Projected Language Models: A Large Model Pre-Segmented Into Smaller Ones

David Grangier [1] **Angelos Katharopoulos** [1] **Pierre Ablin** [1] **Awni Hannun** [1]

Abstract

Large language models are versatile tools but are not suitable for small inference budgets. Small models have more efficient inference but their lower capacity means that their performance can be good only if one limits their scope to a specialized domain. This paper explores how to get a small language model with good specialized accuracy, even when specialization data is unknown during pretraining. We propose a novel architecture, projected networks (PN). PN is a high capacity network whose parameters can be linearly projected into a small network for fine tuning. We assess the empirical effectiveness of our solution compared to small model training, distillation and hard mixture of experts.

1. Introduction

Large Language Models (LLMs) can address a wide range of language tasks when learning from a large, diverse generic training set (Brown et al., 2020; Bommasani et al., 2022). This rich set ensures that the model fits many subdomains close to the tasks it will eventually address. Model generality is particularly impactful for tasks where the cost of collecting a representative training set cannot be justified. However, LLM inference is costly since fitting a large training set requires many parameters, which results in a high inference cost. This cost restricts LLMs to high-value applications. While efficient inference is an active research area (Aminabadi et al., 2022; Sheng et al., 2023; Dettmers & Zettlemoyer, 2023), reducing model size is the main solution for applications under tight inference constraints.

A smaller language model (SLM) cannot fit a generic training set as well as an LLM. It is hence necessary to forgo the generality of the model to devote its limited capacity to the targeted specialization domain. While a large specialized training set for the application at hand would be ideal for training, such a set is costly and usually justified only for high-value applications. Many applications, therefore, have to face both a limited inference budget and a limited in-domain training set size. For instance, some applications cannot afford to collect more than a few million tokens worth of training data (1m tokens $\simeq$ 10 books). Such applications, with low inference and low data collection budgets, are facing a challenging problem.

This paper studies training small specialized models, even with limited domain data. To achieve low perplexity, we propose projected networks, a high-capacity language model, which can be linearly projected into smaller standalone language models. Fine-tuning a single of these models is efficient and yields better perplexity than small vanilla transformer training or knowledge distillation.

2. Projected Language Models

Our inference constraints require that the final specialized model is a low-capacity SLM. Prior to fine-tuning on the specialization data, the capacity limit does not apply to generic pretraining. We devise a pretraining strategy to take advantage of this asymmetry. At pretraining time, we train a network with many parameters, but each example only interacts with a projection of the parameters onto an SLM. Like distillation, this strategy trains a model with many parameters, but unlike distillation, all model evaluations during training are already constrained to operate within the size limits.

Projected network Our Projected Network (PN) trains jointly a collection of small models or *experts*, $\{\text{SLM-pn}_i\}_{i=1}^k$. Each expert is instantiated via its specific linear projection of the large parameters; see Figure 1. We train a large capacity PN network during generic pretraining. Once the specialized training data are available, specialized fine-tuning starts from one of the experts. Different strategies for expert selection are discussed in our experiments.

The PN model adds capacity to the linear layers of a model. It is configured via 3 hyper-parameters h, k, m. h is a multiplicative factor increasing the overall model capacity while k independently controls the number of experts. Finally, m controls the small capacity specifically allocated to each expert. For each SLM-pn$_i$ with $i = 1 \dots k$, the parameter matrix $W^{(l,i)} \in \mathbb{R}^{d \times d'}$ of a layer l is computed via a linear

[1]Apple Inc. Correspondence to: David Grangier <lastname@apple.com>.

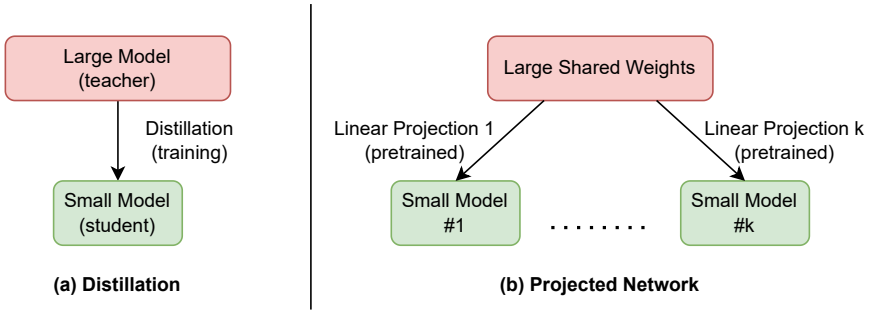


Figure 1. Projected networks (right) unlike distillation (left) instantiate small models in closed-form.

projection,

$$W^{(l,i)} = \sum_{q=1}^{m} E_{i,q} \sum_{r=1}^{h} M^{(l)}_{q,r}\, T^{(l)}_{\ldots,r}$$

where $E_i \in \mathbb{R}^m$ is an expert-specific vector, $M^{(l)} \in \mathbb{R}^{m \times h}$ is a layer-specific matrix and $T^{(l)} \in \mathbb{R}^{d \times d' \times h}$ is a parameter tensor. This decomposition is used to form the matrices $W^{(l,i)}$, and the actual parameters of the network that are learned are the vectors E_i, the matrices $M^{(l)}$ and the tensors $T^{(l)}$. In our experiments, our SLM-pn experts are transformers and we only apply the PN decomposition to the feed-forward layers (i.e. multi-layer perceptron, MLP) which hold most of the model parameters.

