It's Not Just Size That Matters: SLMs Are Also Few-Shot Learners

Project report

Lucas Fourest, Adéchola Kouande, Marius Roger

January 26, 2024

Contents

Our code is publicly available at this link

Introduction

The 175 billion parameters GPT-3 model [1] has amazed the public with its ability to generate text auto-regressively by predicting masked tokens. Another capability of GPT-3 that excited the AI community was its few-shot performance.
Few-shot learning describes a setting where we have a very small set of labeled examples. GPT-3 achieves near state-of-the-art results for some SuperGLUE [7] tasks given just 32 labeled examples through ***priming*** (GPT-3 is given a few demonstrations of inputs and corresponding outputs as context for its predictions, but no gradient updates are performed).
While being straightforward to use, this method has **two major drawbacks**: first, it requires a gigantic Language Model (LM) to work well, making it unusable in many real-world scenarios, and, second, it does not scale to more than a few examples as the context window of most LMs is limited to a few hundred tokens, while GPT-3 can process up to 2040 tokens.
Given the difficulty associated with maintaining and using huge data sets, this learning paradigm using only few labeled examples is very interesting, and would be even more so if standard or even small models were able to perform reasonably well on tasks offering a limited amount of labeled data.

In the paper **"It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners"**, Timo Schick and Hinrich Schütze claim that a 175 billion parameters model is not necessary for few-shot learning in Natural Language Processing (NLP). They adapt their Pattern-Exploiting Training (PET) algorithm [6], which combines generative LMs with discriminative classifiers through the use of **Patterns** and **Verbalizers**. Two alternative techniques are also proposed : iterative PET (iPET) and PET with multiple masks. In this report, we first explain how PET, iPET, and PET with multiple masks work, then we discuss the authors' results, before finally exposing and critiquing our approach.

1 The PET techniques

This summary is that of the paper **"It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners"** written by Timo Schick and Hinrich Schütze. The thesis of PET, and what GPT-3 is hinting at with its in-context learning, is that all NLP tasks can be formulated as language modeling or closed tasks. For example, in Natural Language Inference (NLI), the processing task involves taking inputs x_1 and x_2 and then classifying whether x_2 is the entailment of x_1 (see Figure 1), showing how you can restructure this problem such that you integrate the mask token and turn the NLI problem into Language Modeling Predicted (LMP) the mask out token and mapping the mask prediction to the class label.

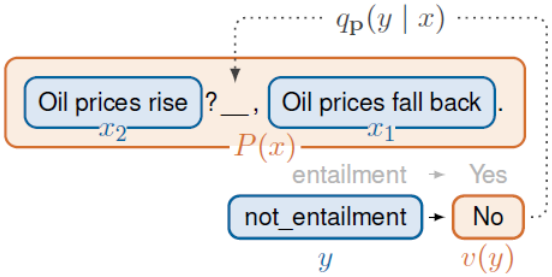

Figure 1: All Tasks are Language Modeling or Closed tasks.

The PET comes with all these clever features that are *Patterns* and *Verbalizers* to perform the mapping of any NLP supervised learning task into Language Modeling. GPT-3 does the same, with the context window containing examples of the task. It then provides a new example and has the GPT-3 model perform the mask prediction to carry out the new task. We begin with a brief recap of the PET algorithm.

1.1 Pattern-Exploiting Training (PET)

The PET idea is a way to leverage Pretrained Language Models (PLM) for downstream supervised learning tasks. The idea is distilling the knowledge learned from these generative language models into discriminative tasks like *Yelp review classification* or NLP inference. Let us examine the example provided in Figure 2.

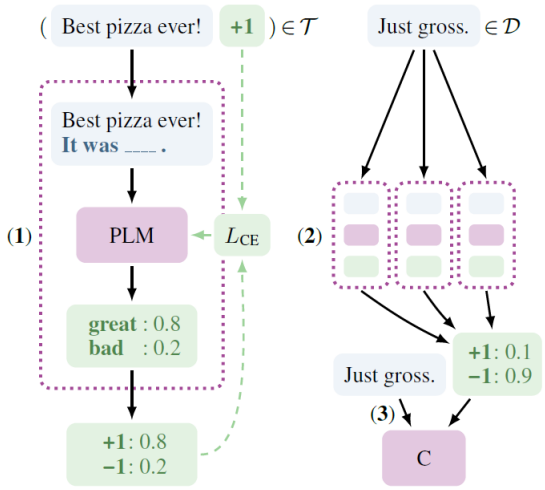

Figure 2: PET for sentiment classification. (1) A number of patterns encoding some form of task description are created to convert training examples to cloze questions; for each pattern, a pretrained language model is finetuned. (2) The ensemble of trained models annotates unlabeled data. (3) A classifier is trained on the resulting soft-labeled dataset.

Let us take the data we have and insert this pattern, so the original data might be something like **Best pizza ever**, and we'll append this context (pattern) It was ____, (It was < mask >). The language model will fill out the mask, and then we will have a verbalizer that maps the language model outputs to the class label. We will use that for supervised fine-tuning of the small labeled dataset that we have, to update our language model and improve its ability to label this data for the supervised learning task. Then use semi-supervised knowledge distillation. The language model will be employed, after it has been fine-tuned on the small labeled dataset as described above, to start labeling a massive set of unlabeled data similarly using the patterns and verbalizers. Another part of the PET algorithm is to construct several of these patterns and use an ensemble of these patterns to help sift out the noise with respect to each of these patterns appended to the given task.

Distilling the knowledge of all individual models into a single classifier C means they cannot learn from each other. As some patterns perform (possibly much) worse than others, the training set $\mathcal{T}_C$ for our final model may therefore contain many mislabeled examples. To compensate for this shortcoming, Schick and Schütze devise iPET [6, Section 3.4], an iterative variant of PET.

1.2 Iterative PET (iPET)

Since some patterns may perform worse than others, the patterns need to be fine-tuned. The strategy employed is the iterative PET algorithm. An example is given in Figure 3.

First of all the original dataset $\mathcal{T}$ (the small training set) is enlarged by labeling selected examples from $\mathcal{D}$ (set of unlabeled examples) using a random subset of trained PET models (Figure 3*a*). Then a new generation of PET models is trained on the enlarged dataset (b); this process is repeated several times (c).

Before we continue, let us briefly summarize how PET works and the transition from PET to iPET. In PET, they begin with supervised learning of the language model, mapping its outputs into class labels for about 32 labeled examples to ensure a fair comparison with GPT-3. Subsequently, they label the unlabeled data and distill it into a smaller model, another model, or the ensemble of models, as in the

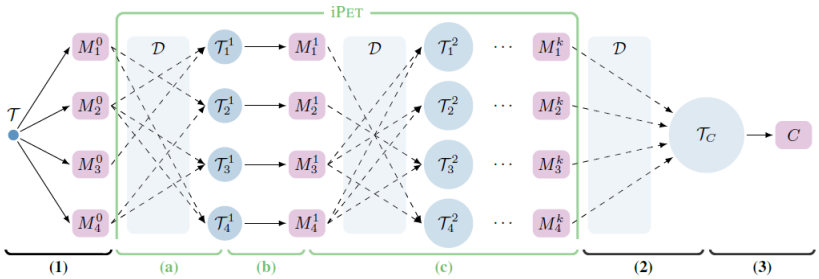

Figure 3: Schematic representation of PET $(1-3)$ and iPET $(a-c)$. (1) The initial training set is used to finetune an ensemble of PLMs. (a) For each model, a random subset of other models generates a new training set by labeling examples from $\mathcal{D}$. (b) A new set of PET models is trained using the larger, model-specific datasets. (c) The previous two steps are repeated k times, each time increasing the size of the generated training sets by a factor of d. (2) The final set of models is used to create a soft-labeled dataset $\mathcal{T}_C$. (3) A classifier C is trained on this dataset.

iterative PET algorithm. This process forms a loop to leverage unlabeled data for representation learning.

