|  |
| --- |
| **Using Word Embeddings to Estimate Quality of Machine Translating Systems** |
|  |
| **Fabio Lopes, Felipe Costa, Jorge Pereira 20200597, 20201041, 20201085** |
|  |
|  |
|  |
|  |

Introduction

Machine Translation (MT) Systems have been widely studied and developed. They are now part of the world in different areas and used to solve different problems. Singh et. al., examined the most recent NLP models that have been applied on different NLP tasks. Evaluate the quality of Machine Translating systems is an important task and it is also part of different studies. Different metrics were proposed like BLEU [2] and METEOR [3]. This task can be addressed by comparing reference sentences (translated by humans) whit MT Systems outputs, and even reference free metrics that only use the source sentence and the MT output.

To compare those sentences a properly representation of the corpus needs to be created. This can be done using Bag of Words model, TFIDF, POS Tagging and other different methods. Words Embeddings it’s one method that create vector representations of the words that can better represent the words in a context.

This project aims to develop a Multi-Language Quality Estimation metric for Machine Translation Systems based on Words Embeddings. The embedding will be extracted using BERT (Bidirectional Encoder Representations from Transformers) and will be used as inputs to Neural Network that will learn the quality of the translation based on previous scores given by humans.

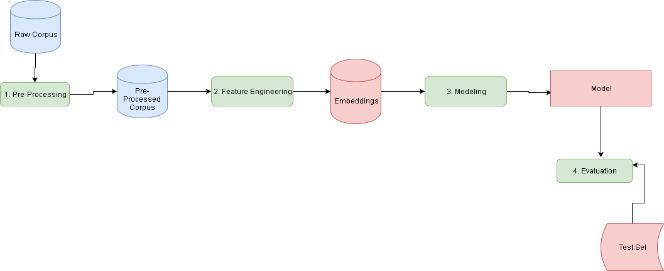
The corpus is omposed by six different translations samples:

* Russian into English
* German into English
* Czech into English
* Chinese into English
* English into Chinese
* English into Finish

The predicted scores must be well correlated with the human scores. Therefore, Pearson and Kendall correlations will be used to analyse the metric. The project was implemented using Python Libraries such as Pandas, Tensorflow and PyTorch.

Method/Approach

Four steps were defined, to organize the experiments [Fig 1]. Every step creates an output that will be used on the following step. This approach creates reusability and avoid to reprocessing all the steps every time.



**Figure 1 – Experiment’s pipeline**

The First Step is the Pre-Processing Stage: all the text are lowercased, html tags, punctuation and numbers are removed. For Chinese corpus every non-Chinese character was removed. The preprocessed data from this stage are then saved on csv files. This step was performed on both reference and translation columns.

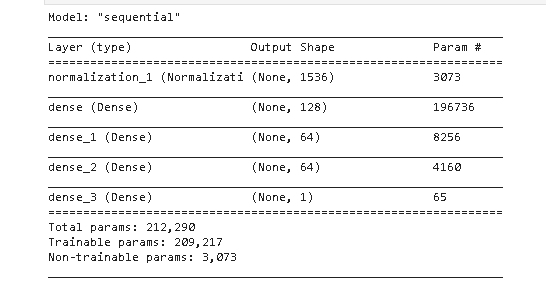
The second step is the Feature Engineering phase: the preprocessed data are loaded, and BERT are used to extract embeddings from the reference and translation. The embeddings are then concatenated and saved into Torch tensors files.

The corpus has diferente languages and for that reason LabSe was used. LabSe stands for Language-Agnostic BERT Sentence Embedding and allows to extract embeddings from different languages. The FeatureExtractor class was designed to extract embeddings and to compute the similarity between the sentences.

This stage was the most complicated one. Before start to use BERT, other approaches were also tested: TFIDF Features combined with POST Tagging and Word2Vec. But the computation time, memory allocation and also was also an issue. BERT was a feasible solution in terms of time consuming and computation time but took a long time to run and allocates a lot of memory. All those experiments consumed a large part of the project’s deadline.

