AIML Capstone Project Proposal

Project Title: Image- Multimodal Media Retrieval and Captioning System

Team Name: Group 1

Course: AIML PGCP

Mentor: Sheshadri mazumdar

Supervisor: Prof . Vineet Gandhi

Date: 24/05/2025

Group Members:

- Aditi Chatterjee

- Jagadeesh Kumar Malla

- Lok Subhash Nammina

- Subrahmanyam Bhavanibhatla

# 1. Introduction

Abstract: Multimodal Media Retrieval and Captioning System

In an era where digital media is growing exponentially, the need for intelligent systems that can effectively search, understand, and describe multimedia content has become increasingly vital. This project introduces a novel Multimodal Retrieval and Captioning System that bridges the semantic gap between visual data and natural language by integrating image-text embedding models with caption generators. The system is designed to enable efficient retrieval of relevant images based on textual prompts , generate human-like descriptions for images, thus supporting advanced applications in content indexing, accessibility, and human-computer interaction.

**Understanding Multi-Modal Retrieval**

Multi-modal retrieval enables searching for information using various types of input, such as text and images. This approach is especially valuable in applications like visual search, content-based recommendation systems, and multimedia analysis.

*Key Components of Multi-Modal Retrieval:*

* Indexing: Generating embeddings for both textual and image data.
* Querying: Handling user inputs and identifying closely related matches.

Our approach leverages the powerful CLIP (Contrastive Language–Image Pre-training) model to learn joint embeddings of text and images, enabling semantic-level similarity matching across modalities. This foundation is further enhanced with a Transformer-based captioning model trained on large-scale datasets, ensuring descriptive accuracy and contextual richness in the generated captions. The combination of retrieval and captioning facilitates a feedback loop where retrieved images are immediately interpretable through captions, significantly improving user experience and system transparency.

The system architecture supports scalability, modular deployment, and real-time inference, making it adaptable for applications such as educational platforms, digital asset management, surveillance review systems, and accessibility tools for the visually impaired. Evaluation metrics including BLEU, ROUGE, and METEOR are used to assess caption quality, while retrieval performance is validated through *Accuracy* . Overall, the proposed system presents a significant step towards unified multimedia understanding by seamlessly connecting visual and linguistic data for intelligent and interactive information systems.

# 2. Task 1: Image-Text Retrieval

## 2.1. Literature Review

Image-text retrieval has gained traction with advancements in dual-encoder architectures such as CLIP by OpenAI. These models encode images and text in the same embedding space to compute semantic similarity using cosine similarity.

## 2.2. What is CLIP and Why We Use It

CLIP (Contrastive Language–Image Pre-training) is a model developed by OpenAI that learns visual concepts from natural language supervision. It aligns images and text in a shared embedding space using contrastive learning.The architecture consists of two separate encoders—one for images (usually a Vision Transformer or ResNet) and one for text (typically a Transformer-based model).\*These encoders generate feature embeddings for both modalities.

It is trained to connect images and text by jointly embedding them into the same latent space. CLIP utilizes a large dataset of image-text pairs and employs a contrastive loss to align image and text embeddings.  
We are using CLIP in our project because:  
- It enables zero-shot image classification and retrieval by directly comparing text queries with image features.  
- It is pre-trained on a large and diverse dataset, making it highly generalizable.  
- It allows semantic similarity comparisons, which is essential for retrieving the correct image given a text or vice versa.



* A large dataset of image-caption pairs is used.
* Each caption (e.g., “pepper the aussie pup”) is passed through a Text Encoder to generate text embeddings T₁ to Tₙ.
* Each corresponding image is processed through an Image Encoder, producing image embeddings I₁ to Iₙ.
* A similarity matrix is computed between all image and text embeddings (e.g., dot product or cosine similarity) to align matching pairs closely while pushing apart non-matching pairs.
* The goal is to maximize similarity for matching image-text pairs and minimize it for mismatches—this is achieved using contrastive loss.
* *Dataset Classifier from Label Text :*During inference or classification tasks, class labels (e.g., "plane", "car", "dog") are converted into prompt templates like “a photo of a {label}”.These prompts are encoded via the same Text Encoder used during training to produce embeddings T₁ to Tₙ representing class concepts.
* *Zero-Shot Prediction* :An unseen test image is passed through the Image Encoder to generate its embedding I₁.The similarity of this image embedding is compared against all class label embeddings.The label with the highest similarity score is selected as the predicted class (e.g., “a photo of a dog”)—this enables zero-shot learning without task-specific training.
* This powerful architecture enables generalizable visual understanding from natural language descriptions.