The PN can separately set the overall network size via h and the number of distinct experts via k. Each expert is associated with a cluster, which is a subset of the generic pretraining set. The details of this clustering are given in Section 3.1. We associate each training example x with a cluster variable $c(x) = 1 \ldots k$ and its loss on x is computed with SLM-pn$_{c(x)}$. Training optimizes all experts jointly by minimizing the expected loss on the generic set.

For specialization, we select one expert SLM-pn$_i$, form the corresponding matrices $W^{(l,i)}$ which are now the learnable parameters of the instantiated network, and fine-tune it on the specialization dataset. On preliminary experiments, we found that picking the pre-trained expert i corresponding to the most frequent cluster in the specialization dataset is an effective strategy.

Hard Mixture of Experts A hard mixture of expert (Gross et al., 2017) divides the generic set in smaller subsets via clustering (see Section Section 3.1), and pretrains a small model, *SLM-mix*, on each subset. Its pretraining cost and its overall number of parameters are high as both scale linearly with the number of clusters. Inference with a hard mixture typically forwards each example to the expert corresponding to the cluster the example belongs to. The hard mixture can be compared to a special case of a PN network in which $h = k$, $E_i = \delta_i \in \mathbb{R}^k$ and $M^{(l)} = \mathrm{I}$. In that case, all weight matrices rely on independent slices of the parameters.

In the hard mixture, unlike in the PN, one cannot set the number of experts k and the capacity multiplier h independently. The expert parameters are not shared and cannot leverage synergies between similar clusters. On the other hand, the learning is embarrassingly parallel since each expert pretraining is independent of the other experts. Like for PN, specialization can be performed inexpensively by fine-tuning a single expert. Despite these differences, our experiments reveal similar benefits for both methods.

3. Experiments & Results

We target an LM with a good specialized perplexity despite having little specialization data and inference constraints limiting the model size. We call Small Language Model (SLM) a transformer of the size allowed by our inference constraint. We consider different limitations in the amount of in-domain specialization data available for training. Besides data from the specialization domain, leveraging a large generic training set is attractive. Our setting suggests widely used baselines.

Our first 3 baselines train an SLM. We first consider an SLM model trained only on the generic data (SLM-generic). This model can be effective if the generic and specialization distributions are close. We then consider an SLM trained only on the specialization data (SLM-nopt). This model can be effective if the specialization training set is large enough. Finally, we consider a model pre-trained on the generic data and fine-tuned on the specialization data (SLM-pt). The amount of fine-tuning can be adjusted via early stopping. This adjusts a trade-off between the proximity to the generic distribution and overfitting to the specialization set.

Since our capacity constraint is motivated by inference efficiency, we can consider larger models at training time and rely on distillation to satisfy the inference requirements. Specifically, we consider training a large language model (LLM) that has a bigger size than the inference constraint allows via generic pretraining and specialization fine-tuning. A pre-trained SLM student is then fine-tuned to a mixture of the specialization data and next-word distributions from

Table 1. Number of parameters for pretraining and inference.

Model	Num. parameters (m)	
	Pretrain	Inference
SLM	126	126
-mix	2,016	126
-pn	1,422	126
LLM	771	771

Table 2. Model throughput, GPUh per 1B training tokens.

Model	Training		Infer.
	G_{eneric}	$S_{\text{specializ}}$	
SLM	2.2	2.2	0.61
-mix	2.2	2.2	0.61
-pn	3.6	2.2	0.61
LLM	7.7	7.7	2.54

Table 3. Baseline perplexity on the Pile (average) (<650 GPUh of pretraining).

Model	P_{retrain}	Specialized		
		1M	8M	64M
SLM	33.0	18.2	14.8	12.0
-nopt	N/A	227.1	45.6	17.6
LLM	28.1	14.4	12.5	10.0

the teacher. We call this model SLM-d.

3.1. Clustering of the Pre-training Data

SLM-pn and SLM-mix rely on a clustering of the generic pre-training set. Clustering associates each training example with a discrete latent variable that we use to condition the projection in projected networks (section 2). To cluster the data, we embed each document in the generic set as a vector using Sentence BERT (Reimers & Gurevych, 2019). The documents longer than the maximum context we consider (1,024) are broken into nonoverlapping windows. Then, we use `k-means` to cluster the generic set into k clusters.

3.2. Methodology

We study the training methods at various training costs. We report training costs and which part of the cost can be shared across multiple domains. We consider 4 important metrics:

Generic training cost: the cost of the training phase that can be performed before the specialization data are available, on a generic training set. This cost is domain-independent and is often called pretraining. The generic training data are essential when specialization data are limited.

Specialization training cost: the cost of the training performed once the specialization data are available. This cost is not shared across different specializations.

Inference cost: the inference cost of a specialized model. Low inference cost allows wider, cheaper deployment.

Size of the specialization set: it varies across applications and influences pretraining and specialization choices.

Taking the inference cost and the specialization data size as hard constraints, we study the operating curves resulting from varying the generic and specialization training costs. We measure training costs in hours of graphic processor compute time (GPUh) on the same hardware (Nvidia-A100). We consider pretraining costs ranging from 10 to 650 GPUh and specialization costs ranging from 0.3 to 120 GPUh.

