Ain Shams University

Faculty of Engineering

Computer Engineering Dept.

Course No. CSE (616)

Neural Networks and Their Applications

Project

By: Mahmoud Aboud Mohammad Nada

Code: 2002387

Supervised by: Dr. Hazem Abbas

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

General Explanation

Problem understanding

Image Captioning is the process of generating a textual description for given images. It has been a very important and fundamental task in the Deep Learning domain. Image captioning has a huge amount of applications. NVIDIA is using image captioning technologies to create an application to help people who have low or no eyesight.
Several approaches have been made to solve the task. One of the most notable works has been put forward by Andrej Karpathy, Director of AI, Tesla in his Ph.D. at Standford.

Image captioning can be regarded as an end-to-end Sequence to Sequence problem, as it converts images, which are regarded as a sequence of pixels to a sequence of words. For this purpose, we need to process both the language or statements and the images, the result should be as the figure below.

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

Approach Overview

There are multiple approaches to building an image captioning model, all of them must have a feature extraction part which is done by any convolutional neural network, but they vary in the language model part, which is the part that learns how to understand sentences and generate them, some use LSTMs but I'd rather use Transformers as they have shown much power in the latest SOTAs.

Our image captioning architecture consists of three models:

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

1. A CNN: used to extract the image features
2. A TransformerEncoder: The extracted image features are then passed to a Transformer based encoder that generates a new representation of the inputs
3. A TransformerDecoder: This model takes the encoder output and the text data (sequences) as inputs and tries to learn to generate the caption.

Model Evaluation Overview

To measure how our model is performing we will use a Metric called the BLEU score, BLEU stands for Bilingual Evaluation Understudy. It is a metric for evaluating a generated sentence to a reference sentence. The perfect match is 1.0 and a perfect mismatch is 0.0.

Dataset

Overview

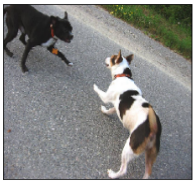
Flicker8K is a new benchmark collection for sentence-based image description and search, consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events.

The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations.

In the figure below we can see how the dataset is constructed and how each image maps to different captions, for training, we split the data for 80% training and 20% testing.

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

a little girl in a pink dress going into a wooden cabin .
a little girl climbing the stairs to her playhouse .
a little girl climbing into a wooden playhouse .
a girl going into a wooden building .
a child in a pink dress is climbing up a set of stairs in an entry way .



two dogs on pavement moving toward each other .
two dogs of different breeds looking at each other on the road .
a black dog and a white dog with brown spots are staring at each other in the street .
a black dog and a tri-colored dog playing with each other on the road .
a black dog and a spotted dog are fighting

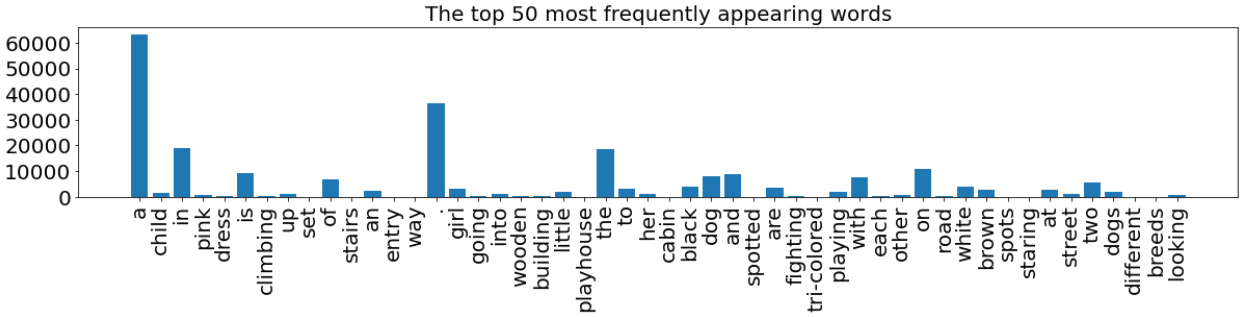
young girl with pigtails painting outside in the grass .
there is a girl with pigtails sitting in front of a rainbow painting .
a small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow on it .
a little girl is sitting in front of a large painted rainbow .
a little girl covered in paint sits in front of a painted rainbow with her hands in a bowl .

man laying on bench holding leash of dog sitting on ground
a shirtless man lies on a park bench with his dog .
a man sleeping on a bench outside with a white and black dog sitting next to him .
a man lays on the bench to which a white dog is also tied .
a man lays on a bench while his dog sits by him .

