RNNs for Visual Question Answering

Daniel He, Duke Lin , Sneha Kondur, Tyler Farnan
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92092
{dhe,d3lin,skondur,tfarnan}@ucsd.edu
https://github.com/dukelin95/vqa_pytorch

Abstract

In recent years, the visual question answering (VQA) challenge is becoming significant as a benchmark in the field of Artificial Intelligence. The challenge requires a good understanding of the image and the textual question features in order to infer the correct answer. It promotes further research in image processing and natural language processing. This paper is mainly focused on introducing the latest deep learning techniques such as drop out and batch normalization to the already existing VQA models and perform a comparative study on the performance of the modified models on the test data. The results showed that only using dropout on the attention has the best performance compared to no dropout or including batch normalization after dropout. Implementing these standard changes to the attention network did not benefit the accuracy as expected, so more novel approaches to improve the attention network will be needed to dramatically increase the performance on VQA challenges.

1 Introduction

Deep Learning is one of the newest trends in the fields of Machine Learning and Artificial Intelligence. It has introduced revolutionary advancements in computer vision, image processing, natural language processing and human-computer interactions. Every now and then, new deep learning techniques are born, outperforming state-of-the-art machine learning. Although research has achieved high accuracy in detecting objects, it is still unable to clearly understand the traits and attributes associated with the detected objects. For example, in figure 1a, the object detection algorithm accurately detects a football and a person in the image. However, besides object detection, there is a large difference in how a human and computer perceives this image. When humans look at this image, they automatically understand the attributes associated with the objects and are capable of answering any questions based on this image's content. The high-level understanding of objects and its associated attributes enables humans to answer questions such as 'What color shoes is the person wearing?', and 'Where is the person playing?'.

1.1 Motivation

In order to make machines as intelligent as humans, more research is needed for computers to learn the attributes of objects. With this research goal focus, Visual Question Answering (VQA) [1] is particularly important and interesting because it allows us to understand what the computer actually sees and learns from an image. In VQA task, we present the model with an image and a question in the form of natural language and the model generates an answer in the form of natural language.

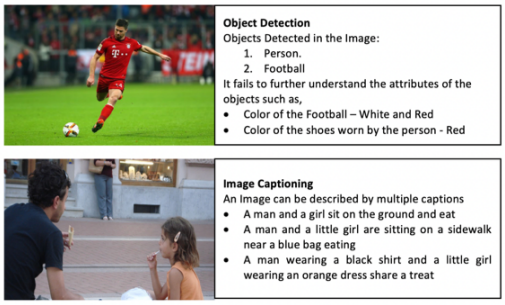

Figure 1: Object Detection and Image Captioning Tasks

1.2 Related Works

A related task to VQA that helps in understanding the attributes of the objects is the image caption generation [2], where the model generates a representative description of an image in natural language. For example, when the computer is presented with an image as shown in fig 1b, it must automatically generate a textual description such as "A man and a girl sit on the ground and eat" or "A man wearing a black shirt and a little girl wearing an orange dress share a treat". The task of image captioning has a few inherent limitations such as having a limited number of unique captions that describe an image and a single caption can be representative of a large number of images. Hence the generated captions are not a perfect representations of what the model actually sees and learns from the given image. However, these limitations are less severe in the VQA task. It is always possible to ask the model narrow questions to receive give a specific answer. Therefore, we can infer that the model performance evaluation of VQA is much more definite and accurate when compared to the performance evaluation in Image captioning, although both tasks help us to understand what the computer truly sees in the image.

A robust VQA system has many potential applications. A good VQA system can act as a helping/assisting tool for blind and visually impaired people. It enables them to get information and understand the real-world images which they are unable to see. For example, as a blind user scrolls through their social media feed, a captioning system can describe the image and then the user could use VQA to query the image to get more insight about the scene. More generally, VQA could be used to improve human-computer interaction as a natural way to query visual content. A VQA system can also be used for image retrieval, without using image meta-data or tags. For example, to find all images taken in a rainy setting, we can simply ask 'Is it raining?' to all images in the dataset. Beyond applications, VQA is an important basic research problem. If a good VQA system is able to solve many computer vision problems, it can be considered a component of a Turing Test for image understanding [3], [4]. A Visual Turing Test rigorously evaluates a computer vision system to assess whether it is capable of human-level semantic analysis of images. Passing this test requires a system to be capable of many different visual tasks. VQA can be considered a kind of Visual Turing Test that also requires the ability to understand questions, but not necessarily more sophisticated natural language processing. If an algorithm performs as well as or better than humans on arbitrary questions about images, then arguably much of computer vision would be solved. But this is only true if the benchmarks and evaluation tools are sufficient to make such bold claims.

There has been significant research in developing some of the novel and interesting techniques for the VQA challenge. We have referenced the paper "Show, Ask, Attend and Answer: A Strong Baseline for Visual Question Answering" [5] as our base paper and have tried to introduce some of the latest deep learning techniques such as Batch normalization and Drop out to further improve the model efficiency.

2 Methodology

A large number of algorithms have been proposed in the last few years to solve the VQA challenge. In general, the VQA algorithms consist of three core components namely:

1. Image Feature Extraction Algorithm
2. Question Feature Extraction Algorithm
3. An algorithm to combine the image and question features and predict an answer

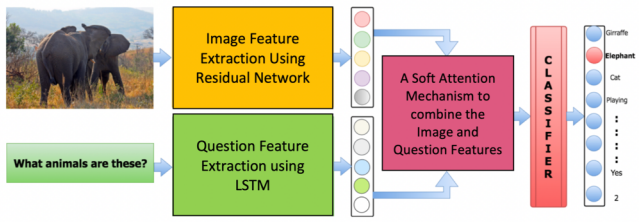

Figure 2: Block Diagram for the VQA Algortihm in [5]

In this section we will outline the architecture proposed in the base reference paper [5]. Fig 2 depicts a block diagram of the algorithm. The model uses a long short-term memory units (LSTM) to encode the questions and extract the question features. It uses a deep residual network to extract the image features. Finally, a soft attention mechanism is used to compute multiple glimpses of the image features based on the state of the LSTM. A classifier is then deployed on the image feature glimpses and the LSTM's final state to predict the probabilities over a fixed set of most frequent answers. The visual question answering task is treated as a classification problem. Given an image I and a question q in the form of natural language, we shall estimate the most likely answer $\hat{a}$ from a fixed set of answers based on the content of the image.

2.1 Image Feature Extraction

This implementation of VQA uses a special class of Convolutional Neural Network (CNN) called a Residual Network (ResNet). In general, a (CNN) is a neural network that uses convolutions instead of matrix multiplication, it has local receptive fields, shared weights throughout a given layer, and uses pooling to condense layers. ResNets in particular incorporate skip connections, which help solve the vanishing gradient problem.

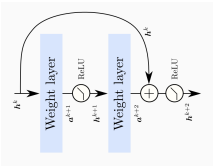

Figure 3: Residual Layer Unit (from ECE 285 notes)

2.2 Question Feature Extraction

This implementation of VQA uses a special class of Recurrent Neural Network (RNN) called a Long Short Term Memory RNN, or (LSTM). An RNN in general is a neural network designed to handle sequences of variable size. They incorporate internal feedback loops that fed the net's output back into the next along with new input, and have achieved state of the art results on time series prediction, especially in the domain of natural language processing. The RNN's major challenge is the ability to learn long-term dependencies, and LSTMs have been designed specifically to solve this problem. LSTM units contain memory cells that can read, write and reset information in its memory. Below is a brief description of LSTM functionality:

Input modulation gate: Determines if the old memory should be passed to the current unit.

Input gate: Determines how the new memory influences into the old memory.

Forget gate: Determines what unnecessary information should be removed from memory.

