Deep Text Complexity Metric Model

Advanced Projects of the Quality & Usabilty Lab, TU Berlin

Faraz Maschhur, Chuyang Wu, Max Reinhard

July 17, 2021

This report documents our model submission for the **Deep Text Complexity Metric Challenge** of the *Quality & Usability Lab of the Technical University of Berlin* in the summer term of 2021.

1. Task description

The task in this challenge was to predict the perceived complexity of German sentences with a (deep) machine learning model. For that, a **dataset of 900 German sentences** and their respective ratings was given. The ratings were given in form of MOS (Mean Opinion Score) values, which are **continuous numerical values from 1-7**. Those were collected by surveying L2 (second language) learners of German and further post-processing and expert assessment. Details about the dataset creation process can be found in [Naderi et al., 2019].
In the field of task categories which are being worked on in machine learning and specifically natural language processing (NLP), this task falls in the category of **sequence classification** or more exactly **sequence regression**.

2. BERT encodings

A central step in most current NLP methods is the numerical encoding of natural language (i.e. word, sentences, text documents) into a representation that can be further used in e.g. a classification model. One state-of-the-art approach to this problem is *BERT* [Devlin et al., 2018], which uses the *Transformer* [Vaswani et al., 2017] architecture to produce said representations. By now there exist several adaptations of the *BERT* model which try to improve it or cater to certain needs. Apart from the differences in model architecture, the performance of a specific model depends heavily on the training data, especially when it comes to its application for different languages. Due to the high resource demands in data, time and computing power to train a well performing language model with an architecture like *BERT*, it is common to build upon pre-trained models which are made publicly available.
To our knowledge, the state-of-the-art general purpose language models for German are the *GBERT* and *GELECTRA* models, which have been published in October 2020 [Chan et al., 2020]. The *HuggingFace transformers* library [Wolf et al., 2019] offers a convenient way to integrate Transformer-based models into custom models and applications. Pre-trained models can easily accessed via their model hub.[1]

3. Model architecture

Our submitted model follows the architecture for sequence classification as it is described in [Devlin et al., 2018]. The processes of predicting the MOS value for an input sentence consists of the following steps:

[1] `https://huggingface.co/models`

1. Tokenization: a) Tokenizing the sentence and truncating at $512 - 2$ tokens. b) Adding `[CLS]` and `[SEP]` tokens. c) Converting the tokens to integer ids and padding with `0` tokens to the maximum sequence length found in the dataset.
2. Passing the token ids through the twelve *GBERT* layers.
3. Pooling: a) Extracting the last hidden sate of the `[CLS]` token. b) Passing it through a dense linear layer. c) Applying a tanh activation layer.
4. Regression: a) Applying a dropout layer. b) Projecting onto a single float value with another linear layer.

A diagram of model our architecture can be found in appendix A, figure 1. This architecture is already implemented in the *BertForSequenceClassification* class[2] of the *transformers* library. As our base model we used `gbert-base`[3], the smaller of the two *GBERT* models. We found that with the larger model and the *GELECTRA* models there was a significantly higher variance in evaluation error when using different random seeds for weight initialization, data splitting and data shuffling.

4. Training process

Our training process follows the fine-tuning approach described in [Devlin et al., 2018]. The entire model is trained end-to-end using the mean squared error loss and the *AdamW* optimizer [Loshchilov and Hutter, 2017]. The hyperparamters of the training process can be found in appendix B, table 1. For development purposes, the dataset was randomly split 3-fold into a train set and test set with a test split size of $0.33 \cdot 900 = 297$. To train the final submitted model, the entire dataset was used as training set. The described training process can be easily realised with the *Trainer* class[4] of the *transformers* library.

5. Discarded models

In the process of developing the presented model, we tried out different model architectures and input features. We started by using sentence based representations generated with the *SentenceTransformer* [Reimers and Gurevych, 2019] library and fed those into different versions of a *MLP*. Then we added text-readability related features, which we extracted with the *readability* library[5] and filtered based on our own data analysis. Those features were concatenated with the former representations and fed into different regression models, including *Random Forest*, *Gradient Tree Boosting*, *SVM* and *MLP*.
Following that we tried a sequence oriented approach by generating token representations with the (frozen) *GBERT* model and fed them into an *BiLSTM* with a linear layer on top. We also tried extending this approach with the readabilty-based features by concatenating them with the last hidden *LSTM* state. The next step was fine-tuning the *GBERT* model on the regression task using the model and process described above. The *LSTM*-based approaches were approaches were also tested with the fine-tuned token representations.

[2] `https://huggingface.co/transformers/model_doc/bert.html#bertforsequenceclassification`
[3] `https://huggingface.co/deepset/gbert-base`
[4] `https://huggingface.co/transformers/main_classes/trainer.html`
[5] `https://github.com/andreasvc/readability/`

References

[Chan et al., 2020] Chan, B., Schweter, S., and Möller, T. (2020). German's next language model. *CoRR*, abs/2010.10906.

[Devlin et al., 2018] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

[Loshchilov and Hutter, 2017] Loshchilov, I. and Hutter, F. (2017). Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

[Naderi et al., 2019] Naderi, B., Mohtaj, S., Ensikat, K., and Möller, S. (2019). Subjective assessment of text complexity: A dataset for german language. *CoRR*, abs/1904.07733.

[Reimers and Gurevych, 2019] Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

[Wolf et al., 2019] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Appendices

A. Figures

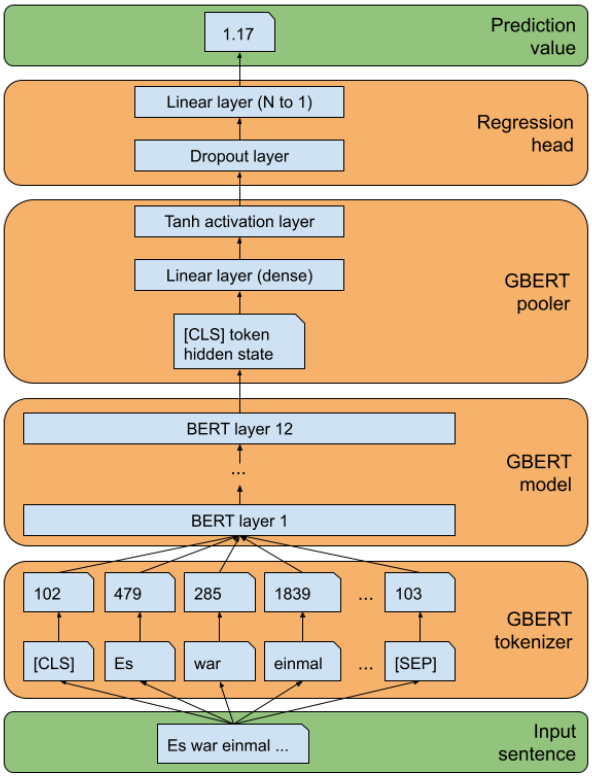

Figure 1: Diagram of the model architecture

B. Tables

hyperparameter	value
number of epochs	3
batch size	16
learning rate init	5e−5
learning rate decay	linear
optimizer parameters	default

Table 1: Hyperparamters for training (fine-tuning)