Final Report - Distillation of HerBERT

Bartek Krzepkowski[*]
b.krzepkowski@student.uw.edu.pl

Dominika Bankiewicz
d.bankiewicz@student.uw.edu.pl

Jacek Ciszewski
jc394143@students.mimuw.edu.pl

Rafał Michaluk
rm360179@students.mimuw.edu.pl

Spyridon Mouselinos
Supervisor
mouselinos.spur.kw@gmail.com

1 Introduction

Transformers are used as state-of-the-art solutions for various NLP tasks. In recent years, there is an increased interest in language models based on transformer architecture for specific languages. HerBERT is a BERT-based model trained on Polish language datasets. It achieves SOTA results on various NLP tasks (from KLEJ benchmark [Rybak et al., 2020]). We expect that Polish NLP models will be used much more frequently in the near future in the industry, due to the growing demand for data processing in the Polish language, for example in the area of e-commerce. However, one of the obstacles related to large models is the difficulty of adapting them to smaller computational infrastructure. In recent years many novel techniques that decrease the size of models (DistilBERT), increase their accuracy (Chinchilla model [Hoffmann et al., 2022]) and reduce training time (Chinchilla model) were developed. Nonethless, they have not been applied to HerBERT yet.

The aim of this study is to test the possibility of distilling the pretrained HerBERT-base model in order to obtain a model of smaller size. As the training set used in HerBERT has not been made available (neither the applied preprocessing), dataset CC-100-pl [Conneau et al., 2019] has been used instead (a set of Common-crawl snapshots available at huggingface). A series of ablations was performed to analyze arbitrarily selected factors that could affect the quality of the smaller model.

The code relevant to this study is publicly available on github repository[1].

The current project is an extension of the final project [2], for an NLP course conducted at MIMUW.

[*]Main contributor.
[1]https://github.com/BartekKrzepkowski/DistilHerBERT_vol2
[2]https://github.com/DistilHerBERT/DistilHerBERT/blob/main/report/DistilHerBERT_report.pdf

2 Related Work

Our experiments are based on the following research:

HerBERT [Mroczkowski et al., 2021]: BERT-based model pretrained on large Polish corpora. This model achieves state-of-the-art results on multiple downstream tasks such as KLEJ [Rybak et al., 2020].

Training Compute-Optimal Large Language Models [Hoffmann et al., 2022] : Proposing a power-law describing the relationship between the number of tokens (D) in a dataset and the number of parameters in the model (N), which was established ($\frac{\#D}{\#N} \approx 20$) on the basis of experiments carried out on the basis of the analysis of 400 language models, ranging from under 70M to over 16B parameters, and trained on 5B to over 400B tokens. We aim to re-use setting and findings from there in our experiment to further improve the accuracy of the distilled model on chosen downstream tasks. We decided to use AdamW optimizer, a cosine schedule with warmup, the precision reduction, and a ratio of the number of tokens to the number of parameters of approximately 23.

DistilBERT [Sanh et al., 2019] : BERT-based model pretrained on English corpora, uses knowledge distillation. It is a compression technique where a compact model (student) is trained to reproduce the behavior of the larger model (teacher). DistilBERT reduces the size of the original BERT model by 40% without significant loss of accuracy compared to BERT.

CC100-pl [Conneau et al., 2019]: corpus crated by processing January-December 2018 Commoncrawl snapshots. The dataset is mainly intended to pretrain language models and word representations. A subset of size of approximately 9GB of this dataset is used (first 50M samples). Then the selected subset was subjected to a cleanup, viz we have implemented a program that automatically checks samples and cleans out HTML code, sentences shorter than 15 characters, eliminates samples with ratio of non-alphanumeric characters bigger than 10%. At the very end, we deduplicated the rest of the set. This subset is referred in the study as CCsub (it consisted of tokens in total).

DeepSpeed [Jeff Rasley, 2020] : deep learning optimization library, developed at Microsoft, which we use to effectively use GPU memory during training, which translates into a larger single batch during gradient accumulation, and thus faster training. For this purpose, we use gradient partitioning (ZeRO stage 2) [Samyam Rajbhandari, 2019].

3 Model

The distillation procedure is composed of the teacher and the student model.

