Name: Ahsan Bilal CMS: 341480

Name: Muhammad Haseeb Nizami CMS: 345783

Image to text using clip

Project Report

Declaration

I hereby declare that this project report entitled "Image-Text Matching using CLIP" is the result of my own work, except where explicitly stated otherwise in the references. I have cited all sources of information and used data in accordance with ethical guidelines.

* Ahsan Bilal
* Muhammad Haseeb Nizami

Dedication

I dedicate this project to my family, whose unwavering support and encouragement have been a constant source of motivation throughout this journey. Their love and belief in my abilities have fueled my determination to pursue excellence in my endeavors.

Abstract

The Image-Text Matching using CLIP project presents a comprehensive framework for matching images with textual descriptions using the CLIP (Contrastive Language-Image Pretraining) model. The project leverages pre-trained models for image and text encoding to create a unified model capable of understanding and comparing both modalities. By encoding images and text into fixed-size vectors and projecting them into a shared space, the model learns to identify relevant matches between images and their corresponding textual descriptions. The project includes components for dataset preparation, image and text encoding, projection head, and inference. Experimental results demonstrate the effectiveness of the CLIP framework for image-text matching tasks, with potential applications in image retrieval, recommendation systems, and accessibility tools.

1. Introduction

The aim of this project is to develop a robust model capable of matching images with textual descriptions using the CLIP (Contrastive Language-Image Pretraining) framework. This task holds significant importance in various applications such as image retrieval, content-based recommendation systems, and accessibility tools for visually impaired individuals. By encoding both images and text into fixed-size vectors and comparing them in a shared space, the model can identify relevant matches between images and their corresponding textual descriptions.

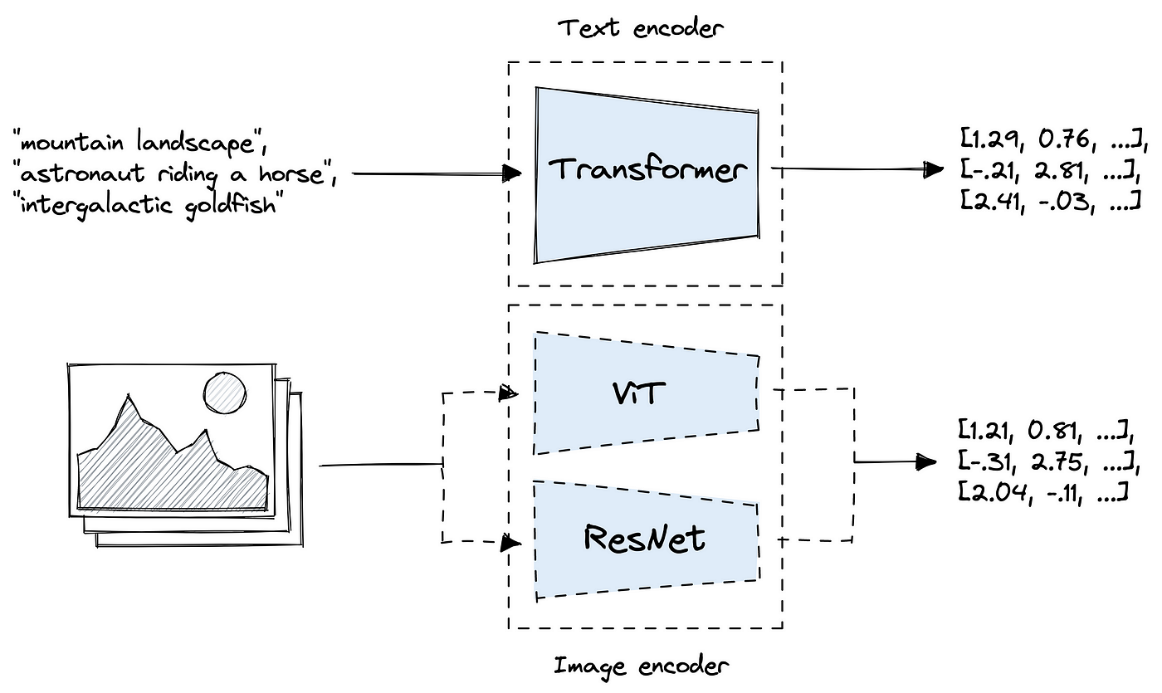
2. Background

The CLIP framework, introduced by OpenAI, represents a significant advancement in multimodal learning. It leverages a large-scale dataset of image-text pairs to pretrain a single model capable of understanding both visual and textual information. CLIP employs a contrastive learning approach, where the model learns to map semantically similar images and text to nearby points in the embedding space, while pushing dissimilar pairs apart. This self-supervised learning paradigm enables CLIP to generalize well across a wide range of tasks without task-specific supervision.