The third step is the Modeling stage: the embeddings from the previous step were then used to create the models. Two models were analyzed and compared:

1. Baseline Model: It’s a vanilla model that consists in look only to the cosine similarity between the two embeddings
2. Deep Neural Network Model (DNN): A deep learning model that will try to predict the scores using the embeddings as inputs. [Fig. 2] show the model architecture with a normalization step and four dense layers.



**Figure 2 – Deep Neural Network Architecture (DNN)**

To train the DNN model all the tensors saved on the previous steps were loaded again and concatenated in one list. Then a train/test split was performed with a test size of 33%. A normalization layer was created and then four dense hidden layers. Adam was used as the optimizer and the network was trained with 100 epochs. Other architectures were also tested.

The final step ,the Evaluation, used the Baseline model to compute the cosine similarity and the DNN model to predict the scores on a totally unseen Test Set. Here several things were analyzed:

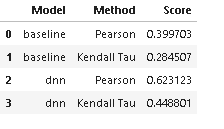
* The loss rate: the model should not have a lower loss rate because this can indicate large errors.
* Person and Kendall Tau correlation: The DNN model scores should correlates well with human scores
* The DNN model should perform better than the baseline model
* The model should have a similar performance on both training and test sets.

Results and Discussion

The baseline was a simple cosine similarity between reference and translation. It was calculated on each corpus dataset and saved on a new column.

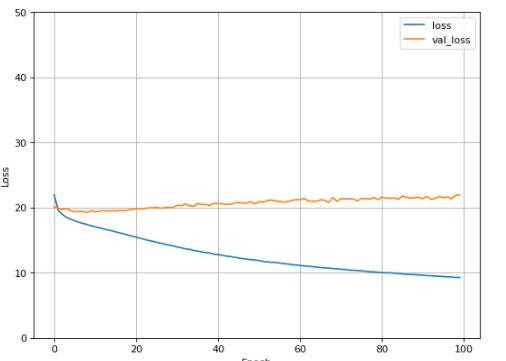
The DNN model was loaded from the file and then used to predict the scores to each dataset.

After running all the steps, the results for both models were computed and analyzed. On [Table 01], the results are summarized. It’s possible to observe that the DNN model correlates better with human scores than the baseline model.



**Table 01 – Pearson and Kendall Tau Correlation on Both Models**

However, the DNN model has a test loss rate of 21.72 which is a bit high. Also, during the training it was identified some overfitting: the validation loss is superior to the training loss [Fig. 3].



**Figure 3 – Training X Validation Loss**

Different architectures were tested to reduce the loss rate. However despite of decrease the loss rate the overfitting increased and the model didn’t perform well on the test data.

Conclusion

This project presented a Quality Estimation Metric to Machine Translations based on Word Embeddings. The Embeddings were extracted using BERT (LabSe). A Deep Neural Network was trained using the embeddings to predict the human scores. The model was compared against a baseline model that uses only cosine similarity to predict the scores. The final model achieved 0.62 Pearson and 0.44 Kendall Tau correlation with human scores.

The most challenging part was to deal with large datasets. Mostly the feature extractor phase took a long time to run and allocate a lot of memory. It will be important on the future look for methods to better address this situation.

Other approaches were tested on the beginning like use only TFIDF features, Word2Vec, POS Tagging. This experiment didn’t went well and consumed a lot of the time available to deliver the project. The final model could be tuned to avoid overfitting and perform better. In future studies other features could be added to the model like POS Tagging to increase the model ability.

References

[1]        Singh, Sushant; Mahmood, Ausif, “The NLP Cookbook: Modern Recipes for Transformer based Deep Learning Architectures”, IEEE Access, vol. 9, 2021.

[2]        Reiter, Ehud, “A structured review of the validity of BLEU”, Computational Linguistics, 2018, vol. 44(3).

1. Appendices

Appendices are material that can be read, and include figures and tables that are not critical to the reading and understanding of the report.