

# Image-to-Sentence Generation

Adil Zhiyenbayev

Nazarbayev University

adil.zhiyenbayev@nu.edu.kz

Rakhat Abdrakhmanov

Nazarbayev University

rakhat.abdrakhmanov@nu.edu.kz

Nurlan Kabdyshev

Nazarbayev University

nurlan.kabdyshev@nu.edu.kz

Zarema Balgabekova

Nazarbayev University

zarema.balgabekova@nu.edu.kz

## Abstract

*Image captioning is the automatic generation of descriptions that conform to human language cognition for an image. Most present image captioning models rely on deep learning and employ encoder-decoder models. In our project, we followed the same approach and used the CNN-RNN model as our baseline and ViT-RNN with self-attention mechanism configurations in our main approach. The obtained models were evaluated on the Flickr8K and Flickr30K datasets using the BLEU score. During the project, we experimented with feature extractors, model architectures, and hyperparameters and could improve our first baseline model both quantitatively in terms of BLEU score and qualitatively.*

## 1. Introduction

Image captioning, a field encompassing image-to-sentence or image-to-text generation, is important in bridging the gap between computer vision and natural language processing (NLP). This process involves leveraging computer vision techniques to comprehend the visual content of an image and employing language models from NLP to generate coherent and meaningful sentences [19].

Describing images involves visual perception, focusing on specific regions, and crafting a descriptive narrative. Image captioning aims to replicate this cognitive process. It tries to achieve this through computational steps, including feature representation, visual encoding, and language generation [24]. The integration of deep learning techniques has become prevalent, with algorithms incorporating Convolutional Neural Networks (CNNs) for feature representation, attention mechanisms or graph models for visual encoding, and Long Short-Term Memory (LSTM) or Transformers for language generation [19, 24].

While image captioning finds applications in various

domains, such as enhancing visually intelligent human-computer and human-robot interaction systems [8], aiding individuals with visual impairment [18], and contributing to computer-aided medical diagnosis [2], it faces several challenges.

Current deep image captioning methods have limitations. Firstly, there's no widely applicable framework, making it hard to find a method suitable for various images and captions. Limited access to large, fitting datasets affects the effectiveness of training and can result in inaccuracies, especially with uneven samples. Lastly, the evaluation metrics used may not fully grasp the complexities of image captioning, leaving uncertainty about how well the results align with human understanding. [24]

Furthermore, there are challenges in creating accurate and natural descriptions that match what's in the picture. Models that rely on matching images to a database of captions can struggle, especially with different scenes and objects. Another problem is ensuring the captions cover the whole picture; existing models predominantly employ local attention mechanisms that may result in unnatural and incomplete captions. Additionally, deep learning models may not work well on images very different from what they were trained on, leading to reduced performance in those cases.

Different methods have been used to solve challenges in image captioning. Models using recurrence, like combining CNN for understanding images and LSTM for generating descriptions, are popular and successful. Transformer-based models, which use global attention to understand the whole image, have gained attention. To improve generalization, some approaches focus on improving training data through techniques like data augmentation or using external knowledge. To ensure a complete visual understanding, some methods use attention to highlight important parts of the image, while others use multi-modal fusion to combine information from different sources. [11]

We tried to overcome such challenges and limitations by

experimenting with different methods for image captioning. We have tried a lot of models for encoders and decoders, tuning the hyperparameters and adding an attention-based layer to the main approach. Overall, we could significantly improve the performance of our first baseline model. In the following sections, we delve into the specifics of our approach, our experiments on the datasets, the obtained results, and the errors and challenges we encountered.

## 2. Literature Review

With the introduction of CNNs, models handling visual inputs have significantly enhanced their performance. The visual encoding phase in image captioning has seen advancements as well. In a straightforward approach, the activation of a CNN’s last layers is utilized to derive high-level representations, subsequently acting as a conditioning factor for the language model. Vinyals et al. utilized the output of GoogleNet [21] to initiate the hidden state of the language model. In the same vein, Karpathy and his team used features from the output of AlexNet [5] to feed to the language model. Another approach used a fully connected model (FC), in which ResNet-101 was used as the encoding mechanism [15]. This FC model leverages the power of convolutional layers for efficient and effective processing of image data, contributing to improved image captioning performance. However, methods based on global CNN features tend to overly compress information, lacking granularity, thus making it challenging for a captioning model to generate specific and finely detailed descriptions.

In the context of the encode-decode framework, the Recurrent Neural Network (RNN) decoder plays a crucial role in generating descriptive sequences, such as image captions. This decoder takes the previously encoded image features as its initial input and proceeds to create a sequence of words step by step. At each time step, it takes the previously generated word as an input and updates its hidden state accordingly. This updated hidden state is pivotal in predicting the probability distribution of the next word using a softmax function. The word with the highest probability becomes the next word in the sequence, and this process continues until it reaches an end-of-sentence token or a predefined maximum sequence length [7]. During the training phase, the RNN decoder is fine-tuned to minimize the discrepancy between its predicted sequence and the ground truth sequence, employing cross-entropy loss. Depending on the specific task and dataset, the RNN decoder can be realized using various types of RNNs, including LSTM or Gated Recurrent Unit (GRU), tailoring its architecture to meet the demands of the application. This mechanism allows for the generation of coherent and contextually relevant textual descriptions, making it a vital component in tasks like image captioning and language generation [9]. In our baseline model, we also used a simple encoder-decoder

framework with the CNN-based encoder and RNN in the decoding step. We experimented by using different CNN-based encoders and LSTM and GRU as decoders.