This work focuses on language modeling. Performance is measured by perplexity, using 20k held-out documents per dataset. We report perplexity on generic and specialization data. We report *macro-averaged* perplexity across specialization domains: we compute the mean negative log-likelihood per token for each domain, average these results, and compute the exponential. All domains hence get the same weight, regardless of the number of tokens per held-out set.

3.3. Datasets & Hyperparameters

Our generic pretraining set is c4, a large dataset of English text derived from commoncrawl (Raffel et al., 2020), tokenized with a 32k sentence piece model. We specialize to nine domains from Pile (Gao et al., 2021): arxiv (science), europarl (parliamentary proceedings), freelaw (legal text), gutenberg (old books), opensubtitles (theatrical subtitles), openwebtext2 (forum), pubmed-abstracts (medical abstracts), stackexchange (Q&A mostly about technical topics), wikipedia (encyclopedia articles). We vary the amount of specialization training data available and consider sets of size 1, 8 and 64m tokens for each domain.

Table 1 reports the number of parameters for the pretrained and specialized models. Table 1 illustrates that SLM-pn and SLM-mix are as small as SLM for inference after specialization while their overall number of pretrained parameters is larger than LLM. Table 2 reports the throughput of the models. All SLM models have the same specialization throughput while SLM-pn has a lower throughput than SLM, SLM-mix for pretraining. LLM is more expensive in all cases. Note that the parameter count and throughput of the SLM-generic, SLM-nopt, SLM-pt and SLM-d is the same as for SLM. Table 3 presents the upper limit in training budgets for pretraining and specialization over all settings

3.4. Results

We vary pretraining budgets and report perplexity on the generic pretraining set (c4) for each method in Figure 2. When we consider SLM-pn and SLM-mix, we observe that even if the number of pretrained parameters is larger than LLM, they do not enjoy as good perplexity. However, their perplexity is better than SLM while they are as efficient when tested or fine-tuned on a single cluster.

Generic perplexity (c4) is not our main goal and we now examine specialized perplexities. Figure 3 reports the results before fine-tuning. Specialized perplexities are much higher

Table 4. Train cost upper limits (GPUh)

Model	P_{retrain}	Specialization		
		1M	8M	64M
LLM	650	0.12	0.5	3.5
SLM	530	0.02	0.07	0.5
-d	1,850	0.7	2.8	21
-mix	650	0.02	0.07	0.5
-pn	650	0.02	0.07	0.5

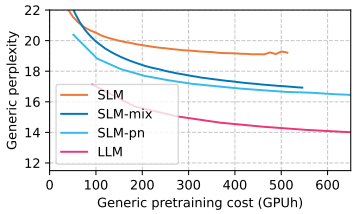


Figure 2. Generic pretrain ppl. on c4.

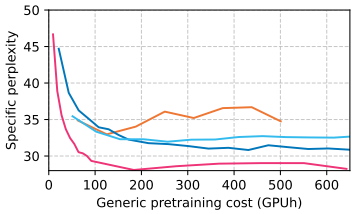
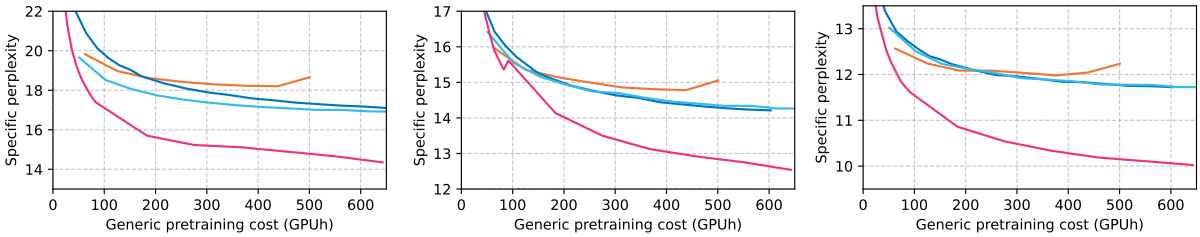


Figure 3. Specialization ppl. before fine-tuning.



(a) 1M tokens (b) 8M tokens (c) 64M tokens

Figure 4. Specialized perplexity on the Pile subsets (average) after fine-tuning with different amounts of specialization data.

than the c4 perplexities, indicating that specialization is necessary. Figure 4 (a) reports the results after fine-tuning several pretrained checkpoints for each method on the 1M token dataset of each domain. Since 1M tokens is a small set, fine-tuning relies on a small learning rate and early stopping (base learning rate divided by 3, stopping when validation loss stops improving, which is always less than 2k fine-tuning steps on one GPU when validation loss stops improving). Fine-tuning is highly beneficial for all methods and results in significantly improved perplexity. We also remark that pre-fine-tuning perplexity on the Pile is not necessarily a good indicator of post-fine-tuning perplexity: e.g. the ordering between SLM-mix and SLM-pn also changes during fine-tuning.

We later consider fine-tuning on 8 and 64 million tokens per domain, see Figure 4 (b) and (c). These experiments fine-tune longer (resp 4k and 30k steps) and keep the base learning rate. We observe that the benefit of a good starting point provided by SLM-pn and SLM-mix (compared to SLM) erodes as the domain training set size increases.

