Video Frame and Timestamp retrieval based on Text Query: Harnessing Multimodal Encoders

A Thesis Submitted
partial fulfillment of the requirements for the award of the degree of
Bachelor of Technology
In
Information Technology

Submitted by
Niharika Mangarai IIB2020011
Priyamvada Priyadarshani IIB2020037
Kalyani Pharkandekar IIT2020196

Under the
Supervision of
Prof. Vijay Kumar Chaurasiya

Information Technology
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY ALLAHABAD
Prayagraj -211015
May, 2023

CANDIDATE DECLARATION

I hereby declare that the work presented in this report entitled "Video Frame and Timestamp retrieval based on Text Query: Harnessing Multimodal Encoders", submitted towards the fulfillment of BACHELOR'S THESIS report of the 8th Semester B.Tech curriculum at the Indian Institute of Information Technology Allahabad, is an authenticated record of our original work carried out under the guidance of Prof./Dr. Vijay Kumar Chaurasiya. Due acknowledgments have been made in the text to all other material used. The project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Niharika Mangarai
IIB2020011
Department of Information Technology-Business Informatics

Priyamvada Priyadarshani
IIB2020037
Department of Information Technology-Business Informatics

Kalyani Pharkandekar
IIT2020196
Department of Information Technology

CERTIFICATE FROM SUPERVISOR

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief. The project titled "Video Frame and Timestamp retrieval based on Text Query: Harnessing Multimodal Encoders" is a record of the candidates' work carried out by her under my guidance and supervision. I do hereby recommend that it should be accepted in the fulfillment of the requirements of the Bachelor's thesis at IIIT Allahabad.

Prof. Vijay Kumar Chaurasiya
Department of Information Technology

CERTIFICATE OF APPROVAL

The forgoing thesis is hereby approved as a creditable study carried out in the area of Information Technology and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approves the thesis only for the purpose for which it is submitted.

Committee on the final examination for the evaluation of the project report:

1. Prof. Vijay Kumar Chaurasiya
   Department of Information Technology

   Signature:

1. Supervisor:
   Prof. Vijay Kumar Chaurasiya, Department of Information Technology

   Signature:

2. External Subject Expert: NA

Dean(A & R)

Plagiarism Report

CONTENTS

Abstract

With the ever-burgeoning internet era and exponential growth of video content, it has become a crucial job to consume and search within for content navigation or frame retrieval. While semantic annotation, basic keyword search, and single-modal encoders fail to capture much insight, the emergence of multimodal learning offers a fusion of different techniques efficiently. For better user-friendliness like exact video frame retrieval, and better accuracy observed in image-text models than video-text models, we propose our multimodal architecture for specific video frame retrieval using text query. Our model incorporates calculating the loss of image embeddings, obtained from the image encoder, and text embeddings obtained from the text encoder, projected into the shared embedding space for cross-modal interactions and training it for the flickr8k and flickr30k datasets. We use Resnet50 and DistilBERT as our image encoder and text encoder respectively. In addition, we used a loss function based on contrastive learning to maximize the alignment of modalities inside a common embedding space. Model Performance testing on Flickr datasets and benchmark dataset MSRVTT was done to calculate recall and other performance metrics. Lastly, a streamlit application that accepts YouTube video links, and text queries allows for testing demos in a real-world scenario.

Keywords: Multimodal fusion, video retrieval, frame localisation, timestamp extraction, ResNet, ViT, BERT, Flickr8k, Flickr30k, MSRVTT.

CHAPTER 1

Introduction

The world we live in has become so advanced in technology that it has changed the way we see, hear, and analyze information. The internet of today is an enormous collection of different multimedia assets. And as we know, with a large amount of data be it texts, images, or videos comes the challenge of finding relevant information efficiently. Conventional search engines usually rely on keyword-based search. These search engines tend to be effective for text-based queries, however, have difficulties in retrieving multimedia content efficiently, especially when the query is strongly dependent on visual or audio features.

Video is one of the most popular forms of media for it can capture events, and knowledge and is the preferred form of content consumption today, for its natural appeal to our visual and auditory senses. Online video platforms like YouTube etc are encouragingly promoting this form of media. But with a wide range of videos available people often have a hard time processing and look for ways to find at what timestamps particular things one can find in the video. Many use cases and application exist for the field. Video editors can quickly locate specific frames or moments within large video datasets, streamlining the editing workflow and enhancing productivity. Students can benefit from targeted video frame retrieval to supplement their studies, access relevant visual examples, or review key concepts. Content-based retrieval systems, which are created to connect the semantic content of a query with the multimedia content in the database are the solutions we found for this limitation.

Content- based retrieval systems employ algorithms to analyze the basic features of multimedia files, like images, video clips, and audio clips to evaluate their relevance. Content- based retrieval systems examine the visual or auditory characteristics of multimedia content, unlike keyword-based systems, which rely only on textual information. These systems integrate methodologies from

computer vision, and natural language processing, as well as machine learning and artificial intelligence to extract and analyze features from multimedia content.
They are a necessity with the growing volume and complexity of online multimedia data. Traditional search engines are failing to provide users with the relevant data that they need due to the millions of photos, videos, and audio samples that are getting uploaded to the internet every day. There is a need for an intelligent system that can not only understand textual queries but also retrieve relevant video frames based on text queries, enabling users to efficiently navigate large video datasets and extract desired information.

Content-based retrieval systems have developed dramatically with recent advancements in deep learning. High performance has been observed in automatically extracting high-level semantic features from unprocessed multimedia content using deep learning approaches, such as transformer-based models, Convolutional Neural Networks(CNNs). Being inspired by the structure and function of the human visual cortex, CNNs are especially good at extracting relevant features from multimedia content, especially images. These networks are capable of automatically picking up hierarchical representations of visual features, from basic low-level features such as textures, edges to more complex higher-level concepts like objects and scenes.

A popular approach to the retrieval problem is similarity learning, where we learn the functionality of the two elements (query and candidate) that are most relevant. Describe their similarities. Then all candidates can be ranked accordingly of their similarity to the query. To achieve this rank, captions as well as images/videos is to be presented as a common multi-dimensional integration space, where similarities can be calculated as the dot product of their respective values respective representations. The important question here is how to learn correctly the representation of captions and videos/images serves as the basis for our similarity estimation.

Starting from RNN, LSTM, and attention, Nature language processing has come a long way to the age of transformers and large language models. Transformer-based models are particularly good at handling text and other

sequential data. By fine-tuning particular downstream tasks after pretraining on massive text corpora, models like BERT(Bidirectional Encoder Representations from Transformers) have obtained state-of-the-art outcomes in tasks of natural language comprehension. The performance of content-based retrieval systems can be improved by extracting rich semantic information from textual and visual inputs by utilizing deep learning.

One of the key challenges in video moment retrieval is the limited recall of existing models and does not fully exploit such cross-modal high-level semantics. By integrating text and image processing techniques, our approach aims to overcome this limitation and improve recall rates. This is particularly beneficial for users who wish to locate specific frames within a video that they remember but cannot easily find using traditional methods.

While traditional video moment retrieval systems provide start-end timestamps for predefined segments, our project takes a novel approach by leveraging an image-text model. Unlike video models, which often suffer from low recall rates (typically ranging from 20-60%) and takes more time for video processing, our system harnesses the power of multimodal encoder architectures for image-text having good accuracy without audio/temporal corporation to enhance retrieval accuracy and recall, to provide efficient user experience

This project aims to create a reliable system for content-based retrieval that can precisely retrieve video frames using textual queries. Our goal is to use deep learning techniques to create a semantic space and map both textual queries and video frames in the same space where the similarity between text and images can be measured directly.

We use Resnet50 and DistilBERT as our image encoder and text encoder respectively. In addition, we used a loss function based on contrastive learning to maximize the alignment of modalities inside a common embedding space. Model Performance testing on Flickr datasets and benchmark dataset MSRVTT was done to calculate recall and other performance metrics. Lastly, a streamlit application that accepts YouTube video links, and text queries allows for testing demos in a real-world scenario. We have incorporated contrastive cross-entropy

loss calculation thus obtaining positive-negative pair similarities. Contrastive cross entropy loss enhances model performance by encouraging similar embeddings for similar inputs and dissimilar embeddings for distinct inputs, facilitating better representation learning in tasks like similarity comparison or clustering. Its utility lies in its ability to effectively guide the model towards learning meaningful embeddings by penalizing the distance between similar pairs while pushing apart dissimilar pairs in the embedding space.

Possible applications of our proposed system include analyzing human behaviour in surveillance footage, studying animal movements in wildlife videos, or examining environmental changes in satellite imagery, our video frame retrieval system provides a powerful tool for data exploration and analysis. Investigators can quickly search through vast amounts of surveillance footage to identify suspects, track their movements, or reconstruct events leading up to a crime. You can easily find at what timestamp the cute dog you wanted to show your friend appears, or the adventure scene you want to practice but don’t remember which video.

1.1 Related Works

Video Frame and Timestamp retrieval based on text query is to pinpoint the exact timestamp along with the frame with the object or scene mentioned in the text query provided by the user. It plays a crucial role in the video understanding field [1,4,6,7,10]. Most of the work done in this domain is under video moment retrieval, which is to localize the correct moment in an untrimmed video that is semantically aligned with a given natural language query [2,5,7,8]. Conversely, many approaches [3,6,9,11] learn directly from images or pre-detected objects, with varying amounts of inductive bias tailored to the QA task. Our method can be seen as an in-between: while not explicitly using the program supervision, it is trained to detect objects that are required for in the user text query.

Recent progress in multi-modal understanding has been mainly powered by pre-training large transformer models to learn generic multi-modal representations from enormous amounts of aligned image-text data [4,8,10,12], then finetuning them on downstream tasks. These methods can be divided into single stream [2, 6, 9,10] and two-stream [1,5,6,11] architectures depending on whether the text and images are processed by a single combined transformer or two separate transformers followed by some cross attention layers. For both these types, the prevalent approach is to extract visual and textual features independently and then use the attention mechanism of the transformers to learn an alignment between the two.While this approach has improved state of the art results on a wide variety of tasks such as fast-video retrieval [1], object detection [3], localizing moments in the video [13] and Image and Sentence Matching [6],it leaves a lot of room for errors, as the accuracy is very low.

While traditional video moment retrieval systems provide start-end timestamps for predefined segments, our project takes a novel approach by leveraging an image-text model. Unlike video models, which often suffer from low recall rates (typically ranging from 26% to 50%), our system harnesses the power of multimodal encoder-decoder architectures to enhance retrieval accuracy and recall.

