Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Bachelor's diploma thesis

in the field of study Data Science

Implementation of multimodal caption generator model based on natural language processing and image analysis

Maja Andrzejczuk

student record book number 313467

Julia Przybytniowska

student record book number 313523

thesis supervisor

D.Sc. Marcin Paprzycki

WARSAW 2024

.....

supervisor's signature

.....

author's signature

Abstract

Implementation of multimodal caption generator model based on natural language processing and image analysis

In today's rapidly developing technology, the focus is on developing innovative solutions that overcome the limitations of previous implementations. However, what is becoming forgotten is their proper presentation to people unconnected with the development of artificial intelligence and technology. Solutions are created, described in papers and presented at conferences, but only some have dedicated demos and even fewer are used to create free applications and seamlessly support users. Our application stands out by prioritizing user-friendly interaction with advanced Visual-Language models. With an emphasis on visual language modelling, we present a web application that integrates the CLIP model for semantic image retrieval and BLIP for image captioning. The implementation we have created not only makes artificial intelligence more accessible but also promotes continuous improvement of Machine Learning models by saving user suggestions for generated predictions.

Keywords: Machine Learning, Multimodality, Web Application, Semantic Image Search, Image Captioning, Flickr

Streszczenie

Implementacja multimodalnego modelu generowania tytułów obrazów wykorzystującego metody przetwarzania języka naturalnego i analizy obrazów

W dzisiejszym dynamicznie rozwijającym się świecie technologii skupiamy się na opracowywaniu innowacyjnych rozwiązań, pokonując limity poprzednich implementacji. Coraz częściej zapominana jest kwestia prawidłowej ich prezentacji dla osób niezwiązanych z rozwojem sztucznej inteligencji i technologii. Choć wiele rozwiązań jest tworzonych, opisywanych w dokumentach i prezentowanych na konferencjach, tylko nieliczne z nich posiadają dedykowane demo, a jeszcze mniejsza ich część jest wykorzystywana do stworzenia bezpłatnych aplikacji, wspierających użytkowników. Nasza aplikacja wyróżnia się poprzez priorytetowe traktowanie przyjaznej dla użytkownika interakcji z zaawansowanymi wizualno-językowymi modelami, z opracowanymi metodami do wspólnego przetwarzania obrazu i tekstu. Stworzyłyśmy aplikację internetową, która integruje model CLIP do semantycznego wyszukiwania obrazów dla danej frazy oraz BLIP do podpisywania obrazów w sposób informatywny. Nasza implementacja nie tylko czyni sztuczną inteligencję bardziej dostępną, ale również sprzyja jej ciągłemu doskonaleniu, poprzez zapisywanie sugestii użytkowników dotyczących wygenerowanych predykcji.

Słowa kluczowe: Uczenie Maszynowe, Multimodalność, Aplikacja Webowa, Semantyczne Wyszukiwanie Zdjęć, Generowanie Tytułów, Flickr

Contents

1. Introduction

In today's dynamically developing technological landscape, a key aspect is to focus on creating innovative applications that effectively support users in using modern technological solutions. In this context, one of our priorities is to encourage greater use of artificial intelligence models by users.

As the field of visual language modelling progresses, there is a growing demand for applications that enable users to effectively use these advanced models in a simple, understandable and friendly way. Our application offers an intuitive environment in which users can effectively interact with images, using the full potential of available Vision-Language Models.

In the context of our project, it is crucial not only to facilitate access to advanced technologies but also to constantly improve them thanks to the involvement of users. Within the application, it is possible to select model-generated answers that, in users' opinion, are the most relevant. Users can also formulate their proposals for the correct output. These functionalities not only increase user engagement but also allow for collecting valuable feedback that can potentially be used to further train and improve artificial intelligence models. This approach lays the foundation for the future implementation of active learning techniques.

The key element of our application is the integration of two advanced models: BLIP in the image captioning task and CLIP in the semantic image search task. The BLIP model was used to generate text descriptions of images, which allows for the precise transformation of visual content into statements understandable to the user. In turn, the CLIP model is used for semantic image retrieval, enabling the identification and discovery of images using text descriptions.

The implementation of our project is based on the use of technologies such as Django REST, which enables effective management of the application backend, and React, which provides a solid foundation for creating a responsive and modern user interface. Our application aims to facilitate the use of the potential of visual language models, making the process of interacting with images more accessible, effective and consistent with user expectations.

1.1. Contribution

Table 1.1: Contribution to the project

Author	Implementation	Document
Maja Andrzejczuk	Backend: - Custom User Model (Functionality, Testing) - Semantic Image Search (Testing), - Caption Generator (Testing), Frontend: - Registration and Login View (Main layout, User authorization), - Semantic Image Search View (Upload from catalog), - Image Captioning View (Upload from catalog, Layout).	Introduction (1), BLIP's description (3.2), Dataset used for application testing (4), Problem statement (5), Application implementation (6), Experimental results (7), Concluding remarks (8)
Julia Przybytniowska	Backend: - Semantic Image Search functionality (CLIP), - Caption Generator functionality (BLIP), - Feedback functionality (saving on server, for both - Semantic Image Search and Image Captioning). Frontend: - Main Page View, - Semantic Image Search View (Main layout, Upload from device, functionality to search from Flickr images, Feedback), - Image Captioning View (Main layout, Upload from device, Feedback).	Abstracts, Theoretical Background (2), CLIP's description (3.1)

2. Theoretical Background

To understand the operation of our web application, it is crucial to explore the basic concepts that shape its methodology. It involves Artificial Intelligence with Machine Learning, Natural Language Processing and Deep Neural Networks to generate results, as well as the evaluation process to examine them. Together, these components generate predictions, forming the basis for an advanced and intelligent user experience.

Artificial intelligence, also known as AI, is a field of science that mimics human intelligence in software-coded heuristics. It enables machines to learn from the past and experience, based on the data, so that they can adapt to the future and know how to respond to it. The spectrum of artificial intelligence applications spans limitlessly, encompassing fields such as reasoning, planning, and learning, as well as the complex fields of natural and visual language processing. The horizon of possibilities in the area of artificial intelligence continues to expand, heralding a future in which intelligent systems bring new dimensions of understanding and innovation.

2.1. Machine Learning

Machine Learning, often shortened to ML, is a branch of Artificial Intelligence (AI) combined with computer science that aims to predict results based on input and previous records. It focuses on the use of statistical algorithms that can effectively generalize and perform tasks without explicit instructions, in a way similar to intelligent human behavior. At their core, the ML algorithms can improve automatically through experience by learning the hidden patterns of the datasets used, allowing computers to make decisions with increased accuracy on new, similar input.

Fundamental to ML is the concept of supervised learning (21), where algorithms are trained on labelled datasets to make predictions or classifications. This involves optimizing parameters through techniques like gradient descent and adjusting hyperparameters for optimal model performance (31). Unsupervised learning techniques (20), such as clustering and dimensionality

reduction, are employed to extract hidden structures and relationships within unlabeled data.

Machine learning algorithms are used in computer vision, speech recognition, and semantic analysis, but also in medicine and business, where manual computation is either too costly or too complex. ML is valued for its ability to control quality, automation and customization.

2.1.1. Training and test data

A dataset, containing various pieces of data, serves as an essential tool to instruct machine learning algorithms on identifying patterns within the entire dataset. It is crucial to provide the AI model with high-quality and quantity data, that will allow the algorithm to analyze trends and make decisions. The best performance of the model can be achieved by dividing the dataset into training, validation, and test subsets. Training data is the largest subset of our real-world data that is fed into a mathematical model to learn hidden commands. The validation set is a subset of the training data used to evaluate the model's performance during training, enabling fine-tuning of hyper-parameters and settings. After training, the unseen data (test subset) is then used to test the final model, enabling evaluation and improvement.

2.2. Natural Language Processing (NLP)

Natural language processing, also known as NLP, is a sub-discipline of artificial intelligence (AI) that aims to give computers the ability to understand text and spoken words in the same way that humans do. NLP systems use a variety of techniques, including word embedding, which represents words as dense vectors to capture semantic relationships, making them capable of interpreting the meaning and sentiment behind the language. Sequence-to-sequence models (27), such as recurrent neural networks (RNNs) (26) and transformers (29), combine computational linguistics with statistical and deep learning models and solve tasks such as machine translation, summarization and sequence data generation. The following sections will describe these approaches and examples of them in more detail.

2.3. Deep Learning

Deep Learning is a branch of Machine Learning, that has recently been considered a state-of-the-art solution to various image analysis and computer vision problems. The approach is capable of learning complex trends and hierarchical representations from data, thanks to its

artificial neural network architecture. A full mathematical understanding of the concept can be found in the paper from 2023, (9).

2.3.1. Neural Network

A neural network is a mathematical model (ML) inspired by the human brain that has the ability to store and analyze information. The network aims to imitate biological neurons' communication to come up with reasonable decisions, which is possible thanks to its developed structure. Neural networks consist of three layers: input, hidden and output. Into the first of them is transmitted raw data of the original dimension, which is then processed by hidden layers and passed to the output layer in the dimension of the expected result. Every layer consists of perceptrons that are linked to nodes from neighboring layers by connections, each of which is assigned a corresponding strength, often called weight. The network learns from the input data and adjusts these weights during the training process, allowing it to create corrections, minimize prediction error, and discover many hidden relationships between the researched parameters.

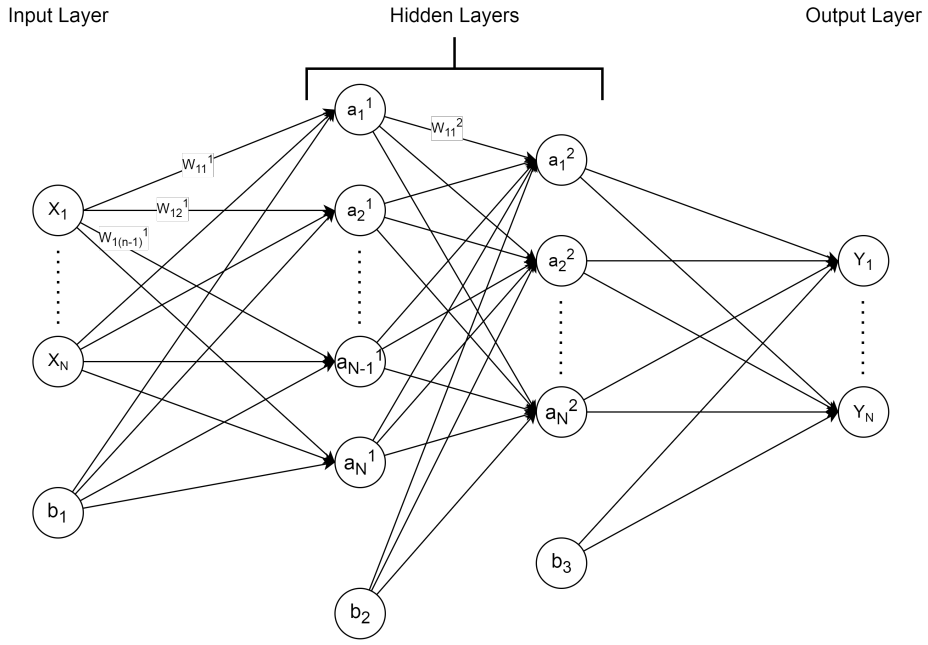

Figure 2.1: Example structure of a neural network with an input, two hidden and output layers together with biases