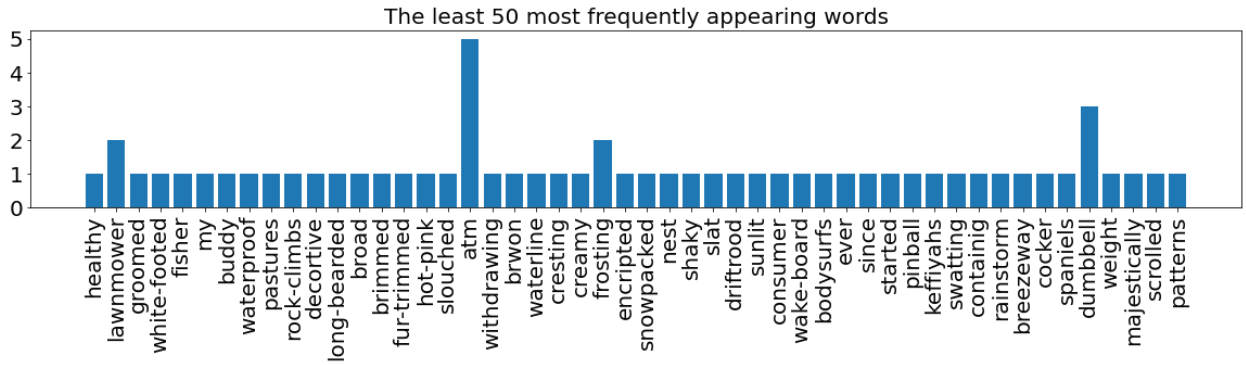
the man with pierced ears is wearing glasses and an orange hat .
a man with glasses is wearing a beer can crocheted hat .
a man with gauges and glasses is wearing a blitz hat .
a man wears an orange hat and glasses .
a man in an orange hat starring at something .

Statistics

We can see that in 40,000 sentences unique word counts are only 8918 (very less). Most of the words are repeated in all the sentences



By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas



Data Preprocessing

To organize and make data ready for the next phase, I built a dictionary where each key is an image path and its values are the caption corresponding to it, also I excluded an image if it has a caption that consists of tokens less than 5, for each token we put a start token at the beginning and end token at the end.

After building this dictionary, I shuffled the keys which are image names, and split them into 80% for training and 20% for testing, those are 6114 training images and 1529 testing images.

Then we want to vectorize the text data, that is to say, to turn the original strings into integer sequences where each integer represents the index of a word in a vocabulary. We will use a custom string standardization scheme (strip punctuation characters except < and >) and the default splitting scheme (split on whitespace).

I also wanted to increase the number of images in the dataset so I added an augmentation layer that only does 3 operations on any image randomly (horizontal flipping, random rotation, and random change in image contrast).

The rest is not much I didn't change that much in the image itself, just reading and resizing it to 299 x 299.
Now the data is ready to move into the next phase which is the training.

Network Architecture

My image captioning architecture consists of three models as I mentioned above and ill explain each part in detail, these models can be split into two parts, Feature Extraction model, Language generation model

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

CNN (feature extractor)

I used a pre-trained network to extract features, this saved me lots of time and computation, the used pre-trained network was Efficientnet as it has proven its amazing capabilities on the latest SOTAs, it's simple I just run the image through the Efficientnet without its final classification layer to get features that represent the image, these features are then handed over the next part (Transformer) to build sentences with.

