Leveraging PET for Few-Shot Learning in Low Resources Languages

Alexis Emanuelli*
M1 Cogmaster
University Paris Cité
alexis.emanuelli@psl.eu

Mathias Vigouroux*
M1 Cogmaster
ENS Paris
mathias.vigouroux@ens.psl.eu

Abstract

The article "It's not just size that matters" by Schick and Schütze (2020) [10] presents Pattern Exploitation Training (PET), a framework that reformulates all tasks as a language modelling problem. PET uses a fine-tuned language model to label unlabeled data, which is then used by a classical classifier, allowing for small training datasets and few-shot learning. We explain the process of PET, including patterns and verbalizers, and replicate results on some dataset of GLUE in English and the Allocine dataset in French, as well as on Hungarian and other datasets with different mask language models.

1 Introduction

Language models like GPT-3 and GPT-4 have achieved remarkable performance on natural language processing tasks, but they require a lot of computational resources. Pattern Exploitation Training (PET) offers a less computationnally expensive approach that reformulates all tasks as a language modelling problem. PET uses predefined *patterns* and *verbalizers* to turn features into sentences and labels into words, and fine-tunes language models on unlabeled data. The resulting soft-labelled dataset can be used by a classical classifier, allowing for few-shot learning and small training datasets. PET, as a result, necessitates the use of unlabeled data. However, obtaining unlabeled data is often simpler than obtaining labeled examples in various real-world applications.

2 The PET algorithm

2.1 Pattern Verbalizer Pair

Pattern transforms an input feature into a close-style question with a "$\langle mask \rangle$" token representing a masked word for the language model to predict.

For example, in sentiment classification, the input feature x (a text like "This film was a real masterpiece") becomes Pattern $P(x)$: "The film was a real masterpiece. This film was "$\langle mask \rangle$"".

The input also has a label $y \in 0, 1$, with a verbalizer v that maps y to natural language words used in the dataset: $v(0) =' good'$ and $v(1) =' bad'$.

These sentences will then be used to fine tune a Masked Language Model (MLM) for instance BERT [4]. Several *Pattern Verbalizer Pairs* (PVP) can be used. For each of these patterns, a MLM will be fine tuned. This first step can be also seen as an amplification procedure.

2.2 Training "small" LM with PVP

For each PVP p, a MLM is finetuned on training examples (x, y) by minimizing the cross entropy between the true label y and the probability distribution over labels from the LM. More formally if V is the vocabulary of our LM. Then, for each sequence $z \in V^\star$ that contains exactly one mask token and $w \in V$, $M(w|z)$ denotes the unnormalized score that the language model assigns to w at the masked position.

Given some input x, the score for label y is defined as :

$$s_p(y|x) = M(v(y)|P(x)) \tag{1}$$

This scores will then be normalized into a probability distribution over labels using a softmax and will be denoted $q_p(y|x)$.

This will be then used to finetune the model M for p thanks to the cross-entropy between $q_p(y|x)$ and the true one-hot distribution of the label y.

2.3 Combining PVP to soft-label

The ensemble of finetuned MLMs, denoted $\mathcal{P}$ is used to annotate a set of unlabeled examples; each unlabeled example $x \in \mathcal{D}$ is annotated with soft labels based on a probability distribution composed of a weighted average, a softmax of the scores given by each pattern. The weights are proportional to the accuracy achieved with p on the training set.

2.4 Using the soft-labeled dataset

The resulting soft-labeled dataset is used to train a regular sequence classifier by minimizing cross entropy between its output and q_p.

Combining the PVPs to annotate an unlabelled dataset and then use this newly labelled dataset to train again a LM, this time finetuned with a standard sequence classification head, ressemble a lot *Knowledge Distillation*. [9] simply refer to this step as *distillation*.

3 Experiments

Our goal was to :

- being able to reproduce the results for different tasks used in [9][10]
- define new Pattern Verbalizer Pairs to leverage PET for few-shot learning on new data-set, and look at the impact in performance arising from those variations.
- incorporate new Pretrained Language Model, from French (fewer resource language than English) and Hungarian (very low resource language), to see the impact of the language and of the PLM on the performances.

3.1 Trying to replicate the results

The first step was to get familiarized with the entire code and how it was organized.

To test the provided code, and our understanding of it, we started by using BoolQ, a natural language processing dataset that consists of questions and answers labeled as true or false. Our goal was to reproduce the fine-tuning of a pre-existing language model (alBERT) on this dataset, in order to improve its accuracy at predicting whether a given statement is true or false.

