Vocabulary Customization for Efficient Domain-Specific LLM Deployment

Christian Herold* **Michael Kozielski** **Nicholas Santavas**
Yannick Versley **Shahram Khadivi**
eBay Inc.

Abstract

When using an LLM to process text outside the training domain(s), an often overlooked factor is vocabulary mismatch, where the general-domain tokenizer fails to capture frequent domain-specific terms, leading to higher token fertility and thus a decrease in processing speed due to suboptimal sub-word splits.

We address this limitation by augmenting the pretrained vocabulary with a set of domain-specific tokens. To this end, we design an algorithm that extends an existing tokenizer while guaranteeing it never decreases tokenization efficiency: every input sequence is segmented into at most the same number of tokens as before.

Evaluated on real-world e-commerce use-cases, the augmented tokenizer significantly shortens input sequences by up to 20% and reduces inference latency on downstream tasks while preserving predictive quality. We further analyze secondary effects, such as the impact on forward pass speed and the rate at which the model adopts the newly introduced tokens, to illustrate the broader benefits of vocabulary adaptation.

1 Introduction

Large Language Models (LLMs) have been established as state-of-the-art approaches for countless downstream applications across many domains and languages, still it is a common occurrence that they are adapted either to underrepresented/unseen languages or to new domains, such as healthcare, finance or e-commerce [Artetxe et al., 2020, Peng et al., 2024, Palen-Michel et al., 2024]. While it is relatively common to see tokenizer adaptation for new languages (see also Section 2), we argue that extending the tokenizer can also show significant benefits for domain adaptation. For example, in e-commerce we can improve the modeling of the language occurring in brand names, stock-keeping units or multilingual descriptors, which occur with high regularity but are often not represented as single tokens in the original vocabulary.

When we extend the vocabulary with a tailored set of additional tokens that cover frequent or semantically critical domain terms, there are a number of tradeoffs and non-trivial questions on which we want to shed light in this paper, specifically: (i) How and to what extent should we select candidate tokens so that the resulting token set maximizes compression without bloating the embedding table? (ii) Can we modify an existing tokenizer incrementally and deterministically, such that the new segmentation is never worse than the original, thus preserving backward compatibility with legacy inputs? (iii) What are the measurable gains, not only in static token counts but also in end-to-end inference time and prediction quality, when the augmented tokenizer is paired with a domain-specific model?

*Corresponding author. Email: cherold@ebay.com.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Continual and Compatible Foundation Model Updates (CCFM).

We answer these questions through three main contributions.

1. We propose a new algorithm for vocabulary extension that is guaranteed to always result in equal or fewer tokens being used.
2. We show that vocabulary extension can be applied to make inference with autoregressive LLMs more efficient for a specific domain (e-commerce) by up to 20% without degrading model quality.
3. We highlight and investigate multiple implications of vocabulary extensions that are often ignored in previous work, such as the impact on forward pass speed and how frequently the final model utilizes the additional tokens.

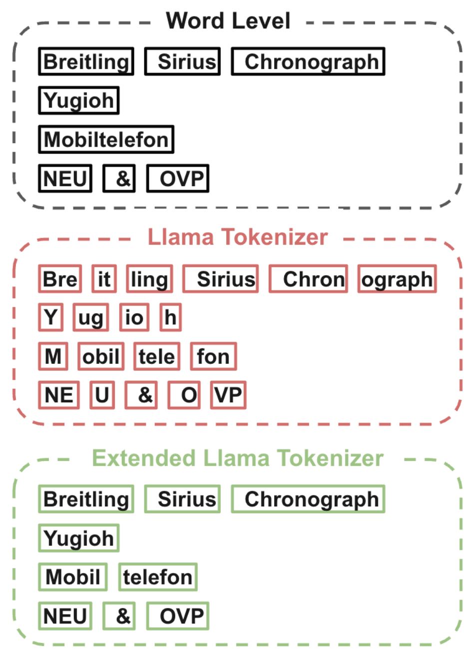

Figure 1: We extend the Llama 3.1 tokenizer with new vocabulary entries and merge operations, which are specific to the e-commerce domain. The result is a much more efficient tokenization for e-commerce specific phrases, which significantly reduces the cost of running such models in production.

