‘Algorithm for Speech and NLP‘: Project Report

It’s Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners

Yannis Cattan
MVA, ENS Paris-Saclay
cattan.yannis@gmail.com

Raphael Chesneaux
MVA, ENS Paris-Saclay
raphael.chesneaux@gmail.com

Raphael Grebert
MVA, ENS Paris-Saclay
raphaelgrebert@gmail.com

Nastassia Tardy
MVA, ENS Paris-Saclay
nastassia.tardy@gmail.com

INTRODUCTION

Our work studies the scientific paper [1] which improves[1] *Pattern-Exploiting Training* (PET) algorithm - a training algorithm that leverages the knowledge of an ensemble of language models (LMs) finetuned on few labeled data by reformulating tasks examples as cloze questions [2]. In this report, we introduce the context and explain how PET algorithm works. Then, we describe the experiments we conducted and discuss the results we obtained.

1 CONTEXT

The studied paper follows the line of research focusing on finding NLP methods reducing the training cost of a language model while preserving or improving its performance.

1.1 Large Language Models

Over the past few years, using large language models (LLMs) pretrained on massive amount of textual data coming from general corpora has become the standard procedure when working on NLP tasks. Indeed, when finetuned on specific tasks, those pretrained LLMs achieve state-of-the-art performance on many tasks [2] [3] [4]. Yet, since these LLMs are getting bigger and bigger (e.g. going from 340 million of parameters for BERT [2] to 175 billion for GPT-3 [5]), they need to be trained on more and more data [6] and cost more and more in terms of training computation.

Given that data labeling is time-consuming, semi-supervised learning methods have been widely developed in the recent years. These methods rely on the usage of small amount of labeled data and large amount of unlabeled data. Yet, using unlabeled data does not solve the problem of the computation cost. In [1], PET algorithm tackles both problems by using few labeled examples and models with 800 times fewer parameters than GPT-3 which still achieve competitive performance.

1.2 PET algorithm

PET algorithm combines two main ideas: *(i)* reformulating specific tasks examples as cloze questions and *(ii)* exploiting unlabeled data to perform a kind of knowledge distillation. Its general overview for a given downstream task is illustrated in Figure 2 in the Appendix.

Preprocessing. The first step consists in reformulating the few labeled examples as cloze questions using PVPs. At a high level, this means transforming the specific task (e.g. sentiment analysis) into a mask prediction task by *(i)* transforming the label into one or mutliple tokens and *(ii)* transforming the sample itself so that it contains the masked token that the model has to predict. We refer the reader to the studied paper [1] for a thorough definition and examples of PVPs.

Finetuning the ensemble. Since finding a relevant pattern for a PVP is not an easy task, they use an ensemble of pretrained LM which will be finetuned on slightly different datasets. Indeed, for each LM, the same 32 FewGLUE [3] examples are transformed using a unique PVP. This results in n different finetuned LM.

Distillation. One can either use the ensemble as a predictor. The alternative is to use the ensemble’s knowledge to label a large amount of unlabeled data coming from SuperGLUE [7] dataset.

Finetuning final classifier. Using this freshly labeled datset, one can finetune one single LM and use it as a predictor. The advantage is that this final predictor is lighter, hence faster and easier to store than the ensemble. In average, knowledge distillation comes with a slight loss in performance. Yet, for some tasks, we observe (see Table 1) that the final classifier gives better results than the ensemble.

2 EXPERIMENTS

After having read the studied paper, we devised several experiments to test PET algorithm. We detail our experiments in the following subsections and present their results in sectoin 3. We decided not to work with iPET since we have limited computing resources and that it requires more computation than PET. Also, for experiments 2.1, 2.2.1 and 2.2.2, we used the dev set to evaluate PET because dev sets contain in average less samples than the test sets, resulting in a shorter evaluation time.

2.1 PET with a smaller model

In their paper, the authors only present results coming from PET with models having more than 200M parameters (e.g. ALBERT-xxlarge-v2 and RoBERTa-large composed of 223M and 355M parameters respectively). With this experiment, our original idea was to analyse how PET would behave with small LMs.

