**IIIT, Hyderabad Talent Sprint AI/ML Executive Post Graduation Programme**

**(Cohort B22, Group 7)**

**Team Members**

Kusal Degapudi

Pradeep Reddy Pochareddy

Raja Rama Krishna B

Sai Nikhil Yanduri

**Mentor**

Seshadri Mazumder

**Professor**

Vineeth Gandhi

Image Captioning



Contents

[1. Abstract 2](#_Toc178485057)

[2. Introduction 2](#_Toc178485058)

[3. Problem Statement 2](#_Toc178485059)

[4. Related Works 2](#_Toc178485060)

[5. Methodology 3](#_Toc178485061)

[Dataset Description & Data pre-processing 3](#_Toc178485062)

[Dataset Versions 3](#_Toc178485063)

[Data Pre-processing 3](#_Toc178485064)

[Image Pre-processing 3](#_Toc178485065)

[Text Pre-processing 3](#_Toc178485066)

[Model Architecture 4](#_Toc178485067)

[Encoder 4](#_Toc178485068)

[Decoder 4](#_Toc178485069)

[6. Experiments 4](#_Toc178485070)

[Models Evaluated 5](#_Toc178485071)

[RESNET-LSTM 5](#_Toc178485072)

[CLIP-LSTM 5](#_Toc178485073)

[CLIP with Attention 5](#_Toc178485074)

[Evaluation Metrics 5](#_Toc178485075)

[Experiment 1- RESNET – LSTM Trained on 8K Dataset 5](#_Toc178485076)

[Experiment 2- RESNET – LSTM Trained on 30K Dataset 5](#_Toc178485077)

[Experiment 3- CLIP – LSTM Trained on 8K Dataset 6](#_Toc178485078)

[Experiment 4- CLIP – LSTM Trained on 30K Dataset 6](#_Toc178485079)

[Experiment 5- CLIP – LSTM with Multi head Attention Trained on 8K Dataset 7](#_Toc178485080)

[Experiment 6- CLIP – LSTM with Multi head Attention Trained on 30K Dataset 7](#_Toc178485081)

[7. Results 8](#_Toc178485082)

[Training Scores 8](#_Toc178485083)

[Test Scores 8](#_Toc178485084)

[8. Discussions and Model Baselining 9](#_Toc178485085)

[Observations 9](#_Toc178485086)

[Model Baselining 9](#_Toc178485087)

[Future Steps 9](#_Toc178485088)

[9. Deployment and Model Availability 10](#_Toc178485089)

[Deployment 10](#_Toc178485090)

[Availability 10](#_Toc178485091)

# Abstract

Image captioning is an interdisciplinary field that combines computer vision and natural language processing to automatically generate textual descriptions for images. This project explores state-of-the-art deep learning techniques for creating a model that can effectively bridge the gap between visual content and language, aiming to contribute to the advancement of automated captioning systems.

# Introduction

Image captioning has numerous practical applications, including creating accessibility tools for the visually impaired, generating automated descriptions for image indexing and search engines, improving content-based image retrieval, and even aiding in human-computer interaction. However, despite significant advancements, challenges such as generating captions with detailed context, handling complex scenes, and ensuring the diversity of generated captions remain key areas of ongoing research.

# Problem Statement

The core problem is how to design a system that can effectively generate human-like captions that not only describe the objects within an image but also capture their relationships, actions, and the overall context.

Key challenges include:

* Handling complex and diverse visual scenes with multiple objects and actions.
* Generating grammatically correct, contextually accurate, and semantically rich descriptions.
* Addressing ambiguity in images, where multiple captions could be valid or where object recognition is difficult due to occlusions or visual noise.
* Ensuring the generated captions are informative and descriptive, rather than generic or overly simplistic.

# Related Works

Image captioning has seen rapid advancements in recent years due to the development of deep learning methods that combine visual and textual understanding. Early approaches to image captioning were rule-based systems that required hand-crafted features and templates to generate descriptions. However, with the rise of neural networks, more effective models have been developed, integrating computer vision with natural language processing.