We compare SML-pn and SLM-mix with distillation, SLM-d. The total training cost of distillation includes the teacher and the student training costs. We vary the overall distillation training cost by considering teachers with different pretraining budgets. Figure Figure 5 shows that distillation is not competitive in terms of perplexity vs training cost compared to SML-pn and SLM-mix.

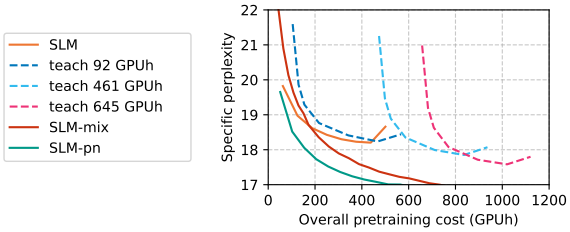


Figure 5. Comparison with distillation, average perplexity after fine-tuning with 1m specialization tokens.

4. Related Work

Domain adaptation for LM has a long history, predating neural network language models (Rosenfeld, 2000). This research stemmed from the observation that models trained on large amount of data, even far from the targeted domain were impactful on end applications (Brants et al., 2007). After neural language models were introduced (Bengio et al., 2000), they were also scaled up to benefit from increasing amount of training data (Raffel et al., 2020; Brown et al., 2020; Chowdhery et al., 2022; Touvron et al., 2023). This growth involves a trade-off between training a model from a large dataset (i.e. reducing estimation errors) or a dataset representative of the end application domain (i.e. having a training distribution representative of test condition), both essential to good generalization (Vapnik, 1995).

Research on efficient LM inference grew as model size increased, including distillation (Hsieh et al., 2023; FitzGerald et al., 2022), weight quantization (Xiao et al., 2023; Dettmers

& Zettlemoyer, 2023) and pruning (Ma et al., 2023; Xia et al., 2023). Mixtures of experts also aim to decouple overall model capacity and inference efficiency (Shazeer et al., 2017; Du et al., 2022; Clark et al., 2022).

5. Conclusions

This work considers a common double practical constraint for language modeling: the scarcity of in-domain training data and a limited inference budget. We propose to train small, efficient language models and improve their accuracy by rethinking the pretraining process on abundant, generic training data. This paper formalizes the problem and introduces Projected Networks, a novel architecture that trains a collection of small models jointly. Each model of the collection can be used on its own, for instance, for fine-tuning on a new domain. The empirical benefit of our contribution is shown across multiple domains, training budgets and training set sizes. Another benefit of the projected networks is that they can be specialized without access to pre-training data. For instance, if the specialization data is sensitive, the data owner can cheaply instantiate and fine-tune the model themselves. Our methodology is not specific to language modeling and we plan to extend it to other modalities where inference constraints are also important (e.g. computer vision).

References

Aminabadi, R. Y., Rajbhandari, S., Zhang, M., Awan, A. A., Li, C., Li, D., Zheng, E., Rasley, J., Smith, S., Ruwase, O., and He, Y. Deepspeed inference: Enabling efficient inference of transformer models at unprecedented scale, 2022.

Bengio, Y., Ducharme, R., and Vincent, P. A neural probabilistic language model. In Leen, T., Dietterich, T., and Tresp, V. (eds.), *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2000. URL https://proceedings.neurips.cc/paper_files/paper/2000/file/728f206c2a01bf572b5940d7d9a8fa4c-Paper.pdf.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D. E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P. W., Krass, M., Krishna, R., Kuditipudi, R., Kumar, A., Ladhak, F., Lee, M., Lee, T., Leskovec, J., Levent, I., Li, X. L., Li, X., Ma, T., Malik, A., Manning, C. D., Mirchandani, S., Mitchell, E., Munyikwa, Z., Nair, S., Narayan, A., Narayanan, D., Newman, B., Nie, A., Niebles, J. C., Nilforoshan, H., Nyarko, J., Ogut, G., Orr, L., Papadimitriou, I., Park, J. S., Piech, C., Portelance, E., Potts, C., Raghunathan, A., Reich, R., Ren, H., Rong, F., Roohani, Y., Ruiz, C., Ryan, J., Ré, C., Sadigh, D., Sagawa, S., Santhanam, K., Shih, A., Srinivasan, K., Tamkin, A., Taori, R., Thomas, A. W., Tramèr, F., Wang, R. E., Wang, W., Wu, B., Wu, J., Wu, Y., Xie, S. M., Yasunaga, M., You, J., Zaharia, M., Zhang, M., Zhang, T., Zhang, X., Zhang, Y., Zheng, L., Zhou, K., and Liang, P. On the opportunities and risks of foundation models, 2022.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. Large language models in machine translation. In Eisner, J. (ed.), *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 858–867, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1090.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. Palm: Scaling language modeling with pathways, 2022.

Clark, A., Casas, D. d. l., Guy, A., Mensch, A., Paganini, M., Hoffmann, J., Damoc, B., Hechtman, B., Cai, T., Borgeaud, S., Driessche, G. v. d., Rutherford, E., Hennigan, T., Johnson, M., Millican, K., Cassirer, A., Jones, C., Buchatskaya, E., Budden, D., Sifre, L., Osindero, S., Vinyals, O., Rae, J., Elsen, E., Kavukcuoglu, K., and Simonyan, K. Unified scaling laws for routed language models. In *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 2022.