By systematically addressing vocabulary mismatch, our work closes an often overlooked gap between laboratory-grade fine-tuning and production-ready LLM deployment. The proposed method is orthogonal to ongoing efforts in model quantization, speculative decoding, and optimized kernels, and can be combined with them to yield multiplicative efficiency gains, all while maintaining, or even enhancing, application-level performance.

2 Related Work

Vocabulary Extensions

Before the rise of autoregressive LLMs, several works have shown that the vocabulary of BERT-style non-autoregressive (encoder-only) models can be extended to adapt the model to a new domain like biomedicine [Tai et al., 2020, Pörner et al., 2020, Sachidananda et al., 2021], news [Mosin et al., 2023], legal Gee et al. [2022], and IT [Zhang et al., 2020, Yao et al., 2021]. In contrast to the present work, the main motivation to adapt the vocabulary of encoder-only models was to improve the model quality in the new domain, while here we are primarily focused on improving the inference efficiency. For encoder-only models, the task is also easier because we do not have to worry about if and how well the model decides to produce the new tokens, because these models are not used for text generation, but for classification etc.

In the realm of autoregressive LLMs, previous work is mostly limited to vocabulary extensions for the sake of expanding to new languages.

There exist many approaches that change the existing vocabulary of LLMs to new languages. This is achieved either by adding new tokens to the existing tokenizer [Yang et al., 2022, Cui et al., 2023, Balachandran, 2023, Larcher et al., 2023, Pipatanakul et al., 2023, Lin et al., 2024, Sha et al., 2025, Fujii et al., 2024, Choi et al., 2024, Nguyen et al., 2023, Tejaswi et al., 2024, Mundra et al., 2024, Yamaguchi et al., 2024c,a, Liao et al., 2024] or by replacing a certain number of tokens or even the full vocabulary [Csaki et al., 2023, Ostendorff and Rehm, 2023, Dalt et al., 2024, Remy et al., 2024, Yamaguchi et al., 2024b, Dobler and de Melo, 2024, Alexandrov et al., 2024, Gu et al., 2024]. From the above, only very few share the details on how the tokenizer extension is performed. Csaki et al. [2023] replace the least frequently used tokens with new ones and add the corresponding merges at the beginning of the merge list. Nguyen et al. [2023] add new tokens from an existing tokenizer based on some language-specific dataset but do not mention if and how they handle new merge operations. Yamaguchi et al. [2024c] extend the existing tokenizer with new, language-specific tokens by adding the corresponding merges at the beginning of the merge list.

Compared to language-adaptation, there are only few works that investigate vocabulary customization of autoregressive LLMs for domain-adaptation. Nakash et al. [2025] introduce a framework called *AdaptiVocab* where they replace existing tokens with domain-specific n-gram-based tokens to improve decoding efficiency for specific domains. The downside of their approach is that inputs that do not fall into the right distribution, it can happen that more tokens are needed than with the original tokenization, while our approach guarantees that always equal or fewer tokens are being used. At the time of writing, they have not yet released their code. Liu et al. [2024] propose an algorithm called *VEGAD* to extend the tokenizer for the legal and medical domains. However, they do not discuss how merge operations are added and they focus on quality improvement rather then efficiency. Dagan et al. [2024] show that inference efficiency on code benchmarks can be improved by adding domain-specific tokens to the model vocabulary. As above work, they also do not discuss how merge operations are added to the existing tokenizer.

Regarding the initialization of the new embedding vectors, previous work has shown that using the average of the existing token embeddings for the respective new tokens performs better than random initialization [Yao et al., 2021].

