ASSIGNMENT 3
Visual Question Answering with PyTorch!

April 24, 2019

Name: Xinjia Yu
Andrew ID: xinjiay

Contents

0.1 Task 1: Data Loader

Q1 Give a summary of your dataset implementation. What should be in the __getitem__ dictionary? How should that information be encoded as Tensors? If you consider multiple options, explain your thought process in picking. What preprocessing should be done?

The data set is implemented based on the VQA API, which is provided in the problem description. Using the VQA function, we could easily find the index list of all questions and images. However, the question is in the sentence level, which is hard to be implemented in the network. Therefore a dictionary was created to collect the words in questions. Based on the dictionary I collected for questions, each question could be encoded to a multi-hot vector which had the same length of the word dictionary we collected. In __getitem__ dictionary, firstly, I collected and resized the image features using opencv library. And then I extracted out the multi-hot vectors of the questions. Meanwhile, the ground truth of the answers was also extracted from the VQA API. It is worth to mention that those three outputs should be be converted to tensors and then packed as a dictionary. For multiple options, I would pick the answer with most confidence as the ground truth. The most confidence could be defined as that the number of that 'correct' answer should has the maximum frequency showing in the answer dictionary. Therefore, we need to apply max-voting for the ansers firstly.

0.2 Task 2: Simple Baseline

Q2 Describe your implementation in brief, focusing on any design decisions you made: e.g what loss and optimizer you used, any training parameters you picked, how you computed the ground truth answer, etc. If you make changes from the original paper, describe here what you changed and why.

Firstly, the question is encoded as a multi-hot vector and embedded towards a feature vector (with the size 2000×1). And then the image feature was extracted using the convolution part of the googlenet. After getting the features, the embedding from word and images could be concatenated and than fed into a fully connected layer to achieve the classification. Ans we chose softmax as the classifier, cross-entropy should be chosen as the criterion. It is worth to mention that the learning rate for embedding layer should be different from other layers (in this case, the learning rate for embedding is 0.8 and for other layers, the learning rate should be 0.01). I did not change any hyper-parameters from the origin paper, because I trust the author.

To evaluate the performance of the training, I used two metrics to test the model. One is Top-5 mAP metric and the other is Top-1 mAP metric. Here is the results.