To conduct this experiment, we used the FewGLUE dataset provided by the authors, making it relevant to compare our results with theirs. In the rest of this report, we refer to their FewGLUE training dataset as Σ_0. For each evaluated task, we used all the patterns made available by the authors in their code. Also, we used their default hyperparameters since conducting a thorough hyperparameters tuning would have been ridiculously long given our limited resources. Finally, we used ALBERT-base-v2 model [8] available on Huggingface. Results and discussions can be found in section 3.1.

[1] The same authors introduced PET algorithm in a previous paper.
[2] A cloze question is a portion of language with certain language items removed (like words). The aim of a cloze test is to find the missing language items.
[3] FewGLUE is a publicly available subset of SuperGLUE dataset [7] created by the authors of [1] to enable reproduction of their experiments.

2.2 Robustness to variability of training set

2.2.1 Different sets of equal size. In Table 6, the authors show that changing the set of training examples can result in large performance differences for PET. Yet, they focus only on the performance of the ensemble. We asked ourselves: do these differences fade away once distillation is performed? Given the fact that the unlabeled dataset is way larger than the the training set for CB, MultiRC and RTE tasks, our intuition was that distillation would indeed reduce these differences.

To test it, for each of the 3 tasks cited above we used three different training sets Σ_1, Σ_2 and Σ_3. Each one of them was created by randomly sampling 32 datapoints from the initial SuperGLUE training set of the given task. These datasets are publicly available in the *data/fewglue/* folder inside this repository: https://github.com/ycattan/pet. We trained PET with ALBERT-base-v2 on these training sets with distillation. Results and discussions can be found in section 3.2.1.

2.2.2 Sets with different size. One of the main strengths of PET algorithm is that it trains LMs using small labeled datasets. Therefore, we can ask ourselves if it is possible to push the experiment further by training the algorithm on even smaller datasets with 16, 8 and 4 examples. This type of experiment will allow us to analyze the relevance of having 32 training examples instead of a smaller labeled dataset. As we decrease the size of the labeled dataset, we naturally expect worse performances in average since we lose information. On the other hand, since the authors show that the choice of the labeled dataset has a huge impact on the performance, we could also expect performance to depend on the choice of these smaller training sets. So we could imagine that some 4-examples subsets might give better performing models than some 'bad' 32-examples subsets.

To perform this experiment, we used sub-datasets of fewglue by randomly sampling 16, 8 and 4 examples for the CB and RTE tasks. These datasets are publicly available in the *data/fewglue/set_different_size_exp* folder inside this repository: https://github.com/raphaelchesneaux/pet. We trained PET with ALBERT-base-v2 on these training sets with distillation. Results and discussions can be found in section 3.2.2.

2.3 Domain shift: domain specific tasks

Context. In this project, we study the interest of reformulating specific NLP tasks into cloze questions to achieve top tier performances on text classification with surprisingly few parameters and low training computational requirements. This had us wondering if restraining not only the range of our model abilities, but also the domain covered by the pretraining dataset could help increase performances on domain specific tasks.

Indeed, the global trends of the latest LLMs such as GPT-3 is to update a colossal amount of parameters on general-domain datasets. For instance, GPT-3 is trained on an augmented version of CommonCrawl, a dataset gathering information from English-language Wikipedia and books corpora. In the case of application to a specific domain and/or task, the models are finetuned with smaller datasets and appropriated training processes. But one could wonder if training from scratch over a determined, domain specific dataset could improve performances and limit the training requirement in terms of computing power and amount of data.

Related litterature and goals. This is quite exactly what Gu et al. did in their study [9], tackling this issue in the specific context of Biomedical NLP. To provide comparable results, they benchmark BERT-related models (trained on different datasets ranging from general domain data to very specific medical domain datasets) and compare them to their own PubMedBERT model, which is trained from scratch on a selected dataset which only gathers medical related data. The benchmark is made on medical-related datasets, and covers standard NLP tasks such as NER, text classification, question answering, etc. The stakes of this study are quite similar to the one of PET: designing a new training procedure that is less resources consuming and more adapted to the specific task (in particular, the medical domain in [9]).

