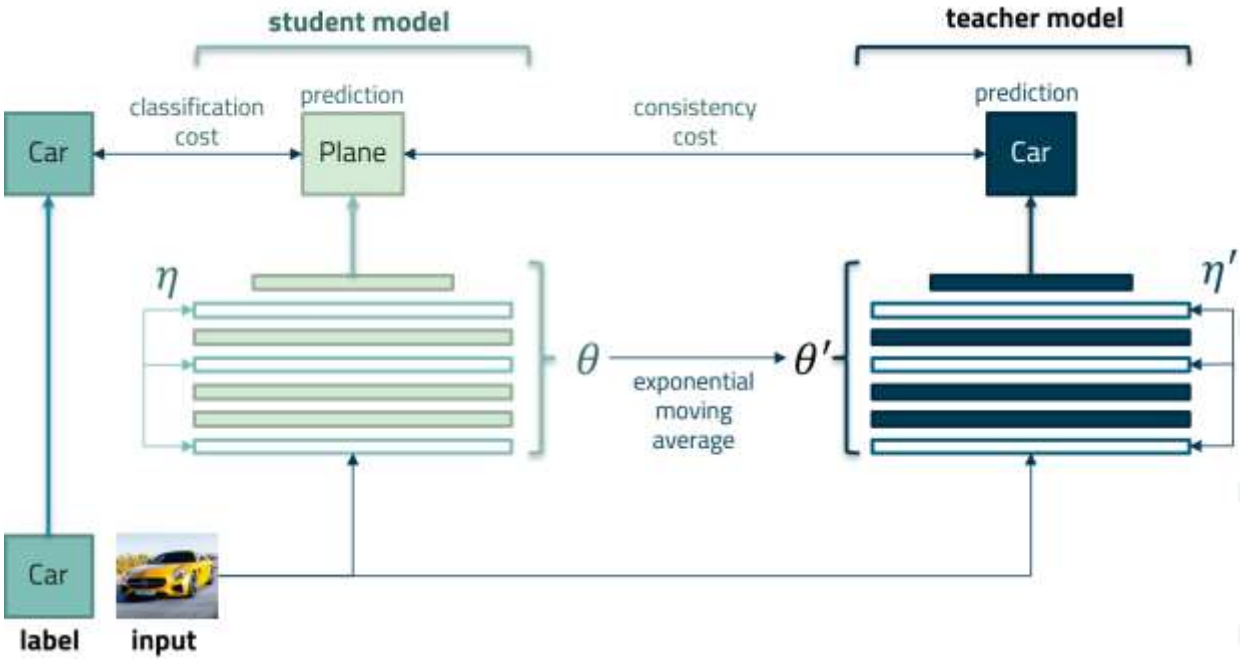


Figure 21. The interaction between the teacher model and the student model in mean teacher learning approach.

where $J(.)$ is the consistency cost function, which is equal to the expecting value for the difference between the predictions of the student model (with the parameters θ and noise η) and the predictions of the teacher model (with the parameters θ' and noise η'). On the other hand, the classification cost is calculated according to the student model's predicted class and the actual label, which can be based on a cross-entropy loss. At time-step t, the parameters θ' are updated based on the EMA of the successive parameters θ. This is formally defined as the equation below:

$$\theta'_t = \alpha\theta'_{t-1} + (1-\alpha)\theta_t,$$

where α is the weight decay indicating the intensity of the update.

3.2 Knowledge Distillation

The latent knowledge encapsulated within a larger network is often referred to as "dark knowledge" [60]. Knowledge Distillation (KD) [61] is the self-supervised process of transferring the dark knowledge from a bigger model to a smaller one, which has been shown that it can be utilized effectively in various vision-language tasks. The same process is called self-distillation when the models have equal sizes. In our case, the teacher model supplies an extra supervision signal to the student model to improve in imitating its behavior by providing predicted soft labels [62].

Neural networks usually pass the output logits l_i through the Softmax function, which computes a probability p_i each logit corresponding to its class. This process is done by comparing the l_i with other logits:

$$p_i = \frac{exp(l_i/\tau)}{\sum_j exp(l_j/\tau)},$$

where τ is the temperature of the Softmax function. The temperature value is usually set to 1, but higher temperatures can be utilized if softened probabilities are desired. The simplest form of distillation includes transferring the soft target distribution computed by the larger model to the distilled model, which is attained by being trained on a transfer set using a high temperature in its Softmax.

Based on the output logits of the distilled model l_i, and each case in the transfer set, the cross-entropy gradient $\partial C/\partial l_i$ can be computed following the equation below:

$$\frac{\partial C}{\partial l_i} = \frac{1}{\tau}(p_i - s_i) = \frac{1}{\tau}\left(\frac{e^{\frac{l_i}{\tau}}}{\sum_j e^{\frac{l_j}{\tau}}} - \frac{e^{\frac{g_i}{\tau}}}{\sum_j e^{\frac{g_j}{\tau}}}\right),$$

where g_i is the output logits of the larger model which lead to the corresponding soft target probabilities s_i. If the logits are of a lower magnitude compared to the temperature, this can be defined as:

$$\frac{\partial C}{\partial l_i} \approx \frac{1}{\tau}\left(\frac{1 + \frac{l_i}{\tau}}{N + \sum_j \frac{l_j}{\tau}} - \frac{1 + \frac{g_i}{\tau}}{N + \sum_j \frac{g_j}{\tau}}\right)$$

If we assume that the logits of each transfer case have been zero-mean, which mean that we have $\sum_j l_j = \sum_j g_j = 0$, the equation can be written as:

$$\frac{\partial C}{\partial l_i} \approx \frac{1}{N\tau^2}(l_i - g_i),$$

hence, the distillation is equivalent to minimizing $1/2(l_i - g_i)^2$ in the high temperature limit, assuming we have $\sum_j l_j = \sum_j g_j = 0$ for each transfer case. It has been shown that using moderate temperatures works best while using very small distilled models that are not able to capture all the knowledge, which indicates that ignoring large negative logits is helpful for the distillation process.

3.3 REINFORCE Algorithm

Reinforcement learning is an area of machine learning focused on the science of training intelligent agents to take out a series of actions with a particular goal in a potentially complex environment. The approach for artificial intelligence to learn the actions is to associate different acts with

rewards or penalties while the agent tries to maximize the reward. The method of maximizing the reward is based on trial and error while correcting the actions based on the corresponding rewards. The only human involvement in this process is limited to modifying the environment and the reward and penalty settings. Another action that can be carried out by humans is to prevent the agent from exploiting the rules. In general, we can summarize the main components of a reinforcement learning system as below:

- **Input:** The initial state where the agent begins searching for the solution from there.
- **Output:** The algorithm can have multiple outcomes since there may be different solutions for the same problem.
- **Training:** Feeding rewards and penalties to the agent during the process of solving the problem.
- **Policy:** The mapping between an agent's state at a given time to the action that should be carried out is called the policy. Basically, the policy indicates behaviour of the agent at a given time.
- **Reward:** At each time-step, a reward value is passed to the agent by the environment. This value is determined according to the action taken by the agent at that time-step, while the agent's objection is to maximize this reward value.
- **Value Function:** Unlike the reward value, which determines the fitness of an action in an immediate sense, the value function is more focused on the long-term consequences of an

action. Basically, the value function predicts the potential reward that can be achieved in the future, based on the current state of the agent.

One of the main challenges in reinforcement learning is the exploitation-exploration dilemma. In order to find the best solution, the agent should exploit the previously found good actions while exploring the environment to find better solutions.

Williams [63] proposed REINFORCE algorithm where in each training step, each network parameter w_{ij} gets incremented by the value defined below:

$$\Delta w_{ij} = \alpha_{ij}(r - b_{ij})e_{ij},$$

Where α_{ij} is the learning rate, r is the reinforcement value, b_{ij} is the reinforcement baseline, and e_{ij} refers to the characteristic eligibility which is defined as $e_{ij} = \partial \ln g_i / \partial w_{ij}$. The name of the REINFORCE algorithm comes from "**RE**ward **I**ncrement = **N**onnegative **F**actor × **O**ffset **R**einforcement × **C**haracteristic **E**ligibility," which matches the structure of the equation.

3.4 Self-Critical Sequence Training

Policy gradient-based reinforcement learning methods, such as REINFORCE [63], have been utilized in image captioning to overcome the mismatch and the exposure bias between the optimizing function and the non-differentiable evaluation metrics [64], [65], [66]. Self-Critical Sequence Training (SCST) [67] is a special case of REINFORCE in which the model's own test-time inference is used to normalize the rewards it experiences rather than estimating the reward and the normalization method.

Rennie *et al.* [67] proposed that directly optimizing the CIDEr metric [57] through the SCST process during the test time can be a highly effective way to overcome the non-differentiability of such metrics and boost the performance significantly. The recurrent network (LSTM) in this model is considered the "agent," while the multi-modal problem space containing words and images is considered the "environment." It is worth mentioning that the CIDEr metric is designed particularly to evaluate generated captions in image captioning. This metric is calculated by comparing the generated caption with a set of ground-truth human-written captions.

The action that is being carried out by the agent is predicting the next word, based on the policy p_θ defined by the network's parameters θ. A reward value is returned to the agent after reaching to the end-of-sequence (EOS) token, which in this case is the CIDEr score of the generated caption. The objective is to minimize the negative expected reward:

$$L(\theta) = -E_{w^i \sim p_\theta}\left[r\left(w^i\right)\right],$$

where $r(.)$ is the reward function, and $w^i = (w_1^i, w_2^i, \dots, w_T^i)$ is the word sampled from the model at time-step t. In addition, we can estimate $L(\theta)$ based on a single sample from p_θ:

$$L(\theta) \approx -r\left(w^i\right), \quad w^i \sim p_\theta.$$

To compute the gradient $\nabla_\theta L(\theta)$, the REINFORCE algorithm is utilized as follows:

$$\nabla_\theta L(\theta) = -E_{w^i \sim p_\theta}\left[r\left(w^i\right)\nabla_\theta \log p_\theta\left(w^i\right)\right].$$

We can estimate the expected gradient based on a single sample from p_θ:

$$\nabla_\theta L(\theta) \approx -r\left(w^i\right)\nabla_\theta \log p_\theta\left(w^i\right).$$

We can generalize the policy gradient based on a reference reward or a baseline. This baseline can be any arbitrary function; however, it should not be dependent on the action w^i. The generalized gradient is formally defined as below:

$$\nabla_\theta L(\theta) = -E_{w^i \sim p_\theta}\left[(r(w^i) - b)\nabla_\theta \log p_\theta(w^i)\right].$$

This gradient expression can be written in a different form as shown below:

$$\nabla_\theta L(\theta) = \sum_{t=1}^{T} \frac{\partial L(\theta)}{\partial s_t}\frac{\partial s_t}{\partial \theta},$$

where s_t represents the Softmax function's input. The following statement will be achieved by utilizing REINFORCE with a baseline b:

$$\frac{\partial L(\theta)}{\partial s_t} \approx \left(r(w^i) - b\right)\left(\mathrm{p}_\theta(w_t|h_t) - 1_{w_t^i}\right),$$

where w_t is the next generated word in the sequence, and h_t is the hidden state at time-step t.

As mentioned previously, in SCST the model's own test-time inference will be used as the baseline for reward values which results to the following expression:

$$\frac{\partial L(\theta)}{\partial s_t} = \left(r(w^i) - r(w^{tti})\right)\left(\mathrm{p}_\theta(w_t|h_t) - 1_{w_t^i}\right),$$

where $r(w^{tti})$ is the reward function based on the model's test-time inference, which leads to promoting the generated samples outperforming the current test-time system and suppressing the generated samples inferior to the current test-time system.