3.1 Teacher architecture

We use base version of HerBERT [Mroczkowski et al., 2021]. Its architecture follows the original BERT [Devlin et al., 2018] model: 12 layers, 12 attention heads and hidden dimension of 768. It has around 124M parameters and weighs 475 MB. In this report we discuss its two versions used for distillation. In the first one (referred as T1) we use exactly the same weights as the original HerBERT (we have used hugging-face "allegro/herbert-base-cased"[3]). The second one (referred as T2) is also HerBERT, but additionally trained on a subset of cc100-pl dataset [Conneau et al., 2019] (CCsub).

3.2 Student architecture

Student architecture is the same in all our experiments. It is a version of the teacher reduced in size, with half of the layers removed (it has 6 layers, 12 attention heads). It has around 81M parameters, which is 65% of parameters of the teacher. It weighs 325.34 MB. We use two different initialization of the student in our experiments. The first one is just random initialization using default pytorch constructors (training the student without previous knowledge). Second is based on the teacher's weights: as the student has 6 layers and the teacher has 12, before experiments, we copy weights from every second layer from teacher to student, starting with the first layer (indexed by 0) of the teacher.

[3]`https://huggingface.co/allegro/herbert-base-cased`

4 Experimental Set-up

4.1 Preprocessing

Examples from the CCsub set were treated with BPE tokenization, which was made available with the HerBERT model. In order to effectively use the input to the model, because, in particular, the calculation of the attention matrix in transformers has a square computational complexity due to the input length (in the HerBERT model the maximum input possibility is 512 tokens), the resulting token strings were concatenated as follows: 1. Each entry has no more than 512 tokens. 2. If the number of tokens from the cut-off point (which was designated as the end of the last word before the 512 token marker) to the beginning of a given text was less than 40, then the beginning of the given text was moved to the next row. 3. If the condition from the second point was not met, the given text was split, and the second part of the tokenized text was added to the next row, as long as it was not shorter than 40.

The texts tokenized in this way were separated by a separator token. Each training example starts with a class token. Thanks to this, from $43.6M$ of samples in the CCsub set, we got $3.8M$ of tokenized inputs to the model, 512 long (results approximately up to $0.1M$). The average length of the relevant entries was thus 506.59 tokens. The processed dataset contains $1913M$ tokens (approximately to $1M$). The data was then separated into the training set and the test set so that the test set contained 1% of the set before splitting.

4.2 Computing environment and training loop

Every model pretrained with either MLM loss or Distil loss are trained for 2 epochs on a single Nvidia Titan V GPU (12 GB) for approximately 6 days. During this training, the accelerator wrapper from the accelerate library was used to reduce the precision to fp16.

Additionally, gradient accumulation was used to perform an optimizer step on a batch up to 2100 examples, and a cosine schedule with warmup which included 10% of all steps in the training, with max learning rate of $5e-5$. We set the weight decay in the AdamW optimizer to $1e-3$.

It should be noted that the distillation process, if any, occurred at the same time as masked language modeling, according to the procedure described in the paper [Sanh et al., 2019]. It should also be mentioned that full word masking has been applied similar to the training of the Herbert model. A gradient norm clipping was also used with the max norm parameter of 3.

In the case of pretraining with distillation, the temperature parameter was used for the case of distillation during the testing phase, according to [Devlin et al., 2018], equal to 1, and during the training phase this parameter was 3.

The experiments consisted on finetuning models on subset of KLEJ benchmark (NKJP-NER, PolEmo2.0-IN and PolEmo2.0-OUT). In order to achieve finetuned model, the architecture of the student model was enlarged by a classification module (dropout and linear layer). Models chosen for the experiment differed in initialization of weights (random or truncated from the teacher model - HerBERT base model), the loss selected during the training of the distillation (MLM loss,Distillation loss or MLM with Distillation Loss) and usage of pretrained teacher. The following scenarios were chosen for student:

S0: Its weights were initialized randomly. No additional pretraining.

S1: Its weights were initialized randomly and pretrained from scratch on CCsub.

S1: Its initial weights were based on the teacher's weights. No additional pretraining.

S3: Its initial weights were based on the teacher's weights. The student was trained using distillation technique on the teacher with only MLM loss (no distillation loss). In this experiment we used HerBERT base as the teacher (T0).

S4: Its initial weights were based on the teacher's weights. The student was trained using distillation technique on the teacher with distillation loss described in DistilBERT. We used HerBERT base as the teacher (T0).

S5: Its initial weights were based on the teacher's weights. The student was trained using distillation technique on the teacher with distillation loss the same as in DistilBERT. We used HerBERT base, with additional previous training on CCsub (T1) as the teacher.

