Term Project Report: Spring 2021

Sajib Biswas, Debanjan Goswami

04.23.2021
Deep and Reinforcement Learning Fundamentals (CAP5619)

Introduction:

This project is an implementation of a pattern recognition technique. In particular, we want to generate captions for images. Generating proper caption of a given image is a problem that requires combined effort of computer vision and natural language processing. In this challenge, we are following an approach that employs deep recurrent neural networks which combines techniques from both computer vision and machine translation. Machine translation is used to transform a sentence written in source language to another target language step-by-step. Due to recent development, this can be done more efficiently by using Recurrent Neural Network's encoder-decoder architecture.

The ability to automatically describe the content of an image using grammatically correct English sentences is a daunting task, but it can have a great impact such as enabling visually impaired people to better understand the content of a webpage. The task of generating captions is more challenging than the commonly studied computer vision problem such as image classification and/or object recognition, since generating caption not only requires identifying the objects and environment in an image, but also revealing the relationships among the objects and the environment they are in. And, last but not the least, these relationships need to be expressed in a natural language i.e. English which adds another layer of challenge.

Motivation:

This project is heavily inspired by the work done by Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan on image captioning named as "Show and Tell: A Neural Image Caption Generator". The work we are doing here will be a basic and Simplified Version of the Show and Tell Model.

The Show and Tell model is a deep Recurrent Neural Network architecture which combines recent advances in computer vision and machine translation. In recent times, translation from one language to another can be done much more efficiently by using Recurrent Neural Network. Under this RNN's encoder-decoder architecture, the encoder takes in the source language and then transforms it into a rich fixed-length vector representation, which is then taken as the initial hidden state by the decoder and is converted to the target language.

However, in this project, the encoder RNN is being replaced by a deep convolutional neural network a.k.a CNN. In recent years, CNN has been extensively used for image classification and related tasks and it has been well-established that CNN can produce a rich representation of the given input image by embedding it to a fixed-length vector, which can then be used as the input to the RNN decoder, which can later generate sentences describing the content of the image.

Technical Details:

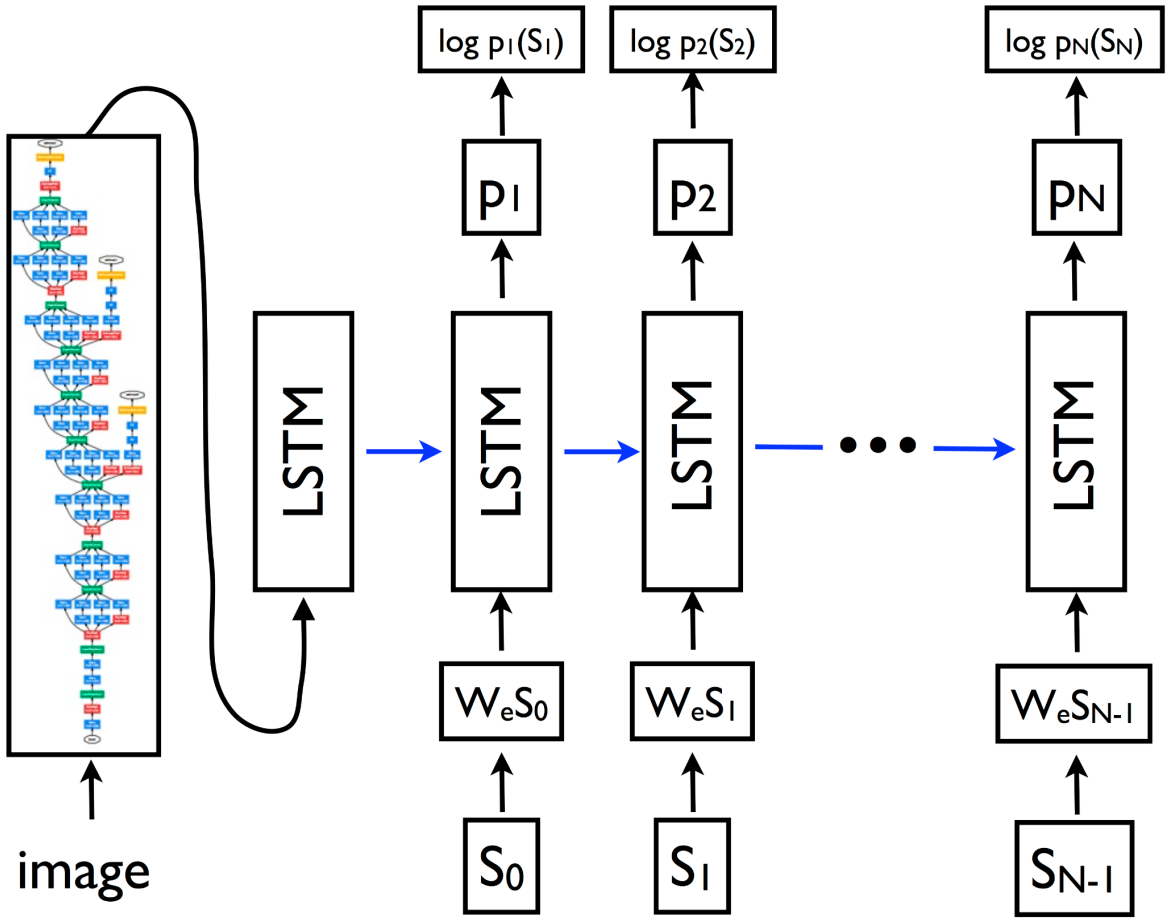
In a nutshell, in this model deep convolutional nets for image classification are combined with recurrent networks for sequence modeling, to create a single network that generates descriptions of given images. In this section, we are going to dive into the details of the networks architecture we are going to use for this problem.

