Deep Learning Assignment
Team 33

Tilburg University

Master Data Science and Society

Agapi Broupi 2048389
Hyun Seon Park 2036947
Lisa Koutsoviti Koumeri 2048041

CodaLab account name: TiU_MSc_DSS_DL_Group_33
Team name: KRxGR

04.10.2020

The purpose of this assignment is to train a model to correctly classify a sentence of original English text or a human translation from French into English. We were provided with a training dataset consisting of the sentences and the corresponding label for each sentence.

Pre-processing is one of the key components in text classification tasks in order to bring the text data into an analyzable format. For this reason, the text was normalized turning it into lowercase characters. The noise was also removed from the text by deleting the special characters and punctuations. In addition, the sentences were tokenized into words and we also declared the Out Of Vocabulary (OOV) tokens to get an index. The sentences were turned into a sequence by replacing each word with an integer and then they were padded in order to be of the same length. Count vectors were created where every cell corresponds to the frequency count of a particular term in a sentence. Additionally, Term Frequency - Inverse Document Frequency (TF-IDF) scores were computed for each term in the sentences and for three-term combinations. Moreover, the dataset was split into training (90%) consisting of 169,897 rows and validation set (10%) consisting of 18,878 rows. Finally, we used the GloVe (300d) algorithm in order to obtain the word representations in a high-dimensional space and an embedding matrix was created to capture the set of vectors that represent our words.

First, Logistic Regression and Naive Bayes were explored as classifiers using as input the frequency counts, the TF-IDF scores, and the word-level trigrams. Naive Bayes classifiers can create simple and good performing models in the field of document classification (Raschka, 2014). Logistic regression has been observed to yield better results in text classification than Naive Bayes (Vikarsa & Rinaldo, 2016) which was also the case in this project. Accuracy scores of all models can be found in Table 1.

Secondly, Recurrent Neural Network (RNN) models were selected for more fine-tuned results. As opposed to using traditional neural networks where the inputs and outputs are independent of one another, RNNs can perform better as they take into consideration the sequence of words. Among many different variants of RNNs, we tried a simple Long Short-Term Memory (LSTM) model and a Bidirectional LSTM model. Using LSTM will resolve the most prominent issues of RNNs, the vanishing/exploding gradients, and with its cell state and gates, the flow of information can be regulated efficiently. In addition to this, the bidirectional model will split the state neurons of RNN into forward states and backward states so that information from both the past and the future can be preserved.

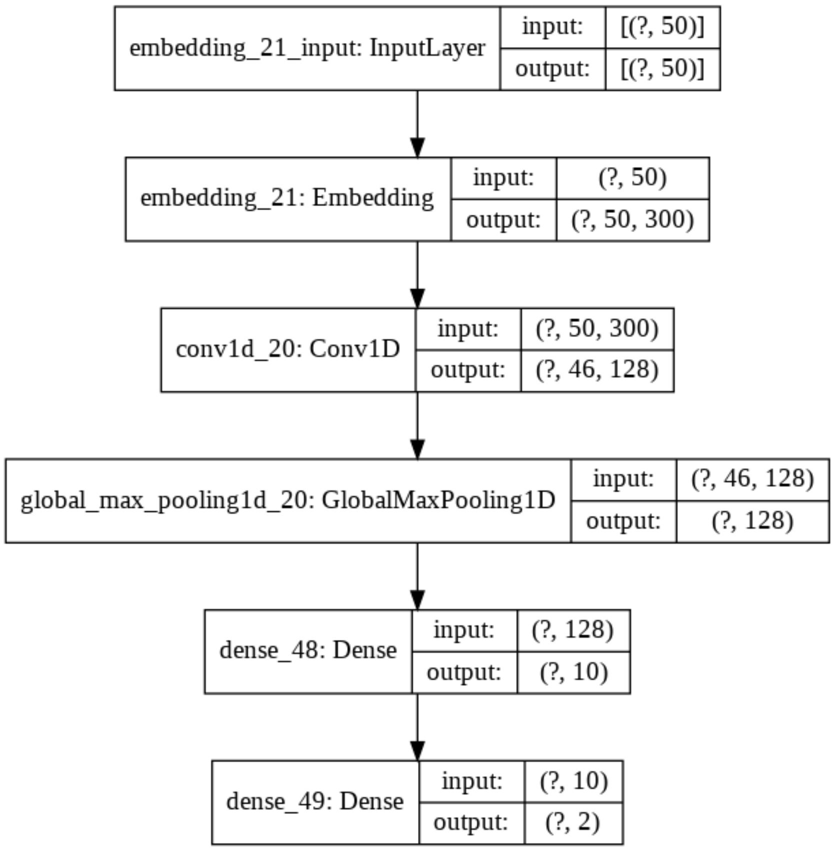
Lastly, a convolutional neural network (CNN) was applied since CNNs are also known to achieve excellent results in sequence processing (Kim, 2014). The Conv1D layer was used to create a convolution kernel that takes a patch of input features that are then multiplied with the filter weights. We varied the number of filters (128, 64, 32), the kernel sizes (3,5,7) and the batch sizes (128, 64, 32) when fitting the model. Adam was used as an optimizer with different learning rates (0.1, 0.001). Counterintuitively to the binary cross-entropy that is usually applied in binary classification, categorical cross-entropy yielded better results as a loss function. L2 regularization for the kernel and dropout layers were also tested, but a simpler version of the model was more successful as concluded by Kim (2014).

Concluding, the CNN model was slightly more successful than the RNN model. We initially expected RNNs to perform better since they recognize patterns across time i.e. long-term dependencies (Minaee et al., 2020). It seems though that in this project recognizing position-invariant patterns across space with CNNs might be a more successful approach.

Table 1: Results

Classifier	**Input**	**Accuracy score**
Logistic Regression	Vectorized sentences	75.6%
	Unigrams	74.2%
	Trigrams	71.1%
Multinomial Naive Bayes	Vectorized sentences	74.1%
	Unigrams	72.3%
	Trigrams	69.3%
RNN LSTM	Epochs: 5, batch size: 128	71%
RNN Bi-LSTM	Epochs: 5, batch size: 128	76.32%
CNN	Epochs: 3, batch size: 128	77.35%

Image 1: CNN design



References

Bansal, S. (2018, April 23). *A Comprehensive Guide to Understand and Implement Text Classification in Python*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/

Janakiev, N. (n.d.). *Practical Text Classification With Python and Keras*. Real Python. https://realpython.com/python-keras-text-classification/

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, *Association for Computational Linguistics*, 1746–1751.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2020, April 6). Deep Learning Based Text Classification: A Comprehensive Review.

Raschka, S. (2014). Naive Bayes and Text Classification I - Introduction and Theory. *arXivLabs*. 10.13140/2.1.2018.3049

Schuster, Mike & Paliwal, Kuldip. (1997). Bidirectional recurrent neural networks. Signal Processing, IEEE Transactions on. 45. 2673 - 2681. 10.1109/78.650093.

Vikarsa, L., & Rinaldo, T. (2016). Using logistic regression method to classify tweets into the selected topics. *ICACSIS 2016*. https://www.researchgate.net/publication/314667612

Zhou, Peng & Qi, Zhenyu & Zheng, Suncong & Xu, Jiaming & Bao, Hongyun & Xu, Bo. (2016). Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling.