VQA v1

Yannis Karakozis

Methodology

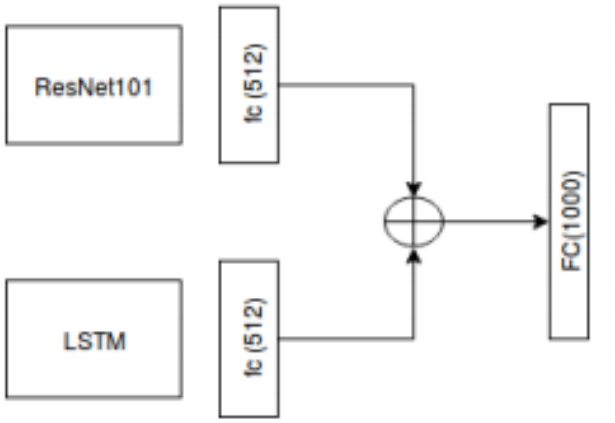

Figure 1: Baseline Model Architecture

I implement a model architecture that is similar to the given baseline. The baseline architecture is shown in Figure 1. The model is composed of two encoders: (1) the image encoder that maps each image to a feature vector and (2) the question encoder that maps each question to a feature vector.

The image encoder receives as input the 2048-dimensional image feature vector output of a ResNet101[3] model pretrained on ILSVRC [6]. It passes the vector through a fully connected layer that outputs a 512-dimensional vector. This is the image feature representation to be combined with the question feature representation.

The question encoder uses the 300-dimensional GloVe [5] word embedding and a single-layer LSTM. The hidden dimensionality of the LSTM is 512. Thus, the GloVe-LSTM pipeline produces a 512-dimensional vector output for each question. I then perform element-wise addition between the image vector and the output of the LSTM. This fused vector is fed into a final fully connected layer with softmax activations, as is done in [1].

VQA is open-ended so there is a massive space of possible answers. I choose the most common 1000 answers and train the model using cross-entropy loss. Thus, the output dimensionality of the final fully connected layer is 1000.

I initialize all fully connected layers by drawing values from $[-s, s]$ uniformly at random. The spread s is given by $2/\sqrt{\text{fan-in} + \text{fan-out}}$, where fan-in is the number of input neurons and fan-out is the number of output neurons of the fully connected layer. For each batch, I also initialize the hidden state and the cell state of the LSTM with random numbers drawn from the standard normal distribution.

I train on all the question-image pairs of the Balanced Real Images of the VQA dataset [1] whose 'multiple_choice_answer' is included in the 1000 most common answers. The 'multiple_choice_answer' is the most frequent ground-truth answer among the 10 possible answers annotated for each question [1]. During inference, I predict the answer that is assigned the highest softmax probability among the 1000 most common answers.

I train for 20 epochs with batches of 64 question-image pairs using the Adam optimizer [4] with learning rate 1e-5. I save the model corresponding to the epoch in which my model has the highest accuracy in predicting the 'multiple_choice_answer' of the question-image pairs of the validation set.

Evaluation

My model achieves 47.88% accuracy when evaluated using the code provided in the Github repository of the original VQA work (https://github.com/GT-Vision-Lab/VQA) [1]. Note that this performance is attained despite the fact that my model automatically predicts the incorrect answer for question-image pairs whose answers do not include any of the 1000 most common answers.

Examples

(a) (b)

(c) (d)

Figure 2: Figures a and b present examples of image pairs whose question was answered correctly by the model. Figures c and d present examples of questions whose question was answered incorrectly by the model.

The question for Figure 2a is "Is this in a restaurant?". The model correctly predicted that the answer

is 'yes'. The question for Figure 2b is 'What color is the woman's shirt?'. The model correctly predicted that the answer is 'white'.

The question for Figure 2c is "How many buses are shown?". The model incorrectly predicted that the answer is '2'. The question for Figure 2d is 'What kind of donut?'. The model incorrectly predicted that the answer is 'donut'.

Ablation

To investigate to what extent the fusion of image and question features is required to perform well on the VQA task, I evaluate the performance of my model when it is just given the input image (model **I**) and when it is just given the input question (model **Q**). I expect the performance of my model (**I+Q**) to be much higher than the performance of **I** and **Q**. If that is not the case, then the model is simply leveraging strong image or language priors and is not really learning to answer questions based on the visual information of the image it receives.

Model	Accuracy
I	23.49%
Q	37.03%
I+Q	**47.88**%

Table 1: Performance evaluation of the components of the VQA model.

Table 1 presents the accuracy achieved by each model. **I** performs the worst, which is to be expected given that it is really hard for the network to predict the answer to a question that it does not see. **Q** performs much better at 37.03% accuracy. Therefore, it seems that a significant portion of the model accuracy is due to language priors. However, there is a significant performance difference between that and the 47.88% accuracy of the full **I+Q** model. This implies that fusing image and question features is needed to do well on VQA.

Conclusion: My model answers questions by reasoning using both the visual information of the image it receives and the language priors it has learnt. If its inference was not truly multimodal, its performance would be at least 10% lower, according to the results show on Table 1.

Reflection

I initially tried a very different model parametrization than the one presented in the first section. In the initial implementation, I used the 50-dimensional version of the GloVe embedding [5]. The output dimensionality of the fully connected layer of the image encoder was 256. Similarly, the hidden dimensionality of the LSTM was also 256.

Under this implementation, I achieved an accuracy of 42.22% on the final evaluation, which was short of the 45% target. I assumed this was because the image feature and question feature representations were not rich enough (i.e. high dimensional enough) to allow the final fully connected layer to establish accurate enough classification boundaries. Thus, to make the answer classification space more separable, I doubled the hidden dimensionality of the LSTM and the output

dimensionality of the fully connected layer of the image encoder. To also enable the LSTM to learn more powerful question representations, I moved from the 50-dimensional to the 300-dimensional version of GloVe.

Future Work

The biggest weakness in the current implementation is the way the 512-dimensional image and question features are combined. Element-wise addition is a very arbitrary feature fusion method (feature fusion methods are also referenced as multimodal pooling in the literature). Its primary issue is that element-wise approaches to multimodal pooling are not as expressive as an outer product of the visual and textual vectors. This is because element-wise operations do not rely on the cross-vector interaction of all the vector components, the way a vector outer product does. Instead, each vector element interacts only with one element of the other vector.

Outer product are typically infeasible in deep learning settings as they require $O(n^2)$ computational complexity, where n is the dimensionality of the input vectors. Since the incoming feature vectors are of high dimensionality (512), an outer product would significantly slow down training and inference. Thus, instead I propose utilizing Multimodal Compact Bilinear pooling (MCB) [2] to efficiently and expressively combine the image and question features.

Due to the limited time and computational resources available, I was not able to experimentally verify whether this adjustment would actually boost performance. However, I expect this alternative feature fusion method to improve accuracy by approximately 3%, as this was the performance boost that was observed over element-wise sum in the work that first leveraged MCB for VQA [2].

Python 3.6.7 Libraries

- **pytorch** for model implementation and training.
- https://github.com/GT-Vision-Lab/VQA for performance evaluation.

References

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

[2] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," *CoRR*, vol. abs/1606.01847, 2016. [Online]. Available: http://arxiv.org/abs/1606.01847

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[5] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[6] 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*, vol. abs/1409.0575, 2014. [Online]. Available: http://arxiv.org/abs/1409.0575