HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION COMMUNICATION TECHNOLOGY

CAPSTONE PROJECT REPORT:

IMAGE CAPTION GENERATOR

Supervised by:

Professor Nguyen Hung Son

Group member:

Name	Roles	ID
Bui Khac Chien	EDA, Evaluation, LSTM Report	20220059
Trinh Hoang Anh	Transformer Model	20225470
Tran Quang Minh	VGG16 – LSTM Model	20225512
Vo Ta Quang Nhat	UI	20225454
Bach Nhat Minh	Slide, Data Preprocessing	20225509

TABLE OF CONTENTS

Abstract

Image caption generation is a key task that bridges the gap between images and their textual descriptions, gaining growing attention in artificial intelligence research. Both images and text are regarded as distinct forms of information representation that align symmetrically to convey the same visual scene. Image captioning refers to the automatic generation of natural language descriptions based on the content present in an image. It plays a vital role in scene understanding by integrating computer vision and natural language processing techniques. The applications of image captioning are broad and impactful, such as facilitating human-computer interaction. This report examines the performance of Transformer and LSTM models on the MS COCO 2017 dataset, a significant and widely adopted benchmark in computer vision and image captioning tasks. Additionally, the strengths and limitations of these models are analyzed using several common evaluation metrics, including CIDEr, SPICE, METEOR, ROUGE-L, and BLEU. Finally, the report discusses some unresolved challenges in the field of image captioning and compare our results with some state-of-the-art model like mPLUG and OFA.

Keywords: Image caption generation, LSTM, Transformer, MS COCO

1. Introduction

1.1. Problem Description

Automatic generation of image captions is drawing increasing interest in artificial intelligence. It is a fundamental task to build a bridge between image and its description in text. The main aim of image caption generation is to make machines generate a textual sentence to accurately depict the content of a given image. Therefore, it is related to two major artificial intelligence fields: computer visual (CV) and natural language processing (NLP). Image caption generation can be applied to different aspects which simultaneously involve visual systems and language systems, such as semantic visual search, visual intelligence in chatting robots, and assisting visual impaired people to perceive the content of surrounding scenes.

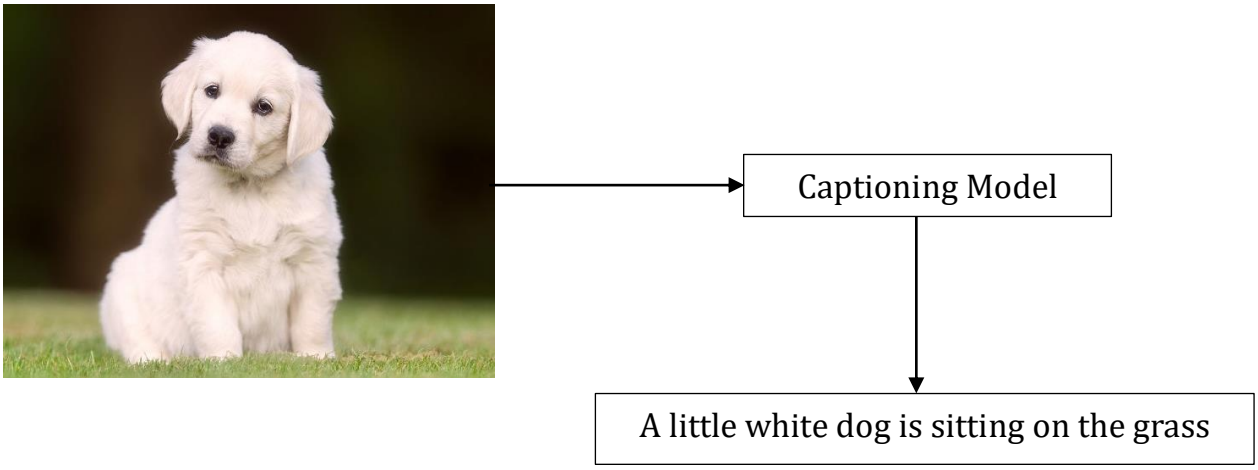

Figure 1: Image Caption Generator

Early methods for image description generation relied on static object class libraries and statistical language models to aggregate image information. For instance, Li et al. [1] introduced an *n*-gram approach that constructed image descriptions by collecting and combining candidate phrases from scratch. Similarly, Lin et al. [2] employed a 3D visual analysis system to identify objects, attributes, and relationships within an image, transforming them into semantic trees and subsequently applying grammar learning techniques to generate textual descriptions.

While these approaches were innovative and unique in their methodologies, they shared a common limitation: they failed to intuitively capture features related to objects or actions in the image and lacked an end-to-end, generalized model to solve the problem comprehensively. The advent of neural networks brought significant advancements to the field of image description, especially with the rise of big data and deep learning techniques.

Initially, image captioning tasks were addressed using Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks. Although these models produced captions effectively, they struggled with the vanishing gradient problem, where gradients diminished when processing longer sequences. To overcome this limitation, the Attention Mechanism was introduced alongside RNNs and LSTMs, enabling models to focus on the most critical parts of the input and make predictions based on relevant features.

In recent years, the Transformer architecture has emerged as a dominant solution for image captioning. Leveraging self-attention mechanisms, Transformers have demonstrated exceptional success not only in image captioning but also across

various natural language processing (NLP) tasks, such as machine translation and text classification.

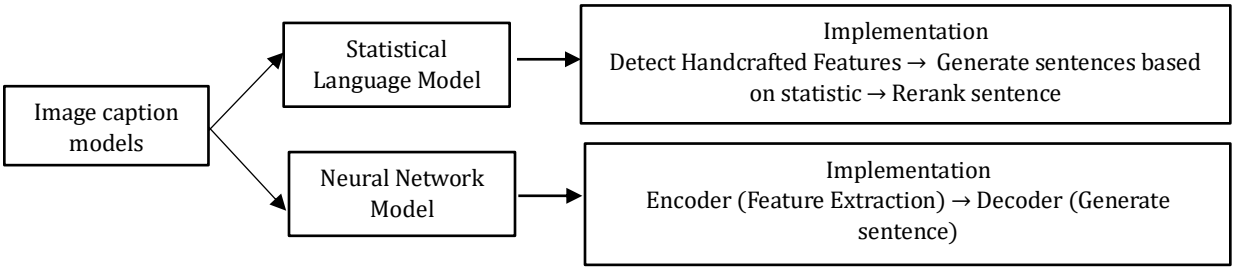

Figure 2: Approaches in Image caption models

In this report, we present an overview of the attention mechanism, along with two models: LSTM and Transformer. We evaluate their performance under identical conditions using the MS COCO dataset for training and validation, employing multiple performance metrics such as BLEU, METEOR, ROUGE-L, CIDEr, and SPICE.
The report is structured as follows: Section 2 provides an overview of the MS COCO dataset and explains the evaluation criteria. Section 3 introduces the fundamental models, methodologies, and attention mechanisms used to enhance model performance and address their limitations. Section 4 analyzes and compares the performance of the two models. Finally, Section 5 summarizes the current work, highlights state-of-the-art models, and discusses future directions and expectations.

1.2. Goals and Targets

a. Primary Goals: Comprehend the Fundamentals of Image Caption Generation

- *Establish an Effective Pipeline*: Identify an efficient workflow to connect visual features from images with output sentences, ensuring the production of contextually relevant and meaningful phrases.
- *Analyze LSTM and Transformer Architectures*: Provide an in-depth exploration of Long Short-Term Memory (LSTM) networks and the Transformer architecture, comparing their performance using standard evaluation metrics.

b. Secondary Goals:

- *Model Optimization*: Compare the performance of LSTM networks with and without the integration of attention mechanisms. Evaluate the impact of various forms of attention, such as *hard attention* and *soft attention*, on model efficiency and accuracy.
- *Quantitative Validation*: Measure and validate the effectiveness of attention mechanisms in improving caption generation, highlighting their contribution to achieving state-of-the-art results.

c. Target Objectives:

- *Develop a Multi-Language Image Caption Generator Application*: Create a user-friendly application that enables users to upload images, choose specific tones (e.g., fun, sad, angry), and include additional descriptions.
- *Explore Cross-Field Applications*: Extend the application's utility to other domains, including social media, robotics, and education, showcasing its versatility and real-world relevance.