## 2.3. Dataset

- Flickr8k: 8,000 images with 5 captions each

- Flickr30k: 31,000 images with 5 captions each

## 2.4. Proposed Methodology

**Phase 1: Retrieval (Image/Text Matching) from pretrained model**

This phase focuses on developing a system that can find relevant images given a text query, or vice-versa, by understanding the semantic relationship between images and text.

**Steps to be followed :**

***Image Embedding Extraction:***

* + *Select a pre-trained CNN model (e.g., ResNet50, EfficientNet, ViT).*
  + *Remove the classification layer (the final fully connected layer(s) used for classification) from the pre-trained model.*
  + *Pass the selected query image through the modified CNN model.*
  + *Extract the feature vector (embedding) from the layer immediately preceding the removed classification layer. This vector represents the image's features.*
  + *Generate image embeddings for all images in the Flickr 8K dataset using the same pre-trained CNN model and extraction method.For Text Embedding extraction* : Use LSTM, Bi-LSTM, or Transformer-based models like BERT (or a custom Transformer). Convert captions to embeddings.

***Text Encoder Extraction :***

* *Prepare Data: Tokenize captions and convert tokens to indices using a vocabulary. Pad sequences for uniform length.*
* *Load/Train LSTM: Load a pre-trained LSTM or train one from scratch. Include an embedding layer.*
* *Process Captions: Feed tokenized captions through the LSTM.*
* *Extract Embeddings: Obtain the final hidden state of the LSTM for each caption. This is the text embedding.*
* *Dump Embeddings: Save the extracted text embeddings (NumPy arrays, etc.) to a file for later use.*

**Objective1: Retrieve and display the nearest neighbor image from the Flickr 8K dataset based on image embeddings extracted using pre-trained CNNs.(Image-Image)**

1. *Image Selection:* Choose an image from the Flickr 8K dataset. This will serve as the query image.
2. *Nearest Neighbor Search:* Using the extracted embedding vector of the query image, search the embedding vectors of all other images in the Flickr 8K dataset.

Employ a distance metric (e.g., cosine similarity, Euclidean distance) to determine the "nearest" image embedding vector to the query vector.

**Objective2: Retrieve the nearest label text from word embeddings for a selected image from the Flicker 8k dataset.(Image-Text)**

1. *Select Image*: Choose an image from the Flickr 8K dataset.
2. *Calculate Similarity:* Compute the similarity (e.g., cosine similarity) between the image embedding and each label text embedding*.*
3. *Retrieve Nearest Label:* Identify the label text with the highest similarity score to the image embedding.
4. *Output:* Display the selected image and its nearest label text*.*

**Objective3: Retrieve and display the nearest neighbor image from the Flickr 8K dataset based for a selected label text from Text Encoder(Text-Image)**

1. *Select Label Text: Choose a label text.*
2. *Calculate Similarity: Compute the similarity (e.g., cosine similarity) between the text embedding and each image embedding.*
3. *Retrieve Nearest Image: Identify the image with the highest similarity score to the text embedding.*
4. *Display: Show the selected label text and the retrieved nearest neighbor image*

**Objective4: Retrieve the nearest label text from word embeddings for a selected label text from Text Encoder(Text-Text)**

1. *Select Label Text: Choose a query label text.*
2. *Calculate Similarity: Compute the similarity (e.g., cosine similarity) between the query text embedding and all other label text embeddings.*
3. *Retrieve Nearest Label: Identify the label text with the highest similarity score.*
4. *Display: Show the selected query label text and the retrieved nearest neighbor label text.*