In addition to model architecture improvements, large-scale datasets have been critical to advancing the field. Datasets like Flickr30k provide rich annotations of images and have become standard benchmarks for evaluating the performance of image captioning systems. These datasets contain diverse scenes and captions that allow models to learn complex relationships between objects, actions, and contextual descriptions.

Despite these advancements, several challenges remain. Current models still struggle with understanding fine-grained details, producing diverse captions, and accurately capturing the relationships between objects in complex scenes. Moreover, models often produce repetitive or overly simplistic captions when faced with ambiguity in images. Recent research focuses on addressing these limitations by exploring graph-based neural networks, self-supervised learning, and improved contextual reasoning techniques.

# Methodology

The process of developing an image captioning model involves integrating deep learning techniques from both computer vision and natural language processing. The methodologies employed in this project can be broadly divided into three phases: data pre-processing, model design, and evaluation.

## Dataset Description & Data pre-processing

The Flickr Image Captioning Dataset is a widely used to benchmark for evaluating image captioning models. The dataset consists of images sourced from Flickr, an online photo-sharing platform, and is paired with textual descriptions. Each image is associated with multiple human-generated captions, making it suitable for tasks that require generating descriptive captions from visual inputs.

### Dataset Versions

* Flickr8K: Contains 8,092 images, each accompanied by 5 captions and token size of 8787.
* Flickr30K: Contains 31,783 images, each paired with 5 captions and token size of 19775.

The images are sourced from a wide range of Flickr users, covering a broad spectrum of content, including people, animals, objects, natural scenes, and various daily-life activities. The captions are provided by human annotators and describe the main objects, actions, and context depicted in the images.

### Data Pre-processing

A large annotated dataset is essential for training an image captioning model. For this project, both Flickr8K and Flickr30K datasets were used for training the models. Data pre-processing steps done for both captions and images.

### Image Pre-processing

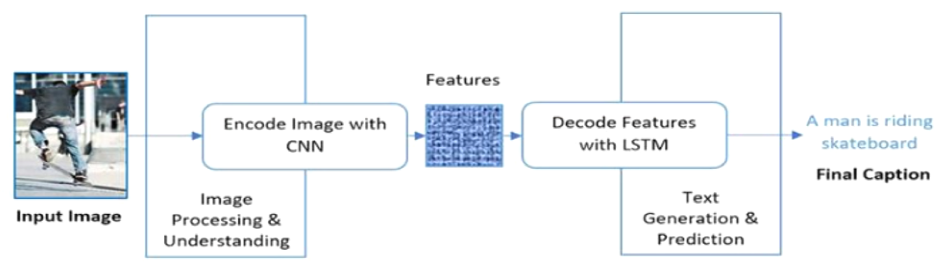
* Images are first resized to 224X224 to ensure consistency across the dataset.
* Visual features are extracted from the images using pre-trained convolutional neural networks (CNNs) such as **ResNet-50**.
* Some of the experiments conducted by extracting CLIP embedding for the images by using model “**ViT-B/32**”.
* These embedding are stored in Pickle file and used the stored embedding, to train the model and to save time in model training.
* These embedding serve as the encoder in the image captioning system, transforming each image into a fixed-length feature vector.

### Text Pre-processing

* Captions were processed using regular expressions and cleared the special characters, white spaces and converted to lower cases.
* Curated captions undergo tokenization, where sentences are split into individual words.
* A vocabulary is constructed based on the most frequent words, and each word is then converted into an integer index. Special tokens such as <start>, <end>, and <pad> are used to signal the beginning, end, and padding of captions.

## Model Architecture

The image captioning model typically follows an encoder-decoder architecture, where the encoder processes the image and the decoder generates the textual description.



### Encoder

* Convolutional Neural Networks (CNN) - ResNet50/CLIP (**ViT-B/32**) is used to extract the features from the image.

### Decoder

* In this project, Long Short-Term Memory (LSTM) and attention techniques were used to model the sequential dependencies between words in the caption.

