

Bengali Image Captioning Using Deep Learning Methods

A thesis

Submitted in partial fulfillment of the requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Submitted by

Aurnab Barua	190204009
Md. Azmain Mahtab	190204032
Md. Sarwar Hossain	190204049

Supervised by

Mr. Md. Masudur Rahman



Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Dhaka, Bangladesh

April 2024

CANDIDATES' DECLARATION

We, hereby, declare that the thesis presented in this report is the outcome of the investigation performed by us under the supervision of Mr. Md. Masudur Rahman, Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh. The work was spread over two final year courses, CSE400: Project and Thesis I and CSE450: Project and Thesis II, in accordance with the course curriculum of the Department for the Bachelor of Science in Computer Science and Engineering program.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Aurnab Barua
190204009

Md. Azmain Mahtab
190204032

Md. Sarwar Hossain
190204049

CERTIFICATION

This thesis titled, “**Bengali Image Captioning Using Deep Learning Methods**”, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in April 2024.

Group Members:

Aurnab Barua	190204009
Md. Azmain Mahtab	190204032
Md. Sarwar Hossain	190204049

Mr. Md. Masudur Rahman
Lecturer & Supervisor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Dr. Md. Shahriar Mahbub
Professor & Head
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this thesis. A special gratitude we give to our supervisor, Mr. Md. Masudur Rahman, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this thesis.

Furthermore, we are thankful for the companionship and teamwork of us.

We would also like to acknowledge the Computer Science and Engineering department for providing the necessary facilities for our research.

Last but not least, we would like to thank our families for their understanding and endless love, through the duration of our studies.

Dhaka
April 2024

Aurnab Barua
Md. Azmain Mahtab
Md. Sarwar Hossain

ABSTRACT

The task of image captioning plays a crucial role in the understanding of vision and language, where in, a model predicts a descriptive caption for a given input image. In this research, we aimed to leverage the potential of large language models (LLM) in Bangla image captioning, which have lately shown promising results in English language caption generation. In our proposed method, we utilized CLIP encoding as a prefix to the caption by employing a mapping network, followed by fine-tuning a language model to generate the image captions. Our main concept revolves around combining the pre-trained language model (GPT2) with CLIP encoding to achieve a comprehensive understanding of both visual and textual data. Furthermore, We explored vision transformer based encoders such as ViT, Swin along with Bangla pre-trained BERT language models as decoders, which have lately performed well in NLP tasks. In addition to BERT decoders, the recently published Bangla pre-trained GPT-2 language model was used to generate captions. We compared the traditional CNN-Transformer-based encoder-decoder approach along with our proposed approaches. The performances were evaluated using BLEU and Meteor evaluation metrics. We used BanglaLekha, the largest indigenous dataset for Bengali image captioning, alongside BNature. Furthermore, we constructed and utilized our own dataset to cover gaps in authentic Bangla datasets. The proposed CLIP based models and the vision encoder-decoder models outperformed the current benchmark results, with BLEU-1, BLEU-2, BLEU-3, BLEU-4, and Meteor scores of 0.688, 0.624, 0.574, 0.529, and 0.380 on the BanglaLekha Dataset and 0.818, 0.755, 0.702, 0.655, and 0.4 on the BNature Dataset.

Contents

<i>CANDIDATES' DECLARATION</i>	i
<i>CERTIFICATION</i>	ii
<i>ACKNOWLEDGEMENT</i>	iii
<i>ABSTRACT</i>	iv
List of Figures	viii
List of Tables	ix
1 Introduction	1
1.1 Overview	1
1.2 Image Captioning Description	1
1.3 Motivation	2
1.4 Objective	2
2 Background Study	3
2.1 Contrastive Language-Image Pre-training (CLIP)	3
2.1.1 CLIP Training Process	4
2.2 Multilayer Perceptron (MLP) neural network	5
2.2.1 Structure of Multilayer Perceptron Neural Network :	5
2.2.2 Working of Multilayer Perceptron Neural Network :	6
2.3 Neural Network	7
2.4 Convolutional Neural Network	7
2.5 Transformer	9
2.5.1 The Transformer architecture	10
2.5.2 Scaled dot-product attention	10
2.5.3 Multi-head attention	10
2.5.4 Positionally encoding different words	12
2.5.5 Transformer challenges	12
2.6 ResNet-50	13
2.6.1 ResNet-50 Architecture	13

2.6.2	Key Features of ResNet-50	14
2.7	Vision Transformer (ViT)	15
2.7.1	Swin Transformer	15
2.7.2	Vision Transformer and Image Classification	17
2.7.3	Vision Transformer ViT Architecture	17
2.7.4	Working Procedure of A Vision Transformer (ViT)	19
2.8	Pretrained Models:	19
2.8.1	Generative Pre-trained Transformer 2 (GPT-2) Based:	19
2.8.2	Electra-Based:	20
2.8.3	Bidirectional Encoder Representations from Transformers (BERT)- Based:	21
2.9	Vision Encoder-Decoder Model	22
2.10	Performance Metrics	22
2.10.1	BLEU	23
2.10.2	METEOR	23
3	Literature Review	25
3.1	Paper Reviews	25
3.1.1	Chittron: An Automatic Bangla Image Captioning Systemcite [1]	25
3.1.2	BORNON: BENGALI IMAGE CAPTIONING WITH TRANSFORMER- BASED DEEP LEARNING APPROACH [2]	26
3.1.3	TextMage: The Automated Bangla Caption Generator Based On Deep Learning [3]	26
3.1.4	Bangla Image Captioning through Transformer-based Encoder - Decoder [4]	27
3.1.5	Improved Bengali Image Captioning via deep convolutional Neural Network Based Encoder-Decoder Model [5]	28
3.1.6	Task-Adaptive Attention for Image Captioning [6]	29
3.1.7	CPTR: FULL TRANSFORMER NETWORK FOR IMAGE CAPTION- ING [7]	30
3.1.8	TrOCR: Transformer-based Optical Character Recognition with Pre- trained Models [8]	31
3.1.9	A Visual Attention-Based Model for Bengali Image Captioning [9]	32
3.1.10	Amharic Language Image Captions Generation Using Hybridized Attention-Based Deep Neural Networks [10]	33
3.2	Research Gap	35
4	Dataset	36
4.1	BanglaLekha Dataset	36
4.2	BNature Dataset	38

4.3	Our Dataset	38
5	Methodology	40
5.1	Dataset Preprocessing	40
5.2	Our Approaches	41
5.2.1	Contrastive Language–Image Pre-training (CLIP) Based Model . . .	41
5.2.2	Vision Encoder-Decoder Based Model	43
6	Experimental Results and Analysis	45
6.1	Experimental Setups	45
6.1.1	Implementation Details	45
6.1.2	Hyperparameter selection	46
6.2	Experimental Results	48
6.2.1	Experimental Results of the Clip Based Model	48
6.2.2	Experimental Results of the Vision Encoder-Decoder Model	50
6.2.3	Experimental Results of the Transformer Based Model	52
6.2.4	Comparison of Existing Models and Our Proposed Models	53
6.3	Summary	54
7	Conclusion and Future Work	58
7.1	Conclusion	58
7.2	Future Work	58
	References	59

List of Figures

2.1	The CLIP training architecture.	4
2.2	Illustration of a Convolutional Neural Network [11]	9
2.3	Transformer Model Architecture [12]	11
2.4	Multi-Head Attention [12]	12
2.5	Resnet Architecture [13]	14
2.6	Swin Transformer Architecture [14]	16
2.7	Vision Transformer ViT Architecture [15]	18
2.8	Performance benchmark comparison of Vision Transformers (ViT) with ResNet and MobileNet when trained from scratch on ImageNet [16]	18
2.9	BERT architecture [17]	22
4.1	Illustration of some images of the BanglaLekha dataset along with their two Bengali captions.	37
4.2	Illustration of some images of the BNature dataset along with their two Bengali captions.	38
4.3	Illustration of some images of our dataset along with their two Bengali captions.	39
5.1	Overview of the Clip based model.	41
5.2	Proposed Clip based model.	42
5.3	Caption generation using Clip based model.	43
5.4	Proposed Vision Encoder-Decoder Method.	43
6.1	Validation accuracy plot for Clip-ViT over 20 epochs.	47
6.2	Validation loss plot for Clip-ViT over 20 epochs.	47
6.3	Illustration of captions generated by best performing CLIP Based Architecture using BanglaLekha, BNature and our datasets.	55
6.4	Illustration of captions generated by best performing Vision Encoder-Decoder Based Architecture using BanglaLekha dataset.	56
6.5	Illustration of captions generated by best performing Vision Encoder-Decoder Based Architecture using BNature and our dataset.	57

List of Tables

5.1	Dataset Splitting Ratio	40
6.1	Results for BanglaLekha Dataset on CLIP Based Model	48
6.2	Results for BNature Dataset on CLIP Based Model	49
6.3	Results for Our Dataset on CLIP Based Model	49
6.4	Results for Different Beam Sizes on CLIP Based Model	49
6.5	Results on BanglaLekha Dataset on Vision Encoder-Decoder Based Model . .	50
6.6	Results on BNature Dataset on Vision Encoder-Decoder Based Model	51
6.7	Results on Our Dataset on Vision Encoder-Decoder Based Model	51
6.8	Results on BanglaLekha Dataset on Vision Encoder-Decoder Based Model . .	51
6.9	Results on BanglaLekha Dataset on Transformer Based Model	52
6.10	Results on BNature Dataset on Transformer Based Model	52
6.11	Results on Our Dataset on Transformer Based Model	52
6.12	A brief comparison of BLEU and Meteor scores for existing models and our proposed models using the BanglaLekha dataset.	53
6.13	A brief comparison of BLEU and Meteor scores for existing models and our proposed models using the BNature dataset.	54

Chapter 1

Introduction

1.1 Overview

Image captioning has become a popular subject among deep learning, computer vision, and natural language learning communities. It falls within the realm of artificial intelligence and utilizes machine learning techniques. The goal is to generate text that describes the content of an image. Recently, there have been advancements in using transformer-based models to encode and decode images into text in Bengali language image captioning. The traditional encoder-decoder method has made great progress, but its potential remains limited compared to the latest state-of-the-art models used in image captioning. In our study, we suggest a method that employs the CLIP(a multi-modal vision and language model) and GPT-2(a large language model) pre-trained models. It is a much faster and simpler method compared to the latest image captioning models while working with large data and also matching those in performance.

1.2 Image Captioning Description

Creating textual descriptions for images is known as image captioning. An image can consist of various elements such as scenes, objects, humans, animals, plants, and more. The goal of image captioning is to describe the contents of the image and their relationships. Most image captioning systems use an encoder-decoder framework, where the input image is encoded into a representation of its information and then decoded into a descriptive text sequence. Image captioning is now applied in various fields, including editing applications, aid for the visually impaired, media and publishing houses, and social media posts.

1.3 Motivation

Image captioning can be utilized in various ways, aiding the visually impaired by converting the visual environment into text and subsequently transforming it into speech for their comprehension. An example of its recent application is enhancing self-driving cars, enabling them to accurately describe their surroundings. CCTV cameras, now essential for security purposes, can also benefit from this technology as it generates relevant captions to identify any potential malicious activities. Moreover, employing image captioning in Bengali can greatly assist Bengali speakers in comprehending images effectively through the provided captions.

1.4 Objective

- Our main goal is to implement the Contrastive Language-Image Pre-training (CLIP) model for Bengali image captioning.
- Explore the use of Bangla Pretrained Large Language Models (LLM) for generating the captions in Bangla.
- Build a large dataset containing strictly sample images showcasing Bangladeshi culture to fill up the void of inadequate Bangla datasets.
- Further explore vision encoder decoder models for comparative analysis.
- Surpass the current state-of-the-art performance metrics for Bangla image captioning on publicly available Bangla Datasets.

Chapter 2

Background Study

2.1 Contrastive Language-Image Pre-training (CLIP)

Contrastive Language-Image Pre-training (CLIP) ¹ is a multimodal learning architecture developed by OpenAI. It learns how to recognize things in pictures based on descriptions in words. It bridges the gap between text and visual data by jointly training a model on a large-scale dataset containing images and their corresponding textual descriptions. Comparable to the zero-shot capabilities seen in GPT-2 and GPT-3, CLIP employs a dual-encoder architecture, which aligns images and text within a shared latent space. This is accomplished through the training of two encoders: one specialized in processing images (utilizing Vision Transformer), and the other in handling text (based on Transformer architecture).

Image Encoder:

The image encoder functions to identify important features within visual input. It accepts an image as its input and generates a high-dimensional vector representation. Typically, this encoder utilizes a convolutional neural network (CNN) structure, such as ResNet, to extract significant image characteristics.

Text Encoder:

The text encoder is responsible for capturing the semantic essence of the accompanying textual description. It receives a 'text caption/label' as input and generates a separate high-dimensional vector representation. Frequently, this encoder employs a transformer-based architecture, such as Transformer or BERT, to handle text sequences.

¹<https://viso.ai/deep-learning/clip-machine-learning/>

Shared Embedding Space:

The two encoders produce embeddings in a shared vector space. These shared embedding spaces allow CLIP to compare text and image representations and learn their underlying relationships.

2.1.1 CLIP Training Process

Contrastive Language-Image Pre-training (CLIP) is a multimodal learning architecture developed by OpenAI. It learns visual concepts from natural language supervision. It closes the divide between text and visual data by training a model on a vast dataset containing images and their associated textual descriptions. This approach mirrors the zero-shot capabilities observed in GPT-2 and GPT-3. CLIP uses a dual-encoder architecture to map images and text into a shared latent space. It works by jointly training two encoders. One encoder for images (Vision Transformer) and one for text (Transformer-based language model).

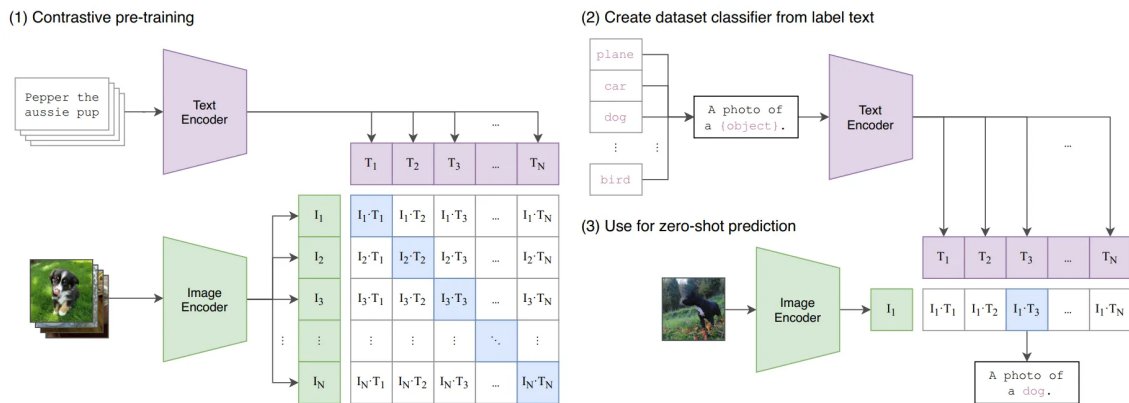


Figure 2.1: The CLIP training architecture.

Contrastive Pre-training:

CLIP is pre-trained on a large-scale dataset of 400 million (image, text data) pairs collected from the internet. In the pre-training phase, the model is exposed to pairs of images and text captions. Among these pairs, some accurately match (where the caption precisely describes the image), while others are mismatched. Through this process, shared latent space embeddings are formed.

Create Dataset Classifiers from Label Text:

Multiple text descriptions are generated for each image, encompassing both the correct description and several incorrect ones. This results in a blend of positive samples (matching pairs) and negative samples (mismatched pairs). These descriptions are fed into the text encoder, generating class-specific embeddings.