3. Literature Review

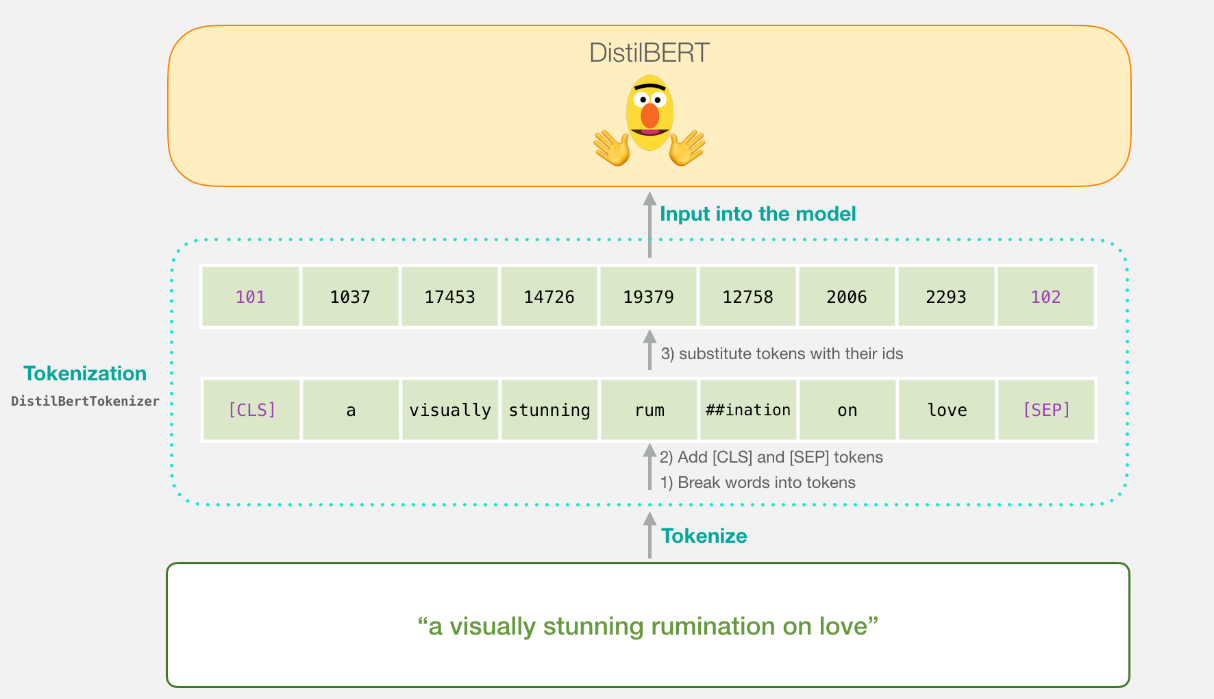
Prior work in image-text matching includes approaches based on handcrafted features, such as bag-of-words representations and visual feature extraction methods. However, these methods often suffer from limited generalization and scalability. With the advent of deep learning, multimodal models have gained popularity for their ability to learn joint representations of images and text. Notable examples include the Joint Embedding model, which learns a shared embedding space for images and text, and the Visual Semantic Embedding model, which leverages visual and textual semantics for image retrieval.

The CLIP framework builds upon these advancements by training a single model on a diverse range of image-text pairs, allowing it to learn rich representations of both modalities. By pretraining on a large dataset, CLIP achieves strong performance across various downstream tasks, including image classification, object detection, and image-text retrieval.



4. Dataset Preparation

The dataset consists of image-text pairs collected from various sources, including image-caption datasets and web-scraped images with associated textual descriptions. The dataset class preprocesses the textual descriptions by tokenizing them using a **DistilBERT** tokenizer, which breaks down the text into tokens and converts them into token IDs. Images are loaded and preprocessed, including resizing, normalization, and augmentation if desired.



5. Image Encoder

The image encoder component utilizes a pre-trained ResNet50 model to encode images into fixed-size vectors. The ResNet50 architecture consists of convolutional layers followed by global average pooling, which aggregates spatial features into a vector representation. The resulting vector serves as the embedding for the input image.

6. Text Encoder

Textual descriptions are encoded into fixed-size vectors using a DistilBERT model. The DistilBERT tokenizer converts the text into token IDs, which are then fed into the DistilBERT model to obtain embeddings. The final embedding of the CLS token represents the entire textual description, capturing its semantic meaning.

7. Projection Head

The projection head component projects both image and text embeddings into a shared space for comparison. This component ensures that the embeddings have similar dimensions, facilitating meaningful comparisons between images and text. The projection head consists of linear layers that map the input embeddings to a lower-dimensional space while preserving their semantic information.

Data Structure

In this project, the embeddings and data are stored using tensors, which are multi-dimensional arrays provided by the PyTorch library. Tensors are the fundamental data structure used for representing numerical data in PyTorch, and they provide efficient computation and storage capabilities suitable for deep learning tasks.