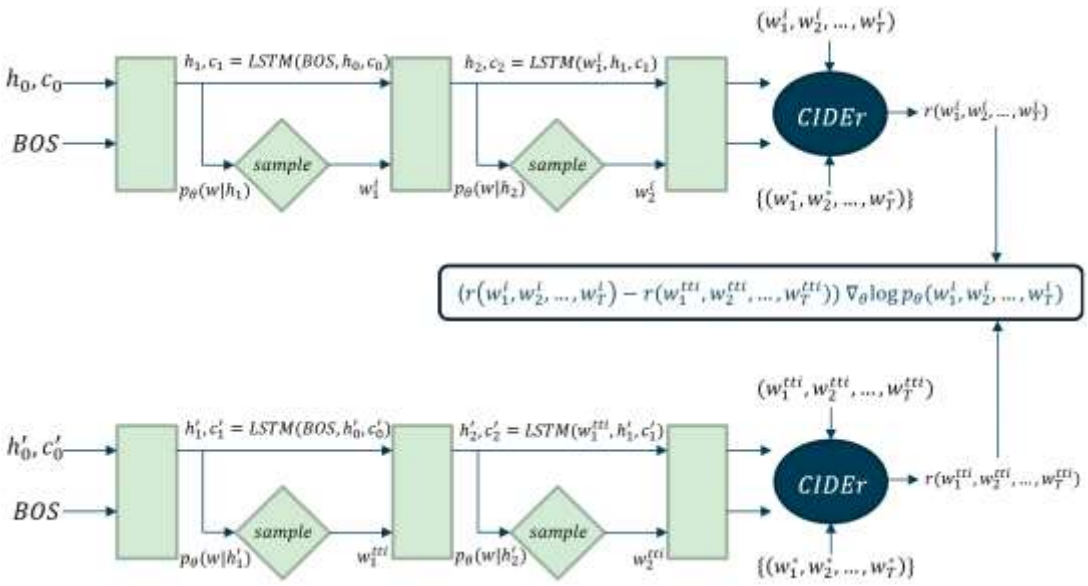


Figure 22. Self-Critical Sequence Training using the CIDEr metric.

3.5 Auxiliary Emotion Classification Tasks

Emotions in ArtEmis dataset [68] are divided into nine emotional classes; we have ***amusement***, ***awe***, ***contentment***, and ***excitement*** as positive emotions, while we have ***anger***, ***fear***, ***disgust***, and ***sadness*** as negative emotions. In addition, a ninth class named ***something else*** has been considered to express having no particular emotions or an additional feeling not listed. Following the work of [68], we employ two classifiers for our auxiliary emotion classification tasks, which will be utilized in both the captioning process and evaluation. We are basically facing a 9-way classification problem corresponding to each emotional class. The first module is a text-to-emotion classifier to predict the dominant emotional class of an utterance. This task is achieved by utilizing a fine-tuned pre-trained Bert model [69] to classify utterances to the emotional class to which they most likely belong. The second module is an image-to-emotion classifier to predict the dominant emotional class of a visual input. The module utilized by [68] is a ResNet-32 encoder pre-trained on the ImageNet [15] dataset.

Chapter 4
The Proposed Model

4 The Proposed Model

Neural Mean Teacher Learning-based Emotion-centric Speaker or Nemesis is the proposed neural speaker capable of leveraging emotional supervision signals in the caption generation process. In this chapter, we elaborate on the methodology, pipeline, architecture, and finally, the training strategies.

4.1 Methodology

The focus of this section is to provide an overview of the proposed model's workflow, starting from the model receiving an input image and ending with the generation of the caption's last token. Further details about each step are discussed in the following sections.

In the first step, the input image is passed to the visual encoder, which extracts the features and passes them to the language models. In the emotionally grounded version of the model, an encoding with respect to the dominant emotional class of the visual input is concatenated with the extracted visual features. The image-to-emotion classifier is responsible for detecting the dominant emotional class, while the emotional encoder obtains the corresponding emotional encoding.

In each language model, the first layer of the memory-augmented encoder receives the visual features and produces the key, value, and query matrices for the bi-directional attention process. In addition, the bi-directional attention process leverages additional memory slots, which are learnable vectors to encode the priori knowledge. The attention result is then passed to a feed-

forward network that gives us the final output of the memory-augmented encoder. Finally, this output is passed to both the next encoder layer and all the decoder layers via the meshed-like connectivity. The next encoder layer repeats the exact same steps.

On the other hand, the meshed decoder is responsible for generating the words one-by-one at each time-step. The meshed decoder receives the outputs of all encoder layers along with the input sequence of vectors. The meshed attention operator connects all the encoder outputs to the input sequence through gated cross-attentions. The query vectors for the cross-attention operation are obtained by applying a right-masked self-attention to the input sequence, while the key and value vectors are obtained according to the encoder outputs. Next, the output of the right-masked self-attention is concatenated with the cross-attention results. The result of the concatenation is then passed to a fully-connected network and a Softmax function to calculate a probability over words in the vocabulary. This process also repeats in each decoder layer to improve the generation of the next tokens and the understanding of the textual input. The caption generation process ends when the model predicts the end-of-sequence token. Figure 23 shows the flow diagram of the mentioned steps of the proposed model's workflow.

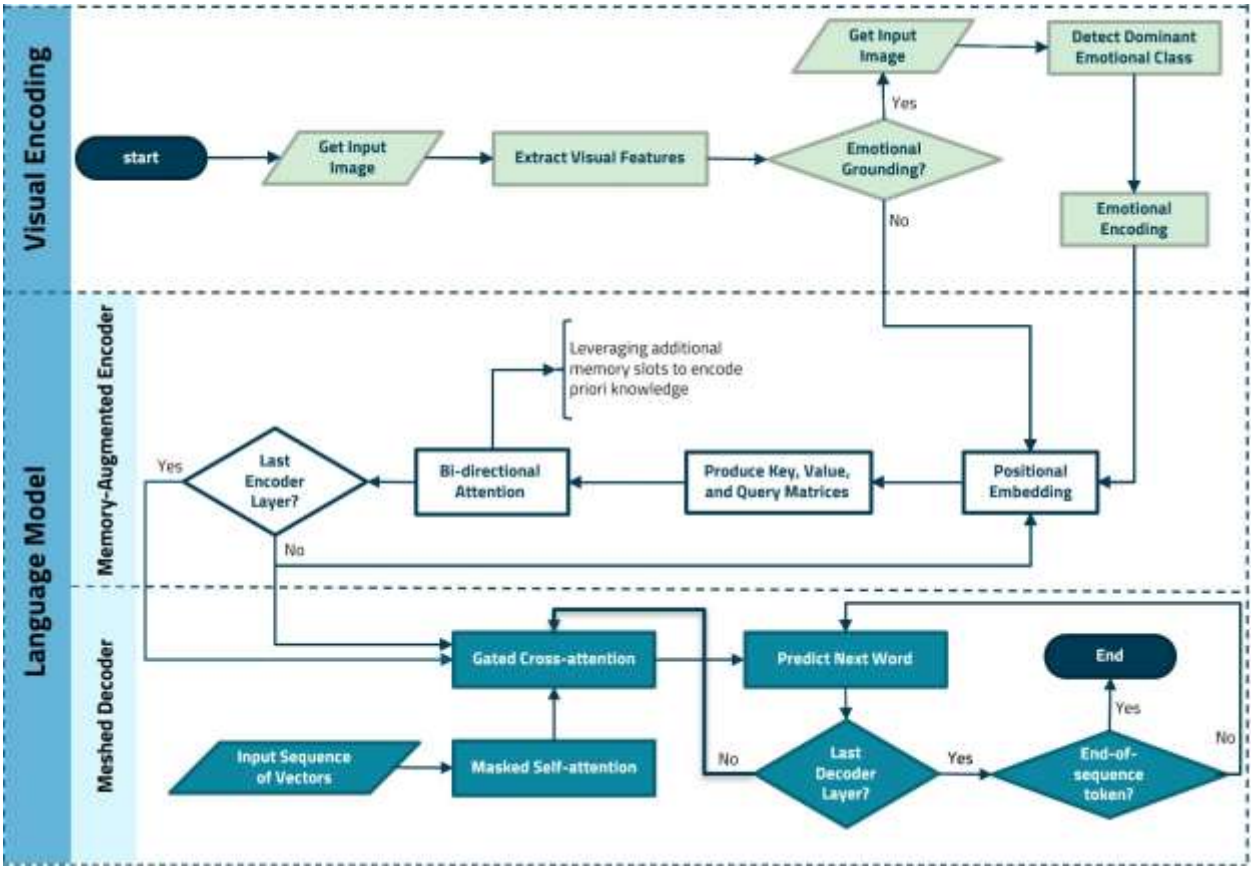


Figure 23. The flow diagram of the different steps of the proposed model's workflow.

4.2 Pipeline

The pipeline of Nemesis, as shown in Figure 24, consists of a visual encoder extracting visual features from the input image, then passing them to both the student model and the teacher model. Inspired by the work of Barraco *et al.* [70], both models have identical architectures linked based on two types of interactions: (1) the self-distillation process, where the teacher model provides regression targets via passing its predicted logits as soft labels to the equally sized student model [62]. This extra supervision signal enhances the ability of the student model to imitate the behavior of the teacher model. (2) The teacher model performs a form of model ensembling by updating its

parameters θ_t according to the exponential moving average (EMA) [71] of the student model's parameters θ_s. This updating procedure can be formally defined by the equation below:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s,$$

where λ is a value between [0, 1], indicating the intensity of this update. The exact role of these interactions in the training process is discussed in the training strategies (Section 4.4).

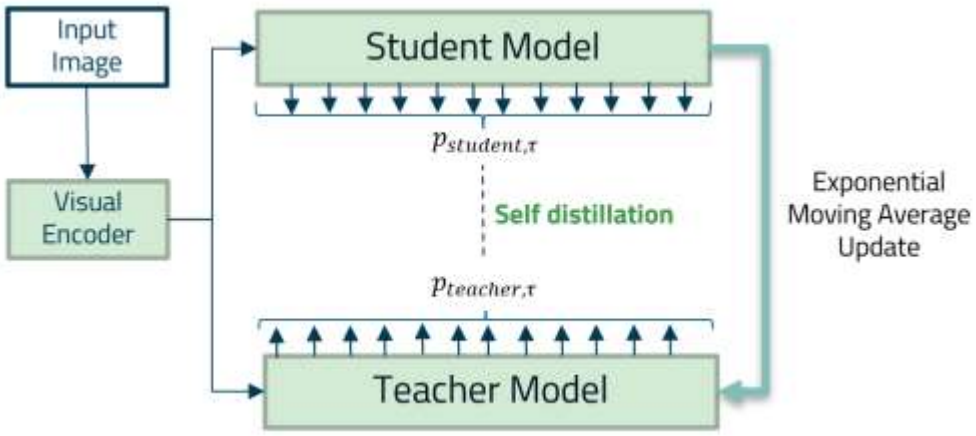


Figure 24. The interactions between our two language models: (1) the EMA update according to the student model's weights. (2) The self-distillation process using the teacher model's predicted logits passed to the student model, which will be treated as soft labels.

4.2.1 Visual Encoding

During the experimental studies of this research, multiple types of visual encoders have been employed in our proposed model. We will elaborate on each of these modules in the following:

- **Contrastive Language-Image Pre-Training** or **CLIP** [72] is a neural network trained on a huge dataset of 400M image-text pairs. This multi-modal pre-trained model has the "zero-shot" ability [73], [74], where the model classifies unseen data based on very few or no labeled examples.