This paper made us wonder is PET is capable of adapting to new domains. Thus, we propose a domain-shift study for the PET algorithm. With the default learning and evaluation procedures proposed by PET's authors in their GitHub, we present a small experiment showing how PET can adapt to a domain-specific task.

Domain shift experiment procedure. We use the PubMedQA dataset [10]. It is aimed to provide training and evaluation sets for domain-specific question answering, and we refactored it to fit the exact same format as the CB task from the PET [1] paper. We split it into training, validation and test sets following Gu et al.'s split [9]. BERT-base-cased [2] and PubMedBERT [9] are successively used as the underlying LM for PET, to provide comparison between their relative performances as provided by Gu et al. [9]. This allows us to compare performances of a LM trained on a general domain dataset vs. a medical domain specific dataset.

We then apply the following procedure for **both models**:

- Measure PET performance for the QA task on PubMedQA validation dataset **without any finetuning**, as a sanity check. The code proposed by [1] makes it easy to only evaluate PET algorithm without performing finetuning on the LMs of the ensemble.
- Finetune the BERT/PubMedBERT-based PET on **CB train dataset** and proceed to measure performance for the QA task on PubMedQA validation dataset.
- Finetune the BERT/PubMedBERT-based PET on **PubMedQA train dataset** and measure performance for the QA task on PubMedQA validation dataset.

We will compute the performances using the accuracy metric, and compare them with results from Gu et al. [9].

The general overview of the domain shift experiment is summarized in Figure 1. Results and discussions can be found in Section 3.3.

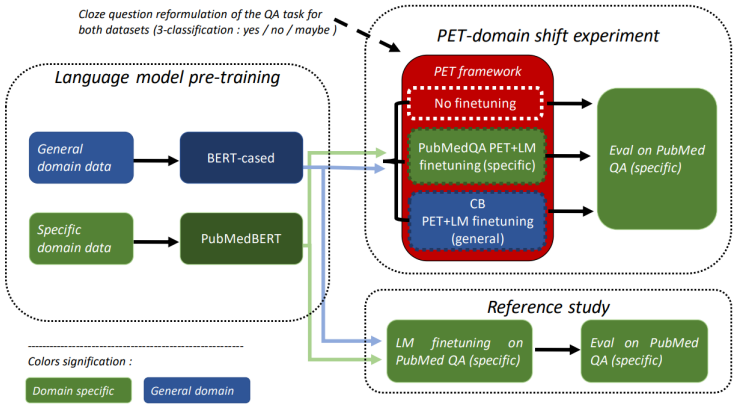

Figure 1: *Domain shift experiment*

2.4 Adapting PET algorithm to other languages

Context. State-of-the-art natural language processing models are generally trained on data in a single language (usually English), and cannot be directly used beyond that language. With this experiment, we adapt PET to another language (French) and observe how it impacts PET's performances.

The task studied. The task chosen for the experiment is XNLI (Natural Language Inference). The goal of this task is to predict textual entailment (does sentence A imply / contradict / neither sentence B). As a consequence, this task is very similar to the CB task used in [1]. The XNLI dataset is available in 15 different languages and was made available by [11].

The pretrained model. We use the camemBERT model developped in [12]. CamemBERT is "a French monolingual language model", trained on the corpora OSCAR. It reaches or improves the state-of-the-art in four downstream tasks: part-of-speech (POS) tagging, dependency parsing, named entity recognition (NER) and NLI. It uses the RoBERTa architecture, an improved variant of BERT architecture. Results and discussions can be found in Section 3.4.

3 RESULTS AND DISCUSSION

This section contains the results and further observations of the experiments we presented in Section 2.

3.1 PET with a smaller model

The results of this experiment are shown in Table 1. We extracted results of GPT-3 Small, GPT-3 Med, GPT-3 and PET combined with ALBERT-xxlarge-v2 from [1] to compare our results.