# Experiments

Multiple experiments were conducted in baselining the image captioning model. Image features extracted using pre trained model passed as input to decoder models. Decode models optimized with different hyper parameters and employed different network architectures in training and inferencing. In this project, we employed several libraries to facilitate the development and execution of our image captioning model. The primary libraries used includes,

* **PyTorch** we utilized PyTorch as our main deep learning framework for building and training our neural network models. PyTorch provides dynamic computation graphs, which are beneficial for experimenting with different architectures in our image captioning task.
* **PIL** (Python Imaging Library): We used PIL to handle image processing tasks such as loading, resizing, and transforming images into formats suitable for model input.
* **Pickle** This library was employed for serialization and deserialization of Python objects. We used it to save our trained model and to load it efficiently for inference.
* **NLTK** (Natural Language Toolkit): We leveraged NLTK for various natural language processing tasks and calculating BLEU scores.
* **spaCy** used for caption processing and tokenization
* **Streamlit** This library was utilized to create a web application for demonstrating our image captioning model.
* **ngrok** used for deployment in public domain, making it accessible for testing and demonstration purposes.

We evaluated the performance of three different image captioning models on the Flickr8K and Flickr30K datasets. Each model was tested for its ability to generate accurate captions based on BLEU scores and similarity metrics.

## Models Evaluated

### RESNET-LSTM

* Image features extracted using a pre-trained RESNET model.
* Captions generated using an LSTM decoder.

### CLIP-LSTM

* Features extracted using the CLIP (ViT-B/32) model.
* Captions generated using an LSTM decoder.

### CLIP with Attention

* Features extracted using the CLIP (ViT-B/32) model.
* Captions generated using an LSTM decoder with additional multi head attention layer.

## Evaluation Metrics

* BLEU2, BLEU3 and BLEU4 scores calculated to assess the model performance after each training epoch on validation data, also calculated on test data after completion of training.
* Sentence similarity scores calculated between actual and predicted caption after each training epoch on validation data, and on test data after completion of training.
* **bert-base-nli-mean-tokens** model was used to calculate similarity scores.

## Experiment 1- RESNET – LSTM Trained on 8K Dataset

|  |  |
| --- | --- |
| * RESNET50 last layer altered to get feature vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers - 3 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 8787 * Batch size - 50 |
| **Training Metrics** | **Training loss** |

## Experiment 2- RESNET – LSTM Trained on 30K Dataset

|  |  |
| --- | --- |
| * RESNET50 last layer altered to get feature vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers - 3 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 19775 * Batch size - 50 |
| **Training Metrics** | **Training loss** |

## Experiment 3- CLIP – LSTM Trained on 8K Dataset

|  |  |
| --- | --- |
| * CLIP **ViT-B/32** model used to extract image embedding vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers - 3 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 8787 * Batch size - 50 |
| **Training Metrics** | **Training loss** |

## Experiment 4- CLIP – LSTM Trained on 30K Dataset

|  |  |
| --- | --- |
| * CLIP **ViT-B/32** model used to extract image embedding vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers - 3 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 19775 * Batch size - 50 |
| **Training Metrics** | **Training loss** |

## Experiment 5- CLIP – LSTM with Multi head Attention Trained on 8K Dataset

|  |  |
| --- | --- |
| * CLIP **ViT-B/32** model used to extract image embedding vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers – 3 * Number of Multi-heads – 8 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 8787 * Batch size - 3 |
| **Training Metrics** | **Training loss** |

## Experiment 6- CLIP – LSTM with Multi head Attention Trained on 30K Dataset

|  |  |
| --- | --- |
| * CLIP **ViT-B/32** model used to extract image embedding vector of size 512. * LSTM hidden Size - 512 * Number of LSTM Layers – 3 * Number of Multi-heads – 8 | * Optimizer - Adam * Learning Rate - 0.0001 * Loss Function - Cross-entropy * Linear output size – 8787 * Batch size - 300 |
| **Training Metrics** | **Training loss** |

# Results

## Training Scores

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Score | **8K Dataset Model** | | | **30K Dataset Model** | | |
| CLIP ATTN | CLIP LSTM | RES LSTM | CLIP ATTN | CLIP LSTM | RES LSTM |
| BLEU2 | 0.37 | 0.38 | 0.35 | 0.42 | 0.42 | 0.34 |
| BLEU3 | 0.19 | 0.2 | 0.18 | 0.24 | 0.24 | 0.17 |
| BLEU4 | 0.09 | 0.1 | 0.08 | 0.12 | 0.12 | 0.08 |
| Similarity | 0.71 | 0.66 | 0.61 | 0.69 | 0.7 | 0.59 |