Considering the current versions of the image captioning models, the decoders are presented in the form of transformers. The fully-attentive architecture was, first, shown by Vaswani et al. [20]. This was then used in the further development of the NLP field and became a part of BERT [6] and GPT [13]. Currently, BERT-like architecture is widely used for image captioning because of its structure. In such paradigms, the visual and language models are combined almost in the beginning and textual parts can be initialized with checkpoints trained with large texts [6].

After the introduction of Vision Transformer (ViT) in [3], researchers started to consider image captioning from the perspective of a sequence-to-sequence problem and to develop convolution-free models by replacing CNNs with vision transformers in the CNN-Transformer configurations. In [10], [4], the authors used the ViT model as the backbone for the image encoder while in [23], another type of vision transformer – Swin Transformer, was utilized. All of these works outperformed CNN-based methods for image captioning. We employed a similar ViT-based framework in the main approach of our project and compared the ViT-RNN configuration with the CNN-RNN one.

## 3. Dataset

Image captioning datasets Flickr30K [12] and Flickr8K [1] were chosen to test the performance of our approach. Flickr8K contains a collection for sentence-based image description and search, consisting of 8,000 images, while Flickr30K contains 31,000 images. Both of the datasets are paired with five different captions which provide clear descriptions of the salient entities and events, providing diverse descriptions of the image content. The datasets are not divided into training and test sets. To have more data for training, we used 90% of datasets to train our models while 10% of data were employed for evaluation. The images in these datasets cover a wide array of scenes, objects, activities, and contexts. For instance, both datasets contain pictures of people engaged in various activities, both indoors and outdoors, everyday activities, landscapes, urban settings, events, social gatherings, and more. While both datasets have a similar structure in terms of images being paired with multiple captions, they are distinct datasets with different sets of images.

These datasets were picked by our team since the datasets cover a wide range of scenes, objects, and activities, enabling models to learn diverse visual features and representations. Also, these datasets serve as benchmarks for evaluating the performance of various image captioning models [19]. Finally, the small size of the datasets compared to other larger datasets makes it more manageable

for experimentation, especially for us with limited computational resources.

We needed to pre-process the images and captions of these datasets to be compatible with our models. The images were pre-processed according to the feature extraction methods. For example, the required image size for feature extractors used in this project is shown in Table 1.

Table 1: Required image sizes for different feature extractors

Feature Extractor	Image Size
VGG	224
Inceptionv3	299
ResNet	224
ViT	384

To reduce the size of the vocabulary and train the models faster, the captions for each image identifier were cleaned via pre-processing techniques:

- Converting to lowercase
- Deleting digits, one-letter words, and special characters
- Deleting additional spaces
- Adding start and end tags to the caption

This pre-processing technique made the model smaller and faster in training [14]. Cleaned text data were then converted into numeric values to be able to use them to train the models.

## 4. Methodology

### 4.1. Baseline

The encoder-decoder design is implemented as the baseline for which the CNN encoder and RNN decoder for the vision and language parts are chosen.

#### 4.1.1 Encoder

The encoder in image captioning stands for extracting the feature vectors from the input via CNN.

VGG-16 is used as the encoder that is pre-trained on ImageNet for image captioning. This model gets 92.7% on the ImageNet competition, demonstrating its ability to identify 1000 images across 1000 categories [17]. VGG-16 is a deep CNN architecture known for its 16 weight layers, which include 13 convolutional layers, 5 max pooling layers, and 3 dense layers. Despite having 21 layers in total, it focuses on 16 layers with learnable parameters. The unique feature

of VGG-16 is its use of 3x3 filters with a stride of 1 and consistent same padding for convolution layers, along with 2x2 filters and a stride of 2 for max pooling layers. The input tensor size is 224x224x3, representing RGB channels. Notably, VGG-16 uses a consistent configuration of convolution and max pool layers, which makes it stand out for its simplicity. Conv-1, Conv-2, Conv-3, and Conv-4/5 are the convolution layers with 64, 128, 256, and 512 filters, respectively.

Transfer learning is employed to extract features, utilizing layers preceding the Softmax layer. These feature maps are stored in a dictionary, associating them with image identifiers. This approach optimizes subsequent stages by enabling quick loading of pre-extracted features [17], [14]. Each image’s feature is represented as a one-dimensional 4096-element vector, efficiently stored in a dictionary on the hard disk alongside its corresponding identifier.

#### 4.1.2 Decoder

The decoder in image captioning stands for generating a group of tokens for caption after taking the feature vector from an image and partial caption.

Long Short-Term Memory (LSTM) is used as the decoder to create captions for the image by common text representation to generate sentences [17], [25]. LSTM is efficient at remembering and predicting long-term sequences of variable sizes and handling the problem of vanishing gradients encountered by traditional RNN. It Comprises of three integral components, which are the forget gate, input gate, and output gate along with a memory cell. Overall, LSTM ensures control over information flow. The forget gate plays a pivotal role in deciding whether to retain or discard data from the previous timestamp, offering a mechanism for selective memory. Concurrently, the input gate assesses the importance of the arriving data and calculates how much it adds to the memory cell. The output token with the greatest score, on the other hand, is coordinated by the output gate, which ultimately shapes the caption. Together, these gates control the cell state, which is the LSTM’s memory.

By loading the features from the image and the text embeddings (partial captions), the LSTM is capable of generating the next word in the caption based on the current hidden state, input embeddings, and context from image features [16]. Afterwards, the generated word becomes the partial caption for the next timestamp. This process is repeated until generating the end token or reaching the maximum caption length. Finally, the LSTM, with its visual features and text embeddings, is capable of generating captions that are coherent and accurate for the images.

### 4.1.3 Model Architecture

In the context of the image encoder, as illustrated in Figure 1, the features extracted from the VGG-16 encoder are fed into a dense layer with a size matching that of the feature vector. A subsequent dropout layer, set at 40%, is applied to mitigate the risk of overfitting. This is followed by another dense layer consisting of 256 units with a rectified linear unit (ReLU) activation function to produce a vector with a fixed dimension of 256.

For sequence processing, the indices of partial captions are initially padded with zero for batch processing. The indices of partial captions undergo loading into an input layer with a size equal to the maximum sequence length of the captions. After that, the indices are mapped to a 256-dimensional vector in the embedding layer, creating an embedding matrix for all the unique words. Following that, a dropout layer of 40% is applied to reduce the overfitting pattern or memorization of the data.

In the decoder phase, the LSTM takes the sequence of embedded words to capture sequential dependencies, generating hidden states. The output of the LSTM is combined with vector of image features via addition operation, allowing the model to capture relationships between the two modalities. After that, combined features are feed-forwarded to a dense layer of 256 dimensions with the activation function ReLU. Finally, the input from previous layers is sent to a final output dense layer of dimension of vocabulary size with the activation function Softmax to output the probability distribution along all the unique words.

## 4.2. Main Approach

Our main approach is also takes use of the encoder-decoder architecture. For encoder we used state-of-the-art ViT model, and for decoder we improved the RNN with the attention layers.

### 4.2.1 Vision Transformer Encoder

ViT [3] utilizes the idea that a raw image can be divided into smaller image patches, which is analogous to tokens in a sentence. For the purpose of this project, we used a pre-trained ViT. The architecture of the ViT is BERT-like and consists of encoder and decoder. The sequence of the patches are at a resolution of  $16 \times 16$  and are linearly embedded. Adding absolute position embeddings also allows us to use the resulting vectors for a Transformer encoder. While the model was originally pretrained on the large-scale ImageNet-21K with images at a resolution of  $224 \times 224$  pixels, it still remains much more efficient and requiring much more less computational resources than CNNs [3].

In our implementation, we used the Keras implementation of the ViT presented in [3]. We chose the model ViT base patch 16 model which was pretrained on the images of

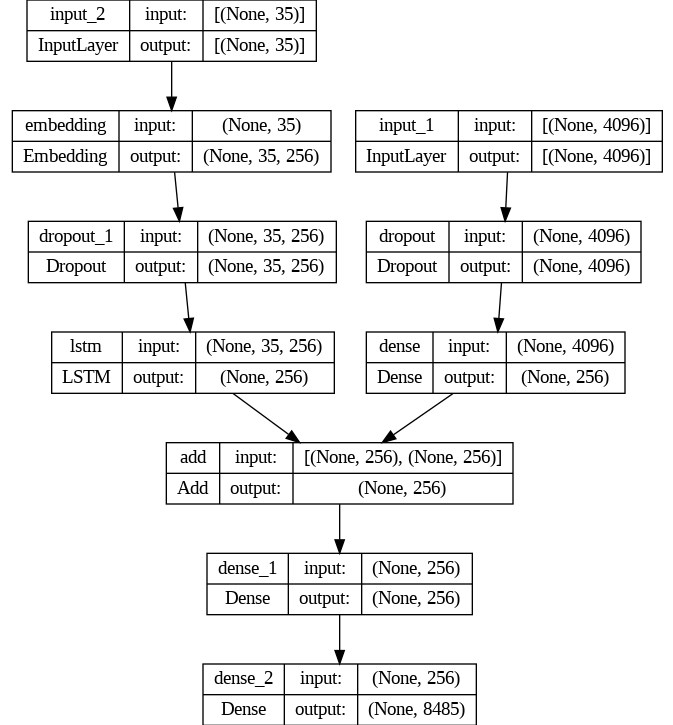


Figure 1: Baseline model architecture with pre-trained VGG-16 and LSTM (all dimensions are shown for Flickr8K)

size  $384 \times 384$ . We dropped the last layers of the model to achieve the embeddings to then feed them into encoder. Lastly, we used the model to extract the features from the dataset’s images and used them as input for encoder.

