Image Captioning

Ahmed Ibrahim & Amr Gouhar

Dr. Mohamed Mostafa

American University in Cairo

Problem Statement

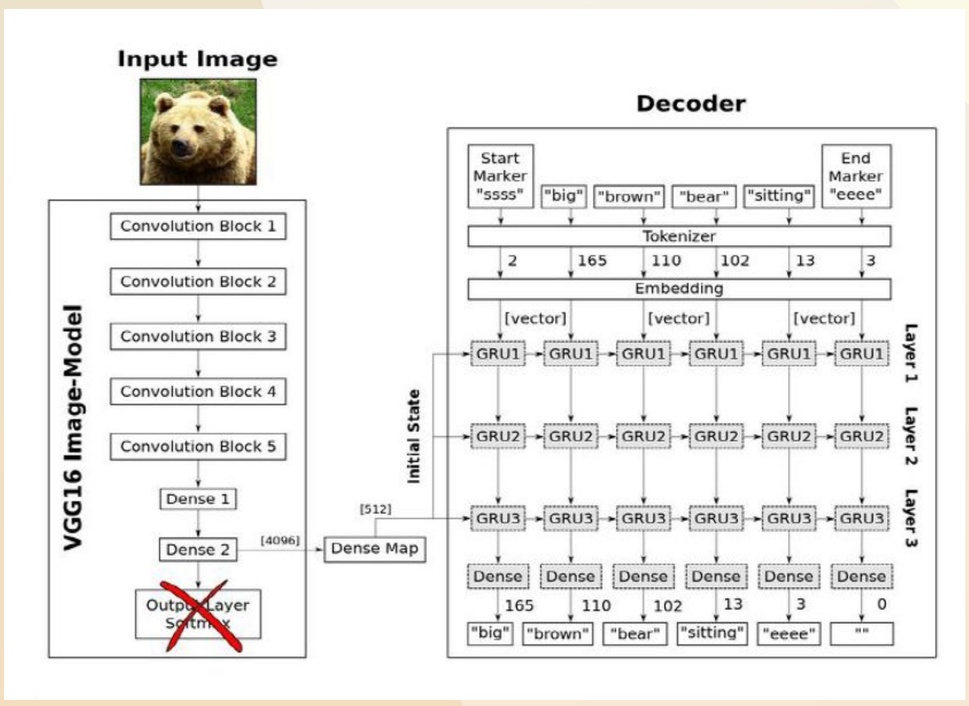
Creating meaningful captions for Images which would aid the visually impaired people to visualize the world.

Introduction

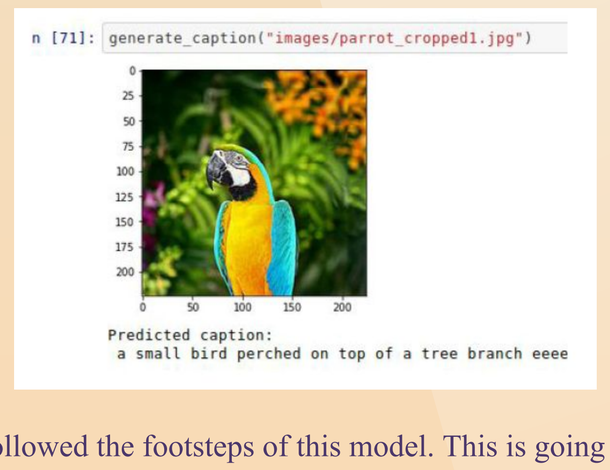
The problem we were trying to tackle using a machine learning model in this project is image captioning; that is, given an image, the model should be able to generate a caption that best describes that image. To solve this problem, we used the COCO Dataset which contains many images for training, validation and testing. Our model is a neural network model that uses both convolutional neural networks and recurrent neural networks. The details for the model will be explained later.

Literature Review

This problem of image captioning is not new. Many people tried working on it. One of the proposed solutions was using the convolutional network model (VGG16) along with three layers of recurrent units (GRUs). The following figure shows the architecture for that model.



The accuracy of that model was not provided; however, it gave pretty good predictions for pictures that did not belong to the training set as it is shown in the following figure.



In our project, we followed the footsteps of this model. This is going to be explained in the experiment section.

Objective

Achieving a simpler network that has higher accuracy than the already made ones.

Limitations

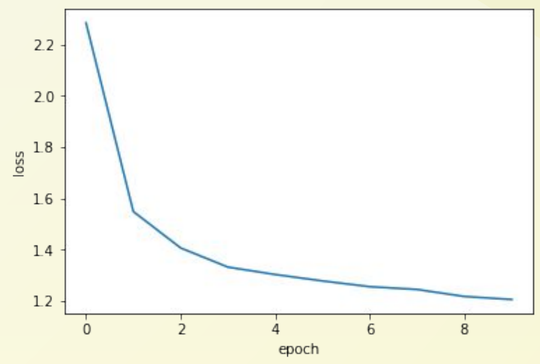
Low computational power and insufficient resources. Constant Kernal Panics despite using a batch size of 20 eventually, and some electricity cuts that required resetting the experiment several times and yielded very low results at the end.