One drawback of PET is its limitation to cases where the answers predicted by the language model correspond to a single token in its vocabulary. This poses a significant restriction, as many tasks are not easily formulated in such a manner. The following section explores the adaptation of PET for tasks that involve predicting multiple tokens.

1.3 PET with Multiple Masks

Another interesting direction explored is the PET algorithm with multiple masks. In this case, the verbalizers are generalized to a functions mapping a set of outputs to the set of all token sequences. The PET is also generalized so that the output space do not assume to be identical for each input: for each input x , the output belongs to the of possible outputs denoted by Y_x.

Let us look at this example: we have this entailment task "Oil prices rise $< mask >$, Oil prices fall back". We have the mask that we are using to signify entailment neutral (*i.e.* , no entailment between these two sentences). So we might also integrate patterns like "More $< mask >$ news" (More good/bad news, for example) that could provide more signals to predict this label, and also integrate "I'm $< mask >$ going to buy Oil", using more of this signal to determine the supervised learning label for this natural language inference task. Then, at the end, we get "More $< mask >$ news: Oil prices rise? $< mask >$, Oil prices fall back. I'm $< mask >$ going to buy Oil."

2 Experiments

Here are a brief description of each of the SuperGLUE tasks and their corresponding Patterns Verbalizers pairs. The vertical bar (|) is used to mark boundaries between text segments. Only three tasks (COPA, WSC, and ReCoRD) require the use of PET with multiple masks.

2.1 Tasks

- **BoolQ** dataset is a question-answering task where each example consists of a passage p, a question q, and the expected answer, which can be yes/true and no/false (verbalizers).
- **CB and RTE** datasets consist of a premise p and hypothesis h, and the task is to predict whether the hypothesis is the entailment of the premise. The pattern predicts yes, no, and maybe (the verbalizers), and those outputs from the language model are mapped to entailment, disagreement, and neutral.

- **COPA** dataset. There are two choices c_1, c_2 and the task is to determine which of these choices are the *cause* or *effect* of the premise p, where the verbalizer for c_1 and c_2 is the identity function.
- **WiC** dataset. Given a word w and two sentences s_1 and s_2 in which it occurs, the task is to determine if w is used with the same sense in both sentences. If it is the case the verbalizer is yes, otherwise it is no. They also use b and 2.
- **WSC** dataset is the task of classifying which noun n a pronoun p refers to in the sentence s. p is highlighted in s by putting it in asterisks. The verbalizer here is the identity function.
- **MultiRC** is similar to BoolQ dataset. Given the passage p and the question q the task is to use yes/true or no/false (the verbalizers) to predict whether the answer candidate a is the good answer of q.
- **ReCoRD** dataset is already set up as a closed question. So given a passage p and a closed question q, the task is to decide which of a given set of answer candidates is the correct replacement for the placeholder in the closed question.

2.2 Schick and Schütze Setup

For PET, they choose **ALBERT-xx large-v2** (see [2]) supplemented by a sequence classification head as a final classifier. PET is run on the FewGLUE training set for all SuperGLUE tasks. For COPA, WSC, and ReCoRD, they use PET with multiple masks, and ordinary PET for the other tasks. They train iPET on all tasks except COPA and WSC, as their unlabeled sets contain well below 1000 examples. Additionally, ReCoRD is excluded from iPET training since it only requires a single pattern-verbalizer pair. They simply reuse PET for these three tasks.

Before providing a comparison between GPT-3 and PET/iPET, let us recall that GPT-3 does something called *priming* or *in context learning*, where having a description of the task, examples of the task it will have a prompt to perform a new task (there is no gradient updates).

2.3 GPT-3 vs. PET

This table shows the comparison between GPT-3, PET/iPET, and the state-of-the-art (SotA).

