Exploring Question Embeddings for Visual Question Answering

Leon Chambers, Joren Lauwers
Massachusetts Institute of Technology
77 Massachusetts Avenue, 02215 Boston
{leonchambers, jorenl}@mit.edu

Abstract

A Visual Question Answering (VQA) system takes as input an image and a natural-language question about that image and returns a natural-language answer to the question. The task of VQA is challenging in that it requires an in-depth understanding of both visual and textual data, as well as the ability to merge those two channels of information in a sensible way. We worked on a somewhat simplified version of the VQA task by using abstract scenes formed from a library of clipart images and answering multiple choice questions instead of open-ended questions. The work in this paper seeks to build off of an existing VQA baseline by experimenting with three different Recurrent Neural Net (RNN) architectures for processing the textual content. The three different models achieved similar accuracy, and they met our expectations for performance.

1. Introduction

The task of Visual Question Answering (VQA) has received significant interest since the publishing of the VQA dataset [1]. A VQA system takes as input an image and a natural-language question about that image and produces a natural-language answer to the question. The task represents a fairly comprehensive AI challenge in that it requires deep understanding of both natural language information and visual information as well as the ability to merge the information from these two separate channels in a sensible way to make a decision. Multi-discipline tasks such as this are considered a step towards "AI-completeness" in that implementations are expected to be required to have an actual understanding of the task itself in order to perform the task well. Furthermore, open-ended questions can require any of a vast variety of AI tasks from the system, from fine-grained object recognition and activity recognition to commonsense reasoning.

The original VQA dataset consists of both real and abstract scenes. The real scenes are taken from the MS COCO dataset [7], and the abstract scenes are constructed by positioning images from a limited clipart library on one of two different background images. The task can either be performed with open-ended or multiple-choice questions. Each question in the dataset are accompanied by a set of 18 multiple choice answer [1]. We chose to work with the multiple-choice questions on abstract scenes. Using the abstract scenes instead of the real ones simplifies the process of object detection, allowing us to focus more on combining the information from the two modalities. Using the multiple-choice questions instead of the open-ended ones allows for a simpler output algorithm and a more contrained problem space.

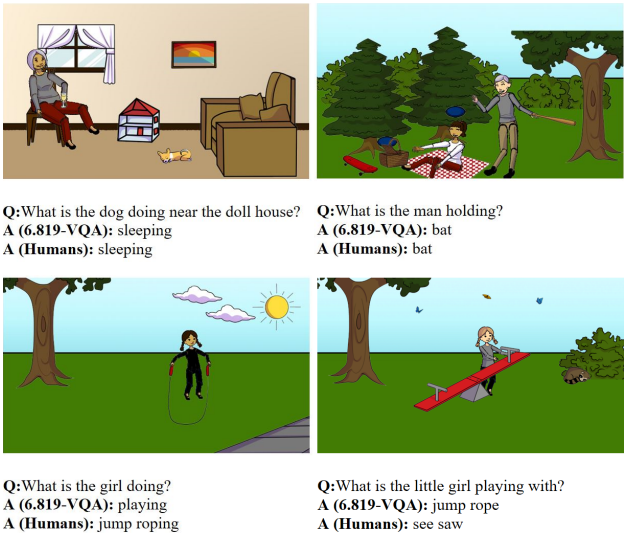

Figure 1. Examples of two correct and two incorrect answers given by our VQA implementation with a BLSTM on the question, compared with the human consesus for each as described in [1].

2. Related Work

2.1. Image Processing

Much work has been put into extracting features from an image. Datasets such as ImageNet [10], Microsoft COCO [7], and Places2 [16] have enabled wide exploration of neu-

ral net architectures for image classification. For example, [11, 3, 12] are all very different neural net architectures developed for the ImageNet Large Scale Visual Recognition Competition [9]. It should be possible to reuse these results for the task of VQA, and there have been attempts to do so already [1].

2.2. Language Processing

Answering text-based questions is a natural challenge in Natural Language Processing (NLP). Significant work has been put into creating word and sentence embeddings to facilitate running neural nets on natural language. For example, Word2Vec [8] is a now state-of-the art method for generating embeddings from words. Similarly, Skip-Thought [5] provides a method of generating sentence embeddings from a series of word embeddings, enabling entire sentences to be passed into a standard neural net easily. There are also systems developed for answering general questions. For example, [14] recently introduced a set of classes of question answering tasks and evaluated the performance of several neural net architectures on each of those tasks.

2.3. Other Vision and Language Tasks

There are several other tasks under active research that require understanding of both visual and textual information, such as image tagging, image captioning, and video captioning [6, 2]. However, the systems to perform these tasks can often be non-specific and might not need a deep understanding of the context of both domains. This is not true of VQA, where the questions can require arbitrarily specific understanding of the image.

2.4. VQA

[1] explores several architectures for the task of VQA. At a high level, all of its architectures generate an embedding of the image using a pre-trained VGGNet [11] implementation and merge it with an embedding of the question. They tried using a Bag of Words model to represent the question, as well a couple different LSTM models.

Much of the recent work in VQA focuses on developing an attention mechanism to use features of the question to decide what part of the image to focus on. [15] parses the question into a 3-element tuple and tries to verify whether or not the corresponding visual concept can be found in the image by looking for the objects mentioned in the tuple and examining the features of those objects and the relationships between them. [13] constructs graphs representing both the image and the question and then searches for correspondences between the 2 graphs so that relationships between different words in the questions and different objects in the image are preserved through the process of merging the two data modalities.

3. Approach

We implemented 3 different models based on the baselines defined in [1]. They all process the image information in the same way but have different ways of processing the question information.

3.1. Image Channel

The images were preprocessed by feeding them through an implementation of the 16 layer VGGNet [11] with frozen weights pretrained for the ImageNet classification task [10]. Specifically, the output of the last hidden layer of the VGGNet implementation was interpreted as an 4096-dimensional embedding of the image.

3.2. Question Channel

In preprocessing, the questions from the training set were used to create a comprehensive vocabulary of all words seen so far. A 300-dimensional embedding was learned for each word in this vocabulary. For every question, the embeddings of the words of the question were passed one at a time into a Recurrent Neural Network (RNN). The 3 different models used different RNN architectures.

1. **LSTM:** The word embeddings are passed one at a time through a 512-dimensional single-layer LSTM. The last cell state and output of the hidden layer are concatenated to form a 1024-dimensional embedding of the question.

2. **Deeper LSTM:** The word embeddings are passed one at a time through a 512-dimensional LSTM with 2 layers. The last cell state and outputs of the 2 layers are concatenated to form a 2048-dimensional embedding of the question.

3. **BLSTM:** The word embeddings are passed one at a time through a 512-dimensional BLSTM. The last cell states and outputs of the forward and backwards layers are concatenated to form a 2048-dimensional embedding of the question.

3.3. Channel Merging

The 4096-dimensional embedding of the image and the 1024- or 2048-dimensional embedding of the question (depending on the choice of RNN) were each passed through a single fully connected layer to transform them into a common space, and then the two were merged through element-wise multiplication.

3.4. Output Channel

In preprocessing, we chose the top 1000 most frequent answers in the training set as possible outputs. The combined question and image embedding was passed through

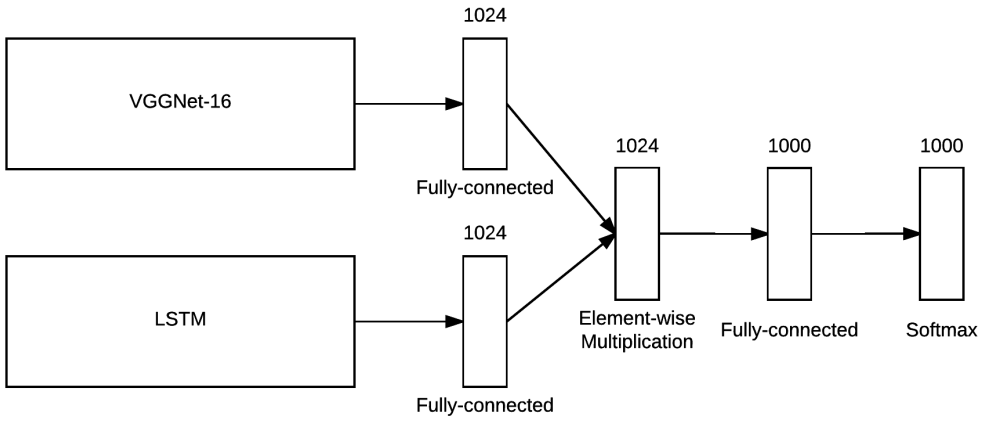

Figure 2. An abstract depiction of the models we used. The question words are run through some LSTM variant one at a time to produce an embedding of the question. The last hidden layer of a pre-trained VGGNet [11] implementation is used to encode the questions. Both the question and images features are transformed to a common space by a fully connected layer and fused by element-wise multiplication. The result of this multiplication is passed through one last fully connected layer followed by a softmax layer to obtain a distribution over the answers

a single fully connected layer followed by a softmax layer to produce a 1000-element vector representing the confidence the network has that a given output is the correct one. To answer a question, we pick the answer choice with the highest output value from this vector. This means that there are some answer choices that can never be chosen, but this only prevents us from correctly answering about 17% of the questions in the training and validation sets.

3.5. Training

The entire model (except for the frozen parameters in the image channel) was implemented in TensorFlow [1] and trained end to end using an Adam optimizer [4] with a learning rate of 0.001 and a dropout rate of 0.5 for all hidden layers.