2. Dataset and Evaluation

2.1. Dataset

In this report, we utilize the COCO 2017 dataset (Common Objects in Context 2017) [3], which is designed for instance segmentation, semantic segmentation, and object detection tasks. It is widely applicable across various domains.

The dataset, captures images from complex daily scences, comprises 163,957 images containing 2,099,063 labeled objects from 80 different categories, such as person, chair, car, dining table, cup, bottle, bowl, handbag, truck, and more, including backpack, book, cell phone, clock, TV, potted plant, couch, dog, sports ball, traffic light, cat, bus, train, and 52 additional classes. Additionally, 41,739 images (25% of the total) are unlabeled, meaning they lack annotations.

The dataset is divided into three subsets: train2017 with 118,287 images, test2017 with 40,670 images, and val2017 with 5,000 images. Each image is annotated by five human annotators, and the validation set is used for parameter tuning.

Table 1: Comparisions of reference captions on 3 well-known datasets MS COCO, Flick8K and Flickr30K

Dataset name	Size			No.Captions/Image	Top-10 Words with Higher Occurences
	Train	Valid	Test		
MS COCO	118,287	5,000	40,670	5	a, on, of, the, in, with, and, is, man, to
Flickr8k	6,000	1,000	1,000	5	a, in, the, on, is, and, dog, with, man, of
Flickr30k	28,000	1,000	1,000	5	a, in, the, on, and, man, is, of, with, woman

The MS COCO 2017 dataset consists of four folders: train, validation, and test, which store images and annotations in JSON file format. For training and validation, the files "a" are used, respectively. Within the JSON files, the two main fields of interest are "images" and "annotations".

- The "images" field contains a list of dictionaries, where each dictionary provides key-value pairs describing details about a single image in the COCO dataset.

- The "annotations" field includes a list of dictionaries, where each dictionary specifies the corresponding image ID and its caption, representing objects within the images in the dataset.

The MS COCO dataset includes 80 annotation classes, with general statistics and class distributions highlighting the 10 most prevalent classes. These top 10 classes, ranked by frequency, provide insights into the overall balance of labeled objects within the COCO 2017 dataset.

Table 2: 10 most prevalent classes statistics and class distributions

Class	Image	Objects	Count on image *average*	Area on image *average*
person	66808	649193	9.72	29.33%
chair	13354	98848	7.4	11.35%
car	12786	110156	8.62	8.71%
dinning table	12338	44412	3.6	43.66%
cup	9579	44616	4.66	4.98%
bottle	8880	55003	6.19	4.23%
bowl	7425	31719	4.27	15.96%
handbag	7133	28871	4.05	3.54%
truck	6377	23846	3.74	16.8%
bench	5805	28267	4.87	12.96%

Source: Dataset Ninja

Caption 1: People are walking and riding motorcycles on the street.
Caption 2: A group of motorists pass very large buildings in asia.
Caption 3: A bunch of bikers are gathered on a city street.
Caption 4: People ride their motorcycles beside some cars, passing by an empty street with stores and apartment buildings.
Caption 5: A view of motorcyclists riding their bikes through heavy city traffic.

Figure 3: An example in MS COCO dataset image

2.2. Evaluation Criteria

Numerous performance metrics exist for assessing the effectiveness of image caption generation. In this study, we utilized five evaluation metrics: BLEU [4], METEOR [5], ROUGE-L [7], CIDEr [8], and SPICE [9] to comprehensively analyze the model's performance. These metrics serve distinct purposes: BLEU and METEOR are traditionally applied to machine translation, ROUGE is typically used for automated summarization, while CIDEr and SPICE are specifically designed for image captioning.

The metrics primarily assess the consistency of n-grams between generated and reference sentences, with a focus on the significance and uniqueness of these n-gram sequences. A key advantage is that all five indicators can be conveniently calculated using the MS COCO caption evaluation toolkit, which offers publicly accessible source code.

2.2.1. BLEU (Bilingual Evaluation Understudy)

A method for evaluating machine translation performance is based on the central principle that "*the closer a machine translation is to a professional human translation, the better it is.*" This method assesses the similarity of n-grams between the machine-generated text and the reference human translation.

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

where:

- BP is Brevity Penalty which penalizes sentences that are too short.

$$BP = \begin{cases} 1, & c > r \\ e^{1-r/c}, & c \leq r \end{cases}$$

  - c is the predicted length = number of words in the predicted sentence
  - r is the target length = number of words in the target sentence
- w_n is the weight applied to the n-gram accuracy score. Weights are often set to 1/n, where n refers to the number of n-gram sizes utilized.
- p_n represents the precision rating (number of words in the predicted sentence that also occur in the target sentence) for the n-gram size.

BLEU Score can be computed for different values of N:

- BLEU-1 (N = 1): uses the unigram Precision score
- BLEU-2 (N = 2): uses the geometric average of unigram and bigram precision.
- BLEU-3 (N = 3): uses the geometric average of unigram, bigram and trigram precision.

The advantage of BLEU is that the granularity it considers is an n-gram rather than a word, considering longer matching information. However, the disadvantages of BLEU are that: it looks only for exact word matches instead of variant of the same word, it does not consider the meaning of words, ignores the importance of words and does not consider the order of words.

2.2.2. METEOR (Metric for Evaluation of Translation with Explicit Ordering)

METEOR is a metric used to evaluate machine translation by aligning the model-generated translation with the reference translation and measuring precision, recall, and F-score across various scenarios. It often uses tools like WordNet or the Porter-Stemmer algorithm to map words one-to-one in the sentences. METEOR was

developed to address several shortcomings of BLEU. Unlike BLEU, it has a stronger correlation with human judgment, not only at the overall corpus level but also at the sentence and segment levels.

The METEOR score is calculated using precision and recall, which are widely used measures in information retrieval:

- Precision refers to the proportion of words in the machine-translated text that are correctly translated, according to the reference.

$$Precision = \frac{TP}{TP + FP}$$

- Recall measures the proportion of words in the reference translation that appear in the machine-translated text.

$$Recall = \frac{TP}{TP + FN}$$

METEOR combines these two measures into a single score using the harmonic mean, which is a type of average that gives a balance between precision and recall.

$$F_{mean} = \frac{10.P.R}{R + \alpha.P}$$

$$\alpha = \begin{cases} < 1, & \text{Recall has a greater influence} \\ = 1, & \text{equally weighted} \\ > 1, & \text{Precision has a greater influence} \end{cases}$$

Previous metrics focused solely on matching single words (unigrams) without considering longer sequences that appear in both the machine-translated sentence (candidate sentence) and the reference sentence. This limitation can result in translations containing correct words but arranged in an incorrect order or lacking coherence.

To address this issue, longer n-gram matches are used to calculate a penalty **p** for the alignment. To compute this penalty, unigrams are grouped into the smallest possible number of **chunks**, where a chunk is defined as a set of unigrams that are adjacent in both the hypothesis and the reference.

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

where:

- c: is the number of chunks
- u_m: is the number of unigrams that have been mapped

The final METEOR score is calculated as:

$$score = F_{mean}.(1\text{-}p)$$

Like other evaluation metrics, METEOR has certain limitations, including its dependence on language-specific features, the increased complexity of its alignment process, and its limited ability to assess the cohesion of entire paragraphs or documents effectively.

2.2.3. ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation)

ROUGE-L is a metric used to assess text summarization algorithms by measuring the longest common subsequence (LCS) between the generated text and the reference. The LCS refers to the longest sequence of words that appear in both texts in the same order, though not necessarily consecutively.

$$\text{ROUGE-L Score} = \frac{2.\text{Precision.Recall}}{\text{Precision} + \text{Recall}}$$

When using ROUGE, there is a trade-off to consider. On the positive side, ROUGE correlates well with human evaluation, is cost-effective to compute, and is language independent. However, its drawback lies in its inability to handle different words with the same meaning, as it focuses on syntactical matches rather than capturing semantic similarities.