	Model	Params (M)	BoolQ Acc.	CB Acc. / F1	COPA Acc.	RTE Acc.	WiC Acc.	WSC Acc.	MultiRC EM / F1a	ReCoRD Acc. / F1	Avg –
dev	GPT-3 Small	125	43.1	42.9 / 26.1	67.0	52.3	49.8	58.7	6.1 / 45.0	69.8 / 70.7	50.1
	GPT-3 Med	350	60.6	58.9 / 40.4	64.0	48.4	55.0	60.6	11.8 / 55.9	77.2 / 77.9	56.2
	GPT-3 Large	760	62.0	53.6 / 32.6	72.0	46.9	53.0	54.8	16.8 / 64.2	81.3 / 82.1	56.8
	GPT-3 XL	1,300	64.1	69.6 / 48.3	77.0	50.9	53.0	49.0	20.8 / 65.4	83.1 / 84.0	60.0
	GPT-3 2.7B	2,700	70.3	67.9 / 45.7	83.0	56.3	51.6	62.5	24.7 / 69.5	86.6 / 87.5	64.3
	GPT-3 6.7B	6,700	70.0	60.7 / 44.6	83.0	49.5	53.1	67.3	23.8 / 66.4	87.9 / 88.8	63.6
	GPT-3 13B	13,000	70.2	66.1 / 46.0	86.0	60.6	51.1	75.0	25.0 / 69.3	88.9 / 89.8	66.9
	GPT-3	175,000	77.5	82.1 / 57.2	92.0	72.9	**55.3**	75.0	32.5 / 74.8	**89.0 / 90.1**	73.2
	PET	223	79.4	85.1 / 59.4	**95.0**	69.8	52.4	**80.1**	**37.9 / 77.3**	86.0 / 86.5	74.1
	iPET	223	**80.6**	**92.9 / 92.4**	**95.0**	**74.0**	52.2	**80.1**	33.0 / 74.0	86.0 / 86.5	**76.8**
test	GPT-3	175,000	76.4	75.6 / 52.0	**92.0**	69.0	49.4	80.1	30.5 / 75.4	**90.2 / 91.1**	71.8
	PET	223	79.1	87.2 / 60.2	90.8	67.2	**50.7**	**88.4**	**36.4 / 76.6**	85.4 / 85.9	74.0
	iPET	223	**81.2**	**88.8 / 79.9**	90.8	**70.8**	49.3	**88.4**	31.7 / 74.1	85.4 / 85.9	**75.4**
	SotA	11,000	*91.2*	*93.9 / 96.8*	*94.8*	*92.5*	*76.9*	*93.8*	*88.1 / 63.3*	*94.1 / 93.4*	*89.3*

Table 1: Results on SuperGLUE for GPT-3 primed with 32 randomly selected examples and for PET/iPET with ALBERT-xx large-v2 after training on FewGLUE. State-of-the-art results when using the regular, full-size training sets for all tasks [5] are shown in italics.

Looking at the Average accuracy column (the last column) of this table, We observe a phase shift in performance as we scale up the size of GPT-3. The largest model (175 billion parameters) achieves an average accuracy of 73% compared to 50% accuracy with 125 million parameters. With 223 million

parameters, PET on ALBERT performs 18 points better compared to GPT-3 Med with similar size (350 parameters) while it performs as well as the largest GPT-3 model.

Remark 2.1 *This is not a fair comparison because PET uses supervised learning with gradient updates, and it also employs semi-supervised knowledge distillation with an unlabeled dataset. In contrast, GPT-3 only sees 32 examples in the context window and then doesn't use the rest of the data. However, if you are interested in conducting this kind of few-shot learning and you have a dataset like this, PET is much more realistic for practical use than GPT-3 would be.*

Before concluding the summary of this paper, let us highlight some interesting analyses that Schick and Schütze made regarding the influence of several factors on the few-shot performance.