The LM we used (ALBERT-base-v2) is made of 11 million parameters, and is thus ∼10 times and ∼30 times smaller than GPT-3 Small and GPT-3 Med respectively. Yet, for all tasks except COPA[4], our little PET outperforms or reaches the performances of GPT-3 Small. This shows PET's ability to accurately train LMs, even when these LMs are relatively small compared to most LMs used nowadays. This observation is of paramount importance as many companies or research labs might not have the computing resources to train or finetune a LM as big as GPT-3. PET might come in handy for these type of users. More impressively, we note that our little PET reaches and significantly outperforms GPT-3 Med on BoolQ and CB respectively.

On the other hand, we also observe that by using a relatively small LM, PET cannot extract and learn enough signal from training data for tasks like RTE and WiC. Indeed, our models do not perform better than a random classifier (as RTE and WiC are binary classification tasks). But this applies also to GPT-3 Small and GPT-3 Med, showing this is more of a learning capacity problem of the LM itself than a limitation from PET.

Hence, we showed that PET can be used in a small LM regime to perfom better than standalone LMs of equivalent size. However, if the underlying LM used in PET cannot extract signal from the training set, PET might not be of a great help.

3.2 Robustness to variability of training set

3.2.1 Different sets of equal size. The results of this experiment are shown in Table 2.

[4]the huge drop in performance after distillation for COPA task can be explained by the relatively small amount of unlabeled data (400) used to finetune the final classifier.

	CB	RTE	MultiRC
Model	Acc./F1	Acc.	EM/F1a
PET ⧹ dist. (Σ_1)	67.4 / 47.8	54.8	14.2 / 62.0
PET ⧹ dist. (Σ_2)	66.8 / 48.1	54.2	15.7 / 62.3
PET ⧹ dist. (Σ_3)	66.0 / 46.0	50.9	16.0 / 61.9
PET (Σ_1)	67.8 / 47.3	54.5	15.1 / 63.6
PET (Σ_2)	67.8 / 47.3	58.4	17.3 / 64.1
PET (Σ_3)	67.8 / 47.3	49.0	18.0 / 64.3

Table 2: Evaluating PET on different training sets (Σ_1, Σ_2, Σ_3) of size 32 with ALBERT-base-v2. "⧹ dist." refers to the performance of the ensemble i.e. before distillation is performed.

Influence of training set. Our intuition was that distillation would reduce the gap between performances of models trained on different sets. Yet, we surprisingly observe that it depends on the task. **For CB, we obtained the behavior we expected i.e. very close performances after distillation**. To be more precise than in Table 2, we obtained the same results up to the 8th decimal for CB. This extreme result can be explained by the really small size of the dev set (57) compared to the unlabeled set (20 000) used for CB. **However, our results for RTE and MultiRC show that the choice of the few labeled samples used for training do play a role in the performance of the final classifier**. As an example, we achieved 58.4 on RTE with Σ_2 and only 49.0 with Σ_3. Therefore, one should be careful with the labeled set they use to train their LM with PET algorithm.

What with Σ_0? This experiment caught our attention on another aspect of [1]. **All 9 pre-distillation results obtained in Table 2 consistently improve upon the results obtained for the same tasks in Table 1 using Σ_0 (for the Acc. and EM metrics)**. This is especially striking for MultiRC: we achieve with Σ_1, Σ_2 and Σ_3 results between twice to three times better than with Σ_0 in terms of exact matches (EM). We do not have sufficient computing resources to train PET combined with ALBERT-xxlarge on these labeled sets, but this would be interesting to see if this behavior scales up to larger LMs. We made our Σ_1, Σ_2 and Σ_3 of each task publicly available for this purpose.

Overall, this observation emphasizes the results obtained by the authors of [1] in their Table 6: labeled sets have a big influence on the ensemble's performance.