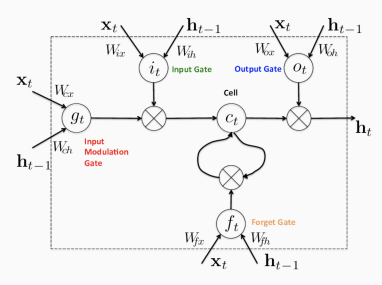

Figure 4: LSTM Architecture (from ECE 285 notes)

Output gate: Controls how much new memory should output to the next LSTM unit.

2.3 Attention Mapping

Attention networks allow the model to focus on a subset of features. Attention maps are produced by computing the correlations between question word features and the image feature map.

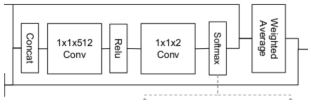

Figure 5: Attention Network Architecture [5]

2.4 Classifier

The classifier is the final module of our architecture. It receives as input the current LSTM state and the glimpse maps produced by the attention network. It produces probability distributions over all answer classes $a_1, a_2, \ldots, a_K$.

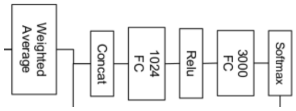

Figure 6: Classifier Architecture [5]

2.5 The Final Loss Function:

$$\mathcal{L} = \frac{1}{K} \sum_{k=1}^{K} -\log P\left(a_k | I, q\right)$$

2.6 Evaluation Metric

$$Acc(a) = \frac{1}{K} \sum_{k=1}^{K} \min\left(\frac{\sum_{1 \leq j \leq K, j \neq k} 1\left(a = a_j\right)}{3}, 1 \right)$$

This is the official evaluation metric from `www.visualqa.org`. The model's answer is counted as correct if it agrees with at least three of the annotations for the given question.

3 Experimental Setting

An ideal VQA dataset needs to be sufficiently large to capture the variability within questions, images, and concepts that occur in real world scenarios. It should also have a fair evaluation scheme that is difficult to 'game' and doing well on it indicates that an algorithm can answer a large variety of question types about images that have definitive answers. If a dataset contains easily exploitable biases in the distribution of the questions or answers, it may be possible for an algorithm to perform well on the dataset without really solving the VQA problem [6].

Our experiments all used the same training parameters, namely, a batch size of 32, initial learning rate of 1e-3, and a learning rate half-life of 50000 for the ADAM optimizer.

3.1 The VQA Dataset

The dataset used in this study is the VQA dataset [7]. The VQA Dataset consists of both real images from COCO and abstract cartoon images, which are not used in this case. The focus of the dataset is solely based on the portion which contains real world imagery from COCO and is named as COCO-VQA.

COCO-VQA comprises of three questions per image with ten answers per question. The questions were generated by Amazon Mechanical Turk workers. A separate group of workers were hired to generate answers for these questions. COCO-VQA contains a moderately large number of questions (614,163 total, with 121,512 for validation, 244,302 for testing, and 248,349 for training), when compared to other VQA datasets. Each of the questions is then answered by 10 independent annotators.

The large collection of diverse questions of COCO-VQA can be precisely answered without using the image due to language biases. Simple image-blind algorithms have achieved 49.6% accuracy on COCO-VQA using only the questions [8]. The dataset also contains open-ended questions which seek opinion-based answers that differs from person to person. Furthermore, some questions require explanations or effusive descriptions. An example of this is given in Figure 3, which also shows unreliability of human annotators as the most popular answer is 'yes' which is completely wrong for the given question. These complications arise due to inter-human agreement or disagreement on the dataset, which is around 83%. Different properties such as 38% of the dataset comprises of Yes/No questions and 59% of these questions have the answer 'Yes' further makes the VQA task more challenging. The dataset used in this study is the official Visual Question Answering, v2.0 Balanced Real Images. The data split for training, validation and testing is summarized in Table 1

Before using the VQA dataset, it must be preprocessed. The images are preprocessed by separating them into batches and then the features are extracted through resnet152. The questions and answers are preprocessed by extracting the most common vocabulary and also making sure the only the most common answers for each question are used.

VQA v2.0 Real Balanced Images	Input Images	Input Questions	Annotations
Training	82,783	443,757	4,437,570
Validation	40,504	214,354	2,143,540
Testing	81,434	447,793	n/a

Table 1: Data split for Training, Validation and Testing

3.2 Experimental Designs to Improve the BaseLine Model

In order to improve the performance of the baseline model, we tried to implement the Deep learning techniques of Drop Out and Batch Normalization. Four different experiments were implemented with the following changes to the models in the attention network: 50% dropout, 0% dropout, 50% dropout followed by batch normalization, and 0% dropout with batch normalization. The intuition behind using these techniques are further discussed below. The understanding of the benefits of these techniques motivated us to implement them over the base model.

3.2.1 Drop Out for regularizing Deep Networks

Large neural networks can overfit the data when they are trained on relatively small datasets. This could make the model memorize the features in the training dataset and result in an increase in generalization error i.e. the performance of the model when evaluated on test dataset is poor. Given the condition that there are no limits on computational complexity of the model architecture, the best way to regularize a model with a fixed size is to average the predictions of all possible parameter settings and weigh each setting by its posterior probability when the training dataset is given. With this idea, one approach to reduce overfitting is to fit all the different possible neural networks on the same dataset and to average the predictions from each model. This is highly impractical and can be approximated using a small collection of different models, called an ensemble. A more serious problem even with the ensemble approximation is that it requires to store and fit multiple models. This is particularly challenging in the case of large models as it requires days or weeks to train and tune the learnable parameters.

Dropout is a technique for improving neural networks by reducing overfitting [9]. Dropout is a regularization method that approximates training a large number of neural networks with different architectures in parallel. During training, a certain number of layer outputs are ignored randomly. They are "dropped out" i.e. they are temporarily removed from the network, along with all its incoming and outgoing connection as shown in fig 3. This makes the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different "view" of the configured layer. Dropout has the effect of making the training process noisy, forcing nodes within a layer to probabilistically take on more or less responsibility for the inputs. This conceptualization suggests that dropout breaks-up situations where network layers co-adapt to correct mistakes from prior layers, in turn making the model more robust.

Dropout is implemented layer by layer in a neural network. It can be used with most types of layers, such as convolutional layers, dense fully connected layers and recurrent layers such as the long short-term memory (LSTM) network layer. Dropout may be implemented on any or all hidden layers in the network as well as the visible or input layer. However, it is not used on the output layer. A new hyperparameter is introduced that specifies the probability at which the outputs of a layer are dropped out, or inversely, the probability at which outputs of the layer are retained. A common value is a probability of 0.5 for retaining the output of each node in a hidden layer and a value close to 1.0, such as 0.8 or 0.9 for retaining inputs from the visible layer. The weights of the network will be larger than normal because of dropout. Therefore, before finalizing the network, the weights are first scaled by the chosen dropout rate. The network can then be used as per normal to make predictions. If a unit is retained with probability p during training, the outgoing weights of that unit are multiplied by p at test time. Dropout is not used after training when making a prediction with the Test dataset.

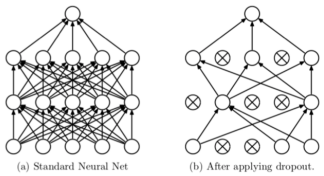

Figure 7: Drop Out Technique

3.2.2 Batch Normalization - Learn Faster, Learn Better

Training deep neural networks is challenging as they can be sensitive to the initial random weights and configuration of the learning algorithm. One possible reason for this difficulty is the distribution of the inputs to layers deep in the network may change after each weight update step. This can cause the learning algorithm to forever chase a moving target. This change in the distribution of inputs to layers in the network is referred to the "internal covariate shift." In general, Covariance shift is a problem that is not unique to deep learning. For instance, if the train and test sets come from entirely different sources the distributions would differ. The reason covariance shift can be a problem is that the behavior of machine learning algorithms can change when the input distribution changes. In the context of deep learning, we are particularly concerned with the change in the distribution of

the inputs to the inner nodes within a network. A neural network changes the weights of each layer over the course of training and this means that the activations of each layer change as well. Since the activations of a previous layer are the inputs of the next layer, each layer in the neural network is faced with a situation where the input distribution changes with each step. This is problematic because it forces each intermediate layer to continuously adapt to its changing inputs.

Batch normalization [10] is a technique for training very deep neural networks that standardizes the inputs to a layer after each iteration. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks. The basic idea behind batch normalization is to limit covariate shift by normalizing the activations of each layer (transforming the inputs to be distributed with mean 0 and unit variance). This, supposedly, allows each layer to learn on a more stable distribution of inputs, and would thus accelerate the training of the network. There are two variants of Batch normalization – depending on whether we normalize the weights either before or after the application of activation layer as shown in fig 4.

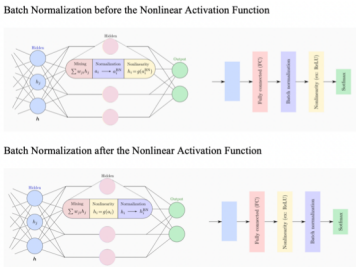

Figure 8: Varients of Batch Normalization

4 Results

The training loss, training accuracy, and validation accuracy over 5 epochs for each model have plotted and are shown below in figure 9.

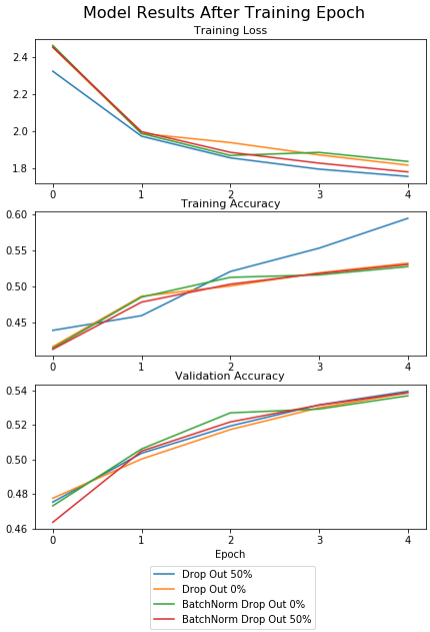

Figure 9: Results

As the training epochs increase, it is clear to see that the model implemented with 50% dropout performs the best in all metrics, especially in training accuracy by a large margin. However, all models have a similar validation accuracy with the model implemented with 50% dropout has highest accuracy, followed by the model implemented with 50% dropout and batch normalization, then by the model implemented with 0% dropout, and finally followed by the model implemented with batch normalization and 0% dropout. The performance in the other two metrics also follow this trend.

These results make sense as dropout improves generalization by ensuring that the network does not memorize the training set also known as overfitting. This is reflected in the results as both models implemented with a 50% dropout perform better than their 0% counter parts. An interesting result is that implementing batch normalization and dropout together results in a decreased performance compared to only dropout implementation counterparts. Our initial belief was that combining both dropout and batch norm would result in higher accuracy as dropout prevents overfitting and batch norm accelerates the training of each network by stabilizing the inputs, so implementing batch norm should result in higher accuracy in each training epoch before reaching the peak convergence. The lower accuracy when only using dropout does make sense based on results of paper [11], which states that implementing both dropout and batch normalization together leads to worse performance due to variance shift.

	Question	**Model Predicted Answers**	**Actual Answers**
1	what does the last sign say?	Batch Norm, No Dropout: stop Batch Norm, Dropout 50%: stop No Batch Norm, No Dropout: stop No Batch Norm, Dropout 50%: stop	Answer 1: ross st Answer 2: ross st Answer 3: ross st Answer 4: ross st Answer 5: ross st Answer 6: ross street Answer 7: ross Answer 8: ross st Answer 9: ross st
2	what color is the sign?	Batch Norm, No Dropout: red Batch Norm, Dropout 50%: red No Batch Norm, No Dropout: red No Batch Norm, Dropout 50%: red	Answer 1: red and yellow Answer 2: red Answer 3: red Answer 4: orange, yellow, white, black Answer 5: red and yellow Answer 6: red/yellow Answer 7: red orange Answer 8: red and orange Answer 9: red and yellow Answer 10: red
3	what color hair does the woman have?	Batch Norm, No Dropout: brown Batch Norm, Dropout 50%: blonde No Batch Norm, No Dropout: brown No Batch Norm, Dropout 50%: blonde	Answer 1: blue Answer 2: blue Answer 3: purple Answer 4: purple Answer 5: blue Answer 6: purple Answer 7: purple Answer 8: purple Answer 9: purple Answer 10: blue
4	What color is the kids hair?	Batch Norm, No Dropout: blonde Batch Norm, Dropout 50%: blonde No Batch Norm, No Dropout: blonde No Batch Norm, Dropout 50%: blonde	Answer 1: blonde Answer 2: blonde Answer 3: blonde Answer 4: blonde Answer 5: blonde Answer 6: blonde Answer 7: blonde Answer 8: blonde Answer 9: blonde Answer 10: blonde

Table 2: Predicted and Actual Answers

The performance for each model on example cases can be seen by referring to table 2 and figure 10. Figure 10 shows which part of the image the nets are focusing on through the glimpse maps, which clearly shows that in general they all perform equally well in locating the specific object in question. However, when including the natural language answers it is clear that there are performance differences between the models.

For the first question and image example, all the models predicted the answer as stop, while all the actual answers had to do with the street name. It is clear based on the glimpse maps that there was no focus on any individual sign board. Based on the incorrect prediction and the lack of focus in the glimpse maps, the models must not recognize certain vocabulary in the question in order to answer it. The models can only answer questions based on their set vocabularies, which are limited to general cases (most common), making specific pronouns and isolated cases harder to answer.

The second question and image example also corroborates this as the sign is clearly the focus of all glimpse maps and is answered correctly by all models. So all models perform well on a general question about a common object.

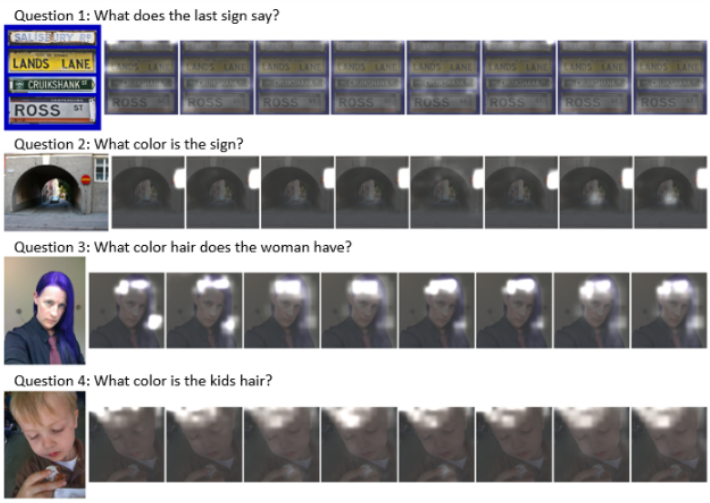

Figure 10: Glimpse Maps for Images and Questions: From left; Original image, 2 Glimpse Maps for Batch Norm with 0% Dropout, 2 Glimpse Maps for Batch Norm with 50% Dropout, 2 Glimpse Maps for No Batch Norm and 0% Dropout, 2 Glimpse Maps for No Batch Norm and 50% Dropout

The third question and image example show that the models recognized that the hair is the main object of focus. However, there are not enough cases where hair is the color indicated in the image or the models did not make the connection that hair can be of this color to answer the question correctly. All the actual answers are in agreement on what color the hair is, while the models answered with differing common hair colors such as blonde or brown.

