Challenge 3

Visual Question Answering

Matteo Pacciani (Codice Persona 10525096 - Matricola 945104)
Oscar Pindaro (Codice Persona 10568511 - Matricola 946671)
Francesco Piro (Codice Persona 10534719 - Matricola 946450)

1 Problem Approach

In this challenge we use a multiple-input single-output model. The two inputs are an image and the relative question, while the output is the answer to the input question, evaluated on the image. Our architecture has a CNN for image processing and an embedding layer followed by a LSTM for text processing. The outputs of those two parts are then concatenated and fed to a dense, softmax-activated layer with a number of neurons equal to the number of the possible answers. The CNN is the VGG architecture pretrained on Imagenet dataset and with all its weights freezed. The embedding layer is initialized with GloVe 300d, that gave us the best results on the test set compared to the 50d and 100d.

For what concerns regularization, early stopping is used, in addition to a dropout layer after the dense one that follows VGG. We also tried Weight Decay, but it didn't make much difference. We even tried with a Global Average Pooling (GAP) layer after VGG to reduce the number of parameters, but it didn't improve performances.

Being ours a heavily unbalanced dataset, with 3 out of 58 answers occurring in 70% of the total questions, we thought that weighting classes could be a good idea. So we set the weights according to a very simple formula, the inverse frequency, scaled in a proper way in order not to alter the loss function value too much. Against our expectations, this led to decreased performances in both training and test phases.
We tried more complex models, with vertically stacked LSTM, or with more dense layers, but without success.

2 Unknown Words

We noticed that while training and validation accuracy were more or less the same, the one on the test set was way lower. So we thought that unknown words in the test questions could be a problem, because the **tokenizer** has to be fit only on the training set. So we introduced the out of vocabulary token, and set a threshold to word occurrences. Words not reaching this threshold were assigned the oov_token during training. The principle was that maybe the model would have learnt to cope with **unknown words**, instead of just removing them from the question. Unfortunately, this didn't improve accuracy on the test set, that stayed stuck 10/15% below the validation accuracy.

3 Nested Classification

Since the accuracy of every model was stuck at 58%, we thought that the model was just memorizing the most frequent answers: "yes" with 33% of frequency, followed by "no" (22%) and "2" (3%). Therefore, we tried to train two different classifiers. The first one had to understand if the answer to the question was "yes", "no", or any other question. The second classifier was trained on all the other classes. When the first model outputs the "other" class, the result of the second classifier is considered. Unfortunately, this approach led to a lower accuracy, probably because the first classifier was not accurate enough.

4 Conclusions

Our hypothesis is that the class imbalance present in the training set is affecting the capabilities of the model to generalize. 5-fold cross-validation results confirmed what we got on the test set.