*A diagram of a dog

AI-generated content may be incorrect.*

*.*

**Phase 2: Image Captioning**

## Literature Review

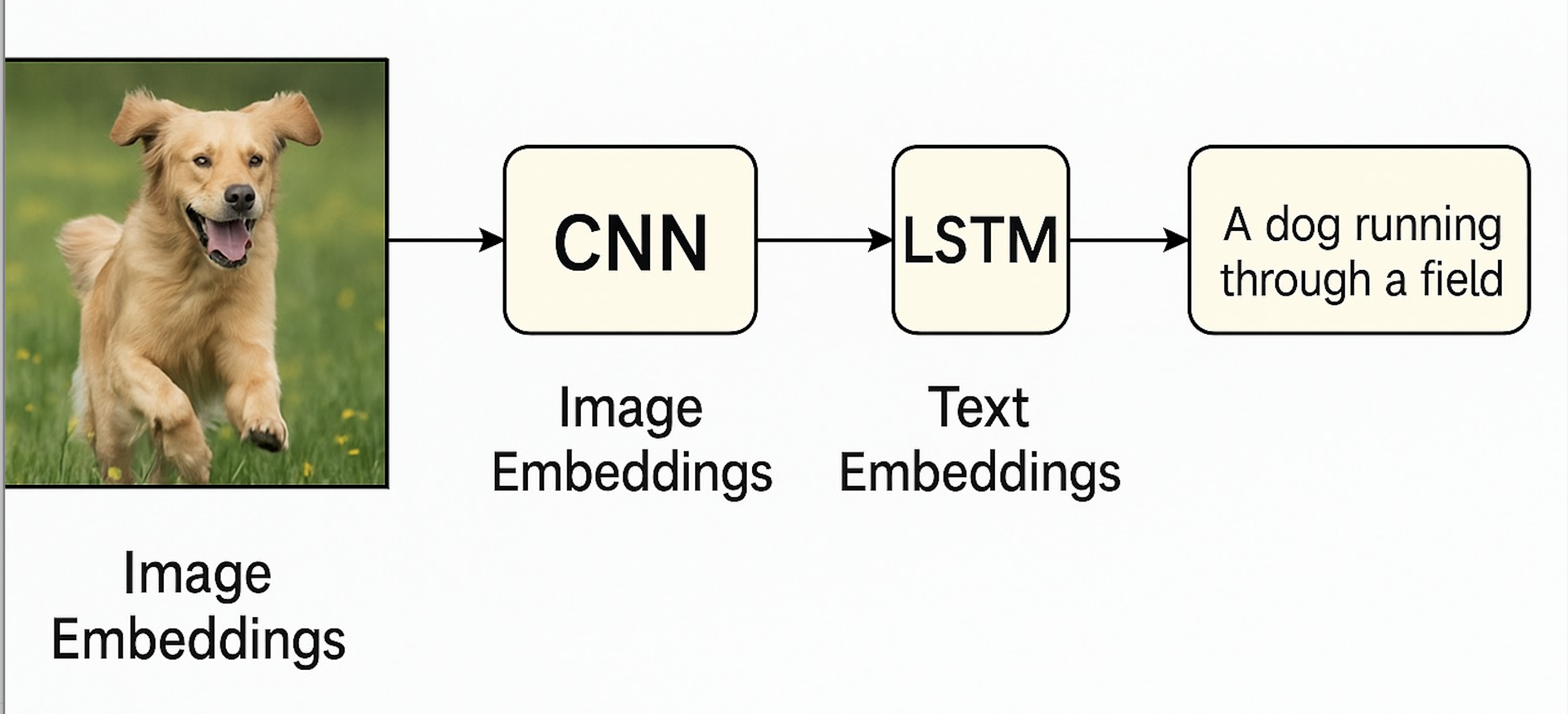
Image captioning combines CNNs and RNNs in an encoder-decoder setup. Recent transformer-based models like ViLT and BLIP have improved contextual captioning.

## Dataset

Same datasets as Task 1: Flickr8k and Flickr30k

## Proposed Methodology

**Step 1: Encoder Decoder Model with CNN and LSTM**



**Step 2: Encoder Decoder Model with VGG-16/RESNET-50 CNN and LSTM**

**A diagram of a dog

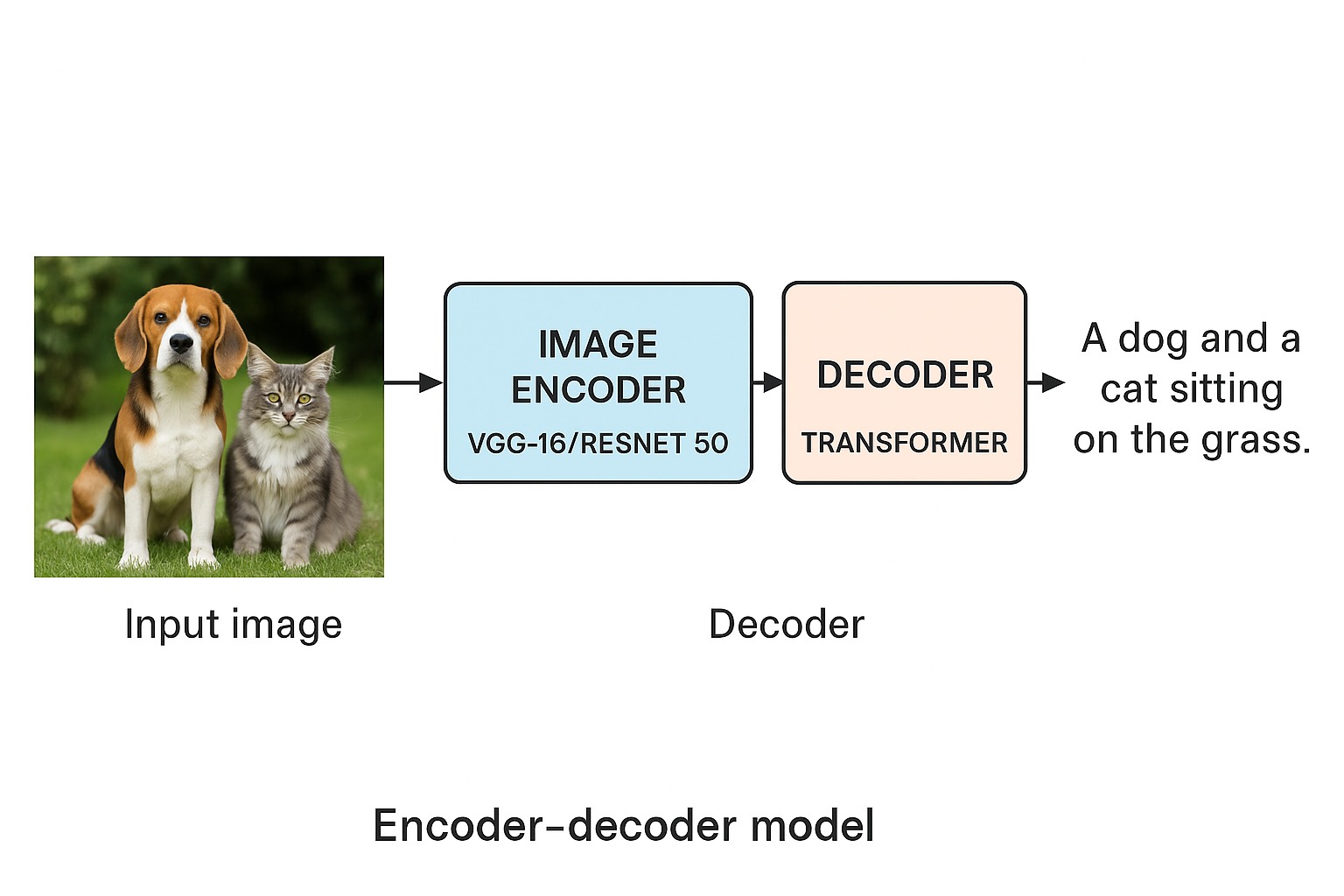
AI-generated content may be incorrect.**

**Step 3: Encoder Decoder Model with VGG-16/RESNET-50 CNN and LSTM+Attention Network**

**A diagram of a dog catching a frisbee

AI-generated content may be incorrect.**

**Step 4: Encoder Decoder Model with VGG-16/RESNET-50 CNN and Transformer Network**

****

**Step 5: Encoder Decoder Model with VITEncoder and Vision Transformer Network**

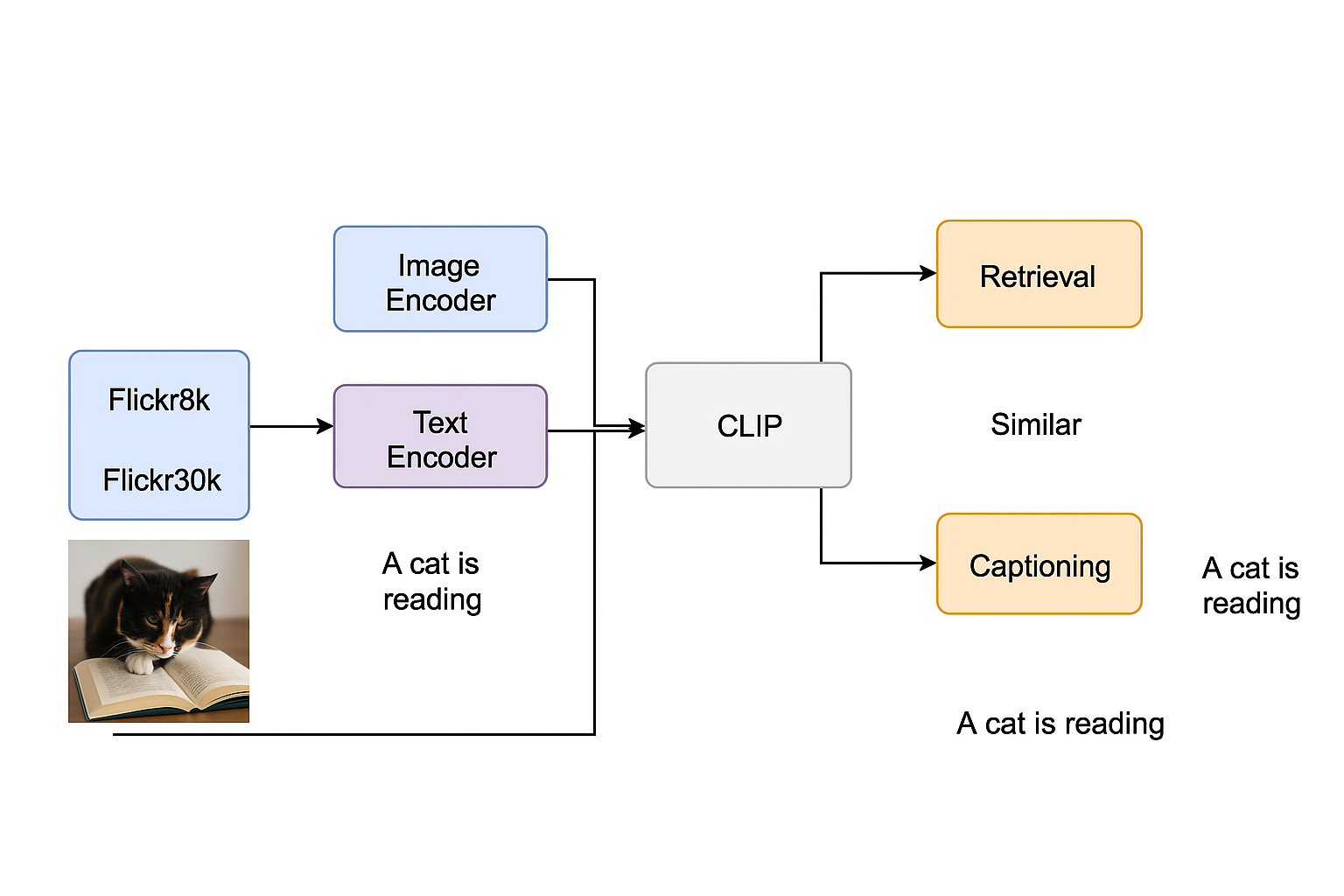
**A diagram of a green field

AI-generated content may be incorrect.**

### Evaluation Metrics

- BLEU (Bilingual Evaluation Understudy): Evaluates the correspondence between machine-generated and reference captions using n-gram overlap. BLEU-1 considers unigrams, while BLEU-4 extends to 4-grams.  
- ROUGE (Recall-Oriented Understudy for Gisting Evaluation): Measures the overlap of words and sequences between generated and reference texts. Useful for assessing informativeness and content matching.  
- WER (Word Error Rate): Calculates the rate of errors (insertions, deletions, substitutions) between generated captions and ground truth. Lower WER indicates higher accuracy.

# 4. System Design Diagram



# 5. Real-World Applications

- Image Search Engines  
- Visual Accessibility Tools  
- Automated Social Media Tagging  
- Surveillance Captioning for Alerts

# 6. Challenges

- High computational resource needs  
- Overfitting on small datasets  
- Aligning semantic spaces across modalities

# 7. Tools & Technologies

- Python, PyTorch, Tensorflow  
- HuggingFace Transformers  
- FastAPI / Streamlit for deployment  
- draw.io for architecture diagrams

# 8. Project Timeline

|  |  |
| --- | --- |
| Week | Task |
| 1 | Literature Review & Dataset Collection |
| 2-3 | Preprocessing & Baseline Model Setup |
| 4-5 | Training Retrieval Model |
| 6-7 | Training Captioning Model |
| 8 | Evaluation, Deployment, and Report Finalization |

# 9. References

- Radford et al., "Learning Transferable Visual Models From Natural Language Supervision" (CLIP)  
- Flickr8k and Flickr30k dataset documentation  
- BLEU, ROUGE, WER evaluation metrics