3.2.2 Sets with different size. The results of this experiment are shown in Table 3.

	CB	RTE
Model	Acc./F1	Acc.
PET ⧹ dist. (4)	46.4 / 39.3	47.3
PET ⧹ dist. (8)	43.3 / 30.0	49.7
PET ⧹ dist. (16)	63.3 / 47.9	48.6
PET ⧹ dist. (32)	66.0 / 46.0	50.9
PET (4)	53.6 / 47.6	47.3
PET (8)	41.1 / 19.4	49.5
PET (16)	67.9 / 47.4	48.0
PET (32)	67.8 / 47.3	49.0

Table 3: Evaluating PET on subsets of Σ_0 of different size (4, 8, 16, 32) with ALBERT-base-v2. "⧹ dist." refers to the performance of the ensemble i.e. before distillation is performed.

Model	Params (M)	BoolQ Acc.	CB Acc. / F1	COPA Acc.	MultiRC EM / F1a	RTE Acc.	WiC Acc.
GPT-3 Small	125	43.1	42.9 / 26.1	67.0	6.1 / 45.0	52.3	49.8
GPT-3 Med	350	60.6	58.9 / 40.4	64.0	11.8 / 55.9	48.4	55.0
GPT-3	175,000	77.5	82.1 / 57.2	92.0	32.5 / 74.8	72.9	55.3
PET (ALBERT-xxlarge-v2)	223	79.4	85.1 / 59.4	95.0	37.9 / 77.3	69.8	52.4
PET (ALBERT-base-v2) ⧹ dist.	11	59.4	65.7 / 47.1	55.5	6.0 / 43.5	50.0	50.2
PET (ALBERT-base-v2)	11	62.1	67.8 / 47.3	45.0	6.0 / 47.8	47.6	48.1

Table 1: Evaluating PET performances on dev set using a relatively small LM (ALBERT-base-v2) and Σ_0 as the labeled training set. The first four lines are taken from the studied paper [1] as a comparison. "⧹ dist." refers to the performance of the ensemble i.e. before distillation is performed.

For this experiment, we had the intuition that by decreasing the size of the training set, we would obtain worse results both before and after distillation. We indeed observe a decrease in performance when we decrease the size of the labeled set, but not in the same way for all the tasks. Indeed, in the case of CB, we notice for the training set of size 4 and 8, a significant decrease of the accuracy and the F1 score compared to sets of size 16 and 32. Yet, for the RTE task, this seems much less obvious with stable accuracy values despite the different labeled set sizes.

Besides, our experiment shows a limitation: each evaluation has been done with only one subset of each size. Since the authors of [1] show that the composition of the training set has an important influence on the performance, we think that our results would have been significantly different with different subsets of the same size, with a bigger variance for small subsets.

To further improve our experiment, one could perform a thorough analysis of the impact of the training set size.

3.3 Domain shift experiment: results and discussion

For our experiment, we perform the **3-way QA task (yes/maybe/no)** discussed in Section 2.3. Our results are obtained with a PET algorithm relying on the pretrained BERT-cased model and the pretrained PubMedBERT model from [9]. We used the training set up and method from the GitHub implementation of the original PET paper [1]. We reformulated the PubMedQA dataset to fit the PET framework for question answering and used a 670/75/140 train/val/test dataset split .

The reference performances from Gu et al.[9] correspond to a BERT-cased model pretrained on general domain data and the PubMedBERT model trained from scratch on PubMedQA. Then, both models are finetuned on PubMedQA for the QA task. Please refer to the original study for details about the dataset processing and training set up.

Training set-up	BERT-cased [2]	PubMedBERT [9]
PET: no finetuning	37.3	50.3
PET: general finetuning	**51.1**	**54.7**
PET: specific finetuning	48.8	53.3
Reference results [9]	**49.96**	**55.84**

Table 4: Question answering performances on PubMedQA. The displayed metric is the accuracy on the testing set.

First, it is clear that **PubMedBERT [9] leverages its medical dataset domain specific pretraining and reaches better performances on the specific domain**. As the underlying model for PET, it indeed obtains significantly higher performances (Table 4) on the specific domain test-set than the generally-trained BERT model for all types of finetuning. This goes accordingly to the findings of Gu et al.[9].

Other results are more surprising and were not anticipated. Especially the fact that for both LLMs, **finetuning PET on a general domain dataset yields better evaluation results on the specific domain than finetuning with the specific domain already** as shown in Table 4.