### 4.2.2 RNN Decoder with Attention Layers

The decoder architecture consists of the three main parts: trainable embedding layer, bidirectional LSTM, and self-attention layer. First, we used unfrozen embedding layer with the output dimension of 500, so that it can learn 500D representation for each token that we gained from the captions in the train part of the dataset. To add more context, we extended the LSTM in the baseline in the form of bidirectional LSTM. Bidirectionality provides better and faster training process. Basically, we have to LSTMs: one takes the first sample from the train set as the first input, and the other one takes the last. The output is a concatenation of the results of the LSTMs. Thirdly, we have self-attention layer (Keras implementation) that further extends our decoder and overall improves the architecture. The mechanism of the attention layer works such that it obtains a set of scoring weights that connects the input with decoder so that it can focus on the most important part of the input se-

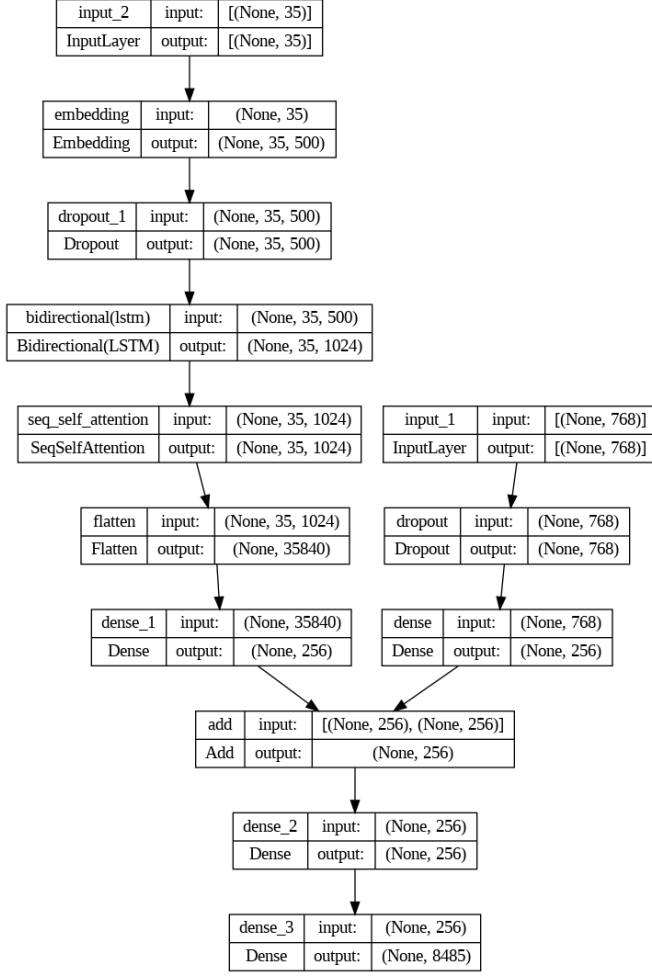


Figure 2: Main approach model architecture (all dimensions are shown for Flickr8K)

quence and generate a better output. The last part of the decoder is a dense layer that produces the vectors of the dimension of 256. The architecture of the decoder can be seen in the figure 2, which represents the main approach architecture.

#### 4.2.3 Model Architecture

Figure 2 represents the main encoder-decoder architecture that we used. The encoding part takes the features produced by ViT with the size of 768. The subsequent dropout layer with the rate of 0.5 provides with regularization of the training process and prevention of the potential overfitting. Last layer in the encoder is a fully-connected layer that is the same as in decoder. The outputs of the encoder and decoder are connected using add layer, which outputs vectors at a dimension of 256. The output of the add layer is transferred

to the dense layer, which is followed by an output dense layer, which connects the feature vectors to the size of the vocabulary.

#### 4.3. Training

The training of both baseline and main approach models was conducted in a supervised manner. Image features and partial captions are the inputs of the model and target variable is the next word after partial caption sequence. The training was conducted for 20 epochs with Adam optimization algorithm and default learning rate setting of 0.001 and batch size of 32. During the training, the model is updated via Adam optimizer and sparse categorical cross-entropy as the loss to backpropagate and update the weights of the model.

#### 4.4. Inference

The inference method that was used for captioning the images is called greedy search. First, we have the vocabulary set for the whole dataset, which consists of all the possible words. When the model takes an image, the input consists of the image itself and 'startseq' token, which indicates the start of the resultant caption. Then, we apply we apply maximum likelihood estimation (MLE) method for predicting the word. In other words, the model chooses the word from the vocabulary set with the highest probability given the image and sequence input. The sequence, which will serve as a caption further, will be supplemented with the predicted word after each iteration. The inference ends when the model predicts 'endseq' token or when the number of words in the sequence is greater than the given threshold.

#### 5. Evaluation Metric

Standard evaluation metrics include BLEU, METEOR, ROUGE, and CIDEr. These metrics are commonly used in NLP tasks and compare the generated captions to reference captions produced by humans [19]. In our work for assessing image captioning systems, one primary metric has been chosen: BLEU (Bilingual Evaluation Understudy), which offers distinct advantages and insights. To calculate the BLEU score the following formula is used:

$$\text{BLEU} = \text{BP} \times \exp \left( \sum_{n=1}^N w_n \cdot \log(\text{precision}_n) \right) \quad (1)$$

where:

- BP is the brevity penalty, penalizing short translations.
- $N$  is the maximum order of n-grams considered.
- $w_n$  is the weighting vector, which is most typically chosen to be equal  $\frac{1}{N}$

- $\text{precision}_n$  is the precision of n-grams, representing the ratio of the number of n-grams in the candidate that appear in at least one reference translation to the total number of n-grams in the candidate.