During the course of our research project, one of the major challenges we faced was the high computational cost of training our model on a GPU. Unfortunately, we had limited access to powerful hardware, which made it difficult for us to replicate the methods described in the research paper. As a result, we had to explore novel approaches that could be implemented within our available resources. For instance, we significantly reduced the number of maximum epochs used in our model, which allowed us to train it on Google Collaboratory (the execution time for a single cell is limited to 2 hours). This came with its own set of challenges. Most of the packages used in the study were dated 2021 and could only be run on Python 3.8, whereas colab was running 3.9. We had to find a way to locally downgrade the Python version, which took a lot of time to figure out. In the end, this configuration allowed for easier code sharing among us.

This limitation reflected in our results, since for this dataset we obtained a 63% using alBERT as a LM, when in the initial paper the accuracy was of 79%. We then extended our trials to new tasks we knew we could transpose in Hungarian : RTE and COPA for which we obtained an accuracy of respectively 59% and 43%.

Furthermore, we also tested replacing alBERT with BERT. To have a way of comparing performances of PET algorithm across languages, we also used XLM-roBERTa as underlying MLM.

MLM \ Task	BoolQ	COPA	RTE
alBERT	0.63	0.43	0.59
BERT	0.62	0.54	0.46
XLM-roBERTa	0.62	0.45	0.55

Table 1: Accuracies for the English datasets.

3.2 Towards the use of the algorithm on another language : French

Our project had initially aimed to use low resource language models for the Hungarian language, which

is one of the few non-Indo-European languages in Europe.

However, since only one of the two students on the team was proficient in Hungarian, we decided to conduct a preliminary experiment in French to become more familiar with the algorithm's mechanisms.

We then had to find a french dataset. To do so, we used the Hugging face website giving a nice preview of dataset for Natural Language Processing[14]. Following, the most intuitive task used by [9], we decide to used the Allocine dataset [2]. Similarly to yelp review, or SST, this dataset contains film review extracted from a french online plateforme and is annotated for sentiment analysis on the film.

3.2.1 Defining a new PVP

Next step was to define a new Pattern-Verbalizer Pair (PVP) suitable for the sentiment analysis task on the Allocine dataset in French.

Defining the Pattern Two patterns were implemented. Let's denote by x an input, which is a film review. The two patterns were defined as follow :

- $P_0 := \langle mask \rangle$! x
- $P_1 :=$ En résumé, ce film est : $\langle mask \rangle$

Defining the verbalizers The verbalizer maps the label to a token that exists in the model's vocabulary. Referring to the documentation reveals that a value of 0 represents a negative review, while a value of 1 corresponds to a positive review.

Here are the two slightly different verbalizers defined to test the impact in the semantics arising from the pre-trained LLM:

$$v(y) = \begin{cases} "nul" & \text{if y=0} \\ "bon" & \text{if y=1} \end{cases} \quad (2)$$

and the following one was also tested :

$$v(y) = \begin{cases} "terrible" & \text{if y=0} \\ "excellent" & \text{if y=1} \end{cases} \quad (3)$$

3.2.2 French pre-trained language models

We focused our attention on two MLM for our French transposition of PET algorithm. The first model we examined is XLM-RoBERTa[3], which is a cross-lingual version of the RoBERTa model. Compared to BERT, RoBERTa has dynamic masking, which means that different parts of sentences are masked during different epochs, making the model more robust. Additionally, RoBERTa was trained on a larger dataset than BERT and has a larger batch size. The second model we looked at is CamemBERT[6], a French-trained model based on the same architecture as BERT. We found CamemBERT on Hugging Face, and because it's part of the transformers package, we were able to easily import and use it. Both XLM-roBERTa and CamemBERT performed very well on our binary classification task.

MLM \ Task	Allociné
camemBERT	0.91
XLM-RoBERTa	0.93

Table 2: Accuracies for the French dataset

The PVP used previously were directly adapted from the PVP used in English in the paper. To double check this first intuition a simple application using transformers models to predict next word or a masked word in a sentence offers a nice Graphical User Interface [12].

3.3 Trying PET in Hungarian

To test the effectiveness of PET on Hungarian language tasks, we used three different datasets: HUCOPA, huSST, and HuRTE. These dataset are all part of the Hungarian Language Understanding Evaluation Benchmark Kit HuLU (Hungarian Language Understanding) [5]. This Kit is derived from the General Language Understanding Evaluation, GLUE [13].

3.3.1 Tasks

huSST is a dataset for the Hungarian version of the Stanford Sentiment Treebank. This corpus was created by translating and re-annotating the original SST [11], which was in English. This task is similar to the The Allocine task, as it is also a sentiment classification task on film review.

huRTE is a dataset for the Hungarian version of the Recognizing Textual Entailment datasets. The corpus was created by translating and re-annotating the instances of the RTE datasets [1] that are part of

the GLUE benchmark. Two sentences are compared and the model needs to infer if the second sentence is entailled by the first one or not.

huCOPA is a dataset for the Hungarian Choice of Plausible Alternatives Corpus (HuCoPA). The corpus was created by translating and re-annotating the original English CoPA corpus [8]. Due to the unique structure of the COPA task, huCopa required a specific modification to the PET algorithm to handle multiple masks.