The fourth and final question and image example show that the models can correctly identify the common attributes of a certain object, in this case the color of hair again. Again, the focus is on the hair, as shown by the glimpse maps. The actual answers are all in agreement and the models are all in agreement for what color the hair is.

These prediction results show the limitations and successes of the model. Overall, as long as the question is general and the image show common object attributes the models can answer correctly. The limitation is mainly attributed to the datasets which include the images, questions and answers, and the vocabulary. The models cannot answer correctly if the answer is not in the vocabulary and the answer will be incorrect if it encounters an uncommon feature in the object.

5 Discussion

5.1 Inference

Based on the improvements that can be made on networks with dropout and batch normalization, it can be inferred that an existing model with both of these improvements implemented will show some increase in accuracy before convergence to a peak accuracy. So, a model with dropout should do better than one without dropout and a model with both batch normalization and dropout should perform the best over earlier epochs of training. This is because dropout ensures that the network does not over fit and batch normalization, in general, improves the learning rate. So on our tests with lower epochs of training, we expect to see better performance.

5.2 Conclusion

Based on the results, we have confirmed that including dropout does improve the performance of the model. In addition, including batch normalization after dropout decreases the overall performance. The limiting factor in the accuracy of answers are based on the size of the vocab size and the data set. It is possible that the incorrect predictions could be based on over fitting, as certain features are indeed possible for specific objects but may not be common. Implementing these standard changes to the

attention network did not benefit the accuracy as expected, so more novel approaches to improving the attention network will be to dramatically increase the performance on VQA challenges.

5.3 Learning

Through implementation of the group project we have learned that it is crucial to set up consistent environments, file paths, and coding conventions in order to make group work streamlined. Through the use of all the tools we have gained experience in using Pytorch, Latex, and Github. Using such a large network has taught us the work flow in experimenting on deep networks. In addition, we have learned how to navigate networks layer by layer in order to find features such as generating the glimpses.

5.4 Challenges

Very long training times (1.5 hours per epoch), coupled with restricted connection time to GPU's on the DSMLP made it very difficult to complete training jobs. Our ability to efficiently collaborate was restricted by our inability to have shared access to very large data files, which include the datasets and weights.

5.5 Future work

Given the wide range of possible types of VQA challenges, it seems necessary to curate more specialized data sets in order rigorously explore the inter-workings of Visual Question Answering. Regarding our implementation, a potentially interesting direction could be to explore different methods for aggregating the image and word features prior to being input into the attention network.

6 References

[1] A. Agrawal et al., "VQA: Visual Question Answering," ArXiv150500468 Cs, May 2015.

[2] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: A Neural Image Caption Generator," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3156–3164.

[3] D. Geman, S. Geman, N. Hallonquist, and L. Younes, "Visual Turing test for computer vision systems," Proc. Natl. Acad. Sci., vol. 112, no. 12, pp. 3618–3623, Mar. 2015.

[4] M. Malinowski and M. Fritz, "Towards a Visual Turing Challenge," 2014.

[5] V. Kazemi and A. Elqursh, "Show, Ask, Attend, and Answer: A Strong Baseline For Visual Question Answering," ArXiv170403162 Cs, Apr. 2017.

[6] K. Kafle and C. Kanan, "Visual Question Answering: Datasets, Algorithms, and Future Challenges," Comput. Vis. Image Underst., vol. 163, pp. 3–20, Oct. 2017.

[7] S. Antol et al., "VQA: Visual Question Answering," in 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 2425–2433.

[8] K. Kafle and C. Kanan, "Answer-Type Prediction for Visual Question Answering," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 4976–4984.

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," J. Mach. Learn. Res., vol. 15, pp. 1929–1958, 2014.

[10] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," ArXiv150203167 Cs, Feb. 2015.

[11] X. Li, S. Chen, X. Hu, and J. Yang, "Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift," ArXiv180105134 Cs Stat, Jan. 2018.

[12] Yan Zhang, pytorch-vqa, GitHub repository, `https://github.com/Cyanogenoid/pytorch-vqa.git`