The brevity penalty (BP) is determined as follows:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-\frac{r}{c})} & \text{if } c \leq r \end{cases} \quad (2)$$

where  $c$  is the length of the candidate translation, and  $r$  is the effective reference length.

BLEU quantifies the similarity between generated captions and human references by comparing n-gram overlaps. It calculates the precision of n-grams (up to a certain length) in the generated sentence compared to the reference sentences and aggregates them using a weighted geometric mean. BLEU is widely used, straightforward to compute, and correlates well with human judgments of translation quality. However, it has limitations in capturing linguistic nuances, tends to favor shorter sentences, and does not account for synonyms or paraphrases [22].

## 6. Results & Analysis

Besides training and testing our baseline and main approaches, we performed several additional experiments on the Flickr8K dataset. Due to limitations in time and computational power of our devices, we present only the results of the baseline model and the model that gave the best performance on Flickr8K for the Flickr30K dataset.

Below, we provide the details and results of the implemented experiments.

### 6.1. Experiment 1: Pre-trained CNNs for Feature Extraction

Apart from VGG16, we extracted features from images using pre-trained VGG19, Inceptionv3, and ResNet (50, 101, and 152) architectures and evaluated results for the baseline configuration. All of these architectures are available in the Keras library. Table 2 summarizes information about obtained BLEU scores for CNN-based feature extractors.

This experiment demonstrated the superiority of ResNet101 and ResNet152 over other pre-trained CNN architectures for feature extraction.

### 6.2. Experiment 2: Pre-trained ViTs for Feature Extraction

After experimenting with CNN-based feature extractors, we concentrated on ViT-based features. ViT-based architectures are also available in the Keras library. Before testing the main approach architecture, we evaluated ViT-based

Table 2: Results for various CNN-based feature extractors

Feature Extractor	BLEU-1	BLEU-2	BLEU-3	BLEU-4
VGG16	0.532331	0.306485	0.188393	0.110595
VGG19	0.54278	0.317753	0.196851	0.114633
Inceptionv3	0.539533	0.321853	0.206724	0.127761
ResNet50	0.537743	0.312815	0.197654	0.119976
ResNet101	0.562416	0.340594	0.219936	0.137453
ResNet152	0.567416	0.344778	0.220902	0.134931

feature extractors in combination with the baseline configuration due to computational complexity considerations. The obtained results can be seen in Table 3.

Table 3: Results for various ViT-based feature extractors

Feature Extractor	image size	BLEU-1	BLEU-2	BLEU-3	BLEU-4
ViT_B16	224	0.589792	0.376309	0.252349	0.1596
ViT_B16	384	0.620533	0.407826	0.275496	0.177146
ViT_B32	384	0.609596	0.391517	0.261425	0.165123
ViT_L16	384	0.615429	0.396339	0.261224	0.161472
ViT_L32	384	0.607313	0.381917	0.247032	0.149615

Due to this experiment, we identified that pre-trained ViT\_B16 results in the extraction of best features from images compared not only with other ViT-based feature extraction algorithms but also with CNN-based feature extractors. Furthermore, we found that resizing images to the image size of 384 results in better BLEU scores than using the image size of 224.

### 6.3. Experiment 3: Changing LSTM to GRU in the Baseline Model

In one of our experiments, we changed LSTM in the baseline configuration to GRU. A comparison of them in combination with ResNet101 and ViT\_B16 (image size = 224) is provided in Table 4.

Table 4: Comparison of LSTM and GRU

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
ResNet101 + LSTM	0.562416	0.340594	0.219936	0.137453
ResNet101 + GRU	0.553384	0.330481	0.210729	0.126979
ViT_B16 + LSTM	0.589792	0.376309	0.252349	0.1596
ViT_B16 + GRU	0.587695	0.369424	0.243477	0.154711

This experiment illustrated the advantage of LSTM, and

therefore, we did not use GRU in combination with other feature extraction methods.

#### 6.4. Experiment 4: Changing Hyperparameters in the Baseline Model

Changing the original dropout rates from 0.4 to 0.3 for image feature layers and from 0.4 to 0.2 for sequence feature layers and batch size from 32 to 64 led to improvement in results in combination with some feature extractors and to deterioration - with others (Refer to Table 5. This means that hyperparameter tuning must be done individually for each model configuration and that hyperparameter tuning techniques like grid search or random search must be applied. Due to limitations in time and computational power of our computers, we did not concentrate on hyperparameter tuning, but this could be a crucial step for further improvement in future work.