4. Results

We trained each model using the training set for 100 epochs on an 8-core Intel Xeon E5-2673 processor, totaling around 25 hours of computation. After each epoch, the entire validation set was evaluated by the network, and the results stored for analysis. The increase in accuracy for each model is shown in Figure 3. It is possible that the models didn't actually converge after the 100 epochs, but it's impossible to tell from the graph.

[1]Code for the TensorFlow model is publicly available at https://github.com/LeonChambers/VQA

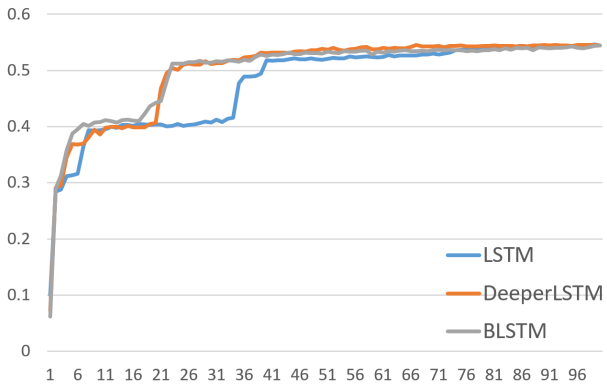

Figure 3. Progression of accuracy measure for each of the three implemented models per training Epoch

4.1. Accuracy Metric

We evaluated our models using the accuracy metric proposed in [1]. We experimented with increasing the leniency of which answers are considered equal by removing punctuation, articles and ordinal numbers to closer approach the human interpretation of a correct answer but this did not affect performance.

$$\text{accuracy} = min(\frac{\#\text{humans that provided that answer}}{3}, 1)$$

We also tried handling a more strict accuracy metric, where we considered the model's answer correct only if it was the

Model	Highest Achieved Accuracy
LSTM	54.62%
DeeperLSTM	54.57%
Bilateral LSTM	54.41%

Table 1. Accuracy measure for each of the three implemented learning models

most common answer provided by human workers. For all three models, the lenient accuracy metric is around 14% higher than the strict one, indicating a correlation between when humans answer questions ambiguously and when our models do.

4.2. Performance

We reviewed random samples of the Bilateral LSTM model at its last training epoch. Several patterns are apparent: 1) The model reliably recognizes yes/no questions but nearly always respond "yes", since this is by far the most frequent correct answer. The authors of [15] released a balanced dataset to counter this issue, but we did not attempt to run our model on this dataset. 2) The performance for counting question appears poor. The model consistently answers "2", which is also the most common number-type answer and the third most common answer overall.

5. Conclusion

The three models had very similar performance. However, the DeeperLSTM and BLSTM seem to have converged more quickly than the LSTM. It is possible that having more weights to train allowed those two models to adapt quicker to the types of questions that it was seeing.

After browsing the results files, the jumps in performance in Figure 3 seem to represent concrete "realizations" that the network had about the questions. After the first epoch, the networks learn to answer "yes" to everything, which provides a large initial boost in accuracy. After a while, the networks seem to learn which questions should be answered with "yes"/"no", which should be answered with numbers, and which should have more general answers. After realizing this, it seems to start understanding the actual structure of the questions and slowly gaining accuracy. We suspect that the models hadn't fully converged after the 100 epochs of training. Given more time, we would have run the training for much longer to determine whether or not the accuracies of the models would start to diverge and whether they would start to over-fit to the training data.

We expected the DeeperLSTM and BLSTM to perform noticeably better than the LSTM, but this didn't happen in practice. Perhaps in the early stages of training, the network was not able to properly take advantage of the more complicated network structures. It is also possible that the more complicated network structures actually inhibited training and that the models are actually comparable, but the results from [1] suggest otherwise.

6. Division of work

I personally implemented and ran the pre-processing script and TensorFlow model and wrote a lot of the report. Joren mostly worked on post-processing and analyzing the data.

References

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: visual question answering. *CoRR*, abs/1505.00468, 2015.

[2] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 15–29, Berlin, Heidelberg, 2010. Springer-Verlag.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[4] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[5] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[6] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating image descriptions. In *Proceedings of the 24th CVPR*, 2011.

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[11] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[13] D. Teney, L. Liu, and A. van den Hengel. Graph-structured representations for visual question answering. *CoRR*, abs/1609.05600, 2016.

[14] J. Weston, A. Bordes, S. Chopra, and T. Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698, 2015.

[15] P. Zhang, Y. Goyal, D. Summers-Stay, D. Batra, and D. Parikh. Yin and yang: Balancing and answering binary visual questions. *CoRR*, abs/1511.05099, 2015.

[16] B. Zhou, A. Khosla, À. Lapedriza, A. Torralba, and A. Oliva. Places: An image database for deep scene understanding. *CoRR*, abs/1610.02055, 2016.