Zero-shot Prediction:

Now, the trained text encoder is used as a zero-shot classifier. With a new image, CLIP can make zero-shot predictions. CLIP can make predictions without further fine-tuning by passing it through the image encoder and the dataset classifier. CLIP calculates the cosine similarity between the embeddings of all image and text description pairs. It adjusts the parameters of the encoders to enhance the similarity of the correct pairs while reducing the similarity of the incorrect pairs. This way, CLIP learns a multimodal embedding space where semantically related images and texts are mapped close to each other. The predicted class is the one with the highest logit value.

2.2 Multilayer Perceptron (MLP) neural network

An MLP² is a type of feedforward artificial neural network with multiple layers, including an input layer, one or more hidden layers, and an output layer. Each layer is fully connected to the next. In this article, we will understand MultiLayer Perceptron Neural Network, an important concept of deep learning and neural networks. The Multilayer Perceptron (MLP) Neural Network works only in the forward direction. All nodes are fully connected to the network. Each node passes its value to the coming node only in the forward direction. The MLP neural network uses a Backpropagation algorithm to increase the accuracy of the training model.

2.2.1 Structure of Multilayer Perceptron Neural Network :

This network has three main layers that combine to form a complete Artificial Neural Network. These layers are as follows:

²www.shiksha.com

Input Layer :

It is the initial or starting layer of the Multilayer perceptron. It receives input directly from the training dataset and passes it on to the hidden layer. The input layer comprises n input nodes, with the number of nodes determined by the features present in the dataset. Each input node corresponds to a feature, and the input values are distributed across these nodes within the hidden layer.

Hidden Layer :

The hidden layer is essentially the core of all Artificial Neural Networks (ANNs). It executes all computations within the neural network. The connections between the nodes in the hidden layer have associated weights, which are multiplied by the values of the nodes. This layer uses the activation function.

There can be one or two hidden layers in the model.

Several hidden layer nodes should be accurate as few nodes in the hidden layer make the model unable to work efficiently with complex data. More nodes will result in an overfitting problem.

Output Layer :

This layer provides the estimated output of the neural network. The number of nodes in the output layer is determined by the nature of the problem being addressed. For tasks involving a single target variable, such as regression, typically one node is used in the output layer. In classification problems with N classes, the neural network employs N nodes in the output layer, with each node representing a class and producing a probability or score indicating the likelihood of the input belonging to that class.

2.2.2 Working of Multilayer Perceptron Neural Network :

- The input node represents the feature of the dataset.
- Each input node passes the vector input value to the hidden layer.
- In the hidden layer, each edge has some weight multiplied by the input variable. All the production values from the hidden nodes are summed together. To generate the output.
- The activation function is used in the hidden layer to identify the active nodes.

- The output is passed to the output layer.
- Calculate the difference between predicted and actual output at the output layer.
- The model uses backpropagation after calculating the predicted output.

2.3 Neural Network

Neural networks³, also known as artificial neural networks (ANNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the signaling process among biological neurons. These networks are composed of layers of nodes, including an input layer, one or more hidden layers, and an output layer. Each node, akin to an artificial neuron, is interconnected with others and carries an associated weight and threshold. When the output of an individual node surpasses the specified threshold, it becomes activated, transmitting data to the subsequent layer of the network. Conversely, if the output fails to meet the threshold, no data is forwarded to the next layer.

2.4 Convolutional Neural Network

Convolutional Neural Network(CNN)⁴ is a type of deep learning architecture specifically designed for processing visual data, such as images. CNNs excel at recognizing and classifying specific features within images and are extensively employed for tasks involving image analysis. The network achieves this by assigning priorities, represented by weights and biases, to different objects or features present in an image, allowing it to differentiate between them effectively. When these layers are stacked, a CNN architecture will be formed as shown in figure 2.2. In addition to these three layers, there are two more important parameters which are the dropout layer and the activation function which are defined below.

- **Convolutional Layer**

The Convolutional Layer serves as the initial stage in extracting diverse features from input images. Here, convolution, a mathematical operation, occurs between the input image and a filter of size $M \times M$. As the filter slides across the input image, it computes the dot product with corresponding portions of the image based on the filter size. The result is termed a Feature Map, providing insights into image characteristics like edges

³<https://www.ibm.com/cloud/learn/neural-networks>

⁴<https://www.upgrad.com/blog/basic-cnn-architecture>

and corners. Subsequently, this Feature Map is forwarded to subsequent layers for learning additional features of the input image.

- **Pooling Layer**

Typically, a Convolutional Layer is succeeded by a Pooling Layer in most cases. The main purpose of this layer is to reduce the size of the convolved feature map, thereby cutting down computational expenses. This reduction is achieved by minimizing connections between layers and individually processing each feature map. Depending on the method employed, various Pooling operations exist. In Max Pooling, the largest element from the feature map is selected, while Average Pooling computes the average of elements within a predefined image section. Alternatively, Sum Pooling calculates the total sum of elements within the specified section. Generally, the Pooling Layer acts as a link between the Convolutional Layer and the Fully Connected (FC) Layer.

- **Fully Connected Layer**

The Fully Connected (FC) layer incorporates weights, biases, and neurons to establish connections between neurons across different layers. Typically positioned before the output layer, these layers constitute the final stages of a CNN Architecture. In this configuration, the input image is flattened and transmitted to the FC layer from the preceding layers. The flattened vector then undergoes few more FC layers where the mathematical function's operations usually take place. In this stage, the classification process begins to take place.

- **Dropout**

Usually, when all the features are connected to the FC layer, it can cause overfitting in the training dataset. Overfitting arises when a model performs exceptionally well on the training data but struggles with new data. To address this issue, a dropout layer is employed. During the training phase, some neurons are randomly omitted from the neural network, thereby reducing the model's size. On passing a dropout of 0.3, 30% of the nodes are dropped out randomly from the neural network.

- **Activation Functions**

Activation functions play a pivotal role in CNN models, influencing their performance significantly. They facilitate the learning and approximation of intricate relationships among network variables. In essence, activation functions determine which information within the model should be activated in the forward direction and which should not, thereby introducing non-linearity. Commonly utilized activation functions include ReLU, Softmax, tanH, and Sigmoid functions, each serving specific purposes. For instance, in binary classification CNN models, sigmoid and softmax functions are often favored, while softmax is typically employed for multi-class classification tasks.

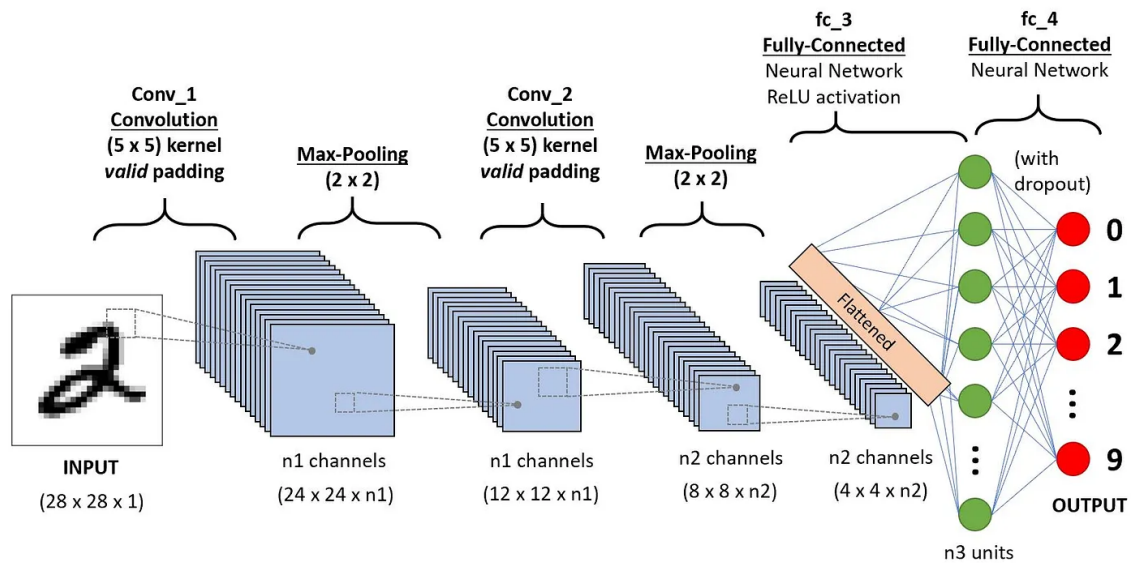


Figure 2.2: Illustration of a Convolutional Neural Network [11]

2.5 Transformer

Transformer [12] is a deep learning (DL) model, based on a self-attention mechanism that weights the importance of each part of the input data differently. It is mainly used in computer vision (CV) and natural language processing (NLP).

Transformers, akin to recurrent neural networks (RNNs), are tailored to handle sequential input data such as natural language, undertaking tasks like text summarization and translation. Nonetheless, transformers differ from RNNs in that they process the entire input simultaneously. Through the attention mechanism, the model can concentrate on the most pertinent segments of the input for each output.

For example, when dealing with input data like natural language sentences, a translator using transformers doesn't have to handle each word individually. This enables parallel processing to a greater extent compared to RNNs, resulting in reduced training duration. Consequently, this advancement has led to the creation of extensive pre-trained systems like Bidirectional Encoder Representations from Transformers (BERT) and Generalized Pre-trained Transformers (GPT).

Introduced by the Google Brain team in 2017, the Transformer architecture is increasingly becoming a model of choice for NLP problems, replacing RNN models such as long short-term memory (LSTM)

2.5.1 The Transformer architecture

The Transformer architecture adopts an encoder-decoder design that operates independently of recurrence and convolutions for output generation. The encoder transforms an input sequence into a sequence of continuous representations. Subsequently, the decoder takes the encoder's output and its own output from the previous time step to produce an output sequence.

Here is an image that visualizes the architecture: Since it is impossible to use strings directly, the architecture first converts the input data into an n -dimensional embedding, which is then fed to an encoder. The encoder and decoder consist of modules stacked on each other several times (represented as $N \times$ in the image). The modules include mainly feed-forward and multi-head attention layers.

Here is an image that shows the multi-head attention bricks in the model:

2.5.2 Scaled dot-product attention

The following equation describes the left part of the attention mechanism in the above image:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.1)$$

Here is an explanation of the equation:

- Q represents a matrix containing the Query.
- K consists of all Keys, the vector representations of all the words in the sequence.
- V represents the Values, which is often the same as K .

The V values are multiplied and summed with attention-weights a . The weights are defined using the following formula:

$$a = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (2.2)$$

The Softmax function normalizes the value of a to a scale of 0 to 1. Next, the model applies these weights to all the words in value V .

2.5.3 Multi-head attention

The illustration above illustrates the parallelization of the attention mechanism. Through the multi-head attention mechanism, the model can simultaneously focus on various aspects

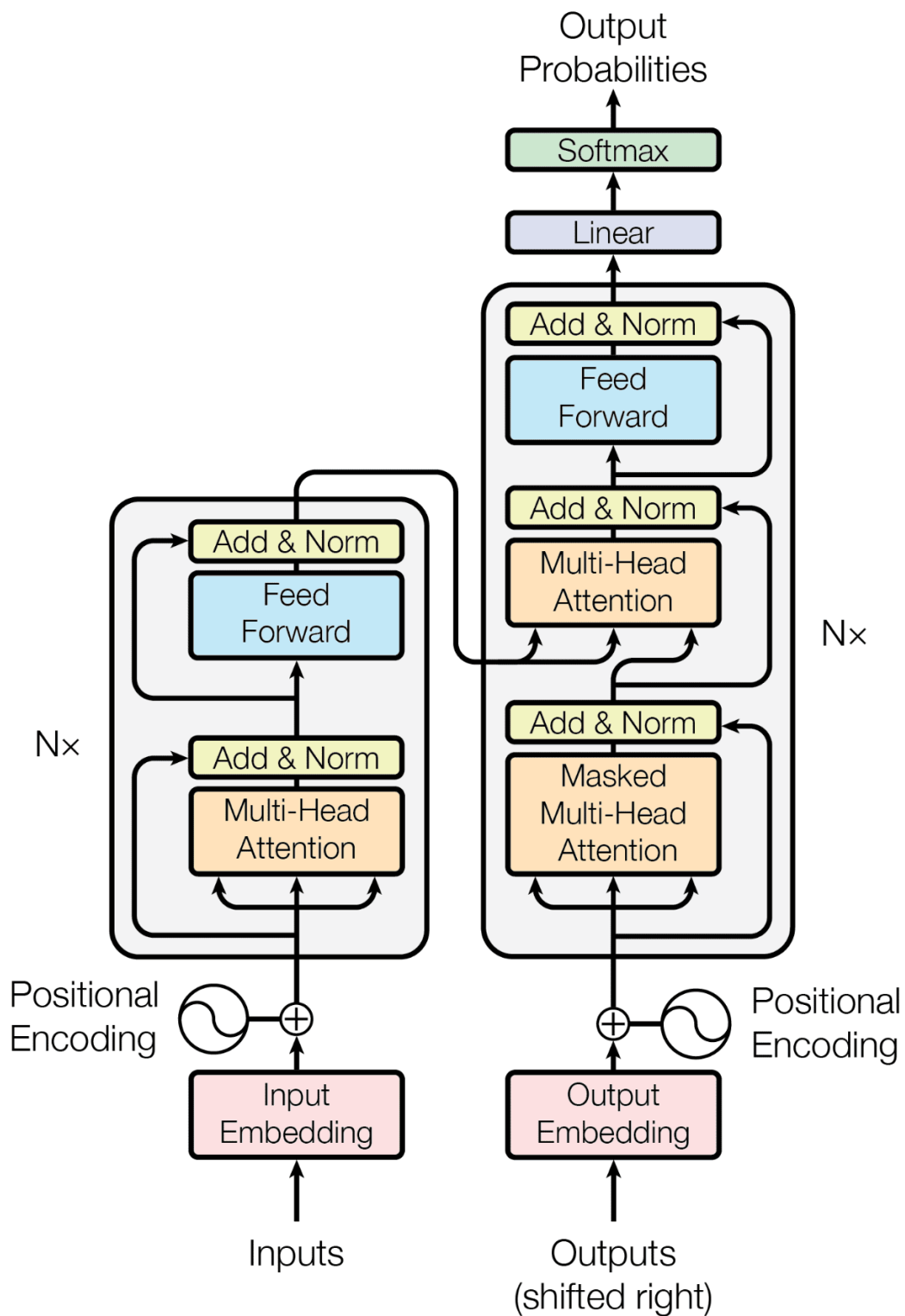


Figure 2.3: Transformer Model Architecture [12]

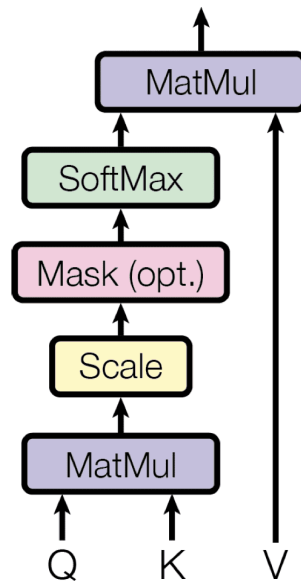


Figure 2.4: Multi-Head Attention [12]

of the key.

2.5.4 Positionally encoding different words

The Transformer does not have an RNN that can remember how input sequences were fed into the model. However, it needs to give every word or part in the sequence a relative position because a sequence depends on its elements' order. To address this requirement, the architecture incorporates positional encodings into the embedded representation of each word, which is represented as an n-dimensional vector.

2.5.5 Transformer challenges

The vanilla Transformer model helps overcome the RNN model's shortcomings but has two key issues:

- **Limited context dependency**-the Transformer surpasses LSTM models in character-level language modeling tasks, but it struggles to maintain long-term dependency information beyond the defined context length. Moreover, it cannot establish correlations with words that occurred several segments ago.
- **Context fragmentation**-context fragmentation occurs in the Transformer because it is trained independently for each segment. This means that no contextual information

is retained in the initial symbols of each segment, which can result in performance challenges.