We use the CLIP-RN50×16 pre-trained visual encoder introduced within the CLIP study, which is based on a ResNet-50 structure following EfficientNet-style [75] scaling. In EfficientNet-style scaling, the dimensions of depth, width, and resolution of a network are scaled by using a compound coefficient. In addition, the improvements proposed by [76] have been leveraged, alongside the utilization of blur pooling and anti-aliasing filters [77]. He *et al.* [76] proposed ResNet-D, which is a modification of the standard ResNet architecture, where 75% of the input feature maps are ignored by the downsampling block's 1×1 convolution. The downsampling block of the ResNet network is responsible for reducing the information in the case of deeper networks. Figure 25 illustrates the downsampling block of the ResNet-D architecture, where a 2×2 average pooling layer with a stride of 2 is added before the convolution, so no information gets ignored.

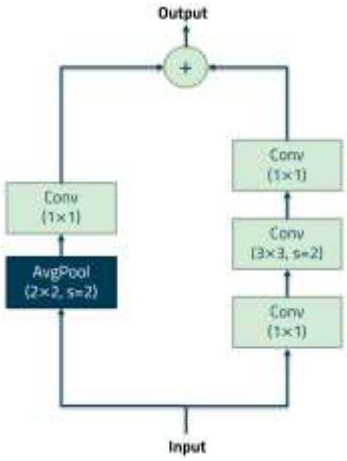


Figure 25. The downsampling block in the ResNet-D architecture.

On the other hand, the blur pooling process is done by replacing the standard average pooling filters with stronger filters to achieve a better shift-equivariance, while anti-aliasing filter selection allows for a choice of the blur kernel. In addition, the pooling

mechanism has been changed from utilizing global pooling layers to attention pooling layers.

The CLIP-RN50×16 has been pre-trained for 32 epochs with a batch size of 32768, using an Adam optimizer with a weight decay of 0.2, and a learning rate of 0.0004 with 2000 warm-up iterations.

- **Bootstrapping Language-Image Pre-training** or **BLIP** [78] is a pre-trained multi-modal model similar to CLIP. In our model, we have utilized the BLIP-ViT-L/16 variant of the BLIP visual encoders, which utilizes a framework based on the Vision Transformer [43]. Therefore, this visual encoder follows a purely transformer-based architecture dividing the input images into different patches. These patches and their positional encodings are embedded sequentially along with the CLS token to represent the visual input.

  The dataset utilized in BLIP is LAION400M [79], which has been gathered from the web through web scraping techniques. A novel method called Captioning and Filtering (CapFilt) has been employed to avoid low-quality and noisy data entries. Figure 26 illustrates the CapFilt process, where the captioner generates the descriptions for the collected images from the web, then the noisy captions are removed through the filtering process. Unlike CLIP, which was trained from scratch, the visual encoder in BLIP is initialized based on a ViT pre-trained on the ImageNet [15]. Then, the visual encoder is pre-trained for 20 epochs on the LAION400M dataset using a batch size of 2400, using the AdamW [80] optimizer with a weight decay of 0.05. The learning rate is warmed-up to 0.0002 and linearly decayed with a rate of 0.85.

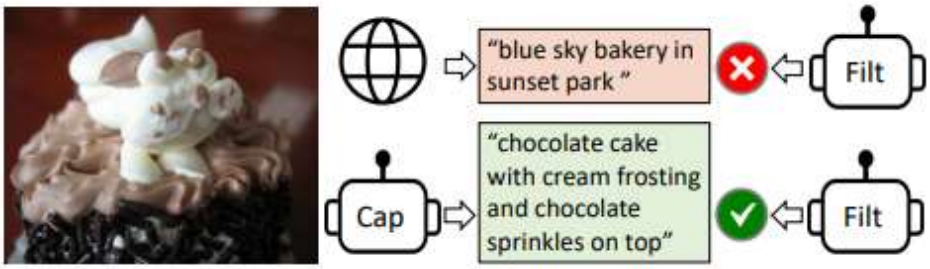


Figure 26. The CapFilter process in BLIP [78].

- **Faster RCNN pre-trained on Visual Genome** [81] is the third visual encoder used in this research, which is a ResNet101 network pre-trained on the Visual Genome [33] dataset. The pre-training has been done for 20 epochs with a batch size of 4, a learning rate of 0.001 with a decay of 5. Figure 27 shows the detected object-bounding-boxes by this visual encoder.

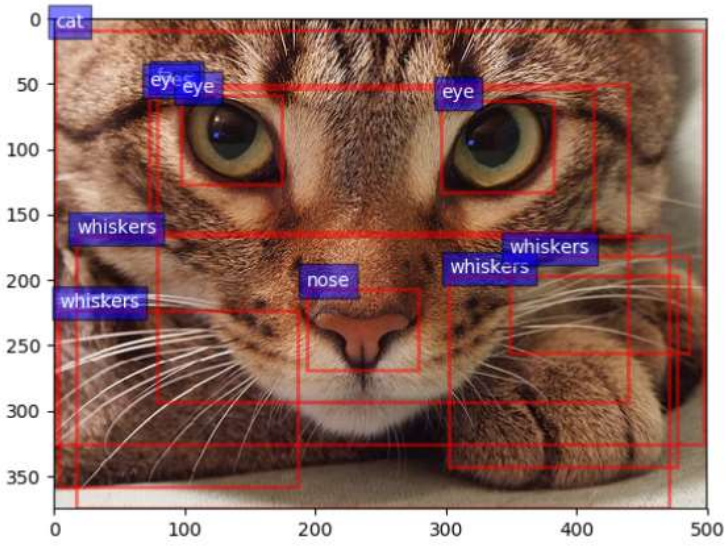


Figure 27. The object-bounding-box detection by [79].

4.2.2 Emotional Grounding

In this process, at each time-step, an additional feature according to the dominant emotional class of the input image is also passed to the language models alongside the extracted visual features. This extra emotional signal will enable the model to decouple the emotion conveyed during the caption generation process from the input image's dominant emotional class. This is inspired by how the human mind works while describing an emotional response, where we decide how to feel about something first, and then we put it into words. The dominant emotional class is selected utilizing the image-to-emotion classifier during evaluation. This process is illustrated in Figure 28. The neural speaker leveraging this supervision signal is called Emotionally Grounded Nemesis or EGNemesis.

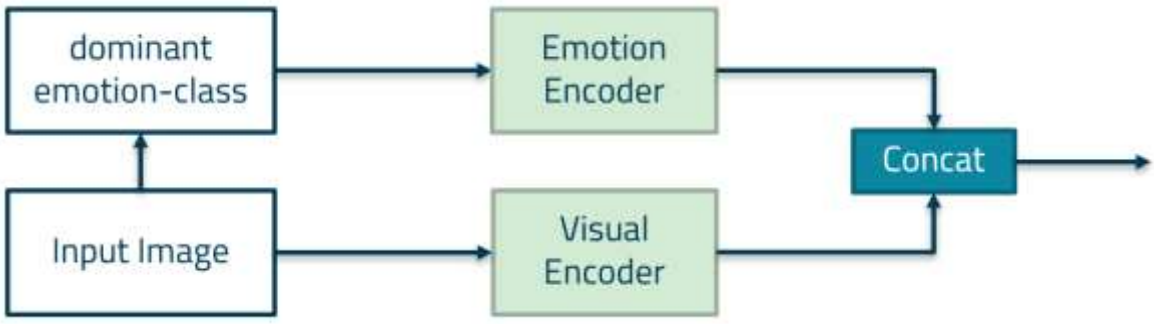


Figure 28. The emotional grounding process, in which the image-to-emotion classifier detects the dominant emotion-class, then a corresponding signal is concatenated to the visual encoding by the emotion encoder.

For the image-to-emotion classifier, a ResNet-50 classifier pre-trained on the Stylized-ImageNet dataset (SIN) [82] has been utilized. Since we are dealing with artworks and sketches as our visual inputs, the textures mostly differ from the ones in the real world. On the other hand, convolutional neural networks trained on the standard ImageNet [15] suffer from being biased towards a texture-based objection. For example, if there is a cat with elephant-like skin in the image, the model

classifies it as an elephant. Therefore, it was necessary to choose Stylized-ImageNet to overcome this bias. The local textures of images in Stylized-ImageNet have been distorted; however, the shapes are kept the same to increase the shape bias. Hence, we basically want our classifier to focus on shapes rather than textures. The style-transfer in Stylized-ImageNet has been achieved using the Adaptive Instance Normalization (AdaIN) [83] technique to change the original style to the style of a randomly selected painting.

Figure 29 shows an example style-transfer of an original ImageNet picture. The local textures are heavily distorted, but the general shapes have remained intact. Hence, this encourages the classifier to detect objects based on the shapes rather than the local textures.

It is worth mentioning that the image-to-emotion classification task is an auxiliary procedure, and the direct accuracy is not important. In chapter 6, we will see that our model performs better with this proposed classifier in comparison with utilizing the original image-to-emotion classifier utilized by [68].

Figure 29. The style-transfer of an original ImageNet picture in the Stylized-ImageNet dataset [82].

4.3 Architecture

Both language models follow the same architecture consisting of a stack of memory-augmented encoders and a stack of meshed decoders [42]. The architecture of both the teacher model and the student model is illustrated in Figure 30, which consists of a stack of memory-augmented encoders and a stack of meshed decoders. The memory-augmented encoder encodes the multi-level visual relationships leveraging the priori knowledge provided by the memory vectors. The meshed decoder generates the textual tokens leveraging the meshed connectivity illustrated by the red arrows in the figure.

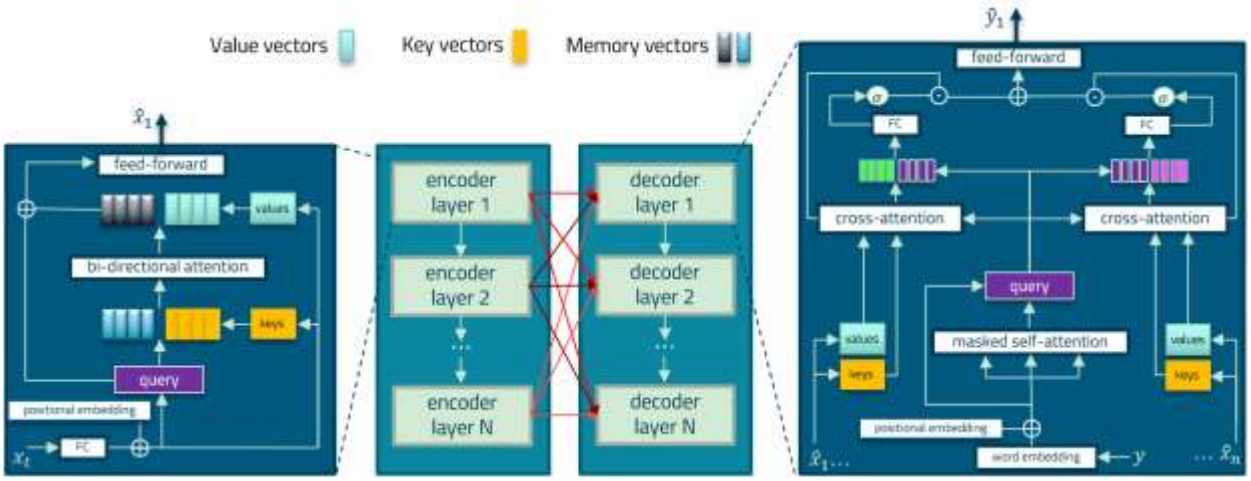


Figure 30. The architecture of both the teacher model and the student model.

4.3.1 Memory-augmented Encoder

Memory-augmented encoder utilizes bi-directional attention to process the visual features received from the visual encoder. However, using only bi-directional attention will deprive us of incorporating any priori knowledge in our encoding procedure. Hence, we utilize additional independent learnable memory vectors along with the key and value vectors to encode the additional priori knowledge. Finally, the encoder's output is the result of a feed-forward network applied to the memory-augmented bi-directional attention result. The outputs of all encoder layers are passed to each decoder layer via a meshed-like connectivity. The memory augmented attention is formally defined by the equation below:

$$MemAug_{att}(X) = Attention\big(W_q X, K_{MemAug}, V_{MemAug}\big),$$

$$K_{MemAug} = concat(W_k X, Mem_k),$$

$$V_{MemAug} = concat(W_v X, Mem_v),$$

where X is the input set, W_q, W_k, W_v are matrices of learnable weights, Mem_k and Mem_v are learnable memory matrices.