Dettmers, T. and Zettlemoyer, L. The case for 4-bit precision: k-bit inference scaling laws. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7750–7774. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/dettmers23a.html.

Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., Zoph, B., Fedus, L., Bosma, M. P., Zhou, Z., Wang, T., Wang, E., Webster, K., Pellat, M., Robinson, K., Meier-Hellstern, K., Duke, T., Dixon, L., Zhang, K., Le, Q., Wu, Y., Chen, Z., and Cui, C. GLaM: Efficient scaling of language models with mixture-of-experts. In

Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5547–5569. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/du22c.html.

FitzGerald, J., Ananthakrishnan, S., Arkoudas, K., Bernardi, D., Bhagia, A., Bovi, C. D., Cao, J., Chada, R., Chauhan, A., Chen, L., Dwarakanath, A., Dwivedi, S., Gojayev, T., Gopalakrishnan, K., Gueudré, T., Hakkani-Tur, D., Hamza, W., Hüser, J. J., Jose, K. M., Khan, H., Liu, B., Lu, J., Manzotti, A., Natarajan, P., Owczarzak, K., Oz, G., Palumbo, E., Peris, C., Prakash, C. S., Rawls, S., Rosenbaum, A., Shenoy, A., Soltan, S., Sridhar, M. H., Tan, L., Triefenbach, F., Wei, P., Yu, H., Zheng, S., Tür, G., and Natarajan, P. Alexa teacher model: Pretraining and distilling multi-billion-parameter encoders for natural language understanding systems. In Zhang, A. and Rangwala, H. (eds.), *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pp. 2893–2902. ACM, 2022. doi: 10.1145/3534678.3539173. URL https://doi.org/10.1145/3534678.3539173.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027, 2021.

Gross, S., Ranzato, M., and Szlam, A. Hard mixtures of experts for large scale weakly supervised vision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5085–5093. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.540.

Hsieh, C., Li, C., Yeh, C., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C., and Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In Rogers, A., Boyd-Graber, J. L., and Okazaki, N. (eds.), *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 8003–8017. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.507. URL https://doi.org/10.18653/v1/2023.findings-acl.507.

Ma, X., Fang, G., and Wang, X. Llm-pruner: On the structural pruning of large language models, 2023.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21 (140):1–67, 2020.

Reimers, N. and Gurevych, I. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Inui, K., Jiang, J., Ng, V., and Wan, X. (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410.

Rosenfeld, R. Two decades of statistical language modeling: where do we go from here? *Proc. IEEE*, 88(8):1270–1278, 2000. doi: 10.1109/5.880083. URL https://doi.org/10.1109/5.880083.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G. E., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Sheng, Y., Zheng, L., Yuan, B., Li, Z., Ryabinin, M., Chen, B., Liang, P., Re, C., Stoica, I., and Zhang, C. FlexGen: High-throughput generative inference of large language models with a single GPU. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31094–31116. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/sheng23a.html.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

Vapnik, V. N. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995. ISBN 0-387-94559-8.

Xia, M., Gao, T., Zeng, Z., and Chen, D. Sheared llama: Accelerating language model pre-training via structured pruning, 2023.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. SmoothQuant: Accurate and efficient post-training quantization for large language models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Appendix

A. Hyper-parameters

All our language model are either instances of SLM or LLM. We rely on the parameters from Table 5. Table 6 extends Table 1 to include the parameter count for the models from all the sections.

B. Interpolated Perplexities

We report the data from the Figures 2 – 4 in Table 7. Since the methods were evaluated at a fixed frequency in steps, we linearly interpolate perplexities and step counts to report results at the same pretraining costs for all methods.

C. Number of Fine-tuning Steps

Figure 6 reports the fine tuning cost each model. This cost corresponds to the number of steps to reach the best validation perplexity. It is an optimistic cost estimates as one usually needs a few more steps to assess that further improvement is not expected. The fine-tuning cost seems to grow $\sim$ 10X when the fine-tuning set size grows 8X. The LLM usually requires less steps than the SLMs but its steps are more expensive. The vanilla SLM overfits earlier than the other SLMs (SLM-mix, SLM-pn) for the small 1M specialization set but not for the larger sets.

D. Clustering & Number of Experts

The clustering of c4 is used by the mixture model to define each expert scope. Similarly it is used as the conditioning variable by the PN. Table 8 report the concentration of each specialization domain from Pile on their most frequent cluster. A high concentration could be positive since it means that, when fine-tuning SLM-pn or SLM-mix conditioned on this cluster, one starts starts from pretrained parameters containing most of the pretraining data relevant to the domain at hand. The table also reports the most frequent cluster on c4 to highlight that the specialization domain distributions differ from the c4 distribution.

Table 5. Transformer parameters

	SLM	LLM
Architecture		
Mum. layers	7	7
Model dimension	1024	2816
Inner MLP dimension	4096	11264
Num. attention heads	8	22
Optimizer		
Optimizer	Adam	Adam
Learning rate	1e-4	1e-4
Clipping norm	5.0	5.0
Linear warmum steps	1,000	1,000

Table 6. Number of parameters (in millions) for pretraining and inference.