3.3.2 Extending to other pre-trained Language Models

Initially, the MLM used was XLM-roBERTa as it is the only Cross-Lingual model implemented for PET. However, since we discover that it did not perform well on the Hungarian language, the huBERT model was found to be a better fit. This model, which is BERT-based, was specifically trained on a large corpus of Hungarian text. This model is ccesible on Hugging-face [7]. To access the pre-trained huBERT model weights, we used the .from_pretrained() method.

3.3.3 Results

MLM \ Task	HUCOPA	HuRTE	HuSST
huBERT	0.44	0.58	0.029
XLM-roBERTa	0.59	0.55	0.028

Table 3: Accuracies for the Hungarian datasets

It is unclear which of the two models that can handle Hungarian have a better accuracy.

Indeed, on one hand XLM-roBERTa perfomed better in the COPA task, yet hubert performed better in the RTE and SST tasks. However, the rise of accuracy in the COPA task with XLM-roBERTa is larger than in the two other tasks for huBERT. However, one significant advantage of huBERT is its faster training time, as it took only around 30 seconds for one iteration compared to 2 minutes and 40 seconds for XLM-roBERTa on our computer with a CPU.

4 Conclusion

In conclusion, our project aimed to replicate the study of [10], but due to hardware limitations, we had to shift our focus towards comparing different MLMs and defining new tasks that can be handled by PVP. Although the tasks we tried from the SuperGLUE dataset yielded low accuracies, sentiment classification tasks seemed to yield far better results even with extremely reduced parameters. We believe this is due to the fact that sentiment classification relies more on word embedding and tokenizer than on a real "learning" of the local semantics (the accuracy of PET on Allociné is relatively high for both models used, and we believe that the extremely low accuracy found on the HuSST task is due to an error in the labelling process in our code that we unfortunately did bot manage to find). The other focus of our project was to create new tasks and use new MLM for comparison between the accuracies of different (limited) few-shot learning paradigm. For instance, the same task COPA in English and its Hungarian version huCOPA for the same language model, XLM-roBERTa, and for the same parameters, the huCOPA yields a better accuracy. However, one should still keep in mind that these outputs are the results of truncated learning since we limited the epochs due to hardware limitations and time restrictions on google colaboratory. Indeed, the accuracy might reach a plateau at some point for the low ressource language. On the same note, RTE on XLM-RoBERTta performed similarly in the English and the Hungarian version. However, once again these accuracy are lower than the one in [10] due to epochs truncation. Therefore, it remains an open question if the power law will apply differently to the different types of language.

5 Statement of Contribution

* The two authors contributed equally on this work. More precisely,

- Alexis Emanuelli: tried to make functionning the existing PET framework within our computer limitations and defined the french PVP for Allociné
- MathiasVigouroux: defined the PVP for the Hungarian datasets and also incorporated new pre-trained language models.

Code

The code can be found on the Github : Project MVA NLP 23

References

[1] L. Bentivogli, I. Dagan, H. T. Dang, D. Giampiccolo, and B. Magnini. The Fifth PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the TAC Workshop*, 2009.

[2] T. Blard. French sentiment analysis with bert. `https://github.com/TheophileBlard/french-sentiment-analysis-with-bert`, 2020. GitHub repository.

[3] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019.

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

[5] N. Ligeti-Nagy, G. Ferenczi, E. Héja, K. Jelencsik-Mátyus, L. J. Laki, N. Vadász, Z. G. Yang, and T. Váradi. Hulu: magyar nyelvű benchmark adatbázis kiépítése a neurális nyelvmodellek kiértékelése céljából. In *XVIII. Magyar Számítógépes Nyelvészeti Konferencia*, pages 431–446, Szeged, 2022. Szegedi Tudományegyetem, Informatikai Intézet.

[6] L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, É. V. de la Clergerie, D. Seddah, and B. Sagot. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[7] D. M. Nemeskey. Introducing `huBERT`. In *XVII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2021)*, page TBA, Szeged, 2021.

[8] M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011.

[9] T. Schick and H. Schütze. Exploiting cloze questions for few-shot text classification and natural language inference. *Computing Research Repository*, arXiv:2001.07676, 2020.

[10] T. Schick and H. Schütze. It's not just size that matters: Small language models are also few-shot learners. *Computing Research Repository*, arXiv:2009.07118, 2020.

[11] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics.

[12] R. Violin. Next word prediction. `https://github.com/renatoviolin/next_word_prediction`, 2020. GitHub repository.

[13] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding, 2019.

[14] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics.