**Visual Question Answering**

Bianchi Chiara, Dominoni Lorenzo, Savino Matteo

For this third homework we implemented a model in which we have combined a CNN architecture with a RNN one in order to solve the visual question answering problem. In particular, with our network we were able to achieve our best score on the test set with an accuracy of 0.62193. Here we explain how we built it and which are the challenges we faced.

First of all we resized our images, built the data generator and implemented other functions in order to prepare the data to then feed our neural network. In particular in this part of our code we split the whole dataset into two parts, the training and the validation set, we match the questions with the corresponding images and answers and we create batches of them. We also create the tokenizer to convert words to integers.

After that we built our architecture from scratch with two different branches:

1. First we implemented a CNN in order to extract features from our images: we alternate convolutional layers with an increasing number of filters of dimension 3x3, ReLU, MaxPooling and Batch Normalization. In the end to flatten output, after CNN, a dense layer worked better than globalmaxpooling, even if with many parameters.

For this part of the network we also tried to use transfer learning with VGG and Efficientnet with complete fine tuning but in both cases with worse results with respect to our custom version. We think that this can be due to the fact that our problem and our images are very different from those used to train these two architectures. However we have reported the code we used in the notebook for completeness.

1. Then we built a RNN for the questions: we start with a custom embedding layer and we add bidirectional LSTM to catch all the context, low dropout, and a self attention mechanism to focus on the more important words in the questions, even if they are short it gave a boost to the performances.

For this part we tried to use some pre-made embeddings but also in this case we didn’t obtain improvements in the performances, so for the final results we used the custom version but in the notebook you can also see this part of code.

1. In the end we concatenate the feature vectors obtained by the two different branches and after a dense layer we perform classification with softmax.

To train the network we used early stopping to limit overfitting and decaying learning rate starting from 0.001 to refine the solution, both based on the metric val\_loss.

After the training, in order to understand how our model performs and which are the bigger problems, we introduced in our notebook some diagnostics plotting the images with the corresponding questions and answers for some data in the training and in the test set. Looking at the results we can conclude that our model is very good with the questions, in fact it is almost always able to understand what kind of answers it must return (for example it is able to understand if the correct answer is a number, a colour, yes or no, or other) but it is not so accurate in interpreting the image to get the correct result of a given type. We think that attention could be the key for this improvement, so we tried to introduce an attention mechanism to catch some correlation between questions and images and so to focus on some particular features on the images but we didn’t obtain better results, however in the notebook we have reported, commented, the code that we used. We also tried coattention, not reported in the notebook, very suited for vqa, but unfortunately also in this case without useful results.