Language Generation

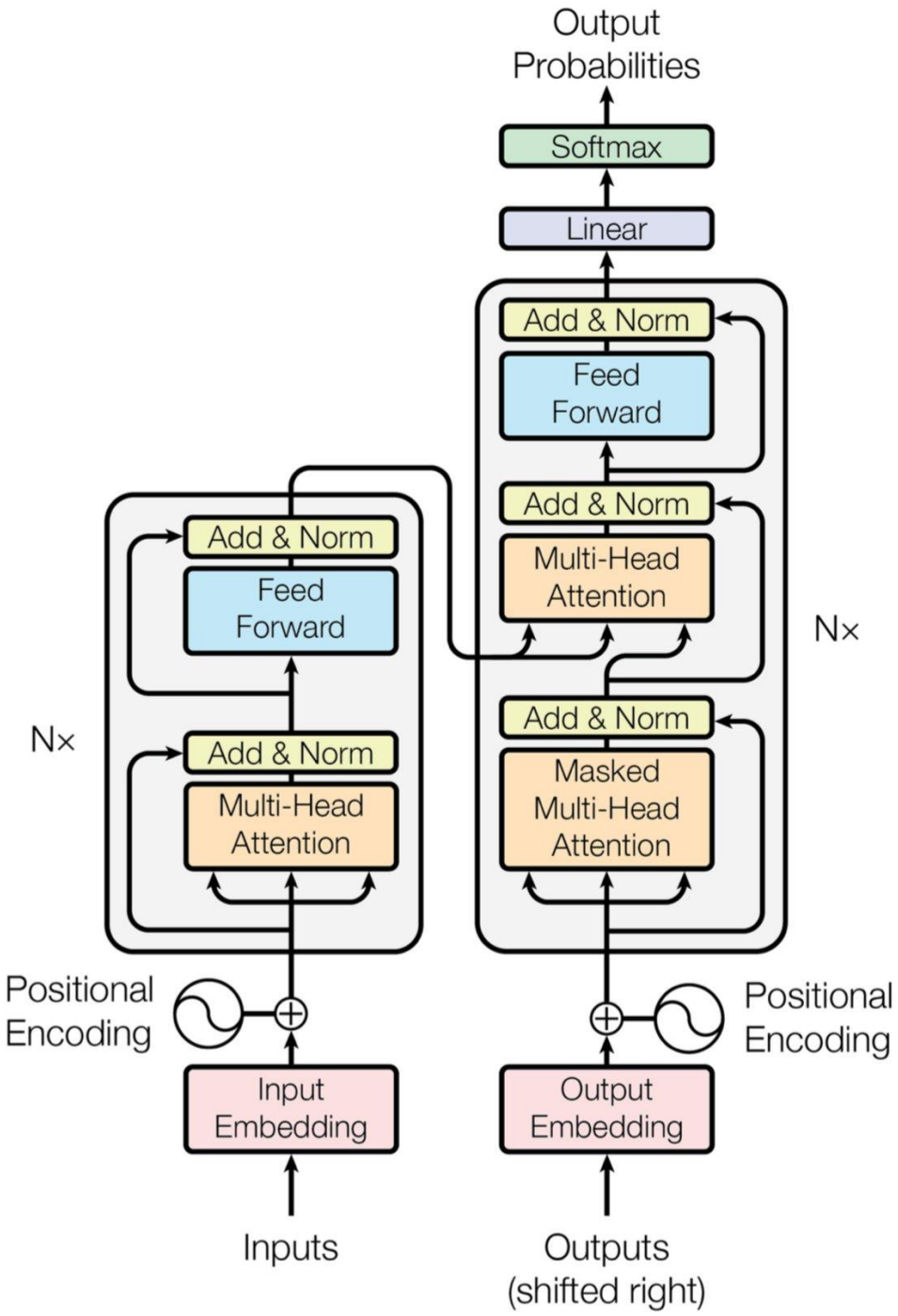
For language generation, we had two options, RNNs or Transformers since RNNs are slower to train and lately were outperformed by Transformers. I wanted to go with transformers.

