Hangjun Piao (CS major, sophomore)
Hongyi Wang (CS major, junior)
Xiaofei Liu (CS major, junior)
Zirui Tao (CS major, junior)

Image Captioning with TensorFlow

ABSTRACT (1/4 page)

Image captioning involves both image processing and natural language processing tasks, which is a popular sub-topic in Machine Learning. Recent approaches to such problems often adopted the long short term memory (LSTM) neural network, a powerful recurrent neural network structure for memory-based learning, first proposed by Kalchbrenner et. al.[1]. For the purpose of studying LSTMs structures and its applications, we implemented the such image captioning task with appropriate reference to the author of the paper "Show and Tell: Lessons learned from the 2015 MicroSoft COCO (MSCOCO) Image Captioning Challenge", Vinyals et. al.[2]. Finally we also presented our experiment on several parameters regarding to refining the existing LSTM structures and settings.

INTRODUCTION

In this project we aimed to perform the image captioning task, that is, given an image and encoded word inputs, generate a description of the content of the image. Natural language processing and image recognition have seen several breakthroughs in terms of the accuracy and performance during the last few years. Our team, consist of four undergraduates, lack of research experience , found it too ambitious to start a new research topic, set the goal of learning more about different kinds of neural networks. As a result we decided to implement existing model with good results and get more hands on experience of implementing neural network using modern technology such as the tensor flow library and the cloud services. The final choice of image captioning consists of both image processing and natural language problem. Another important reason for us to chose this

task is the usage of LSTM. Due to the nature of the LSTM architecture, it is better suited for learning time series data compared to traditional RNN structures. Regarding the approach of our project, most of the work in this project was inspired by the work done by Vinyals et. al.[2]. Although we were well guided by the paper, we faced some difficulties. One big obstacle is that unlike common LSTM applications, the initial state of our network came from the output of the CNN group and that caused complications during the training process as the state and the training captions needed to be synchronous. During the process, we tried different settings of the parameters and the structures of the network. For all experiments, the total cross entropy in the model converges to 2 with a rather fast speed. However, despite all the effort to increase the performance of the network, all the networks seems to converge to around same number no matter the number of epoch is.

PROBLEM DEFINITION AND ALGORITHM

The picture above shows the architecture of the model used for image captioning task. The model is combination of a CNN and a LSTM neural network. In terms of image representation, we feed in an image into a pre-trained convolutional neural network and extract the feature vectors from the last fully connected layer as an initial input to the LSTM network. The feature vector is 1000x1 and is linearly transformed to the size of hidden dimension of LSTM cell, which is 512x1.

As for caption generation, we represent the input captions as an array of word indexs (S_i), which are used to extract corresponding word embedding from word embedding matrix. In the training process, the captions in the training set are formatted with a <START> token and <END> token. The input to LSTM is a caption from S_0 (<START> token) to S_{n-1}. The output of each LSTM cell is linearly transformed to 1x|V| so that softmax function can be applied to calculate the probability for all words in the vocabulary. The cross entropy function is used to calculate the loss of each iteration and parameters are updated by stochastic gradient descents. In the evaluation phrase, the input to LSTM is the feature vector of an image and the <START> token. The word with highest probability is selected as the output of current LSTM cell and is used as the input

for next LSTM cell. The LSTM stops after the <END> token is generated. Because, in the training step, the LSTM model is trained to recognize the <END> token, it is certain that the model will generate an <END> token in the evaluation process.

Algorithm Definition (1/4 - 1 page)

The basic algorithm we used is the standard LSTM model introduced in class. One change we made was to connect the output of the previous layer as the input to the next layer, which helps to generate a full sentence that is grammatically correct. The other change we did is that we synchronized the initial state vector of the network using the output of pre trained CNNs, thus each picture is trained with all the captions related to it. We utilized the data preprocessing script provided by Stanford CS231n[3], for the other parts we mimicked the code written by Shallue[4] and the github repository maintained by Wang[5].

2.EXPERIMENTAL EVALUATION

2.1 METHODOLOGY AND RESULTS

In terms of evaluation of the model, we use BLEU matrix to evaluate the accuracy of captions generated by our model and use standard loss function to monitor the training process. The BLEU algorithm compares the correspondence between a reference sentence and a generated caption. It gives a score between 0 and 1 and outputs 1 only if two sentences have exact match. Therefore, even if BLEU outputs a low score, the generated sentence might still be syntactically reasonable. Below is our score on BLEU metric.