Model			Num. parameters (m). Overall	Inference
SLM			126	126
SLM-pn	16	experts	756	126
	32		1,422	126
	64		2,770	126
SLM-mix	4	experts	504	126
	16		2,016	126
	64		8,064	126
	256		32,256	126
LLM			771	771

D.1. Number of Experts for the Hard Mixture of Experts

The overall size of the mixture and its training cost are proportional to the number of clusters. Our main results (Fig. 2, Fig. 4, etc) use 16 experts. We compare results with 4 to 256 experts. Intuitively, if the number of experts is too large, the model would cost more to train and each cluster would not contain enough data to train a model of the size of SLM. Conversely, if the number of experts is too small, the training cost is low but each SLM-sized expert would be trained from a large cluster and would underfit its training set. Also, the large clusters might be too generic and far from the distribution of the targeted set. Figure 7 shows the macro-averaged perplexity on the Pile as a function of the generic pretraining time for the different mixture sizes in the case of the 1M token specialization set.

D.2. Number of Experts for the Projected Network

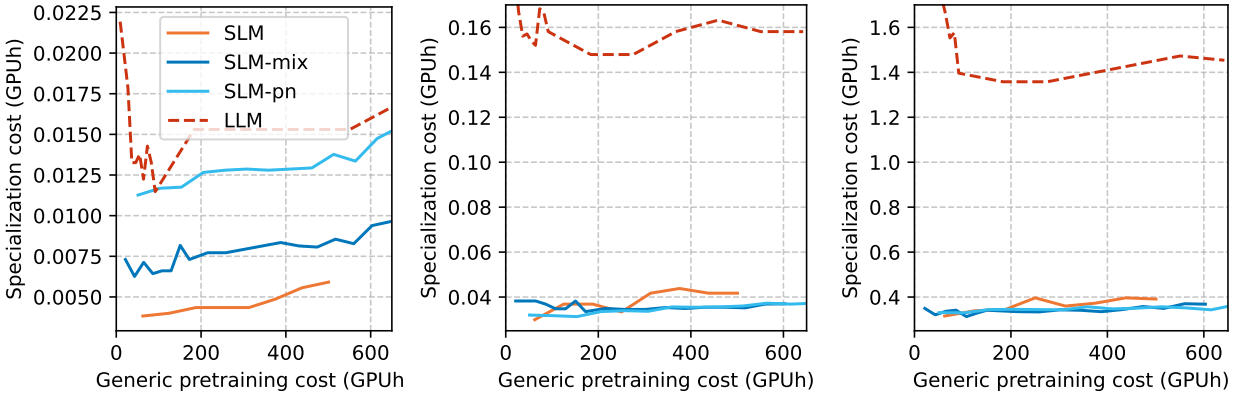
The number of experts in our PN allows tuning the overall number of parameters while keeping the size of the inference model constant. Figure 8 shows perplexity on the Pile subsets after fine-tuning on 1M tokens. More experts always perform better per iteration, however, 32 experts is more compute-time efficient in our setup.

E. Individual Subset Results

Figure 9 decomposes the results in Figure 4 (b) per domain. The subset results are mostly consistent with the average but we observe few differences. SLM-pn and SLM-mix have a close average and the best method among them varies per subset. Also we notice that both methods do not outperform SLM on wikipedia and openwebtext2. The disadvantage of SLM-pn and SLM-mix over SLM can be observed before fine-tuning, as shown on Figure 10. We report the entropy of the cluster histograms in Table 9 and observe that wikipedia and openwebtext2 are the domains with the highest entropy. This means that the c4 data similar to these datasets is more spread across clusters than for the other domains. Conditioning SLM-pn and SLM-mix on a single cluster

Table 7. Interpolated perplexities at fixed pretraining costs (GPUh)

Model	Pretrain cost	Num. steps	Num. GPU	Generic PPL	Spec. PPL No ft	1M	8M	64M
SLM	100	798k	8	20.51	33.74	19.31	15.61	12.37
SLM-mix	100	464k	16	17.13	34.35	19.82	15.82	12.62
SLM-pn	100	195k	8	18.90	33.44	18.57	15.58	12.53
LLM	100	108k	8	17.00	29.22	17.11	15.49	11.55
SLM	200	1597k	8	19.71	34.43	18.58	15.12	12.09
SLM-mix	200	928k	16	15.92	31.94	18.48	14.98	12.15
SLM-pn	200	390k	8	17.74	32.30	17.76	14.95	12.13
LLM	200	217k	8	15.58	28.18	15.62	14.03	10.81
SLM	400	3195k	8	19.17	36.61	18.22	14.80	12.00
SLM-mix	400	1000k	16	15.82	31.04	17.56	14.42	11.84
SLM-pn	400	780k	8	16.90	32.54	17.17	14.48	11.86
LLM	400	434k	8	14.54	28.98	15.03	13.05	10.28
SLM-mix	600	1000k	16	15.82	31.03	17.18	14.21	11.73
SLM-pn	600	1170k	8	16.53	32.53	16.95	14.29	11.74
LLM	600	651k	8	14.09	28.62	14.50	12.64	10.07



(a) 1M tokens (b) 8M tokens (c) 64M tokens

Figure 6. Fine tuning cost as a function of the pretraining cost.

Table 8. Fraction of data in the most frequent cluster, per domain.