In order to understand how a network adjusts connection weights and strives for good prediction, one should understand the definition behind a feedforward neural network and its operations. Feedforward Multilayer Perceptron topology is one of the most popular designs of Neural Network architecture, which moves only in one direction - from the input nodes to the output, via nodes located in hidden layers. It consists of two phases:

1. Forward Propagation - In it, input data is fed into the network and propagated further through the network. In each hidden layer, the weighted sum of the data from the previous layer (input in the case of the first hidden layer) is calculated and passed through the previously defined activation function. The process is repeated until the result is obtained at the output layer. The procedure can be demonstrated by the schema:

$$\begin{aligned}
\text{Input layer:} \quad & a^{[0]} = X \\
\text{Hidden layers:} \quad & z_i^{[l]} = \sum_{j=1}^{n^{[l-1]}} W_{ij}^{[l]} a_j^{[l-1]} + b_i^{[l]} \\
& a_i^{[l]} = \sigma(z_i^{[l]}) \\
\text{Output layer:} \quad & z_k^{[L]} = \sum_{i=1}^{n^{[L-1]}} W_{ki}^{[L]} a_i^{[L-1]} + b_k^{[L]} \\
& \hat{y}_k = a_k^{[L]} = \sigma(z_k^{[L]})
\end{aligned}$$

where:
- X represents the input features,
- $z_i^{[l]}$ is a weighted sum of inputs and bias at hidden layer l,
- $a_i^{[l]}$ is an activation function applied to the weighted sum $z_i^{[l]}$ at hidden layer l,
- L represents an output layer,
- $\hat{y}_k$ is a final prediction.

The activation function that is implemented in the network plays a crucial role in the type and quality of the obtained prediction. This function introduces non-linear properties of the model, which makes it possible to learn more complex patterns.

One activation function that is not presented above is a Softmax function, that scales numbers/logits into probabilities. The output is a vector with probabilities for each potential outcome so that for all possible predictions, the values sum up to 1.

$$\text{softmax}_\text{i} = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.1}$$

where:
- x represents an input vector to softmax.

2. Backpropagation - Once the prediction is obtained, the difference between the output and the expected result is calculated. This error is then propagated back through the layers

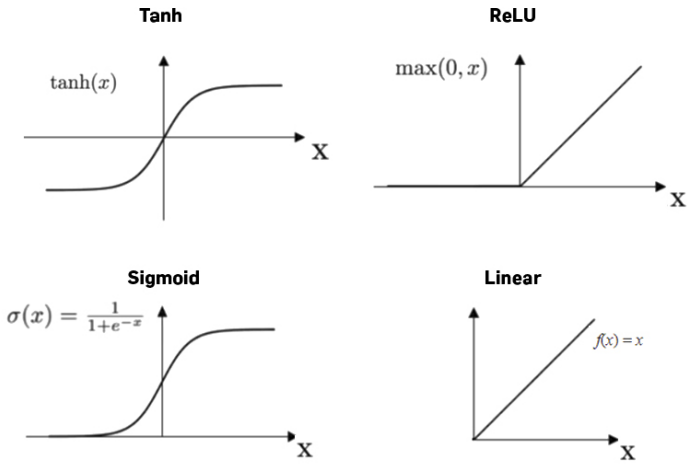

Figure 2.2: Most popular activation functions. Sourced from machine-learning.paperspace.com.

while updating the weights in a way that is described in the paper: (19).

Whereas, Deep Neural Networks utilized in Deep Learning problems are essentially Neural Networks with multiple hidden layers. The depth of the network, that is, the number of hidden layers, allows the model to automatically learn hierarchical and abstract features from raw data, thus highlighting trends for further analysis. Thanks to this ability, Neural Networks have achieved tremendous popularity in Computer Vision and Natural Language Processing tasks.

2.4. Transformers

In order to fully understand the structure of models implemented in our application, it is essential to acknowledge the structure of The Transformer, a solution first introduced in the article "Attention Is All You Need" (29) in 2017.

The Transformer architecture has revolutionized the world of natural language processing, thanks to its outstanding ability to handle text data, which is difficult to process with its sequential characteristics. This neural network takes a text sequence as input, processes it all at once while capturing context, and then returns the text sequence as output.

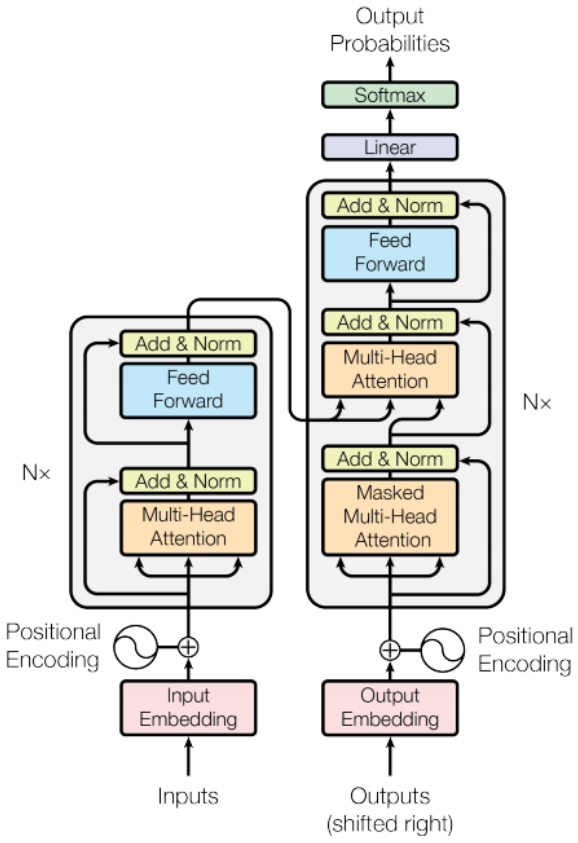

Figure 2.3: Architecture of the Transformer. Sourced from the original paper (29).

The architecture consists of two main components - encoding (a stack of encoders identical in structure, however, with different weights) and decoding (a stack of decoders). The encoder is constructed from a self-attention layer and a Feedforward Neural Network, while the decoder, in addition to these two layers, also has an Encoder-Decoder attention layer between them.

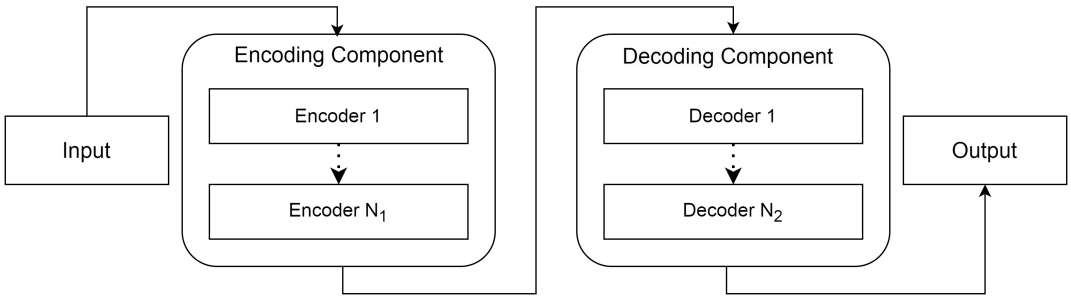

Figure 2.4: High-Level Transformer architecture

2.4.1. Tokenization

The initial and crucial step in NLP tasks involves preprocessing the input text into a format that facilitates a better understanding of the model. This process is called tokenization, as it involves dividing text into smaller units called tokens, which are then converted into corresponding identifiers from a dictionary. Tokens can be representations of different forms, depending on how they are defined, such as words, phrases, subwords or characters. The model is fed with a sequence of integers and returns the same form of data as output, which, after decoding, is a text.

2.4.2. Embedding

When a text sequence is tokenized and passed to the model, the model does not know the meaning of the tokens or the links between them. For this, a process called embedding is performed, which is a representation of the text using a high-dimensional vector. When training the model, a representation of the token data is created so that those with close meaning are close to each other in vector space. The matrix created in this way is a vocabulary, used later in the phase of inference. There are many algorithms constructed, able to accurately embed words and capture their features, that are in detail described in the paper (11).

Independently, the model is fed with a Positional Encoding, which is used to provide a relative position for each token or word in a sequence. It is well known that the meaning of a word is dependent on the words around it and the context of the sentence, so the model needs to be aware of these dependencies in order to be efficient. Word indexing may seem natural in this case, however, it is inefficient, especially for long sentences when these indexes are not easy to compute. The scheme presented in the transformer architecture solves this problem, by using a d-dimensional vector containing information about a specific position in a sentence for each given token. The following formula is used:

$$PE(\text{ position}, 2i) = \sin\left(\frac{\text{position}}{10000^{\frac{2i}{d_{\text{model}}}}}\right)$$

$$PE(\text{ position}, 2i+1) = \cos\left(\frac{\text{position}}{10000^{\frac{2i}{d_{\text{model}}}}}\right)$$

where:

- *position* represents the position of the word within the original sentence,
- i is an index of the dimension,
- d_{model} is a embedding length.

The approach using Sinusoidal functions has several features that allow one to maximize the efficiency of the model. The first is the cyclicity of these functions, repeating patterns, allowing the model in subsequent phases to pay attention to words not necessarily close to the one analyzed. Another important characteristic of this formula is the limited values, bounded from -1 to 1, so words far away in the matrix will have smaller values. Last but not least, sinusoidal functions are independent of the length of the embedded word and easy to compute, so the encoding is able to be continuous and smooth even for sequences of different lengths and positions.

Finally, when embeddings and positional encoding are calculated, the matrices are summed and passed to the first layer of the bottom-most encoder, which is the self-attention layer.

2.4.3. Self-attention mechanism

The self-attention mechanism is a key component of modern machine learning models, particularly when dealing with sequential data such as text data. It is thanks to this mechanism that transformers have become so popular in NLP or Visual tasks.

With the self-attention layer, the model is able to capture dependencies in input sequences. It enables the algorithm to assess the importance of different parts and the relationship between them by looking at other words from the input while encoding a specific word.

Previously, recurrent neural networks (RNNs) were the most popular, an overview of which can be found in the article (26). This neural network architecture was state-of-the-art for modelling sequences, but it had a few weaknesses. Its architecture is easiest to visualize with the help of a diagram:

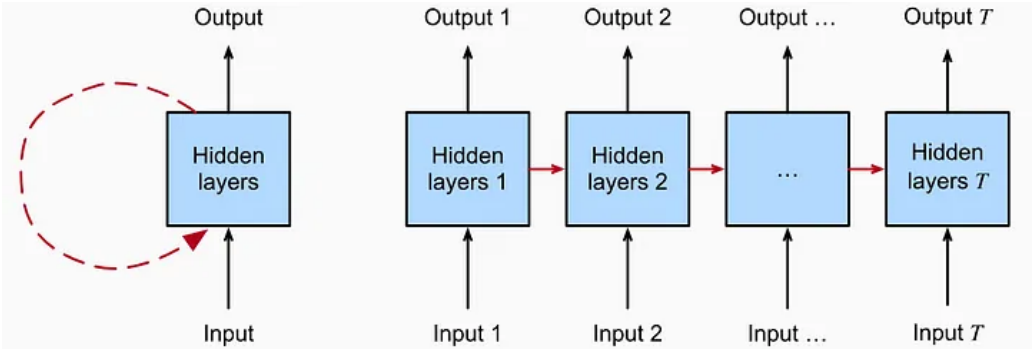

Figure 2.5: Folded and unfolded architecture of RNN. Sourced from Medium.com.

It consists of smaller neural networks, one for each token in the input, and each subsequent one is connected to the network of the previous part of the sentence multiplied by the weight. With long sequences given as input, the learning phase could be very slow and expensive, but the more significant problem is that the meaning of a given token is fully dependent on

previous sequences. Even bidirectional networks (3) did not solve this problem, because they only analyzed the previous and next input data separately and then merged them into the output, so the true meaning could be lost.

The self-attention mechanism addresses these problems by comparing all input sequence embeddings with each other. Every vector representing the single embedding is going through a split for Q - the query vector (indicating what the user is looking for), K - the key vector (what the model is able to offer), and V - the value vector (what the model actually offers), using a simple weight distribution. After that, the self-attention matrix is generated by the following formula:

$$\text{self attention} = \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_k}} + M\right) V \tag{2.2}$$

Firstly, the dot product from the Query and Key matrix is calculated and then scaled by the root of the vectors' length for each element, done to deduce the variance of the product and hold all values in a similar range. Then the matrix of dimension $lengthOfSequence \times lengthOfSequence$ is summed with a Mask matrix. Mask makes sure that the context of the generated words is not affected by those generated after them, in the future. For this reason, the step of adding a mask is optional in the encoder layer, and only used in the decoder. The matrix that will make the model not look at past words is a modified lower-diagonal matrix, in which the main diagonal and all values below it are filled with zeros, and all values above have a negatively infinite value. After that, the softmax function (described in the 2.1 equation) is activated to get a probability distribution for each part of the sequence, which is then multiplied by the Value matrix. The resulting self-attention matrix informs the model how much attention it should pay to other embeddings, in order to fully understand the meaning of the vector being processed.

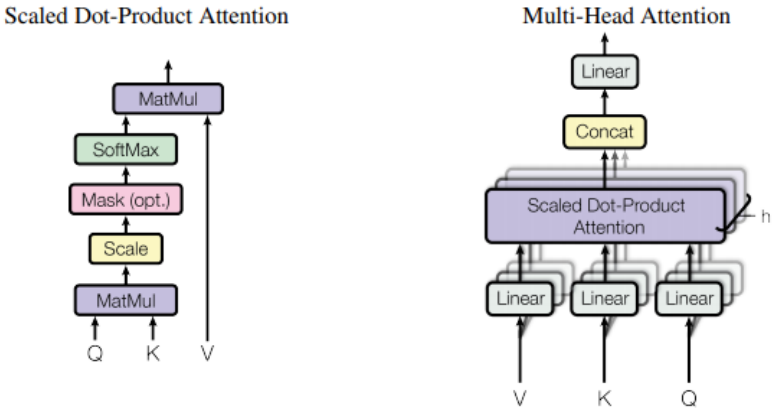

Figure 2.6: Self-attention architecture. Sourced from the original paper (29).

Transformer uses a multi-head attention mechanism, which simplifies to many layers of single-head attention (described in the previous paragraph) and then concatenates the results on top of each other using a trained matrix with weights. This expands the model's ability to focus on different positions and gives the attention layer multiple subspace representations.

One of the key properties of the Transformer is that the word in each position flows through its own path in the encoder, so that the abilities of modern graphics cards are used. There is one more important element inside the encoder/decoder components, such as the residual connection along with Layer Norm after each sublayer, shown in figure 2.7.

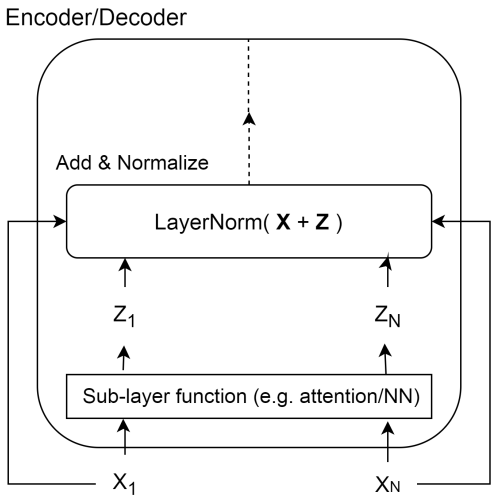

Figure 2.7: Transformers residual connection with layer normalization

Having understood the encoder stack, it is time to delve into the decoder components, in which most layers are shared with the encoder. The main difference is that the decoder processes two inputs and applies multi-head attention, with one of them being masked. One of the inputs is the output of the encoder, specifically its weighted distribution for the K - key and V - value matrices. It passes through the Cross Multi-Head Attention layer along with the Q - query matrix returned by Masked Multi-Head Attention. Initially, the inference decoder has no information on the input named "output shifted to the right," only the <start> token, and will be fed continuously when the next output in the sequence is predicted (it will have all outputs already predicted from 0 to i-0 before the i-th prediction). This solution ensures that the model will not be influenced by future predictions and will rely only on already-predicted outputs. The prediction will continue until the decoder is fed with the <end> token.

2.5. Vision Transformers

The Vision Transformer, also known as ViT, was first introduced in 2020 (5) as a state-of-the-art solution for various computer vision problems. This deep Neural Network architecture is breaking the gap between language and image, thus creating a unified field of science. Its implementation relies on The Transformer architecture (described in Section 2.4) with key modifications for more efficient image processing.

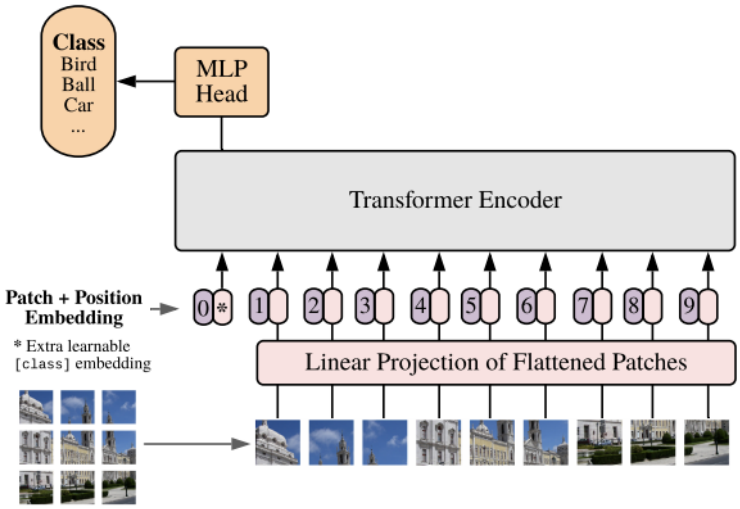

Figure 2.8: Architecture of Visual Transformer. Sourced from the original paper (5).

The goal was to adapt the transformer to process visual data, while preserving the features that make the model perform so well. The core modification used in the vision transformer is the transformation of the input image into non-overlapping parts of the image, called patches. Image patching is the corresponding tokenization process, that converts an image into a sequence of small images and enables the model to fully utilize the capabilities of the Self-attention mechanism.

After that, patches are forwarded to the Linear Projection Layer, where they are transformed into embedded vectors. These vectors are composed of the features of each patch, creating a dimension that can already be passed to the transforming encoder.

In addition to the generated embeddings, another embedding called an extra class/learnable embedding is also provided. It is a token passed to every sequence of patch embedding as a 'special token', representing no actual meaning, which is later passed throughout several layers of encoding. After the training phase, it represents the meaning of the whole sequence, thus enforcing the model to catch the general representation and allowing accurate predictions to be produced.

Additionally, these embeddings are summed with learned positional embeddings. As with the sequences fed into the original transformer, the order and position of image patches have a huge impact on how the initial image will be understood. These embeddings are learned during the initial training or fine-tuning phases, where for each patch, the highest cosine similarity is with patches that are placed in their neighborhood. The embeddings prepared in this way are passed to the encoder component of the transformer architecture.

2.6. Zero-shot Learning

The developed web application, described in the following chapters, has the ability to recognize new objects without preparing for them in advance, thus having a good zero-shot performance. Zero-shot learning is an approach to machine learning problems in which a model developed for one task can be used as a solution for another task. Research in this area has been going on for more than a dozen years and has been described in several papers: (12), (13), (25). Its meaning is often reduced to Zero-Shot Classification, where a pre-trained model is able to recognize and classify classes that were not part of the training data. This is achieved by the models' ability to learn from prior knowledge, relationships between classes, or their embeddings, and generate new classes based on this information. This type of approach is valuable in situations where there is not enough training data (and it is too expensive to create) or where the data is highly unbalanced.

2.7. Model Evaluation

Model evaluation is a fundamental step in building an effective predictive model and accessing its capabilities. It is important to analyze the strengths and weaknesses of the algorithm during the training stage, but even after its deployment, using various evaluation metrics. The way of understanding the reliability and assessing prediction quality varies depending on the use case (24). The rest of the paper describes natural language generation systems, a task that is much more complex to evaluate than simple label classification. For this reason, new metrics were created, ROUGE, BLUE and METEOR, that can examine the effectiveness of the model, while taking into account the ambiguity of the correct results.

2.7.1. ROUGE

Recall Oriented Understudy for Gisting Evaluation is a group of metrics that compare an automatically produced summarization, or a translation outcome, against a set of high-quality reference sentences in natural language processing. The package consists of a few slightly different case-insensitive metrics, all based on recall:

1. ROUGE-N

   The measure focuses on the number of matching n-grams between given sentences (human-generated reference and models outcome), where n-grams describe a group of consecutive words of size n. Independently, when n is taken into account, the ROUGE calculates the value of 3 other metrics:

   - Recall = $\frac{\text{number of n-grams found in model and reference}}{\text{number of } n\text{-grams in reference}}$

     This way of evaluation can be successful when the goal is to capture all the information from the example sentence, but it does not protect against too many useless words in the output.

   - Precision = $\frac{\text{number of n-grams found in model and reference}}{\text{number of } n\text{-grams in model}}$

     The metric that ensures not too many words are generated.

   - F1-score = $2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

     Thus, it gives the highest evaluation score for output that captures most information from the reference sequence but does not generate 'trash' words.