2.2.4. CIDEr (Consensus-based Image Description Evaluation)

CIDEr is a metric tailored specifically for image captioning tasks, assessing the similarity between a machine-generated caption and a group of reference captions. It evaluates the accuracy of image descriptions by computing Term Frequency-Inverse Document Frequency (TF-IDF) weights for each n-gram. Here, each sentence is regarded as a “document,” represented as a TF-IDF vector, and the similarity is quantified using cosine similarity between the generated caption and the reference captions. In essence, $CIDE_{r_n}$ functions within a vector space framework. For a

candidate sentence c_i and a set of reference descriptions $S_i = \{s_{i1},\ldots,s_{im}\}$, the $CIDE_{r_n}$ score is calculated accordingly.

$$CIDE_{r_n}(c_i, S_i) = \frac{1}{n}\sum_j \frac{\boldsymbol{g^n}(c_i).\boldsymbol{g^n}(s_{ij})}{\|\boldsymbol{g^n}(c_i)\|\|\boldsymbol{g^n}(s_{ij})\|}$$

where:

- $\boldsymbol{g^n}(c_i)$: is the TF-IDF vector for a candidate description
- $\boldsymbol{g^n}(s_{ij})$: is a TF-IDF vector for reference sentence j of image i.

Scores from different n-grams (up to 4) are arithmetic averaged to give the final CIDEr metric:

$$CIDE_r(c, S_i) = \frac{1}{4}\sum_{n=1}^{4} CIDE_{r_n}(c_i, S_i)$$

This indicator compensates for one of the disadvantages of BLEU, that is, all words on the match are treated the same, but in fact, some words should be more important.

2.2.5. SPICE (Semantic Propositional - based Image Caption Evaluation)

SPICE is a metric designed to assess image captioning models by emphasizing the **semantic content** of captions rather than just **n-gram overlaps**. Unlike other metrics such as BLEU, ROUGE, or CIDEr, which focus on syntactic similarity, SPICE seeks to evaluate the meaning conveyed by the generated captions in comparison to the reference captions.

It calculates the overlap between the captions produced by the model and the human-annotated captions in terms of recall, precision, and F1-score. Recall evaluates the proportion of relevant information in the human-annotated captions that is captured by the model's captions, while precision measures the relevance of the model-generated captions with respect to the reference captions. The F1 score combines both metrics by calculating the harmonic mean of recall and precision, providing a balanced evaluation.

3. Methodology

3.1. VGG16 + LSTM

This design system comprises four main processes: image and caption processing, image feature extraction utilizing the VGG16 model, caption generation

employing a Undirectional LSTM, and evaluation or scoring using the Greedy Search Argmax function. The system is depicted in Figure 4.

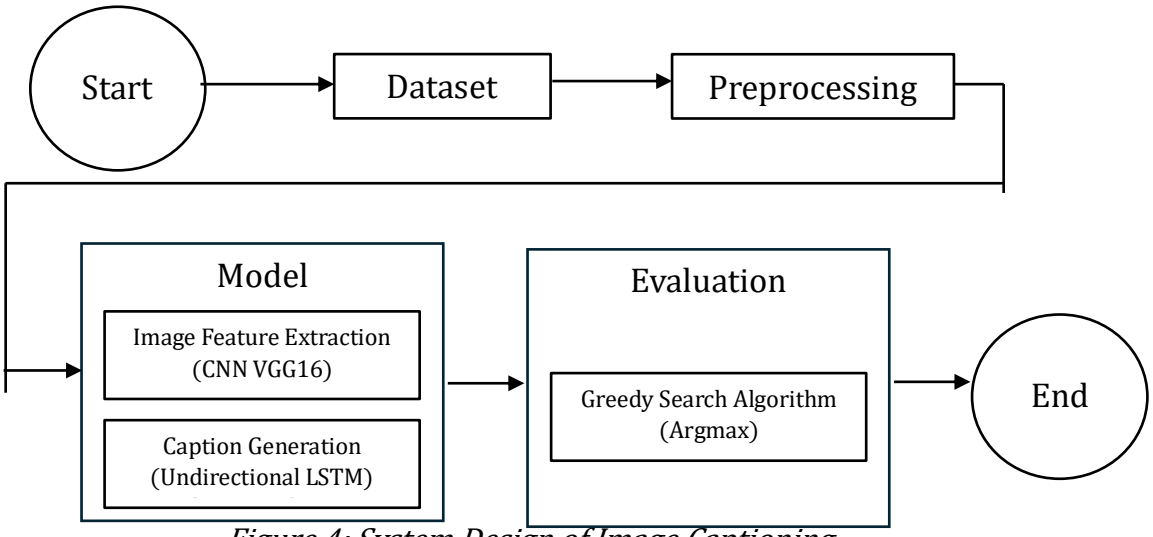

Figure 4: System Design of Image Captioning

a. Preprocessing on Image

Before training the model, the data must be prepared in a way that fits the model's input requirements. Here we use data set from CoCo 2017. The dataset consists of pairs of images and text, where the images need to be pre-processed to extract meaningful features. In this model, the images are processed through the VGG16 network to extract a 4096-dimensional feature vector. These features represent the image in a more compact form suitable for the model.

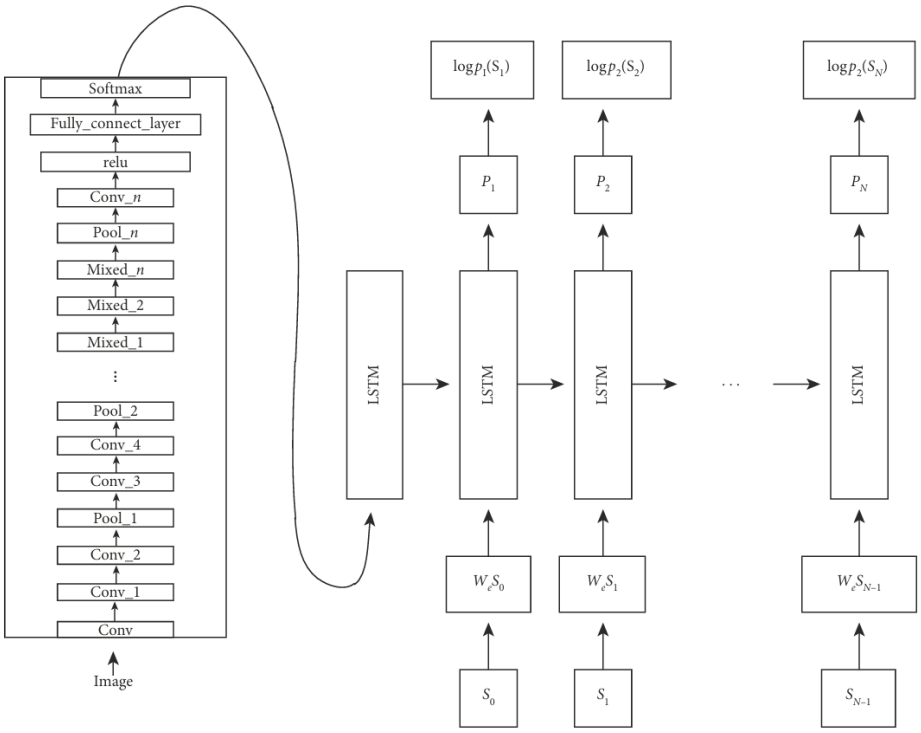

Figure 5: Training CNN and LSTM Flow Diagram

b. Preprocessing on Caption

The textual data, which corresponds to captions for the images, must be tokenized and padded to fit the model's input expectations. The *max_length* parameter using the *CoCo_train* dataset equal to 49 tokens, determines the length of the input text sequences, and each word in the sequence is converted into an integer token. These tokens are then passed through an embedding layer that converts them into dense vectors of fixed size. The preprocessing of both image and text ensures the data is in the correct format for input into the model.

Caption preprocessing involves cleaning the data by removing punctuation, converting all tokens to lowercase, eliminating tokens containing numbers, and separating tokens with spaces. Additionally, the token *"seqstart"* is added at the beginning of each description to signal the start of sentence generation, while the token *"seqend"* is appended at the end to indicate the stopping point. The system flow for determining the next word is illustrated in Table 3.

Table 3: The Next Sequence of Text and Word Input in the Sequence for the Training Phase

Input Sentence	Next Word
seqstart	an
seqstart, an	office
seqstart, an, office	with
seqstart, an, office, with	desk
seqstart, an, office, with, desk	computer
seqstart, an, office, with, desk, computer	endseq

c. Convolutional Neural Networks Model: Encoders

This section describes the Convolutional Neural Network (CNN) model utilized in the encoder process. CNN is a Deep Learning algorithm primarily designed for processing structured grid-like data, such as images. In the context of image captioning, the CNN functions as a feature extractor that processes input images to produce a feature vector, represented as a matrix containing pixel values aligned with the input image and applied filters.