Domain	Num. clusters 4	16	64	256	1024
arxiv	0.95	0.92	0.55	0.52	0.29
europarl	0.52	0.53	0.45	0.44	0.27
freelaw	0.48	0.73	0.87	0.72	0.35
gutenberg	0.75	0.54	0.35	0.27	0.29
opensubtitles	0.97	0.68	0.26	0.28	0.32
openwebtext2	0.53	0.35	0.12	0.04	0.02
pubmed abs.	0.94	0.54	0.41	0.20	0.06
stackexchange	0.95	0.94	0.78	0.61	0.31
wikipedia	0.71	0.58	0.21	0.07	0.03
c4	0.32	0.12	0.04	0.02	0.00

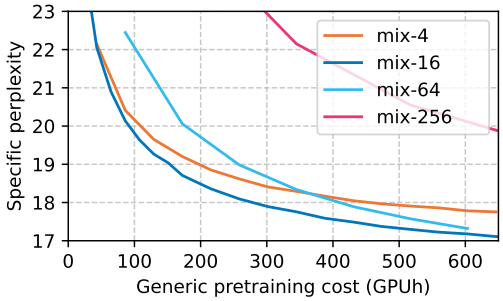


Figure 7. Specialization perplexity of mixture models with 4-256 experts on Pile subsets (average) after fine-tuning on 1M tokens.

variable might not model well these domains. Of course, this correlation between entropy and fine-tuned perplexity of SLM-mix, SLM-pn could be fortuitous. This motivates us to investigate the impact of the different clustering methods and their metrics in future research.

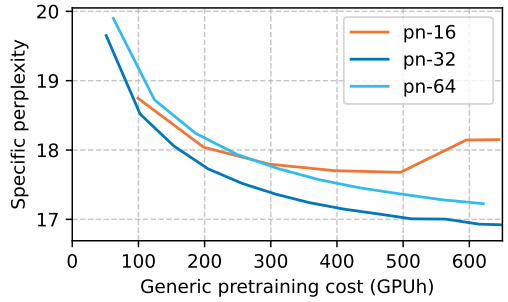


Figure 8. Specialization perplexity for PN with different number of experts after fine-tuning on 1M tokens.

Table 9. Entropy of the cluster histogram for each domain.

Domain	Num. clusters			
	16	64	256	1024
arxiv	0.41	1.02	1.80	2.58
europarl	1.48	1.83	2.31	3.14
freelaw	1.01	0.70	1.44	2.49
gutenberg	1.57	2.42	3.21	3.85
opensubtitles	1.16	2.61	2.95	3.44
openwebtext2	2.19	3.60	4.89	6.12
pubmed abs.	1.07	2.14	3.22	4.43
stackexchange	0.39	0.97	1.78	3.24
wikipedia	1.73	3.20	4.54	5.64
c4	2.73	4.07	5.46	6.85