2.6 ResNet-50

ResNet-50 [13] is a type of convolutional neural network (CNN) that has revolutionized the way we approach deep learning. It was first introduced in 2015 by Kaiming He et al. at Microsoft Research Asia.

ResNet stands for residual network, which refers to the residual blocks that make up the architecture of the network.

ResNet-50 is based on a deep residual learning framework that allows for the training of very deep networks with hundreds of layers.

The ResNet architecture was developed in response to a surprising observation in deep learning research: adding more layers to a neural network was not always improving the results.

This was unexpected because adding a layer to a network should allow it to learn at least what the previous network learned, plus additional information.

To address this issue, the ResNet team, led by Kaiming He, developed a novel architecture that incorporated skip connections.

These connections allowed the preservation of information from earlier layers, which helped the network learn better representations of the input data. With the ResNet architecture, they were able to train networks with as many as 152 layers.

The results of ResNet were groundbreaking, achieving a 3.57% error rate on the ImageNet dataset and taking first place in several other competitions, including the ILSVRC and COCO object detection challenges.

This demonstrated the power and potential of the ResNet architecture in deep learning research and applications.

2.6.1 ResNet-50 Architecture

ResNet-50 consists of 50 layers that are divided into 5 blocks, each containing a set of residual blocks. The residual blocks allow for the preservation of information from earlier layers, which helps the network to learn better representations of the input data.

The following are the main components of ResNET.

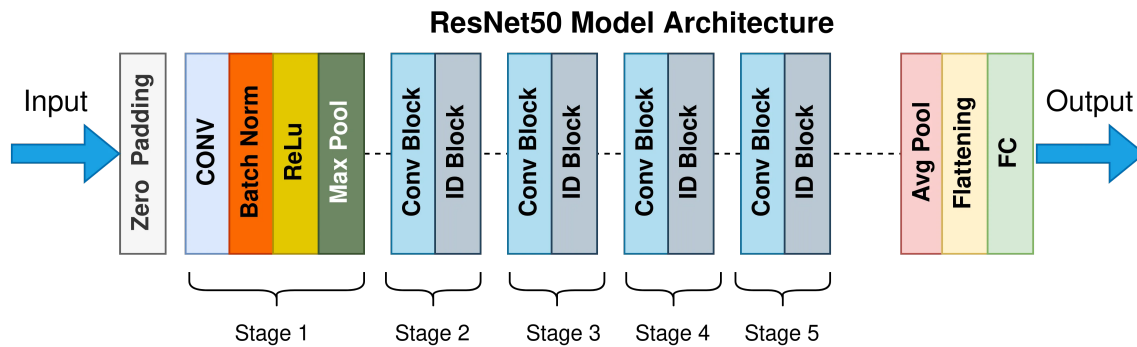


Figure 2.5: Resnet Architecture [13]

Convolutional Layers

The initial layer of the network is a convolutional layer responsible for convolving the input image. Subsequently, a max-pooling layer downsamples the output of the convolutional layer. Following this, the output of the max-pooling layer undergoes a sequence of residual blocks.

Residual Blocks

Every residual block is comprised of two convolutional layers, with each layer succeeded by a batch normalization layer and a rectified linear unit (ReLU) activation function. The output of the second convolutional layer is then added to the input of the residual block, which is then passed through another ReLU activation function. The output of the residual block is then passed on to the next block.

Fully Connected Layer

The ultimate layer of the network is a fully connected layer, which accepts the output from the last residual block and projects it onto the output classes. The number of neurons in this fully connected layer matches the number of output classes.

2.6.2 Key Features of ResNet-50

- ILSVRC'15 classification winner (3.57 % top 5 error).
- 152 layer model for ImageNet.
- Has other variants also (with 35, 50, 101 layers)
- Every 'residual block' has two 3×3 convolution layers

- No FC layer, except one last 1000 FC softmax layer for classification
- Global average pooling layer after the last convolution
- Batch Normalization after every convolution layer
- SGD + momentum (0.9)
- No dropout used

2.7 Vision Transformer (ViT)

The Vision Transformer [15] (ViT) model architecture was introduced in a research paper published as a conference paper at ICLR 2021 titled “An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale”. It was developed and published by Neil Houlsby, Alexey Dosovitskiy, and 10 more authors of the Google Research Brain Team.

2.7.1 Swin Transformer

The Swin Transformer [14] is a novel architecture designed for computer vision tasks, proposed in the paper titled "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows" by Ze Liu et al., published in November 2021. The Swin Transformer introduces a hierarchical processing mechanism that efficiently handles high-resolution images, addressing scalability challenges faced by previous transformer-based models like Vision Transformers (ViTs).

Here's a detailed overview of the Swin Transformer:

Hierarchical Processing:

The Swin Transformer divides the input image into non-overlapping patches and processes them hierarchically across multiple stages or "layers" of the model. This hierarchical processing enables the model to capture both local and global information effectively.

Shifted Windows:

Unlike traditional ViTs, which use fixed-size windows to aggregate information, the Swin Transformer employs shifted windows. Shifted windows allow the model to capture spatial relationships more effectively by processing overlapping regions across consecutive layers. This approach enhances the model's ability to capture fine-grained details and long-range dependencies.

Local Self-Attention Mechanism:

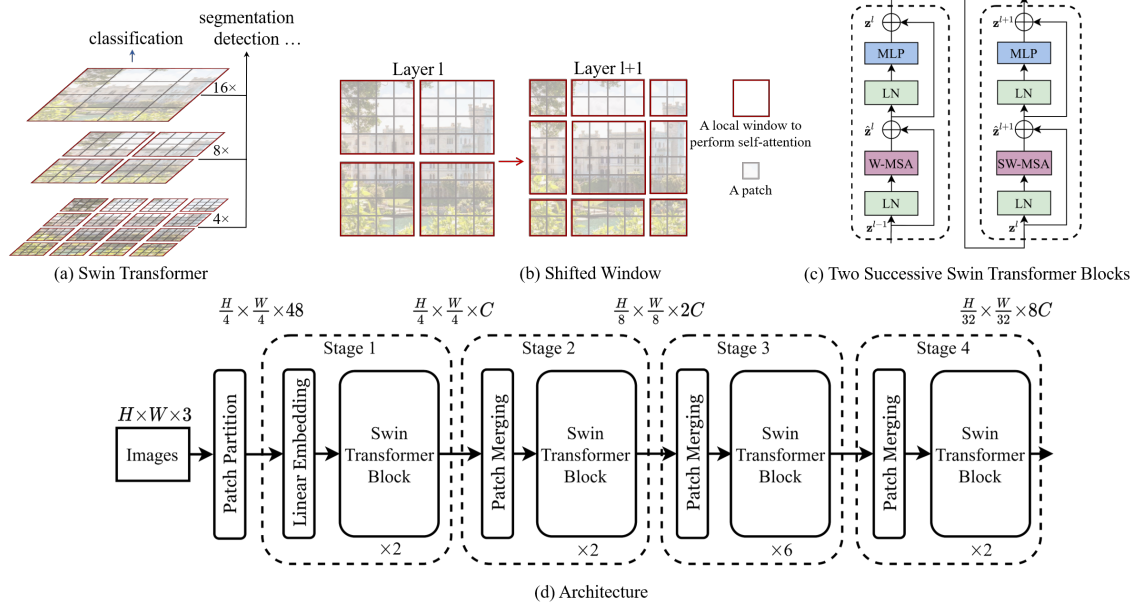


Figure 2.6: Swin Transformer Architecture [14]

At each layer, the Swin Transformer utilizes a local self-attention mechanism to aggregate information within each patch. This mechanism enables the model to focus on relevant features while reducing computational complexity, making it more efficient for processing high-resolution images.

Patch Merging:

After processing the patches within each layer, the Swin Transformer merges adjacent patches to form larger representations. This patch merging process helps the model capture hierarchical features at different scales, facilitating robust feature extraction.

Multi-Scale Feature Representation:

By processing images hierarchically and incorporating shifted windows and patch merging, the Swin Transformer generates multi-scale feature representations that capture both local and global information. This multi-scale representation is beneficial for various computer vision tasks, including image classification, object detection, and semantic segmentation.

Experimental Validation:

The Swin Transformer has been extensively evaluated on standard benchmark datasets, demonstrating superior performance compared to existing transformer-based models and convolutional neural networks (CNNs) across various tasks. The experiments highlight the scalability, efficiency, and effectiveness of the Swin Transformer in handling high-resolution images and achieving state-of-the-art results.

In summary, the Swin Transformer represents a significant advancement in transformer-based architectures for computer vision. Its hierarchical processing mechanism, combined

with shifted windows and patch merging, enables efficient and effective feature extraction from high-resolution images, making it a promising approach for a wide range of visual recognition tasks.

2.7.2 Vision Transformer and Image Classification

Image classification is a crucial task in computer vision, aiming to categorize images according to their content. Traditionally, deep convolutional neural networks (CNNs) such as YOLOv7 have been at the forefront of image classification methods. However, recent progress in transformer architecture, initially developed for natural language processing (NLP), has demonstrated significant potential in achieving comparable outcomes in image classification endeavors.

2.7.3 Vision Transformer ViT Architecture

Several vision transformer models have been proposed in the literature. The overall structure of the vision transformer architecture consists of the following steps:

1. Split an image into patches (fixed sizes)
2. Flatten the image patches
3. Create lower-dimensional linear embeddings from these flattened image patches
4. Include positional embeddings
5. Feed the sequence as an input to a state-of-the-art transformer encoder
6. Pre-train the ViT model with image labels, which is then fully supervised on a big dataset
7. Fine-tune the downstream dataset for image classification

Vision Transformers (ViT) is an architecture that uses self-attention mechanisms to process images. The Vision Transformer Architecture consists of a series of transformer blocks. Each transformer block consists of two sub-layers: a multi-head self-attention layer and a feed-forward layer.

In the self-attention layer, attention weights are computed for each pixel in the image by considering its interactions with all other pixels. Subsequently, the feed-forward layer applies a non-linear transformation to the output of the self-attention layer. The multi-head attention

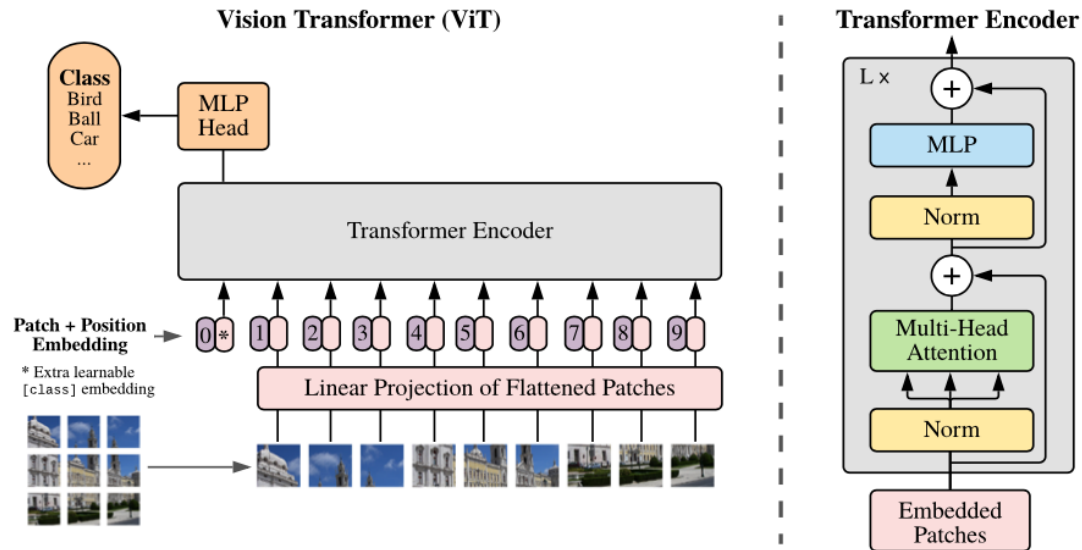


Figure 2.7: Vision Transformer ViT Architecture [15]

mechanism enhances this process by enabling the model to focus on various segments of the input sequence concurrently.

ViT incorporates an extra patch embedding layer, which partitions the image into fixed-size patches and assigns each patch to a high-dimensional vector representation. These patch embeddings are subsequently inputted into the transformer blocks for additional processing.

The final output of the ViT architecture is a class prediction, obtained by passing the output of the last transformer block through a classification head, which typically consists of a single fully connected layer. While the ViT full-transformer architecture is a promising option for

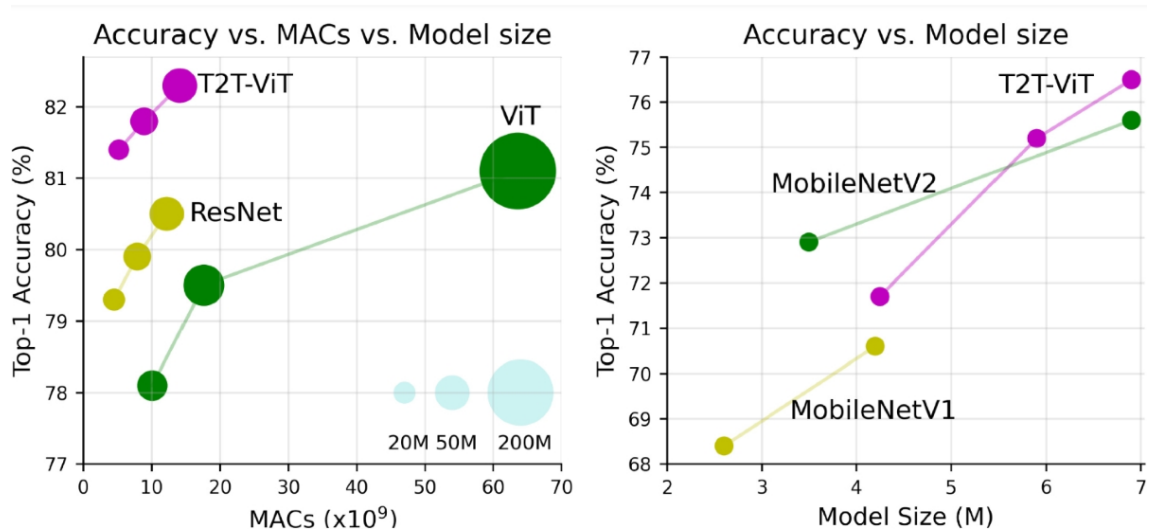


Figure 2.8: Performance benchmark comparison of Vision Transformers (ViT) with ResNet and MobileNet when trained from scratch on ImageNet [16]

vision processing tasks, the performance of ViTs is still inferior to that of similar-sized CNN

alternatives (such as ResNet) when trained from scratch on a mid-sized dataset such as ImageNet. Overall, the ViT architecture allows for a more flexible and efficient way to process images, without relying on pre-defined handcrafted features.

2.7.4 Working Procedure of A Vision Transformer (ViT)

The performance of a vision transformer model depends on decisions such as that of the optimizer, network depth, and dataset-specific hyperparameters. Compared to ViT, CNNs are easier to optimize.

The challenge with a pure transformer lies in integrating a transformer with a CNN front end. In typical ViT setups, a 16x16 convolution with a stride of 16 is utilized. However, employing a 3x3 convolution with a stride of 2 enhances stability and improves accuracy.

A CNN transforms raw pixels into a feature map, which is subsequently converted into a sequence of tokens by a tokenizer. These tokens are then fed into the transformer, where the attention mechanism is applied to generate an output token sequence.

Ultimately, a projector reconnects the output tokens to the feature map, enabling the examination to explore potentially important pixel-level details. This reduces the number of tokens that require analysis, resulting in significant cost savings.