A transformer model handles variable-sized input using stacks of self-attention layers instead of RNNs or CNNs. This general architecture has a number of advantages:

- It makes no assumptions about the temporal/spatial relationships across the data. This is ideal for processing a set of objects (for example, StarCraft units).
- Layer outputs can be calculated in parallel, instead of in a series like an RNN.
- Distant items can affect each other's output without passing through many RNN-steps, or convolution layers (see Scene Memory Transformer for example).
- It can learn long-range dependencies. This is a challenge in many sequence tasks.

The downsides of this architecture are:

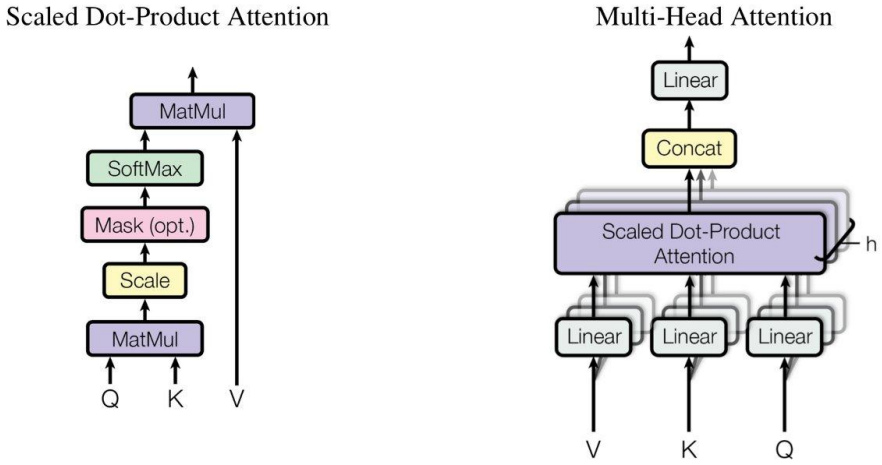
- For a time series, the output for a time step is calculated from the entire history instead of only the inputs and current hidden state. This may be less efficient.
- If the input does have a temporal/spatial relationship, like text, some positional encoding must be added or the model will effectively see a bag of words.



The Encoder is on the left and the Decoder is on the right. Both Encoder and Decoder are composed of modules that can be stacked on top of each other multiple times, which is described by Nx in the figure. We see that the modules consist mainly of Multi-Head Attention and Feed Forward layers. The inputs and outputs (target sentences) are first embedded into an n-dimensional space since we cannot use strings directly.

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

One slight but important part of the model is the positional encoding of the different words. Since we have no recurrent networks that can remember how sequences are fed into a model, we need to somehow give every word/part in our sequence a relative position since a sequence depends on the order of its elements. These positions are added to the embedded representation (n-dimensional vector) of each word.
Let's have a closer look at these Multi-Head Attention bricks in the model:



Let's start with the left description of the attention mechanism. It's not very complicated and can be described by the following equation:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Q is a matrix that contains the query (vector representation of one word in the sequence), K is all the keys (vector representations of all the words in the sequence) and V is the values, which are again the vector representations of all the words in the sequence. For the encoder and decoder, multi-head attention modules, V consists of the same word sequence as Q. However, for the attention module that is taking into account the encoder and the decoder sequences, V is different from the sequence represented by Q.
To simplify this a little bit, we could say that the values in V are multiplied and summed with some attention-weights a, where our weights are defined by:

$$a = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

This means that the weights are defined by how each word of the sequence (represented by Q) is influenced by all the other words in the sequence (represented by K). Additionally, the SoftMax function is applied to the weights to have a distribution between 0 and 1. Those weights are then applied to all the words in the sequence that are introduced in V (same vectors as Q for encoder and decoder but different for the module that has encoder and decoder inputs).
The right-hand picture describes how this attention mechanism can be parallelized into multiple mechanisms that can be used side by side. The attention mechanism is repeated multiple times with linear projections of Q, K, and V. This allows the system to learn from different representations of Q, K, and V, which is beneficial to the model. These linear representations are done by multiplying Q, K, and V by weight matrices W that are learned during the training. Those matrices Q, K, and V are different for each position of the attention modules in the structure depending on whether they are in the encoder, decoder or in-between encoder and decoder. The reason is that we want to attend to either the whole encoder input sequence or a part of the decoder input sequence. The multi-head attention module that connects the encoder and decoder will make sure that the encoder input-sequence is taken into account together with the decoder input-sequence up to a given position.

After the multi-attention heads in both the encoder and decoder, we have a pointwise feed-forward layer. This little feed-forward network has identical parameters for each position, which can be described as a separate, identical linear transformation of each element from the given sequence.

So in summary we have an encoder that generates a new representation of the inputs and a decoder that tries to learn to generate the caption.

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

Training / Testing Loss & Accuracy

In training we know that there are 5 captions for each image, so we use the BLEU score with n-grams = 5, for each real caption we calculate the matches and

In training we use a simple form of calculating these metrics, we just see if the words in the predicted caption are in the real caption or not and calculate our loss and accuracy accordingly.

Hyperparameters

Optimizer, other hyperparameters

I used Adam optimizer with learning rate scheduler and warmup learning rate that is used for the first small number of epochs than the normal 0.0001 learning rate is used and scheduled. For the other hyperparameters:

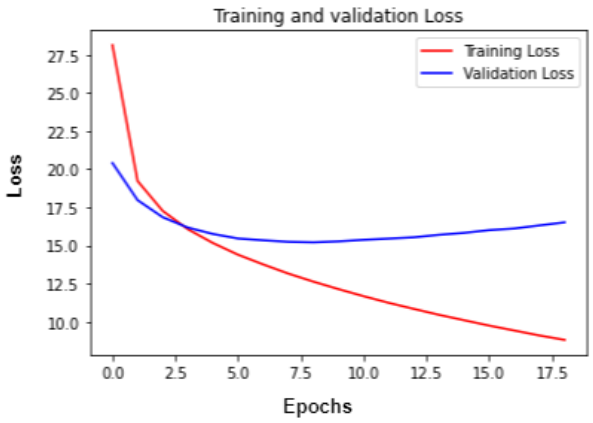
- Vocabulary size = 10000
- Caption sequence length = 25
- Tokens embedding dimensions= 512
- # Other training parameters
- Training batch size= 64
- Epochs = 100

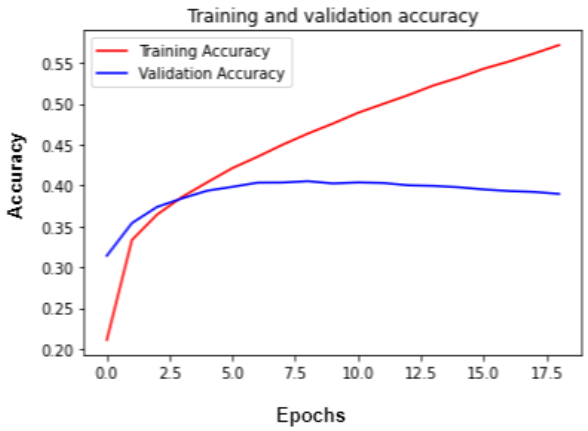
Training

For each training step, we open the batch images and extract features from them by the EfficientNet, then we pass these extracted features to the transformer encoder that generates a new representation of the inputs, these representations are delivered to the transformer decoder to generate in one shot the captions, in testing it's different because we generate caption first caption word then use it as an input again to get next word.

Due to the early stop that was implemented the model stops training after 10 epochs if the loss is not getting lower, you can see the training process below.