3 Methodology

In a subword tokenizer [Sennrich et al., 2016], the list of words is split into pieces using two main components:[2]

1. The **list of merge operations** tells us which tokens should be merged. A merge operation is a tuple consisting of two strings, e.g., `('th', 'e')`. This example tells us that we need to merge the tokens `'th'` and `'e'`, forming a new token `'the'`. The ordering of the list of merge operations tells us in which order we need to apply these merges.

2. The **model vocabulary** is a dictionary that maps each token to a certain index. One entry in the vocabulary could look like this: `'th': 873`, indicating that the token `'th'` is mapped to the integer `873`. The value of the index corresponds to the order in which the tokens were added to the tokenizer. That means for example the tokens `'t'` and `'h'` will have an index that is smaller than the index of the token `'th'` which is the result of merging the two.

When the tokenizer is applied to input text, it first decomposes the text at the byte level (for UTF-8 in Western languages, characters). We then consider the ordered list of merge operations: For each merge operation, we match the corresponding pairs in the text and merge the two tokens to form a single new token. After exhausting the full list of merge operations, the model vocabulary maps each remaining token to an integer, which form the LLM's input.

To extend the vocabulary of an existing tokenizer, it is necessary to modify both the model vocabulary and the list of merge operations. In our approach to creating a domain-adapted tokenizer we have the following steps, described in detail in Appendix A.3:

1. Training of an in-domain tokenizer using a domain-specific dataset.

2. Extension of the existing 'original' tokenizer with new in-domain tokens from the tokenizer trained in (1).

3. Initialization of the new embedding and projection vectors in the tokenizer-extended LLM.

4. Continuous training of the tokenizer-extended LLM to optimize for the new vocabulary.

5. Evaluation of the final tokenizer-extended LLM.

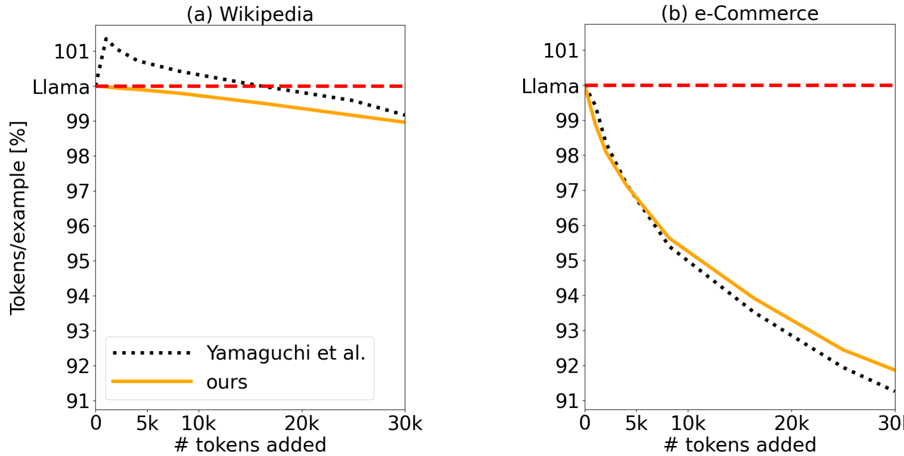

Figure 2: Impact of extending the Llama-3.1 tokenizer with e-commerce specific tokens. Shown is the average number of tokens needed to encode a document vs the number of new tokens added to the tokenizer. We compare our algorithm for tokenizer extension against Yamaguchi et al. [2024c]. Impact on tokenization of (a) Wikipedia articles; (b) downstream e-commerce tasks.

4 Experiments

4.1 Experimental setup

For tokenizer training, we utilize the Hugging Face tokenizers library [Wolf et al., 2019]. For tokenizer evaluation, we employ a comprehensive set of multilingual, e-commerce-specific, in-house downstream tasks (14 tasks in total).

As the starting LLM for tokenizer extension, we use a version of the Llama-3.1 8B model that has already been adapted towards the e-commerce domain via continuous pretraining (see Herold et al. 2025 for details). This adaptation has been done without changes to the tokenizer, so the model still uses the original Llama 3.1 tokenizer with 128k vocabulary size Dubey et al. [2024].

For LLM continued training, we utilize the same data as Herold et al. [2025], which consists of a multilingual, 50-50 mixture of general domain and e-commerce-specific data. As training framework, we use the Megatron-LM framework from NVIDIA [Shoeybi et al., 2019, Narayanan et al., 2021]. Training was conducted using 60 nodes, each having 8 NVIDIA H100 80GB GPUs (a total of 480 GPUs). The GPUs are connected via NVIDIA NVLink (intra-node) and InfiniBand (inter-node). Training as described in Section 4.3 takes less than 24h. The hardware is part of the eBay compute platform.

4.2 Vocabulary-size trade-off study

In terms of the efficiency/speed gains from the tokenizer extension, there is a tradeoff determined by two aspects: On the one hand, adding additional in-domain tokens should (hopefully) reduce the amount of tokens needed for the desired model output, and hence less computation in the hidden layers of the LLM overall. On the other hand, increasing the vocabulary size leads to more computation (per token) in computing embeddings and logits, since the embedding/projection matrices are larger. We have to consider this tradeoff and find an optimal point that maximizes the requests per second (RPS) of the deployed model.

Fortunately, this can be done based on an input sample and the model geometry and without expensive LLM training. We build multiple versions of the extended tokenizer with varying amount of tokens being added, and subsequently check the average number of tokens needed to encode the text sample. We test the encoding efficiency for a general domain dataset (English Wikipedia), as well as for our

[2]Most tokenizers have further components, like pre- and post-processing operations, special tokens, special templates etc. But these remain unaffected by the tokenizer extension framework and will just be copied over from the original tokenizer.

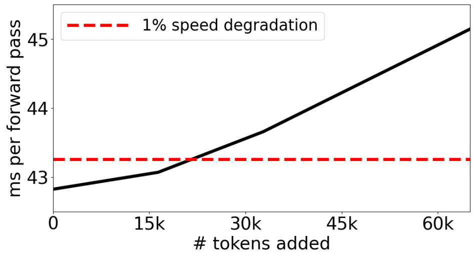

Figure 3: Impact of adding additional tokens to the Llama-3.1 8B model (and therefore increasing the size of embedding and projection matrices) on the speed of a single forward pass. Model is deployed via vLLM on a single H100 GPU.

Tokenizer	# input/output words	RPS	efficiency gain
Llama	300	29.19	-
Llama ext.		35.23	20.7%
Llama	3000	1.95	-
Llama ext.		2.52	29.2%

Table 1: Throughput of the Llama-3.1 8B model for a fixed number of input and output words. With the extended Llama tokenizer we need 20% fewer tokens both for model input and output, resulting in a respective gain in RPS. Model is deployed via vLLM on a single H100 GPU.

set of in-house downstream tasks. We compare our algorithm outlined in Algorithm 1 against the algorithm of Yamaguchi et al. [2024c], at the time of writing the only who have released their code and exact algorithm for tokenizer extension. The results can be seen in Figure 2.

As anticipated in the introduction, we find a more pronounced effect from extending the tokenizer with e-commerce-specific tokens on the specialized domain (b) than on general-domain Wikipedia texts (a). While Yamaguchi et al.'s approach prepends the merge operations to the list, our approach is more conservative by appending the new merges. This avoids the surprising decrease in efficiency when adding more tokens seen for general-domain texts, but leads to slower gains in efficiency in specialized text. Considering potential yet-unseen tasks, we think that our more conservative approach with a guarantee of an upper limit on the tokenization is preferable.