We did not have much time to try more models and also we there were technical issues with the machines and so the results are not the best since the working conditions were not the best.

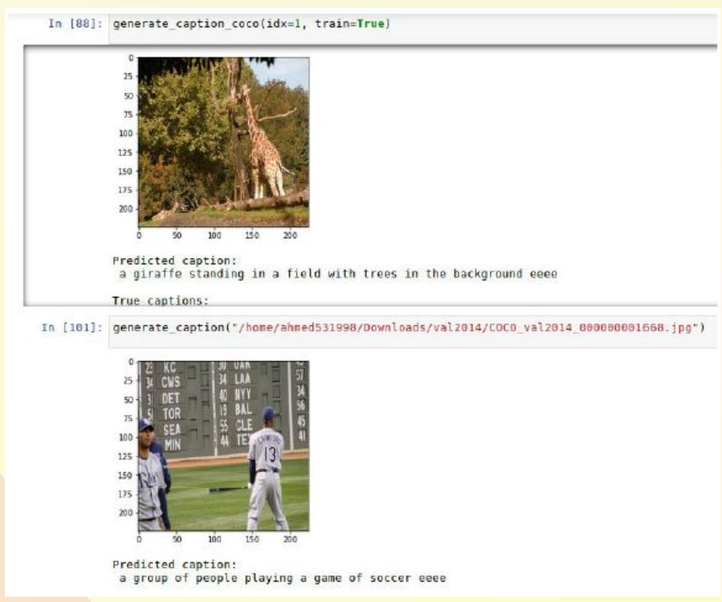
Experiment and Results

Our model consisted of the following: the convolutional network and the recurrent network. For the convolutional network, we used the already trained VGG16 model and so we did not train that. We only extracted the features from the final fully connected layer and fed that into our recurrent model. Based on research, we found that using more than three recurrent layers does not help with the accuracy and it just causes an overfitting and so in the models that we tried, we did not use more than three recurrent layers.

As for the hyperparameters, we just used a learning rate of 1e-3 which is, based on practice and convention, a suitable learning rate. We used it with the RMSProp optimizer. We also used the sparse cross entropy loss to calculate the loss.
For the recurrent layers, we started by 3 layers, each with 512 units, but we noticed that the loss started to decrease for around 3 epochs but then it started increasing for the remaining epochs. So, we decided to reduce it to 2 layers and also to reduce the number of units in each layer to 50. This gave us better results as it is shown in the following figure since the loss kept decreasing.



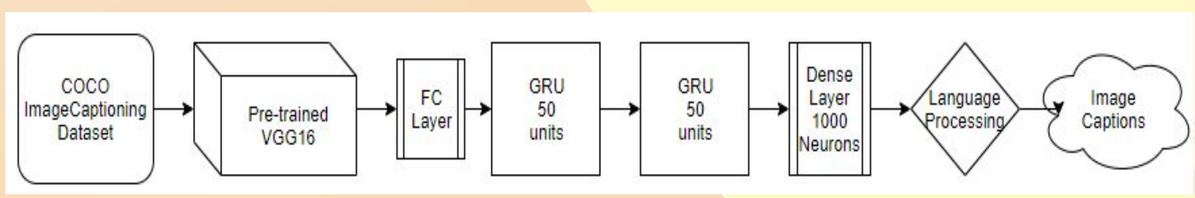
Therefore, we stuck to this model. Also, a sample prediction is shown in the following figure.



The score that we got when we submitted it was (0.1363).

Final Architecture

We used the pre-trained VGG16 as our CNN which we feed the input to, and then the output from the final FC layer is fed to our 2-layer GRU RNN with 50 units at each unit. The output is then fed to the Dense layer of 1000 neurons signifying the 1000 words used, then using language processing the caption is created and stored for submission.



Future Work

We'd suggest using the pre-trained ResNet instead of VGG16, as well as replacing all the FC layers with 1-D CNNs. Using an ensemble of 3 RNNs would definitely increase the resultant accuracy. Performing Data Augmentation on the input data by applying different simple filters to the data is also expected to increase the accuracy. However, we were not able to do any of these suggestions due to low computational power and insufficient resources.

References

https://github.com/tylin/coco-caption
https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/22_Image_Captioning.ipynb
https://github.com/amaiasalvador/imcap_keras
https://arxiv.org/pdf/1805.09137.pdf