Finetuning the T5 for Frame Identification

Oliver Tautz
Deep Learning for Natural Language Processing – SS21 – Philipp Cimiano and Philipp Heinisch

September 1, 2021

1 Introduction

Transfer learning with powerful, pretrained transformers is used to produce state-of-the-art results in many natural language processing tasks. (Wolf et al., 2020) For this assignement a relatively new transformer, the T5 (Raffel et al., 2020) is used. The T5, which will be described in more detail in chapter 3, differs from encoder only, BERT-like (Devlin et al., 2019) models as it uses an encoder-decoder architecture and can natively solve text-to-text problems. It was shown that many NLP tasks can be translated to such text-to-text problems and solved by a fine tuned version of the T5.(Raffel et al., 2020)

The proposed shared task by *webis* (Al-Khatib et al., 2021) is called frame identification. This is a classification task on full, natural text input. Each example text is an argument, which is labeled with its primary frame. The frame is a particular perspective of the argument which emphasizes an aspect of the associated topic. An example could be the frames 'economics', 'job market', or 'education' for an argument about a school reform. Frames are ambiguous and can be difficult to label even for humans. An automated system solving the task could be used by social scientists to gain more insight into what frames people respond to and evaluate what topics are important to the public or expose fraudulent abuse of framing in political debates.

As there is an almost unlimited number of frames a standard classification approach is limited to ones it has already seen and was provided enough data about. Thus the generative abilities of the T5 are explored for solving the task posed as a text-to-text problem, with mixed results, showing some promise but underperforming in comparison to already established techniques.

2 Related work

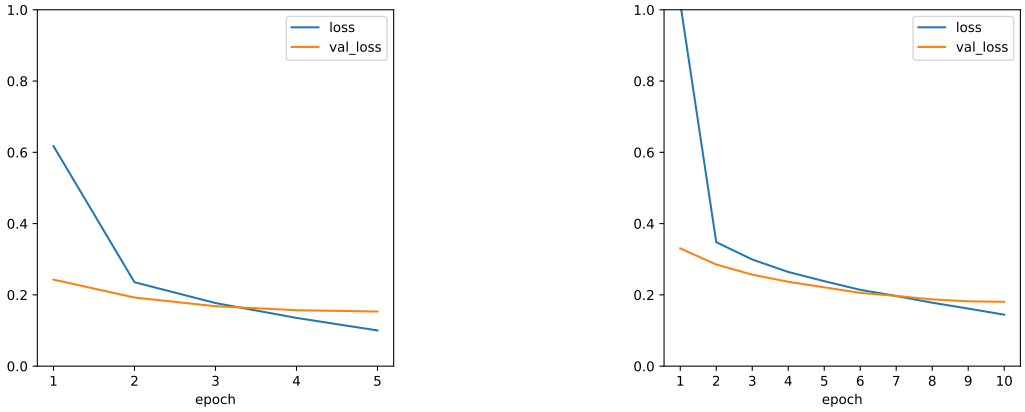
The approach implemented by the task organizers (Ajjour et al., 2019) uses an unsupervised clustering algorithm on heavily preprocessed input and achieves a macro weighted f1-score of 0.28.

Another established approach to solve NLP classification tasks is to use a pre-trained transformer to produce word embeddings from text and train a classifier on top of that , for example a KNN classifier. BERT was used(Sikos and Padó, 2019) to great effect on the (Das et al., 2014) dataset. Better results were achieved by also fine tuning BERTs embeddings, reaching 91.26 accuracy.

3 Method

3.1 Model

The T5 is chosen because we can solve frame identification as a text-to-text task, which removes the general limitation of only being able to predict frames already considered while labeling. With great generalisation the model could theoretically be able to predict frames on general arguments on topics not in the training set. It is an encoder-decoder transformer using a standard architecture with



(a) t5-base (b) t5-small

Figure 1: learning curves of crossentropy loss

stacked blocks of self attention layers and fully-connected feed-forward networks in the encoder. For the decoder a normal attention layer is added. Regularization mechanisms, namely dropout and layer normalization are used in addition to resnet-like skip-connections.(Raffel et al., 2020) Checkpoints trained on an unsupervised task on the large C4 (Raffel et al., 2019) dataset are used as a starting point for training.

The *keras* (Chollet et al., 2015) T5 implementation and weights of *huggingface transformers* (Wolf et al., 2020) are used.

3.2 Problem Definition

As the T5 can solve text-to-text tasks a very simple problem definition is chosen. Input **x** is the full argument as natural text, prediction target and label **y** is the text representation of the frame. A pair **(x,y)** is called an argument or sample subsequently.

3.3 Data

The dataset used (Ajjour et al., 2019) comes with a lot of fields per sample, most of which are ignored. Used fields are 'premise', 'conclusion' and 'frame'. Concatenating 'premise' and 'conclusion' forms input **x**. The frame field can be used without processing as label **y**. There are 12 326 labeled arguments in the dataset. The frames are very unevenly distributed with some labels being present only once, and the most prominent frame 'economics' counting 119 samples.

3.4 Preprocessing

Experiments adding the task specific prefix 'summary' (Raffel et al., 2020) to the input showed no effect. So minimal processing is used as described in 3.3. The dataset is split before training into train-, test- and validation set with a split ratio of 0.2, resulting in test-size of 2466 and val-size of 1973 respectively. The split is done in a stratified fashion, making a special case for samples with unique frames and distributing them according to the split ratio among train, test and val set. Arguments with rare frames, counting less than five samples are disregarded for training to achieve a more stable training signal, resulting in a train-set of 5351 samples. For training and predicting the input is tokenized by the T5Tokenizer (Wolf et al., 2020), also padding and truncating sequences to the T5s max length of 512 tokens.

