Image Captioning Using Transformer

Abhishek Tarapara (22CSM1R01)
National Institute of Technology, Warangal
An Institute of National Importance
dept. of Computer Science and Engineering (of NITW.), Warangal, India

***Abstract*—Generating natural language descriptions of an image's content for captioning is a difficult undertaking. In recent years, the task has received a lot of interest from the computer vision and natural language processing areas, and numerous ways have been put out to address it. In this situation, a potent solution for image captioning has been found in the combination of pre-trained Convolutional Neural Networks (CNNs) for image feature extraction and Transformers for natural language production.**

***Index Terms*—Image captioning, Pretrained CNN, Transformer, Natural language processing, Computer vision, Deep learning, Convolutional neural networks, Attention mechanism, Transfer learning, Image recognition, Language generation, Neural networks, Fine-tuning, Encoder-decoder architecture, Multimodal learning**

I. Introduction

A difficult problem in computer vision and natural language processing is image captioning, the process of creating a natural language description of a picture. By utilising the capabilities of deep neural networks, particularly convolutional neural networks (CNNs) and transformers, substantial advancements in this area have been made recently. Transformers are renowned for their capacity to recognise long-term connections in natural language sequences, whereas CNNs are famed for their capacity to extract significant information from images.

Utilising pre-trained CNNs as feature extractors and feeding the extracted features into a language model, such as a transformer, to produce captions is one method of image captioning. This method's ability to produce captions of a high calibre has attracted a lot of interest. Due to their outstanding performance in image classification tasks, pre-trained CNNs like VGG, ResNet, and Inception have been extensively used in this field. Text creation jobs that require natural language processing have been found to benefit greatly from transformers, such as the popular BERT and GPT-2 models.

This article provides a thorough analysis of current studies on the use of transformers and CNNs for image captioning. We go over the different strategies that have been employed in this situation, such as optimising pre-trained CNNs, combining CNNs and transformers, and employing pre-trained transformers for caption production. We also go into the datasets utilised for analysis, the metrics for judging caption quality, as well as the problems and directions this field is headed in the future. Overall, the state-of-the-art methods for picture captioning employing pre-trained CNNs and transformers are thoroughly explained in this study.

The use of pre-trained CNNs and Transformers for picture captioning has been the subject of numerous research articles, with many reaching cutting-edge performance on popular benchmark datasets like COCO and Flickr30k. To further improve their effectiveness, these models frequently combine attention mechanisms and fine-tuning strategies.

Overall, pre-trained CNNs and Transformers for picture captioning provide a potential research avenue with a wide range of applications, including content creation, image search, and assistive technology for the blind.

II. Realated Work

A. Image Captioning

A well-researched issue in computer vision and natural language processing, image captioning has received a lot of attention recently. We go over some of the most well-known studies on pre-trained convolutional neural networks (CNNs) and transformers for picture captioning in this part.

Using a deep neural network composed of a pre-trained CNN for feature extraction and a recurrent neural network (RNN) for language modelling, Vinyals et al. proposed a ground-breaking model for image captioning in 2014. The model, known as Show and Tell, was state-of-the-art on numerous benchmark datasets after being trained end-to-end on a sizable dataset of image-caption pairs. This research provided the framework for other studies in the area of image captioning.

Later studies looked on feature extraction using pre-trained CNNs for picture captioning. For example, Fang et al. citefang2015captions extracted picture characteristics using the VGGNet citesimonyan2014very pre-trained on the ImageNet dataset citedeng2009imagenet and integrated them with a language model built using an LSTM network. They showed that the performance of image captioning models can be greatly enhanced by pre-trained CNNs.

Researchers have looked into using transformers for picture captioning in light of the success of pre-trained CNNs. The transformer model for natural language processing was proposed in 2018 by Vaswani et al., who obtained state-of-the-art performance on a number of language tasks. Researchers investigated the use of transformers in picture captioning as a result of this accomplishment. For instance, ViLBERT, a model that combines a vision transformer with a language transformer to simultaneously describe visual and textual information, was proposed by Lu et al. (cite:lu2019vilbert). They demonstrated that ViLBERT met all of the benchmarks for image captioning with state-of-the-art performance.