One of the key challenges in video moment retrieval is the limited recall of existing models. By integrating text and image processing techniques, our approach aims to overcome this limitation and improve recall rates. This is particularly beneficial for users who wish to locate specific frames within a video that they remember but cannot easily find using traditional methods.
Chen and Zitnick [6] use a multimodal auto-encoder for bidirectional mapping, and measure the similarity using the cross-modal likelihood and reconstruction error. Mao et al. [8] propose a multimodal RNN model to generate sentences from images, in which the perplexity of generating a sentence is used as the similarity. Donahue et al. [3] design a long-term recurrent convolutional network for image captioning, which can be extended to image and sentence matching as well. Vinyals et al. [13] develop a neural image captioning generator and show the effectiveness of the image and sentence matching. These models are originally designed to predict grammatically-complete sentences, so their performance on measuring the image-sentence similarity is not very well. Different from them, our work focuses on the similarity measurement after storing embeddings of all images and text in the same vector space, which is especially suitable for the task of image and query matching.

Models like GPT (Generative Pre-trained Transformer) and ELMo (Embeddings from Language Models) have demonstrated impressive performance by pre-training on large text corpora and then fine-tuning on specific downstream tasks[17]. However, these models are unidirectional, processing text input in either left-to-right or right-to-left direction, which limits their ability to capture bidirectional context. BERT on the other hand learns bidirectional representation of words by masking random tokens and training the model to predict them based on surrounding context. The model we used however, DistillBERT, is a compressed version of BERT, designed to be smaller, faster, cheaper, and lighter[16]. By distilling knowledge from BERT, DistilBERT can significantly reduce the number of parameters and computational resources required while maintaining a competitive level of performance.
Because of its emphasis on video captioning, we decided to test our models on the MSRVTT dataset. Ten thousand video clips with numerous human-generated subtitles are included in this collection. In contrast to the Flickr datasets, MSRVTT offers video content, posing special opportunities and problems for

captioning models. We can test our models' performance in producing captions for films by running them on MSRVTT. This allows us to examine how well our models grasp temporal linkages and visual context.

In contrast to other datasets like Visual Genome and COCO (Common Objects in Context), these datasets have a number of benefits. Primarily, Flickr8k and Flickr30k offer superior captions that are more consistent with natural language and offer more context for the photos. Furthermore, the research community has made extensive use of both datasets, which has improved model benchmarking and comparison. In contrast, MSRVTT offers an extensive library of video clips with a wide range of material, which makes it appropriate for testing and training video captioning algorithms.

1.2 Methodology

1.2.1 Dataset

In our project, we utilized three main datasets: Flickr8k, Flickr30k, and MSRVTT. These datasets have been widely used in the field of computer vision and natural language processing for tasks such as image captioning and video captioning. While there are other datasets available for similar tasks, these three datasets offer several advantages that make them preferable for our project.

Flickr8k: The Flickr8k dataset consists of 8,000 images collected from the Flickr website, each paired with five different captions. This dataset is relatively small compared to others, but it is well-suited for training and evaluating image captioning models due to its diverse range of images and high-quality captions. The captions are concise and descriptive, providing valuable context for the images.

Flickr30k: Similar to Flickr8k, the Flickr30k dataset contains 30,000 images from Flickr, each with five captions. This larger dataset offers a more extensive collection of images, covering a broader range of scenes, objects, and activities. The captions in Flickr30k are also more varied and detailed, capturing a wide range of linguistic expressions and styles.

MSRVTT (Microsoft Research Video to Text): Unlike the Flickr datasets, MSRVTT focuses on video captioning rather than image captioning. It consists of 10,000 short video clips, each annotated with multiple human-generated captions. This dataset is valuable for training models to generate captions for videos, making it an essential resource for tasks such as video understanding and retrieval.

Compared to other datasets, such as COCO (Common Objects in Context) and Visual Genome, these datasets offer several advantages. Firstly, Flickr8k and Flickr30k provide high-quality captions that are more aligned with human language and provide better context for the images. Additionally, both datasets have been widely used in the research community, enabling better comparison and benchmarking of models. MSRVTT, on the other hand, provides a

comprehensive collection of video clips with diverse content, making it suitable for training and evaluating video captioning models.

Overall, the combination of Flickr8k, Flickr30k, and MSRVTT datasets offers a diverse and comprehensive set of images and videos, along with high-quality captions, making them ideal choices for our project on image and video captioning.

As Flickr8k and Flickr30k have 5 captions for each image, we sorted these captions based on probabilities of these captions in descending order as some captions might be more accurate than others in the datasets. We used `OwlViTModel`, a pre-trained model from Hugging Face to find the probabilities of these captions and sorted them using a simple for loop.

1.2.2 Multimodal Fusion

The significance of multimodal fusion lies in employing the unique strengths of each individual modality while contemporaneously addressing their essential limitations. In a world where data comes in colorful forms and formats, incorporating information from different modalities enables us to overcome the limitations of each individual data source. This comprehensive approach allows us to capture details, patterns, and connections that might remain retired when considering data sources in insulation. The coupling of modalities is analogous to combining mystification pieces, where the whole picture emerges only when all the rudiments are combined.

- **Early fusion**- In this approach, raw data from different modalities is combined at the input position before being fed into a model. For illustration, combining textbook and image data into a single input vector.
- **Late fusion** -In this approach, data from each modality is reused singly through separate models, and the labors from these models are also combined at a after stage.
- **Intermediate fusion** -This approach combines data from different modalities at colorful intermediate processing stages within a model armature.

- **Hybrid fusion**- Hybrid approaches combine different emulsion strategies to achieve the asked results. Multimodal is a subset of deep literacy models that deals with data from multiple modalities similar as textbook, images, audio, illustrations or any other types of data.

The core idea behind these models are that they learn the features learned from different types of data and combine them to perform better at more complex machine literacy tasks. The ideal of these types of models is to produce participated representation of features uprooted from different modalities.

1.2.3 Image encoding

The 50-layer ResNet architecture includes the following elements, as shown in the table below:

- A 7×7 kernel convolution alongside 64 other kernels with a 2-sized stride.
- A max pooling layer with a 2-sized stride.
- 9 more layers—3×3,64 kernel convolution, another with 1×1,64 kernels, and a third with 1×1,256 kernels. These 3 layers are repeated 3 times.
- 12 more layers with 1×1,128 kernels, 3×3,128 kernels, and 1×1,512 kernels, iterated 4 times.
- 18 more layers with 1×1,256 cores, and 2 cores 3×3,256 and 1×1,1024, iterated 6 times.
- 9 more layers with 1×1,512 cores, 3×3,512 cores, and 1×1,2048 cores iterated 3 times.
- (up to this point the network has 50 layers)
- Average pooling, followed by a fully connected layer with 1000 nodes, using the softmax activation function.

1.2.4 Softmax activation function

Softmax with temperature, often used in conjunction with contrastive loss, is another technique to enhance the learning process in multimodal models.
It involves applying softmax activation to the pairwise similarity scores between embeddings, scaled by a temperature parameter.

By adjusting the temperature parameter, the model can control the smoothness of the probability distribution over the pairwise similarities, influencing the training dynamics.
Softmax with temperature is often applied during the calculation of similarity scores and used as part of the contrastive loss function to compute the final loss

$$\mathrm{softmax}(x)_i = \frac{e^{\frac{y_i}{T}}}{\sum_j^N e^{\frac{y_j}{T}}}$$

Figure 1.1 Softmax function

1.2.5 Text encoding

The thing of distillation is to produce a lower model which can imitate a larger model. In practice, it means that if a large model predicts a commodity, also a lower model is anticipated to make a similar prediction. To achieve this, a larger model needs to be formerly pretrained(BERT in our case). also an architecture of a lower model needs to be chosen. To increase the possibility of successful reproduction, it's generally recommended for the lower model to have a similar architecture to the larger model with a reduced number of parameters. Eventually, the lower model learns from the predictions made by the larger model on a certain dataset. For this ideal, it's vital to choose an applicable loss function that will help the lower model to learn better. In distillation notation, the larger model is called a teacher and the lower model is referred to as a student. Generally, the distillation procedure is applied during the pretaining but can be applied during the fine- tuning as well.

DistilBERT
DistilBERT learns from BERT and updates its weights by using the loss function which consists of three components:

- Masked language modeling (MLM) loss
- Distillation loss
- Similarity loss

Softmax function Softmax normalizes all model outputs, so they sum up to 1 and can be interpreted as probabilities.
The temperature T controls the smoothness of the output distribution:
If T > 1, then the distribution becomes smoother.
If T = 1, then the distribution is the same if the normal softmax was applied.
If T < 1, then the distribution becomes more rough.

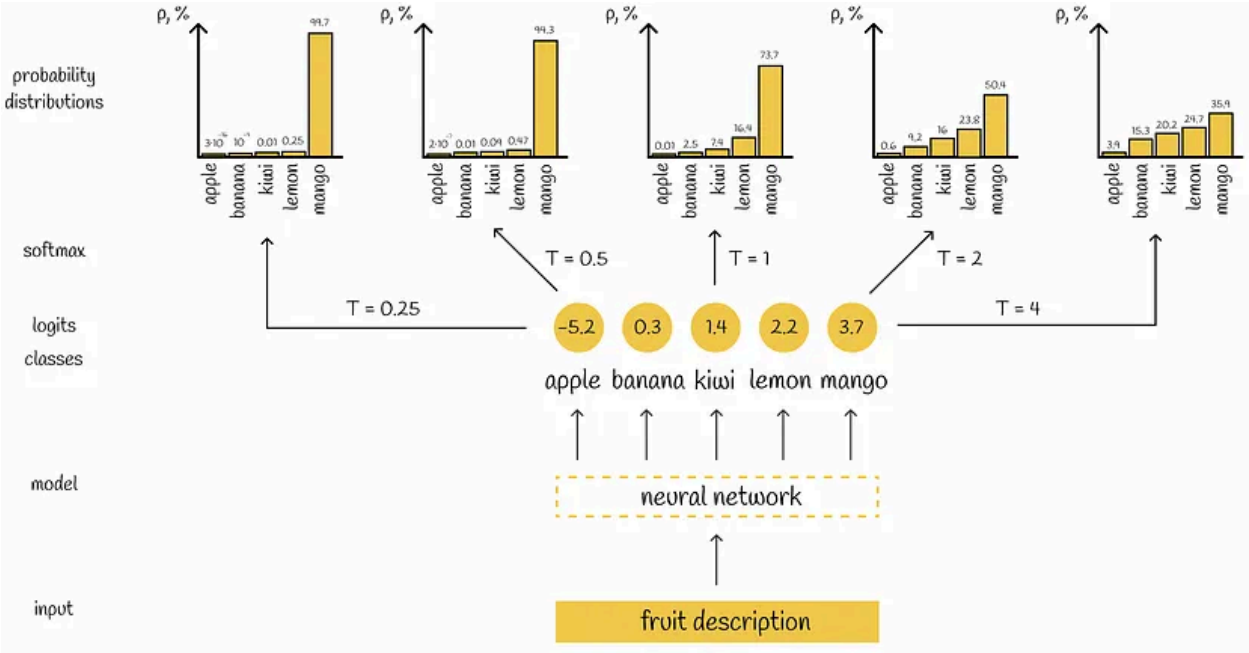


Figure 1.2 Probability distribution at different temperature values in DistilBERT