We are not quite sure about the exact reason of this behavior. Looking at the data from both datasets, it is clear that the PubMedQA data is way more complicated than the CB data with longer prompts and really technical questions that are extremely specific. Thus, we have not only a variation in domain specificity but also in task complexity when comparing CB vs PubMedQA. This may explain why the finetuning on the more complex PubMedQA dataset does not work as well. It would be interesting to finetune our PET algorithms with more data on the more complex domain-specific dataset to verify this hypothesis.

Finally, we can state that **using PET for domain specific tasks is a promising procedure**. Indeed, a BERT-cased based PET algorithm yields significantly better results (Table 4) than the original BERT-cased LM alone for the domain specific QA task. This is not true for PubMedQA which still achieves better results than its PET version, even though the PET version performs really competitively when compared to the general results from Gu et al.[9].

But as we suggested in the previous paragraph, this may be due to the fact that the prompts from the PubMedQA dataset are quite complex. We would need to build a larger finetuning dataset for the PubMedQA-based PET algorithm to verify this, and maybe reach or even outperform the accuracy of PubMedQA from Gu et al.[9]'s original results.

Additional experiments could include studying different NLP tasks with other datasets than PubMedQA.. We did not conducted this experiment ourselves mostly because our trials where limited by the available computing resources (Google Colab public environment).

3.4 PET with CamemBERT

For this experiment, we perform language inference (entailment / neutral / contradiction). Our results are obtained with PET algorithm, relying on the pretrained camembert-base model from [12]. The dataset was downloaded on Huggingface, and formatted to keep only 32 initial training examples. We kept the same proportion of each class in the training, unlabeled, val and test sets, but we did not tune the sentences to keep only the 32 training examples yielding the best results with PET.

To make PET algorithm work with another language, we had to translate the NLI PVPs in French. To do so, we kept the PVPs architecture of the CB task designed in [1]. The CB task (for CommitmentBank [13]) is the closest one from the NLI task among the ones tested in the original paper. Having the pretrained CamemBERT and the freshly adapted XNLI dataset, we could use the code from [1] to conduct our experiment. With two datasets which are both disjoint subsets of french XNLI, we obtained an accuracy of 35.8 % and 33.9 % respectively. These results are quite low in comparison to the ones of mBERT and finetuned CamemBERT presented in [12], which are 76.9 % and 82.5 % of accuracy respectively.

We think this performance gap is probably due to an error in our implementation. This intuition is emphasized by the fact that we reach scores slightly better than the ones of random classifiers.

Another interesting experiment would be to adapt PET to the 15 languages of the XNLI dataset, to get a more comprehensive comparison. But this experiment would require a pretrained model for each of the languages tested, which is hard to find in the open source literature.

4 CONCLUSION

In this report, we present PET algorithm from [1] and design experiments to test various of its properties: robustness to the underlying LM used and to the training set in terms of size, domain, and language. The following three main takeaways can be drawn from our observations: *(i)* PET can be used in combination with relatively small LMs and still achieve better performance than single LMs of bigger size, *(ii)* performance of PET after distillation still depends on the few labeled examples used upstream, and *(iii)* PET can be used to achieve high performance on domain specific tasks.

REPRODUCIBILITY

Smaller model experiment. For experiment 2.1, we used the code made publicly available by the authors of [1]: https://github.com/timoschick/pet.

Different training sets. For experiments 2.2.1 and 2.2.2 we made public the datasets we used to train our models. You can find them at https://github.com/ycattan/pet and https://github.com/raphaelchesneaux/pet respectively.

Domain shift experiment. The paper [9], the model PubMedBERT that we have used with PET in our domain shift experiments, as well as the links to its related databases can all be found here: https://microsoft.github.io/BLURB/index.html. This github is provided by the authors of [9], and we only used its ressources to make our own experiments.

Other language experiment. For experiment 2.4, the model CamemBERT [12] is used in combination with the data from [11]. The experiment is made available at https://github.com/nastassiatardy/pet. Especially, you can find both XNLI datasets used to get our results: *XNLI_formatted* and *XNLI_formatted_2*.