Recently, researchers have also looked into using transformers and pre-trained CNNs together to caption images. For instance, Zhou et al. suggested a unified architecture that incorporates a pre-trained CNN with a transformer-based language model in their paper cited as "zhou2020unified". On numerous benchmark datasets, they demonstrated that their model, known as UNICORN, beat cutting-edge techniques.

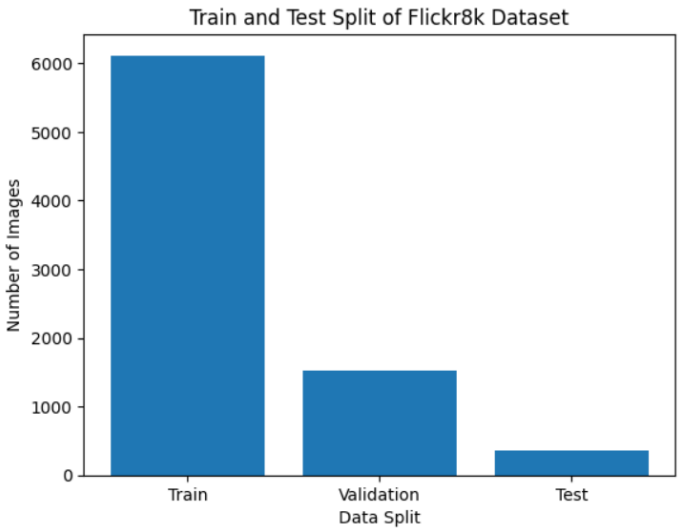
In conclusion, the performance of image captioning models has greatly improved with the usage of pre-trained CNNs and transformers. To further increase the precision and effectiveness of these models, researchers are investigating novel structures and methodologies.

B. Data Pre-processing

We must perform data pre-processing in order to provide images for model training. To achieve this, we load a dataset of image captions, process the captions, and divide the data into training and validation sets. The 8,000 photos with five captions that make up the Flickr8k dataset are used. The script produces a dictionary that maps each image to its caption by reading in a text file containing the names of the images and their matching captions.

After deleting any captions that are either too short or too long, the script processes the remaining captions by giving each one a start and end token. The data is then divided into training and validation sets, with the training set including a user-specified portion of the data and the validation set comprising the remaining portion.

The 6114 training samples and 1529 validation samples that were produced are the script's final outputs. A machine learning model that can produce captions for fresh images can be trained using this processed dataset as its input.



The next step in our data preparation is to perform text vectorization. Text vectorization is the process of converting text into numerical data that can be fed into a deep learning model. In our case, we will use the TextVectorization class from the TensorFlow library to perform this task.

We will start by defining a custom function called custom standardization which will be used to standardize the text data. This function will perform the following operations on the input text

- Convert all uppercase letters to lowercase
- Remove any characters in the strip-chars variable, which contains a set of special characters we want to remove from the text

We will then call the adapt method of the TextVectorization instance and pass in our preprocessed text data as an argument. This will build the vocabulary based on the most frequent words in the data.

III. Model

This Transformer model for image captioning uses a convolutional neural network (CNN) encoder and is implemented in TensorFlow. The TensorFlow Keras API is used to define the model.