Particularly, if the ViT model is trained on huge datasets that are over 14M images, it can outperform the CNNs. If not, the best option is to stick to ResNet or EfficientNet. The vision transformer model is trained on a huge dataset even before the process of fine-tuning. The only change is to disregard the MLP layer and add a new D times $KD \times K$ layer, where K is the number of classes of the small dataset.

To fine-tune in better resolutions, the 2D representation of the pre-trained position embeddings is done. This is because the trainable linear layers model the positional embeddings.

2.8 Pretrained Models:

2.8.1 Generative Pre-trained Transformer 2 (GPT-2) Based:

According to Radford et al. (2020) [18], the coming of advanced languages such as GPT-2 has been possible with the advancement in natural processing of language (NLP). These models have attracted a lot of attention due to their potential to change everything in areas like creative writing, automated customer service and so on, because they can read and write just like humans do. As stated in, recent strides in natural language processing (NLP)

have ushered in a new era, marked by the emergence of sophisticated language models such as GPT-2. These models, endowed with the capacity to comprehend and produce human-like language, have garnered significant attention owing to their transformative potential in various domains, ranging from creative writing to automated customer service.

Prior to the arrival of GPT-2, varied alternatives were tried by the researchers to equip computers with language abilities. Among these contenders are LSTM networks and RNNs, which are recurrent neural networks. Their efficacy was somehow good, but they failed to understand sequential texts that are long in length, hence new ways had to be sought.

The breakthrough arrived with the introduction of transformer-based models exemplified by GPT-2, which revolutionized the landscape of NLP. Unlike their predecessors, transformer models harnessed the power of self-attention mechanisms to grasp intricate linguistic nuances across extensive bodies of text, unlocking unprecedented capabilities in language understanding and generation.

However, the utility of these models comes at a cost, as their training necessitates vast amounts of data and computational resources. For languages with limited linguistic resources, such as Bangla, researchers encountered formidable challenges. Traditional approaches involved the utilization of multilingual frameworks trained on a plethora of languages, yet these proved suboptimal for languages like Bangla.

To address this gap, the scholarly community proposed BanglaGPT [19], a bespoke model tailored specifically for the Bengali language. This endeavor involved the meticulous curation of a sizable dataset comprising Bangla text from diverse sources, followed by the meticulous training of BanglaGPT on this corpus. The overarching objective was to equip BanglaGPT with the ability to comprehend and generate Bangla text with a level of accuracy and fluency akin to that exhibited by GPT-2 in the English language domain.

2.8.2 Electra-Based:

Natural language processing (NLP) has a new pretraining method. It's named Efficiently Learning an Encoder that Classifies Token Replacement Accurately (ELECTRA). This technique is inspired by Replaced Token Detection (RTD) [20]. The ELECTRA paper introduces this approach. It allows for better learning from input tokens compared to traditional methods. ELECTRA is more sample-efficient for pretraining language models. Specific tokens within the input sequence get replaced by viable options from a small generator network in the ELECTRA framework. A discrimination model then gets trained to predict whether each token comes from the generator network or not. This approach principally involves the discrimination model identifying original tokens versus tokens the small generator network produced. A big change from BERT's Masked Language Model (MLM) technique is

how ELECTRA uses Replaced Token Detection (RTD) instead. Unlike masking some input tokens, RTD allows the model to utilize every token for learning. The discrimination model spots which tokens got swapped, so ELECTRA analyzes the full input sequence. This adjustment boosts its performance and efficiency compared to masked learning on partial inputs. ELECTRA is a significant step towards better NLP pretraining, learning more from the data and with fewer data compared to the traditional unsupervised method like MLM. Especially by using the RTD and training a discriminator to distinguish between original and replaced tokens, we learn more effectively and affordably from the entire input context and significantly improve the learning effect and large-scale language capability.

2.8.3 Bidirectional Encoder Representations from Transformers (BERT)-Based:

In the past few years, natural language processing (NLP) was totally transformed. BERT or Bidirectional Encoder Representations from Transformers played a pivotal role. It was a significant breakthrough introduced by Devlin et al. in their highly influential research paper [17]. Conventional NLP models analyzed text in one direction only. They often failed to fully grasp the context, leading to subpar performance across different language processing tasks. BERT solved this issue by using deep bidirectional transformers - an innovative approach for better context comprehension. BERT uses a masked language model pre-training method. It predicts missing words in sentences. This helps BERT understand language structure. BERT creates contextualized representations that show deeper comprehension of language. This innovation influenced later transformer-based models. BERT became standard for many NLP tasks. It is used for named entity recognition, question answering, sentiment analysis, and natural language inference. BERT provides a foundation for advancing linguistic understanding and future research. BERT tokenizes input to convert it into numbers. It pre-trains on data by masking some words, learning to predict them. This bidirectional approach lets it understand context both ways. Next, BERT fine-tunes using labeled task data. Embedding layers represent words numerically. Encoder blocks process contextual information. A pooler layer aggregates this for output. BERT's architecture captures complex language patterns. It enables diverse NLP applications through these core features.

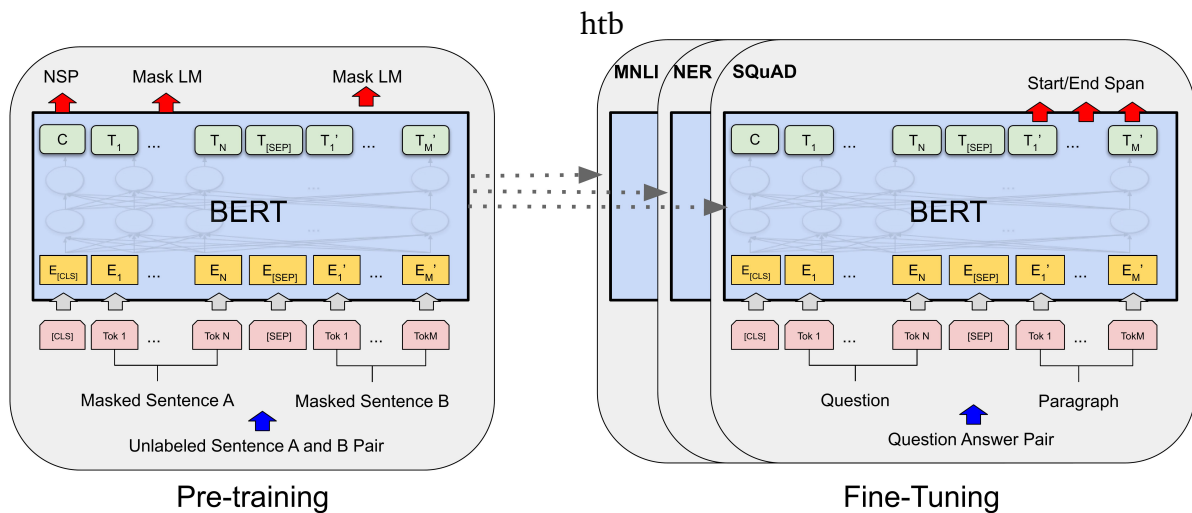


Figure 2.9: BERT architecture [17]

2.9 Vision Encoder-Decoder Model

The Vision Encoder-Decoder Model employs a transformer based design, initially introduced in Tr:OCR [8]. It includes a decoder that uses visual cues and past predictions to generate wordpiece sequences, while an encoder examines image patches as input. Image-based transformer models, like ViT, Swin etc, extract features from input photographs. The transformer generates patch embeddings by transferring flattened patches to vectors using a linear projection. The model balances encoder-decoder attention between output and input. Linear projection and softmax functions are used to project hidden states and calculate probability within a language. The final result is produced using a beam search strategy.

2.10 Performance Metrics

Upon the completion of training various models, it becomes imperative to assess their respective performances. To this end, Evaluation Metrics serve as the standard for appraising the efficacy of the models. It is crucial to employ data that have not been previously used in the training phase for testing purposes. This approach prevents the potential risk of overfitting, which can occur if the same dataset is applied for both training and testing. In our study, we have employed two distinct evaluation metrics: the Bilingual Evaluation Understudy (BLEU) [21] and the Metric for Evaluation of Translation with Explicit Ordering (METEOR) [22], to appraise the test data using the models that have been trained.

2.10.1 BLEU

The Bilingual Evaluation Understudy (BLEU) [21] metric is currently the most prevalent tool for assessing text quality. It measures the naturalness of sentences in comparison to those crafted by humans and is widely employed in gauging the efficacy of machine translation systems. The comparison of sentences is conducted using a modified n-gram precision approach to produce BLEU scores, which are calculated as per the equations that follow:

$$P(i) = \frac{\text{Matched}(i)}{H(i)} \quad (2.3)$$

where $P(i)$ is the precision that is for each i -gram where $i = 1, 2, \dots, N$, the percentage of the i -gram tuples in the hypothesis that also occur in the references is computed. $H(i)$ is the number of i -gram tuples in the hypothesis and $\text{Matched}(i)$ is computed using the following formula:

$$\text{Matched}(i) = \sum_{t_i} \min_j (C_h(t_i), \max C_{hj}(t_i)) \quad (2.4)$$

where t_i is an i -gram tuple in hypothesis h , $C_h(t_i)$ is the number of times t_i occurs in the hypothesis, $C_{hj}(t_i)$ is the number of times t_i occurs in reference j of this hypothesis.

$$\rho = \exp \left\{ \min \left(0, \frac{n-L}{n} \right) \right\} \quad (2.5)$$

where ρ is brevity penalty to penalize short translation, n is the length of the hypothesis and L is the length of the reference. Finally, the BLEU score is computed by:

$$\text{BLEU} = \rho \left\{ \prod_{i=1}^N P(i) \right\}^{\frac{1}{N}} \quad (2.6)$$

2.10.2 METEOR

The Metric for Evaluation of Translation with Explicit Ordering (METEOR) [22] metric employs a unigram matching strategy that compares machine-predicted sentences to reference sentences. This comparison is quantified using the harmonic mean of unigram precision and recall, with a greater emphasis placed on recall over precision. METEOR was developed to address certain limitations identified in the BLEU metric. The calculation of unigram precision (P) is as follows:

$$P = \frac{m}{w_t} \quad (2.7)$$

Where m is the number of unigrams in the candidate translation that are also found in the reference translation, and w_t is the number of unigrams in the candidate translation.

Unigram recall R is computed as follows:

$$R = \frac{m}{W_r} \quad (2.8)$$

Where m is as above, and w_r is the number of unigrams in the reference translation. Precision and recall are combined using the harmonic mean. There recall is weighted 9 times more than precision as shown in the equation below:

$$F_{mean} = \frac{10PR}{R + 9P} \quad (2.9)$$

To account for congruity with respect to larger segments that appear in both the reference and the candidate sentence a penalty p is added. The penalty is calculated using the following equation.

$$p = 0.5 \left(\frac{C}{u_m} \right)^3 \quad (2.10)$$

Where C is the number of chunks, and u_m is the number of unigrams that have been mapped. Finally, the METEOR score for a segment is calculated as M as shown in the equation below.

$$M = F_{mean}(1 - P) \quad (2.11)$$

Chapter 3

Literature Review

3.1 Paper Reviews

3.1.1 Chittron: An Automatic Bangla Image Captioning Systemcite [1]

"Chittron: An Automatic Bangla Image Captioning System" is a pioneering work in the field of Bangla image captioning. Recognizing the lack of research and resources in this domain compared to other languages, the authors introduce Chittron, a system for automatically generating captions in Bangla for images. To address the challenge of limited data, they built a valuable dataset of 16,000 Bangladeshi contextual images with corresponding Bangla captions. Chittron's deep learning model leverages a pre-trained VGG16 for image feature extraction and stacked LSTMs to handle the sequential nature of language for caption generation.

This paper holds significant value for two main reasons. Firstly, it initiates research in automatic Bangla image captioning, paving the way for further advancements in this field. Secondly, the creation of a Bangla image captioning dataset serves as a crucial resource for future research endeavors. While the current dataset size might limit captioning accuracy, and the evaluation focuses primarily on qualitative aspects, Chittron successfully demonstrates the feasibility of training a model for Bangla image captioning. Future directions could involve utilizing larger datasets, exploring more advanced architectures like transformers, and incorporating quantitative evaluation metrics to achieve even more accurate and informative Bengali image captions.

3.1.2 BORNON: BENGALI IMAGE CAPTIONING WITH TRANSFORMER-BASED DEEP LEARNING APPROACH [2]

Building upon the groundwork laid by Chittron, the research paper "BORNON: Bengali Image Captioning with Transformer-Based Deep Learning Approach" tackles the limitations of previous methods for Bengali image captioning. While Chittron established the concept with LSTMs, BORNON takes a significant leap forward by introducing a Transformer-based architecture.

Traditional encoder-decoder architectures using CNNs and RNNs, like LSTMs in Chittron, have been successful in image captioning. However, RNNs struggle with capturing long-range dependencies within sentences, which are crucial for accurate and comprehensive captions. BORNON addresses this challenge by employing Transformers. Transformers excel at capturing these long-range dependencies and enable parallel processing, potentially leading to more informative Bengali captions.

Furthermore, BORNON specifically focuses on the Bengali language, which has received less research attention in image captioning compared to English. By evaluating their model on three different Bengali image captioning datasets, the authors provide a comprehensive analysis of its effectiveness in this under-explored domain.

In essence, BORNON builds upon the foundation laid by Chittron's automatic Bangla image captioning system. It leverages the strengths of Transformer architectures to address the limitations of RNNs and explores the potential of Transformers for Bengali image captioning, paving the way for further advancements in this field.

3.1.3 TextMage: The Automated Bangla Caption Generator Based On Deep Learning [3]

This paper presents TextMage, an automated image captioning system that can generate captions in the Bangla language for images with a South Asian context. The authors note that most existing image captioning systems have a Western bias in terms of the data and languages used.

The paper provides an overview of recent work related to image captioning using deep learning techniques. This includes:

- Using convolutional neural networks (CNNs) like Inception-v3 for image classification and feature extraction
- Employing recurrent neural networks (RNNs) and long short-term memory (LSTM) models for sequence prediction and text generation
- Combining CNNs and RNNs/LSTMs into an end-to-end trainable model for image captioning
- Exploring attention mechanisms and generative adversarial networks (GANs) for improved caption quality

The authors created a new dataset called BanglaLekhaImageCaptions containing 9,154 images along with two human-written Bangla captions for each image. This dataset aims to reduce the Western bias present in commonly used datasets like Flickr and MSCOCO.

For their image captioning model, the authors used a CNN (VGG-16) for image feature extraction and an RNN with LSTM cells for caption generation. They trained the CNN and RNN components separately before combining them into an end-to-end model.

The results show that their model achieved good accuracy on the BanglaLekhaImageCaptions dataset during training (0.92) and validation (0.74). They evaluated the quality of the generated captions using metrics like BLEU and METEOR, comparing against benchmark results from prior work on English datasets like Flickr8K and MSCOCO.

In discussing their work, the authors highlight the need to develop image captioning systems for underrepresented languages and locales to reduce dataset bias. They suggest exploring newer architectures like attention models could further improve performance.

Overall, this paper makes a valuable contribution by creating a Bangla image captioning dataset and associated deep learning model tailored for the South Asian context.

3.1.4 Bangla Image Captioning through Transformer-based Encoder - Decoder [4]

Bengali image captioning research is making strides towards generating more accurate and informative descriptions. This particular paper explores a novel approach that leverages the power of Transformer-based architectures.

Previously, Bengali image captioning relied on encoder-decoder structures with Convolutional Neural Networks (CNNs) for image analysis and Recurrent Neural Networks (RNNs)

for caption generation. While this approach achieved some success, RNNs struggle with capturing long-range dependencies between words in sentences. These dependencies are crucial for generating captions that accurately reflect the content and relationships within an image.