## Test Scores

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Score | **8K Dataset Model** | | | **30K Dataset Model** | | |
| CLIP ATTN | CLIP LSTM | RES LSTM | CLIP ATTN | CLIP LSTM | RES LSTM |
| BLEU2 | 0.36 | 0.39 | 0.34 | 0.42 | 0.41 | 0.32 |
| BLEU3 | 0.19 | 0.20 | 0.17 | 0.24 | 0.24 | 0.15 |
| BLEU4 | 0.08 | 0.09 | 0.08 | 0.12 | 0.12 | 0.07 |
| Similarity | 0.70 | 0.66 | 0.61 | 0.68 | 0.70 | 0.52 |

# Discussions and Model Baselining

## Observations

* **Richer Features with CLIP:** The CLIP model provided richer features compared to ResNet, leading to better captioning inferences with the same decoder model.
* **Improvement with Multi-head Attention:** Adding a multi-head attention layer in the decoder improved grammatical correctness and increased BERT similarity scores for captions.
* **Impact of Batch Size:** Model accuracy was sensitive to batch size. Larger batch sizes required more epochs for improved accuracy, while smaller batch sizes increased training time per epoch but resulted in better accuracy by the end of training.
* **Compute Intensity and GPU Costs:** Training image captioning models is compute-intensive, raising GPU costs. To mitigate this, image embedding were pre-processed and stored in a pickle file for faster training and validation.
* **BLEU Score Observations:** BLEU-3 and BLEU-4 scores were lower due to exact word sequence mismatches in some captions, which led to zero scores. However, BERT scores were higher, indicating that many captions were still of high quality.
  + Out of 1618 observations, 1121 had BLEU4 scores of zero, but the average similarity score was 0.73, with 720 observations achieving a similarity score above 0.70.
  + An example mismatch includes the predicted caption "a group of dogs are running in a field" and actual captions like "A man with four running dogs in nature" and others, reflecting context but not exact word matches.
* **Ambiguity in Actions or Context**: The model struggles when it comes to capturing the finer details of what’s happening in a scene. For example, if someone is riding a specific type of motorcycle, the model might just say “someone is riding through mud,” missing important context like the type of motorcycle or the precise action. It tends to generalize when the action is more detailed or nuanced.
* **Object Recognition Errors**: The model sometimes gets confused when there are multiple or less common objects in the image. For instance, it might describe a camel as a horse or forget to mention something important like a dog carrying a stick. It can miss or misidentify key objects, which makes the caption less accurate.
* **Simplified Descriptions**: When there’s a lot happening in a scene or multiple subjects in the frame, the model often picks up on only the most obvious thing and ignores the rest. It might describe a person standing without mentioning the interaction with other objects or people, making the caption feel incomplete or overly simple.
* **Incorrect Identifications of Animals or People**: Sometimes, the model just gets it wrong when describing animals or people. For instance, it might call a “blonde dog digging a hole” a “white dog standing up,” which changes the scene completely. These errors can make the caption feel out of touch with what’s actually happening.

## Model Baselining

Based on above results both 8K and 30K CLIP-LSTM with Attention network model performing better. This can be used for further improvements.

## Future Steps

* Decoder can be enhanced to use transformer architecture for better results.
* Create Encoder model for getting image features.

# Deployment and Model Availability

## Deployment

We deployed the model using ngrok and Streamlit python libraries, below is the screenshot for the same.

|  |  |
| --- | --- |
| C:\Users\RBHUVANA\Downloads\WhatsApp Image 2024-09-28 at 9.17.49 PM.jpeg | C:\Users\RBHUVANA\Downloads\WhatsApp Image 2024-09-28 at 9.18.16 PM.jpeg |

## Availability

We created GitHub repository and uploaded all the documents along with model source code and made publicly available. <https://github.com/RRKBHUVANA/ImageCaptioningIIIT>