- One layer of the Transformer encoder is implemented by the Transforme rEncoderBlock class. The layer is made up of two fully connected layers, a multi-head self-attention mechanism, and residual connections with layer normalisation. The call method accepts an input tensor of the form (batch-ize, sequence-length, embed-dim) and generates an output tensor of the same form by using the attention mechanism and completely connected layers.
- The positional encoding layer utilised by the Transformer decoder is defined by the PositionalEmbedding class. By adding positional embeddings to the token embeddings, the layer accepts an input tensor of shape (batch-size, sequence-length) representing the input sequence and returns an output tensor of shape (batch-size, sequence-length, embed-dim).
- One layer of the Transformer decoder is implemented by the TransformerDecoderBlock class. The layer is made up of two completely connected layers, a multi-head attention mechanism with encoder output, a multi-head attention mechanism with masking, and residual connections with layer normalisation. The call method applies the attention mechanisms and fully connected layers to produce a probability distribution over the output vocabulary of size VOCAB-SIZE. It takes an input tensor of shape (batch-size, sequence-length) representing the target sequence, the output of the CNN encoder, and an optional mask tensor representing the padding of the input sequence. The decoder's masked self-attention mechanism uses a causal attention mask, which the get-causal-attention-mask method returns.
- This neural network approach for captioning images is used. The model consists of an image feature extraction Convolutional Neural Network (CNN) and caption generation Transformer-based Decoder. The Transformer Decoder is made up of a stack of TransformerDecoderBlocks, and the CNN is built on the EfficientNetB0 architecture. Each caption token is created by the TransformerDecoderBlock by paying attention to both the picture features and the tokens that have already

been created. The model has dropout regularisation to avoid overfitting and positional embeddings to capture the tokens' sequential order.

Words which we are trying to encode	Words that we are trying to attend to: The	dog	is	playing	outside
The	masked	masked	masked	masked	masked
dog	can attend	masked	masked	masked	masked
is	can attend	can attend	masked	masked	masked
playing	can attend	can attend	can attend	masked	masked
outside	can attend	can attend	can attend	can attend	masked

In sequence-to-sequence models like the Transformer, masking in the decoder is a method used to stop the model from paying attention to upcoming time steps during training.

The output of the decoder is determined by the input and output sequences up to that time step. However, the output sequence is not known ahead of time during training. In order to avoid the decoder from attending to future time steps where the output sequence is not yet available, we employ masking.

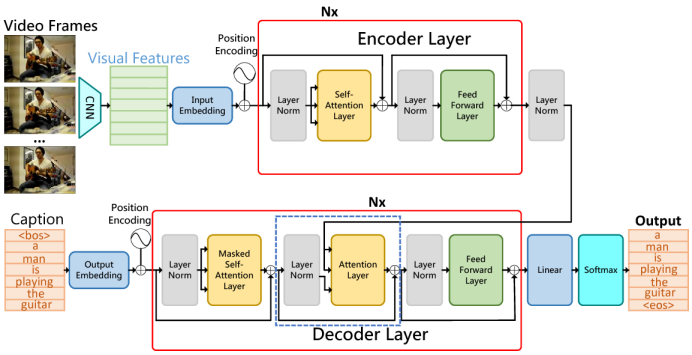


Fig. 1. BASE MODEL ARCHITECURE

This code specifies a neural network model for image captioning that is transformer-based. The model architecture comprises of a transformer-based decoder to produce the captions and a convolutional neural network (CNN) to extract features from the input image.

Using the EfficientNetB0 architecture that has already been trained on ImageNet, the get-cnn-model function defines the CNN portion of the model. This model's feature extractor layers are frozen to prevent the updating of their weights during training.

The single transformer encoder block that will be used to encrypt the image features is defined by the TransformerEncoderBlock class. It employs layer normalisation, feed-forward layers, residual connections, and multi-head attention. The input sequences receive positional embeddings from the PositionalEmbedding class.

The single transformer decoder block that will be utilised to decode the picture features and produce the captions is defined by the TransformerDecoderBlock class. Layer normalisation, residual connections, and feed-forward and multi-head attention layers are also used. When anticipating the current word, the get-causal-attention-mask method creates a causal mask that is used to hide the words that will appear in the future. During training, the model is regularised using the dropout layers.

The task of removing features from the input image falls to the encoder. In this implementation, the encoder is an EfficientNetB0 model that has already been trained. It accepts an image of size IMAGE-SIZE as input and produces a tensor with the following properties: batch-size, num-patches, and hidden-dim. The input picture is divided into the specified number of patches, and the output tensor is flattened into a 2D tensor with the shape (batch-size, num-patches * hidden-dim), where hidden-dim is the hidden dimension of the output feature maps. The decoder then receives this tensor.

With the help of the features the encoder extracted, the decoder creates a caption. A series of tokens representing the words in the caption serve as the decoder's input. A learnable embedding layer is used to first embed each token into an embed-dim-dimensional vector. The embeddings are then given a positional encoding to inform the model of the order of the words in the sequence. A stack of num-layers is then fed the resulting vector sequence. blocks for decoding transformers.

A self-attention layer and a feedforward neural network layer are the two sub-layers that make up each Transformer decoder block in the stack. The self-attention layer uses the output of the preceding layer and applies multi-head attention, paying attention to various sequence points to identify various word relationships. The outputs of the self-attention layer are then subjected to a non-linear transformation by the feedforward neural network layer. Before being fed into the following layer, the output of the feedforward layer is first routed via another layer for normalisation and residual connection.

One vector each token in the input sequence serves as the decoder's final output. A dense layer with a softmax activation is applied to each of these vectors in order to produce a probability distribution across the vocabulary. The projected next word in the caption is determined by selecting the word with the highest likelihood. Starting with a unique start token and continuing until an end token is generated or a maximum sequence length is achieved, the decoder generates the sequence one word at a time. Overall, this code defines a powerful model architecture that can generate captions for images, and it is designed to be trained end-to-end using

the Adam optimizer and the categorical cross-entropy loss function.