ACKNOWLEDGMENTS

We would like to thank the authors of the studied paper for their interesting work.

REFERENCES

[1] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners, 2021.
[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 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, 2019.
[4] 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, 2020.
[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.
[6] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.
[7] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems, 2020.
[8] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations, 2020.
[9] Yu et al. Gu. Domain-specific language model pretraining for biomedical natural language processing., 2021.
[10] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering., 2019.
[11] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.
[12] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online, July 2020. Association for Computational Linguistics.
[13] Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124, Jul. 2019.

A APPENDIX

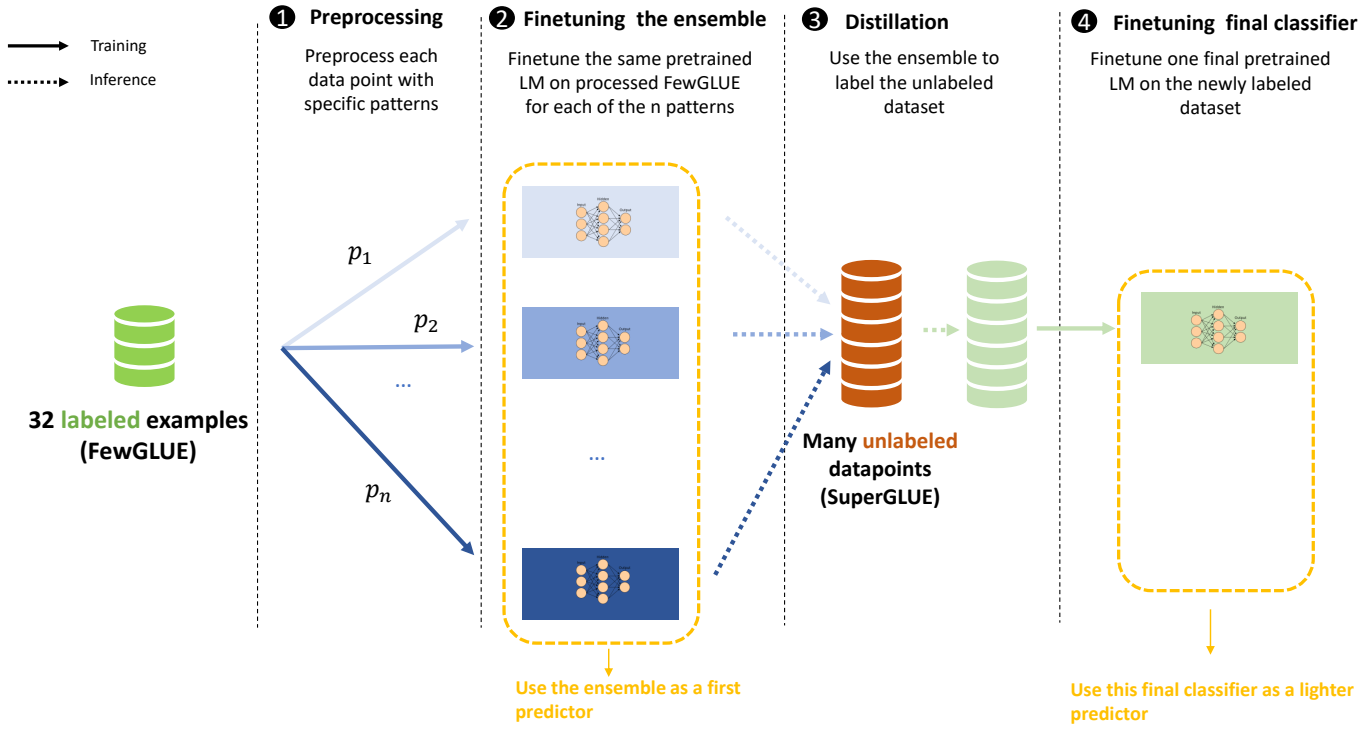

Figure 2: Overview of PET algorithm.