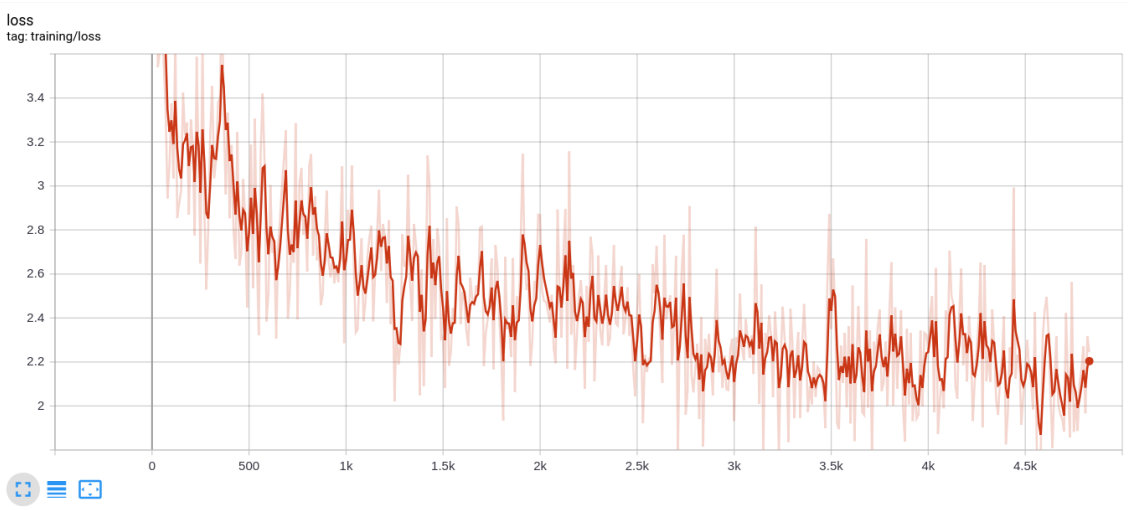

Figure 1: Loss for training with simple baseline

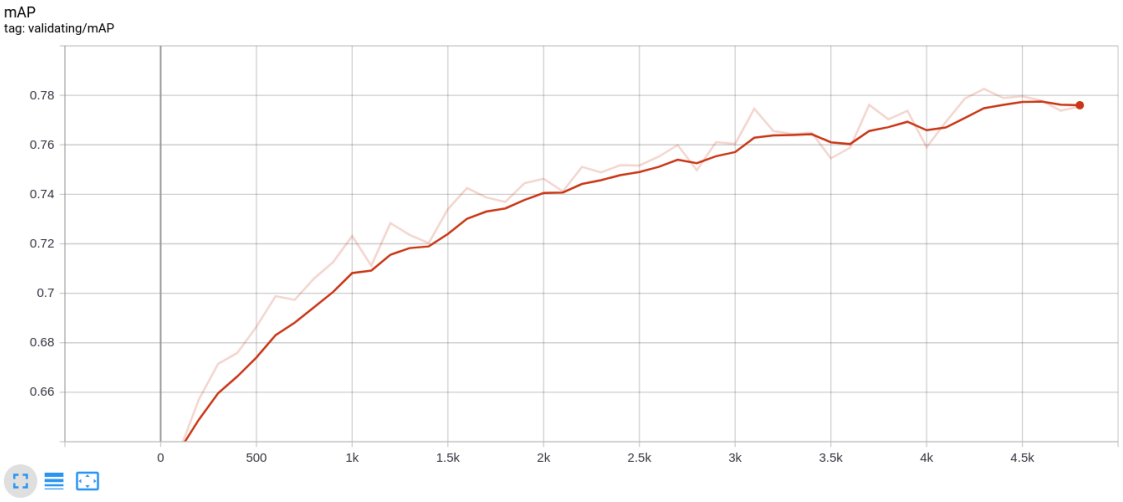

Figure 2: mAP for Top-5 metric

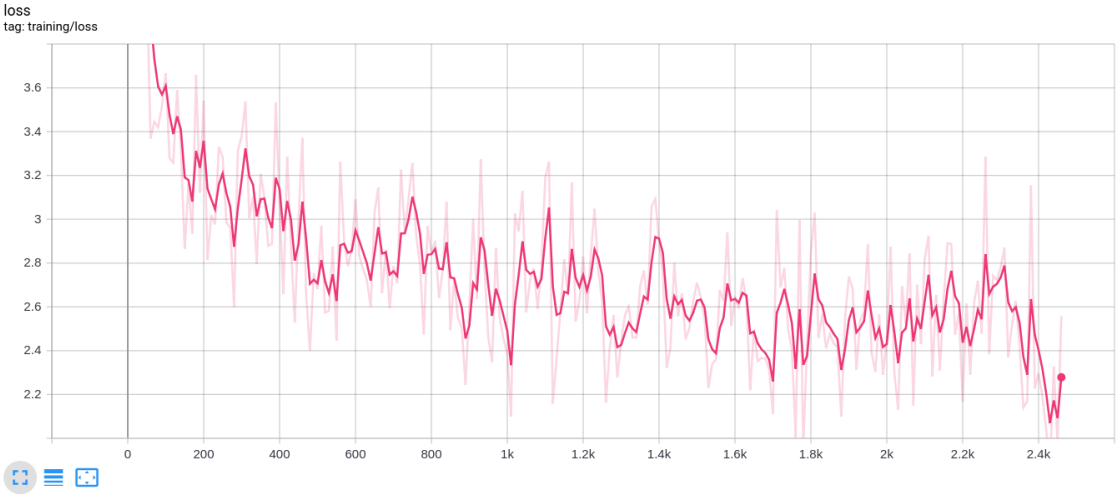

Figure 3: Loss for training with simple baseline

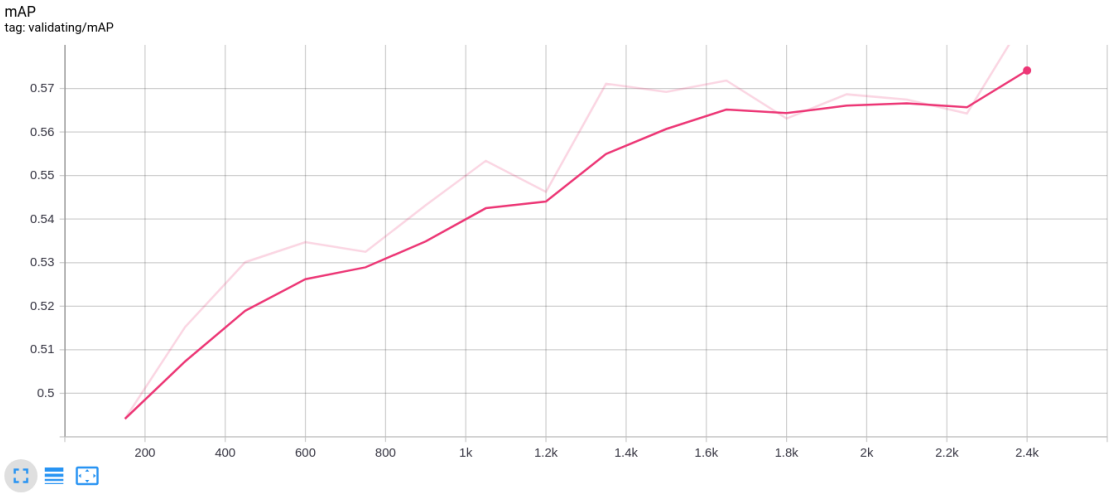

Figure 4: mAP for Top-1 metric

Obviously, the model is powerful enough to handle the VQA problem. The mAP for TOP1 metric could reach 55% or even higher.

0.3 Task 3: Co-Attention Network

Q3 As in the above sections, describe your implementation in brief, e.g loss, optimizer, any decisions you made just to speed up training, etc. If you make changes from the original paper, describe here what you changed and why.

I strictly followed the instruction for the article. The model I applied here is the parallel co-attention model. This model have two main modules, which are language embedding module and co-attention module. The language embedding module extracts information from bottom up to top using embedding layer, 1-d convolution layer and lstm layer. Therefore it could extract the information from three different level of language, which are word level, phrase level and question level. After getting the three level information of the questions, the network assembles the information from language and images together with three co-attention module, and each co-attention module should corresponds to one language level. The optimizer I used in this task is Adam. And similar to task 2, I did not change any hyper-parameters because I trust the author.