For this report, the VGG16 encoder model is employed - one of the leading models in the field of computer vision. VGG16 is a widely recognized variant of the VGGNet architecture (Visual Geometry Group) which was trained on ImageNet datasets, developed by the Department of Science and Engineering at Oxford University.

The VGG16 network comprises a total of 16 layers, including 13 convolutional layers and 3 fully connected layers. For feature extraction, the input image dimensions must be 224 * 224 pixels. Each convolutional block layer uses a filter size of 3 × 3. The convolutional outputs are then passed to max pooling layers with dimensions of 2 * 2 pixels. The first to third convolutional blocks utilize a stride length of 2, while the fourth and fifth blocks apply a stride length of 1. The resulting output is then fed into a fully connected hidden layer consisting of 4096 units. The detailed architecture of the VGG16 model is illustrated in Figure 6.

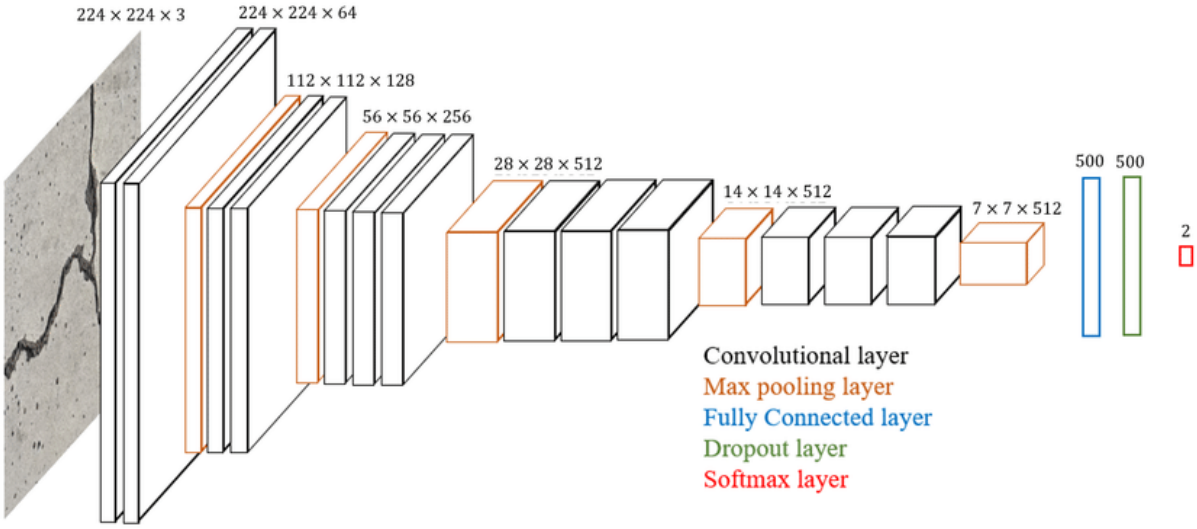

Figure 6: Model Architecture VGG16

d. Recurrent Neural Networks Model

Encoder Block

This is the inception of layers to retrieve sequences features which includes embedding layer, dropout layer and a structure of LSTM network. This encoder block receive input which is a sequence of texts with the length equal to 49 tokens. Each token in the sequence corresponds to an integer that represents a word or subword in the vocabulary. The input tokens are passed through an Embedding layer, which converts the integer tokens into dense vectors of size 256. The embedding layer learns a low-dimensional representation of the words in the sequence, which facilitates better modeling of semantic relationships between words. Next, we use dropout layer to reduce the overfitting. Finally, a structure of LSTM (Long short term memory) with hidden size = 256 is used to capture the sequential dependencies in the text. The LSTM processes the sequence of words and outputs a fixed-length vector that represents the textual information. This vector captures the context and meaning of the entire input text.

Decoder Block

The Decoder is responsible for combining the image and text features and generating the final output, which in this case is the next predicted word in the caption. The image and text features are combined using an Add layer, which performs an element-wise addition of the image feature vector and the textual feature vector. This combination allows the model to merge the visual and textual information, enabling it to generate a more accurate and context-aware caption.

After combining the features, a Dense layer with 256 units and a ReLU activation function is applied. This layer learns the interactions between the image and text features, helping the model understand the relationships between the visual and textual information. The final layer in the Decoder is another Dense layer with a number of units equal to the size of the vocabulary, vocab_size = 27551. This layer uses a Softmax activation function to produce a probability distribution over the vocabulary, which represents the likelihood of each word being the next word in the sequence.

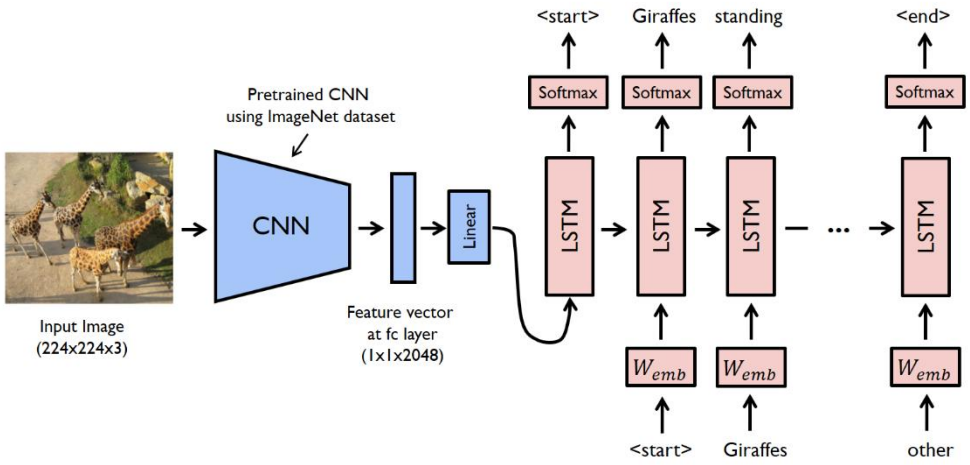

Figure 7: VGG16 + LSTM Model

e. Attention Mechanism

While the LSTM model is capable of generating brief text descriptions of an image, it often overlooks certain local and detailed information during the description process. This missing information is crucial for enhancing the richness and accuracy of the description. Additionally, the model faces challenges with the vanishing gradient problem when processing longer sentences. To address this issue, attention mechanisms were introduced. Originating from studies of human vision, attention is a complex cognitive ability in neuroscience that allows humans to focus on key information while filtering out less relevant details. This selective focus is referred to as attention. The attention mechanism was first applied to visual image classification using RNN models. For example, when humans read lengthy texts, their attention naturally gravitates toward keywords, events, or significant entities.

In neural networks, the attention mechanism enables the model to focus on specific subsets of inputs or features, thereby prioritizing essential information.

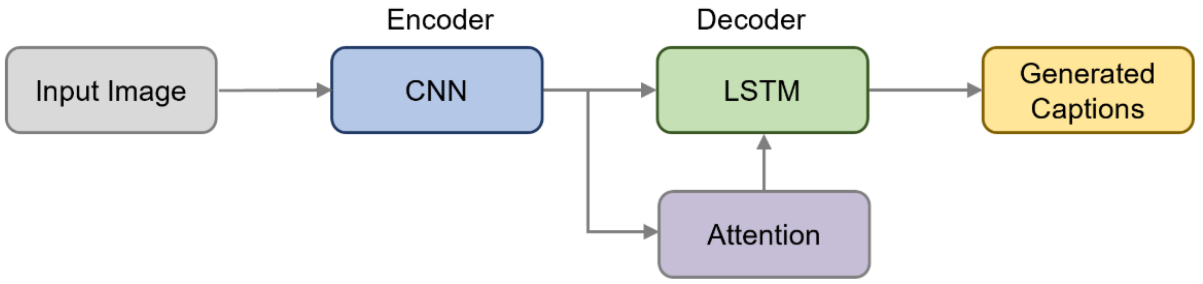

Figure 8: Encoder – Decoder with attention

The mechanism primarily involves two key aspects:

- Determining which parts of the input require focus.
- Allocating limited processing resources to the most important parts.