Model	f1-macro	Accuracy	Rouge1-f1
t5-base 3 epochs	0.111	0.295	0.322
t5-base 5 epochs	0.192	0.429	0.449
t5-small 5 epochs	0.025	0.118	0.148
t5-small 10 epochs	0.127	0.316	0.338

Table 1: results on test set

3.5 Training Setup and Hyperparameters

Two models are trained, one starting from the provided, pre-trained checkpoint 't5-base' and one starting from 't5-small'. The 't5-small' is a smaller version of the T5. The base model is trained for 5 epochs, the small one for 10 epochs optimized by ADAM (Kingma and Ba, 2017) with a small, constant learning rate of 1×10^{-4} and cross entropy loss. A batch size of 6 and 24 is used for the base and small model respectively. Both batch sizes are smaller than in the T5-paper but are chosen to be suitable for training on *google colab*. Learning curves can be seen in figure 1. They show symptoms of overfitting after a few epochs for both models, quicker and more pronounced for the base model. Training is stopped early to improve validation and test performance.

4 Evaluation

4.1 Metrics

Three metrics are computed on the test set. To roughly compare to the BERT-classifier the 'full lexical accuracy', which means accuracy on all test samples, is computed. To compare to the task organizers clustering the macro-weighted f1-score is used. This is the mean of f1-scores per class, weighing them all equally regardless of size. For both the output is trimmed to text without padding and a prediction counts as correct only if the predicted and label string match. Because the T5 generates the frame text these are unsatisfying metrics as it can produce similar, yet not identical results. Therefore in addition the ROUGE-1 f1 (Lin, 2004) score is computed. This metric is usually used to compare the similarity of full text summaries but is also applicable here. It is less harsh, as e.g. the prediction 'popular opinion' for a 'public opinion' labeled sample would be counted as partially correct. The T5 could still predict semantically similar frames, which is disregarded by both metrics.

4.2 Results

Metrics computed on the test set can be seen in table 1. Although seemingly overfitting the training data, shown by diverging train- and validation-loss curves in figure 1 the models continue to improve their test scores. The small model takes longer to reach a reasonable score. Accuracy, albeit evaluated on a totally different dataset, is low across the board compared to the BERT-classifier (Sikos and Padó, 2019) , reaching a maximum of 0.43 for the base model. ROUGE scores are only marginally higher, meaning most prediction hits are accurate and not diffuse ones. The best model achieves an f1-score of 0.192 on the test set, also notably worse than the clustering approach of (Ajjour et al., 2019). The bigger model reaches significantly higher scores than the 't5-small', both still unusably low. Training time was relatively short, with the base model needing 440s/epoch and the small one taking only 140s/epoch on a *Tesla P100* with 16GB of VRAM.

5 Conclusion

The fine tuned T5 underperformed in comparison to both established classification models, but the results are still promising. In search of a general classification approach, predicting text instead of a class removes the limit of predefined classes and could leverage language understanding of machine learning models on another level. On this particular task there are lots of opportunities to explore.

The rapid overfitting problem which is typical for high capacity model finetuning should be met with different techniques such as learning rate schedules, data augmentation, more balanced datasets and 'gradual unfreezing' which were out of scope for this assignment.

Without those techniques it was not possible to produce usable results. Generating the frame text seems to be a more difficult problem, requiring a better adapted model or training routine to be solved. The small model exhibited less overfitting problems but its reduced capacity did not seem fit to the task. The normal 't5-base' improved quickly but could not generalize well enough, overfitting quickly. For subsequent research the full 't5-base' or even the here not explored 't5-large' model is recommended .

References

Ajjour, Yamen et al. (Nov. 2019). "Modeling Frames in Argumentation". In: *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP 2019)*. Ed. by Kentaro Inui et al. ACL, pp. 2922–2932. URL: https://www.aclweb.org/anthology/D19-1290.

Chollet, François et al. (2015). *Keras*. https://keras.io.

Das, Dipanjan et al. (Mar. 2014). "Frame-Semantic Parsing". In: *Computational Linguistics* 40.1, pp. 9–56. ISSN: 0891-2017. DOI: 10.1162/COLI_a_00163. eprint: https://direct.mit.edu/coli/article-pdf/40/1/9/1812895/coli_a_00163.pdf. URL: https://doi.org/10.1162/COLI%5C_a%5C_00163.

Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv: 1810.04805 [cs.CL].

Al-Khatib, Khalid et al. (2021). *Touché @ ArgMining21 Frame Identification*. https://webis.de/events/argmining-21/. Accessed: 2021-08-31.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG].

Lin, Chin-Yew (July 2004). "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, pp. 74–81. URL: https://aclanthology.org/W04-1013.

Raffel, Colin et al. (2019). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *arXiv e-prints*. arXiv: 1910.10683.

— (2020). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. arXiv: 1910.10683 [cs.LG].

Sikos, Jennifer and Sebastian Padó (Apr. 2019). "Frame Identification as Categorization: Exemplars vs Prototypes in Embeddingland". In: DOI: 10.18653/v1/W19-0425.

Wolf, Thomas et al. (Oct. 2020). "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.