This paper proposes a solution by introducing a Transformer-based encoder-decoder network. Transformers excel at capturing long-range dependencies, allowing the model to better understand the relationships between different elements in the image. Additionally, the model employs a pre-trained ResNet-101 model within the encoder. This pre-trained component helps extract richer and deeper visual features from the image, providing the decoder with a more comprehensive understanding of the scene.

The decoder itself utilizes a Transformer architecture with an attention mechanism. This mechanism allows the decoder to selectively focus on specific parts of the extracted image features while generating each word of the caption. This targeted focus leads to a more relevant and cohesive description, ensuring the generated caption accurately reflects the content of the image.

By incorporating these advancements, the paper presents a significant contribution to Bengali image captioning. The use of Transformers and attention mechanisms addresses the limitations of RNNs, paving the way for more accurate, informative, and grammatically correct Bengali captions.

3.1.5 Improved Bengali Image Captioning via deep convolutional Neural Network Based Encoder-Decoder Model [5]

Generating accurate and natural language captions for Bengali images presents a unique challenge. The paper "Improved Bengali Image Captioning via deep convolutional Neural Network Based Encoder-Decoder Model" tackles this by proposing a novel encoder-decoder architecture that leverages the strengths of deep learning.

Existing encoder-decoder models used for Bengali image captioning might struggle to capture the finer details within an image or the nuances of the Bengali language. This paper addresses this by introducing a unique two-pronged approach:

Enhanced Image Representation: The model employs a pre-trained ResNet-50 model as the image encoder. This powerful Convolutional Neural Network (CNN) excels at extracting rich visual features from images. It goes beyond basic object recognition, capturing intricate details and spatial relationships within the scene. This comprehensive visual representation provides a strong foundation for accurate caption generation.

Language-Specific Encoding: In addition to the image encoder, the model incorporates a separate encoder specifically designed for Bengali. This language encoder analyzes existing Bengali captions to understand the language's intricacies. By encoding this linguistic knowledge, the model ensures the generated captions are grammatically correct and fluent in Bengali.

The decoder component then takes the extracted visual features and the encoded language information to generate Bengali captions. The paper's evaluation demonstrates that this approach achieves state-of-the-art performance compared to existing methods. This improvement can be attributed to the combined strengths of the pre-trained image encoder providing detailed visual data and the language encoder ensuring grammatically accurate Bengali output.

Overall, this research significantly advances Bengali image captioning. By incorporating deep convolutional neural networks for robust image analysis and a dedicated language encoder for Bengali, the model paves the way for generating more accurate and natural-sounding Bengali captions from images. This can have numerous applications, such as improving accessibility for visually impaired Bengali speakers or enriching image search results with informative captions in Bengali.

3.1.6 Task-Adaptive Attention for Image Captioning [6]

The task of automatically generating captions for images, known as image captioning, has witnessed significant advancements with deep learning techniques. However, a key challenge lies in efficiently directing the model's attention to relevant image regions while generating captions. Traditional attention mechanisms often allocate attention to the entire image for every word in the caption, which can be computationally expensive and might not be necessary for all words.

Imagine a scenario where the model is generating the caption "The red car is parked in

the driveway." The word "the" likely doesn't require focusing on any particular object in the image. In contrast, the word "red" would benefit from attending to the specific region containing the car. Adaptive Attention tackles this challenge by incorporating a separate network called the "Visual Sentinel."

The Visual Sentinel acts as a predictor, estimating how much visual information is necessary for generating the next word. Based on this prediction, the attention mechanism dynamically assigns weights to different image regions. Words that require more visual context, like "car" in our example, receive higher weights for the relevant areas of the image. Conversely, less visually dependent words like "the" receive lower weights across the entire image.

This approach offers several potential benefits. By focusing on relevant image areas, the model can potentially reduce the computational cost required for attention computation, leading to faster training times. More importantly, by selectively attending to informative regions, the model might generate more accurate and relevant captions, as it focuses on the most crucial visual cues for each word.

In conclusion, the concept of task-adaptive attention presented in this paper offers a promising direction for improving image captioning. By dynamically allocating attention based on the word being generated, the model can potentially achieve better efficiency and accuracy in captioning tasks. While this paper focuses on image captioning in general, the concept could be potentially applied to Bengali image captioning models that also utilize attention mechanisms, like BORNON, for further advancements in this domain.

3.1.7 CPTR: FULL TRANSFORMER NETWORK FOR IMAGE CAPTIONING [7]

The paper "CPTR: Full Transformer Network for Image Captioning" presents a novel approach for generating image captions by leveraging the power of Transformer architectures. It deviates from the traditional "CNN + Transformer" paradigm used in image captioning tasks.

Current image captioning methods often rely on a combination of Convolutional Neural Networks (CNNs) for image feature extraction and Recurrent Neural Networks (RNNs) for caption generation. While effective, RNNs struggle with capturing long-range dependencies between words in sentences. These dependencies are crucial for generating accurate and

comprehensive captions that accurately reflect the relationships within an image.

CPTR presents a promising alternative to existing image captioning methods. By utilizing a full Transformer network, it addresses the limitations of RNNs and offers the potential for generating more accurate and informative captions that capture the relationships within an image.

3.1.8 TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models [8]

The paper "TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models" introduces a novel approach to Optical Character Recognition (OCR) for various text types, including printed, handwritten, and scene text. It leverages the power of Transformer architectures, moving away from traditional methods in OCR.

Conventional OCR systems often rely on a combination of Convolutional Neural Networks (CNNs) for image understanding and Recurrent Neural Networks (RNNs) for character-by-character text generation. Additionally, a separate language model might be required for post-processing to improve overall accuracy. CNNs might not capture long-range dependencies within the image, crucial for accurate recognition of words or phrases. RNNs struggle with handling long sequences, which can be an issue for complex text recognition tasks.

TrOCR proposes an end-to-end OCR system built entirely on Transformer architecture. Here's its core structure:

- An image Transformer encoder processes the input image, extracting visual features that represent the characters within the text.
- A text Transformer decoder, equipped with an attention mechanism, generates the recognized text word-by-word. The attention mechanism allows the decoder to focus on specific parts of the extracted visual features while generating each word, leading to more accurate recognition.
- TrOCR leverages pre-trained image and text Transformer models for improved performance. These pre-trained models provide a strong foundation for the system to learn from vast amounts of existing data, improving its ability to recognize diverse text types.

By utilizing Transformers, TrOCR can capture long-range dependencies within the image, leading to more accurate recognition of characters, words, and even scene text. Unlike traditional methods with separate components, TrOCR is an end-to-end system, streamlining the OCR process and potentially enhancing efficiency. Pre-trained models provide a significant advantage, allowing TrOCR to perform well even with limited training data for specific OCR tasks.

Overall, TrOCR presents a significant advancement in OCR by introducing a Transformer-based approach. This approach addresses the limitations of CNNs and RNNs, offering improved accuracy and efficiency for recognizing various text types from images.

3.1.9 A Visual Attention-Based Model for Bengali Image Captioning [9]

Introduction:

The paper proposes an encoder-decoder model using a convolutional neural network (CNN) as the encoder and a bidirectional gated recurrent unit (BGRU) as the decoder to generate captions in the Bengali language from images. Generating image captions is a challenging task that combines computer vision and natural language processing.

Dataset:

The authors created a new Bengali image captioning dataset called BNATURE containing 8,000 images with 5 captions each, as existing Bengali datasets had errors. This enables training their model on a reasonably large Bengali dataset.

Methodology:

The approach follows an encoder-decoder framework, using the Inception V3 CNN as the image encoder and a bidirectional GRU as the caption decoder. Beam search and argmax are used to generate captions during inference.

Evaluation:

The authors evaluated their Bengali image captioning model on the BNATURE test set of 1,000 images using the standard BLEU and METEOR metrics. They found that using Beam Search decoding with a beam size of 3 produced the best results. With Beam Search, they obtained BLEU scores of 42.6 for BLEU-1, 27.95 for BLEU-2, 23.66 for BLEU-3, 16.41 for BLEU-4, and a METEOR score of 28.7. In comparison, using argmax decoding gave lower scores of 40.54 for BLEU-1, 25.22 for BLEU-2, 20.59 for BLEU-3, 13.5 for BLEU-4, and 28.1 for METEOR. So Beam Search provided around 2% absolute improvement in BLEU

scores across the different n-gram orders compared to argmax, while the METEOR scores were more similar between the two decoding methods. Overall, the scores demonstrate promising results in generating Bengali image captions using their CNN-BGRU model on the new BNATURE dataset.

The results are promising, demonstrating the model's ability to generate reasonable Bengali captions from images using the new BNATURE dataset. However, there is still room for improvement compared to human-written captions. The authors discuss some limitations of RNNs and outline potential benefits of their system.

In summary, this paper makes a useful contribution by developing one of the first image captioning models for the Bengali language and creating a new Bengali dataset. It builds on established encoder-decoder approaches while adapting the architecture for the low-resource Bengali setting. The literature review covers the relevant background and context for this work.

3.1.10 Amharic Language Image Captions Generation Using Hybridized Attention-Based Deep Neural Networks [10]

Introduction:

The paper aims to develop a hybridized deep learning model for generating semantically meaningful image captions in the Amharic language. Image captioning is a task that combines computer vision and natural language processing (NLP) domains. The authors acknowledge that existing studies in the English language primarily focus on visual features to generate captions, resulting in a gap between visual and textual features and inadequate semantic representation.

Proposed Approach:

To address the limitations of previous models, the authors propose a hybridized attention-based deep neural network (DNN) model. The model consists of an Inception-v3 convolutional neural network (CNN) encoder to extract image features, a visual attention mechanism to capture significant features, and a bidirectional gated recurrent unit (Bi-GRU) with attention decoder to generate the image captions.

The visual attention mechanism allows the model to focus on the most relevant parts of the image, while the Bi-GRU with attention decoder selects the most appropriate words to describe the image content, reducing the gap between visual and textual features and leading to semantically richer image captions.

Methodology:

The authors describe in detail the dataset preparation, text preprocessing techniques, image preprocessing, and evaluation metrics used in their study.

1. Dataset Preparation: The study uses the Flickr8k and BNATURE datasets, which contain 8,000 images each, with five captions in English and Amharic languages for each image. The English captions were translated into Amharic using Google Translate and reviewed by Amharic language experts to correct grammar and semantic errors.

2. Text Preprocessing Techniques: The authors performed data cleaning, including lowercasing words, removing special characters and numbers, and manually adjusting inappropriate words and symbols to maintain the original meaning of the translated captions. Additionally, tokenization and vectorization were performed to convert the captions into numerical representations suitable for the algorithms.

3. Image Preprocessing: The images were resized to a resolution of 299 x 299 pixels with 3 color channels. Grayscale histogram and data augmentation techniques were applied. The Inception-v3 CNN model was used for feature extraction, removing the last SoftMax layer and focusing on the 2048 features of each image.

4. Evaluation Metric: The BLEU (Bilingual Evaluation Understudy) metric, a precision-based metric widely used in NLP, was employed to evaluate the quality of the machine-generated text by comparing the n-grams of the predicted output to the actual data.

Experiments and Results:

The authors conducted two experiments to evaluate the performance of the proposed model in comparison to the base model (CNN-Bi-GRU encoder-decoder) and the Bag-LSTM model (Cao et al., 2019).

The results showed that the proposed hybridized model outperformed both baseline models on all four BLEU scores (1G-BLEU, 2G-BLEU, 3G-BLEU, and 4G-BLEU) for both the Flickr8k and BNATURE datasets. The proposed model achieved a 21% improvement in the 4G-BLEU score compared to the CNN-Bi-GRU and Bag-LSTM models.

The authors attribute the improved performance to the integration of visual attention and the Bi-GRU with attention mechanism, which allows the model to focus on the most relevant visual and textual features, reducing the gap between them and generating semantically richer image captions.

Conclusion:

The study highlights the effectiveness of the hybridized attention-based approach in generating Amharic language image captions with better semantic meaning. The authors conclude that their proposed model addresses the existing gaps in the area by incorporating attention mechanisms and a Bi-GRU architecture, enabling the model to capture both visual and

textual features more effectively.

While the proposed model demonstrated promising results, the authors acknowledge some limitations, including increased model complexity, potential performance issues with complex or unusual images, and the need for further research to improve generalizability across different languages and cultures.

Overall, the literature review provides a comprehensive overview of the existing approaches to image captioning, highlights the limitations of previous models, and presents the authors' proposed hybridized attention-based DNN model as an effective solution for generating semantically meaningful Amharic language image captions.

3.2 Research Gap

- CLIP encoders have not yet been implemented for Bangla captioning.
- Large Language models have not yet been implemented for Bangla captioning.
- Works done for Bengali Image Captioning are too few.
- The maximum sample of the native Bangla dataset used is, 9154 with two captions for each image in Bangla.
- Lack of large native Bangla datasets like COCO and Flickr30k
- No use of image object detection of any sort.

Chapter 4

Dataset

Our goal is to generate Bengali Captions. In order to complete this task, we require a dataset in Bengali comprising numerous images along with a corresponding text file containing captions written in the Bengali language for each image. So for Dataset, we have used the following datasets: BanglaLekha, BNATURE and our own dataset.

4.1 BanglaLekha Dataset

BanglaLekha dataset [1] consists of images and annotations in Bengali. The images are human annotated in Bengali by two adult native Bengali speakers. All popular image captioning datasets have a predominant western cultural bias with the annotations done in English. Using such datasets to train an image captioning system assumes that a good English to target language translation system exists and that the original dataset had elements of the target culture. Both these assumptions are false, leading to the need of a culturally relevant dataset in Bengali, to generate appropriate image captions of images relevant to the Bangladeshi and wider subcontinental context. The dataset consists of 9,154 images. One problem with this dataset is that it has only two captions associated with each image resulting in 18308 captions for those 9154 images. Some images of the BanglaLekha dataset along with their associated captions are shown in figure 4.1

	<p>Caption 1: নৌকা নিয়ে একটা মেয়ে শাপলা ফুল তুলছে। Caption 2: নদীতে নৌকা নিয়ে একটা মেয়ে শাপলা ফুল তুলছে।</p>
	<p>Caption 1: দুইজন শিশু গরুর উপর বসে আছে ও গরুটি ঘাস খায়। Caption 2: দুইজন শিশু গরুর উপর বসে আছে।</p>
	<p>Caption 1: দুইটি মেয়ে ঝাউগাছের সামনে দাঁড়িয়ে আছে। Caption 2: খোলা চুলের দুটি মেয়ে পার্কের মধ্যে দাঁড়িয়ে আছে।</p>

Figure 4.1: Illustration of some images of the BanglaLekha dataset along with their two Bengali captions.

4.2 BNature Dataset

The BNature [9] dataset consists of 8000 images with the dimension of 500 x 375 pixels . All these pictures represent Bangladeshi lifestyle and nature. Every image consists of 5 Bengali captions. The structure of the dataset is followed by recognized datasets like flicker8k [23], flicker30k [24], MSCOCO [25]. Main disadvantage of this dataset is the captions are less descriptive which means captions do not contain the description of all the objects of an image. Some images of the BanglaLekha dataset along with their associated captions are shown in figure 4.1




	<p>Caption 1: মাঠের মাঝে দুইটি ঘর।</p> <p>Caption 2: পাহাড়ের মাঝে দুইটি ঘর।</p> <p>Caption 3: মেঘলা আকাশ ও পাহাড়।</p> <p>Caption 4: পাহাড়ি রাস্তা ও কিছু ঘর।</p> <p>Caption 5: পাহাড়ের মাঝে দুইটি লাল ঘর।</p>
	<p>Caption 1: পানির মধ্যে অনেকগুলো পাথর।</p> <p>Caption 2: অনেকগুলো পাথর।</p> <p>Caption 3: পানির মধ্যে অনেক পাথর ও পাহাড় দেখা যায়।</p> <p>Caption 4: পানির মধ্যে অনেক পাথর ও সবুজ গাছ দেখা যায়।</p> <p>Caption 5: পানির মধ্যে অনেক পাথর ও সবুজ গাছ।</p>
	<p>Caption 1: নদীতে একজন মাঝি নৌকা চালায়।</p> <p>Caption 2: নদীর মাঝে একটি নৌকা।</p> <p>Caption 3: একজন মাঝি নদীতে নৌকা চালায়।</p> <p>Caption 4: নদীতে একজন লোক নৌকা চালায়।</p> <p>Caption 5: একজন লোক নদীতে নৌকা চালায়।</p>

Figure 4.2: Illustration of some images of the BNature dataset along with their two Bengali captions.