Table 5: Comparison of original and tuned hyperparameters

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
ResNet50 original	0.537743	0.312815	0.197654	0.119976
ResNet50 tuned	0.602774	0.380594	0.253107	0.159685
ViT_B16 original	0.620533	0.407826	0.275496	0.177146
ViT_B16 tuned	0.593804	0.388224	0.258429	0.164382

#### 6.5. Experiment 5: Baseline vs. Main Approach

Finally, we used the main approach model in combination with VGG16 and ViT\_B16 features instead of the baseline model. Table 6 shows the obtained results.

Table 6: Comparison of baseline and main approaches

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
VGG16 baseline	0.532331	0.306485	0.188393	0.110595
VGG16 main approach	0.522181	0.294183	0.181005	0.103889
ViT_B16 baseline	0.620533	0.407826	0.275496	0.177146
ViT_B16 main approach	0.598732	0.383359	0.254808	0.158814

From the results, we can see that our main approach performs worse compared to our baseline approach for both feature extraction methods. In general, more complex models should be utilized with large datasets. Given the sizes of the Flickr8K and Flickr30K datasets, it is enough to apply the baseline model for them.

Starting from VGG16 for feature extraction and the baseline model and then experimenting by changing the architecture and hyperparameters, we identified that the best BLEU scores are obtained with ViT\_B16 feature extractor (image size = 384) in combination with the baseline model. In Table 7, we summarized the results for our first/baseline, main approach, and the best-obtained model for Flickr8K. For Flickr30K, we do not provide the results of the main approach model due to worse performance on Flickr8K and computational complexity, which results in long training time.

Table 7: Summary of results for Flickr8K and Flickr30K

Dataset	Metric	VGG16 baseline	ViT_B16 baseline	ViT_B16 main approach
Flickr8K	BLEU-1	0.532331	<b>0.620533</b>	0.598732
	BLEU-2	0.306485	<b>0.407826</b>	0.383359
	BLEU-3	0.188393	<b>0.275496</b>	0.254808
	BLEU-4	0.110595	<b>0.177146</b>	0.158814
Flickr30K	BLEU-1	0.535358	<b>0.634464</b>	-
	BLEU-2	0.288893	<b>0.392751</b>	-
	BLEU-3	0.169337	<b>0.251295</b>	-
	BLEU-4	0.091678	<b>0.150727</b>	-

Table 8 illustrates some qualitative results of our models. The difference between our first baseline model based on VGG16 features and our best-obtained model based on ViT\_B16 features is especially pronounced for more complex images. For example, ViT-based models successfully identified the main context of basketball and protests in the second and third images, while VGG16-basel models failed to do this. Though the ViT\_B16 baseline model gives the best quantitative results in terms of BLEU scores, the ViT\_B16 main approach model seems to generate better captions in terms of grammar. Overall, the obtained models cannot generate high-quality captions for any possible image since they are limited by words in the Flickr8K and Flickr30K datasets. In Fig. 3, the frequency distribution of the 60 most frequent words in these datasets is provided. For example, 'dog' is a frequent word in both datasets; therefore, it is not surprising that the models misidentified a cat as a dog in the fourth image.

## 7. Error and Limitations Analysis

In our quest to develop a robust Image-to-text generator, a series of experiments were conducted to understand the system's strengths and weaknesses. While valuable insights were gained, it is essential to acknowledge certain limitations and potential errors.

### Experiment 1: Pre-trained CNNs for Feature Extraction

Among various pre-trained CNN architectures, ResNet101 and ResNet152 demonstrated superiority. However, we performed the experiments only on Flickr8K. It is also needed to conduct experiments on other datasets to verify our results.

### Experiment 2: Pre-trained ViTs for Feature Extraction

ViT\_B16 with an image size of 384 showed promising results. Computational constraints prevented a comprehensive evaluation on other datasets, and as in the case of pre-trained CNNs, performance may be sensitive to the Flickr8K dataset.

### Experiment 3: Changing LSTM to GRU in the Baseline Model

LSTM outperformed GRU, emphasizing the architecture-dependent nature of image captioning. Therefore, more architectures should be examined to get better results.

### Experiment 4: Changing Hyperparameters in the Baseline Model

Hyperparameter tuning led to mixed results, which means that more consistent hyperparameter tuning methods should be utilized.

**Experiment 5: Baseline vs. Main Approach** The main approach unexpectedly performed worse in terms of BLEU score, highlighting the importance of considering model complexity and dataset sizes. Furthermore, it may be needed to evaluate image captioning systems with more than one metric.

A major con of our method lies in our approach of captions pre-processing. For example, by excluding one-letter words, we get rid of the article 'a' in the generated text, which affects the grammar of generated captions. Furthermore, limited dataset sizes, computational constraints, and time limitations significantly influenced the obtained results. However, experimenting with different feature extraction algorithms, model architectures, and hyperparameters resulted in useful insights and can be advantageous for the development of more robust image captioning methods.

## 8. Future Work

In future work, there are opportunities for improving our Image-to-text generator. We should conduct more thorough hyperparameter tuning, using methods like grid search or random search. Furthermore, considering ensemble models and combining predictions from multiple models might enhance overall performance. Implementing data augmentation techniques to expand and diversify the dataset or using

a larger dataset could improve the model's ability to generalize. Fine-tuning pre-trained models for feature extraction also holds the potential for better results.

## 9. Code

We submit our codes together with the report. Additionally, the codes, all extracted features, and models can be found via the following link: Codes.

## References

- [1] K. Anitha Kumari, C. Mouneeshwari, R. Udhaya, and R. Jasmitha. Automated image captioning for flickr8k dataset. In *Proceedings of International Conference on Artificial Intelligence, Smart Grid and Smart City Applications: AISGSC 2019*, pages 679–687. Springer, 2020.
- [2] H. Ayesha, S. Iqbal, M. Tariq, M. Abrar, M. Sanaullah, I. Abbas, A. Rehman, M. F. K. Niazi, and S. Hussain. Automatic medical image interpretation: State of the art and future directions. *Pattern Recognition*, 114:107856, 2021.
- [3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [4] Z. Fang, J. Wang, X. Hu, L. Liang, Z. Gan, L. Wang, Y. Yang, and Z. Liu. Injecting semantic concepts into end-to-end image captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18009–18019, 2022.
- [5] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.
- [6] J. D. M.-W. C. Kenton and L. K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2, 2019.
- [7] P. Kinghorn, L. Zhang, and L. Shao. A hierarchical and regional deep learning architecture for image description generation. 119:77–85.
- [8] Z. Li, Y. Mu, Z. Sun, S. Song, J. Su, and J. Zhang. Intention understanding in human–robot interaction based on visual-nlp semantics. *Frontiers in Neurorobotics*, 14:610139, 2021.
- [9] M. Liu, L. Li, H. Hu, W. Guan, and J. Tian. Image caption generation with dual attention mechanism. 57(2):102178.
- [10] W. Liu, S. Chen, L. Guo, X. Zhu, and J. Liu. Cptr: Full transformer network for image captioning. *arXiv preprint arXiv:2101.10804*, 2021.
- [11] H. Parvin, A. Reza Naghsh-Nilchi, and H. Mahvash Mohammadi. Image captioning using transformer-based double attention network. 125:106545, 2023.
- [12] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE interna-*



- tional conference on computer vision*, pages 2641–2649, 2015.
- [13] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.
  - [14] H. Realm. Image Caption Generator using Python — Flickr Dataset — Deep Learning Tutorial — hackersrealm.net. <https://www.hackersrealm.net/post/image-caption-generator-using-python>. [Accessed 01-12-2023].
  - [15] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7008–7024, 2017.
  - [16] K. Rungta, G. Chau, A. Dewangan, M. Wagner, and J.-L. Huang. Image captioning using an lstm network.
  - [17] A. S. Salim, M. B. Abdulkareem, Y. E. Fadhel, A. B. Abdulkareem, A. M. Shantaf, and A. B. Abdulkareem. Novel image caption system using deep convolutional neural networks (vgg16). In *2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–6. IEEE, 2022.
  - [18] A. Stangl, N. Verma, K. R. Fleischmann, M. R. Morris, and D. Gurari. Going beyond one-size-fits-all image descriptions to satisfy the information wants of people who are blind or have low vision. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–15, 2021.
  - [19] M. Stefanini, M. Cornia, L. Baraldi, S. Cascianelli, G. Fiameni, and R. Cucchiara. From show to tell: A survey on deep learning-based image captioning. *IEEE transactions on pattern analysis and machine intelligence*, 45(1):539–559, 2022.
  - [20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
  - [21] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.
  - [22] J. Wang, W. Xu, Q. Wang, and A. B. Chan. On distinctive image captioning via comparing and reweighting. 45(2):2088–2103.
  - [23] Y. Wang, J. Xu, and Y. Sun. End-to-end transformer based model for image captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2585–2594, 2022.
  - [24] L. Xu, Q. Tang, J. Lv, B. Zheng, X. Zeng, and W. Li. Deep image captioning: A review of methods, trends and future challenges. *Neurocomputing*, page 126287, 2023.
  - [25] Y. Yu, X. Si, C. Hu, and J. Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.



