Artificial neural networks and deep learning

Homework 3: Visual Question Answering

Motta Dennis Ostrovan Eugenio

January 30, 2021

1 Initial approach

Our first step in approaching the problem was to carefully inspect the requirements and the provided data: the dataset consists in a set of elements made of a question, an image and an answer. In the provided training set there are 65204 questions and 29333 images, so it's not a small dataset. Questions have a high variety of topics and length ($average = 6.1$, $max = 22$, $min = 2$ [words]). But the distribution of the answers is very skewed: most answers are yes or no.

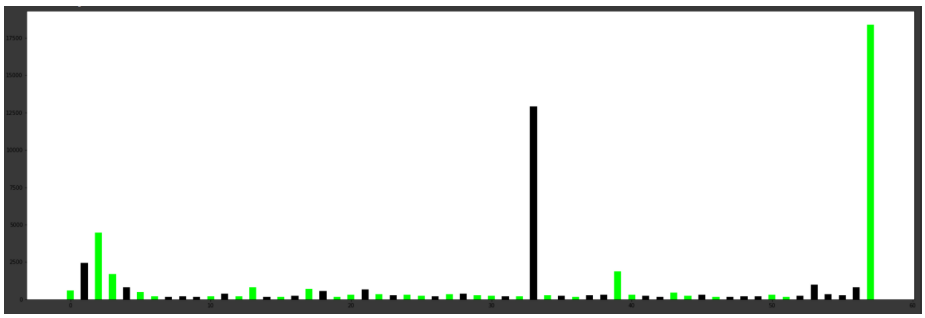

Figure 1: The distribution of the answers.

Due to the unbalance in the training set we noticed that we could build a baseline model that simply assigns yes/no answers randomly, this approach achieved an accuracy of 0.24 on the test set.

2 A model with heterogeneous data inputs

Images and text had to be combined during learning to assign answers. This was achieved using a custom data set implementation. The model processes the image through a convolutional network while the text is embedded and processed with an LSTM. These two processed inputs are then combined by multiplication.

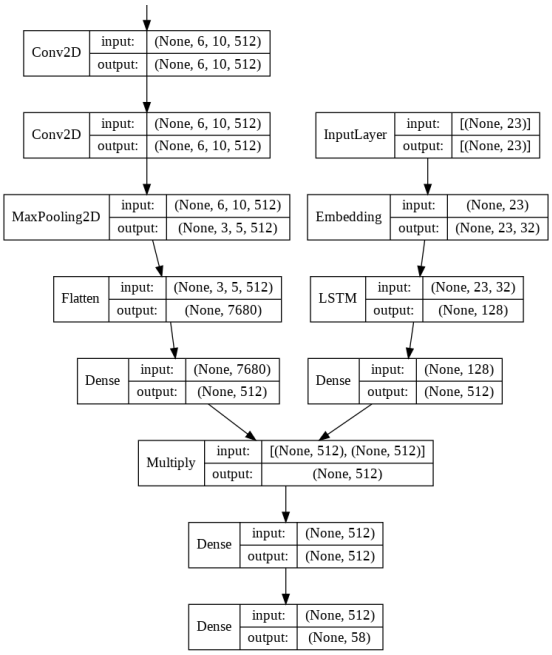

Figure 2: How the two inputs are combined

3 First model

Thanks to the experience we had from the previous homeworks we were able to quickly implement a functional and powerful model, we used the VGG16 model with imagenet weights of homework-1 for the convolutional part and added quickly the tools that we also previously used to analyze and improve the model:

- Hold-out validation;
- Tensorboard visualization;
- Checkpoint saving;
- Early stopping;
- A callback for reducing the learning rate on the plateau;
- Confusion matrix;
- Classification report;
- Displaying a few samples of questions of the test set with the respective prediction on the answer.

In this phase we achieved a test accuracy of 0.58.

4 Input image size

The first model had a problem: it had more than 80M parameters and made the training prohibitively expensive. This was due the fact that we used the images in their entirety, keeping the size of 400x700. Of course this resulted in a disproportional model with a huge number of parameters, almost all used for the image input. To fix this problem we resized the images and also added extra maxpooling layers. Equivalently a convolution with $stride > 1$ reduces the size of the image. The difference between using maxpooling and convolutions with stride to reduce the image size are the following:

- maxpooling: less expensive computationally than convolutions, but no features are learned in the downsampling process
- strided convolution: the operation parameters can be learned, but it also increases network depth and could make gradient propagation harder

In our model we applied striding to an already existing convolutional layer and with these changes we achieved a test accuracy of 0.61.

5 Additional analysis

After training our model we looked at the distribution of its predictions and compared it with the distribution of answers in the test set (1).

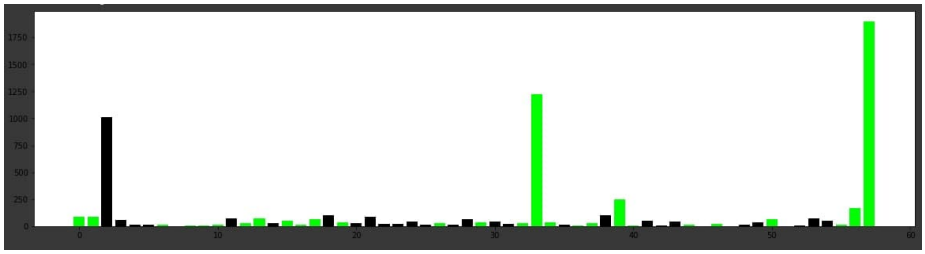

Figure 3: Distribution of classes in predictions

We can see that it is similar to the distribution of the training data, except the class '2' which corresponds to the number 2: in the training set it has a much lower count than the other two prominent classes while here it is very comparable.

6 Search on the hyper-parameters

We then performed a manual hyper-parameters search using the holdout validation set. We tried using different numbers of units in LSTM, a different learning rate and small variations in the model (strides in the convolutional part). But

with all of these changes we didn't obtain an improvement on the accuracy on the validation set, and also on the test set we only reached a maximum accuracy of 0.60.

7 Conclusion

The main concept that we learned thanks to this homework was how to manage and use text in a neural network by using tokens and the LSTM layer of Keras. But we also learned how to manage problems with heterogeneous inputs. We enjoyed being able to experiment practically with all the main concepts of Deep Learning while also having fun competing with some friends on the leaderboard.