Masked language modeling loss

Masked language modeling loss is similar to the teacher's model(BERT), during pre-training, the Student(DistilBERT) learns language by making predictions for the masked language modeling task. After producing a prediction for a certain token, the predicted probability distribution is compared to the one-hot decoded probability distribution of the schoolteacher's model. The one-hot decoded distribution designates a probability distribution where the probability of the most likely token is set to 1 and the chances of all other commemoratives are set to 0. As in utmost language models, the cross-entropy loss is calculated between predicted and true distribution and the weights of the pupil's model are

streamlined through backpropagation

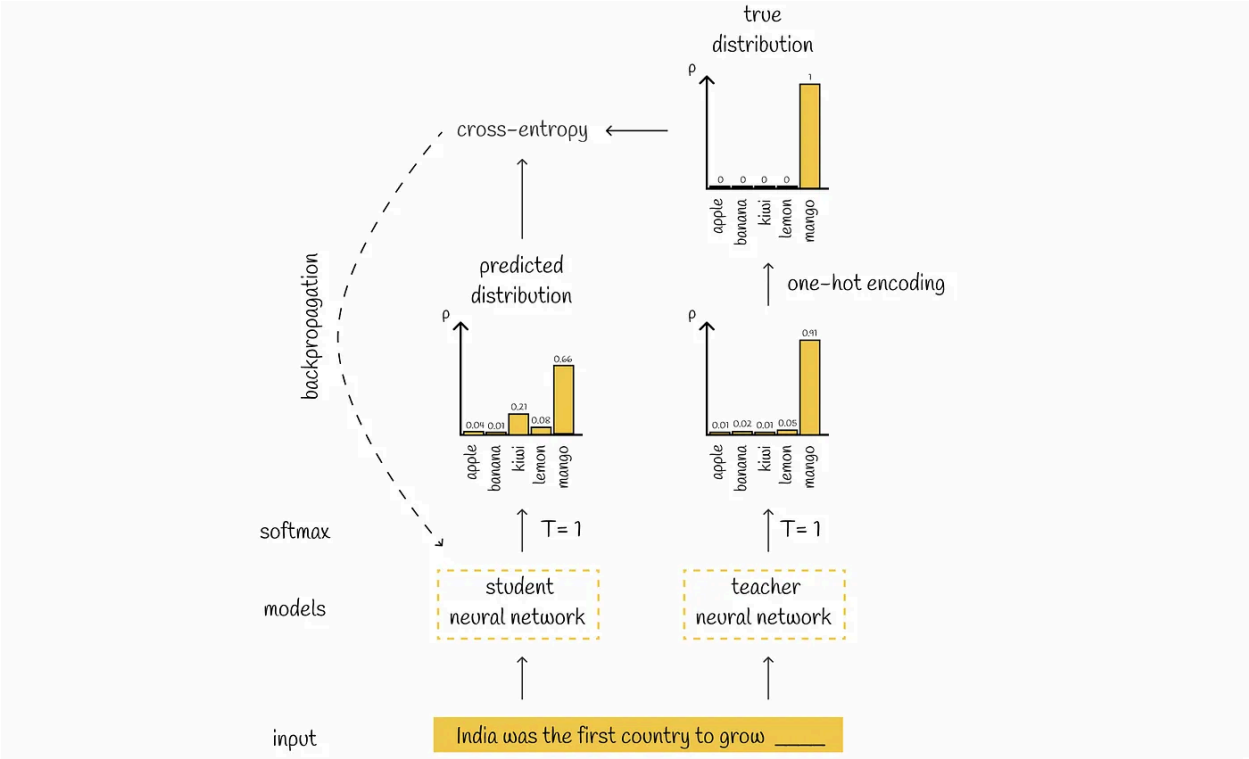


Figure 1.3 Masked language modeling loss

Distillation loss

Actually it's possible to use only the student loss to train the student model. Still, in numerous cases, it might not be enough. The common problem with using only the pupil loss lies in its softmax transformation in which the temperature T is set to 1. In practice, the performing distribution with T = 1 turns out to be in the form where one of the possible labels has a very high probability close to 1 and all other label probabilities come low being close to 0. Such a situation doesn't align well with cases where two or further classification labels are valid for a particular input; the softmax subcaste with T = 1 will be very likely to count all valid markers but one and will make the probability distribution close to one-hot encoding distribution. This results in a loss of potentially useful information that could be learned by the student model which makes it less different. That's why the authors of the paper introduce distillation loss in which softmax chances are calculated with a temperature T> 1 making it possible to easily align chances, therefore taking into consideration several possible answers for thestudent.In

distillation loss, the same temperature T is applied both to the pupil and the teacher. One-hot encoding of the teacher's distribution is removed. In distillation loss, the same temperature T is applied both to the pupil and the teacher. One-hot encoding of the teacher's distribution is removed

Similarity loss

The researchers also state that it is beneficial to add cosine similarity loss between hidden state embeddings.

$$L_{cos}(x, y) = 1 - \cos(x, y) = 1 - \frac{x \cdot y}{\|x\| \cdot \|y\|}$$

Cosine loss formula

This way, the student is likely not only to reproduce masked tokens correctly but also to construct embeddings that are similar to those of the teacher. It also opens the door for preserving the same relations between embeddings in both spaces of the models.

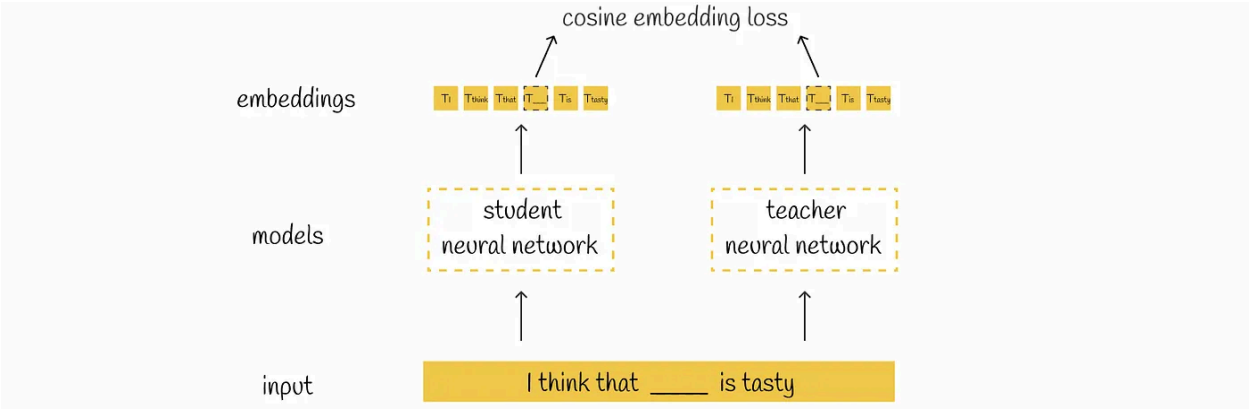


Figure 1.4 Cosine loss embedding loss calculation

Similarity loss computation example

Triple loss

Finally, a sum of the linear combination of all three loss functions is calculated which defines the loss function in DistilBERT. Based on the loss value, the backpropagation is performed on the student model to update its weights.

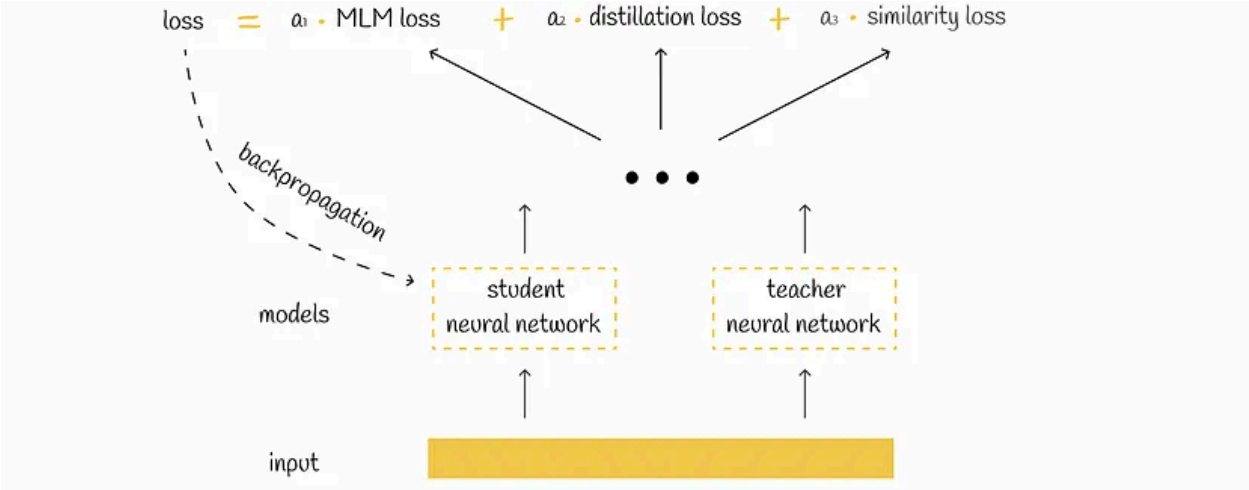


Figure 1.5: DistilBERT Architecture

1.2.6 CONTRASTIVE LOSS

Contrastive Objective- " Given a batch of N(image, text) pairs, is trained to predict which of the N × N possible(image, text) pairings across a batch actually passed. " It learns amulti-modal embedding space by concertedly training its encoders to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch while minimizing the cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings, By mapping both images and texts into the same fine space, we can perform two critical tasks maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch minimize the cosine similarity of the embeddings of the N(N — 1) incorrect pairings That participated understanding(between text and image) enables it to generalize and perform zero- shot literacy, handed it can understand the textbooks and match them with corresponding images(and vice versa).

The main idea behind contrastive pre-training is to educate the model to separate between " positive " and " negative " pairs, The main idea behind contrastive pre-training is to educate the model to separate between " positive " and "

negative ” pairs. In the environment of Positive Pairs These are pairs of images and text that are truly related or semantically meaningful. Those pairs will have high cosine similarity. For illustration, an image of a cat should have a positive text description like “ a cute cat. ” Negative Pairs These are pairs where the text and image don't match ” .