Bleu Score on 1147 test samples

BLEU:	4.63			
Precision x brevity:	4.88 x 94.92			
Type	1-gram	2-gram	3-gram	4-gram
Individual	38.21	8.52	2.34	0.74
Cumulative	36.27	17.12	8.67	4.63

As for experiments, we want to explore how different encoding methods of inputs and various models of LSTM will affect the accuracy of the model.

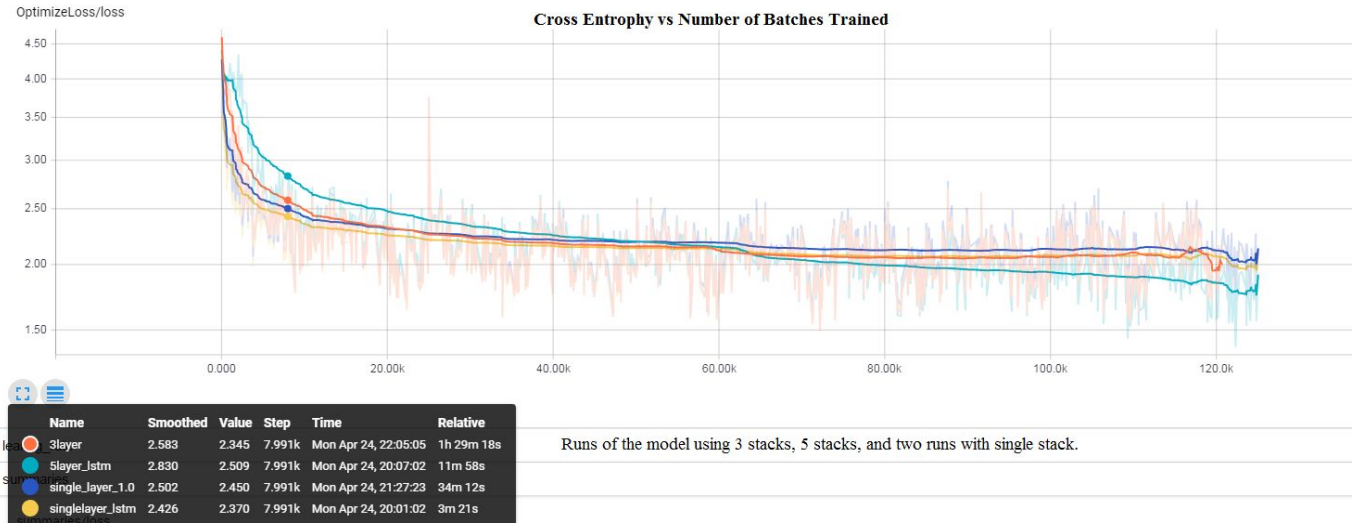
Hypothesis 1: Training word embedding matrix along with the program can result in higher accuracy than using the pre-trained word embedding matrix.

In comparison with Show and Tell model, Vinyals et. al. randomly initialized a word embedding matrix and updated the parameters of the matrix using backpropagation from calculated loss of correct captions, we trained a word embedding matrix using GloVe model solely based on the training captions without associated images. However, we believed that using a randomly initialized word embedding matrix will have higher accuracy because it can incorporate the representation of an image to the word representation during the learning process.

Hypothesis 2: A Multi-layer LSTM improves accuracy by providing additional level(s) of semantic information and can capture more efficient semantic representation of words better than a single layer LSTM.

Hypothesis 3: Choice of CNN would affect the training result. i.e. Using VGG-16 or AlexNet to represent images will have different performance.

Regarding retrieving the data set and deploying our work, our group utilized the pre-existing compressed representation of the dataset. The original MS COCO contained about 83000 pictures with each associated with 5 human-generated captions. After cleaning the dataset, we used the 82000 pictures in total. The whole dataset was about 40G after decompress and it became unrealistic to load the whole dataset into the memory. Instead, we pushed all the picture through the pre-trained CNN models and organized them in a relevance based order that was defined in the Stanford CS231n dataset[6]. The final dataset for VGG network was about 500 mbytes and the final dataset for Alexnet was about 800 mbytes. The training and validation (test) set separation followed the before mentioned Stanford dataset. Regarding the scalability, we deployed our network on both Amazon AWS, Google Computing Engine and personal computers with GPU instances.