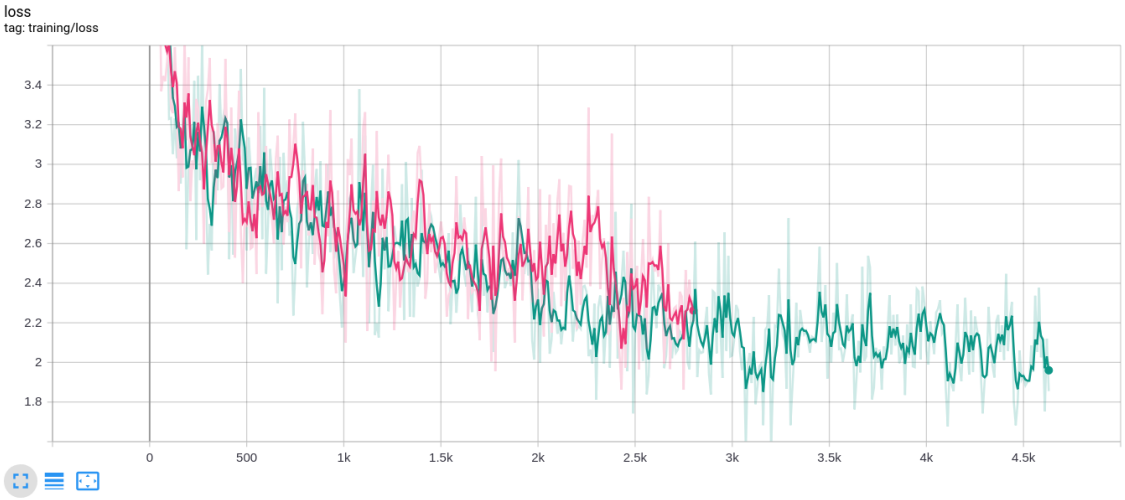

Figure 5: Loss for training with coattention model

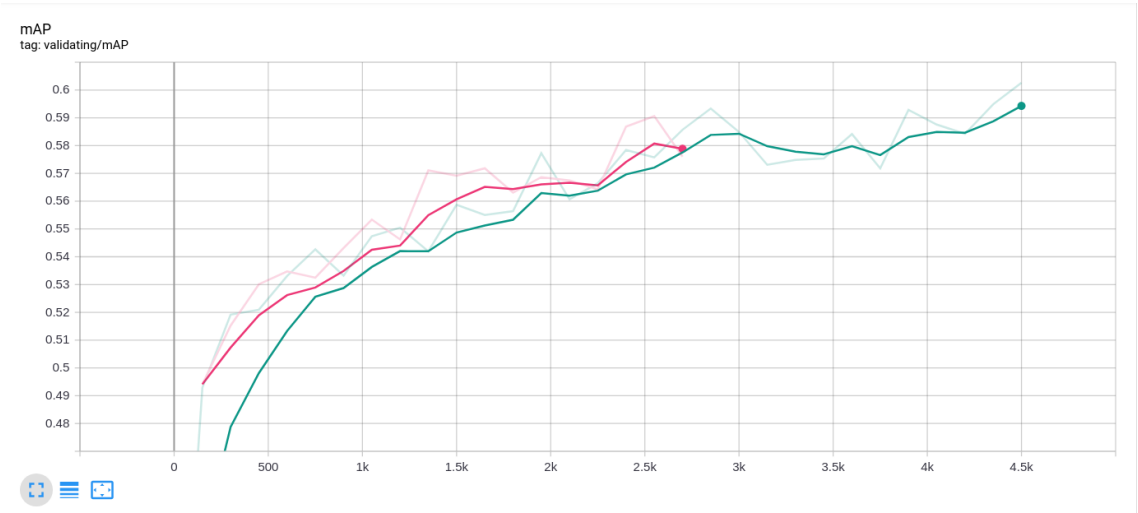

Figure 6: mAP for Top-1 metric with coattention model

Unfortunately, the loss and accuracy of coattention model does not be improved a lot as what I expected before. One possible benefit of co-attention model compared to simple baseline is that the model could have stronger ability to handle the difficult task. The performance could be improved if I got more time the train the model. But unfortunately, I made a stupid bug with the direction of softmax layer which cost me a lot of time to debug. Right now I do not have enough time to go through the whole training progress.

0.4 Task 4: Custom Network

Brainstorm some ideas for improvements to existing methods or novel ways to approach the problem.

1. It seems that the co-attention model does not consider the magnitude difference between image and question. One possible improvement is to use normalization technique or use weighted input features for training.

2. Although in the paper, the author suggested that the full model could achieve the best performance compared with the model with only one attention module. But it seems in the full model condition, overfitting may happened because of the complexity of the model(that is perhaps why the author added lots of drop out layer to avoid this issue). I believe that it is worth to try the model with only one attention model.

3. Based on the ablation research in the article, the author mentioned that the Question-level information seems to have more influence on the prediction accuracy. But however, this model is a bottom-to-up model, which we uses 1-d convolution and LSTM to collect the information from bottom, information may be lost during this progress. Therefor, if we can develop a top-to-bottom technique the process the question, I believe the performance could be further improved.