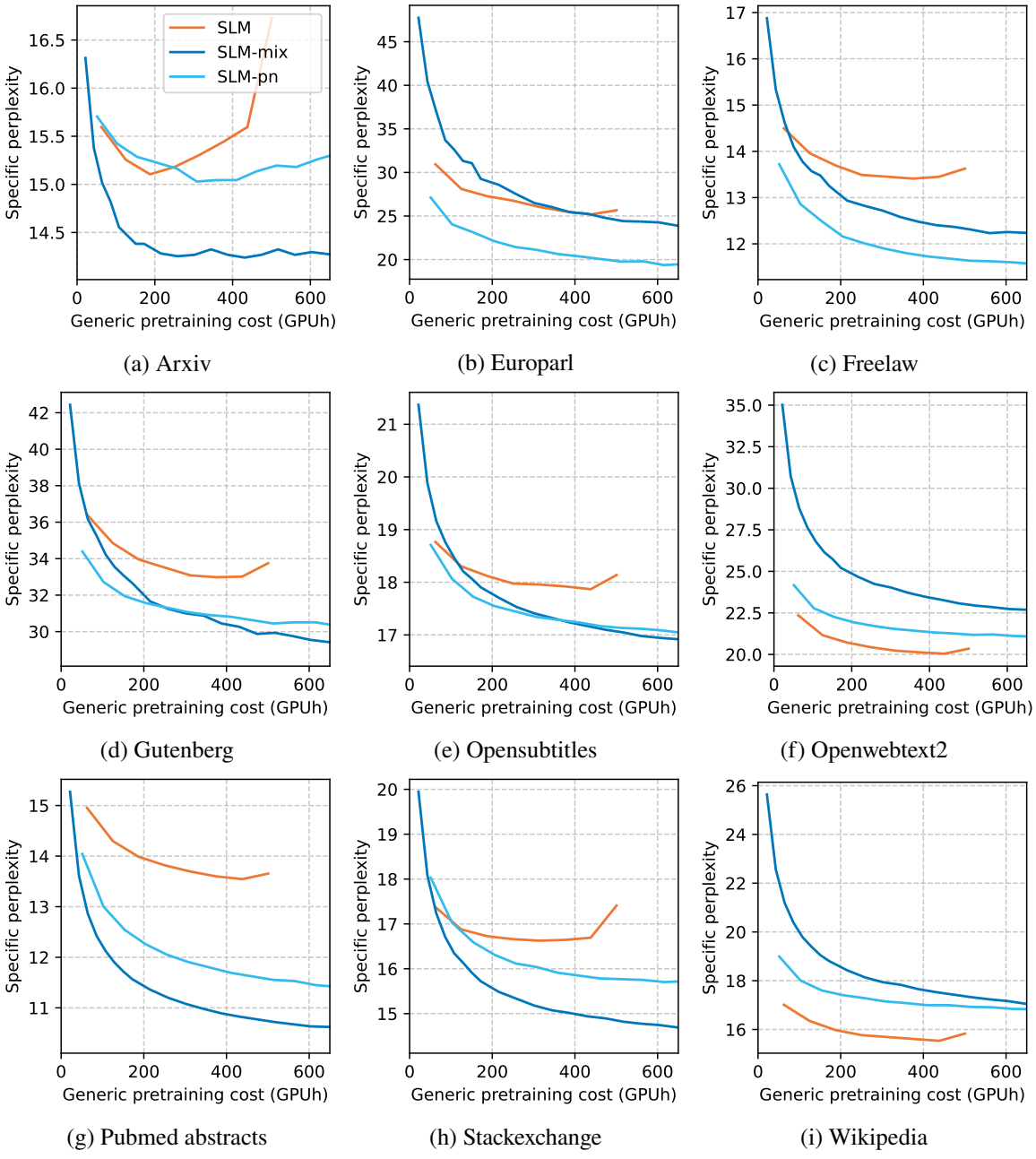


Figure 9. Specialized perplexity on individual subsets after fine-tuning on 1M tokens.

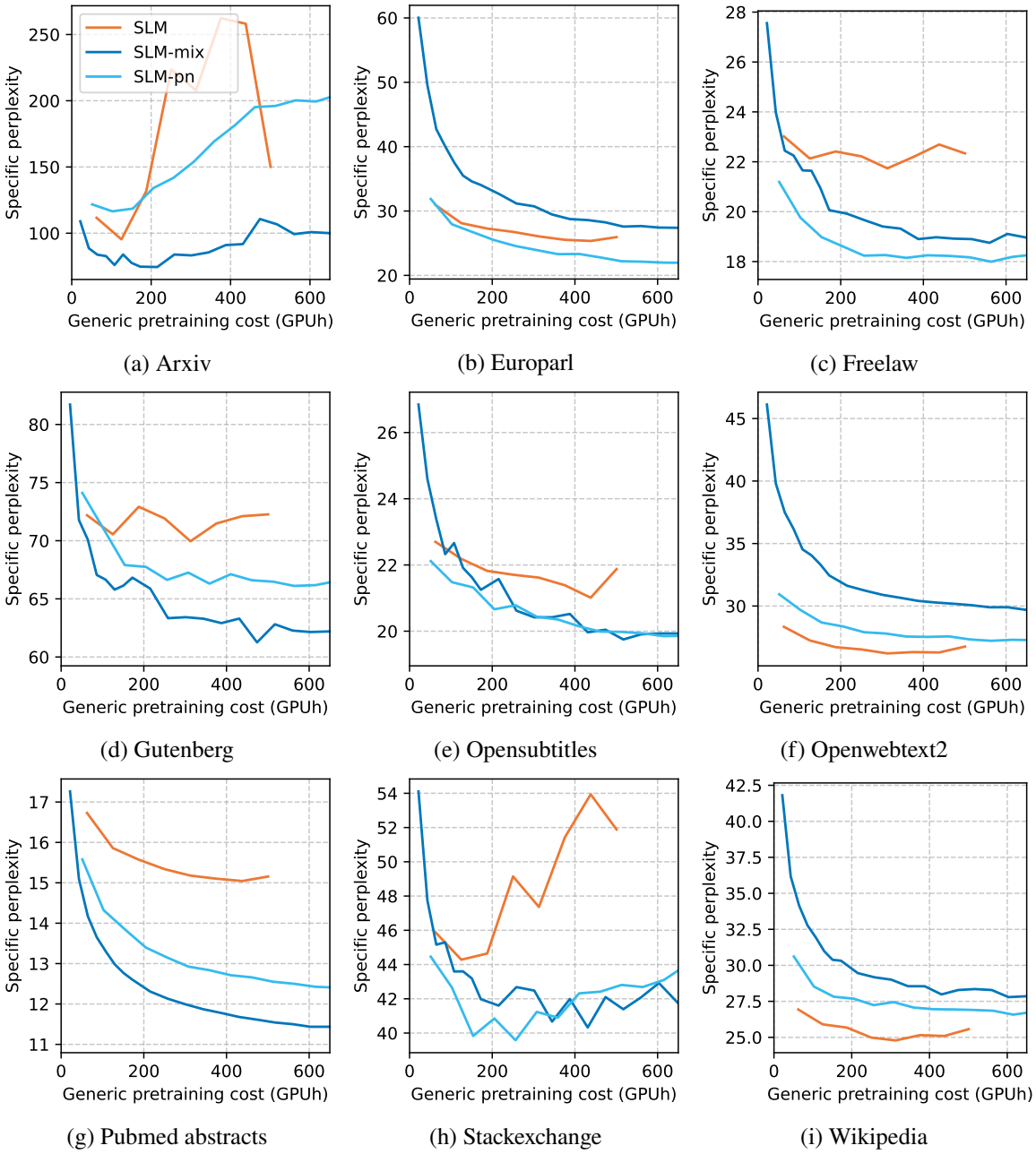


Figure 10. Specialized perplexity on individual subsets after fine-tuning on 1M tokens.