3 Analysis

The choice of patterns and verbalizers, the usage of both unlabeled and labeled data, and the proprieties of the underlying LM can impact the performance.
Let us begin with the **Patterns**. The reformulations of the tasks as closed questions can be arbitrarily complex. For example, the pattern used by GPT-3 for **WSC** contains an introductory section of almost 30 word and it is unclear if and how this formulation has been optimized. To investigate that, they train ALBERT using PET with three sets of Pattern Verbalizer Pair (PVP) (The initial one defined in Section 2.1 ($\mathbf{p}_{\text{ours}}$), the single PVP used by GPT-3 ($\mathbf{p}_{\text{GPT}-3}$), and the combination of both ($\mathbf{p}_{\text{comb}}$)), and consequently, $\mathbf{p}_{\text{GPT}-3}$ outperforms $\mathbf{p}_{\text{ours}}$ on the task **RTE** whereas $\mathbf{p}_{\text{ours}}$ performs much better on **MultiRC**. This is one example of the importance of finding a good way to express tasks as closed questions.
In the real-world settings, **Unlabeled Data** distributed similarly to data in our use case could be difficult to obtain. To investigate the importance of unlabeled data for regular PET, they compare the performance of the final classifier in PET to that of directly using the ensemble of models corresponding to individual PVPs. Without distillation, averaged across the three tasks **CD**, **RTE**, and **MultiRC**, the ensemble performs better than the classifier. Then if the goal is only to achieve good performance, unlabeled data is not necessary.
To investigate the effect of how **Labeled Data** is used, they compare PET with regular supervised training (*i.e.* without using patterns), and with a fully unsupervised model(*i.e.* , the ensemble using all PVPs but no labeled training examples). As a consequence, given 32 examples, PET clearly outperforms both baselines.
About the impact of the **Model Type**, they compare LM on PET by comparing **ALBERT** with **RoBERTa large** [3] and **GPT-2 medium** [4]. With no distillation, it comes out that using ALBERT as an underlying LM is crucial for PET's strong performance. Exchanging for example ALBERT with **RoBERTa** results in an average performance drop of 8 points. However RoBERTa still outperforms **GPT-3 13B**, and PET with GPT-2 performs much worse than with the two other models.
Regarding the impact of using **PET With Multi Masks**, they look at the three tasks **COPA**, **WSC**, and **ReCoRD**, comparing PET with untrained ALBERT to measure the effectiveness of the proposed training loss. PET outperforms untrained ALBERT for the three tasks.

4 Our Contributions

In this section, we detail our set up and the experiments we chose to conduct. We made the personal choice to work on the IMDb movie reviews dataset for binary sentiment classification. This support is suited to the topic at hand because every movie review sentiment pair can be reformulated as a close question, answerable by yes or no, as illustrated below

(*What a great movie*, 1) → (The review is: *What a great movie* Is it positive?, Yes)

From that statement, we can generate several PVPs and apply them to IMDb data in view to perform PET. Our first idea was to assess models trained with PET and compare them to models trained on

the exact same data with the classical training procedure. The goal is to try to prove the efficiency and usefulness of close question style reformulation, especially in the context of few shot learning with gradient updates (i.e fine tuning a model with very few labeled examples).

To this end we conducted a first experiment to compare PET and classical training by performing gradient updates on the same 32 kept appart examples from IMDb, with each method, using DistilBERT and BERT as backbone architectures for our models, and we summarized the obtained results in Table 2. We considered all the possible PVPs using the following patterns and verbalizer:

P_1: $r \rightarrow$ The review is: r Is it a positive review?
P_2: $r \rightarrow$ r Did this user like the movie ?
P_3: $r \rightarrow$ Read the following review: r Did this user enjoy its experience?
P_4: $r \rightarrow$ The review is: " r ". Is it a positive review?
P_5: $r \rightarrow$ " r ". Did this user like the movie ?
P_6: $r \rightarrow$ Read the following review: " r ". Did this user enjoy its experience?

$$v = \begin{cases} 1 \rightarrow \text{yes} \\ 0 \rightarrow \text{no} \end{cases}$$

The patterns $4, 5, 6$ are reproducing the three first ones, adding more punctuation as well as quotes to isolate the review. Our underlying assumption is that it might provide a slightly better context and ease the model understanding. We will later discuss this and interpret the impact of each different PVPs on the overall performance 5.