2. ROUGE-L

   It focuses on the longest common subsequence of two given sentences, assigning a higher similarity to those in which this subsequence was the longest. Both of the metrics from ROUGE-N, recall and precision, are used here as well, but with a minor modification - matching n-grams are replaced with the longest common subsequence length.

3. ROUGE-S

   Referred to as the skip-gram convergence metric, it searches for words from the reference text in the text returned by the model, allowing for words that are not consecutive.

This evaluation indicator is known for being easy to understand and implement, as well as for its uncomplicated calculation, which is significant in solving high-cost problems. However, it would not be a good choice when one cares about analyzing the sentiment and relationships between sentences, as well as the quality of the text. More details can be found in the original paper (6).

2.7.2. BLEU

The BLEU metric, or Bilingual Evaluation Understudy, was introduced in 2002 (22) and is widely used when it comes to evaluating natural language processing. It measures the overlap of n-grams between generated text and reference sentences, but only focuses on literal word overlap, missing contextual nuances of language, and relying heavily on the quality of the text. Its simplicity comes with drawbacks, some of which are resolved in the metric described next.

2.7.3. METEOR

The metric for Evaluation for Translation with Explicit Ordering is a type of model evaluation created to overcome the limitations of BLEU and access the quality of the generated text with a higher correlation to human judgment, introduced in 2021 (1). It not only considers exact word matches but also incorporates stemming and synonyms into the evaluation, by uniquely balancing the precision and recall with a penalty for word order.

METEOR is based on the calculation of Precision and Recall, using the fraction that common 1-grams (unigrams) in the model and reference text have in relation to their number appearing in the reference and model outcome, respectively. After that, the harmonic mean is taken to give higher importance to the value of recall, using the following formula:

$$\text{F-mean} = 10 \times \frac{\text{Precision} \times \text{Recall}}{\text{Recall} + 9\text{Precision}}$$

Next is the calculation of the penalty for word order:

$$\text{Penalty} = 0.5 \times \left(\frac{\text{number of words from the model in the same order as in the reference}}{\text{number of unigrams found in model and reference}}\right)^3$$

The final METEOR score is computed as follows:

$$\text{Score} = (1 - \text{Penalty}) \times \text{F-mean}$$

By selecting this metric to evaluate the model's predictive performance, one can expect a better text comparison than with ROUGE or BLEU. Using a harmonic average with precision and recall (with a higher emphasis on recall) creates a good balance of these results, so it is certain that all the words of the example text are used. It takes into account the match between words, focusing on paraphrase matches, thereby creating a result that is sensitive to linguistic consistency, so it can be used to translate many languages with their own rules. However, its calculations are more complex and expensive, so it may not be the best choice for voluminous data.

To accurately assess the performance of the results generated by the model, we used the ROUGE-1 F1-Score and METEOR metrics to have a broader view of the exact sentence match, as well as to capture conceptual differences.

In summary, this chapter provides a comprehensive overview of the basic concepts of deep machine learning and natural language processing (NLP), along with a delving into the complexities of neural networks, transforms and the evolving field of vision transforms. Complementing methodologies for model evaluation and assessment, these concepts collectively establish a robust theoretical foundation in multimodal architectures.

Having understood the theoretical background, it is possible to move on to the practical application of these concepts in the development of a web application with a Vision-Language Models (VLM). In the subsequent chapters, this methodology will be applied to the implementation and evaluation of a system that aims to bridge the gap between theory and real-world application.

3. Related work

Moving from theoretical foundations to real-world applications, this chapter conducts a thorough exploration of related work, providing an in-depth look at outstanding vision and language processing solutions. The section focuses on highlighting the architectures and key approaches used in the CLIP and BLIP models. They offer a distinctive approach to jointly processing hitherto disparate forms of input data, and thanks to this we have found them suitable for integration into our application.

3.1. Contrastive Language-Image Pre-training (CLIP)

The Contrastive Language-Image Pre-Training Model, often referred to as CLIP, was introduced in early 2021 by representatives of the company OpenAI (23). The purpose of this neural network is to understand texts and images collectively, which until then had been treated as separate information. To achieve this, a latent space is created in which it projects both representations with identical dimensions, in such a way that similar ones are in proximity. This enables a contrastive learning technique (30) in which the training phase focuses on increasing the similarity of correct image-text pairs and decreasing the similarity of incorrect ones. It was trained on various text-image pairs, so that can be applied to natural language to predict the most appropriate text fragment, given the image, without directly optimizing for the task.

3.1.1. Problem Statement

While there have been tremendous developments in deep learning and Vision-Language Models, it was noted that new solutions replicated the difficulty of bypassing problems. CLIP implementation addresses these problems and aims to solve them.
The most modern computer-vision implementations of the time were trained to predict a fixed set of predetermined object categories. This approach used unimaginatively expensive datasets to be created manually by humans, with limited variety and scalability. By this means, model prediction of another new concept required additional data and more manual work.

It seemed far more effective to teach the model based on natural language, which has developed significantly in recent years. NPL utilizes text sets on an Internet scale, with the computer vision of the time being based on manually labelled crowdsourced data.
In addition, standard vision models were implemented in such a way that they performed well at only one image recognition task, for which they were designed. Adapting such a model involves a great deal of cost and commitment, often failing to achieve the expected results.

The paper (23) points out the importance of supervised natural language, as there is a lot of publicly available data in this format on the internet. Many solutions, up to that point in time, were relying fully on existing datasets, which didn't utilize the full potential of research. To address this limitation, a new dataset of 400 million pairs (image, text), selected from various online sources, was created. In order to cover a wide range of visual concepts, the construction process involved identifying (image, text) pairs based on a set of 500,000 queries, and then a basic list was created from words occurring at least 100 times in the English Wikipedia. This way, a new dataset has been created, called WIT (WebImageText), which is strongly comparable to the magnitude of the dataset used to train GPT-2. This collection design allows for a broad representation of visual concepts and reinforces the importance of natural language supervision.

3.1.2. Solution

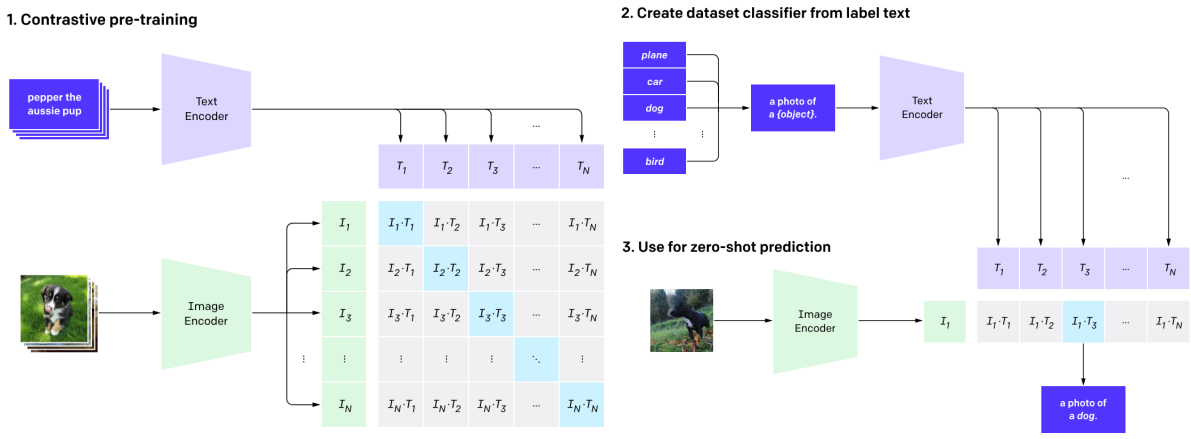

Figure 3.1: CLIP's architecture. Sourced from the original paper (23).

Providing a group of N pairs (image, text) as input, CLIP aims to predict which $N \times N$ possible pairs are actually true (which occurred in the training dataset). This is possible thanks to its multimodal characterization of the space, in which it places the embedded image and text together. It then computes the cosine similarity using symmetric cross-entropy loss, maximizes it for those that are real pairs in the batch, and minimizes those $N^2 - N$ incorrect embeddings.

The main idea of how CLIP works can be explained by the pseudocode:

```
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter
# feature representations of image and text
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T) #[n, d_t]
# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)
# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)
# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
```

When training and selecting the best models, two different architectures performing the role of image encoder were considered. The first is ResNet-50, which is known for its adaptability and proven effectiveness, presented in 2015 (7). The second architecture that gave good results and which was also considered by us in the following part of the work is the Vision Transformer, later referred to as ViT (5). This model has been plotted with only minor adjustments, such as the inclusion of an additional normalization layer for the combined embedding of patches and positions in front of the transformer, and the use of a slightly revised initialization scheme.

As a text encoder, the authors used the Transformer (29) with a self-attention mechanism, a method that allows processes to be initialized using a pre-trained language model. Thanks to this solution, not only did the model have a good starting point, but it was also able to capture dependencies and relationships within the input sequences.

This model was trained from scratch, employing a linear projection to map the representation of each encoder to a multimodal embedding space. During model training, a random square frame from variable-size images was also used along with a temperature parameter to control the range of logits as a function of softmax activation. Adam optimization with regularized decoupled weight decay and decay the learning rate using a cosine schedule (described in (17) and (18)) was also applied. Ultimately, five models based on the ResNet architecture and three utilizing ViT were released, all of which were trained at a duration of 32 epochs.

3.1.3. Results

A new model for multimodal artificial intelligence, CLIP, has achieved state-of-the-art performance and proven effectiveness in natural language processing and computer vision understanding.

The foundation for CLIP's success lies in the zero-shot transfer learning feature (2). This approach enables the model to accurately predict new concepts or classes, not seen in the training phase, hence ranking highest in tasks such as image classification or object detection. CLIP is also characterized by its universality, being able to adapt to the problem without the need for special training. Its performance in zero-shot transfer was compared to Visual N-Grams, an approach known for its abilities to leverage text for an image, over three datasets:

	aYahoo	ImageNet	SUN
Visual N-Grams	72.4	11.5	23.0
CLIP	**98.4**	**76.2**	**58.5**

Figure 3.2: Zero-shot performance of CLIP in comparison with Visual N-grams algorithm. Sourced from the original paper (23).

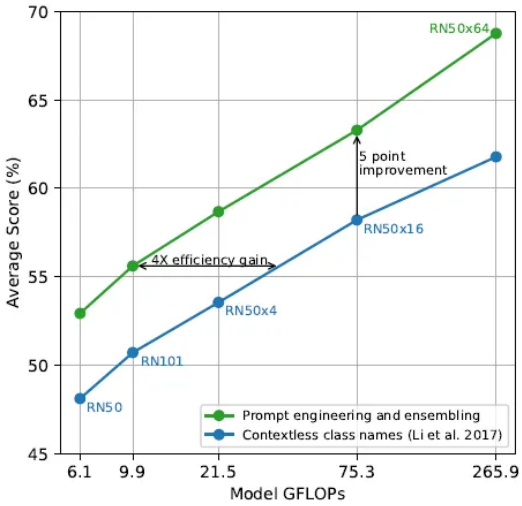

Figure 3.3: Prompt engineering and ensembling - Improvement of zero-shot performance by almost 5 points on average across 36 datasets. Sourced from the original paper (23).

The next important feature that was implemented to reach satisfactory results was prompt engineering and ensembling the data. The authors used a prompt template as 'A photo of a <label>.', to help the model specify the text regarding the image content, thus reducing the gap between a single word label and a sentence describing the image. In addition, applying prompt ensembling constructed in the embedding space, using the concepts of 'A photo of a big

<label>' and 'A photo of a small <label>' resulted in an even better understanding of visual concepts.

To get an adequate comparison of CLIP with a comprehensive set of existing models, 66 models were trained on 27 different datasets. Fine-tuned algorithms are difficult to evaluate accurately and computationally expensive to compare a diverse set of techniques, so the authors chose to compare linear classifiers that require minimal hyper-parameter tuning with conventional evaluation procedures.

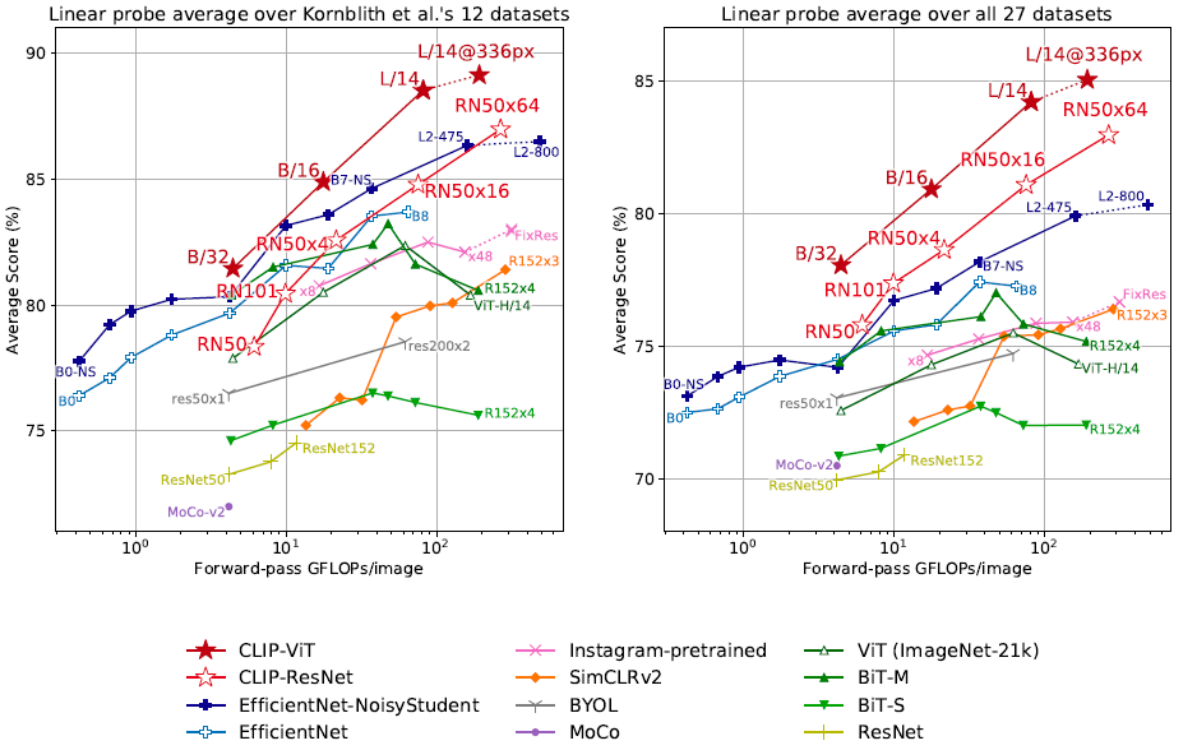

Figure 3.4: Linear probe performance in comparison with state-of-the-art computer vision models. Sourced from the original paper (23).

These results quantitatively show that the best model from the CLIP family is ViT-L/14, which outperforms the best existing models by an average of 2.6%. For this very reason, this model was chosen for inclusion in our application.

3.2. Bootstrapping Language-Image Pre-training (BLIP)

The Bootstrapping Language-Image Pre-Training Model, known as BLIP, was introduced in 2022 as a new innovative Vision-Language Pre-Training (VLP) framework (14). It stands out due to its versatile applicability, addressing both vision-language understanding and generation tasks. This is a notable distinction, as earlier models were typically designed for only one of these tasks. BLIP excels in various vision-language tasks, showcasing its effectiveness through consistently strong results. What is more, BLIP stands out by leveraging noisy web data through a bootstrapping approach. This involves generating synthetic captions and effectively filtering out noise, leading to enhanced performance.

3.2.1. Problem Statement

Visual-linguistic pre-training (VLP) models have demonstrated remarkable success in several multimodal tasks involving both textual and visual elements. However, the dominant solutions struggled with two limitations.

The majority of previous solutions relied on encoder or encoder-decoder methods. These methods however do not support approaches that seek adaptability across tasks. The encoder model is not suitable for text generation tasks, while the encoder-decoder method model is not suitable for tasks such as image-text search.

An important issue in existing approaches is the datasets used to pre-train the models. Due to the high costs associated with the creation of image-text datasets described by human annotators, recent solutions (15) have turned to Internet-derived datasets automatically collected from websites. However, such a solution often results in datasets containing inaccurately labelled data, making these image-text pairs suboptimal for training models.

To address these limitations, additional contributions were proposed within the framework of the new solution. The model architecture introduced is named Multimodal Mixture of Encoder-Decoder, referred to as MED. It can function as a unimodal encoder, an image-grounded text encoder, or an image-grounded text decoder. Furthermore, a new bootstrapping method for learning from a noisy image-text dataset, named Captioning and Filtering, referred to as CapFilt, was introduced.

3.2.2. Solution

The primary principles of the new solution involve the introduction of the MED architecture and the integration of CapFilt for dataset bootstrapping.

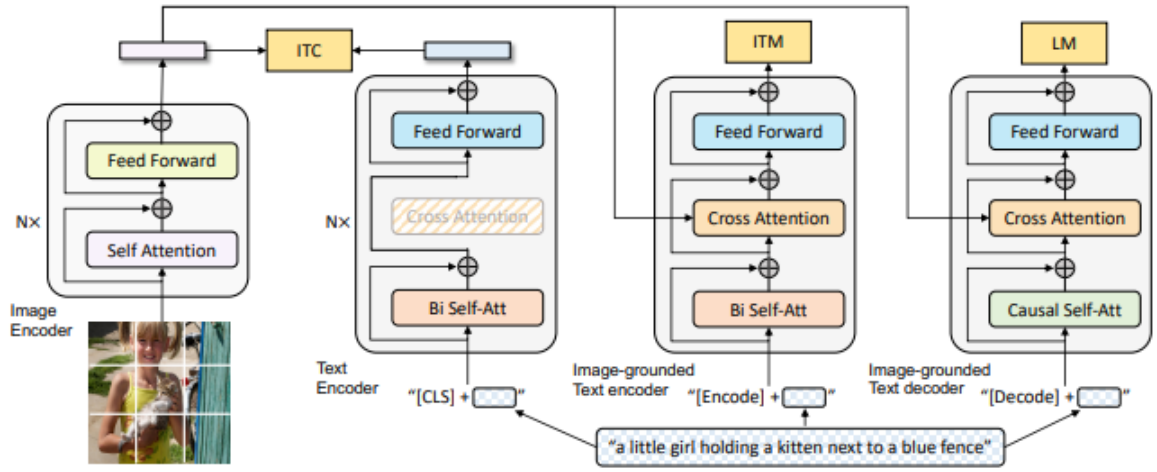

Figure 3.5: Pre-training model architecture and objectives of BLIP. Sourced from the original paper (14).

In the proposed solution, image encoding is accomplished using a Vision Transformer (5). This transformer processes images by dividing them into patches and encoding them as a sequence of embeddings, with an additional [CLS] token to represent the global image feature.

The architecture of the multi-task model can operate with the three functionalities delineated below.

Unimodal Encoder performs the processing of images and text separately. In the operation of the text encoder, a [CLS] token is added at the beginning of the inputs to summarize the sentence.

Image-grounded text encoder introduces visual information through an additional cross-attention (CA) layer. An integrated approach is employed by appending a task-specific [Encode] token to the text, with the resulting output embedding [Encode] functioning as a compelling multimodal representation that effectively processes both text and image information within the context of the task.

In **Image-grounded text decoder**, the bidirectional self-attention layers are replaced with causal self-attention layers when compared to an image-based text encoder. A [Decode] token is used to signal the beginning of a sequence, and an end-of-sequence token is used to signal its end(14).

In the integrated model, every image-text pair undergoes processing through a visual transformer and one of three text transformers. During the pre-training phase, the model architecture incorporates three distinct loss functions, each tailored to optimize specific tasks. The purpose of Image-Text Contrastive Loss (ITC) is to strengthen positive image-text pairs by encouraging similarity in their representations and distinguishing them from negative pairs. The Image-Text

Matching (ITM) Loss trains a binary classifier to predict if an image-text pair is positive or negative. In the paper, researchers used a hard negative mining strategy introduced in (15). Lastly, Language Modeling Loss (LM) loss activates the image-text decoder, optimizing cross-entropy and providing the generalization capability to convert visual information into coherent captions.

To eliminate noisy data in the solution, the Captioning and Filtering method was introduced (3.6). This method consists of two modules, both based on a pre-trained MED model that has been fine-tuned using the COCO dataset (16).

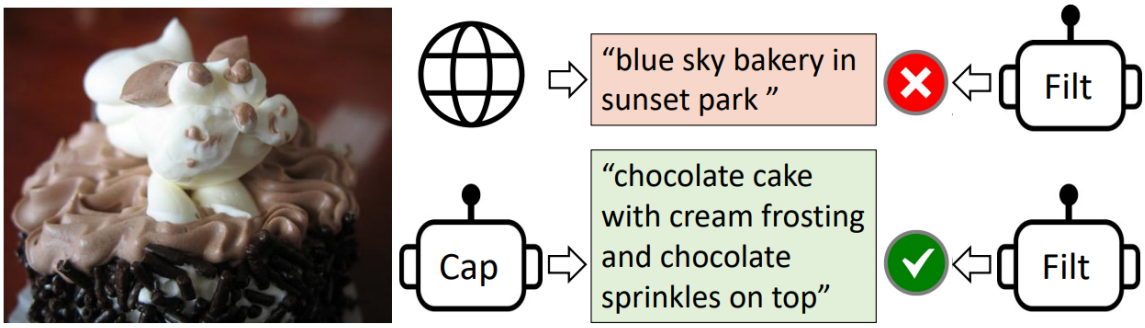

Figure 3.6: Captioning and Filtering method. Sourced from the original paper (14).

The captioner module is a text-decoder, producing synthetic captions for web images, while the filter module, trained with ITC and ITM objectives, evaluates text-image alignment, removing noisy texts. This process creates a new dataset by combining image-text pairs generated by the CapFilt with human-annotated data.