4.3.2 Meshed Decoder

Our decoder predicts the next word in an auto-regressive manner according to both the previously generated words and the encoder outputs. It applies right-masked self-attention to process the input sequence and utilizes cross-attention to process the encoder outputs received through the meshed-like connection. This meshed-like connectivity enables our model to extract both low-level and high-level features through a meshed cross-attention process. The cross-attention module uses queries based on the self-attention results and keys and values from the encoder outputs. Finally, the output of the position-wise feed-forward layer gives us the output logit at each time-step. The meshed cross-attention process is formally defined by the equation below:

$$Meshed_{Att}(\hat{X}, Y) = \sum_{i=1}^{N} \gamma_i \odot CrossAtt(\hat{X}_i, Y),$$

where $\hat{X}_i$ is the i^{th} encoder output, Y is the input sequence of vectors, $CrossAtt(.,.)$ is the encoder-decoder cross-attention function, and γ_i is a learnable matrix representing the contribution of the i^{th} encoder layer along with its relevance importance compared to other encoder layers. The cross-attention function is defined by the equation below:

$$CrossAtt(\hat{X}_i, Y) = Attention(W_q Y, W_k \hat{X}_i, W_v \hat{X}_i),$$

where W_q, W_k, W_v are matrices of learnable weights.

4.4 Training Strategies

The purpose of the training process in image captioning is to optimize the model to generate a proper linguistic description for an image. At each time-step, the model predicts the most probable word according to a learned distribution over the vocabulary, based on both the previously generated words and the visual input. Hence, the prediction is a unigram-based process, which is based on a 1 to n-gram input, where n is the number of previously generated words. One of the most popular approaches for this process is to maintain several sequence candidates and choose between them at the final step. This will avoid the issue of accumulated errors that can happen with maintaining a single sequence during the generation process.

Our training stage includes two phases: 1) cross-entropy (XE) training. 2) SCST fine-tuning.

4.4.1 Cross-entropy (XE) Training

In this phase, the student model faces two objectives. The first objective is to optimize the cross-entropy loss according to the previously generated utterances, the input image, and the model parameters. This process is formally defined by the equation below:

$$\mathcal{L}(\theta) = \mathbb{E}_{x\sim D} \sum_{\tau} \log(u_\tau | u_{k<\tau}, i, \theta),$$

where θ is the model parameters, u_τ is the generated utterance at time-step τ, and i is the input image. Cross-entropy loss optimizes the probability of each word in the ground-truth caption without considering the longer-range dependencies between the words in the generated sequence.

The second objective for the student model is to optimize the self-distillation loss with respect to the student model's parameters. The self-distillation loss is defined as the Mean Squared Error (MSE) between the output logits of both models. This process follows the expression below:

$$\min_{\theta_s} \sum_{\tau} \left(p_{t,\tau} - p_{s,\tau}\right)^2,$$

where $p_{t,\tau}$ and $p_{s,\tau}$ are the output logits over the vocabulary of N words at time-step τ for the teacher model and the student model, respectively.

The next step is the EMA update of the teacher model's parameters θ_t based on the student model's parameters θ_s. This process is done after each Stochastic Gradient Descent (SGD) update of the student model. It will enable the teacher model to keep up with the improvement of the student model in a stable and steady manner. Algorithm 1 is the pseudocode for the emotionally grounded version of the cross-entropy (XE) training loop, in which the emotional encodings and the visual encodings are concatenated in the teacher and student models, as depicted in Figure 28.

```
#lambda: the momentum of the teacher model's update
#I: the input image, E: the encoded emotional class, U: the utterance
#tm, sm: the teacher model and the student model

for I, U in dataloader:
      t = tm(I.concat(E), U)                    #teacher output
      s = sm(I.concat(E), U)                    #student output

      loss = XE(softmax(s, dim=-1), U) + MSE(t, s)
      loss.backward()                           #backpropagate

      update(sm)                                #Stochastic Gradient Descent
      tm.parameters = lambda * tm.parameters + (1 - lambda) * sm.parameters

def MSE(teacher, student):
      teacher = teacher.detach()                #stop gradient
      return (teacher - student).square().mean()
```

Algorithm 1. The cross-entropy (XE) training loop.

4.4.2 SCST Fine-tuning

This phase is a special case of REINFORCE [63], where the idea is to weight the samples outperforming the current test-time model positively and, in contrast, weight the samples inferior to the current test-time model negatively. This weighting process is done via the reward function utilized to assign a proper score to the generated utterances at each time-step. For this purpose, the CIDEr-D [57] metric has been used as the reward function, and we use the model's own test-time inference to normalize the rewards. Specifically, at each time-step, the top-1 utterance in each of the *k* returned beams is assigned with a proper reward, and the average of the rewards is used as a baseline to normalize them and reduce the variance.

This phase enables us to overcome the non-differentiability of such metrics and boost the performance significantly. The SCST-based fine-tuning process is formally defined by the expression below:

$$\nabla_\theta \mathcal{L}(\theta) = -\frac{1}{k}\sum_{j=1}^{k}((r(u^j) - ((\sum_j r(u^j))/k))\,\nabla_\theta \log p(u^j)),$$

where u^j is the j-th utterance in the beam, $r(.)$ is the reward function, $(\sum_j r(u^j))/k$ is the average of rewards to normalize the value.

The utilization of other metrics such as Emotional-Alignment [68] has been experimented as well. In this metric, the text-to-emotion classifier is utilized to predict the emotional class of the generated caption, then it is compared to the dominant emotional class of the ground-truth captions, and the percentage of matches is our score. However, the results were not acceptable due to the high variance of such metrics. It is worth mentioning that following the work of [68], a fine-tuned pre-trained Bert model [69] has been utilized as the text-to-emotion classifier.

Chapter 5
Experimental Settings

5 Experimental Settings

In this chapter, we will have a more experimental approach to discuss the proposed model in this research. We will discuss the dataset, utilized resources, implementation environment, tools, frameworks, and hyperparameters.

5.1 Dataset

The ArtEmis dataset [68] has been utilized for training and evaluating our proposed model. It contains 454,684 emotion-centric utterances related to 80,031 artworks publicly available in the WikiArt[1] dataset. These artworks include 27 styles of paintings created between the 15th and 21st centuries. These painting styles are Abstract Expressionism, Action Painting, Analytical Cubism, Art Nouveau Modern, Baroque, Color Field Painting, Contemporary Realism, Cubism, Early Renaissance, Expressionism, Fauvism, High Renaissance, Impressionism, Mannerism, Late Renaissance, Minimalism, Naïve Art Primitivism, New Realism, Northern Renaissance, Pointillism, Pop Art, Post Impressionism, Realism, Rococo, Romanticism, Symbolism, Synthetic Cubism, and Ukiyo-e. These styles include paintings with different levels of abstraction, colors, forms, shapes, values, and textures. Figure 31 shows an example from the ArtEmis dataset, which includes multiple emotional responses to the same artwork.

[1] https://www.wikiart.org

The reason for using artworks is that they are the best tools to trigger emotional responses. Through Amazon's Mechanical Turk (AMT) service, 6,377 annotators have been recruited to create a corpus of 36,347 distinct words, while the utterances include abstract concepts (e.g., love or freedom) and a wide variety of similes and metaphors. For each artwork, at least five annotators have expressed their emotional responses while explaining the reasons. Figure 32 shows an example AMT interface which has been utilized for the data collection, where the annotators first select the emotional response, then they explain the reason for choosing that specific emotion.

Figure 31. An example from the ArtEmis dataset containing multiple emotional responses to the same artwork. You can see the different descriptions along with their corresponding emotional class (in bold font).

Each utterance belongs to one of these nine emotional classes: ***amusement***, ***awe***, ***contentment***, and ***excitement*** as positive emotions, and ***anger***, ***fear***, ***disgust***, and ***sadness*** as negative emotions. In addition, a ninth class named ***something else*** has been considered to express having no particular emotions or an additional feeling not listed. Efforts have been made to include at least one negative

and one positive emotional response for each artwork. Partitions of 85%, 5%, and 10% have been considered for train, validation, and test splits, respectively.

Figure 32. AMT interface utilized for the data collection of ArtEmis [68].

5.2 The Digital Research Alliance of Canada

The Digital Research Alliance of Canada (the Alliance) is an organization responsible for the coordination of Canada's advanced research computing (ARC) platform. A total of six national clusters have been installed, which offer various computational services, resources, and software packages. These clusters are Arbutus, Béluga, Cedar, Graham, Narval, and Niagara.

In this research, the Graham cluster has been utilized by accessing the provided virtual machine through an SSH connection. Users can configure a cloud account and specify a computing environment according to their requirements. The cloud services also offer shared resources for the researchers to collaborate and share various contents. The Graham cluster offers a total volume of 133TB for the home space while offering a total volume of 3.2PB and 16PB for the scratch and project spaces, respectively. The home space is the location for the home directories, the scratch

space is used for active or temporary storage while the computational jobs are running, and the project space is a large adjustable storage dedicated per project.

5.2.1 GPU Types

The nodes in the Graham cluster offer a total of 520 GPU devices to provide the computational power required for various research purposes. This includes 320 NVIDIA P100 Pascal GPUs, 56 NVIDIA V100 Volta GPUs, 16 NVIDIA V100l Volta GPUs, and 144 NVIDIA T4 Turing GPUs. The P100 Pascal GPUs offer 12GB of HBM2 memory, and they are high-performance cards used for all purposes. The V100 Volta GPUs offer 16GB of HBM2 memory, and they provide twice the computational power compared to P100 GPUs for standard purposes, while they provide 8x computational power for deep learning computations by using the tensor core computation units. The V100l Volta GPUs are similar to the V100 GPUs with the difference of providing 32GB of HBM2 memory and also utilizing NVIDIA NVLink interconnection, increasing the scaling capabilities in high-performance computing (HPC). Finally, the NVIDIA T4 Turing GPUs offer 16GB of GDDR6 memory, and they are specifically designed to handle deep learning workloads. However, the T4 GPUs lack efficiency in double precision computations, which occupy 64 bits in memory instead of 32 bits. They also offer tensor core computation units; however, the V100 Volta GPUs are 2.2x-3.6x faster than the T4 Turing GPUs.

5.2.2 Horovod

The Digital Research Alliance of Canada also offers the option of utilizing Horovod to use multiple GPUs simultaneously. Horovod is a framework originally designed by Uber for distributed deep learning training. It helps reduce the training time by a matter of minutes and hours, or even days and weeks, based on how many GPUs are incorporated in the computation.

In Horovod, users can scale up the computation to run up to hundreds of GPUs by only writing a few lines of Python code. It is easy to use Horovod with different machine learning frameworks once configured, such as PyTorch, TensorFlow, and Keras. Horovod provides the same level of control as the DistributedDataParallel class in PyTorch effortlessly by abstracting away the configuration of process groups and handling environment variables of the cluster scheduler. In this research, Horovod has been utilized while training with the BLIP [77] visual encoder.

5.3 Visual Encoding Modes

The visual encoding process has been done in two different approaches based on the timing of the object-bounding-box detection. These modes are elaborated in the following:

- **Online Visual Encoding:** The object-bounding-box detection is done during the training phase in real-time. The CLIP-RN50×16 [72] and BLIP-ViT-L/16 visual encoders have been employed in this approach. In training with the CLIP visual encoder, a single NVIDIA V100 Volta GPU has been utilized, and each training epoch takes ~4 hours in the cross-entropy (XE) training stage and ~6 hours in the SCST fine-tuning phase. On the other hand, while training with the BLIP visual encoder, two NVIDIA V100l Volta GPUs were utilized via the Horovod framework, and each training epoch takes ~1 hour in the cross-entropy (XE) training stage. In the SCST fine-tuning, a single NVIDIA V100l Volta GPU has been utilized, where each epoch takes ~7 hours.