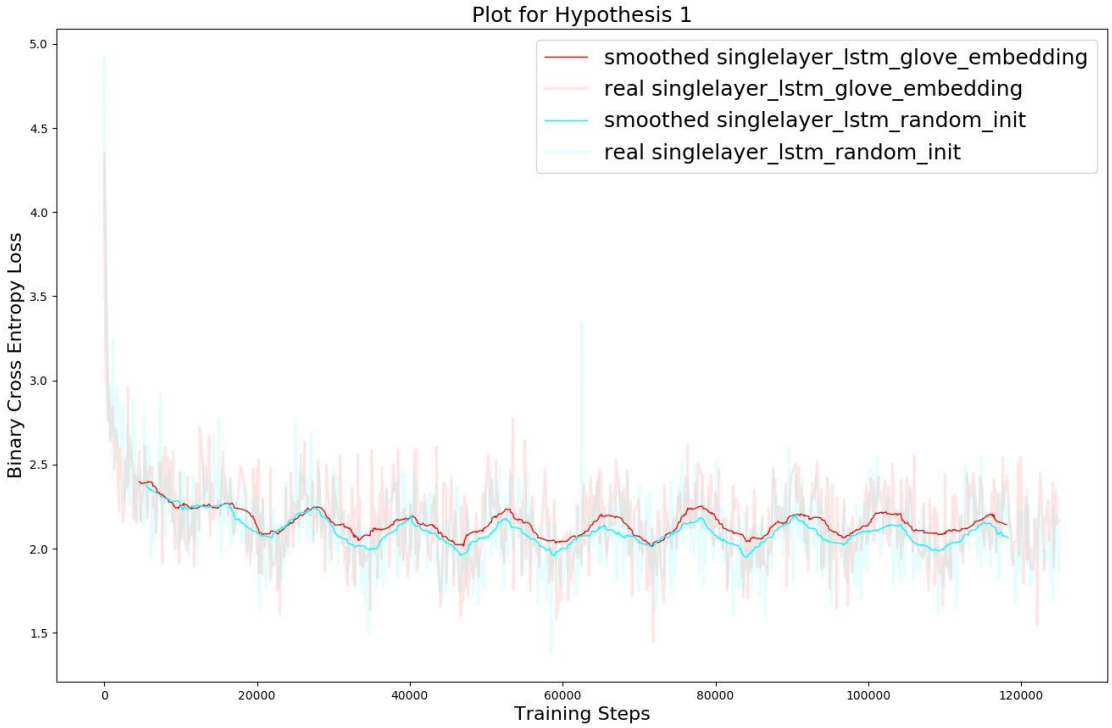


Runs of the model using 3 stacks, 5 stacks, and two runs with single stack.

2.2 DISCUSSION

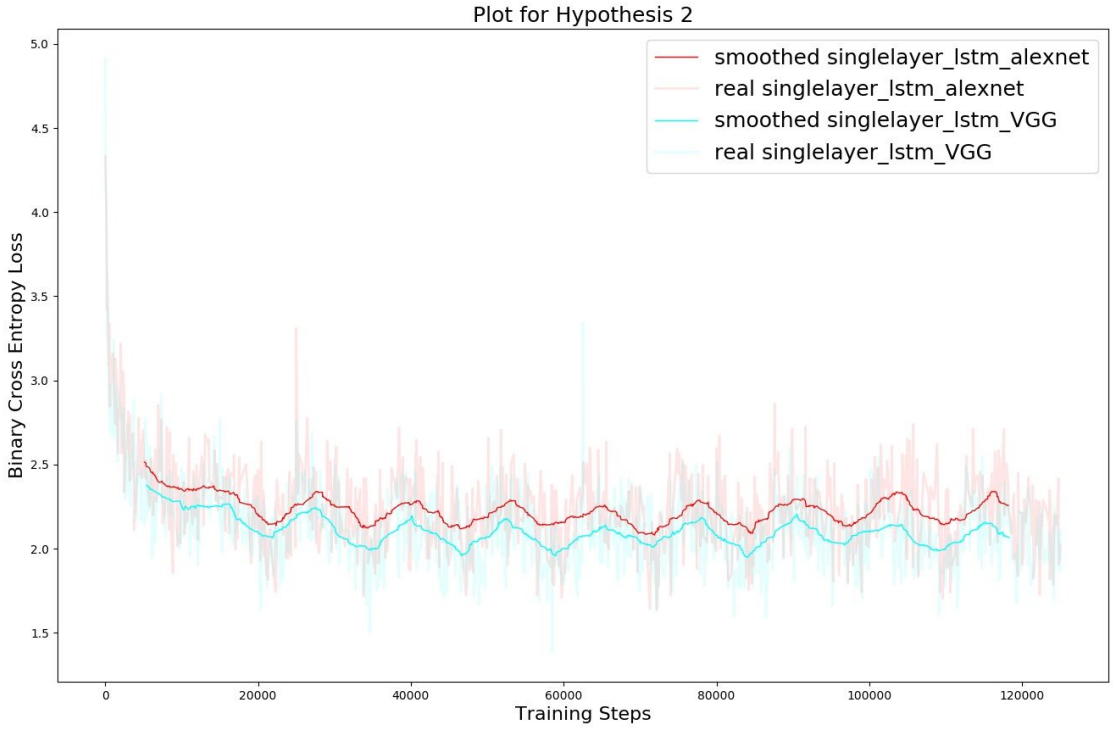
Our group presents the experiment result with plot in which we mainly applied moving average to analyze the trend of the curve and the moving average from range [0, window size] and range [end - window size, end] are truncated since it may skew results due to the insufficient data points in the sliding window. The y as binary entropy loss is plotted against x as global steps.

Regarding hypothesis 1, we compared the training result with two different word embeddings and presented the results as following:



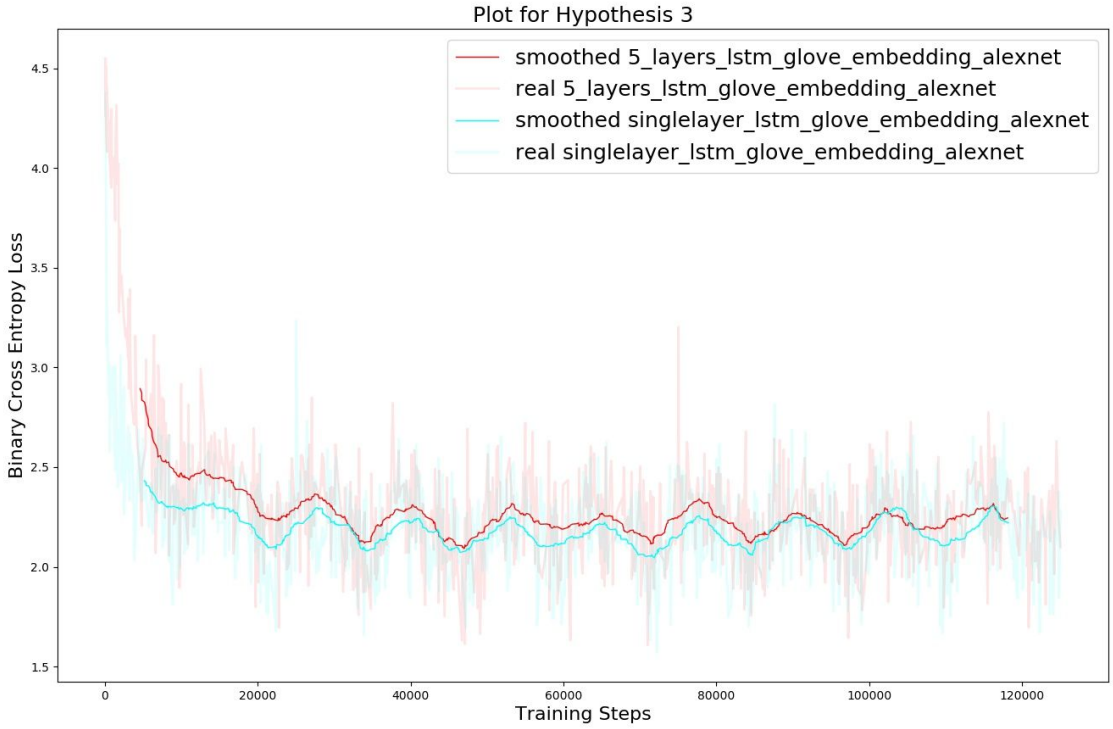
Regarding word embedding, we found among two word embedding Glove Embedding (pre-trained) had inferior performance than word embedding trained along with the image-captioning training process with t-test p value:7.596e-11. Therefore, we accepted our hypothesis that different training process, whether happend before or along with image-captioning training process, does affect the performance of the network.

For hypothesis 2, we also fixed all the parameter setting except vertical layers of LSTM and presented the results as following:



For hypothesis 3, We experimented two different CNN types with identical environment settings.

According to both the moving average plot and the t test p value with 3.444e-61 (almost 0), we accept our hypothesis that different CNN affects the performance of the network by comparing the average loss value.



For the last hypothesis, surprisingly, we found that single layer LSTM outperforms the multi-layer LSTM structure regarding average loss value. The t-test p value of 2.377e-29 also support the conclusion. Therefore, we reject our hypothesis and conclude that single layer LSTM would be more suitable for this image-captioning tasks (may need to further justify since the current setting of word embedding may vary with ones in future settings)

The strength is of our method is first, very indicative since we presented the result both through obvious visual representation and rigid statistic test results. However, the weakness of our method is that it may not generalize well since our experiment conclude only one environment settings (other parameter settings) and did not show the potential relationship between those fixed parameters and one of interests. However, we still believe that our metric for measuring prediction, BLEU, is not indicative enough to present our success of project. Since it only captures the syntactic result from predictions, evaluation for semantic meaning may be skewed and our result could be shown with strong support.

3. INDIVIDUAL CONTRIBUTIONS (1/4 page)

Hangjun Piao: Search for advanced topics about improvement on our current model, which could be an enhanced model that can be used to generate more accurate caption. Run new experiment of our enhanced model on both Amazon EC2 and Google computing engine. Guide the team on exploring even more advanced thoughts on how to improve the implementation of our enhanced model.

Hongyi Wang: Data preprocessing, word embedding, CNN connection, cloud deployment, change of network structure and parameters.

Xiaofei Liu: Build the model

Zirui Tao: Performed the experiments of parameters of LSTM structure and finalized the report.

4. RELATED WORK

The image caption task involves both computer vision and natural language processing. Vinyals et al. (2015) build an end-to-end learning model that transfers the result of a convolutional neural network to a LSTM network, which can generate a sentence of different length. Prior to this model, most models proposed by researchers involve an deep convolutional neural network to generate words to describe an image, which are pieced together based on sentence templates. Those works are specifically hand-designed to solve the sentence generation task. After Vinyals et al.'s work, various LSTM networks, such as multi-layer LSTM and bidirectional LSTM, are explored to discover various combination between LSTM and CNN.

5. FUTURE WORK

Our current model using a single feature vector might not fully capture information in a image and then generate an accurate caption. An enhanced model is to use attention. We can use multiple feature vectors to represent an image by sampling scratches from an image and using CNN to encode those scratches. Those feature vectors can be recursively applied to the LSTM to generate words to represent the corresponding scratches. This allows us to dynamically get image caption words generated precisely and at the same time preserve detailed information, which is supposed to be a better way in order to get a more descriptive sentence.

6.CONCLUSION

Our group successfully finished the image-captioning task and experimented several parameters regarding its effect on task performance. Some validation result such as printing the caption prediction on image and standard prediction accuracy metric such as BLEU was tested under our model. In experiment part, we proposed three hypotheses and answered them one by one based on realistic data results. We concluded that all of our mentioned three factors affect the prediction performance, with two of them agreeing with and one contrasting to our hypothesis statement. We also explored the advantage and disadvantage of the cloud platform deployment. Therefore, we successfully finished the thorough theoretical study and concrete application of LSTM neural network. Based on this experiment, our group also provided constructive suggestions on potential improvement on selecting task environment (CNN, word embedding and vertical layer number). However, the result we got incites some further questions such as why is every network all converge to similar value and the converge speed seems to be identical. These questions might lead to further discovery and learning about the combination of CNN and LSTM. Here in the end of our report, we want to thank the professor for the fantastic semester and wish you happy retirement.

BIBLIOGRAPHY

1. Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

2. [Special issue]. (2016). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PP*(99). https://doi.org/10.1109/TPAMI.2016.2587640

3. CS231n assignment 3. (n.d.). Retrieved May 11, 2017, from http://cs231n.github.io/assignments2016/assignment3/

4. Shallu, C. (2017, March 18). Show and Tell: A Neural Image Caption Generator. Retrieved April 27, 2017, from GitHub Repository: https://github.com/tensorflow/models/tree/master/im2txt

5. Wang, W. (2017, March 30). Image Captioning Model in TensorFlow. Retrieved April 24, 2017, from GitHub Repository: https://github.com/aaxwaz/Image-Captioning-Model-in-TensorFlow

6. MSCOCO dataset. (n.d.). Retrieved April 20, 2017, from http://cs231n.stanford.edu/coco_captioning.zip