As already mentioned before, the Show and Tell model is an example of an encoder-decoder neural network. It works by first "encoding" an image into a fixed-length vector representation, and then "decoding" the representation into a natural language description.

The image encoder is a deep convolutional neural network. This type of network is widely used for image tasks and is currently state-of-the-art for object recognition and detection.

The decoder is a long short-term memory (LSTM) network. This type of network is commonly used for sequence modeling tasks such as language modeling and machine translation. In the Show and Tell model, the LSTM network is trained as a language model conditioned on the image encoding.

Words in the captions are represented with an embedding model. Each word in the vocabulary is associated with a fixed-length vector representation that is learned during training. The following diagram illustrates the model architecture.



In this diagram, $\{s_0, s_1, ..., s_{N-1}\}$ are the words of the caption and $\{w_e s_0, w_e s_1, ..., w_e s_{N-1}\}$ are their corresponding word embedding vectors. The outputs $\{p_1, p_2, ..., p_N\}$ of the LSTM are probability distributions generated by the model for the next word in the sentence. The terms $\{\log p_1(s_1), \log p_2(s_2), ..., \log p_N(s_N)\}$ are the log-likelihoods of the correct word at each step; the negated sum of these terms is the minimization objective of the model.

Models Used:

In this section, we are going to describe details about the two models we used for this experiment.

In the first model, which has been implemented using *Keras* libraries, we use the VGG16 model as the image encoder. This model consists of 13 convolutional layers with *3x3* kernels. The model is also able to classify scenes along with identifying the objects(people, dog etc.) in it. The model also uses intermediate fully-connected layers to further reduce the dimensionality of the image vectors, and to match the dimensionality with the word embedding vectors. Then, for the decoder we use an LSTM based model. The LSTM model is trained to predict each word of the sentence after it sees the image and all the preceding words. First, the VGG a.k.a CNN model extracts features of the given image and these features are transformed to a common embedding space as the words. Each word in the input caption is represented as a one-hot vector. The word embeddings are fed into the LSTM at different time steps. The output of the LSTM is connected to a fully-connected layer which has a hidden unit corresponding to every word in the vocabulary we provide. It can thus be viewed as a *k*-class classification problem. During testing time, we only feed the input images to the network and expect the model to generate corresponding captions for the image. To evaluate the performance of the LSTM as a language model, we sample the most relevant word at each time step of the LSTM network. Though there are some other methods such as Beam Search, in this model we only experiment with the Sampling method for generating captions.

For the second model, *Pytorch* is used to implement the network. Here, we use a pretrained ResNet-101 model which is readily available in the PyTorch module. We discarded the last two layers which are required for classification, since we use this CNN to only encode the image. We also fine-tune some convolutional blocks in the ResNet. There is also an Attention layer which is composed of only linear layers and a couple of activations. For the decoder, we use LSTM in PyTorch. In this model, we supply the ground-truth as the input at each decode-step during validation. This technique is called Teacher Forcing. This model also incorporates Beam Search. It reads an image, encodes it, and applies the layers in the Decoder in the correct order. At the same time it uses the previously generated word as the input to the LSTM at each timestep.

Goals:

The goal of this project is to test the previously described models with a popular dataset for an empirical study of the neural net architecture. We are going to compare between two different implementations of the image captioning model.

For this purpose, we are going to use the Flickr8k, Flickr30k, and MSCOCO Sentence Datasets. These datasets mostly consist of images and their description in English language. The captions are relatively visual and unbiased.

To achieve good results, given a trained model and an image we use beam search to generate captions for that image. We also use teacher forcing, which is a strategy for training recurrent neural networks that uses model output from a prior time step as an input.

To evaluate the results of the models, we use the BLEU score. This gives us an estimate of precision between the hypothesis caption and the reference caption. Usually, BLEU scores upto 4-grams are reported. BLUE is known to give captions that are more inline with a human interpreter, a better score. We are going to compare the BLEU scores given by the two different solutions of the Image Captioning problem.

Results:

Both of the models in this experiment are an example of an encoder-decoder neural network.

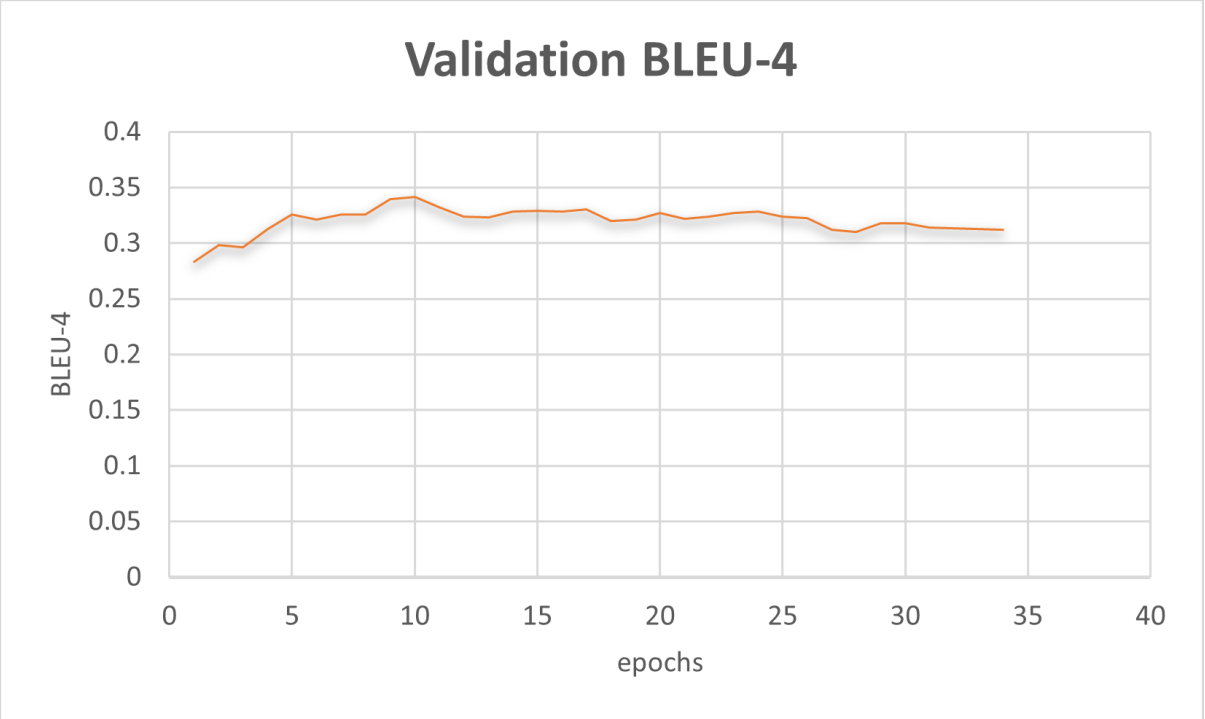
For the Keras-based model we listed the BLEU-4 scores against Flickr8k, and Flickr30k datasets in the following table.

DataSet	BLEU-4 Score
Flickr8k	14.7
Flickr30k	16.7

For the PyTorch-based model, which is used on the MSCOCO dataset, we are using Beam Search. By variating the size of Beam, we get the following values of BLEU-4.

Beam Size	Validation BLEU-4
1	29.98
3	32.95
5	33.17

For this model, the evolution of BLEU score along with the number of epochs can be seen from the following graph.



Conclusion:

In this project, we have conducted an empirical study of two Deep Learning based Image Caption Generation models and showed their comparative performances, by experimenting on the well known Flickr8k, Flickr30k, and MSCOCO datasets. From the BLEU-4 scores of the models, we can learn about their efficacy in generating image captions in languages understandable to humans.

References:

1) Show and Tell: A Neural Image Caption Generator
2) https://vision.cs.uiuc.edu/pascal-sentences/
3) Microsoft COCO: Common Objects in Context
4) https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/