- **Offline Visual Encoding:** The object-bounding-box detection is done separately and before the training phase. The detections are stored in a .TSV document for further utilization during the training process. A Tab-Separated Values (TSV) file stores data in a table structure, where each separate line represents a record in the table, and the columns

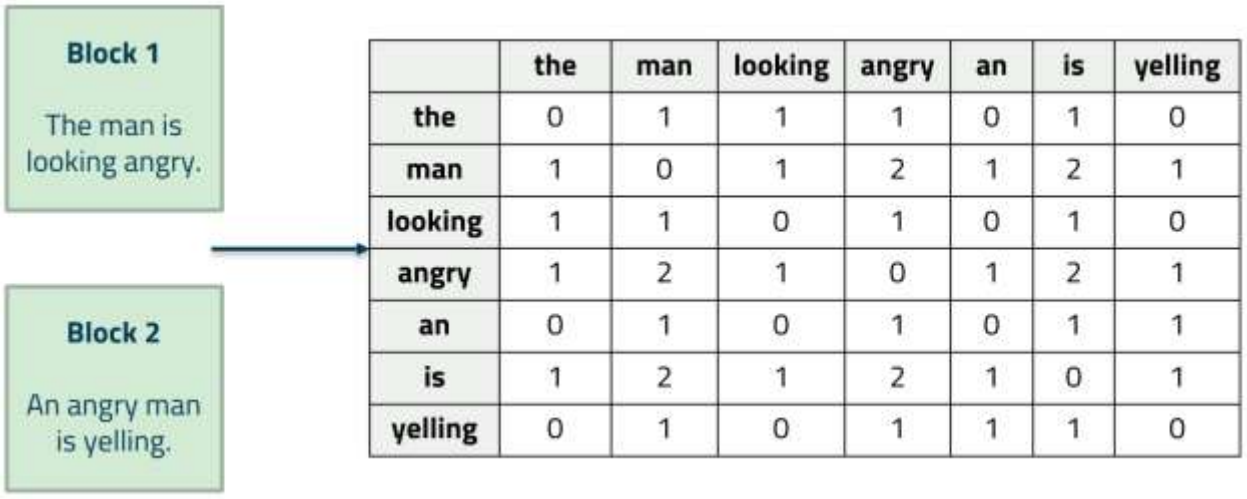
are separated by tabs. The “Faster RCNN pre-trained on Visual Genome” [81] visual encoder has been employed in this visual encoding mode. Each cross-entropy (XE) training epoch takes ~1 hour on a single NVIDIA P100 Pascal GPU. However, the SCST fine-tuning was not done in this mode because of the incompetent results.

5.4 Implementation Details

The utilized framework in this research is PyTorch [84], which is a framework based on Python and the Torch [85] library. It is open-source and free and can be utilized to train and develop neural networks in deep learning models. The back-propagation process in neural networks is handled by computation graphs, where the neural network’s nodes and their connections are represented by the computation graph’s nodes and edges. Unlike most other popular libraries like TensorFlow [86], PyTorch employs a dynamic computation approach, allowing it to build complex architectures more flexibly. In addition, most Python debugging tools can be integrated because of the dynamic graph computations at run-time. This framework provides various machine learning packages and is compatible with multiple Python-based libraries, such as NumPy and SciPy. It is worth mentioning that PyTorch is also programmable via C/C++ since it shares the backend with Torch. However, a Python-based implementation has been done in this research.

Furthermore, the following Python packages have been utilized: torch, torchvision, tensorboard, nltk, pandas, scipy, scikit-learn, plotly, Pillow, dask, traitlets, nbconvert, MarkupSafe, tifffile, decorator, networkx, ipython, ipykernel, jupyter, tqdm, seaborn, termcolor, scikit-image, PyWavelets, symspellpy, cycler, cython, fonttools, ftfy, kiwisolver, matplotlib, packaging, psutil, pycocotools, pyparsing, python-dateutil, wcwidth, numpy, timm, fairscale.

Regarding the hyperparameters, for the cross-entropy training and SCST fine-tuning stages, batch sizes of 50 and 30 have been considered, respectively. In the text-to-emotion classifier's training, GloVe word embeddings [87] have been utilized for word representation. Global Vectors for word representations or GloVe is a tool to capture the meaning, semantics, and context of different words. The word embeddings in GloVe are obtained based on the aggregation of the global word co-occurrence matrices from a given corpus. The co-occurrence matrix contains the frequency of specific pairs of words occurring together, as shown in Figure 33.



	the	man	looking	angry	an	is	yelling
the	0	1	1	1	0	1	0
man	1	0	1	2	1	2	1
looking	1	1	0	1	0	1	0
angry	1	2	1	0	1	2	1
an	0	1	0	1	0	1	1
is	1	2	1	2	1	0	1
yelling	0	1	0	1	1	1	0

Figure 33. Co-occurrence matrix in GloVe.

We use different statistic measurements to draw relationships between the words. Instead of using the co-occurrence probabilities directly, we use the ratios of co-occurrence probabilities as a starting point to get the word embeddings. GloVe's objective is to minimize the difference between the dot product of two word-vectors and the logarithm of their number of co-occurrences. This process is formally defined as below:

$$J = \sum_{i,j=1}^{V} W(X_{ij})\left(w_i^T \widehat{w}_j + b_i + \hat{b}_j - \log X_{ij}\right)^2$$

where w_i and b_i are the vector and bias for word i, $\widehat{w}_j$ and $\widehat{b}_j$ are the vector and bias for word j, X_{ij} is the number occurrences of word i in the context of word j, and J is the weighted least squared objective.

In the captioning model's training phase, Byte Pair Encoding (BPE) [88] method has been utilized to represent the words. The first step is to initialize the symbol vocabulary, where each word is represented by a sequence of characters ending with a special end-of-sequence symbol ".". The algorithm iteratively counts the occurrences of symbol pairs and replaces the most frequent symbol pair ("X," "Y") with a new symbol, "XY". This merging process produces an n-gram character, and the frequent n-gram characters will merge into a single symbol eventually. In general, the BPE algorithm ensures that the frequent words are represented as a single token in the vocabulary, while the rare words are represented as two or more subword tokens.

The sinusoidal positional encodings [46] have been employed to represent word positions. These positional encodings have the same dimension as the input embeddings; hence, they can get summed. The sine and cosine functions of different frequencies have been utilized:

$$PosEnc_{(pos,2d)} = \sin(pos/10000^{2d/d_{model}}),$$

$$PosEnc_{(pos,2d+1)} = \cos(pos/10000^{2d/d_{model}}),$$

where pos is the position, d is the positional encoding dimension, and d_{model} is the embedding dimension. Since for each arbitrary fixed offset k, $PosEnc_{pos+k}$ is a linear function of $PosEnc_{pos}$, the model can attend according to the relative positions.

Three layers of both encoders and decoders have been utilized, each with a dimensionality of 512. The emotional dimension is equal to 10, along with a feed-forward dimensionality of 2058 in EGNemesis and 2048 in Nemesis and a head-number of 8. The memory size has been set to 40. In addition, a dropout of 0.1 is applied to each sub-layer output. The position-wise feed-forward network applies identically to each position separately.

In a single attention function, the keys, queries, and values have a dimensionality of 512, while in multi-head attention, we use 8 different learned linear projections to d_{key}, d_{query}, and d_{value} dimensions. Then, the attention function is applied to each of these projections in parallel, which leads to d_{value}-dimensional output values. The concatenation of these output values will give us the final result of the multi-head attention. This process which is also illustrated in Figure 34, follows the below equation:

$$MutliHeadAtt(Q,K,V) = Concat\left(head^{(1)}, head^{(2)}, \dots, head^{(n)}\right)W_O,$$

$$head^{(i)} = Attention\left(QW_Q^i, KW_K^i, VW_V^i\right),$$

where $W_Q^i \in \mathbb{R}^{d_{model} \times d_{key}}$, $W_K^i \in \mathbb{R}^{d_{model} \times d_{key}}$, $W_V^i \in \mathbb{R}^{d_{model} \times d_{value}}$, and $W_O \in \mathbb{R}^{nd_{value} \times d_{model}}$ are the projection matrices. Since we are using 8 attention heads, the dimensions are $d_{value} = d_{key} = d_{model}/8 = 64$.

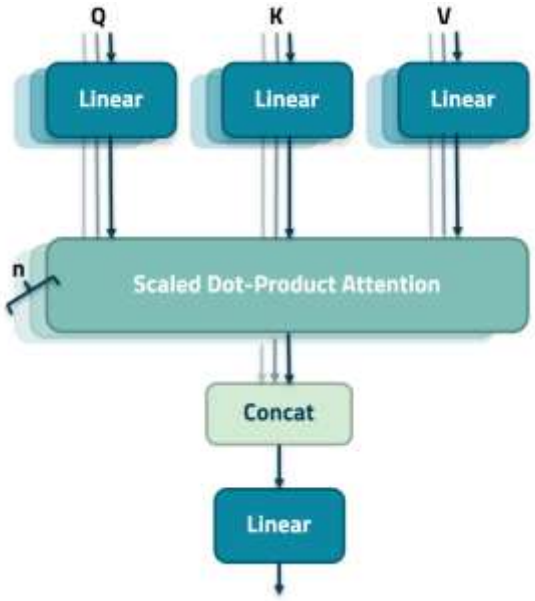


Figure 34. Multi-head attention.

Adam [89] optimizer has been employed in all experiments along with a beam size of 5. In the cross-entropy training, the typical transformer learning rate scheduling strategy [46] has been utilized with a 10,000 iteration warmup. While in the SCST fine-tuning phase, a fixed learning rate of 5×10^{-6} has been considered, with a momentum λ of 0.999 for the teacher model.

Chapter 6
Evaluation and Results

6 Evaluation and Results

The evaluation is one of the most important steps of research since it allows us to measure the performance of the proposed model. This chapter will discuss the evaluation metrics, ablation studies, and a comparison with the state-of-the-art.

6.1 Evaluation Metrics

In this research, we employ the following evaluation metrics: BLEU [54], METEOR [55], ROUGE [56], CIDEr [57], Emotional-Alignment [68]. These metrics are elaborated on in this section.

6.1.1 BLEU

The BiLingual Evaluation Understudy or BLEU metric [54] is an algorithm to evaluate the quality of a machine-generated text by comparing it to a set of ground-truth reference texts. The result is a value between 0 and 1 depending on the similarity, with values closer to a score of 1 indicating more similarity. However, some essential factors such as grammatic correctness and intelligibility are not considered in this metric. In addition, the BLEU metric is biased towards assigning higher scores to shorter texts.

If we have a set of candidate captions $Candidate = \{y_1, y_2, \dots, y_n\}$ and a set of ground-truth reference captions $GroundTruth = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n\}$, for each candidate string $y = \{y_1, \dots, y_k\}$ we define a set of n-grams as follows:

$$NGrams^{(n)}(y) = \{y_1, \dots, y_n, y_2, \dots, y_{n+1}, \dots, y_{k-n+1}, \dots, y_k\}$$

On the other hand, we define a substring count function $Count(s, y)$ to calculate the number of times the string s appears as a substring of y.