Currently, the mainstream attention mechanism is mathematically formulated using the equations (1) and (2). The core design links the target module (m_t) with the source module (m_s) through a defined function, followed by normalization to produce a probability distribution.

$$a_t = \text{align}(m_t, m_s) = \frac{\exp\left(f(m_t, m_s)\right)}{\sum_s \exp\left(f(m_t, m_s)\right)} \tag{1}$$

$$f(m_t, m_s) = f(x) = \begin{cases} m_t^T m_s, & \text{dot}, \\ m_t^T W_a m_s, & \text{general}, \\ W_a[m_t; m_s], & \text{concat}, \\ v_a^T \tanh(W_a m_t + U_a m_s), & \text{perception} \end{cases} \tag{2}$$

Building on the advantages of the attention mechanism discussed earlier, this chapter elaborates on two types of attention applied to LSTM for generating image descriptions.

Soft Attention

Dzmitry et al. [11] were the first to introduce the soft attention model and applied it to machine translation. The term "soft" refers to the probability distribution across the attention weights. For each word in the input sentence SSS, the probability distribution is determined based on the context vector ZtZ_tZt. Ultimately, a weighted sum of all regions is computed to obtain the final probability distribution.

$$E_{p(s_t|a)}[\widehat{z_t}] = \sum_{i=1}^{L} \alpha_{t,i}\, a_i \tag{3}$$

A derterministic attention model is formulated by computing a soft attention weighted attention vector:

$$\phi(\{a_i\},\{\alpha_i\}) = \sum_{i=1}^{L} \alpha_i a_i \tag{4}$$

The objective function can be written as follows:

$$L = -\log\big(P(y|x)\big) + \lambda \sum_{i}^{L} \left(1 - \sum_{t}^{C} \alpha_{t,i}\right)^2 \tag{5}$$

Soft attention is parameterized and therefore can be embedded for direct training. Gradient can be passed back through the attention mechanism module to other parts of the model.

Hard Attention

In contrast to the soft attention mechanism, which computes the weighted sum of all regions, hard attention focuses on a single location by randomly selecting a specific region. Instead of relying on the hidden states of the entire encoder, it samples the hidden state of the input based on probabilities. The context vector Z_t is calculated as follows [12]:

$$\begin{aligned} p\big(s_{t,i} = 1 \mid a\big) &= \alpha_{t,i}, \\ \widehat{z_t} &= \sum_{i=1}^{L} \alpha_i a_i \end{aligned} \tag{6}$$

where $s_{t,i}$ refers to whether to select the i-th position in the L feature maps:

$$s_{t,i} = \begin{cases} 1, & if\ selected \\ 0, & otherwise \end{cases}$$

To enable gradient backpropagation, Monte Carlo sampling is used to estimate the gradient of the module. A key disadvantage of hard attention is that the information is selected using maximum sampling or random sampling. As a result, the functional relationship between the final loss function and the attention distribution cannot be established, making it impossible to apply backpropagation for training.

Table 4: Comparision of attention mechanism modelling method

Attiontion name	Method	Comment
Soft Attention	Assign a probability based on the context vector for each word in the input sentence when determining the attention probability distribution.	Parameterization Derivative enable Definitely

Hard Attention	Focus exclusively on a randomly selected location by using Monte Carlo sampling to estimate the gradient.	Randomly On the base of probability Simple

f. Training Model

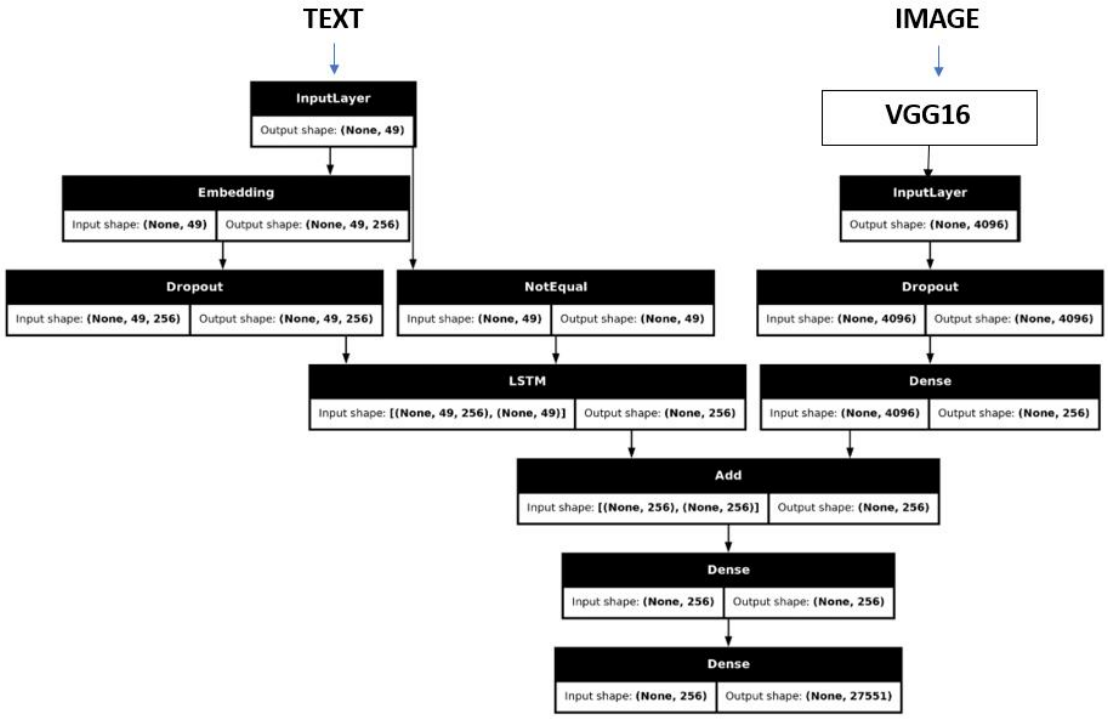

Figure 9: Training Model

The process of training the model involves several stages, including data preparation, model compilation, fitting the model, evaluation, and saving the trained model. This model is designed for image captioning tasks, where both image and textual data are combined to predict the next word in a caption based on the given image.

Each input pair, consisting of an image and a corresponding caption, has a corresponding output, which is a probability distribution over the vocabulary. The model aims to predict the next word in the sequence of the caption based on the input data.

Once the data is prepared, the model must be compiled. The compilation process involves selecting the loss function and optimizer. The loss function chosen for this model is *"categorical crossentropy"*, as the task is a multi-class classification

problem where the goal is to predict the correct word from a vocabulary. This loss function compares the predicted word distribution with the true distribution and helps minimize the error during training.

For optimization, the Adam optimizer is employed. Adam is a widely used optimization algorithm in deep learning due to its ability to adjust the learning rate during training, which allows the model to converge faster and more efficiently. The learning rate, defined as LEARNING_RATE = 0.001, controls the step size during the gradient descent process, and it must be set to a value that enables efficient training without causing instability.

With the model compiled, the next step is to fit the model to the training data. Fitting the model involves training it on both image and text data, allowing it to learn how to generate captions based on the provided inputs. The training process is done using the fit function, where the model will learn to map the input image and text features to the correct predicted word in the caption.

The data is fed into the model in batches, where the batch size is a crucial parameter that determines how many samples are processed together during each iteration. The batch size should be chosen based on the system's available memory and the dataset size. The model will undergo multiple epochs, with each epoch consisting of one complete pass through the training data. The number of epochs should be selected based on the model's performance; too few epochs may result in underfitting, while too many can cause overfitting. During training, the model's weights are updated to minimize the loss function, allowing it to improve its predictions over time.

As the model is trained, it will use both the pre-processed image features and the tokenized text sequences as inputs. The image features are processed through the VGG16 network, and the text data is passed through the embedding and LSTM layers. The output of the model is a probability distribution over the vocabulary for predicting the next word in the caption sequence. This output is compared to the true label, and the loss is computed to guide the optimization process.

3.2. Transformer

First introduced in *"Attention is All You Need"* by Vaswani et al. (2017) [15], the Transformer architecture revolutionized natural language processing (NLP). By replacing recurrent models with a Multi-Head Attention mechanism and an encoder-decoder structure it effectively addresses the vanishing gradient problem, which arises in conventional RNN and LSTM models while processing long text inputs [16] while provides the ability to process data in parallel which make the model more scalable, flexible, and efficient. Since then, it has extended its applications beyond

NLP and has soon become the 'state of art' when tackle various sequence-to-sequence tasks. In the case of image captioning, by encoding visual features and generating text descriptions, Transformers provide a powerful framework for bridging vision and language in deep learning systems.

a. Model Architecture

Like the previous proposed model, the Transformer can also be divided to 2 main parts, the encoder and the decoder. In transformer, these layers are built by stacking layers of identical blocks.