1.2.7 Cosine Similarity

The cosine similarity score provides a measure of how well the input image and textbook match in the participated embedding space. Advanced cosine similarity scores indicate stronger semantic alignment, suggesting that the text and image are more closely affiliated or that the image corresponds to the textual description. Lower scores suggest weaker alignment or a mismatch. Note The cosine similarity score between two vectors is reckoned as the cosine of the angle between them, ranging from-1(fully different) to 1(fully analogous), with 0 indicating orthogonality or no similarity.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure 1.6: cosine similarity formula

1.2.8 Projection layer in both encoders

Projection layers, often used in models like CLIP, are responsible for transforming the input embeddings (such as image features or text embeddings) into a shared embedding space where they can be effectively compared and matched. Let's delve deeper into how projection layers work and their significance:

- Purpose: The primary purpose of projection layers is to map the input embeddings from their original high-dimensional space into a lower-dimensional space while preserving the essential semantic information.

By projecting both images and texts into the same shared embedding space, CLIP enables meaningful comparisons between different modalities, facilitating tasks like image-text matching and retrieval.

- Linear Transformation: Projection layers typically consist of linear transformation operations, often implemented as fully connected (or linear) neural network layers.

  These layers apply a linear transformation to the input embeddings, where each element in the output embedding is a weighted sum of the input elements, followed by an optional bias term.

- Dimensionality Reduction: One common objective of projection layers is dimensionality reduction, where the high-dimensional input embeddings are mapped into a lower-dimensional space.

  Reducing the dimensionality of the embeddings can help alleviate computational burden, reduce overfitting, and promote better generalization.

- Shared Embedding Space: In the context of CLIP and similar multimodal models, the projection layers ensure that both image features and text embeddings are projected into the same shared embedding space.

  This shared embedding space enables direct comparisons between images and text, allowing the model to learn meaningful associations and similarities between different modalities.

- Non-linear Activation: To introduce non-linearity and enable the model to capture complex relationships, projection layers often include activation functions such as ReLU (Rectified Linear Unit) or GELU (Gaussian Error Linear Unit).

  These activation functions apply element-wise transformations to the output of the linear transformation, allowing the model to learn non-linear mappings between input and output embeddings.

1.2.9 Architecture diagrams

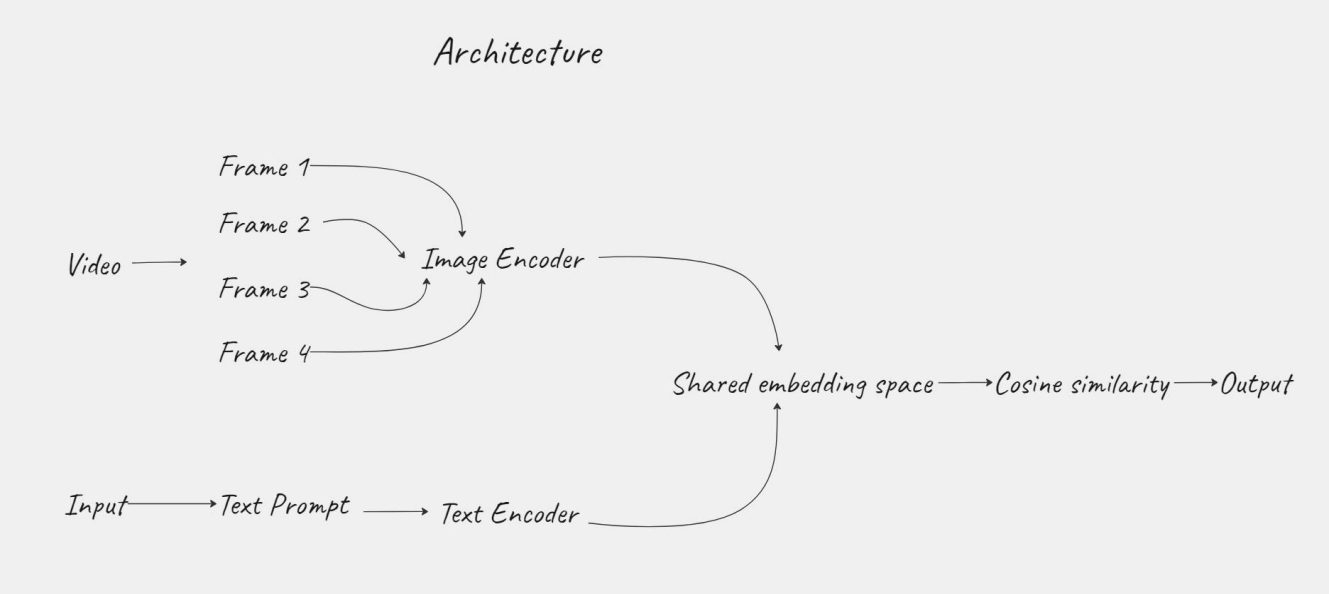


Figure 1.7: Overall Architecture of proposed model

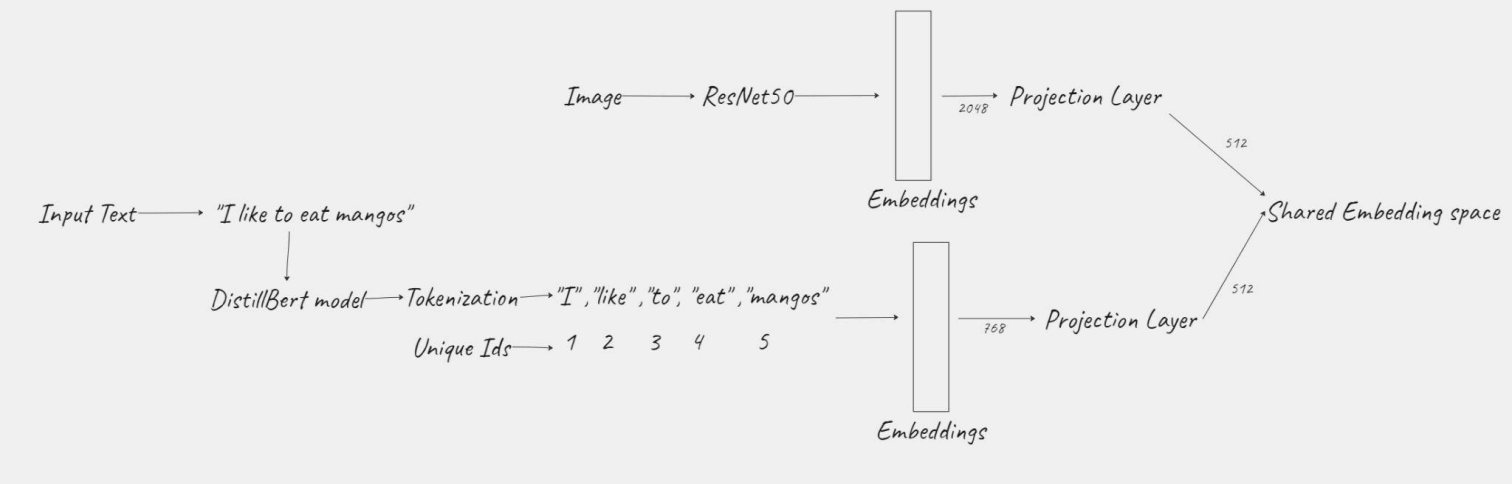


Figure 1.8: DistillBert and ResNet Training Architecture & Projection layer

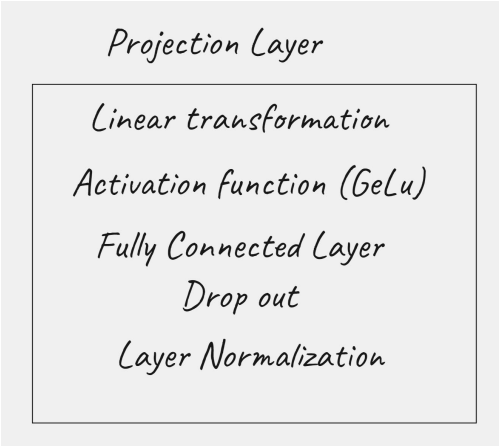


Fig 1.9: Projection Layer

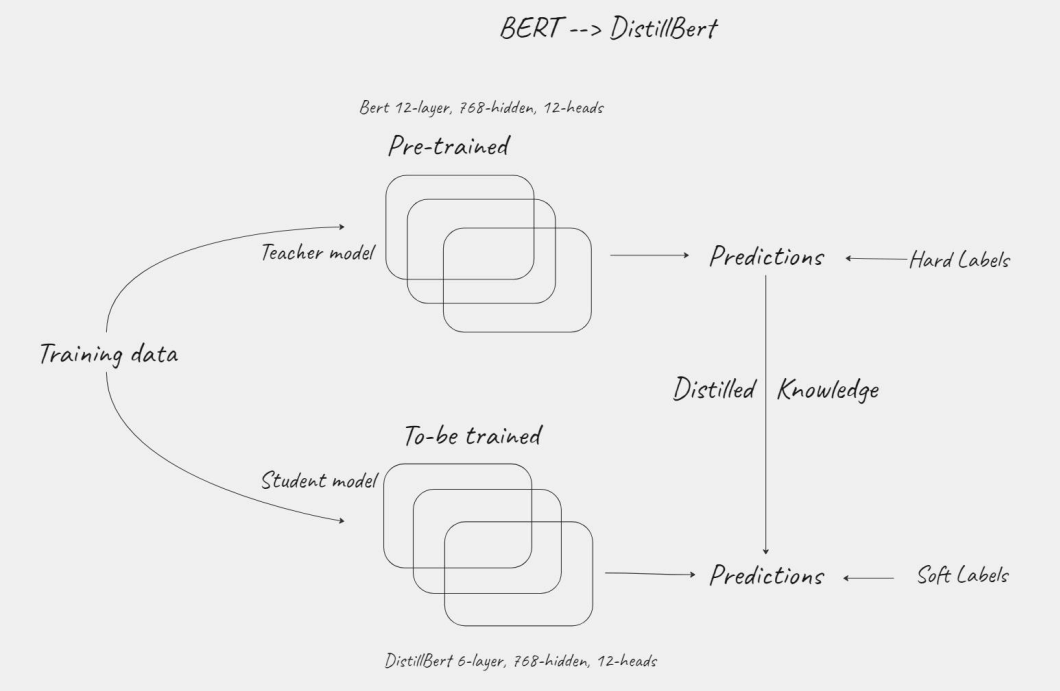


Fig 1.10: Distillation of Bert

CHAPTER 2

2.1 Experiments & Implementation details

2.1.1 Pre-trained ResNet50:

Pre-trained ResNet50 with its robust feature extraction capabilities learned from ImageNet. By removing its last layers, we retain the convolutional base while discarding the classification head, enabling us to obtain image features instead. This process allows us to leverage the pre-trained ResNet50's ability to capture high-level features and spatial information from images.