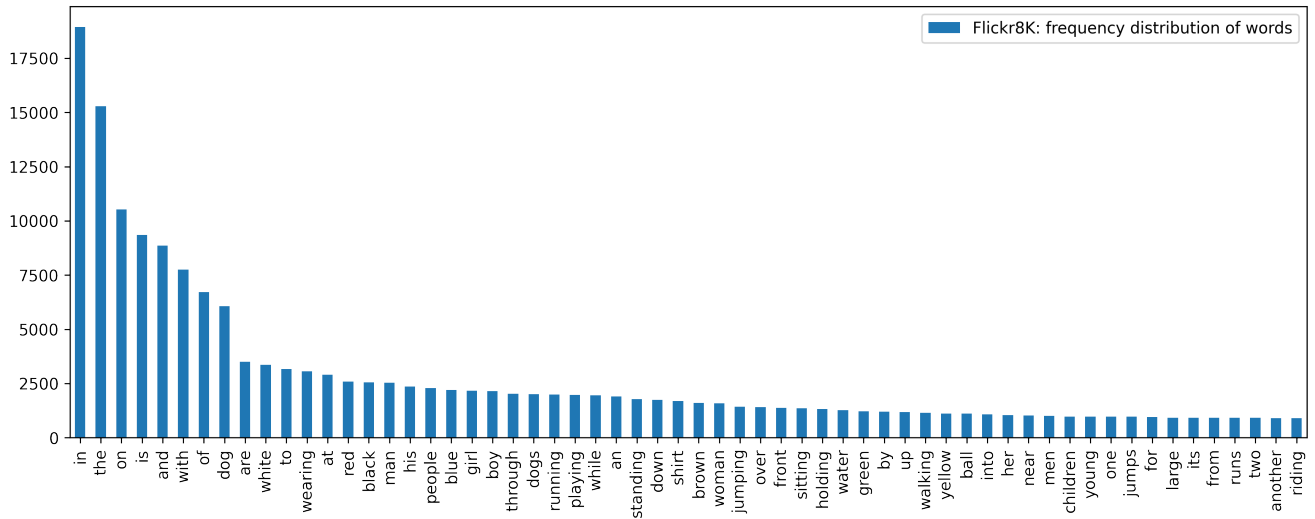
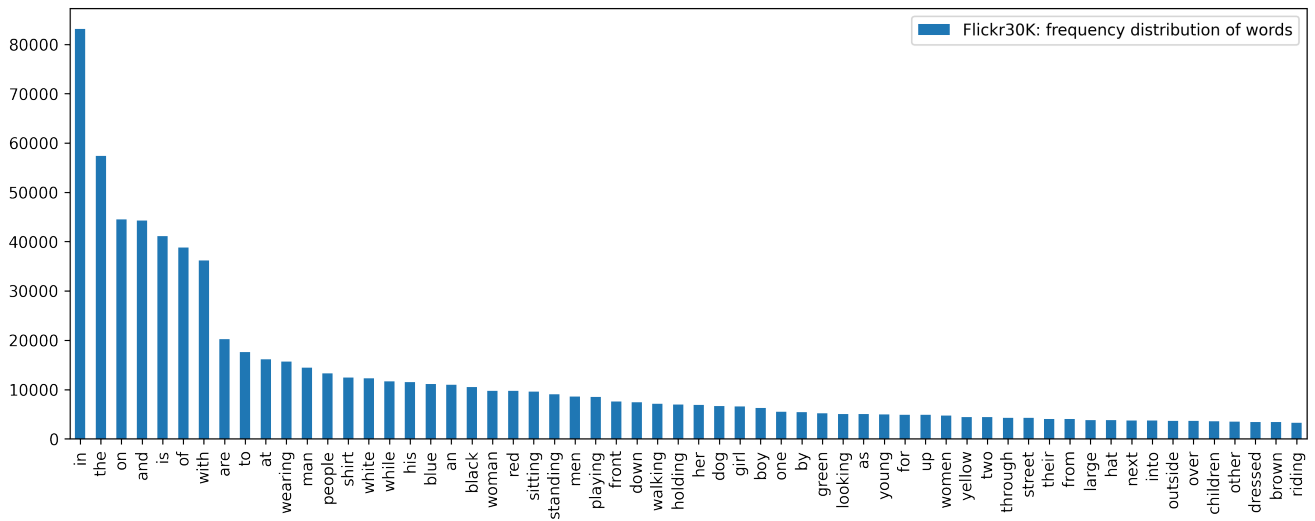
Image from Flickr8K	Captions by models trained on Flickr8K	Ground truths
	<p><b>VGG16 baseline:</b> girl in pink swimsuit is swimming in pool</p> <p><b>ViT_B16 baseline:</b> girl in watermelon swimsuit plays in water</p> <p><b>ViT_B16 main approach:</b> girl in pink bathing suit is playing in the water</p>	<p><b>GT1:</b> girl makes an arc of water with her hair in pool</p> <p><b>GT2:</b> young girl doing back flip in the water</p> <p><b>GT3:</b> the girl is taking her hair out of the water</p>
Image not from datasets	Captions by models trained on Flickr8K	Ground truths
	<p><b>VGG16 baseline:</b> two girls are playing in karate game</p> <p><b>ViT_B16 baseline:</b> two men playing basketball</p> <p><b>ViT_B16 main approach:</b> two men are playing basketball</p>	<p><b>GT: —</b></p>
Image from Flickr30K	Captions by models trained on Flickr30K	Ground truths
	<p><b>VGG16 baseline:</b> man in black shirt and black pants is walking down the street</p> <p><b>ViT_B16 baseline:</b> protesters are protesting against censorship of war</p>	<p><b>GT1:</b> people protesting immigration</p> <p><b>GT2:</b> male picket holders protesting immigration outside of building</p> <p><b>GT3:</b> four men stand holding picket signs referencing stances on immigration</p>
Image not from datasets	Captions by models trained on Flickr30K	Ground truths
	<p><b>VGG16 baseline:</b> two white and white dogs running through the grass</p> <p><b>ViT_B16 baseline:</b> two dogs are playing in field</p>	<p><b>GT: —</b></p>

Table 8: Captions generated by models and some corresponding ground truths



(a) Flickr8K: frequency distribution of words



(b) Flickr30K: frequency distribution of words

Figure 3: 60 most frequent words in Flickr8K and Flickr30K