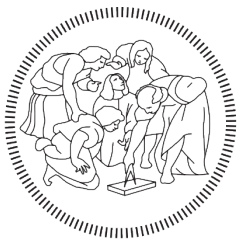


POLITECNICO
MILANO 1863

Homework 3

Visual Question and Answering

Artificial Neural Networks and Deep Learning

Authors

Ali Arslan - CP: *10807090*

Sorrentino Alessandro - CP: *10746269*

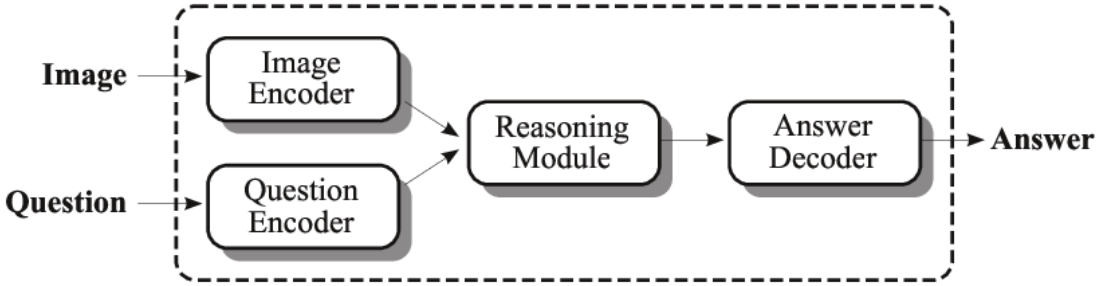
Menta Andrea - CP: *10636205*

General approach: For this third challenge, our team followed mainly 3 different kind of approaches: LSTM, Transformer and Pipeline (not presented).
Although we have tried a transfer learning approach (`VGG16` and `Inception-Resnet`) regarding the CNN part however that didn't give us neat performance improvement.

The custom CNN, used in all models is defined by the following blocks:

- Conv2D (3x3)
- BatchNormalization
- ReLU Activation Function
- MaxPool2D (2x2)
- Dropout(0.2)

The only difference is multiplying the number of filters by a factor of 2 resambling the VGG structure.



Dataset: The dataset has been divided 3 sets: *training*, *validation* and *test* with a 70% - 30% relation in the RNN model and a 80% - 20% relation in the Transformer model. Analyzing the dataset, we noticed an imbalance in the classes. We have 58 classes, divided into 3 possible categories: 'yes / no' answers, 'counting' answers (from 0 to 5) and 'other' (e.g., colors, objects, etc.). Numerically, the data relating to the first category are double compared to the other 2 categories, so we decided to implement a custom weighted loss function, using a vector of weights, where each element of the vector represents a class. Unfortunately this solution has not brought improvements in any model, keeping the accuracy value constant.

$$w_j = \log\left(\frac{\left(\frac{\#questions}{\#classes}\right)}{f_j}\right)$$

Where w_j is the weight to associated to the j^{th} class and f_j is the frequency of occurrences of the j^{th} class.

1 Recurrent Neural Newtwork (LSTMs)

Model A (test accuracy $\approx 63\%$)[1] Our first approach to the challenge was through a classic RNN architecture based on LSTM blocks. To boost the performance of the

[1] First model delivered

text analysis we applied a preprocessing:

- Transformed all words in lower case
- Removed some meaningless characters
- Performed *lemmatization*, in order to reduce related words to the same base form

Then we created our model as the composition of a CNN and a Word2Vec-like Model, fused with a point wise multiplication layer, followed by a classic feed forward part. The output, given the fixed answers, was performed using a softmax with 58 classes. To exploit the entire dataset and limit biased predictions we performed also a `K-Fold Cross Validation` (K=8) and a batch size of 256, ensembling then the K models in order to obtain an average prediction of the test set.
That gave us an improvement of $\approx 2\%$. Also Majority Voting, Data Augmentation (images related) and Transfer Learning (CNN related, with `VGG` and `Inception Resnet v2`) techniques were tested but resulted in a worse performing model (maybe because of the toon-like nature of the images). At the end we reached a mean val accuracy of $\approx 62\%$.

2 Transformer (Bert)

Model B (test accuracy $\approx 62\%$)[2] After finishing exploring recurrent architectures, we moved to the state of the art for natural language processing, the *Transformer* . We had to modify our custom dataset a little in order to make it compliant with transformer input parameters (also concerning attention masks). For the transformer, we used `Bert`[3] (uncased) and `BertTokenizer` as sequence tokenizer. A section of Bert was used regarding the words embedding. The overall network architecture remains very similar to the RNN one. The model takes three inputs: image for the CNN, question and attention mask for the transformer. Both outputs are then max-pooled and fed to a separate feed forward neural network. A point-wise multiplication is then performed, fusing both worlds of encoding of text and encoding of images. The output remains softmax with 58 classes. Batch size was decreased to 128 because of RAM limitations given by the dimension of the transformer. This gave us a very good result with val accuracy $\approx 62\%$ (pretty consistent with the test).
We also tried other Transformers architectures (such as `Bert large`, `RoBerta` and `RoBerta large`) but they didn't give us better results. Using K-fold like in the previous RNN, this value could have improved but because of various limitations (concerning training time and training power) we didn't try. However, we believe this could have had a big potential.

Please, consult the notebooks comments for further specifications.

[2] Second model delivered
[3] Model from Hugging Face repo