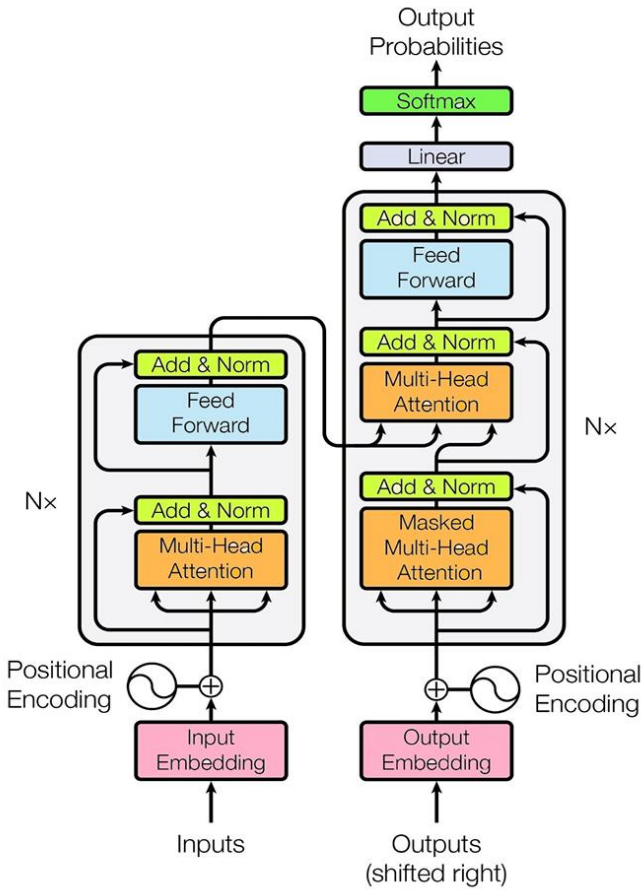

Figure 10: Transformer architecture
Source: Adapted from [15]

The encode is used to processes the input sequence, transforming it into a sequence of contextualized representation for the decoder. Before entering the first encoder's block, the data sequences will first be tokenized, which means, broken it up into small pieces. After then, each token underwent an embedding process that converted it into a vector encoded with that piece's meanings. As the transformer's main idea is to eliminate the recurring process, the embedded vectors must now also

provide the positional context of the token via positional encoding. After the embedding process is completed, embedded vectors are pushed through the encoder's blocks. These layer blocks contain two key sub-layers: the Multi-Head Self-Attention and the Feed-Forward Network (FFN). Attention is a mechanism allows the model to focus on different parts of the input simultaneously. It assigns relevance scores to each token in the sequence, helping the model determine which words or features are most important for a given context. This process is computed using two learnable vectors: Query (Q), Key (K), and a learnable matric Value (V). The attention score is calculated as the dot product of Q and K, scaled by the square root of the dimension, followed by a softmax operation. This score is used to weight the values (V), producing context-aware representations. This attention mechanism was written in a very compact ways in the original 'Attention is All you Need' paper [10] as follow

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

To prevent softmax function from saturating, which occurs when the input to softmax is too high (or too small), causing gradients to approach zero, the term QK^T will be divide by $\sqrt{d_k}$ to normalize.

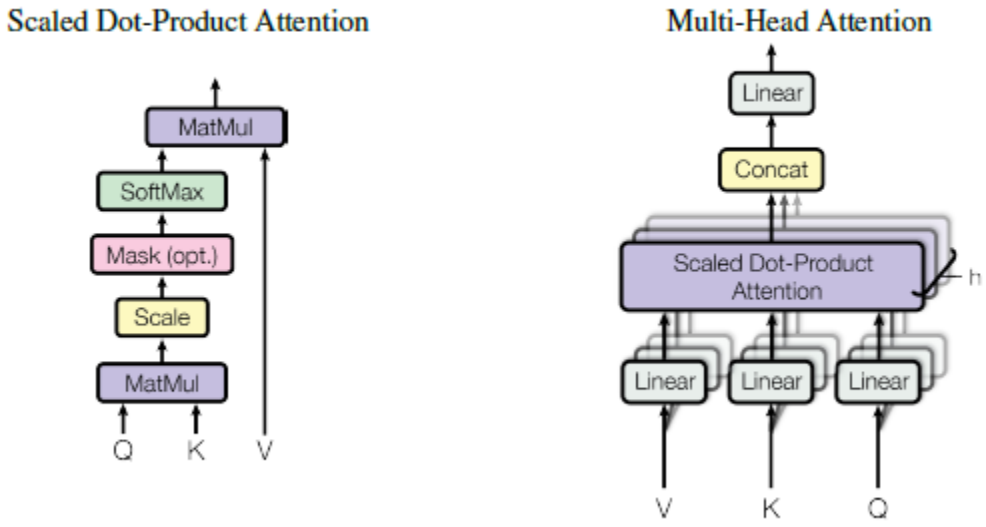

Figure 11: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention
Source: Adapted from [15]

Multi-head attention enhances the standard attention mechanism by running multiple attention processes, or "heads" in parallel. Each attention head

independently learns to focus on different parts of the input sequence or features, thereby capturing a variety of relationships and dependencies within the data. The outputs from all heads are subsequently concatenated and passed through a linear transformation, enabling the model to generate richer and more comprehensive contextual representations. This method empowers Transformers to effectively handle complex dependencies, particularly in tasks involving sequential data. Additionally, a fully connected neural network serves as the second sub-layer, refining the attention outputs to learn higher-level abstract features. Both sub-layers are followed by Layer Normalization and Residual Connections, which stabilize the training process and mitigate challenges like vanishing gradients.

For our image captioning task, we construct the model's encoder using the Vision Transformer (ViT), a pre-trained model that adapts the Transformer architecture for computer vision tasks. Unlike traditional convolutional neural networks (CNNs), ViT segments the input image into fixed-size patches, flattens them, and embeds these patches into linear vectors. These patch embeddings, combined with positional encodings, are fed into a standard Transformer encoder. By utilizing the self-attention mechanism, ViT effectively captures long-range dependencies and global context across the entire image. While ViT was initially designed as an encoder-only model for tasks like image classification and segmentation, in our project, we repurpose it as the encoder to produce

contextualized sequences from images, which are then passed to the decoder for generating captions.

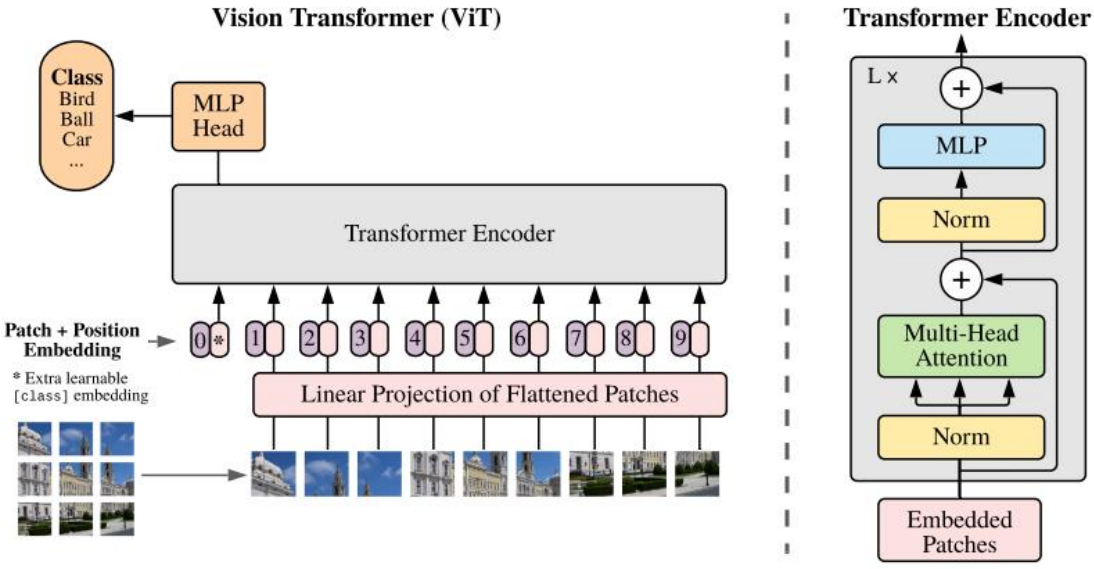

Figure 12: Overview of ViT
Source: Adapted from [17]

The decoder produces output sequences by attending to both the encoder's output and its previously generated tokens. Each decoder layer consists of three sub-layers: Masked Multi-Head Self-Attention, Multi-Head Cross-Attention, and a Feed-Forward Network (FFN). The Masked Multi-Head Self-Attention sub-layer functions similarly to the encoder's multi-head attention but applies masking to the lower triangular portion of the query-key matrix. This masking ensures that the decoder does not access future tokens during training, aligning with its design for autoregressive tasks. The Multi-Head Cross-Attention sub-layer aligns the decoder's current token with relevant features from the encoder output, enabling context-aware predictions (Fig. 12). Similar to the encoder, the decoder integrates a fully connected network to enhance the learning of deeper representations. Each sub-layer is followed by residual connections and layer normalization to stabilize training.