Model	*classic*	*pvp=11*	*pvp=21*	*pvp=31*	*pvp=41*	*pvp=51*	*pvp=61*
DistilBERT	51.55	56.61	**56.82**	56.00	55.37	53.62	55.06
BERT	51.62	52.69	53.72	**54.65**	53.00	54.13	53.62

Table 2: Comparisons of classical and PET training, best PVP per model in bold

In a second time, to take advantage of the performance and diversity across all PVPs, we went a step further by ensembling the finetuned MLMs as explained in the original paper, to obtain a "global" prediction supposedly more robust and stable. This is done as a weighted average of the MLMs logit scores accross all PVPs 1.

$$q_p(y|x) \propto \sum_{p \in P} w_p s_p(y|x) \tag{1}$$

where P stands for the ensemble of PVPs, s_p the score logits of the MLM related to p and w_p the importance weights determined by the initial accuracy performance on training examples. The results are summarized in Table 3 and the impact of ensembling will also later be discussed.

Model	*classic*	*PET ensemble*
DistilBERT	51.55	**56.52**
BERT	51.62	**54.10**

Table 3: Comparisons of classical training and PET with ensembling across all PVPs

5 Analysis of our experiments

5.1 Interpretation

In the current section, we will try to analyze and interpret the results of our experiments 2 3. First and foremost, we observe that, regardless of the chosen model or method, the accuracy always revolves around 50-60%: would it be with a normal classifier, an MLM trained with PET or an ensemble of MLMs, we perform slightly better than a random classifier in expectation. Therefore, the present results do not

really underline the relevancy and efficiency of the PET method for fine-tuning with very few examples, at least on the IMDb dataset. We believe it might be due to the fact that sentiment classification on this "real world" dataset is a complex task sometimes involving very long and ambiguous reviews as well as unusual words, and these intricacies (compared to standard benchmark data as GLUE used in the original paper) may necessitate consequently more examples to reach "reasonable" performances.

Despite the previous facts, we still observe some improvements for MLMs trained with PET compared to their classically trained counterpart. It is noteworthy that the accuracy can vary substantially across patterns, and that the patterns leveraging the best performances are not always the same for different models. Interestingly enough, the best performance is always obtained among the first 3 patterns, which contradicts our first asssumption made in Section 4: adding quotes and punctuation may just complicate the "understanding" task for the model, and lighter formulations might be preferable. Overall, the choice of an optimal pattern cannot simply be anticipated and this remains very empirical, highlighting the necessity of trying out several different patterns, as explained in the original paper.

We also note that DistilBERT seems to outperform BERT on almost every PVP, while the latter one is a larger LLM. A possible explanation to this counterintuitive observation is that BERT, because of its supposedly greater complexity, quickly overfit the few 32 training examples, as its language modelling is not powerful enough to capture the actual signification and real intricacies of these examples (leading to overfitting). The fact that (as we observed) the training accuracy quickly grows to very high values during PET supports this explanation.

The second part of our experiments was intended to illustrate the benefits of ensembling over all patterns to obtain a stable and robust global prediction, originating from the fact that performances can substantially vary across patterns (and that ensembling is known as a powerful medicine against high variance). The overall results, available in Table 3, support more or less this claim, in the sense that the global prediction accuracy is, in both cases, close to the best one reached across all patterns. We recall that in a "real world" scenario test metrics are not available, and therefore we cannot know in advance the best pattern. Our results illustrate the relevancy of using all the patterns at once to approach the "best pattern we can do", without knowing (among the ones on our list) which pattern it corresponds to. We still observe in each case that a few patterns ($\sim$2) outperform the global performance, which calls into question the reliability of importance weights, and of the way they are computed: does it make sense to take them as the initial accuracy performance of the related pattern and MLM before training ? It is indubitable that there exists more accurate weights and that an optimal configuration should do better than the best patterns themselves, as patterns are supposed to compensate each other weaknesses in the global prediction.

5.2 Limitations

The main limitation we met, that has been challenging for our experiments, and compromised the reliability of our results, is material. We believe that size is an important and limiting factor for the ability of a model to fine tune on very few examples, and the language modelling of rather "small scale" LLMs such as BERT or DistilBERT might not be able to capture the intricacies of the raw and complex reviews from IMDb, thus requiring much more finetuning. It contradicts the title and overall conclusion of the original paper ("not just size that matters") but most of the models presented in the original paper results table seem larger (according to the associated number of parameter displayed) and couldn't be tried in our own experiments due to hardware limitations.

We also faced the fact that IMDb reviews sometimes exceeded 512 tokens (max window length), even though most of them seem not to reach this limit (see Figure 4), it might have been a degrading factor in our experiments, which overall raise the issue/limitation of "real world" input lengths.

Finally, the search of somewhat "good" hyperparameter combinations to reproduce PET on this new dataset was hard and time-consuming, mainly due to the fact that the outcomes of the experiments were highly sensitive regarding some specific hyperparameters such as the learning rate. It refrained us from exploring deeper specific points, or trying alternative paths, that are expressed in Section 6.

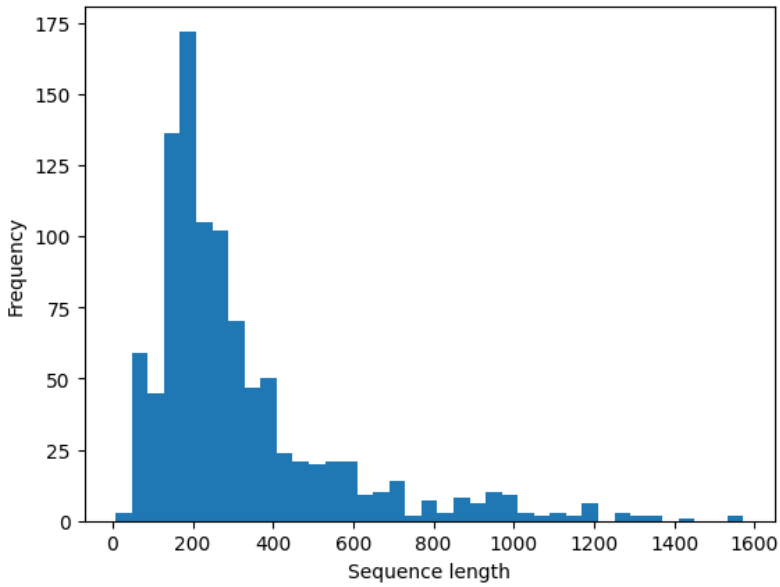

Figure 4: IMDb tokenized reviews lengths distribution

6 Future ideas

To begin with, from a material point of view, we can consider working in the future on more suited environments, such as cloud computing servers, that might reduce our dependency on hardware limitations. We would also add more metrics or more complete ones (such as F1-score) to monitor our experiments. We can also think about extending our work to more different datasets to enable a better understanding of the PET methods regarding the data specificities.

Concerning the experiments themselves, we could have gone even further with the *distillation* experiment, which consists of iteratively labeling a large unlabeled dataset with iPET and the soft-labeling process. A final experience could have been to assess the quality of the obtained soft labels, for instance by comparing the performances of a model trained with the ground-truth dataset (from which we initially kept only 32 examples) against one trained with the final soft labeled dataset. This last point would allow us to understand to what extent iPET can exploit unlabeled data and how close it can approach "reality".

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[3] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[5] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[6] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*, 2020.

[7] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.