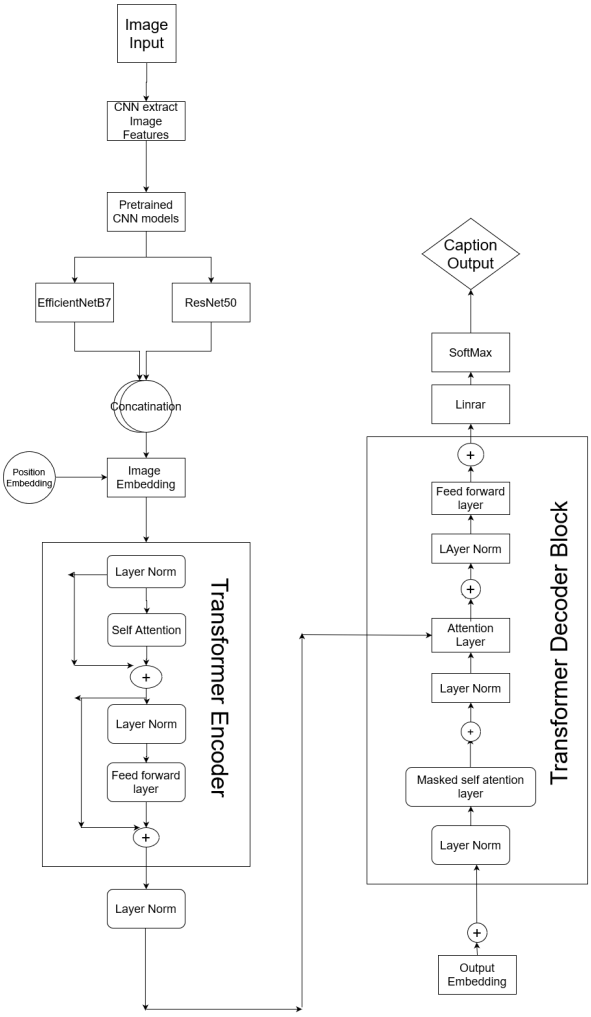


Fig. 2. Proposed Model Architecture

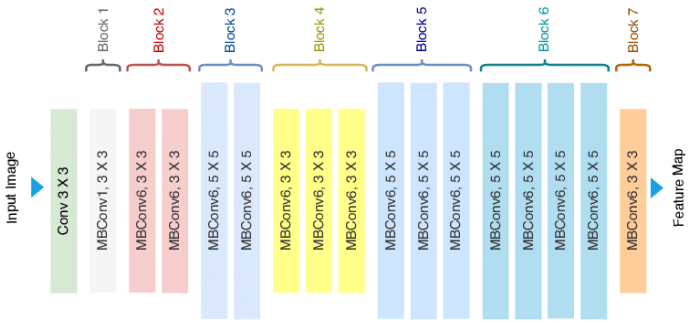


Fig. 3. EfficientNetB0 architecture

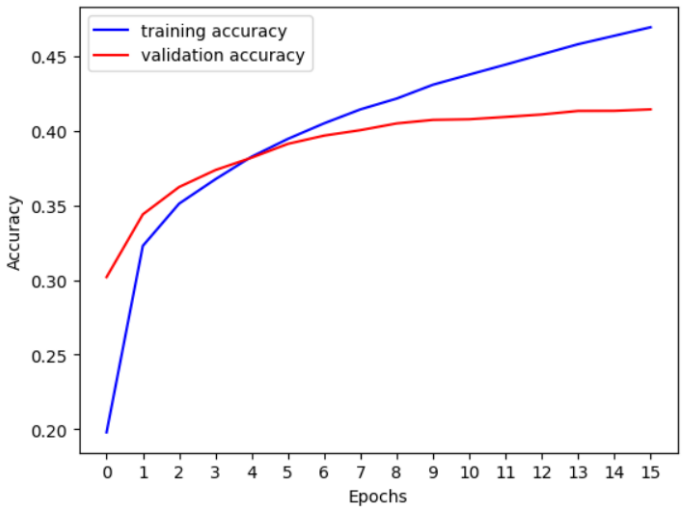


Fig. 4. Accuracy

IV. Results

A. *EfficientNetB0*

EfficientNet B0 is a deep convolutional neural network model that is typically used for image classification tasks. The model takes an image as input and outputs a probability distribution over a set of pre-defined classes.

B. *EfficientNetB7 and ResNet152*

The ”get-cnn-model” method defines a convolutional neural network (CNN) model using EfficientNetB7 and ResNet152 as feature extractors, two previously trained models. The model’s input layer has a form that corresponds to the dimensions of the input image. The output layers of the pre-trained models are concatenated with their weights from the ImageNet dataset. The output is then modified to take on a 3D shape. The output from the two pre-trained models is combined with the input layer to form a new model.

