ANNDL Homework 3 Report

Introduction

In the following report we are going to describe how we approached the Visual Question Answering problem given as an assignment. Our goal was to implement a model able to associate the correct answer to a combination of image-question. In particular, there were 58 possible answers between which the model had to learn to choose from.

The accuracy of our model, from the beginning, was about 0.50, so we decided to focus on the implementation of this first approach on which we applied modification and fine tuning of the chosen hyperparameters until reaching the score of 0.647.

Dataset Management

In the first section of the code we worked on collecting and reorganizing images and questions of the training dataset. Tokenization and padding of the questions was applied using tf.keras.preprocessing tools and the custom generator (*batch_generator*), which allows a correct generation of batches for the network, was developed. A further improvement in the performance could have been reached by incrementing the batch size to more than 70 (which was our final choice) which wasn't possible to test due to exhaustion of GPU memory on Google Colab.

Model implementation

The structure of the model is inspired by a paper [1] on VQA, which aims to solve the problem that we were given with an original architecture, which consists of two parts: one for the processing of the images and one for the processing of the texts, joined together with a *concatenate* layer.

Image processing model

Initially, we chose the VGG16 model for transfer learning, and successively switched to VGG19 which improved the performance of our model. All pre-trained layers were set as not trainable.

For this part, we made an alternative attempt. As a matter of fact, we tried to set the first layers non-trainable, and to keep the others trainable. The idea behind this attempt was that the first layers of VGG would be already trained to capture the most generic features of the dataset's images while the bottom layers had to be fine tuned to learn the specific features of our training dataset. At the end, this attempt didn't reach as good results as the initial implementation.

Text processing model

In order to embed the questions, the model uses a two layer LSTM to encode the questions. Initially we used LSTM layers and Dropout layers. In a second moment, we perceived that this part was less powerful than the Image processing part of the model. Therefore, for building a RNN with a longer range dependencies, we made Bidirectional the first LSTM layer. This layer permits the propagation of input in the RNN.

Final model and Fine tuning

In the final part of the implementation of the model, the image features are normalized. Then, both the question and image features are transformed to a common space and fused via element-wise multiplication, which is then passed through a fully connected layer followed by a softmax layer to obtain a distribution over answers.

We then proceeded to tune the hyperparameters and the structure of the model. Following are some of the parameters that influenced it:

- The compilation was likely standard, using Adam optimizer and Categorical_Crossentropy (as we used one-hot representation for the target).

- the learning rate was changed from 1e-4 to 5e-4 and eventually to 1e-3 to reduce oscillations

- modifications of percentage used in the Dropout layers

- Leaky ReLu as activation function in dense layers instead of standard ReLu

- addition of a Learning Rate Plateau, that allowed us to dynamically change the model's learning rate whenever the val_loss parameter didn't change

References

[1] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Batra, D. Parikh. VQA: Visual Question Answering.

Authors

Arianna Galzerano
Francesco Fulco Gonzales
Alberto Latino