For our implementation, we experimented with two models. The first model was built from scratch, with the encoder based on the ViT architecture. For the decoder's tokenizer, we employed the pre-trained DistilBERT tokenizer, while the embeddings for both the encoder and decoder were trained alongside the model. Sinusoidal positional embeddings were chosen for the text sequence in the decoder.

To address resource limitations, we downscaled the model compared to the original ViT and DistilBERT architectures. Specifically, each FFN has a hidden size of 192, and the patch size was reduced to 8. Additionally, we implemented six encoder-decoder blocks, each with eight attention heads in the multi-head attention mechanisms.

The second model leverages Hugging Face's VisionEncoderDecoderModel framework, with ViT as the encoder and BERT as the decoder. In this report, we refer to this model as *Transformer-Pre*, where "Pre" denotes its reliance on pre-trained components. BERT (Bidirectional Encoder Representations from Transformers) is a Transformer-based, encoder-only language model that generates context-aware representations by considering both past and future tokens within a sequence. Its bidirectional design makes it highly effective for understanding contextual word meanings. In our implementation, ViT extracts visual features from images, while BERT generates descriptive text sequences. By combining ViT's visual feature extraction capabilities with BERT's language generation strengths, we aim to produce coherent and contextually accurate image captions.

b. Data preprocessing

The original COCO dataset includes approximately five captions per image. To prepare the data for model training, we first sampled a single caption for each image to ensure consistency. Next, we preprocessed the images by resizing them to meet the input size requirements of our models: 128x128 for our custom-built model and 224x224 for the Vision Encoder-Decoder model, as this is the standard image size used by the ViT model. For data augmentation, we applied Random Erasing with a 50% probability. Finally, the images were normalized using ImageNet mean and standard deviation values.

Regarding the captions, minimal preprocessing was required since we utilized Hugging Face's AutoTokenizer. The AutoTokenizer is a versatile tool capable of automatically loading the appropriate tokenizer for a given pre-trained model. By simply providing the text sequence and the name of the pre-trained tokenizer, it effectively manages tasks such as tokenization, padding, truncation, and special token insertion. This streamlined process ensured the captions were ready for input into the model without additional manual processing.

c. Training

The training process aims to develop and optimize two models for the image captioning task: a custom-built Transformer and a pre-trained ViT-BERT model. The custom model was designed from scratch to explore the fundamental principles of image captioning, while the ViT-BERT model leverages pre-trained architectures for enhanced performance. Both models were trained on the COCO 2017 dataset, with

the objective of generating meaningful and contextually accurate captions from input images. This section outlines the training setup, configurations, and strategies used to enhance the performance of both models.

As previously mentioned in the model design section, the custom-built model was trained from the ground up, while the ViT and BERT models were fine-tuned to work together. Since both ViT and BERT are encoder-only models, additional components were added to enable BERT to function as a decoder. Specifically, we encoporate a Multi-Head Cross-Attention mechanism in each decoder block, taking the encoder's output sequence and the masked output sequence from the decoder's Multi-Head Self-Attention layer as inputs. These newly added layers were trained alongside the rest of the model during fine-tuning.

For both models, we used nearly identical hyperparameters and training setups. The learning rate was set to $1 * 10^{-4}$, with the Adam optimizer and Cross-Entropy loss function. Dropout was applied in the feed-forward layers to prevent overfitting. Additionally, a custom Token Drop function was implemented, which randomly masked 50% of the tokens in the sequence with a placeholder token [unused01], acting similarly to a mask token for regularization purposes.

For the custom-built model, we used a batch size of 128 and trained the model for 100 epochs. Early stopping and a learning rate scheduler were applied. If the validation loss did not decrease for 10 consecutive epochs, the learning rate was reduced by 10%. If the validation loss remained stagnant for 15 epochs, early stopping was triggered, ending the training process. The model with the lowest validation loss was saved as the final version.

The ViT-BERT model's training process was similar but adjusted for its larger memory requirements. We used a batch size of 64 and trained for 60 epochs, focusing on fine-tuning rather than full training. For the first 15 epochs, the entire ViT encoder was frozen to reduce GPU memory usage and prevent catastrophic forgetting. Afterward, only the last two encoder blocks were unfrozen and trained until epoch 30. This approach allowed the model to retain pre-trained features while adapting to task-specific details such as object relationships and scene context. After 30 epochs, all encoder parameters were unfrozen to further enhance

performance. The model with the lowest validation loss was saved as the final version.

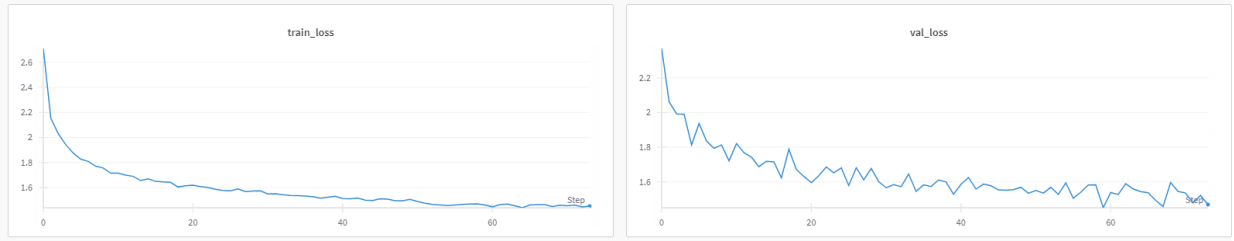

Figure 13: Train and Val Loss while training built-from-scratch model

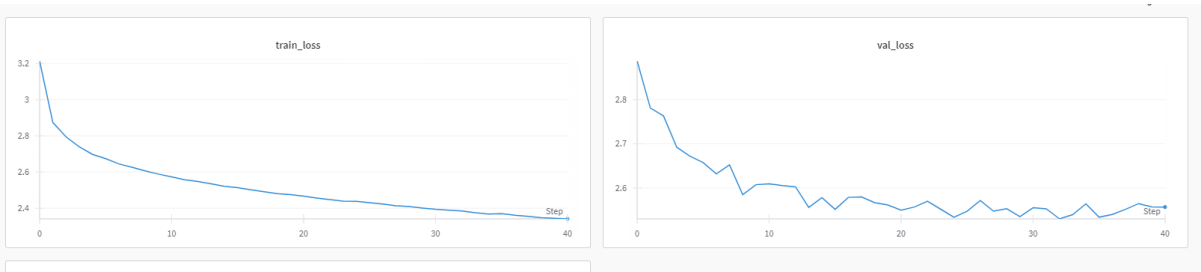

Figure 14 Train and Val Loss while training ViT-BERT model

4. Result

We utilize the **MS COCO 2017 validation dataset**, which contains 5,000 samples. Due to CPU limitations, the dataset is divided into five subsets, each consisting of 1,000 samples. The results are then averaged over five trials.

The generated captions are evaluated using several metrics, including BLEU, METEOR, ROUGE-L, CIDEr, and SPICE. These metrics measure the similarity between the generated captions and the reference captions, providing a comprehensive assessment of the model's performance.

