HW4: VISUAL QUESTION ANSWERING

ARJUN MANI

1. Introduction

Visual question answering is a challenging task in artificial intelligence that combines computer vision with natural language processing. The problem is as follows: given an image and a question associated with the image, use the image to answer the question correctly. Many deep learning approaches to this problem map both the question and the image to some d-dimensional space. The embeddings are then combined and the output is transformed to a probability distribution over a set of possible answers. Note that while the space of possible answers is infinite, finding the probability distribution over the N most common answers tends to approximate the task well.

We use the *Balanced Real Images* dataset for the VQA task. The training set contains about 80,000 images from the COCO dataset along with 400,000 question-answer pairs. We will evaluate the performance of our model on the validation set.

2. Approach

Our approach consists of two parts: (1) an image encoder and (2) a question encoder. The job of the encoders is to convert the input (image or text) into a d-dimensional embedding representation of that input. In this case, $d = 512$. Following the encoding step, the two embeddings are combined via element-wise addition. This combined embedding is then fed into two fully-connected layers with 1000 nodes each, the first layer with ReLU activation and the second with softmax activation. Note that although the space of possible VQA answers is infinite, we choose to predict only the 1000 most common answers and train the model using cross-entropy loss.

2.1. Question Encoder.

The question encoder uses LSTM networks to create a 512-dimensional embedding of the input. The question is first converted from text to an embedding, where each word in the question is represented by a 300-dimensional embedding vector. To represent each word, we use the GLoVE word embeddings, which span a vocabulary of 400,000 words and are very commonly used for word embedding [1]. The question embedding is then fed to a two-layer LSTM, where each layer in the LSTM has an output dimension of 512. The output of the question encoder is the output of the second layer at the last time step (that is, the output of the LSTM for the last word of the question).

2.2. Image Encoder.

To represent each image, we use the features that were provided to us by the assignment. The features are of dimension 2048 and are extracted from a pretrained ResNet 101 model. They are fed to a fully-connected layer with 512 nodes and ReLU activation. The output of the image encoder is the 512-dim activations of this fully-connected layer.

2.3. **Implementation and Model Training.** As mentioned, the model is trained using cross-entropy loss and only predicts the 1000 most common answers. We use the Adam optimizer with a learning rate of 0.001 to train the model for 20 epochs. Since the implementation of this model is highly nontrivial, we briefly touch upon the implementation details, in particular the creation of the training set. For the questions, we convert each word to a unique index (based on its frequency in the training set questions). In order to train using batches, we pad each question in the batch to a size of 30, which is larger than the length of any question in the training set. This 1D representation is then converted to the GLoVE embedding. For the answers, we find the 1000 most common answers in the training set (removing spaces when necessary) and encode them using one-hot encoding. Only questions with one of the 1000 answers are used to train the model. We can thus train the model using cross-entropy loss.

3. Results

The accuracy of our model on the validation set is **44.16%**. Note that this accuracy is taken over the entire validation set, not just the questions with one of the top 1000 training set answers. Here are some of the sample answers from our model.

Image ID: 262162
Q: Is that a folding chair?
A: yes

Image ID: 393226
Q: What is the man doing in the street?
A: skateboarding

Image ID: 131115
Q: How many people are on the field?
A: 2

Image ID: 393267
Q: Why are some people wearing hats?
A: happy

Image ID: 393277
Q: What year is the car?
A: American

Although these examples provide only anecdotal evidence, they are representative of some larger trends. The model tends to output yes or no for yes/no questions (it achieves $\sim 70\%$) on these questions. For "how many" questions, the model tends to output a reasonable answer (i.e. a number), though the accuracy on these question is only 30%. For open-ended questions, the model tends to output reasonable answers for questions that can be directly inferred from the image (like the second question). However, on questions that require reasoning (4) or knowledge beyond the image itself (5), the model tends to do very poorly. The overall accuracy on these questions is 30%.

4. Ablation

For the ablation study, we examine how removing the image input entirely affects the accuracy on the validation set. In theory, the question requires the image to produce

the correct answer. Often, however, there are strong priors on the answers to certain types of questions, such that a model just trained on the questions can predict the correct answers with reasonably high accuracy. The purpose of this ablation is to investigate whether this is the case, or whether the image is indeed useful for answering a question as it should be.

The ablated model simply removes the image input from the current model. Thus, the input to the model is a GLoVE embedding of the question, and the model consists of a two-layer LSTM, each of hidden dimension 512, followed by two dense layers of dimension 1000 (with softmax on the last layer). The accuracy of the model on the validation set is **37.33%**. This is roughly a 7% drop from the accuracy with image+question, indicating that the model uses the image to achieve the results we report. Thus, our ablation study is a success.

There are a few interesting things to note. First, the fact that the question-only model was still able to achieve a reasonable accuracy, especially across an infinite state space of answers, indicates that there are still strong priors on the answers to some types of questions. The question-only model performs comparably to the full model on the yes/no and number questions (63% and 28% respectively). The biggest drop is in the 'other' questions, to 20% from 30%. This is sensible, since these questions are open-ended and thus least likely to have priors associated with their answers. Therefore, the model is more likely to require the image to produce the correct answer.

5. Reflection

Reaching the baseline accuracy of 45% was very challenging, and indeed I did not reach it despite coming very close. Initially, the model performed at an accuracy of about 42%. I experimented with several changes to the model, including changing the merge layer to element-wise multiplication, adding a fully-connected layer after the LSTM output, and removing one of the dense layers following the embedding, but none of them improved the accuracy. I even briefly tried a 1024-dimensional embedding that combined the hidden state and cell state of the last time-step for the two-layer LSTM. This, however, did not train very well.

In order to raise the accuracy, I switched to using the GLoVE embedding. Previously, I was training an embedding layer to generate 300-dimensional embeddings for the question words. Since the embeddings are initialized to a random normal distribution, it seems likely that the model would train better if given high-quality pretrained embeddings instead of having to jointly learn an embedding and model. This proved to be the case, as using the GLoVE embeddings resulted in approximately a 2% increase in accuracy.

6. Next steps

The first next step would be to train more complex models. In [2], for the multi-layer LSTM, the cell state and hidden state at the last time step are concatenated for each layer, for a total embedding of $512 \times 4 = 2048$. I briefly attempted a similar model where only the cell state and hidden state of the second layer are concatenated, but was not able to successfully train the model. I would like to try more such models that work with the cell state of the LSTM. Given the results of [2] for the 2-layer LSTM + Image, I could expect a 10% improvement in accuracy if I am able to successfully train the model.

In addition, I would like to train an image encoder myself. I am curious whether using different CNN architectures could improve the performance of the model. As an example, [2] uses VGG-16 instead of ResNet-101, and given the results I would be inclined to try different models for the image encoder. Since all these networks differ by at most 2-3% on image classification tasks, I would not expect the accuracy on the VQA task to increase by more than this.

The farthest next step might be to experiment with a Relation Network, which was developed by DeepMind [3]. I have not explored these networks in-depth, but their ability to reason about relations between objects might prove successful at visual question answering.

7. References

[1] Jeffrey Pennington, Richard Socher, Christopher D. Manning. GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/projects/glove/.

[2] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. VQA: Visual Question Answering. In International Conference on Computer Vision, 2015.

[3] Adam Santoro, David Raposo, David G.T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, Timothy Lillicrap. A simple neural network module for relational reasoning. arXiv.