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas



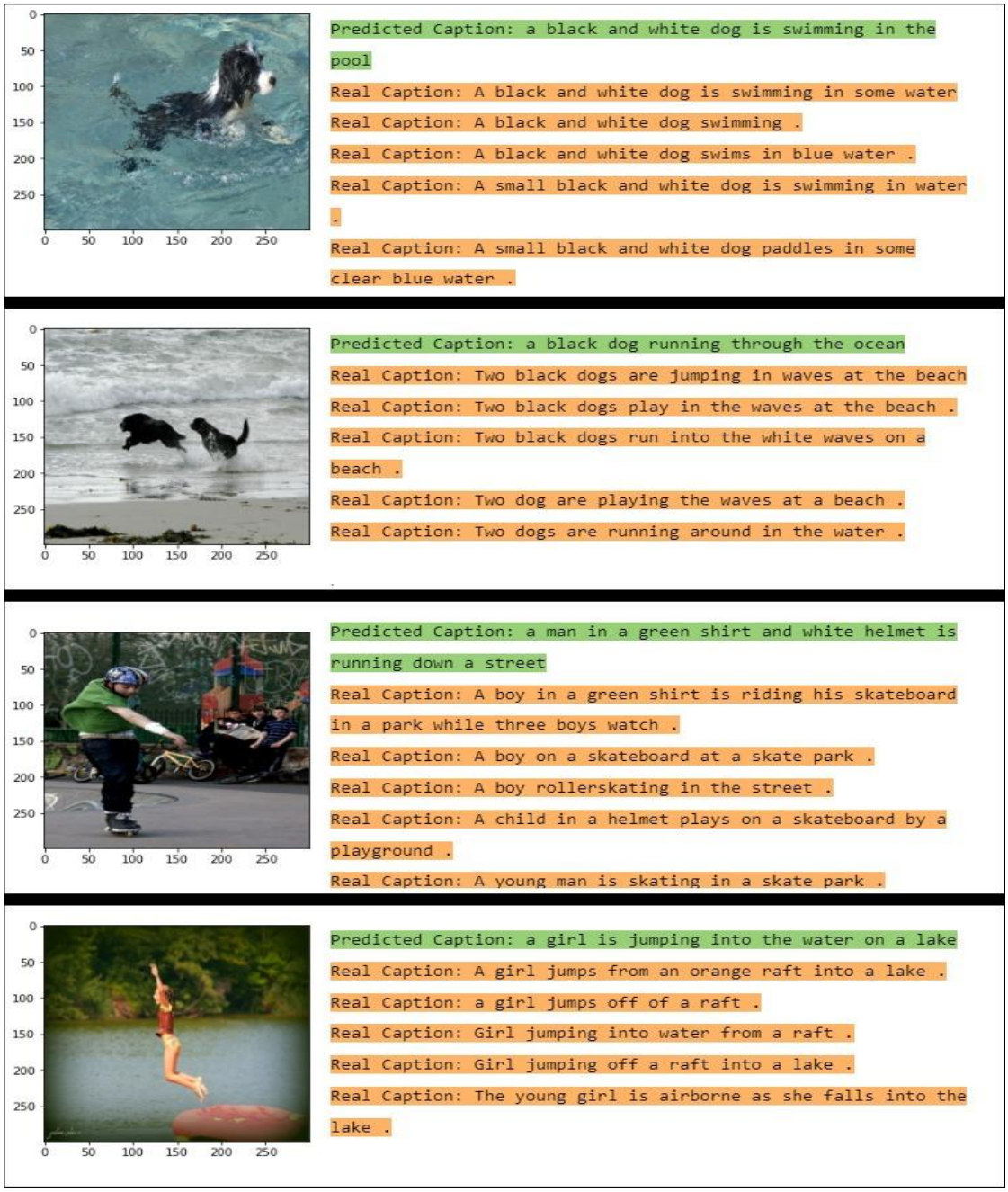


By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

Conclusion

Results

My model scored a BLEU score of around 0.5 and here are some results.



By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas

Future Work

- We can use different features extractor like Resnet.
- We can also use more attention heads
- We can try LSTMs and compare the results.
- We can use more augmentation or get more data to overcome the overfitting

References

- https://keras.io/examples/vision/image_captioning/
- https://arxiv.org/abs/1706.03762
- https://www.youtube.com/watch?v=iDulhoQ2pro
- https://www.youtube.com/watch?v=FWFA4DGuzSc
- https://www.youtube.com/watch?v=66seIToeguE
- https://www.tensorflow.org/text/tutorials/transformer#decoder_layer
- https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04?utm_source=pocket_mylist

By: Mahmoud Aboud Mohammad Nada
Code: 2002387
Supervised by: Dr Hazem Abbas