Table 1: Experiments description.

Model name	Init	MLM Loss	Distillation Loss	Pretrained Teacher
S0	Rand	X	X	X
S1	Rand	✓	X	X
S2	Trunc	X	X	X
S3	Trunc	✓	X	X
S4	Trunc	✓	✓	X
S5	Trunc	✓	✓	✓

The models description is summarized in Table 1. The column called "MLM Loss" distinguishes whether the model has been pretrained on a CCsub dataset, using the Masked Language Model Loss. The column "Distillation Loss" is a question whether the model has been distilled, with the help of a teacher, which is the mentioned HerBERT-base model, and also with the above-mentioned dataset. This distillation was carried out in accordance with the paper [Sanh et al., 2019]. The "Pretrained Teacher" column is the question of the teacher's adjustment to the CCsub.

Each model was finetuned separately on each dataset, therefore each model was trained 3 times, with initialization for classifiers appropriate for the huggingface implementation. (The seed 42 was used throughout the experiment.). The accuracy metric was chosen to compare the results. The training lasted for 20 epochs with batch 32 and the model was evaluated after each epoch. The results are calculated as a maximum of accuracy across all epochs.

5 Results and Discussion

The results of the finetuning of the aforementioned student-models, on chosen tasks from KLEJ benchmark, are presented in Table 2 Each row represents the description and accuracy score on datasets for a given model from the experiments. Additionally, the table shows the result of the teacher T1 model (HerBERT base) on the datasets, according to the result from the leaderboard of KLEJ benchmark[4].

[4]https://klejbenchmark.com/leaderboard/

Table 2: Experiments results.

Model name	PolEmo2.0 IN	PolEmo2.0 OUT	NKJP NER
S0	80.22	55.84	47.55
S1	X	X	X
S2	86.58	74.9	90.21
S3	X	X	X
S4	90.04	75.3	**90.88**
S5	**90.32**	**75.51**	90.52
T0	89.9	**79.35**	92.68
T1	89.9	77.33	**93.1**

As can be seen from the table, the model S4 had the highest results on NKJP-NER and PolEmo2.0-IN among the models from the experimental scenarios. Model S3 had the second highest accuracy score on PolEmo2.0-OUT dataset. As expected, the model initialized with random weights S0 and without pretraining phase performed worse than any other model considered. The usage of MLM and Distillation loss improves the performance of the model.

Our research suggests that distilling an pre-trained T1 model on a different CCsub dataset than the original dataset (obtaining S3) on which the model was originally pretrained differs slightly in effectiveness, on KLEJ benchmark, from the model which was created from the extraction of even-numbered layers from the T1 model, and then when it is retrained on the CCsub set, in favor of the former.

Moreover, our research suggests that additional pretraining of the T0 model, on CCsub, further improves the quality of the resulting smaller model S5. However, the enhancement caused by pretrained teacher is not visible for every dataset.

6 Conclusions

It should be taken into account that the S0-5 models training were not long, compared to the training needed to train in comparison to the distilBERT model [Sanh et al., 2019]. The authors of this paper are not sure if the pre-training and finetuning phase is properly optimized in terms of regularization. Therefore, further investigation of the parameters related to the data or the training itself is required.

The usage of pretrained teacher is open to further investigation. The next stage should include better processing of the text, a more diverse collection of text data, similar to the one on which HerBERT was taught, and a longer period of pretraining the models. It is suggested in [Conneau et al., 2019] that the number of tokens should be approximately 20 times larger than the number of parameters. This fact and the extension of the training time may be the reason for better results than those obtained on the first attempt.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Olatunji Ruwase Yuxiong He Jeff Rasley, Samyam Rajbhandari. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. *https://dl.acm.org/doi/10.1145/3394486.3406703*, 2020.

Robert Mroczkowski, Piotr Rybak, Alina Wróblewska, and Ireneusz Gawlik. Herbert: Efficiently pretrained transformer-based language model for polish. *arXiv preprint arXiv:2105.01735*, 2021.

Piotr Rybak, Robert Mroczkowski, Janusz Tracz, and Ireneusz Gawlik. Klej: comprehensive benchmark for polish language understanding. *arXiv preprint arXiv:2005.00630*, 2020.

Olatunji Ruwase Yuxiong He Samyam Rajbhandari, Jeff Rasley. Zero: Memory optimizations toward training trillion parameter models. *1910.02054*, 2019.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.