Finally, we define the n-gram precision function as follows:

$$Precision^{(n)}(\hat{y}; y) = \frac{\sum_{s \in NGrams^{(n)}(\hat{y})} \min(Count(s, \hat{y}), Count(s, y))}{\sum_{s \in NGrams^{(n)}(\hat{y})} Count(s, \hat{y})}$$

The result of this function will give us the BLEU score. BLEU-n assigns a single weight to the computed n-grams of one to n, for example, BLEU-2 considers 1-grams and 2-grams associated with a weight of 0.5, while the final result will be the geometric mean of these values.

6.1.2 METEOR

The Metric for Evaluation of Translation with Explicit ORdering or METEOR metric [55] is based on unigram matching between the machine-generated and ground-truth reference texts. This matching process can consider stemmed forms and synonyms, in addition to the exact word matching, as can be observed in Table 3. The strength of the METEOR metric is in making sentence-level or segment-level correlations.

Table 3: The matching process by different modules in METEOR.

Module	Candidate	Reference	Match
Exact	Good	Well	Yes
Stemmer	Good	Good	Yes
Synonym	Bad	Good	No

After the matching process, the unigram precision is calculated as follows:

$$Precision = \frac{Num_{match}}{Num_{uc}},$$

where Num_{match} is the number of matches between unigrams of the candidate caption and unigrams of the ground-truth reference caption, and Num_{uc} is the number of unigrams in the candidate captions.

Next, the unigram recall is obtained as follows:

$$Recall = \frac{Num_{match}}{Num_{ur}},$$

where Num_{match} is as above, and Num_{ur} is the number of unigrams in the ground-truth reference caption.

These two values are then combined to calculate the harmonic mean as below:

$$F_{mean} = \frac{10 \times Precision \times Recall}{Recall + 9 \times Precision}$$

On the other hand, a penalty is calculated to consider the larger segment matchings in addition to the word-level matches. This penalty is calculated as follows:

$$Penalty = 0.5(\frac{Num_{chunk}}{Num_{um}}),$$

where Num_{chunk} is the number of adjacent unigrams in the candidate and ground-truth captions, which are referred to as chunks. Moreover, Num_{um} is the number of mapped unigrams.

Finally, the METEOR score is calculated as below:

$$METEOR = F_{mean}(1 - Penalty)$$

6.1.3 ROUGE

The Recall-Oriented Understudy for Gisting Evaluation or ROUGE [56] is a recall-based metric, as its name suggests. This metric is suitable for evaluating the quality of text summarization; however, the performance downgrades while evaluating summaries in more than one text. There are different variants of ROUGE with respect to the mathematical approach:

- **ROUGE-N:** Calculates an n-gram recall between a candidate caption and a set of ground-truth reference captions.

- **ROUGE-L:** Searches for the longest co-occurring sequences between a candidate caption and its corresponding ground-truth captions. This is a Longest Common Subsequence or LCS-based approach.

- **ROUGE-W:** A weighted version of ROUGE-L with a bias towards favoring consecutive longest common subsequences.

- **ROUGE-S:** Measures co-occurring skip-bigrams between a generated caption and a group of ground-truth captions. Skip-bigram is any pair of words with respect to their sentence order.

In this research, we have utilized the ROUGE-L; hence, we are referring to this specific variant when we mention the ROUGE metric.

6.1.4 CIDEr

The Consensus-based Image Description Evaluation or CIDEr metric [57] is a metric designed particularly to evaluate generated captions in image captioning. This metric evaluates the consensus between the generated captions and human judgment by utilizing Term-Frequency Inverse Document Frequency (TF-IDF). CIDEr measures the number of common n-grams between the generated caption and the ground-truth captions. In addition, the n-grams in the ground-truth captions that do not exist in the generated caption are considered. On the other hand, frequent n-grams in the entire dataset are likely to contain less information; hence, lower weights are assigned to them. Calculation of the CIDEr score for n-grams of a specific length follows the below expression:

$$CIDEr_n(c_i, G_i) = \frac{1}{m}\sum_j \frac{t_n(c_i).\, t_n(g_{ij})}{\|t_n(c_i)\|\|t_n(g_{ij})\|},$$

where $CIDEr_n$ is the CIDEr score function for n-grams of length n, c_i is the i^{th} caption, g_{ij} is the j^{th} ground-truth caption for the i^{th} caption, G_i is the set of all ground-truth captions for the i^{th} caption, and t_n is the TF-IDF weighting for all n-grams of length n. Furthermore, the overall CIDEr score is formally defined as below:

$$CIDEr(c_i, G_i) = \sum_{n=1}^{N} w_n CIDEr_n(c_i, G_i),$$

where $w_n = 1/N$ is a uniform weight associated to each length-specific CIDEr score.

6.1.5 Emotional-Alignment

The Emotional-Alignment metric was proposed by Achlioptas $et\ al.$ [68] to evaluate emotion-centric captions according to their emotional class. A text-to-emotion classifier detects the emotion-class of the generated caption and compares it with the dominant emotion-class between the ground-truth captions for the same image; the percentage of the matches determines the Emotional-Alignment score. Following the work of [68], a fine-tuned pre-trained Bert model [69] has been utilized as the text-to-emotion classifier. This metric was utilized as a reward function in the SCST fine-tuning stage, which was unsuccessful because of the high variance of the scores, as explained in Section 4.4.2.

6.2 Ablation Study

In this section, we evaluate the impact of removing or modifying different components of the proposed model. This will help us to understand how each of the proposed components contributes to the overall performance of the model.

6.2.1 Visual Encoder

First, we evaluate the role of employing different models as the visual encoder to extract visual features. We have experimented detecting the object bounding boxes using the "Faster RCNN pre-trained on Visual Genome" [81] visual encoder. Also, extracting grid-based features via CLIP [72] and region-based features via BLIP [78] have been examined. In particular, the CLIP-RN50×16 variant has been utilized, which is based on an EfficientNet-style [75] scaling. On the other hand,

we have utilized the BLIP-ViT-L/16 variant, which is based on the Vision Transformer [43] approach. The details of these visual encoders have been discussed previously (Section 4.2.1).

Table 4 and Table 5 contain the evaluation scores corresponding to the utilization of each visual encoder in the student model and the teacher model, respectively. As it can be observed, the best performance is achieved by the teacher model of Nemesis utilizing CLIP-RN50×16 as the visual encoder; hence, we will be referring to this exact configuration when mentioning the Nemesis. For the EGNemesis, the best performance is achieved by the student model utilizing BLIP-ViT-L/16; therefore, this particular configuration will be referred to as the EGNemesis in the following sections.

Table 4: Performance of the student model utilizing different visual encoders for both the Nemesis and EGNemesis models. (B: BLEU, M: METEOR, R: ROUGE, C: CIDEr)

Model	Visual encoder	B-1	B-2	B-3	B-4	M	R	C
Nemesis	Faster R-CNN	0.498	0.273	0.151	0.086	0.130	0.276	0.087
	CLIP-RN50×16	0.532	0.304	0.172	0.102	0.137	0.290	0.120
	BLIP-ViT-L/16	0.509	0.290	0.165	0.097	0.137	0.281	0.116
EGNemesis	Faster R-CNN	0.455	0.233	0.122	0.066	0.114	0.243	0.066
	CLIP-RN50×16	0.472	0.251	0.134	0.076	0.124	0.254	0.095
	BLIP-ViT-L/16	**0.479**	**0.260**	**0.141**	**0.080**	**0.129**	**0.262**	**0.099**

Table 5: Performance of the teacher model utilizing different visual encoders for both the Nemesis and EGNemesis models. (B: BLEU, M: METEOR, R: ROUGE, C: CIDEr)

Model	Visual encoder	B-1	B-2	B-3	B-4	M	R	C
Nemesis	Faster R-CNN	0.503	0.277	0.154	0.089	0.141	0.278	0.093
	CLIP-RN50×16	**0.539**	**0.311**	**0.178**	**0.106**	**0.141**	**0.294**	**0.130**
	BLIP-ViT-L/16	0.526	0.304	0.175	0.105	0.138	0.291	0.127
EGNemesis	Faster R-CNN	0.458	0.233	0.121	0.066	0.118	0.242	0.070
	CLIP-RN50×16	0.475	0.252	0.136	0.076	0.124	0.254	0.095
	BLIP-ViT-L/16	0.470	0.252	0.137	0.077	0.123	0.255	0.099

6.2.2 SCST Fine-tuning

Table 6 shows the effect of the SCST fine-tuning stage, where the CIDEr metric is used as the reward function to encourage the model's generations that outperform the current test-time model following the process discussed in Section 4.4.2. As it is observable, the model's performance is boosted with respect to all utilized metrics in comparison with the same model after the cross-entropy training phase. The most significant improvements are related to the CIDEr and BLEU-1 scores. The BLEU-1 metric has been increased from 0.539 to 0.711 for the Nemesis, and from 0.479 to 0.700 for the EGNemesis. On the other hand, the CIDEr score has boosted from 0.130 to 0.219 for the Nemesis, and from 0.099 to 0.224 for the EGNemesis.

Table 6: The comparison of the results before and after applying the SCST fine-tuning.

Metric	*Nemesis*	$Nemesis_{SCST}$	*EGNemesis*	$EGNemesis_{SCST}$
BLEU-1	0.539	0.711	0.479	0.700
BLEU-2	0.311	0.406	0.260	0.403
BLEU-3	0.178	0.211	0.141	0.214
BLEU-4	0.106	0.113	0.080	0.115
METEOR	0.141	0.166	0.129	0.165
ROUGE-L	0.294	0.341	0.262	0.336
CIDEr	0.130	0.219	0.099	0.224

6.2.3 Image-to-Emotion Classifier

A comparison of the model's performance according to the utilization of different image-to-emotion classifiers can be found in Table 7. These classifiers are: (1) the ResNet-32 classifier pre-trained on the ImageNet dataset (IN), which gave the best performance in the previous work by Achlioptas et al. [68]. (2) The ResNet-50 classifier pre-trained on the Stylized-ImageNet dataset (SIN), which is our proposed classifier.

Table 7: Results with respect to the utilized image-to-emotion classifier.

Metric	$EGNemesis_{IN}$	$EGNemesis_{SIN}$
BLEU-1	0.466	0.479
BLEU-2	0.251	0.260
BLEU-3	0.137	0.141
BLEU-4	0.077	0.080
METEOR	0.128	0.129
ROUGE-L	0.253	0.262
CIDEr	0.093	0.099

As it is observable, the performance has improved by using our proposed classifier module in all the utilized metrics. For instance, the BLUE-1 score has improved from 0.466 to 0.499, and the CIDEr metric has improved from 0.093 to 0.099. This proves that the decrease in texture bias while increasing shape bias will achieve a better performance in our auxiliary image-to-emotion classification task.

6.2.4 Emotional Grounding

The results with and without incorporating the extra emotional supervision signal are shown in Table 6. This signal is provided based on the emotional class indicated by the image-to-emotion classifier during the training time to keep the assessment fair. As it can be observed, the evaluation scores experience a decrease after emotional grounding; however, this degradation in evaluation metrics does not necessarily indicate a decrease in the quality of generated captions in our case. Most evaluation metrics return a higher score if the generated caption includes more words from the ground-truth captions or their synonyms, which is not the best way to assess generated captions

in our subjective emotion-centric utterances. In fact, an increase in the diversity of the captions can result in a degradation of evaluation metrics. As you can see in Figure 35, the generated caption of Nemesis for the first image is “it looks like a cold winter day.“. While the EGNemesis generated “this painting makes me feel nostalgic. it reminds me of my childhood” grounded on the “Contentment” emotion-class, which is more emotionally rich according to human judgment. However, the emotionally grounded utterance will achieve a lower evaluation score since it does not contain the frequent words in the ground-truth captions, which are “cold” and “winter.”

Figure 35. Examples of the generated captions for unseen artworks. These samples include utterances from Nemesis model, and EGNemesis model along with the emotion-class extracted by the image-to-emotion classifier.

6.3 Comparison with the State-of-the-art

In this section, we will compare our proposed model's performance with the state-of-the-art emotion-centric neural speakers. This comparison is with respect to both the auxiliary classification task and the emotion-centric image captioning task.

6.3.1 Auxiliary Classification

The 9-way emotional classification problem is an extremely challenging task because of the subjectivity of emotions and diversity of emotional utterances. In the previous work, a user study has been done to measure the accuracy of this classification task by humans [68]. This study consisted of three human experts attempting to guess the dominant emotion-class based on a ground-truth utterance of ArtEmis, where they achieved an accuracy of only 61.2%. This depicts how challenging this task is, even based on human judgment. However, the BERT-based text-to-emotion classifier utilized in our model achieved a 64.8% accuracy, which is a surprising performance compared to the human results.

For the image-to-emotion classification task, which is arguably more difficult than the text-to-emotion classification, the ResNet-32 module pre-trained on ImageNet (IN) achieved a 60.2% accuracy, while the ResNet-50 module pre-trained on Stylized-ImageNet (SIN) achieved a 59.4% accuracy. However, the accuracy of this auxiliary task is not directly important to us. As mentioned earlier, Table 7 shows that our model performs better utilizing the classifier trained on SIN because of the more diverse and shape-driven label predictions.

6.3.2 Emotion-centric Image Captioning Task

We compare our proposed neural speaker to the best performing emotion-centric image captioning models introduced by [68] on the ArtEmis dataset. These neural speakers include a captioning model inspired by Meshed-Memory Transformer (M^2) [42] architecture, and an LSTM-based model inspired by "Show, Attend and Tell" (SAT) [23]. In addition, this comparison includes the emotionally grounded variations of these models (i.e. M^2-EG and SATEG).

As can be observed in Table 8, our proposed model outperforms both the emotionally grounded and standard variations of M^2 and SAT speakers with respect to all incorporated metrics. This improvement is more notable in models after the SCST fine-tuning stage. The only exception is the EGNemesis, which is the reason for this degradation has been elaborated on previously. Figure 36 shows some generated utterances from SATEG and EGNemesis models, where EGNemesis appears to generate more abstract, diverse, emotionally rich, and human-like captions.

Table 8: Comparison of state-of-the-art results and Nemesis after both cross-entropy training and SCST fine-tuning.

Metric	*SAT*	*SATEG*	M^2	M^2-*EG*	*Nemesis*	$Nemesis_{SCST}$	*EGNemesis*	$EGNemesis_{SCST}$
BLEU-1	0.536	0.520	0.507	0.511	0.539	0.711	0.479	0.700
BLEU-2	0.290	0.280	0.282	0.282	0.311	0.406	0.260	0.403
BLEU-3	0.155	0.146	0.159	0.154	0.178	0.211	0.141	0.241
BLEU-4	0.087	0.079	0.095	0.090	0.106	0.113	0.080	0.115
METEOR	0.142	0.134	0.140	0.137	0.141	0.166	0.129	0.165
ROUGE-L	0.297	0.294	0.280	0.286	0.294	0.341	0.262	0.336

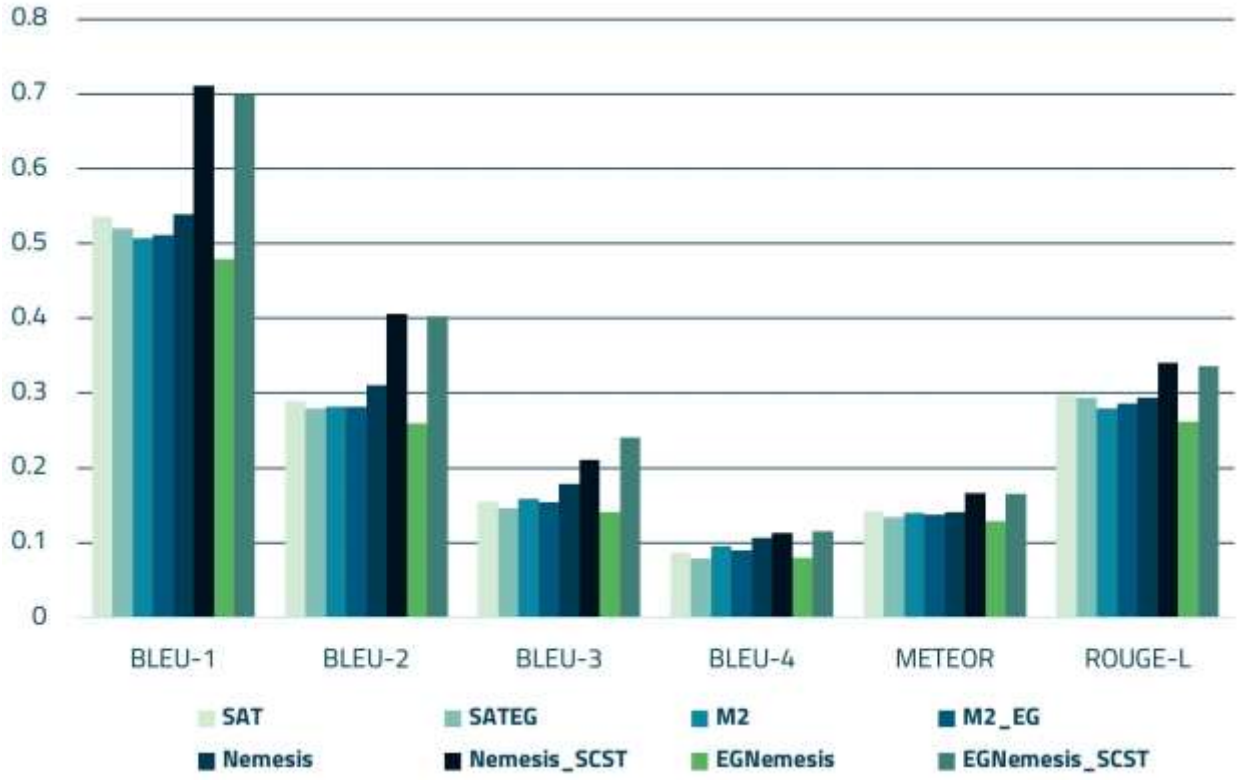


Figure 36. Comparison of state-of-the-art results and Nemesis after both cross-entropy training and SCST fine-tuning.

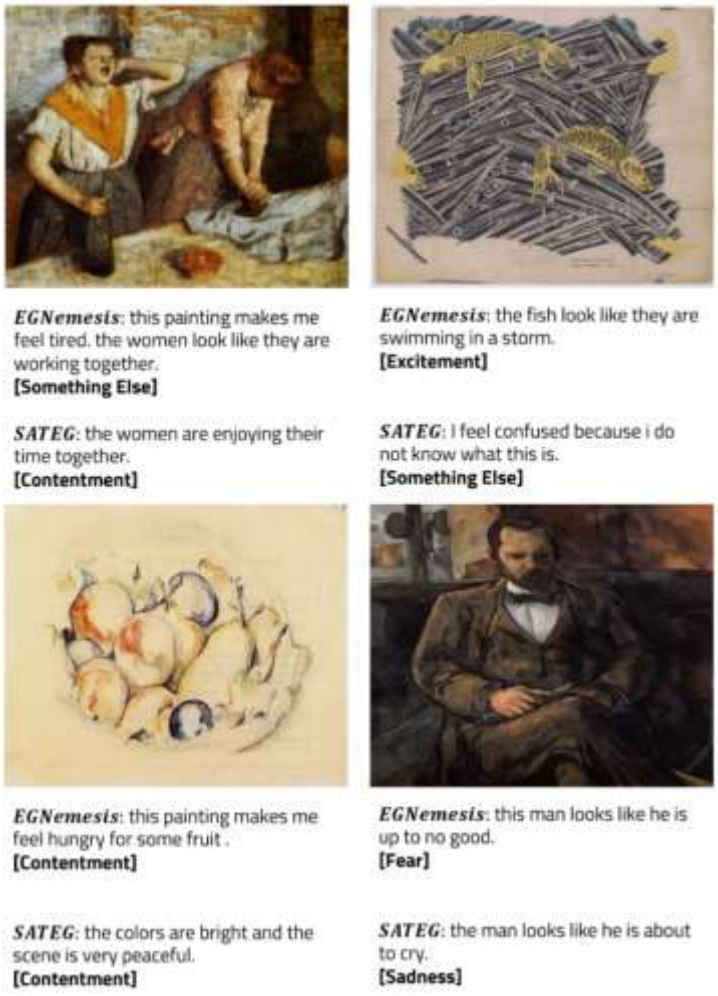


Figure 37. A comparison between the examples of generated captions by EGNemesis and SATEG models along with the emotional class extracted by the image-to-emotion classifier, which has been utilized in the emotional grounding process.

Chapter 7
Conclusions and Future Work

7 Conclusions and Future Work

Most of the previous work in the image captioning field has been focused on generating purely logical objective-based captions according to the objects in the input image. These captions lack some human-like attributes such as personality and emotion since they only contain factual information in a neutral tone. This has led to a lack of connection between the user and the generated caption.

Neural speakers capable of producing affective utterances are an important step toward generating more engaging captions by provoking human emotions. As humans, emotions are a crucial part of expressing ourselves when we aim to describe different phenomena. Therefore, it is logical to expect the automatic image captioning process to consider this essential aspect of our perceptions.

In this research, we introduced Nemesis, Neural Mean Teacher Learning-based Emotion-centric Speaker, an image captioning model capable of describing emotional responses to visual stimuli. We showed that incorporating a mean teacher learning-based approach followed by SCST-based fine-tuning, which utilizes extra emotional supervision signals, is a promising path toward generating more human-like emotion-centric descriptions. This was achieved by both experimenting with the utilization of different modules in the proposed pipeline and comparing it with the latest state-of-the-art methods. The results indicated that utilizing the emotional grounding process improved the capability of incorporating various emotions to impact the word selection

and sentencing processes, which led to the increased diversity of the generated captions. On the other hand, we observed that the SCST-based fine-tuning stage boosted the evaluation scores significantly by overcoming the non-differentiability of the utilized metrics.

Another part of this research was the auxiliary emotion classification tasks to classify both captions and images to the different emotion-classes. We proposed a novel image-to-emotion classifier that overcame the existing texture bias in the previously widely used classifiers. This was achieved by encouraging the model to favor the bias toward shapes instead of textures.

7.1 Future Work

The main challenge in the task of emotion-centric image captioning is the lack of a proper evaluation metric that aligns well with human judgment. Most of the evaluation metrics focus on comparing words in the generated captions with the words or synonyms in the reference captions, which is not the best approach for our diverse and subjective task. For example, the newly introduced StyleCIDEr [90] metric can be a suitable alternative to utilize in the SCST fine-tuning stage. This metric compares the captions with respect to their styles, which in our case, the styles are the emotion-classes.

Another path for future research can be focused on utilizing a more efficient visual encoder to improve the scene-understanding ability of the model. A crucial indicator of the capability of a multi-modal pre-trained is the utilized dataset in the pre-training stage. Bigger datasets will enhance the model's generalizability, leading to better performance in the downstream tasks. Recently, the LAION-5B [91] dataset was introduced, including more than 5 billion CLIP-filtered [72] image-text pairs, which can be used to pre-train more capable visual encoders.

Finally, Mohamed *et al.* [92] showed that the ArtEmis dataset is biased towards some emotions. Therefore, they introduced the ArtEmis v2.0 dataset, which contains contrastive samples to balance the ArtEmis [68] dataset as a complementary dataset. Utilizing the balanced dataset can improve the proposed model's performance.

References

[1] M. Stefanini, M. Cornia, L. Baraldi, S. Cascianelli, G. Fiameni, and R. Cucchiara, “From show to tell: A survey on deep learning-based image captioning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. PP, pp. 1–1, 2022.