The results of the evaluation are as follows:

Metric	VGG16 - LSTM	Transfomer - Pre	Transformer
Bleu_1	0.464	0.693	0.593
Bleu_2	0.275	0.521	0.403
Bleu_3	0.158	0.390	0.266
Bleu_4	0.088	0.295	0.178
METEOR	0.161	0.261	0.189
ROUGE_L	0.333	0.530	0.438
CIDEr	0.473	0.998	0.556
SPICE	0.127	0.190	0.117

As shown in the table, the Transformer-Pre model consistently outperformed both the VGG16-LSTM and the Transformer models across all evaluation metrics. This superior performance can be attributed to the pre-trained Transformer architecture's ability to capture long-range dependencies within images and leverage knowledge from a large corpus of text and image data. Notably, the Transformer-Pre model achieved significantly higher scores in CIDEr, ROUGE-L, and BLEU-4 metrics, indicating its superior ability to generate captions that are more semantically accurate, grammatically correct, and closely aligned with the image content. While the VGG16-LSTM model also produced reasonable captions, particularly in BLEU-1 and BLEU-2 scores, its overall performance was inferior to the Transformer-Pre model.

These results highlight the advantages of the Transformer architecture for image captioning. The attention mechanism in the Transformer models allows it to capture long-range dependencies within the image and generate captions that are more contextually relevant. In contrast, the LSTM model, while effective for sequential tasks, may struggle to capture global information in images due to its recurrent nature.

Overall, the Transformer-Pre model demonstrated superior performance in generating image captions compared to the VGG16-LSTM model and the Transformer model. This highlights the potential of Transformer-based models for advancing the state-of-the-art in image captioning.

5. Discussion and Conclusion

5.1. Discussion

The results presented in the previous section indicate a clear performance advantage of the Transformer-Pre model over the other two models in the image captioning task. Notably, the Transformer-Pre model had a BLEU-1 score of 0.693, while the VGG16-LSTM and Transformer models had scores of 0.464 and 0.593, respectively. The Transformer-Pre model also had a higher CIDEr score of 0.998 compared to the VGG16-LSTM model's score of 0.473 and the Transformer model's score of 0.556. The Transformer-Pre model's superior performance across all metrics suggests that it is better able to capture the complexity of the images and generate more accurate and relevant captions.

This observation can be attributed to several factors, primarily the architectural differences between the models and the advantage of pre-training. The Transformer-Pre model, based on the Transformer architecture, leverages the self-attention mechanism, which allows it to capture long-range dependencies within the image and generate captions that are more contextually relevant. In contrast, the VGG16-

LSTM model, while effective for sequential tasks, may struggle to capture global information in images due to its recurrent nature.

Furthermore, the Transformer-Pre model benefits from pre-training on a large corpus of text and image data. This pre-training allows the model to learn rich representations of language and visual concepts, which can be transferred to the image captioning task. Both the VGG16-LSTM and Transformer models are trained from scratch on the MS COCO dataset, which may limit their ability to generalize to unseen images.

Comparison with State-of-the-Art Models

While the Transformer-Pre model demonstrates strong performance on the MS COCO dataset, it is essential to place its results in the context of the broader field of image captioning. Several state-of-the-art models, such as mPLUG and OFA[18], have achieved even higher scores on the CIDEr metric, indicating their superior ability to generate captions that align with human consensus.

For example, mPLUG achieves the highest CIDEr score (1.551) among the models listed. Notably, mPLUG surpasses human performance on this metric, a significant achievement in the field. OFA follows closely with a CIDEr score of 1.549, demonstrating its effectiveness as a unified architecture for image captioning.

These models incorporate advanced techniques, such as cross-modal skip-connections and unified architectures, to achieve their impressive performance.[14]

The Transformer-Pre model, while not surpassing the absolute top performers, still demonstrates the potential of Transformer-based models for advancing the state-of-the-art in image captioning. Future work could explore incorporating techniques from other state-of-the-art models, such as cross-modal skip-connections or more advanced pre-training strategies, to further improve the performance of the Transformer-Pre model.

5.2. Conclusion

In this report, we evaluated the performance of three image captioning models, VGG16-LSTM, Transformer and Transformer-Pre, on the MS COCO 2017 dataset. Our findings demonstrate that the Transformer-Pre model consistently outperforms the other two models across all evaluation metrics. This superior performance can be attributed to the Transformer architecture's ability to capture long-range dependencies within images and the benefits of pre-training on a large corpus of text and image data.

While the Transformer-Pre model shows promising results, it is essential to acknowledge that several state-of-the-art models achieve even higher performance on the MS COCO dataset. These models incorporate advanced techniques, such as

cross-modal skip-connections and unified architectures, which contribute to their superior performance.

Future work could explore incorporating these techniques into the Transformer models to further improve their performance. Additionally, we will implement attention mechanism and bidirectional instead of using undirectional LSTM to enhance performance and address shortcomings of model. Besides, we can implement some another model architecture like VGG16 – BiGRU, Up – Down Attention Model which demonstrate one of SOTA current model.

Overall, this project highlights the potential of Transformer-based models for advancing the state-of-the-art in image captioning. As research in this field continues, we can expect to see even more sophisticated models that can generate highly accurate and human-like captions for images

6. References

[1] S. Li, G. Kulkarni, T. L. Berg, and Y. Choi, "Composing simple image descriptions using web-scale N-grams" in Proceeding of Fifteenth Conference on Computational Natural Language Learning, pp. 220–228, Association for Computational Lin guistics, Portland, OR, USA, June 2011.

[2] D. Lin, C. Kong, S. Fidler, and R. Urtasun, "Generating multi sentence lingual descriptions of indoor scenes" pp. 2333–9721, Computer Science, 2015, http://arxiv.org/abs/1503.00064.

[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, "Microsoft COCO: Common Objects in Context". https://arxiv.org/abs/1405.0312

[4] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation", IBM T. J. Watson Research Center Yorktown Heights, NY 10598, USA.

[5] "METEOR," Wikipedia, 2024, https://en.wikipedia.org/wiki/METEOR.

[6] "The METEOR Metric: An NLP Classic," The Deep Hub, Medium, Sep. 22, 2023.https://medium.com/thedeephub/the-meteor-metric-an-nlp-classic-42552cb6ce69.

[7] "Two minutes NLP: Learn the ROUGE metric by examples," NL Planet, Medium, Aug. 16, 2021, https://medium.com/nlplanet/two-minutes-nlp-learn-the-rouge-metric-by-examples-f179cc285499.

[8] Ramakrishna Vedantam1 , C. Lawrence Zitnick2 Devi Parikh1, "CIDEr: Consensus-based Image Description Evaluation", https://arxiv.org/pdf/1607.08822

[9] Peter Anderson, Basura Fernando, Mark Johnson, Stephen Gould, "SPICE: Semantic Propositional Image Caption Evaluation", https://arxiv.org/abs/1607.08822
[10] "Convolutional Network – VGG16", https://www.jasonosajima.com/convnets_vgg.html
[11] B. Dzmitry, K. Cho, and Y. Bengio, "Neural machine trans lation by jointly learning to align and translate," Computer Science, 2014, http://arxiv.org/abs/1409.0473.
[12] K. Xu, J. Ba, K. Ryan et al., "Show, attend and tell: neural image caption generation with visual attention," in Pro ceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2048–2057, Boston, MA, USA, June 2015
[13] Zhibin Guan, Kang Liu, Yan Ma , Xu Qian and Tongkai Ji, "Sequential Dual Attention: Coarse-to-Fine-Grained Hierarchical Generation for Image Captioning"
[14] Haoran Wang, YueZhang, and Xiaosheng Yu, "An Overview of Image Caption Generation Methods", College of Information Science and Engineering, Northeastern University, China
[15] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need.(Cornell University), 30, 5998–6008.
[16] Dandwate, P., Shahane, C., Jagtap, V., & Karande, S. C. (2023, March 5). Comparative study of Transformer and LSTM Network with attention mechanism on Image Captioning
[17] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020, October 22). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*.
[18] Transparent Human Evaluation for Image Captioning - ACL Anthology, accessed December 15, 2024, https://aclanthology.org/2022.naacl-main.254.pdf
[19] COCO Captions Benchmark (Image Captioning) | Papers With Code, accessed December 15, 2024, https://paperswithcode.com/sota/image-captioning-on-coco-captions?metric=CIDER
[20] HackersRealm. (2022, April 27). Image caption generator using Python. HackersRealm. https://www.hackersrealm.net/post/image-caption-generator-using-python