4.3 Our Dataset

Our dataset consists of 7729 images and annotations in Bengali. The images are human annotated in Bengali by five adult native Bengali speakers. Each image have five captions. The dimension of the images are not same for all considering the fact that in reality there can varying dimensions of images. Also captions of our dataset are more descriptive. Each caption of an image has all the description of the image. Also excessive repetition of any word is avoided in most of the captions. Some images of the BanglaLekha dataset along with their associated captions are shown in figure 4.3



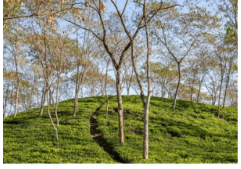
	<p>Caption 1: দুটি পাকা লেবু দেখা যাচ্ছে।</p> <p>Caption 2: দুটো পাকা লেবু দেখা যাচ্ছে।</p> <p>Caption 3: দুটি পাকা লেবু দেখা যায়।</p> <p>Caption 4: দুটো পাকা লেবু দেখা যায়।</p> <p>Caption 5: দুটি পাকা লেবু দেখা যাচ্ছে।</p>
	<p>Caption 1: একটি মালবাহী ট্রাক দেখা যাচ্ছে।</p> <p>Caption 2: একটা মালবাহী ট্রাক আছে।</p> <p>Caption 3: এখানে একটি মালবাহী ট্রাক আছে।</p> <p>Caption 4: এখানে একটা মালবাহী ট্রাক আছে।</p> <p>Caption 5: একটা মালবাহী ট্রাক দেখা যাচ্ছে।</p>
	<p>Caption 1: একটি উঁচু পাহাড়ের মধ্যে কয়েকটি গাছ এবং চা বাগান দেখা যাচ্ছে।</p> <p>Caption 2: উঁচু পাহাড়ে কিছু গাছ ও চা বাগান রয়েছে।</p> <p>Caption 3: উঁচু পাহাড়ে কিছু গাছ ও চা বাগান দেখা যায়।</p> <p>Caption 4: উঁচু পর্বতে কিছু গাছ ও চা বাগান রয়েছে।</p> <p>Caption 5: একটি উঁচু পাহাড়ে কিছু গাছ ও চা বাগান রয়েছে।</p>

Figure 4.3: Illustration of some images of our dataset along with their two Bengali captions.

Chapter 5

Methodology

Image captioning is a multimodal task, as it leverages both Computer Vision and Natural Language Processing (NLP). We used different visual models and sequence models and combining them to form an Encoder-Decoder architecture to generate meaningful Bangla captions. In the chapter, we will be diving deep into our approaches.

5.1 Dataset Preprocessing

The datasets mentioned in Chapter 4 have been split into three parts such as training, validation and testing. In our experiments, we used split ratio as shown in table 5.1. The data had to be pre-processed before passing it to the visual feature extraction models as input. The images were converted into 224x224x3 for all the proposed models as described in 5.2. The 3 color channels R (Red), G (Green), B (Blue) are denoted by the 3 value in the input image size. The train captions are pre-processed by the sequence model as the captions are tokenized and start/end tokens are added. The dataset includes images identified by unique image ID values. Each image has multiple associated captions, each identified by a unique ID within the image. Captions are written in Bangla and describe the content or scene depicted in the corresponding image.

Table 5.1: Dataset Splitting Ratio

Dataset	Total Images	Training	Validation	Testing
BanglaLekha	9154	7414(81%)	824(9%)	916(10%)
BNature	8000	6480(81%)	640(8%)	880(11%)
Our Dataset	7729	6259(81%)	619(8%)	851(11%)

5.2 Our Approaches

Our Two employed approaches to generating meaningful image captions in Bangla language are described in detail in this section.

5.2.1 Contrastive Language–Image Pre-training (CLIP) Based Model

In this method, we used the CLIP, a pretrained model, which has been trained over an extremely large number of images, so is capable of generating semantic encodings for arbitrary images without additional supervision. To produce meaningful sentences, we fine-tune a pre-trained GPT-2. The key idea is to use the CLIP encoding as a prefix to the textual captions by employing a simple MLP over the raw encoding, and then fine-tune our language model to generate a valid caption.

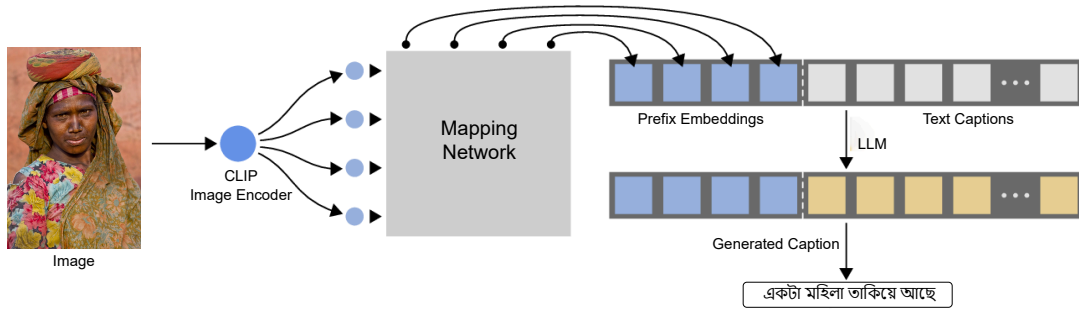


Figure 5.1: Overview of the Clip based model.

At first, embeddings were generated for the input images by using the clip model’s image encoder. Vision Transformer(ViT) and ResNet50x4 which has 4x higher width than the original ResNet50 architecture are used as the image encoder. The embeddings created are of shape [total captions, 512] for the Vit and [total captions, 640] for the ResNet50. Then these embeddings are passed onto a Multi Layer Perceptron(MLP) which generates a sentence of 10 tokens as the prefix length is initialized as 10. Then those tokens are concatenated with the captions. Our new list of tokens that contains the image tokens and caption tokens is used to fine-tune BanglaGPT [19]. After fine-tuning the large language model, we will generate the captions. Now we will use beam search taking our model, GPT-2 tokenizer and prefix embeddings as parameters.

We employed a simple Multi-Layer Perceptron as the Mapping Network. We utilized only a single hidden layer because the clip is pre-trained for vision language. As we are to train the language model, MLP is used to project the image features into the same space as the GPT2 embeddings. The MLP is used to generate the fixed length prefix text from image embeddings.

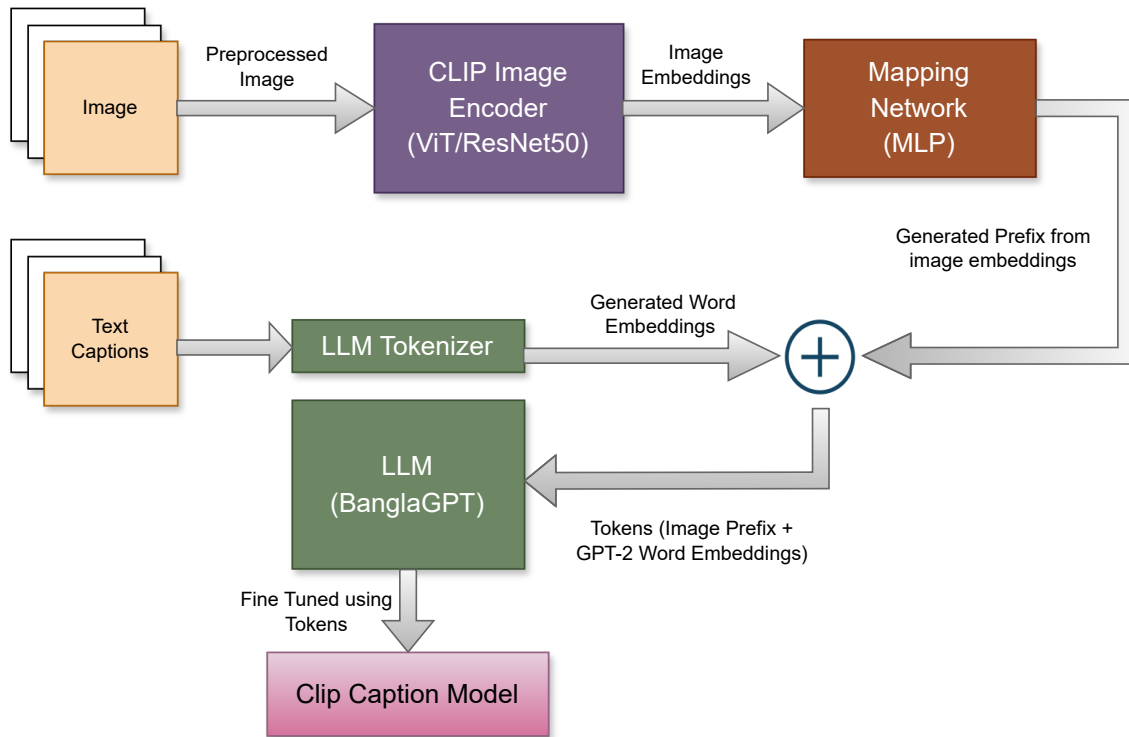


Figure 5.2: Proposed Clip based model.

The MLP network is frozen while the language model is fine-tuned. Another model using a Transformer model as the mapping network was also used where the language model was frozen instead the mapping network was trained. After fine-tuning, text generation is done through beam search which is a systematic search strategy that explores multiple possible sequences of words, known as "beams," and selects the most likely sequence based on a scoring mechanism, and it iteratively generates the next token for each beam until the maximum sequence length is reached, or all beams have encountered the stop token. At each step, the model generates logits for the next token based on the current sequence. Logits are normalized using softmax and adjusted by temperature for randomness. If it's the first step, the top beam size tokens are selected as candidates for each beam. For subsequent steps, the beam search algorithm selects the top beam size candidates based on the sum of scores along each beam, while ensuring that stopped beams are not considered. The selected tokens are appended to the sequences and used for the next step of generation. In Fig 5.3, caption generation is visualized given beam size of 2.

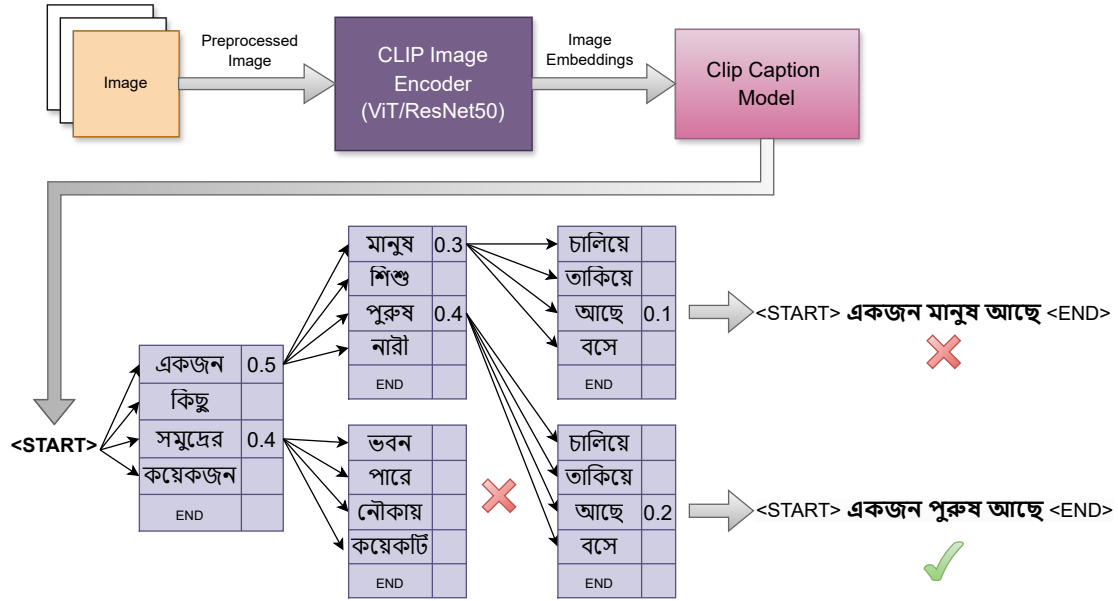


Figure 5.3: Caption generation using Clip based model.

5.2.2 Vision Encoder-Decoder Based Model

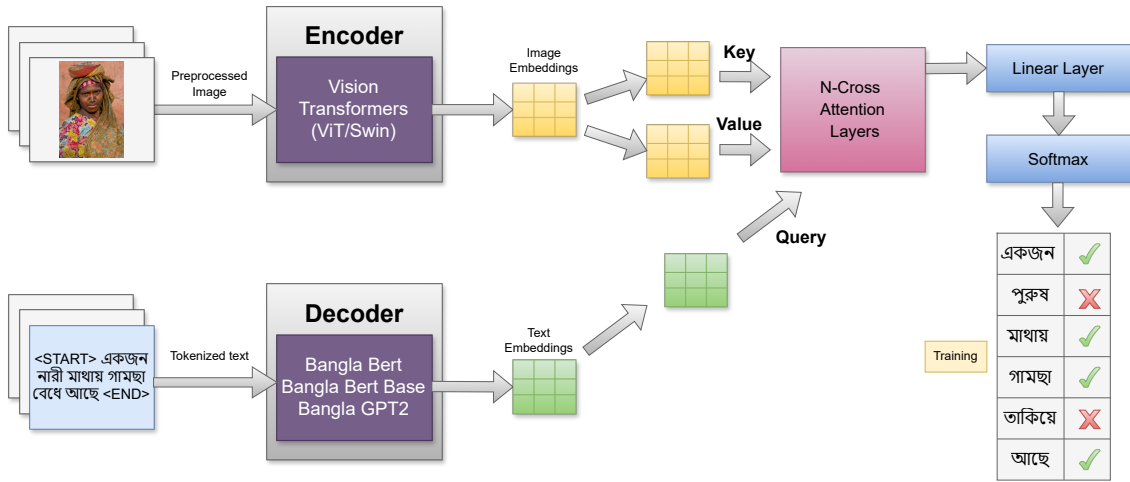


Figure 5.4: Proposed Vision Encoder-Decoder Method.

Vision encoder decoder model connects two models, such as the image encoder model and text sequence generator decoder model. For the image encoder, we used pretrained ViT and Swin vision transformers. For ViT a Base variant model with patch size of 16x16 pixels, input image size of 224x224 which was pretrained on imagenet21k was used. For Swin a Base variant model with patch size of 4x4, window size of 7, input image size of 224x224 which was pretrained on imagenet22k was used. Also, a Tiny Swin variant model not pretrained on imagenet22k was used. For text sequence generator decoder model, we used BanglaBert [26] (Electra model), BanglaBertBase [27] (BERT model) and BanglaGPT [19] (GPT2 model), all of which are language models pretrained in Bangla language. A cross-

attention layer connects the embeddings of an image encoder with a language decoder. Using cross-attentional techniques, the model can successfully incorporate visual and textual input. The image embeddings generated by the encoder are used as Key & Value and the text embeddings generated by the decoder are used as Query in the cross-attention head. With image and text inputs, the model learns to generate the input caption, thus understanding the correlation between image objects and caption words. The output from the linear layer is a vector of size [Sequence Length, Vocabulary Size]. After applying softmax, the final sequence is generated which is then compared to the input caption and through a loss function for incorrect word, loss is calculated. Through back propagation, the loss is minimized. After training is complete, the captions are generated by masking every word except the word after the start token. The model predicts the masked words one by one until it reaches the end token or the maximum output sequence length.