[2] Xie, E., Ding, J., Wang, W., Zhan, X., Xu, H., Sun, P., Li, Z. and Luo, P., 2021. Detco: Unsupervised contrastive learning for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 8392-8401).

[3] Singh, A., Chakraborty, O., Varshney, A., Panda, R., Feris, R., Saenko, K. and Das, A., 2021. Semi-supervised action recognition with temporal contrastive learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10389-10399).

[4] Ling, J., Feng, Z., Zheng, D., Yang, J., Yu, H. and Xiao, X., 2021. Robust adaptive motion tracking of piezoelectric actuated stages using online neural-network-based sliding mode control. Mechanical Systems and Signal Processing, 150, p.107235.

[5] Fan, A., Bhosale, S., Schwenk, H., Ma, Z., El-Kishky, A., Goyal, S., Baines, M., Celebi, O., Wenzek, G., Chaudhary, V. and Goyal, N., 2021. Beyond english-centric multilingual machine translation. Journal of Machine Learning Research, 22(107), pp.1-48.

[6] Baevski, A., Hsu, W.N., Conneau, A. and Auli, M., 2021. Unsupervised speech recognition. Advances in Neural Information Processing Systems, 34.

[7] Garcia, K. and Berton, L., 2021. Topic detection and sentiment analysis in Twitter content related to COVID-19 from Brazil and the USA. Applied soft computing, 101, p.107057.

[8] Pan, Jia-Yu, et al. “Automatic image captioning.” 2004 IEEE International Conference on Multimedia and Expo (ICME)(IEEE Cat. No. 04TH8763). Vol. 3. IEEE, 2004.

[9] Ordonez, Vicente, Girish Kulkarni, and Tamara Berg. “Im2text: Describing images using 1 million captioned photographs.” Advances in neural information processing systems 24 (2011).

[10] Yang, Yezhou, et al. “Corpus-guided sentence generation of natural images.” Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 2011.

[11] Gupta, Ankush, Yashaswi Verma, and C. Jawahar. “Choosing linguistics over vision to describe images.” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 26. No. 1. 2012.

[12] Yao, Benjamin Z., et al. “I2t: Image parsing to text description.” Proceedings of the IEEE 98.8 (2010): 1485-1508.

[13] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3156–3164.

[14] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3128–3137.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[16] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille, “Deep Captioning with Multimodal Recurrent Neural Networks (m- RNN),” in ICLR, 2015.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in NeurIPS, 2012.

[18] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in ICLR, 2015.

[19] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, “Image captioning with semantic attention,” in CVPR, 2016.

[20] Q. Wu, C. Shen, L. Liu, A. Dick, and A. Van Den Hengel, “What Value Do Explicit High Level Concepts Have in Vision to Language Problems?” in CVPR, 2016.

[21] F. Chen, R. Ji, X. Sun, Y. Wu, and J. Su, “GroupCap: Group-based Image Captioning with Structured Relevance and Diversity Con- straints,” in CVPR, 2018.

[22] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, “Self- critical sequence training for image captioning,” in CVPR, 2017.

[23] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in ICML, 2015.

[24] J. Lu, C. Xiong, D. Parikh, and R. Socher, “Knowing when to look: Adaptive attention via a visual sentinel for image captioning,” in CVPR, 2017.

[25] Z. Yang, Y. Yuan, Y. Wu, W. W. Cohen, and R. R. Salakhutdinov, “Review Networks for Caption Generation,” in NeurIPS, 2016.

[26] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.- S. Chua, “SCA-CNN: Spatial and Channel-wise Attention in Convolutional Networks for Image Captioning,” in CVPR, 2017.

[27] W. Jiang, L. Ma, Y.-G. Jiang, W. Liu, and T. Zhang, “Recurrent Fusion Network for Image Captioning,” in ECCV, 2018.

[28] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, “Bottom-up and top-down attention for image captioning and visual question answering,” in CVPR, 2018.

[29] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in NeurIPS, 2015.

[30] Z.-J. Zha, D. Liu, H. Zhang, Y. Zhang, and F. Wu, “Context-aware visual policy network for fine-grained image captioning,” IEEE Trans. PAMI, 2019.

[31] L. Guo, J. Liu, J. Tang, J. Li, W. Luo, and H. Lu, “Aligning linguistic words and visual semantic units for image captioning,” in ACM Multimedia, 2019.

[32] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in ICLR, 2017.

[33] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei, “Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations,” IJCV, vol. 123, no. 1, pp. 32–73, 2017.

[34] X. Yang, K. Tang, H. Zhang, and J. Cai, “Auto-Encoding Scene Graphs for Image Captioning,” in CVPR, 2019.

[35] Z. Shi, X. Zhou, X. Qiu, and X. Zhu, “Improving Image Captioning with Better Use of Captions,” in ACL, 2020.

[36] T. Yao, Y. Pan, Y. Li, and T. Mei, “Hierarchy Parsing for Image Captioning,” in ICCV, 2019.

[37] G. Li, L. Zhu, P. Liu, and Y. Yang, “Entangled Transformer for Image Captioning,” in ICCV, 2019.

[38] S. Herdade, A. Kappeler, K. Boakye, and J. Soares, “Image Cap- tioning: Transforming Objects into Words,” in NeurIPS, 2019.

[39] L. Guo, J. Liu, X. Zhu, P. Yao, S. Lu, and H. Lu, “Normalized and Geometry-Aware Self-Attention Network for Image Captioning,” in CVPR, 2020.

[40] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, “Attention on Attention for Image Captioning,” in ICCV, 2019.

[41] Y. Pan, T. Yao, Y. Li, and T. Mei, “X-Linear Attention Networks for Image Captioning,” in CVPR, 2020.

[42] M. Cornia, M. Stefanini, L. Baraldi, and R. Cucchiara, “Meshed- Memory Transformer for Image Captioning,” in CVPR, 2020.

[43] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly et al., “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” ICLR, 2021.

[44] L. Zhou, H. Palangi, L. Zhang, H. Hu, J. J. Corso, and J. Gao, “Unified Vision-Language Pre-Training for Image Captioning and VQA,” in AAAI, 2020.

[45] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in CVPR, 2015.

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

[47] Gan, Chuang, et al. “Stylenet: Generating attractive visual captions with styles.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[48] Mathews, Alexander, Lexing Xie, and Xuming He. “Semstyle: Learning to generate stylised image captions using unaligned text.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[49] Guo, Longteng, et al. “Mscap: Multi-style image captioning with unpaired stylized text.” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

[50] W. Zhao, X. Wu, and X. Zhang, “Memcap: Memorizing style knowledge for image captioning,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 07, 2020, pp. 12 984–12 992.

[51] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, Computer Vision – ECCV2014, pages 740–755, Manhattan, New York, USA, 2015. Springer, Springer International Publishing. ISBN 978-3-319-10602-1.

[52] P. Sharma, N. Ding, S. Goodman, and R. Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In Proceedings of the

56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2556–2565, Stroudsburg, PA, USA, July 2018. Association for Computational Linguistics. doi:10.18653/v1/P18-1238. URL https://aclanthology.org/P18-1238.

[53] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. International Journal of Computer Vision, 123(1):74–93, May 2017. ISSN 1573-1405. doi:10.1007/s11263-016-0965-7. URL https://doi.org/10.1007/s11263-016-0965-7.

[54] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," ser. ACL '02. Association for Computational Linguistics, 2002, p. 311–318.

[55] A. Lavie and A. Agarwal, "Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments," pp. 228–231, 07 2007.

[56] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," 2004, p. 10.

[57] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Jun. 2015.

[58] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," arXiv preprint arXiv:1610.02242, 2016.

[59] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deeplearning results,” Advances in neural information processing systems, vol. 30, 2017.

[60] G. Xu, Z. Liu, X. Li, and C. C. Loy, “Knowledge distillation meets selfsupervision,” in European Conference on Computer Vision. Springer, 2020, pp. 588–604.

[61] G. Hinton, O. Vinyals, J. Dean et al., “Distilling the knowledge in a neural network,” arXiv preprint arXiv:1503.02531, vol. 2, no. 7, 2015.

[62] J. Ba and R. Caruana, “Do deep nets really need to be deep?” Advances in neural information processing systems, vol. 27, 2014.

[63] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” Machine learning, vol. 8, no. 3, pp. 229–256, 1992.

[64] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, “Improved image captioning via policy gradient optimization of spider,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 873–881.

[65] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” Advances in neural information processing systems, vol. 12, 1999.

[66] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, “Sequence level training with recurrent neural networks,” arXiv preprint arXiv:1511.06732, 2015.

[67] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, “Self-critical sequence training for image captioning,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Jul. 2017.

[68] P. Achlioptas, M. Ovsjanikov, K. Haydarov, M. Elhoseiny, and L. J. Guibas, “Artemis: Affective language for visual art,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 11 569–11 579.

[69] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, in Proceedings of the 2019 Conference of the North. Association for Computational Linguistics, 2019.

[70] M. Barraco, M. Stefanini, M. Cornia, S. Cascianelli, L. Baraldi, and R. Cucchiara, “CaMEL: Mean Teacher Learning for Image Captioning,” in International Conference on Pattern Recognition, 2022.

[71] Y. Wang, C. Albrecht, and X. Zhu, “Self-supervised vision transformers for joint sar-optical representation learning,” 04 2022.

[72] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (pp. 8748-8763). PMLR.

[73] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI’08). 646–651.

[74] Xian, Yongqin, Bernt Schiele, and Zeynep Akata. "Zero-shot learning-the good, the bad and the ugly." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4582-4591. 2017.

[75] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in Proceedings of the 36th International Conference on Machine Learning, vol. 97, 2019, pp. 6105–6114.

[76] He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., and Li, M. Bag of tricks for image classification with convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 558– 567, 2019.

[77] Zhang, R., 2019, May. Making convolutional networks shift-invariant again. In *International conference on machine learning* (pp. 7324-7334). PMLR.

[78] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," 01 2022.

[79] Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., and Komatsuzaki, A. Laion-400m: Open dataset of clipfiltered 400 million image-text pairs. arXiv preprint arXiv:2111.02114, 2021.

[80] Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." *arXiv preprint arXiv:1711.05101* (2017).

[81] Shilrley6. 2020. Faster R-CNN with model pretrained on Visual Genome. https://github.com/shilrley6/Faster-R-CNN-with-model-pretrained-on-Visual-Genome#faster-r-cnn-with-model-pretrained-on-visual-genome. (2022)

[82] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness." In International Conference on Learning Representations, 2019.

[83] Huang, Xun, and Serge Belongie. "Arbitrary style transfer in real-time with adaptive instance normalization." In *Proceedings of the IEEE international conference on computer vision*, pp. 1501-1510. 2017.

[84] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. and Desmaison, A., 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, *32*.

[85] Collobert, R., Bengio, S. and Mariéthoz, J., 2002. *Torch: a modular machine learning software library* (No. REP_WORK). Idiap.

[86] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)* (pp. 265-283).

[87] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” vol. 14, 2014, pp. 1532–1543

[88] R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” 2016, pp. 1715–1725.

[89] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” CoRR, vol. abs/1412.6980, 2015.

[90] Li, Chengxi, and Brent Harrison. "StyleM: Stylized Metrics for Image Captioning Built with Contrastive N-grams." *arXiv preprint arXiv:2201.00975* (2022).

[91] Schuhmann, Christoph, Romain Beaumont, Cade W. Gordon, Ross Wightman, Theo Coombes, Aarush Katta, Clayton Mullis et al. "LAION-5B: An open large-scale dataset for training next generation image-text models."

[92] Mohamed, Youssef, Faizan Farooq Khan, Kilichbek Haydarov, and Mohamed Elhoseiny. "It is Okay to Not Be Okay: Overcoming Emotional Bias in Affective Image Captioning by

Contrastive Data Collection." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21263-21272. 2022.