2.1.2 Pre-trained Distil BERT:

It has 40% less parameters than Bert-base-uncased, runs 60% faster while preserving over 95% of BERT’s performances.Pre-trained Distil BERT on the same corpus as the original BERT model: a concatenation of English Wikipedia and Toronto Book Corpus. It was trained on 8 16GB V100 GPUs for approximately 90 hours. Distil Bert retains 97% of Bert Performance on GLUE benchmark.

2.1.3 Model complexity

Number of parameters. Using multiple modalities does not impact the number of parameters significantly. Interestingly, most of the parameters correspond to the BERT caption encoding module. The number of parameters of a transformer encoder is independent of the number of input embeddings, as are the parameters of a CNN from the image size, like ResNet50.
Our mutli-modal architecture using 2 modalities has: 90.32M parameters, including caption encoder: 67M (Projections:0.675M , DistilBERT:66.36M), image encoder: 24.3 M (Projections:0.788M, ResNet50: 25.5M)

Pre-trained models have already learned meaningful representations and may not require as much regularization during fine-tuning. Applying weight decay to pre-trained model parameters might hinder the fine-tuning process and could

potentially degrade performance. However, just it being 0 is also good as its presence is useful for penalising larger parameters

2.1.4 Training Hyperparameter:

- Shared Embedding Size: The dimensionality of the shared embedding space where image and text embeddings are projected. using a relatively lower-dimensional embedding space like 512 helps to reduce the computational burden during training and inference compared to higher-dimensional spaces, while still providing sufficient representational capacity for many NLP and computer vision tasks.
- Dropout Rate: The probability of dropout regularization applied to the model during training, kept 0.1-0.2 since we are fine-tuning.
- Temperature Value: A hyperparameter is used in the contrastive clip loss function to scale the logits. If we set t=1, the model almost always produces the correct answer with very high confidence, which has very little influence on the cross-entropy cost function during the transfer of knowledge. We set it to 1 and 0.2 and let it train and learn. t is used to scale the logits (unscaled scores) before applying the softmax function in the contrastive loss function. In the context of contrastive loss or similar methods, the temperature value serves to control the smoothness of the probability distribution over positive and negative pairs. A higher temperature value makes the distribution softer, meaning that the model becomes less confident about its predictions, leading to smoother gradients during optimization. On the other hand, a lower temperature value results in a sharper distribution, making the model more confident about its predictions.

- Maximum Sequence Length: The maximum length of input sequences for tokenization.
- Number of Epochs: The total number of training epochs for the model. (25)
- Batch Size: The number of image-text pairs processed in each iteration of the training loop. For fine-tuning pre-trained models or transfer learning tasks, using a smaller batch size can facilitate fine-grained adjustments to

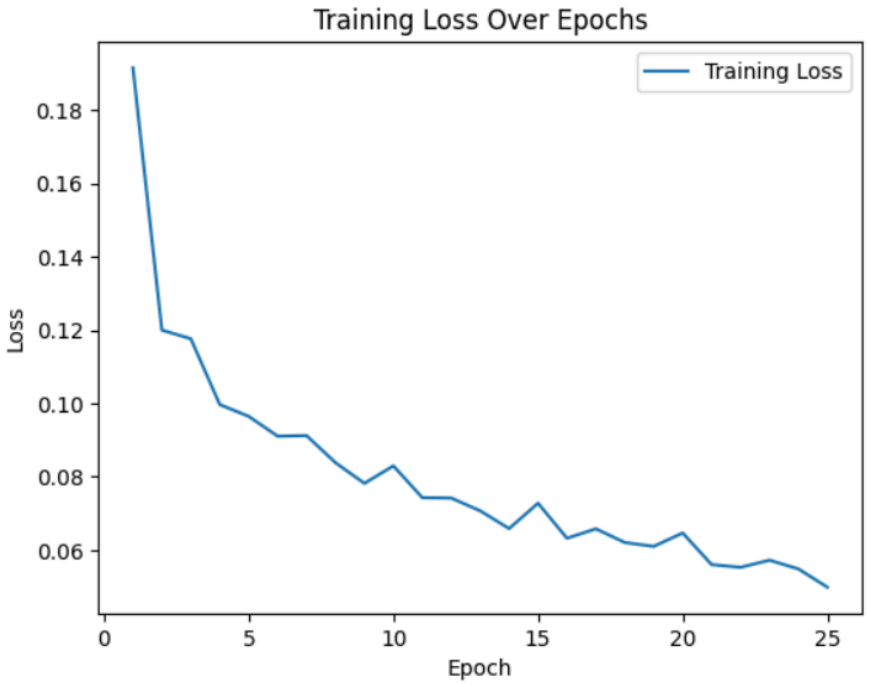
the model parameters without drastically altering the learned representations from the pre-trained model.

The higher learning rate can accelerate the learning process for these parameters, while weight decay helps prevent overfitting.

2.1.5 Training loop:

- Forward pass: Compute the projections for both image and text inputs.
- Compute Loss: Calculate the contrastive clip loss function based on the projections.
- Backpropagation: Update the model parameters by backpropagating the loss gradients.
- Loss Monitoring: Track and monitor the training loss to assess model performance.
- Learning Rate Scheduling: Adjust the learning rate based on the observed loss trends to facilitate convergence.

First trained for flickr30k dataset for 25 epochs and then on flickr8k dataset for 25 epochs which finally results in our final model.



Training loss for flickr8k fine-tuning(over model already fine-tuned for flickr30k)

CHAPTER 3

3.1 Results and Performance

We will show the recall and testing of our model on text-to-image queries where we try to find the most similar image in the image pool based on user query, to show its performance for image-text query level. Next, we prepare our architecture and functions for text-based testing on video inputs.

We are calculating similarity score using cosine similarity for matching image-texts and similarity score matrix looks like:

```
100%|██████████| 4/4 [00:00<00:00, 142.43it/s]
Similarity Scores Matrix:
        Image 0 Image 1 Image 2 Image 3 Image 4 Image 5 Image 6 Image 7 Image 8 Image 9
dog     [array(0.49768162, dtype=float32), array(0.551692, dtype=float32), array(-0.0614527, dty
cat     [array(0.43913266, dtype=float32), array(0.534873, dtype=float32), array(-0.13369888, dt
bird    [array(0.31750268, dtype=float32), array(0.349255, dtype=float32), array(0.13410538, dty
man on swing    [array(-0.48241502, dtype=float32), array(-0.17214695, dtype=float32), array(-0.
```

```
                        Image 1 Image 2 Image 3 Im
Text Query: dog
                        -------------------------
Image 1:        |       0.4976816177368164      |
Image 2:        |       0.551692008972168       |
Image 3:        |       -0.0614527016878l2805   |
Image 4:        |       0.06895801424980164     |
Image 5:        |       0.08515579998493195     |
Image 6:        |       -0.46063336730003357    |
Image 7:        |       -0.17454519867897034    |
Image 8:        |       0.5667823553085327      |
Image 9:        |       0.6078510284423828      |
Image 10:       |       -0.29884132742881775    |
```

Fig: Similarity score matrix view

Evaluation metric:

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

R@1, R@5, and R@10 are evaluation metrics commonly used in information retrieval tasks, including video frame retrieval. They measure the accuracy of a retrieval system by assessing whether the correct item (in this case, video frame) is retrieved within the top 1, 5, or 10 ranks, respectively.

For example, R@1 (Recall at 1) measures the proportion of queries for which the correct video frame is retrieved as the top-ranked result. Similarly, R@5 measures

the proportion of queries for which the correct video frame is retrieved within the top 5 ranks, and R@10 measures the proportion within the top 10 ranks.

These metrics provide insights into the effectiveness of a retrieval system in returning relevant results to users, with higher values indicating better performance.

3.2 Flickr Testing and Results:

After the saved model is loaded, it is set to eval state. Now its parameters are fixed during testing, preventing any further training. Next, the create_image_embeddings function is defined to generate image embeddings. The function takes images as input, processes them through the ResNet50 model pre-trained on ImageNet and fine-tuned for our work to extract features, and then projects these features using the image projector and normalized output is returned as discussed previously. These embeddings are stored in a list for later use.
We later designed image_retrieval_function, where the input text query undergoes preprocessing to generate text embeddings using a pre-trained DistilBERT model and text projector. Then, the cosine similarity between the normalised text embeddings and each image embedding after they finally belong to the shared embedding space from the training dataset is calculated using the dot product. This process identifies images that are most relevant to the input query based on their similarity scores. Finally, the function retrieves the indices of the n most similar images based on their similarity scores. If the display parameter is set to True, it displays these retrieved images.

Comparison: Flicker30k has observed text-image retrieval recall @1, @5, and @10 of ranges 40-93% , 70-99%, and 79-99% respectively for its best models. We observed a modest recall @1, @5, and @1 of 83.66,, 95.02, and 98.46 respectively establishing its base- groundwork.

3.3 Test Demos for flickr8k Test dataset as image pool on random queries:

Query = “fire” along with similarity score

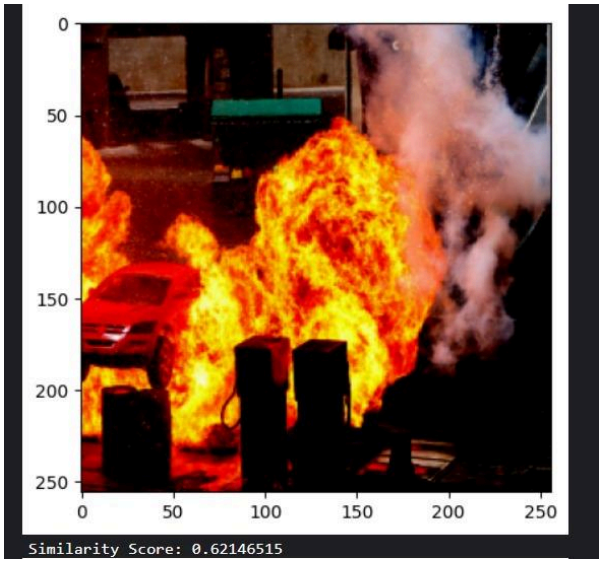


Fig: Testing in flickr8k returning image with highest similarity score 0.62 for the query

```
input_query = "A little boy is playing with a water squirting toy in the swimming pool ",
image_retrieval_function(input_query, n=5, display=True)
```

```
100%|██████████| 8091/8091 [00:00<00:00, 22116.79it/s]
```