Chapter 6

Experimental Results and Analysis

This chapter details how we implemented two proposed approaches from chapter 5. This section includes an in-depth study of the experimental results achieved through these three methods. The article concludes with examples of Bangla captions made using various approaches and test images.

6.1 Experimental Setups

Since our suggested approaches are deep learning models, a graphics processing unit (GPU) is required to train them. Otherwise, training our models would take several days to complete. We also improved our models' accuracy through hyperparameter adjustment. This section covers the tools used to implement the models and the hyperparameters for each method.

6.1.1 Implementation Details

The codes used for the experiments were run on Jupyter Notebook on a local machine. PyTorch library version 2.1.1 was used to develop our models. Model training and inference was done on NVIDIA RTX 3070 GPU (Driver Version - 551.86) which consists of 5888 CUDA cores along with 8GB GDDR6 VRAM. The clip based models took one to five hours to train and the vision encoder-decoder models took one to seven hours depending on batch size and epochs.

6.1.2 Hyperparameter selection

This section discusses the different hyperparameters utilized for all of our proposed approaches.

- Hyperparameters of CLIP based Model

The model has been trained with a batch size variations of 8,16,32,40. Prefix length for the image encodings after passing through the MLP has been set 10. We constructed MLP using a single layer and with a Tanh activation function. Furthermore, this model was trained for 5,10 epochs using the AdamW Optimizer [28] with a learning rate of $2e-5$ and a custom learning rate scheduler, with 5000 as the warmup step. Additionally, Cross Entropy loss [29] was used as the loss function. The caption generation was done using Beam Search [30] technique where beam size was set to 3,4,5,6 and max length of the sequence was set to 120 along with temperature set to 1.

In Figure 6.1 and Figure 6.2, the training was done using BanglaLekha Dataset over 20 epochs using batch size of 32, here it is clear that the training was saturated thus the model overfitting after 10 epochs as the validation accuracy remain the same after that. Also, the validation loss began to increase instead of decreasing, thus making it clear that overfitting was occurring as training went past over 10 epochs. So, 10 epochs were selected as the maximum number of epochs to train for.

- Hyperparameters of Vision Encoder Decoder Model

The model has been trained with a batch size variations of 4,8,16. Furthermore, this model was trained for 5,10 epochs using the AdamW Optimizer with a learning rate of $5e-05$ and a custom learning rate scheduler, with 1024 as the warmup step. Additionally, Cross Entropy loss was used as the loss function. The caption generation was done using Beam Search technique where beam size was set to 4 and max length of the sequence was set to 30. The size of n-grams that should not be repeated in the generated sequences, set to 3 and length penalty applied during beam search decoding set to 2.

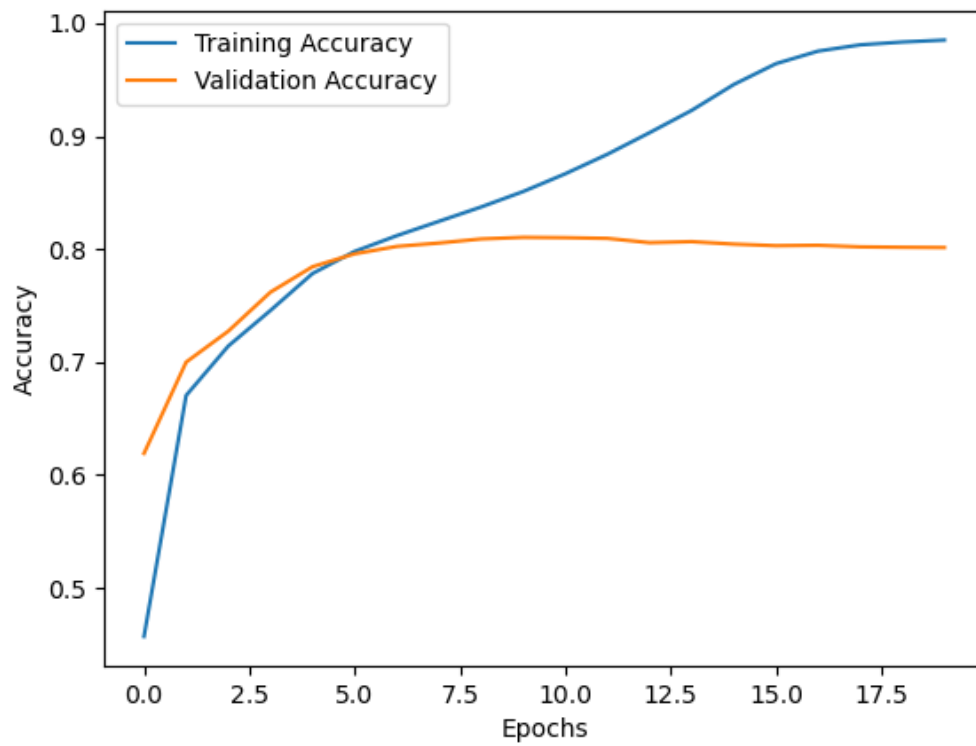


Figure 6.1: Validation accuracy plot for Clip-ViT over 20 epochs.

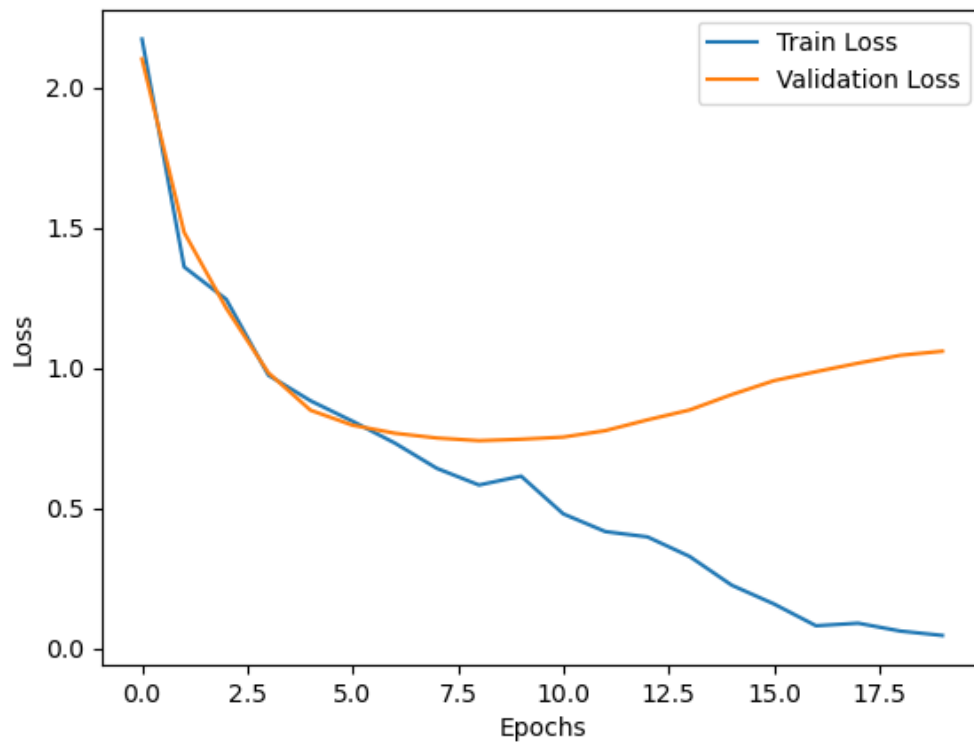


Figure 6.2: Validation loss plot for Clip-ViT over 20 epochs.

6.2 Experimental Results

To ensure accurate captions, it's crucial to compare them to human-annotated captions. We used two evaluation measures (BLEU and METEOR) to validate the accuracy of our proposed models. This section shows the evaluation scores achieved using proposed methodologies.

6.2.1 Experimental Results of the Clip Based Model

From table 6.1, we can observe that as batch size is increased, the scores decrease. Batch size of 8 gave us the best results throughout both ResNet50x4 and ViT image encoders. For BanglaLekha Dataset, BLEU-1, BLEU-2, BLEU-3, BLEU-4 and Meteor scores of respectively 0.688, 0.624, 0.574, 0.529 and 0.38 were the best results observed for both the encoders. The ResNet50x4 encoder performed the best out of both. Furthermore, 10 epochs gave the best scores, compared to 5 epochs for the ResNet50x4 encoder, but the opposite was the case for the ViT image encoder. The best results using the ViT image encoder were 0.682, 0.62, 0.571, 0.526, and 0.372 for BLEU-1, BLEU-2, BLEU-3, BLEU-4, and Meteor scores respectively.

Table 6.1: Results for BanglaLekha Dataset on CLIP Based Model

Encoder	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ResNet50x4	40	10	0.594	0.540	0.510	0.461	0.351
	32	10	0.658	0.6	0.552	0.509	0.365
	16	10	0.674	0.612	0.562	0.517	0.366
		5	0.660	0.603	0.557	0.514	0.366
	8	10	0.688	0.624	0.574	0.529	0.38
		5	0.674	0.614	0.566	0.522	0.38
ViT	40	10	0.583	0.535	0.496	0.459	0.346
	32	10	0.648	0.591	0.544	0.501	0.358
	16	10	0.684	0.618	0.566	0.519	0.37
	8	10	0.68	0.611	0.556	0.507	0.353
		5	0.682	0.62	0.571	0.526	0.372

From table 6.2, for BNature Dataset, BLEU-1, BLEU-2, BLEU-3, BLEU-4 and Meteor scores of respectively 0.819, 0.755, 0.702, 0.655 and 0.409 were the best result observed for both the encoders. The ResNet50x4 encoder performed the best out of both. Furthermore, 10 epochs gave the best scores, compared to 5 epochs for the ResNet50x4 encoder, but the opposite was the case for ViT image encoder. The best results using the ViT image encoder were 0.82, 0.751, 0.69, 0.638 and 0.382 for BLEU-1, BLEU-2, BLEU-3, BLEU-4 and Meteor scores respectively.

From table 6.3, for our Dataset we observe that as batch size is increased, the scores decrease. Batch size of 8 gave us the more favorable results. BLEU-1, BLEU-2, BLEU-3, BLEU-4

Table 6.2: Results for BNature Dataset on CLIP Based Model

Encoder	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ResNet50x4	16	10	0.818	0.754	0.698	0.649	0.409
	8	10	0.819	0.751	0.692	0.641	0.388
		5	0.818	0.755	0.702	0.655	0.4
ViT	16	10	0.82	0.751	0.69	0.638	0.38
	8	10	0.811	0.74	0.679	0.626	0.368
		5	0.819	0.749	0.69	0.638	0.382

and Meteor scores of respectively 0.726, 0.644, 0.585, 0.537 and 0.408 were the best result observed for both the encoders. The ResNet50x4 encoder performed the best out of both. Furthermore, 10 epochs gave the best scores, compared to 5 epochs for both encoders. The best results using the ViT image encoder were 0.72, 0.638, 0.578, 0.531 and 0.406 for BLEU-1, BLEU-2, BLEU-3, BLEU-4 and Meteor scores respectively.

Table 6.3: Results for Our Dataset on CLIP Based Model

Encoder	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ResNet50x4	8	10	0.726	0.644	0.585	0.537	0.408
		5	0.713	0.63	0.571	0.523	0.392
ViT	16	10	0.709	0.626	0.567	0.519	0.4
	8	10	0.72	0.638	0.578	0.5310	0.406
		5	0.705	0.623	0.564	0.515	0.394

We conducted experiment with different beam sizes in table 6.4 for the best performing ResNet50x4 encoder using BanglaLekha Dataset. Using a beam size of 3 the captions were more conservative and not very divisive, thus not giving accurate captions. We see that beam size of 5 gave us the best scores of 0.689, 0.625, 0.575, 0.530 and 0.380 for BLEU-1, BLEU-2, BLEU-3, BLEU-4 and Meteor scores respectively as it captures longer phrases accurately. Thus, the rest of the experiments were conducted with a beam size of 5. But with beam size of 3, BLEU-1 score was the highest observed, meaning it generated captions with more accurate short phrases.

Table 6.4: Results for Different Beam Sizes on CLIP Based Model

Beam Size	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
3	0.692	0.625	0.573	0.526	0.376
4	0.690	0.625	0.574	0.528	0.379
5	0.689	0.625	0.575	0.530	0.380
6	0.689	0.625	0.575	0.530	0.380

6.2.2 Experimental Results of the Vision Encoder-Decoder Model

In table 6.5 comparison between 3 different vision transformers and two different language models. For Swin transformer, a tiny variant with smaller parameters and a base variant with higher parameters are used. As suspected, the base variant performs better than the tiny variant. Moreover, out of the two language models, banglabert the electra based model performs better than the banglabertbase the BERT based model. When comparing ViT and SwinB, ViT gave the best results for lower n-gram BLEU scores, implying that it works best generating accurate short phrases while SwinB works best generating long phrases sequences as shown by higher n-gram BLEU scores. Furthermore, the meteor scores were best for the SwinB encoder, as it was able to generate captions with greater contextual similarity to the reference captions and word order. Also, epochs greater than 10 were giving inferior scores, while 5 giving the best results overall and batch size of 16 was ideal for good performance.

Table 6.5: Results on BanglaLekha Dataset on Vision Encoder-Decoder Based Model

Model	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ViT+ banglabert	16	20	0.659	0.566	0.493	0.435	0.210
		15	0.664	0.570	0.496	0.438	0.206
		10	0.666	0.578	0.508	0.453	0.218
		5	0.691	0.597	0.525	0.468	0.235
	8	5	0.656	0.571	0.504	0.451	0.226
ViT+ bangla-bert-base	16	20	0.681	0.559	0.500	0.383	0.106
		15	0.675	0.552	0.455	0.378	0.105
		10	0.683	0.600	0.461	0.384	0.106
		5	0.666	0.551	0.458	0.386	0.103
	8	5	0.663	0.548	0.455	0.382	0.099
SwinT+ banglabert	16	5	0.655	0.567	0.497	0.442	0.219
	8	5	0.654	0.567	0.500	0.446	0.219
SwinB+ banglabert	16	5	0.664	0.588	0.526	0.476	0.252
	8	5	0.671	0.590	0.524	0.471	0.247

In table 6.6 using the BNature Datset, ViT gave the best results for BLEU-2, BLEU-3, BLEU-4 and Meteor with batch size 8 and 5 epochs, as only the inferior SwinT transformer was used to evaluation performance on this dataset. Also, going beyond 10 epochs didn't result in better performance.

For our dataset, minimal testing was done for the Vision Encoder-Decoder Based Model, as stated in table 6.7. Only ViT encoder with 8 and 16 batch sizes were used for testing, along with 5 epochs training. But our observation was that it was performing similar to the previous model, getting scores between the other 2 datasets in average.

In table 6.8, there the results are the accurate visualization of how a large language model should be performing as they perform very good at higher n-gram BLEU scores as they are

Table 6.6: Results on BNature Dataset on Vision Encoder-Decoder Based Model

Model	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ViT+ banglabert	16	5	0.775	0.705	0.642	0.588	0.3
	8	5	0.787	0.712	0.647	0.591	0.303
SwinT+ banglabert	16	10	0.790	0.710	0.640	0.579	0.288
		5	0.754	0.681	0.617	0.562	0.285
	8	15	0.780	0.689	0.611	0.546	0.256
		10	0.781	0.696	0.622	0.561	0.281
		5	0.763	0.685	0.616	0.559	0.277

Table 6.7: Results on Our Dataset on Vision Encoder-Decoder Based Model

Model	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ViT+ banglabert	16	5	0.731	0.631	0.563	0.511	0.322
ViT+ banglabert	8	5	0.734	0.636	0.5695	0.517	0.331

good at generating longer phrases accurately. Also, the meteor scores are not high enough for a LLM. There seems to be an issue with generated captions in this model compared to the previous model results and needs further testing and evaluation.