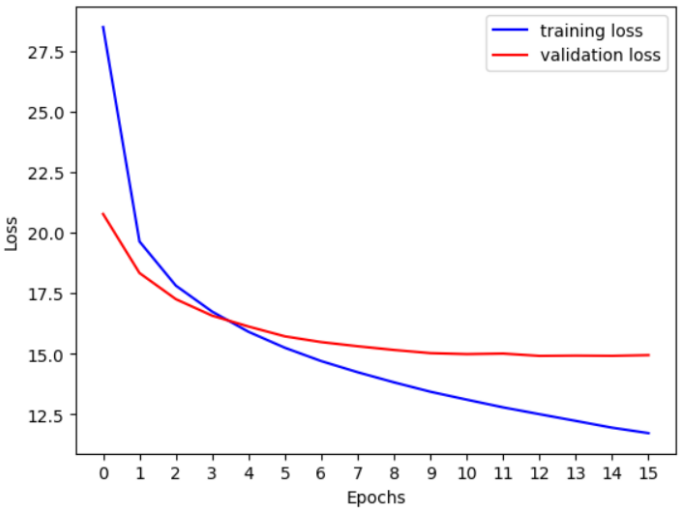


Fig. 5. Losses

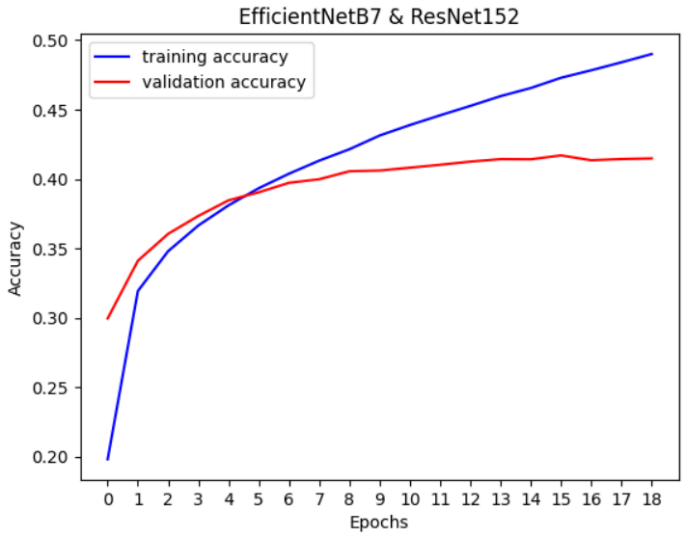


Fig. 6. Accuracy

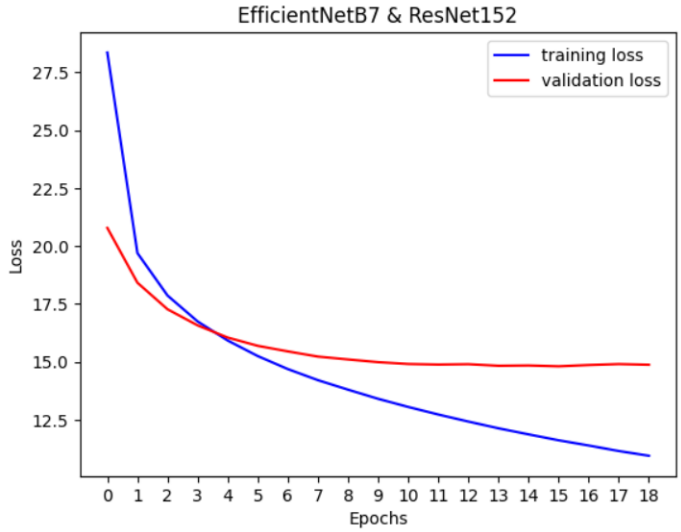


Fig. 7. Losses

V. Predictions

Conclusion

In conclusion, the suggested method defines a neural network model for image captioning based on the transformer architecture. The model comprises of a transformer decoder block to produce the captions and a convolutional neural network (CNN) to extract features from the image. Multiple transformer encoder blocks, each with multiple heads of attention and feed-forward layers with layer normalisation, make up the transformer decoder block. A positional embedding layer is also included in the decoder block to take care of the sequence's ordering. The method specifies the model architecture as well as the layers that make up the model, such as the positional embedding layer and the blocks for the transformer encoder and decoder.

Acknowledgment

I would like to express my heartfelt gratitude to all those who have helped me in completing this project on Image Captioning Firstly, I would like to thank Respected Faculty Dr. M. Srinivas and Dr. P. Radha Krishna and P.hD Scholar Vishnu sir and Murukessan sir for their invaluable guidance and support. Their expertise and encouragement have been crucial in shaping the direction of this project.
I would also like to extend my sincere thanks to all the experts and professionals who have provided me with their insights and knowledge related to Deep learning. Their inputs have helped me gain a deeper understanding of this subject.
I am also grateful to my friends and colleagues who have supported me throughout this project. Their suggestions and feedback have been invaluable in improving the quality of this Project.
I would like to acknowledge the support and resources provided by my institution NIT Warangal, without which this project would not have been possible.
Lastly, I would like to thank my family for their unwavering support and encouragement throughout this project. Their love and motivation have kept me going during the tough times. Once again, I would like to express my gratitude to all those who have helped me in completing this project on Deep learning.

References

[1] "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention" by Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho,

Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2016.

[2] "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention" by Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2016.

[3] "Image Captioning with Transformer" by Xiaojun Wan, Jianxiong Yin. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[4] "Unicoder-VL: A Universal Encoder for Vision and Language by Li Dong, Nan Ding, Xiaodong Liu, Ting Yao, Yuejian Fang, Zhenhua Ling, Daxin Jiang, Wael AbdAlmageed. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.