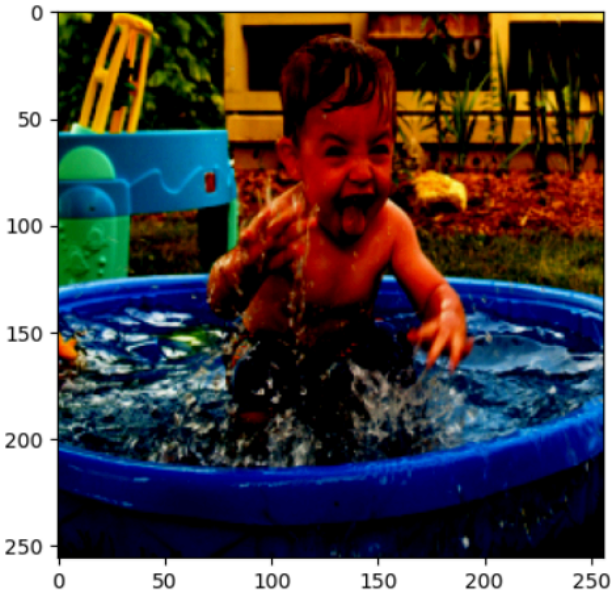


Fig: Testing in flickr8k

```
input_query = "A group of woman in shirts are singing",
image_retrieval_function(input_query, n=5, display=True)
```

```
100%|██████████| 8091/8091 [00:00<00:00, 22219.34it/s]
```

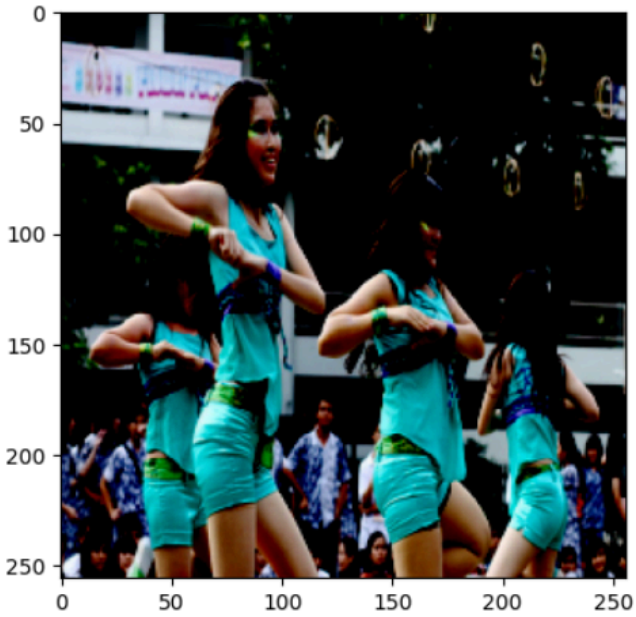


Fig: Testing in flickr8k

```
input_query = "Two tired people rest on couches at an art museum ",
image_retrieval_function(input_query, n=5, display=True)
```

```
100%|██████████| 8091/8091 [00:00<00:00, 21635.60it/s]
```

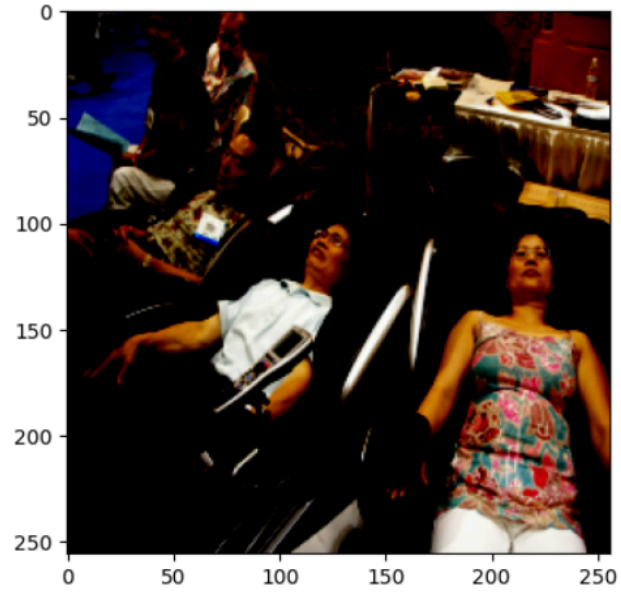


Fig: Testing in flickr8k

```
input_query = "A snowboarder sits on a slope",
image_retrieval_function(input_query, n=5, displa
```

```
100%|██████████| 8091/8091 [00:00<00:00, 21871.14it/s]
```

Fig: Testing in flickr8k

3.4 MSRVTT testing results:

This system not only downloads videos from the internet but also breaks them down into individual frames, capturing the essence of every moment. By embedding these frames into a mathematical embedding space, our model learns to understand the visual content. Then, when presented with a textual query, it magically matches it with the most relevant frames after they are in the same embedding space, showcasing the power of artificial intelligence in action.

display_frames_matching_query function. This function serves as the interface between our video frame retrieval system and the MSRVTT dataset. When provided with a video URL and a textual query, it initiates the retrieval process. First, it downloads the video from the specified URL and extracts individual frames along with their timestamps. The video file is opened using cv2.VideoCapture(), and its frames are read sequentially using cap.read(). Frames are extracted at a specified interval (interval), which is calculated based on the video's frames per second (fps), initially. But later to make the process

faster and efficient by default, one frame per second is extracted as specified by us. For each extracted frame, its corresponding timestamp (in seconds) is calculated based on the current position of the video capture (cap.get(cv2.CAP_PROP_POS_MSEC)). This timestamp represents the time at which the frame occurs in the video. Later we free up the system.

Then, utilizing our trained model, it identifies frames that best match the given query. These selected frames, along with their timestamps, are then displayed interactively using Matplotlib. Extract url, and query from wos of msr-vtt that we got from hugging face for testing.

Comparison: MSRVTT has observed text-Video retrieval recall @1, @5, and @10 of ranges 4.2-63.9% , 20.9-84.3%, and 19.9-89.6% respectively for its proposed multimodal models that incorporate comprehensive video embedding, by harnessing audio, and temporal features to make the model more complex and heavy. It is the recall for video moment retrieval so not exactly our use case since it works on action. We observed a modest recall @1, @5, and @1 of 54.86, 72.56, and 79.50 respectively. Since our model only has image embeddings, its light, less complex and faster and give efficient assistance to its user to locate specific frames

Testing Examples:

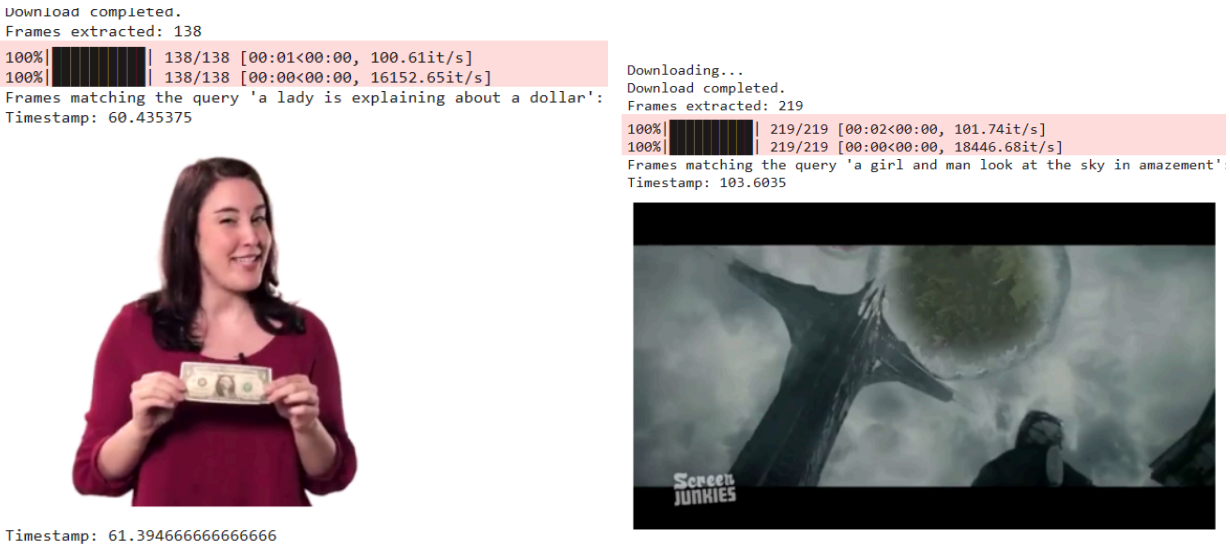


Fig: Examples for MSRVTT Testing

CPU& GPU Usage Details during processing of videos at an instant:

- CPU: 157% RAM: 2.8 GB
- GPU: 16% GPU Memory: 3.5GB

Frames processed per second ranged:12-17sec

Screenshot of an example:

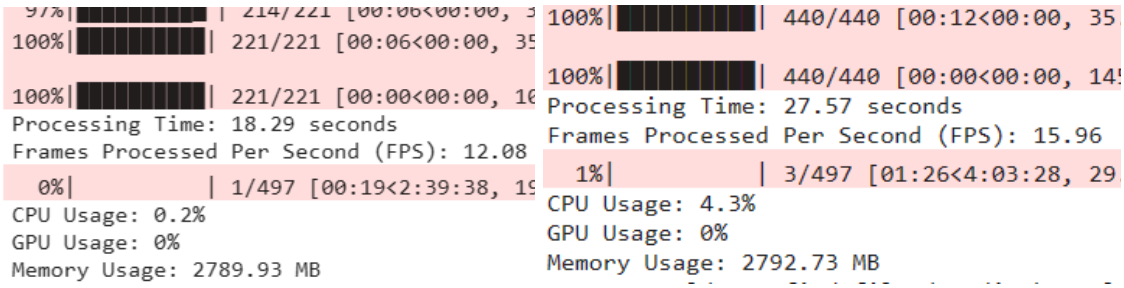
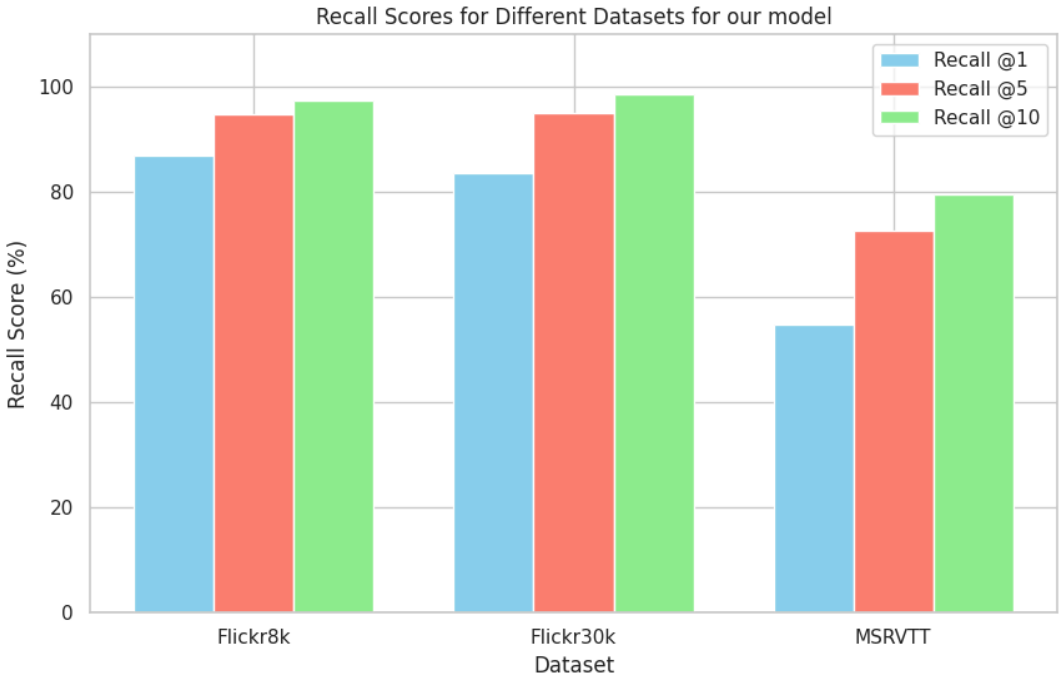