Table 6.8: Results on BanglaLekha Dataset on Vision Encoder-Decoder Based Model

Model	Batch Size	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
ViT+BanglaGPT	4	20	0.680	0.567	0.487	0.424	0.228
		15	0.675	0.562	0.480	0.417	0.216
		10	0.672	0.556	0.473	0.409	0.212
		5	0.674	0.563	0.485	0.425	0.228
	8	20	0.678	0.566	0.486	0.423	0.225
		15	0.670	0.554	0.471	0.407	0.202
		10	0.677	0.561	0.478	0.414	0.210
		5	0.682	0.572	0.493	0.431	0.240

6.2.3 Experimental Results of the Transformer Based Model

We reconstructed the Transformer Based model used in [2] using the same split as we did on our proposed models on BanglaLekha dataset, to do an apples to apples comparison. We used a batch size of 32 in training. Here, we only found the necessity to recreate the results for 2 different number of head, as further hyperparameter tweaking was already observed on the reference paper. Our findings were just as we expected, since slightly greater margins in BLEU-1(7%) and BLEU-2(13.31%) scores were observed compared to our best performing models. The same goes for BNature dataset, where 6.7% better BLEU-1, 12.36% better BLEU-2, 18.18% better BLEU-3, 23.89% better BLEU-4 and 41% better Meteor scores were observed.

Table 6.9: Results on BanglaLekha Dataset on Transformer Based Model

Model	Head	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
InceptionV3+ Transformer	1	10	0.644	0.547	0.475	0.415	0.256
	2	10	0.631	0.536	0.466	0.408	0.247

Table 6.10: Results on BNature Dataset on Transformer Based Model

Model	Head	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
InceptionV3+ Transformer	1	10	0.765	0.668	0.585	0.516	0.264
	2	10	0.74	0.635	0.549	0.478	0.216

For our dataset, similar performance was observed compared to our proposed models. Our dataset performed the best in terms of Meteor score, showing an 18% to 21% lead over BNature and BanglaLekha. Furthermore, our dataset managed to outperform BanglaLekha dataset, showing 6% and 8% increase in BLEU-3 and BLEU-4 scores.

Table 6.11: Results on Our Dataset on Transformer Based Model

Model	Head	Epoch	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
InceptionV3+ Transformer	1	10	0.663	0.554	0.476	0.417	0.288
	2	10	0.672	0.575	0.505	0.450	0.316

6.2.4 Comparison of Existing Models and Our Proposed Models

We have compared our best performing CLIP Based model and Vision Encoder-Decoder Based model with previous or existing models to evaluation our performance as a whole on the same dataset, even if the splits were not the same.

On BanglaLekha Dataset, our CLIP Based model achieved 2.2% better BLEU-1, 12% better BLEU-2, 19% better BLEU-3, 26% better BLEU-4 and 25% better Meteor scores than the next best metrics for existing models. On BNature Dataset, our CLIP Based model achieved 30% better BLEU-1, 41% better BLEU-2, 47% better BLEU-3, 51.34% better BLEU-4 and 33% better Meteor scores than the next best metrics for existing models.

On BanglaLekha Dataset, our Vision Encoder-Decoder Based model achieved 2.5% better BLEU-1, 7.28% better BLEU-2, 9.79% better BLEU-3, 14% better BLEU-4 scores than the next best metrics for existing models. On BNature Dataset, our Vision Encoder-Decoder Based model achieved 26% better BLEU-1, 35% better BLEU-2, 39% better BLEU-3, 41.47% better BLEU-4 and 5.42% better Meteor scores than the next best metrics for existing models.

Big margins were seen in higher n-gram BLEU scores for our models, meaning longer sequence of words are matching more to the reference captions, and it captures the context and coherence over longer sequences of words. Thus, the model is capable of generating accurate long, specific phrases. The greater meteor scores implies that the model is able to capture semantic similarity between the generated and reference captions, as meteor evaluates not only exact word matches but also considers synonyms and different word forms. This indicates that the model performs well in capturing both the meaning and structure of the text.

So, Our proposed model far outperformed the current benchmark in terms of same publicly available datasets. Thus fulfilling our goal while proving the worth of Large Language models in image captioning for bangla language.

Table 6.12: A brief comparison of BLEU and Meteor scores for existing models and our proposed models using the BanglaLekha dataset.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
BanglaLekha	InceptionV3+Transformer [2]	0.665	0.556	0.476	0.408	0.255
	Xception+BiGRU [2]	0.674	0.527	0.454	0.344	-
	VGG-16+LSTM [3]	0.667	0.436	0.315	0.238	-
	CNN-ResNet-50 [5]	0.651	0.426	0.278	0.175	0.297
	Clip-ResNet+BanglaGPT (Our Model)	0.689	0.625	0.575	0.530	0.381
	ViT+banglabert (Our Model)	0.691	0.598	0.525	0.469	0.236

Table 6.13: A brief comparison of BLEU and Meteor scores for existing models and our proposed models using the BNature dataset.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
BNature	InceptionV3+BiGRU [9]	0.426	0.279	0.236	0.164	0.287
	InceptionV3+BiGRU [10]	0.606	0.501	0.437	0.388	-
	Clip-ResNet+BanglaGPT (Our Model)	0.818	0.756	0.702	0.656	0.400
	ViT+BanglaBert (Our Model)	0.787	0.713	0.648	0.591	0.303

6.3 Summary

This chapter discusses the experimental setups for two of our proposed techniques. Furthermore, the experimental findings and Bengali captions generated by two of the proposed techniques utilizing test images are presented here. The CLIP-based and Vision Encoder-Decoder using BERT/Electra models performed similarly, however the CLIP-based method performed the best. Some generated captions for the models tested using random test split photos are shown in Figure 6.3, 6.4 and 6.5.

	BanglaLekha Dataset
	<p>RN50x4 : তিনটি শিশু আছে।</p> <p>ViT: চারজন মানুষ আছে।</p>
	<p>RN50x4 : একজন পুরুষ বক্তৃতা দিচ্ছে।</p> <p>ViT: একজন পুরুষ কাজ করছে।</p>
	BNature Dataset
	<p>RN50x4 : একটি ছেলে পানিতে দাড়িয়ে আছে।</p> <p>ViT: একটি শিশু পানিতে আছে।</p>
	<p>RN50x4 : রাস্তায় কিছু রিকশা আছে</p> <p>ViT: রাস্তায় অনেক মানুষ আছে।</p>
	Our Dataset
	<p>RN50x4 : একটি সাগরের মধ্যে একটি বড় নৌকা দেখা যাচ্ছে।</p> <p>ViT: একটি পুকুরের মধ্যে অনেক গাছ দেখা যাচ্ছে।</p>
	<p>RN50x4 : এটি দেখা যাচ্ছে যে বর্ষাকালে রাস্তাঘাটে পানি ডুবে গিয়েছে।</p> <p>ViT: বর্ষা মৌসুমে সড়ক পানি ডুবে গেছে।</p>

Figure 6.3: Illustration of captions generated by best performing CLIP Based Architecture using BanglaLekha, BNature and our datasets.

	<p>ViT + bangla-bert-base : রাসতা দিঘে কযেকজন মানষ চলাচল করছে ।</p> <p>ViT + bangla-bert : রাস্তা একজন মানুষ হেঁটে যাচ্ছে । দূরে একটি সেতু আছে ।</p> <p>Swin : একটি সেতু আছে । দূরে আছে ।</p>
	<p>ViT + bangla-bert-base : একজন বযসক পরষ আছে ।</p> <p>ViT + bangla-bert : একজন পুরুষ আছে ।</p> <p>Swin : একজন পুরুষ হাসি মুখে আছে ।</p>
	<p>ViT + bangla-bert-base : সমদরের পাড দিঘে অনেক মানষ হেটে যাচ্ছে ।</p> <p>ViT + bangla-bert : সমুদ্রের পারে অনেকগুলো মানুষ আছে ।।</p> <p>Swin : সমুদ্রের পারে অনেকগুলো মানুষ আছে ।</p>
	<p>ViT + bangla-bert-base : একজন নারী ও একটি শিশ আছে ।</p> <p>ViT + bangla-bert : একজন নারী বসে কাজ করছে ।</p> <p>Swin : একজন নারী বসে কাজ করছে ।।</p>
	<p>ViT + bangla-bert-base : একটি হরিণ ও কযেকটি হরিণ আছে ।</p> <p>ViT + bangla-bert : একটি হরিণ ও একটি হরিণ আছে ।</p> <p>Swin : একটি হরিণ আছে ।</p>

Figure 6.4: Illustration of captions generated by best performing Vision Encoder-Decoder Based Architecture using BanglaLekha dataset.






BNature Dataset	
	<p>Swin + bangla-bert: একটি বানর বসে আছে ।</p> <p>ViT + bangla-bert : একটি বানর আছে</p>
	<p>Swin + bangla-bert: একজন মানুষ বসে আছে ।</p> <p>ViT + bangla-bert : একজন মানুষ ক্যামেরা ছবি তুলছে ।</p>
	<p>Swin + bangla-bert: একটি গাছের নিচে একজন মানুষ বসে আছে</p> <p>ViT + bangla-bert : একজন মানুষ বসে আছে ।</p>
Our Dataset	
	<p>ViT + bangla-bert(BS 16): সমুদ্রে বেশ কিছু নৌকা দেখা যায় ।</p> <p>ViT + bangla-bert(BS 8) : নদীর উপর বেশ কয়েকটি নৌকা ভেসে বেড়াচ্ছে ।</p>
	<p>ViT + bangla-bert(BS 16): দুটি গরু সামনের দিকে এগিয়ে যাচ্ছে ।</p> <p>ViT + bangla-bert(BS 8) : দুটি গরু তাদের ঘাস খাচ্ছে এবং তাদের পাশে একটি লোক দাঁড়িয়ে আছে ।</p>

Figure 6.5: Illustration of captions generated by best performing Vision Encoder-Decoder Based Architecture using BNature and our dataset.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

We have come close to filling our objectives. The dataset we created have shown good results compared to BanglaLekha and BNature datasets. So, it will enrich the Bangla image caption dataset space. Our proposed models performed as we expected, getting benchmarks performance on both publicly available datasets tested on both models. The vision encoder-decoder model needs further testing and tweaking as we believe more performance can be extracted from that model. Our clip based model has proven to be the best models available for Bangla image captioning, showcasing impressive higher n-gram BLEU scores and Meteor scores. Large language models in Bangla image captioning has now been tested to be the new state-of-the-art along with the help of CLIP model image encoder which has been trained on vast amount of data greater than ImageNet [31]. Bert and Electra Bangla pretrained models also have promising results in creating meaningful captions but lagged slightly behind GPT-2 in terms of accurate long sequence and contextual text generation.

7.2 Future Work

- Increasing the size of our dataset to 20000 images along with five captions for each image.
- In the vision encoder-decoder model, training the decoder & tokenizer before using in the final model to minimize unknown words in the vocabulary.
- Replace GPT-2 with BERT in our CLIP based model.
- Further tweaking hyper-parameters of our model and fine-tuning it to achieve better results.

References

- [1] N. M. S. M. Matiur Rahman, Nabeel Mohammed, “Chittron: An automatic bangla image captioning system,” in *10.1016/j.procs.2019.06.100*.
- [2] F. M. Shah, “Bornon: Bengali image captioning with transformer based deep learning approach,” in *arXiv preprint arXiv:2109.05218 (2021)*.
- [3] A. H. Kamal, M. A. Jishan, and N. Mansoor, “Textmage: The automated bangla caption generator based on deep learning,” in *2020 International Conference on Decision Aid Sciences and Application (DASA)*, pp. 822–826, 2020.
- [4] M. A. H. Palash, “Bangla image caption generation through cnnttransformer based encoder-decoder network,” in *arXiv preprint arXiv:2110.12442 (2021)*.
- [5] M. F. Khan, S. M. S. Shifath, and M. S. Islam, “Improved bengali image captioning via deep convolutional neural network based encoder-decoder model,” *CoRR*, vol. abs/2102.07192, 2021.
- [6] C. Yan, “Task-adaptive attention for image captioning,” in *IEEE Transactions on Circuits and Systems for Video technology 32.1 (2021)*, pp. 43–51. *Intelligence. Springer. 2021*, pp. 217–229.
- [7] W. Liu, “Cptr: Full transformer network for image captioning,” in *arXiv preprint arXiv:2101.10804 (2021)*.
- [8] M. Li, T. Lv, J. Chen, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, “Trocr: Transformer-based optical character recognition with pre-trained models,” 2022.
- [9] M. M. A. M. R. F. M. K. M. Al Momin Faruk, Hasan Al Faraby, “Image to bengali caption generation using deep cnn and bidirectional gated recurrent unit,” in *arXiv:2012.12139v1 [cs.CV] 22 Dec 2020*.
- [10] M. Rodas Solomon, “Amharic language image captions generation using hybridized attention-based deep neural networks,” in *Applied Computational Intelligence and Soft Computing*, vol. 2023, Article ID 9397325, 11 pages, 2023.

- [11] L. Pulmano, L. Joykutty, and J. Caulkins, "Python-based prediction of rapid intensification from mimic-tc ensemble (prime)," *Journal of Student Research*, vol. 11, 03 2023.
- [12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.
- [13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
- [14] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," 2021.
- [15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.
- [16] B. Ali, N. Benblidia, F. Reguieg, M. Bouakkaz, and H. Felouat, "An arabic visual speech recognition framework with cnn and vision transformers for lipreading," *Multimedia Tools and Applications*, 02 2024.
- [17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.
- [18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.
- [19] M. S. Salim, H. Murad, D. Das, and F. Ahmed, "Banglagpt: A generative pretrained transformer-based model for bangla language," in *2023 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD)*, pp. 56–59, 2023.
- [20] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," 2020.
- [21] T. W. W. Z. K. Papineni, S. Roukos and Y. Heights, "Ibm research report bleu : a method for automatic evaluation of machine translation," in *Science (80-.)*, vol. 22176, no. February, pp. 1–10, 2001, doi: 10.3115/1073083.1073135.
- [22] M. Denkowski and A. Lavie, "Meteor universal: Language specific translation evaluation for any target language," in pp. 376–380, 2015, doi: 10.3115/v1/w14-3348.
- [23] A. V. X. H. S. B. Yin Cui, Yin Cui, "Learning to evaluate image captioning," in *arXiv:1806.06422v1 [cs.CV]* 17 Jun 2018.

- [24] C. M. C. J. C. C. J. H. S. L. Bryan A. Plummer, Liwei Wang, “Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models,” in *arXiv:1505.04870 [cs.CV]*.
- [25] S. B. L. B. R. G. J. H. P P D. R. C. L. Z. P D. Tsung-Yi Lin, Michael Maire, “Microsoft coco: Common objects in context,” in *arXiv:1405.0312 [cs.CV]*.
- [26] A. Bhattacharjee, T. Hasan, W. U. Ahmad, K. Samin, M. S. Islam, A. Iqbal, M. S. Rahman, and R. Shahriyar, “Banglabert: Language model pretraining and benchmarks for low-resource language understanding evaluation in bangla,” 2022.
- [27] S. Sarker, “Banglabert: Bengali mask language model for bengali language understanding,” 2020.
- [28] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” 2019.
- [29] A. Mao, M. Mohri, and Y. Zhong, “Cross-entropy loss functions: Theoretical analysis and applications,” 2023.
- [30] M. Freitag and Y. Al-Onaizan, “Beam search strategies for neural machine translation,” in *Proceedings of the First Workshop on Neural Machine Translation*, Association for Computational Linguistics, 2017.
- [31] 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*, pp. 248–255, 2009.

Generated using Undergraduate Thesis L^AT_EX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This thesis was generated on Wednesday 17th April, 2024 at 6:05am.