3.2.3. Results

BLIP effectively leverages noisy web data through a bootstrapping mechanism involving the generation of synthetic captions by a captioner and the subsequent removal of noisy captions by a filter. The approach achieves state-of-the-art results across a broad spectrum of vision and language tasks.

Figure 3.7: Web descriptions vs. synthetically generated captions. Sourced from the original paper (14).

Data created using the new method represents notably higher quality (3.7). The database is

enriched with high-quality titles generated by the Captioner. Additionally, inconsistent captions, whether directly sourced from the internet or synthetically generated, are effectively filtered out, contributing to an overall improvement in data quality (14).

BLIP, compared to existing language-visual pre-training (VLP) methods, is achieving very impressive results on a wide range of vision-language downstream tasks. It excels in both image-to-text and text-to-image retrieval on the COCO and the Flickr30K datasets. Notably, with 14 million pre-training images, BLIP outperforms the previous best model ALBEF (10) by +2.7% in average recall@1 on COCO using the same amount of images.

In the image captioning task, BLIP with 14M pre-training images substantially outperforms methods using a similar amount of pre-training data. BLIP with 129M images achieve competitive performance as LEMON with 200 million images, highlighting its efficiency. In the context of Visual Question Answering (VQA), BLIP adopts an answer generation approach, surpassing ALBEF by +1.64% on the test set using 14 million images. Using 129 million images, BLIP outperforms SimVLM, which uses 13 times more pre-training data and a larger vision backbone.

Expanding its prowess, BLIP achieves excellent results in Natural Language Visual Reasoning (NLVR2), introducing computational-efficient modifications for reasoning over two images. Notably, it outperforms existing methods on NLVR2, excluding ALBEF. In Zero-shot Transfer to Video-Language Tasks, BLIP's strong generalization to video-language tasks is seen. It outperforms models in text-to-video retrieval by +12.4% in recall@1, surpassing even models finetuned on the target video dataset.

Results from the CapFilt Ablation Study confirm that CapFilt's effectiveness is not caused just by longer training. It is crucial to train a new model on the bootstrapped dataset, while continuing training from a previous model does not enhance performance (14).

3.3. Models in our Solution

Vision-Language Models such as CLIP and BLIP achieve excellent results, making them very useful for a variety of applications. In our web application, CLIP serves as a robust backbone for semantic image search based on a given phrase, offering an innovative multimodal reasoning approach for textual and visual data. This integration enables the transformation of user queries into meaningful representations, facilitating precise image alignment. Additionally, we have also incorporated BLIP into the image captioning task. An image-grounded text decoder powered by BLIP is used to convert visual information into consistent captions. This feature enriches the user experience by enabling the creation of descriptive text for interpreted images.

4. Dataset used for application testing

In our application, we utilize the Flickr 30k dataset, leveraging its diverse collection of images. This dataset serves a dual purpose, as it is not only used for comprehensive testing of our application but is also seamlessly integrated, so users can take advantage of semantic search. By integrating images from this dataset into the application, users can easily analyze and evaluate the effectiveness of the model on a dataset that is renowned for its richness and diversity. Let us now describe this dataset in more detail.

4.1. Dataset Introduction

Flickr30k was introduced in the paper "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions" (8). In this paper, a fresh perspective on measuring semantic similarity is presented. The method involves using images and their descriptions to construct a denotation graph, which can be beneficial in tasks requiring semantic inference. Semantic inference is a process associated with deriving new data without explicitly adding new data. Instead, new data is created from existing data, utilizing established rulesets and ontologies to infer new facts about existing data.

4.2. Content of the Dataset

The dataset comprises 31,783 images sourced from Flickr, accompanied by 158,915 captions for the provided photos contributed by human annotators. This results in an average of five reference sentences for each image. Flickr is a web service created to collect and share photos online.

In the dataset, its creators focused on collecting photos that target people and everyday human activities, i.e., running, mountain climbing, cycling, swimming, riding public transportation, and many others. The dataset also includes a lot of photos of dogs, which we believe is intentional due to the strong association of dogs with human activities. What is interesting

is that the collected photos are captured in a variety of landscapes and weather conditions. The photos feature scenes from mountains, seasides, homes, and urban environments, and cover various seasons.

Gray haired man in black suit and yellow tie working in a financial environment.
A graying man in a suit is perplexed at a business meeting.
A businessman in a yellow tie gives a frustrated look.
A man in a yellow tie is rubbing the back of his neck.
A man with a yellow tie looks concerned.

A butcher cutting an animal to sell.
A green-shirted man with a butcher's apron uses a knife to carve out the hanging carcass of a cow.
A man at work, butchering a cow.
A man in a green t-shirt and long tan apron hacks apart the carcass of a cow while another man hoses away the blood.
Two men work in a butcher shop; one cuts the meat from a butchered cow, while the other hoses the floor.

Figure 4.1: Sample data in the Flickr dataset. Sourced from the original paper (8).

Image captions created by human annotators feature greater detail compared to descriptions generated by models. Annotators focus on the essential elements of the image, carefully considering key details. Their descriptions not only cover the general situation depicted in the image but also highlight the most important features, capturing the essence of the image in a more precise and complete manner.

4.3. Dataset Utilization in various Benchmarks

The dataset has become one of the most popular tools for vision-language tasks. It is used in a variety of benchmarks, such as Zero-Shot Cross-Modal Retrieval, Image Retrieval, Image-to-Text Retrieval, Node Classification, Image Captioning and Phrase Grounding.

Due to the precise descriptions accompanying the images, the dataset has become an efficient evaluation tool. It allows testing the effectiveness of models and measuring various metrics, emphasizing its significant role in advancing research and development in these fields.

5. Introduction to the developed application

In response to continuous technological development, where the quantity of stored photographs is constantly increasing, we have observed a demand for applications that enable the management of photo collections. Our application was created to support users in utilizing advanced models for image-related tasks.

5.1. Motivation

The creation of our application was motivated by the goal of creating a simple tool for users to manage photos on their devices. Often, people have disordered photos that they have not had the opportunity to use. Sometimes people do not even know what photos they have in their gallery. Our application allows the user to search by phrase and save photos with model-generated titles. With a possible future improvement, the user would be able to sort the photos in their gallery into specific folders by typing a few key phrases. With this capability, the gallery would be organized by the user's most important keywords, allowing quick access to specific and unique photos in the future.

This application is intended to assist individuals in their daily activities. It should enable the automation of tasks, thereby saving users the time they would otherwise spend on manual efforts. The application streamlines searches by organizing photos into smaller, more focused groups. For example, if a user intends to create a collage of their dog's photos, they could specifically target a reduced set of images instead of manually sorting through their entire photo gallery and selecting from all the pictures. This simplifies the process and minimizes the time and effort required.

Applications should be designed with the ability to familiarize users with the capabilities of Vision-Language Models for machine learning. Through the application, people have the opportunity to explore the functionality of models that recognize objects in pictures and to check how precisely some of the best machine learning models currently operate.

An additional goal of our application is to create a tool that enables ordinary users to support

the development of NLP models. With our application, individuals are given the opportunity to assess the accuracy of existing models and actively contribute to the ongoing progress of artificial intelligence by providing valuable feedback. The feedback mechanism provides valuable information on images that the models are currently unable to recognize sufficiently, indicating areas that need further improvement. At the same time, this method provides the opportunity to actively participate in the advanced stages of refining the model by incorporating additional training data.

5.2. Functionality Scope

The main elements of the application include a user-friendly interface that provides the user with easy access to the necessary functionalities. Our goal is to create an intuitive application. Users should be able to revisit photos they have previously submitted to us without the need to resend images with which our application has already interacted with.

In our application, we intend to ensure that users experience a distinct separation between the tasks they select. This is achieved by implementing two separate tabs for each functionality. This design decision is meant to offer users a straightforward and intuitive method to differentiate between different features. By taking this approach, our goal is to improve the overall navigation and usability of the application.

The functionality we aimed to cover in our application is the ability to search for photos based on specific phrases entered by the user, and to allow users to generate titles for photos they upload.

6. Application implementation

The implemented application was developed using Python and JavaScript languages, relying on popular and easily accessible technologies such as Django REST and React. The application serves as a comprehensive tool that includes functionalities such as semantic image search and image captioning. It provides additional capabilities for logged-in users, allowing management of previously uploaded images. The easy-to-use application enables the use of advanced multimodal Vision-Language Models through an intuitive user interface.

6.1. Technical details

Our application has been built using widely adopted and popular technologies in the developer community. To build the REST API, the backend for our application, we utilized a free and open-source framework called Django REST. This framework, developed in the Python language, supports developers by providing ready-to-use, built-in components, facilitating faster and more efficient web development. By leveraging Django REST, our application offers the following key features:

- **Admin Panel:** Django has a built-in model-centric interface that allows the administrator and authorized users to manage the data contained on the site
- **User authentication:** Django includes a comprehensive user authentication system that manages user accounts, groups and permissions. The framework allows for easy extension and customization of the features to fit project requirements. Django's authentication system supports both user authentication and authorization to enable certain actions on the site or access to certain data

Django follows the MVT (Model-View-Template) design pattern, involving three fundamental components. The first of these components is the model, which serves as a source of information about data and encompasses the necessary fields and behaviors of the stored data. Each model corresponds to a single table in the database. In our application, models play a crucial role

in defining the structure of the database and managing data related to image classification, semantic image search, user choices, and user authentication.

Another Django's component extensively used in our application is the View, responsible for managing the flow of data between the application and the user, determining how the website should respond to the user's request. In our application, views facilitate the uploading of data to the model and obtaining model results. They enable the addition of new user information to the database and provide access to photos saved in the user's catalog, as well as images from Flickr.

The user interface was created using React. It has been the most popular front-end framework for years, known for its effectiveness in building applications. The integration of diverse React components, including buttons, text fields, grids and models, has enabled the creation of an aesthetically pleasing and intuitively user-friendly interface.

In the application, we also utilized Node.js, which is a cross-platform, open-source JavaScript runtime environment that serves as a popular tool for developers. It was employed in our application to run the front-end.

6.2. Login and Registration

One of the developed features is registration, which enables users to save the photos they have previously entered into the application. During registration, if the user fails to meet the password length requirements, an information message about an unsuccessful register appears.

We used the BaseUserManager and the AbstractUser models, expanding and customizing them to create fields such as a list of photos belonging to the user. Using the corresponding views created, we have the ability to create and log in a user in a dedicated tab. After logging in, an authorization token appears on the page, allowing us to access log-in-only functionalities, such as access to directories containing previously uploaded photos. This enables the user to engage in photo search and title generation not only with images directly uploaded to the model but also with those previously utilized within our application.

6.3. Semantic Image Search Task

In our application, The Semantic Image Search feature enables users to perform advanced searches using semantic features. This feature enables efficient searches for specific images. Upon

entering this tab, users encounter three distinct options. Users have the ability to upload photos directly from their devices, search among previously uploaded photos from a catalog (which is only possible for logged-in users), or explore external catalog images through a dataset search from Flickr (8).

After selecting one of the search options, users can enter a specific phrase to specify their search. Upon entering the phrase and pressing Enter, the system uses the CLIP model (23) to provide relevant results based on the semantic content of the images.

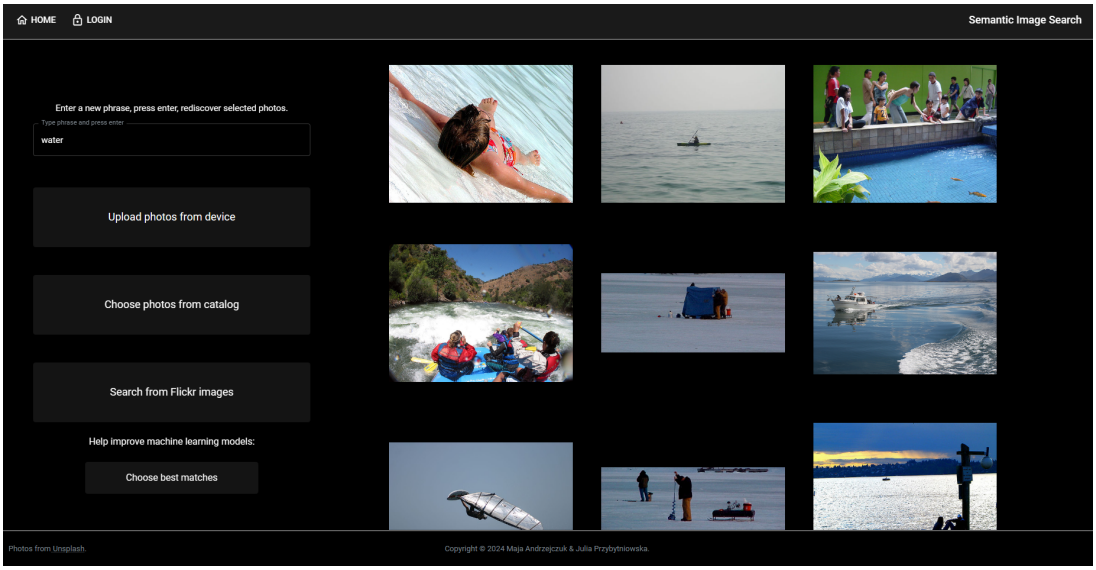

Figure 6.1: Example of generated output in the Semantic Image Search View

The system uses a set of probabilities for image-phrase pairs, determining how closely an image aligns with a given phrase. The application then sends the user images that it considers most similar according to the model. If the user wishes to find an image that better represents their desired concept, they can enter a different phrase. Subsequent searches are expedited as the application stores previously processed images in a numerical list format, optimizing search speed.

In some cases, there might be a situation where the model fails to find a matching image for the entered phrase. This could occur due to a probability threshold set to avoid displaying images that are in no way related to the user's search. In such instances, the user will be prompted to enter a different phrase. This approach ensures that only relevant images are presented, enhancing the overall search experience.

6.4. Image Captioning Task

The Image Captioning Tab provides a caption generation feature. The task involves describing the analyzed content of an image with a generated textual description. The system uses a BLIP model (14) for the task of Image Captioning. This model comprises a vision encoder responsible for processing the input image and a text decoder that generates textual output based on the encoded visual information.

The tab provides two options for selecting a photo for which a title is to be generated. Users have the ability to upload photos directly from their devices or choose catalog selection, which involves selecting a photo from the catalog of previously uploaded images (which is only possible for logged-in users).

After selecting an image, users can generate a caption by clicking on the *Generate Caption* button. Once clicked, the image goes through a transformation process by an encoder, which divides the images into patches and encodes them into a sequential series of embeddings. The model then employs an image-text decoder to transform the visual information into corresponding textual descriptions for the image. The system provides users with various caption options based on the content of the image.

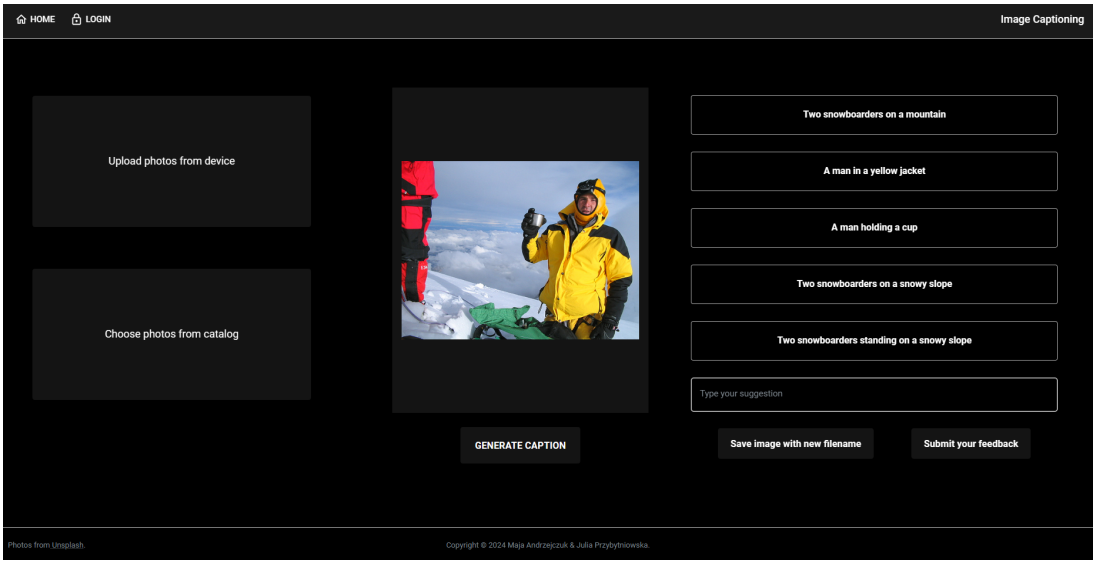

Figure 6.2: Example of generated output in the Image Captioning View

Users can choose a caption that best fits the image, or if none of the suggestions suit their preference, they can enter a custom caption. If the user wishes to do so, they have the option to save the photo with the newly created title to their device.

6.5. Saving feedback

In the application, one of the key functionalities is the implementation of a feature that allows users to provide feedback. Saving feedback aims to enable users to contribute to the development of machine learning models. This process provides us with significant opportunities for fine-tuning future models, translating into continuous improvement of our solutions.

In the app, we enabled users to leave feedback in both tabs. In the Image Captioning Task, users are presented with two options. They can contribute to the evaluation of the working model and mark the sentences that they think are generated correctly. After the selection of model suggestions, pairs of images and captions that best match users' preferences from the available options are saved in the database. The user also has the option to enter their title, which they believe best suits the selected photo. This way, the database is enriched with titles generated by humans, serving as valuable data.

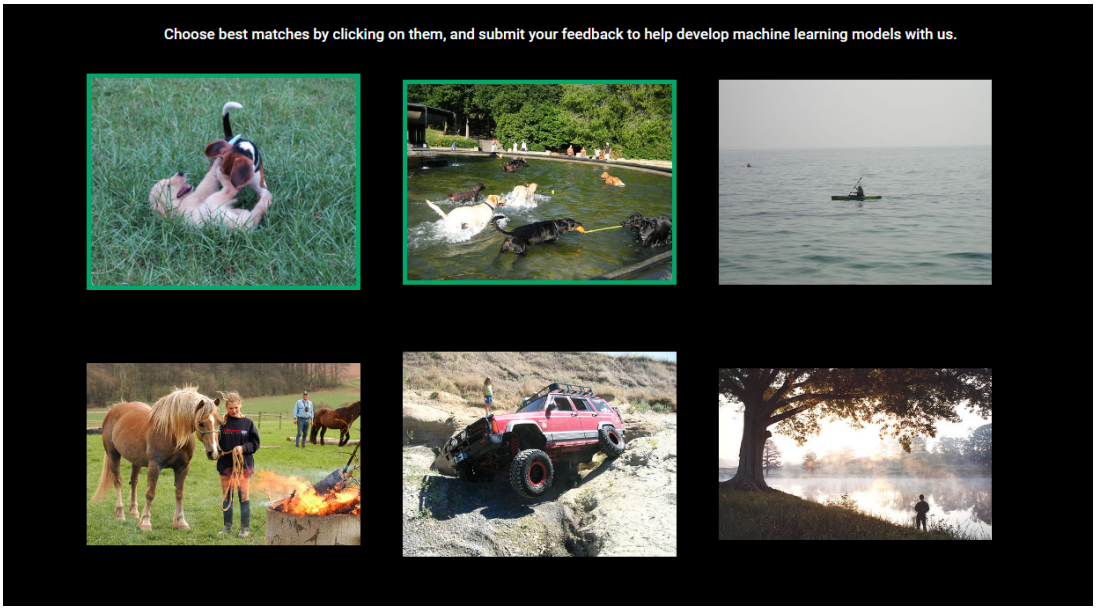

Figure 6.3: Saving feedback in Semantic User Search - Choosing the most fitting images based on user-entered phrases or keywords in the search.

The second feedback option is presented in the tab related to Semantic Image Search (6.3). After entering a phrase, users have the option to choose the photos they believe best match the entered phrase and submit image-caption pairs to the database.

7. Experimental results

Our application is designed to familiarize users with the capabilities of Vision-Language Models for machine learning. Through the application, people have the opportunity to explore the functionality of models that recognize objects in pictures and to check how precisely some of the best machine learning models currently operate.

In the image generation tests, we employed the METEOR metrics. We randomly selected a sample of 500 images from the Flickr dataset and calculated the metric. In our dataset, we have an average of 5 reference sentences per image. Therefore, the computed metric was averaged across all reference sentences. The objective was to explore the dataset by specifically choosing pairs of image-text with the lowest and highest scores. This enabled us to analyze the types of images for which our model encounters challenges in generating titles, as well as those it handles exceptionally well.

Generated Title: a cloudy sky
Reference Title: Three people are facing the mountains

Generated Title: a water is blue
Reference Title: A guy catches a wave on his surfboard

Generated Title: the sky is blue
Reference Title: Two dogs racing across the water toward a beach

Figure 7.1: Illustration of generated titles for pairs with lower METEOR scores. Images are sourced from the dataset Flickr30 (8).

In situations where the context is complex or contains specific elements, the model appears to encounter difficulties, leading to incorrectly generated descriptions. Examples of such cases include image descriptions where the model does not accurately capture the essence of the scene (7.1). For instance, in the case of a photograph depicting three standing people looking at towering mountains, the description "a cloudy sky" may be accurate but lacks all the information we desire.

Another example arises when the model generates the description "the water is blue" for an image representing a surfer catching a wave. This suggests that the model may struggle to account for details related to water activities, which could create inaccuracies in the generated captions.

Similarly, when the original description refers to two dogs chasing each other through the water toward the beach, the title "the sky is blue" indicates difficulties in the model accurately reproducing dynamic elements of the scene.

These instances of incorrectly generated descriptions suggest that the model may require additional adjustments, particularly in handling diverse contexts. Corrections may be necessary to enhance the precision of analyzing human activities that do not occupy a significant portion of the image but are crucial from a human observer's perspective.

Generated Title: a young girl running down a long hallway
Reference Title: A little girl in pyjamas runs down a hall with hardwood floors

Generated Title: a dog running in the snow
Reference Title: A blond lab runs in the snow

Generated Title: a man and woman sitting on a bench
Reference Title: A man and a woman on a bench

Figure 7.2: Illustration of generated titles for pairs with higher METEOR scores. Images are sourced from the dataset Flickr30 (8).

With correctly generated descriptions, the model appears to effectively handle certain scenarios, especially those that are more straightforward. The accurately generated titles displayed in the image (7.2) suggest that the model can effectively handle photos depicting specific situations, especially those with a clearer context. In these photos, the dynamic elements, crucial for human understanding, are prominently featured in the foreground, occupying a significant portion of the photo. This allows the model to focus specifically on these key aspects, contributing to its

effectiveness in accurately capturing and describing the scenes.

Generated Title: a man wearing red jacket
Reference Title: A father and son looking at a funny-looking Santa

Figure 7.3: Illustration of Generated Title for Christmas-Related photo. The image is sourced from the dataset Flickr30 (8).

A fascinating example from the dataset is an image depicting the character of Santa Claus (7.3). In an attempt to describe this unique image, the model generated the caption, "A man wearing a red jacket". This serves as a perfect example to highlight certain imperfections in models when it comes to recognizing events that hold significance for people, especially those associated with holidays.

We can observe how the model, although successfully identifying a person in a red jacket, failed to accurately capture the context related to the figure of Santa Claus. This phenomenon underscores the challenges that artificial intelligence faces in fully understanding the subtleties and essence of specific events, such as holidays, which hold unique emotional and cultural significance for humans.

What we have noticed interesting during title generation is that the model does not use articles much and does not discuss emotions. The description is flattened to a general type like "a man in a blue shirt pointing his finger at the camera", but it does not mention that he is angry. Similarly, a picture of a sad woman is described as "A woman with long hair in the dark". The model can say that the woman is smiling because it recognizes her smile, but it will not convey that she is happy.

In the context of analyzing the CLIP model in searching for images from textual descriptions, our observations are promising. In our experiment, by introducing the description "car on the road" into the CLIP model, we obtained diverse and adequate images (7.4). The results obtained

not only included an image of a car on the road but also a bicycle on the road and a car behind a stall.

Figure 7.4: Images found by the CLIP model based on the description "car on the road". Images are sourced from the dataset Flickr30 (8).

This suggests the flexibility of the CLIP model in interpreting the diverse contexts associated with a given description. The model appears to efficiently incorporate various aspects of textual descriptions, searching for images that most closely match a given description. Observations like these indicate the potential of the CLIP model to effectively combine text semantics with visual space, which is important for meaning-based search applications.

Figure 7.5: Images found by the CLIP model based on the phrase "snowboarding". Images are sourced from the dataset Flickr30 (8).

During another search, this time using the brief phrase 'snowboarding', numerous images related to a wintry mountain landscape emerged 7.5. The model demonstrated its capability to associate this sports activity with the surrounding snowy scenery.

A valuable insight was to test the CLIP model's ability to recognize dynamic actions among the searched images. In this case, we used the short phrase "running" (7.6). The observations were successful, as the three photos with the highest probability showed sports competitions with running, as well as two scenes with dogs running. This demonstrates that the model is not only able to identify dynamic activities but can also successfully identify them in both human and animal contexts.

Figure 7.6: Images found by the CLIP model based on the phrase "running". Images are sourced from the dataset Flickr30 (8).

Both models show high performance. The generated titles, in most cases, accurately describe the image content. An image search based on the semantic search task of the CLIP model finds images strongly related to the phrases entered. Nevertheless, both of these models have some limitations, as there are situations in which they do not work perfectly. This is often because they operate more on a technical level, focusing on general descriptions, which makes them lack the emotional depth characteristic of the human perspective. Therefore, additional datasets with annotations made by humans could be very helpful in further improving them.

8. Concluding remarks

Our engineering thesis presents a web application that capitalizes on the capabilities of advanced Vision-Language Models, particularly BLIP for image captioning and CLIP for semantic image search, all within a user-centric interface. This integration into a cohesive system, built upon a robust Django REST backend and a responsive React frontend, is a significant advancement in making state-of-the-art AI technologies accessible to a wider audience.

The core achievement of our application lies in its capacity to convert complex AI model processes into an intuitive and interactive user experience. It allows users to interact seamlessly with images, fully utilizing the potential of visual language models for semantic searches and automated captioning. This functionality not only enhances user engagement but also serves as a crucial channel for collecting feedback, which is essential for the continuous improvement and evolution of these AI systems.

An important part of the application is its ability to be extended and adapted to new needs and to enable the addition of new functionalities that contribute to its continuous development.

One of the key features we aim to create in the future is a new dataset. Through user feedback during the application usage process, we would have the opportunity to create an additional dataset consisting of image-text pairs based on phrases provided by users. The filter module, an integral part of the CapFilt system (described in Section 3.2) introduced in the BLIP model (14), is designed to eliminate noisy image-text pairs. Through the use of this module, we would be able to successfully filter out potential errors entered by users into the dataset, providing the opportunity to selectively choose only the data that meets high-quality standards. This would enable us to select only high-quality data. In this way, we would obtain a dataset containing descriptions created by humans, allowing for the fine-tuning (4) of models in areas where synthetically generated or pre-existing dataset texts may have gaps and result in inaccuracies in the current model.

Additionally, users' feedback provides analytical insights into the images for which models are currently malfunctioning and need improvement. This provides an opportunity to contribute knowledge of what datasets should be created to develop current models and improve their imperfections.

A valuable and highly beneficial improvement for our application would involve integrating one of the best machine translation tools (28) to expand the application's usage among a broader audience. People from various countries, even those unfamiliar with the English language, would have the opportunity to use the application. Users could select any language supported by the translation model.

Phrases entered by users during Semantic Search tasks would be seamlessly translated through the translation model and fed into the CLIP model in English. In the case of the Caption Generator task, after users choose a photo, the resulting title generated by the model would be passed to the translation model and presented in the output, displayed to users in the language chosen on the website.

The application has the potential to become a supportive tool for users in their daily photo management tasks. Its capabilities offer users seamless navigation of functions, allowing for efficient photo searches using specific phrases, generating image titles and seamless access to previously uploaded photos. Furthermore, the app has the potential to serve as a tool supporting the development of machine learning in fine-tuning natural language models through the utilization of a dataset derived from user feedback.

Bibliography

[1] Rohan Chandra, Xijun Wang, Mridul Mahajan, Rahul Kala, Rishitha Palugulla, Chandrababu Naidu, Alok Jain, and Dinesh Manocha. Meteor:a dense, heterogeneous, and unstructured traffic dataset with rare behaviors, 2022. arXiv:2109.07648.

[2] Zixiang Chen, Yihe Deng, Yuanzhi Li, and Quanquan Gu. Understanding transferable representation learning and zero-shot transfer in clip, 2023. arXiv:2310.00927.

[3] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction, 2019. arXiv:1801.02143.

[4] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping, 2020. arXiv:2002.06305.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. arXiv:2010.11929.

[6] Kavita Ganesan. Rouge 2.0: Updated and improved measures for evaluation of summarization tasks, 2018. arXiv:1803.01937.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. arXiv:1512.03385.

[8] Peter Young Alice Lai Micah Hodosh Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. 2014. URL: https://aclanthology.org/Q14-1006.pdf.

[9] Arnulf Jentzen, Benno Kuckuck, and Philippe von Wurstemberger. Mathematical introduction to deep learning: Methods, implementations, and theory, 2023. arXiv:2310.20360.

[10] Akhilesh Deepak Gotmare Shafiq Joty Caiming Xiong Steven Hoi Junnan Li, Ramprasaath R. Selvaraju. Align before fuse: Vision and language representation learning with momentum distillation, 2021. arXiv:2107.07651.

[11] Kian Kenyon-Dean, Edward Newell, and Jackie Chi Kit Cheung. Deconstructing word embedding algorithms, 2020. arXiv:2011.07013.

[12] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014. doi:10.1109/TPAMI.2013.140.

[13] Hugo Larochelle, Dumitru Erhan, and Y. Bengio. Zero-data learning of new tasks. volume 2, pages 646–651, 01 2008. URL: https://www.researchgate.net/publication/221606655_Zero-data_Learning_of_New_Tasks.

[14] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022. arXiv:2201.12086.

[15] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. Align before fuse: Vision and language representation learning with momentum distillation, 2021. arXiv:2107.07651.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014. arXiv:1405.0312.

[17] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017. arXiv:1608.03983.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. arXiv:1711.05101.

[19] Popescu Marius-Constantin, Balas Valentina E, Perescu-Popescu Liliana, and Mastorakis Nikos. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7):579–588, 2009. URL: http://www.wseas.us/e-library/transactions/circuits/2009/29-485.pdf.

[20] Samreen Naeem, Aqib Ali, Sania Anam, and Munawar Ahmed. An unsupervised machine learning algorithms: Comprehensive review. *IJCDS Journal*, 13:911–921, 04 2023. doi:10.12785/ijcds/130172.

[21] Vladimir Nasteski. An overview of the supervised machine learning methods. *HORIZONS.B*, 4:51–62, 12 2017. doi:10.20544/HORIZONS.B.04.1.17.P05.

[22] Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. Bleu: a method for automatic evaluation of machine translation. 10 2002. doi:10.3115/1073083.1073135.

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. arXiv:2103.00020.

[24] Sebastian Raschka. Model evaluation, model selection, and algorithm selection in machine learning, 2020. arXiv:1811.12808.

[25] Marcus Rohrbach, Michael Stark, and Bernt Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. *CVPR 2011*, pages 1641–1648, 2011. URL: https://api.semanticscholar.org/CorpusID:14700310.

[26] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020. URL: http://dx.doi.org/10.1016/j.physd.2019.132306.

[27] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014. arXiv:1409.3215.

[28] Zhixing Tan, Shuo Wang, Zonghan Yang, Gang Chen, Xuancheng Huang, Maosong Sun, and Yang Liu. Neural machine translation: A review of methods, resources, and tools, 2020. arXiv:2012.15515.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. arXiv:1706.03762.

[30] Zihu Wang, Yu Wang, Hanbin Hu, and Peng Li. Contrastive learning with consistent representations, 2023. arXiv:2302.01541.

[31] Jiawei Zhang. Gradient descent based optimization algorithms for deep learning models training, 2019. arXiv:1903.03614.

List of Figures

Figures presented in this document that lack a linked source have been generated by the authors for illustrative purposes, and do not represent external references. The figures attached in Chapter 6 (6.1, 6.2, 6.3) are screenshots of a browser window from the application, the source code of which has been published on the Github repository.