Fig: Kaggle GPU performance on MSRVTT Testing:



	Recall @1	Recall @5	Recall @10
Flickr8k	82.83	94.68	97.33

Flickr30k	83.66	95.02	98.46
MSRVTT	52.86	72.56	79.50

Testing for real world videos available on youtube randomly:

1. Hum saath saath movie song search

```
video_url= "https://www.youtube.com/watch?v=3UuSmbx6Xsg
query="Riding on decorated elephants are a group of peo
display_frames_matching_query(video_url, query)
```

```
Downloading...
Download completed.
Frames extracted: 372
100%|██████████| 372/372 [00:03<00:00, 96.03it/s]
100%|██████████| 372/372 [00:00<00:00, 24202.04it/s]
Frames matching the query 'Riding on decorated elephants are a grou
Timestamp: 21.0
```

```
Timestamp: 55.0
```

2. Lagaan movie ball catching scene search

```
video_url= "https://www.youtube.com/watch?v=vAmsPNCRKro&pp=ygUMbGFnYWFuIG1vdmll"
query="a man catching a ball while playing cricket "
display_frames_matching_query(video_url, query)
```

```
Downloading...
Download completed.
Frames extracted: 11291
100%|██████████| 11291/11291 [01:57<00:00, 95.74it/s]
100%|██████████| 11291/11291 [00:00<00:00, 25775.52it/s]
Frames matching the query 'a man catching a ball while playing cricket ':
Timestamp: 8141.508374999999
```

```
Timestamp: 7696.397041666666
```

```
Timestamp: 8048.457083333333
```

CHAPTER 4

4.1 Steamlit demo Application details

First, we wrote our Streamlit application in app.py, which is a really neat tool for building interactive web apps with Python. Our app allows users to input a YouTube video link and a text query, and it extracts relevant frames from the video based on that query.

A simplified explanation of what's happening:

- User Input: Users can input a YouTube video link and a query related to the content of the video.

- Frame Extraction: When the user clicks the "Extract Frames" button, the app downloads the specified YouTube video and extracts frames from it. It then uses a pre-trained ResNet model to generate image embeddings for each frame.

- Text Embeddings: The app also generates text embeddings for the user's query using a pre-trained DistilBERT model.

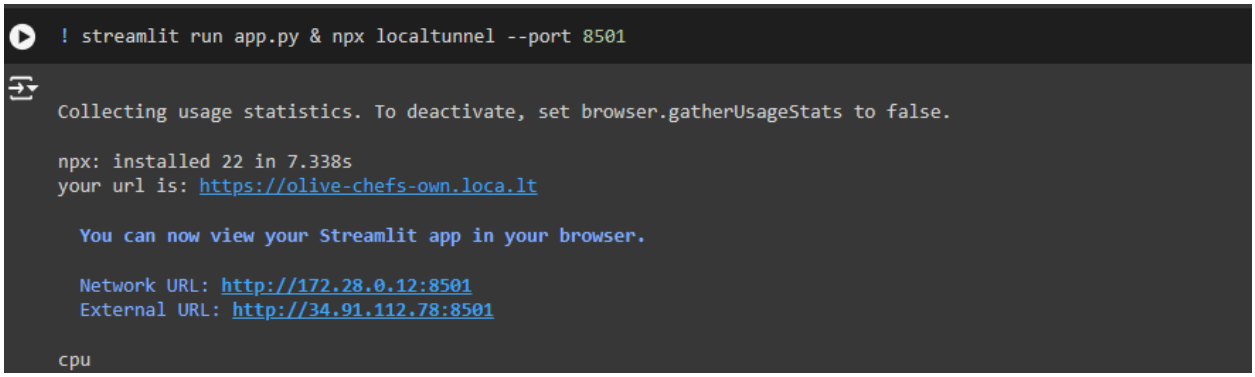
- Frame Retrieval: It calculates the similarity between the text embeddings of the query and the image embeddings of each frame. Based on this similarity score, it selects the top frames that best match the query.

- Display: The selected frames are displayed along with their timestamps. Additionally, the app provides the option to view the entire video with marked timestamps corresponding to the selected frames.

- Performance Metrics: Throughout this process, the app calculates various performance metrics such as download speed, extraction speed, CPU usage, memory usage, frames processed, and frames

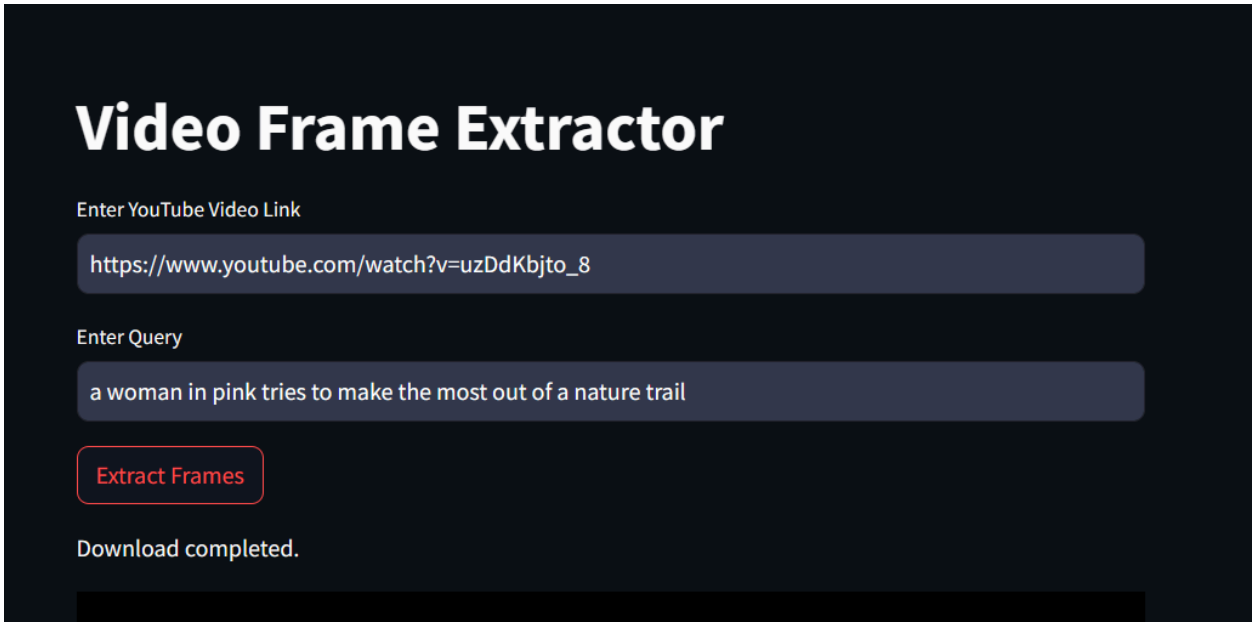
per second (FPS). These metrics help assess the efficiency and resource usage of the frame extraction process.

After starting the Streamlit app, we used localtunnel, a tool that creates a secure tunnel from our local machine to the internet. This allows us to share our Streamlit app with others, even though it's running on our local computer. By combining Streamlit with localtunnel, we can easily showcase our app to friends or colleagues without having to deploy it to a web server.



Overall, it's a simple yet effective way to share our project and get feedback from others. Plus, it's really satisfying to see our work in action on the web, even if it's just temporarily!

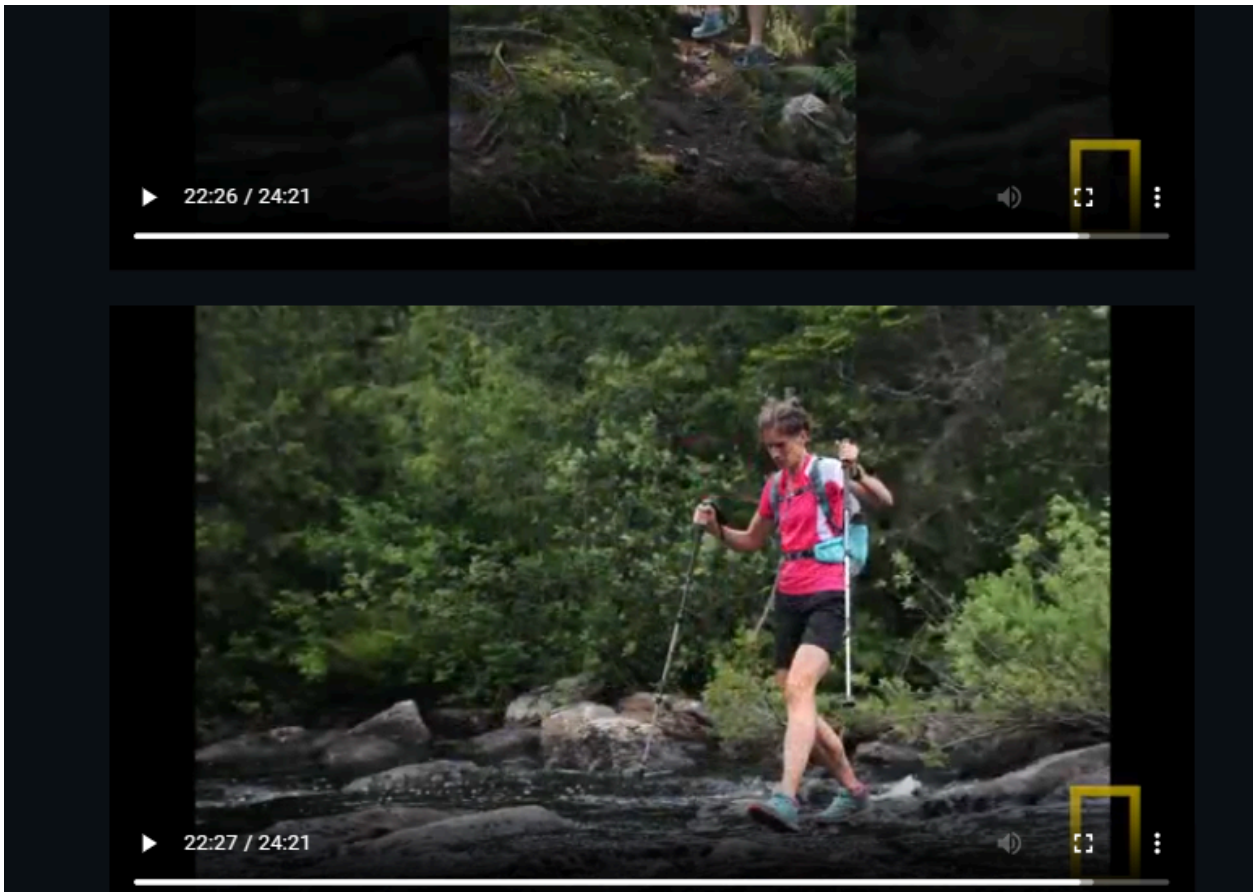
Demo:


Video Frame Extractor
Enter YouTube Video Link
https://www.youtube.com/watch?v=uzDdKbjto_8
Enter Query
a woman in pink tries to make the most out of a nature trail
Extract Frames
Download completed.

Timestamp: 1346.9456

Timestamp: 1347.9132333333334

Timestamp: 1381.7804



In our project, we implemented various performance metrics to assess the efficiency and effectiveness of our video frame retrieval system in real-world scenarios. Here's a breakdown of what we measured:

- Downloading Speed: We calculated the speed at which the YouTube video was downloaded. This metric gives us an idea of how quickly the video can be accessed and processed by our system.

- Extraction Speed: This metric measures how fast our system can extract frames from the downloaded video. It indicates the efficiency of our frame extraction process.

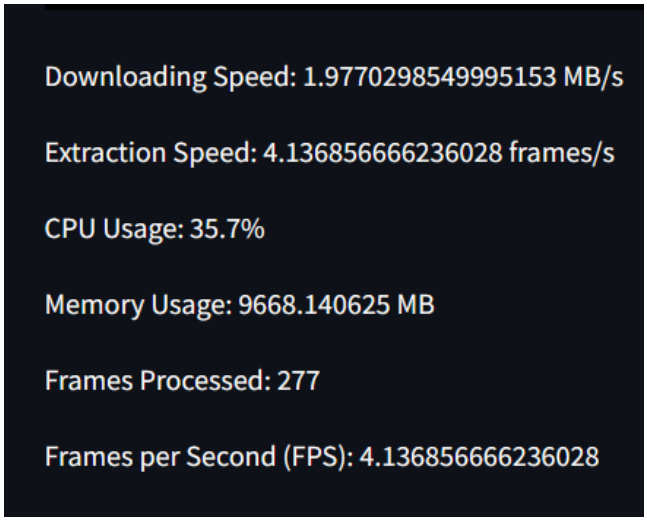
- Frames per Second (FPS): We calculated the number of frames processed per second during the extraction process. A higher FPS value implies smoother and faster frame extraction.

- CPU Usage: We monitored the percentage of CPU resources utilized by our system during the extraction process. This helps us understand the computational demands of our application.

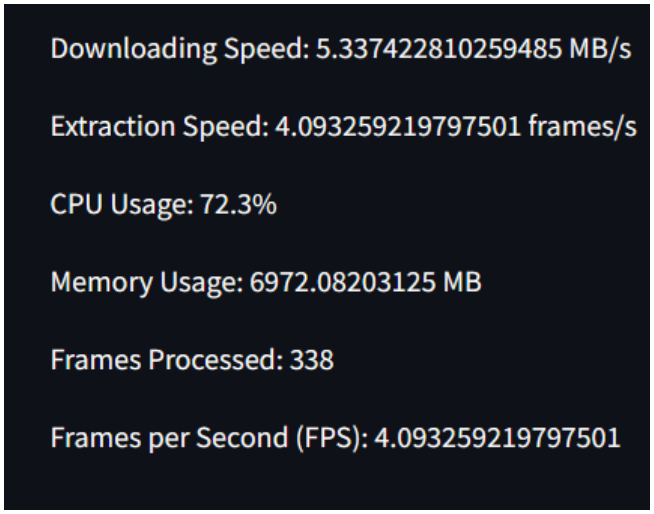
- Memory Usage: We measured the amount of memory consumed by our system during operation. This metric helps ensure that our system is not overly taxing on system resources.

- Frames Processed: We counted the total number of frames processed by our system. This metric gives us an overview of the scale of video data handled by our application.

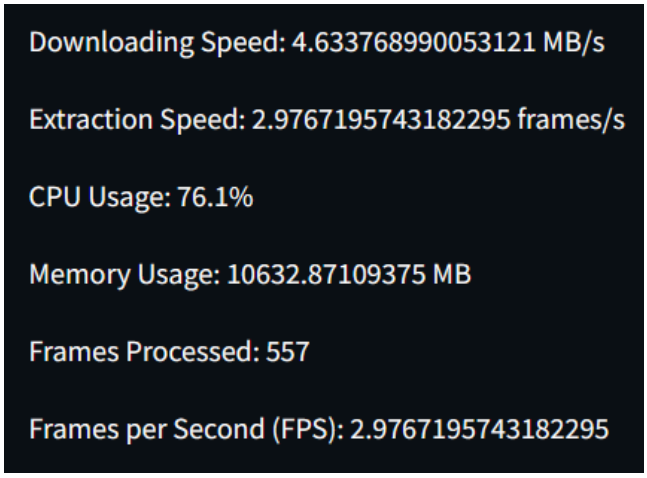
These performance metrics provide valuable insights into the efficiency, scalability, and resource requirements of our video frame retrieval system.
Our performance metrics on Colab cpu:
For Video length: 4.37



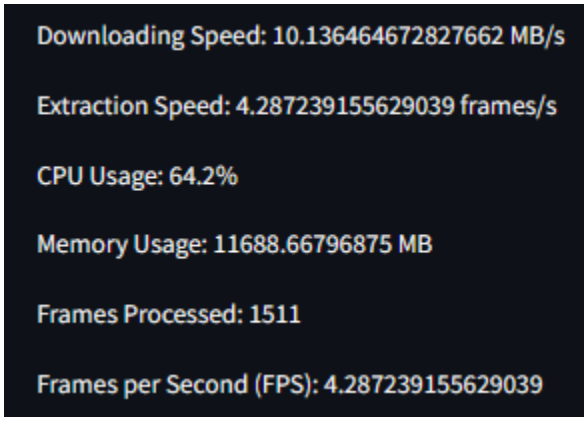
For Video Length: 5:37



For Video length:8.58



For video length: 24.21



CHAPTER 5

Conclusion

We introduced multi-modal fusion architecture based on image and text encoders, projectors , utilising contrastive loss with a learnable temperature and other hyperparameters to obtain our final model. A ResNet50-based image encoder and, a transformer-based architecture for text capable of attending on different parts of the input sequence simultaneously, capturing various aspects of context and dependencies. We incorporate this image encoder and a caption encoder in a cross-modal framework to perform caption-video matching and obtain efficient results for video frame retrieval. By harnessing the synergy between text and image modalities, we saved up on complexity and space required for audio and temporal embeddings, our project represents a paradigm shift in multimedia retrieval, offering a more intelligent and user-centric approach.

Key differentiators of our approach include leveraging image-text models for moment retrieval, enhancing recall rates through multimodal architectures, overcoming limitations in existing video moment retrieval systems, democratizing access to timestamped videos, and empowering users with greater control over content navigation.

The system's ability to accurately retrieve specific video frames based on textual queries, coupled with its efficiency and scalability, opens up new possibilities for content discovery and exploration in the rapidly evolving landscape of online video content.

Through rigorous testing and performance evaluation on Flickr datasets and the MSRVTT benchmark dataset, our system demonstrates superior recall rates and overall effectiveness in video frame retrieval tasks. Additionally, the development of a Streamlit application allows for real-world testing and demonstrations, further highlighting the practical utility and versatility of our solution.

In summary, our project stands at the forefront of multimedia retrieval innovation, offering an efficient solution for precise video frame retrieval based on textual queries. As future work, we would like to work on audio and temporal encoding for video and text.

References

[1]Junyu Gao and Changsheng Xu. Fast video moment retrieval. 2021.

[2]Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. 2021.

[3]Nicolas Carion? , Francisco Massa? , Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers .2020

[4]Taivanbat Badamdorj, Mrigank Rochan, Yang Wang, Li Cheng. Contrastive Learning for Unsupervised Video Highlight Detection.

[5]Joao Carreira , Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. 2017

[6]Shaoxiang Chen, Yu-Gang Jiang. Semantic Proposal for Activity Localization in Videos via Sentence Query.

[7]Ye Liu, Siyuan Li, Yang Wu, Chang Wen Chen, Ying Shan, Xiaohu Qie. UMT: Unified Multi-modal Transformers for Joint Video Moment Retrieval and Highlight Detection.2022.

[8]Valentin Gabeur, Chen Sun, Karteek Alahari, Cordelia Schmid. Multi-modal Transformer for Video Retrieval. 2020.

[9] Jin Yang, Ping Wei, Huan Li, Ziyang Ren. Task-Driven Exploration: Decoupling and Inter-Task Feedback for Joint Moment Retrieval and Highlight Detection. 2024

[10]WonJun Moon1, Sangeek Hyun1, SangUk Park, Dongchan Park, Jae-Pil Heo. Query-Dependent Video Representation for Moment Retrieval and Highlight Detection.

[11]Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell,Bryan Russell. Localizing Moments in Video with Natural Language. 2017.

[12]Anthony Gillioz, Jacky Casas, Elena Mugellini, Omar Abou Khaled. Overview of the Transformer-based Models for NLP Tasks.

[13]Yan Huang, Qi Wu, Liang Wang. Learning Semantic Concepts and Order for Image and Sentence Matching. 2017.

[14]Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li. A Survey on Multimodal Large Language Models. 2024.

[15]Dahun Kim, Anelia Angelova, Weicheng Kuo.Region-Aware Pretraining for Open-Vocabulary Object Detection with Vision Transformers. 2023.

[16]Victor SANH, Lysandre DEBUT, Julien CHAUMOND, Thomas WOLF. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. 2020.

[17]Alexander Turchin , Stanislav Masharsky , Marinka Zitnik. Comparison of BERT implementations for natural language processing of narrative medical documents. 2023.