Assignment 2

Faharad Bayrami, Gianluca Di Mauro, Kaan Molla and **Leonardo Monti**
Master's Degree in Artificial Intelligence, University of Bologna
{ farhad.bayrami, gianluca.dimauro, mahmutkaan.molla, leonardo.monti3 }@studio.unibo.it

Abstract

Multi-label text classification using pre-trained models, like BERT, has been a task of major interest in the last years. This report delves into the exploration of classification of arguments on inherent human values underlying natural language arguments, with a specific emphasis on higher-order values such as "Openness to change," "Self-enhancement," "Conservation," and "Self-transcendence".

1 Introduction

The main interest was in developing three different models able to correctly classify arguments on human values, basing the evaluation only on the arguments' conclusions for the first, then adding also the premises in the second one and finally including the stances in the third one, which express if the specific argument is "in favor of" or "against" the specific discussed topic. The approach was first to utilize an auto-model to better understand the problem, then we moved on a custom model, in order to better address all the requirements. Operating on a train dataset of only 5393 entries, some inaccuracies have been faced, anyway promising results were obtained especially by the second and the third model, with F1-scores up to "0.75".

2 System description

The system runs entirely on Python. Six different datasets, three for arguments and three for labels, are downloaded and assembled into train, validation and test datasets. Then merging over categories is applied to obtain human values labels. Two baseline models were used: a majority classifier and a random uniform classifier. The aforementioned Bert based models, were all implemented on a pre-trained BertModel and for each, dataset tokenization, through bert-base uncased pre trained tokenizer, is applied with small differences. Worthy of mention some trials using an Huggingface "Automodel-for-sequence-classification" (Wolf et al., 2020). Despite decent results were obtained, switching to a custom model class was preferred. Like tokenization functions, training and evaluation functions have been tailor-made to the models, allowing to better operate on different parameters, like the loss function, which will be discussed in the next section. The models are implemented and trained using the Pytorch framework for reproducibility and experimental reasons. The architecture mimics the one of Huggingface automodels, it includes the Bert model and a linear classifier. An additional value is stacked to the Bert output when the Stance is used for the classification.

3 Experimental setup and results

To train the model we opted for the AdamW optimizer, which allows to tune the Weight Decay parameter. Then BCEWithLogitsLoss was used as loss function, after some trials using instead F1, with poor results. This loss combines a Sigmoid layer and the BCELoss (Binary Cross Entropy) in one single class. This version is more numerically stable than using a plain Sigmoid followed by a BCELoss as, by combining the operations into one layer, we take advantage of the log-sum-exp trick for numerical stability (Paszke et al., 2019). Due to train set class imbalance, applying weights to the loss function was beneficial. Weights were calculated using a custom function based on the square root of the inverse of the class frequency. In particular the first class is the least represented and a major weight of "1.666" was applied; other weights applied as follows: "1.529, 1.158, 1.188". This ensured improvements in predicting the first class. The last parameter needed for Adam to train the model is the learning rate. Initial tests with a learning rate of 2e-5 were done, following proven results (Sun et al., 2019). The best results on the less frequent classes were obtained that way. Further experiments were done raising the learning rate. Table 2 shows the results for learning rates of 2e-5 and 3e-5. Worth of attention the already mentioned Weight Decay, which didn't improve results, despite trying a wide range of values. All the results provided are averages over the results of three executions using different random seeds, to edge against training variability.

Classifier	Macro F1	Per-Category F1
Uniform	0.48	[0.39, 0.41, 0.58, 0.58]
Majority	0.49	[0.37, 0.46, 0.59, 0.56]

Table 1: Baselines' F1 Scores

Model	OtC	SE	C	ST	Macro
lr 2e-5					
C	0.40	0.59	0.85	0.85	0.67
C-P	0.57	0.67	0.85	0.85	0.73
C-P-S	0.56	0.65	0.85	0.85	0.73
lr 3e-5					
C	0.26	0.62	0.85	0.85	0.64
C-P	0.60	0.68	0.85	0.85	0.75
C-P-S	0.56	0.68	0.85	0.85	0.74

Table 2: Models' average F1 Scores per category and Macro

4 Discussion

Many experiments were made, allowing to progressively build a more well-suited model. First of all the two baseline models were tested; for both the random uniform classifier and the majority classifier, overall F1 score and per category F1 score were calculated, as shown in Table 1. Then, several tests were done using an auto-model, which produced unsatisfactory results, which are not reported not being relevant. On the other hand, the custom models surely performed better than both of them. The conclusion only model was the weakest of the three obtaining low F1 score values that are still better than the baselines. For the other two models the results improved which shows how including the premise was beneficial, while the stance didn't provide too much of a help to the models' accuracy. Results are reported in Table 2. In Figure 1 we compare the performances of the classifiers with the frequencies of the class in the training set. It is evident how the Bert model using only the conclusion with learning rate = 3e-5 overfits on the first category, that is the least represented. A similar issue is discussed in (Kiesel et al., 2022).

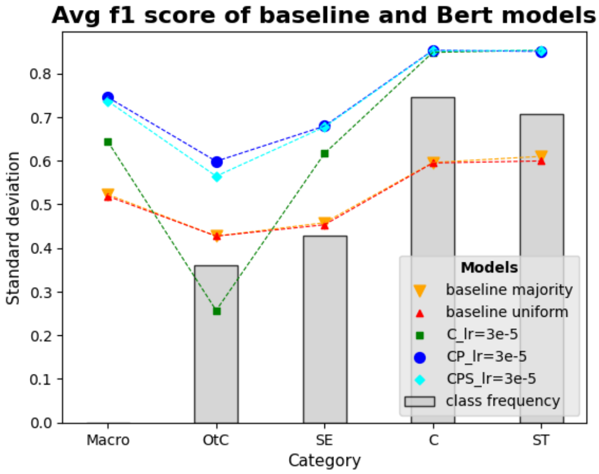

Figure 1: Average F1 scores of Baselines and Bert Classifiers, compared with class frequencies for each category.

5 Conclusion

The implemented models performed quite well, considering the limited data and resources available. The first model was the weakest of the three, being more prone to overfitting especially on the first category. The second model improved results by far, demonstrating that adding to the input arguments' conclusion was beneficial. Lastly for the third model, adding stance didn't provide a significant difference. What emerged, as is shown in Figure 1, is some correlation between value frequencies and classifier performances, this may be due to dataset being too small for training reliable classifiers on the infrequent values as suggested by (Kiesel et al., 2022) The main limitations were surely encountered due to a very small training set in terms of entries, but also in length of arguments provided. Possible future developments could consider further pre-training the bert model on the dataset, to suite the model to the specific implemented task as described by (Sun et al., 2019)

6 Links to external resources

All the materials of this assignments are available at the following GitHub repository

References

Johannes Kiesel, Milad Alshomary, Nicolas Handke, Xiaoni Cai, Henning Wachsmuth, and Benno Stein. 2022. Identifying the human values behind arguments. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4459–4471, Dublin, Ireland. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? In *Chinese Computational Linguistics - 18th China National Conference, CCL 2019, Kunming, China, October 18-20, 2019, Proceedings*, volume 11856 of *Lecture Notes in Computer Science*, pages 194–206. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.