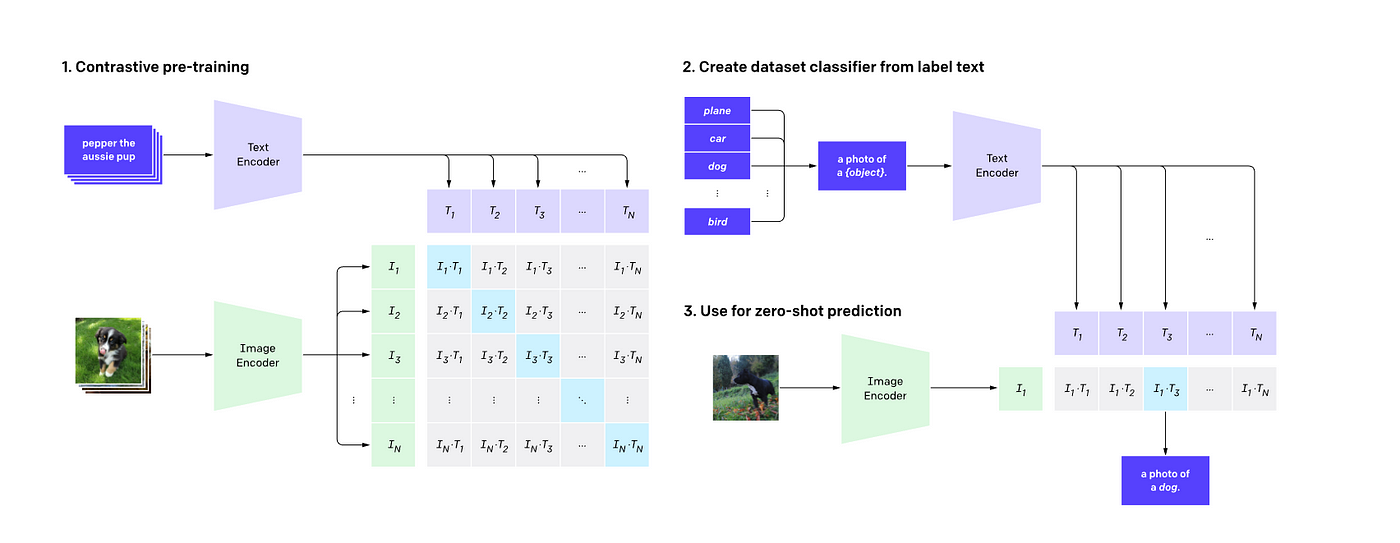
**Embeddings:** The embeddings produced by the image and text encoders are stored as tensors. These tensors have fixed dimensions and represent the numerical representations of images and textual descriptions in a continuous vector space.

**Data:** The input data, including images and textual descriptions, are processed and stored as tensors before being fed into the respective encoder modules. For images, the data is loaded and preprocessed using functions provided by PyTorch's torchvision module, resulting in tensors representing the image pixel values. For textual descriptions, the data is tokenized and converted into token IDs using the DistilBERT tokenizer, and then passed as input to the DistilBERT model, which produces embeddings stored as tensors.

By using tensors to store embeddings and input data, the project leverages the efficient computation and optimization capabilities provided by PyTorch, facilitating seamless integration with deep learning models and frameworks.

8. CLIP Model

The CLIP model combines the image and text encoders with the projection head to create a unified model for image-text matching. During training, the model computes the loss based on the dot product similarity between image and text embeddings. The model aims to minimize the distance between relevant image-text pairs and maximize the distance between irrelevant pairs, effectively learning to discriminate between them in the shared embedding space.



9. Inference

After training, the model can perform inference by retrieving relevant images based on textual queries. Given a text query, the model computes embeddings for both the query and the images in the validation set. It then finds the images with the highest similarity to the query based on the dot product between their embeddings.

10. Conclusion

In conclusion, this project demonstrates the effectiveness of the CLIP framework for image-text matching tasks. By leveraging pre-trained models for image and text encoding, the CLIP model is capable of accurately matching images with their corresponding textual descriptions. Future work may involve fine-tuning the model on specific domains or exploring alternative architectures for further improvements.

11. Acknowledgements

We would like to express our gratitude to OpenAI for their pioneering work on the CLIP framework, which has greatly influenced this project. Additionally, we acknowledge the contributions of the Hugging Face and PyTorch communities for providing valuable tools and libraries for natural language processing and computer vision tasks.

12. References

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020.

Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., & Ng, A. Y. (2014). Grounded compositional semantics for finding and describing images with sentences. Transactions of the Association for Computational Linguistics, 2, 207-218.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., & Parikh, D. (2015). Vqa: Visual question answering. In Proceedings of the IEEE international conference on computer vision (pp. 2425-2433).

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165.