Figure 3 shows the effect that the increased size of embedding and projection matrices have on the forward pass timing (8B parameters, in vLLM framework, 300 tok/sequence at a batch size of 128 on H100). For our 8B model, we find 30k additional tokens to be a good tradeoff between encoding and forward pass efficiency. With 30k additional tokens, the forward pass duration increases by 1%. Taking into account the token efficiency however, we can expect an average speedup of 8% on e-commerce-specific downstream tasks (see Figure 2(b)), with up to 20% speedup on specific tasks.[3]

For a more direct comparison, we measure the inference RPS of base and tokenizer-extended LLM for different input/output sequence lengths in Table 1. A 20% reduction of tokens with the extended tokenizer, as we have seen for some of our downstream tasks, then shows an expected efficiency gain of around 20% for the shorter 300 words sequences, while for 3000 word sequences, this gain reaches almost 30% due to attention calculation having a larger impact. This demonstrates that tokenizer extension is even more beneficial when dealing with longer sequences.

4.3 Impact on model quality

Besides the efficiency tradeoff discussed above, we need to consider model quality, which may be affected by the continuous training necessary for new embeddings. Using the findings from the previous section, we extend the Llama-3 tokenizer with 30k new in-domain tokens. Following Yao et al. [2021], we initialize the new embedding and projection vectors with the average of the corresponding vectors from the existing tokens (see Algorithm 2). We continue to train the LLM for 10,000 iterations.[4] After training, we evaluate the LLM on the same tasks as in Herold et al. [2025]. The results can be seen in Table 2. We find that the tokenizer-extended LLM exhibits the same quality both on general domain and in-domain tasks as the domain adapted e-Llama model we started with.

[3]We note here that the tradeoff is model size dependent, as Dagan et al. [2024] also point out: for larger models, the impact will be smaller and vice versa.

[4]Learning rate is reduced in a cosine schedule from 1.0e-5 to 5.0e-7, all other hyperparameters are identical to Herold et al. [2025].

Model	General domain benchmarks (↑)					E-commerce benchmarks (↑)	
	En			non-En		En	non-En
	NLU	Lead.	MMLU	NLU	Lead.	avg.	avg.
8B LLM	71.6	12.6	63.5	54.0	42.4	60.5	47.9
+ extend vocab	71.8	12.1	63.4	53.7	42.9	60.1	47.6

Table 2: Impact of extending the Llama-3.1 model vocabulary with additional e-commerce specific tokens on the model quality.

sequence length [# words]	New tokenization being used
<15	95.3%
15 - 49	98.0%
>= 50	97.8%

Table 3: How frequently is the new tokenization used vs the old tokenization. For sequences larger than 15 words, the adapted model prefers the new tokenization in roughly 98% of cases.

4.4 Behavior of Tokenizer-Extended LLM

In this section we discuss how well the tokenizer-extended LLM actually utilizes the newly added in-domain tokens. In theory, the model could simply ignore all new tokens during generation, and we do not get any benefit of the tokenizer extension. We note that this only applies to the text generation. The input that the model receives at the beginning will of course always be tokenized using the new tokens, so here we are guaranteed to get the benefit of more efficient encoding. To the best of our knowledge, we are the first to analyze this crucial part of LLM tokenizer extension.

We take again our set of in-house downstream tasks. For each example, we go through the text word-by-word. For each word, we look at the probability distribution of the tokenizer-extended LLM model, given all previous words as context. Going through the words, we count, how often the model wants to predict the 'old' tokenized version of the word, and how often it wants to predict the 'new' tokenized version. The results are shown in Table 3.

We find, that in almost all cases, the model prefers the new tokenization vs the old one. For short sequences <15 words, there is still around 5% chance that the model produces the old tokenization, but for longer sequences, the adapted model prefers the new tokenization in almost 100% of cases.

5 Conclusion

In this work we identify vocabulary mismatch as an often-overlooked, yet decisive, bottleneck when autoregressive LLMs are moved from broad-domain pretraining to the specialized distribution of a specific target domain. We present a tokenizer-extension algorithm that adds the most frequent in-domain tokens without ever increasing the number of tokens required to encode any sequence, thus guaranteeing backward compatibility.

Experiments on a multilingual, production-grade e-commerce set of in-house downstream tasks show that the extended tokenizer shortens input sequences by up to 20%. When deployed, this translates into 20%-30% higher throughput, depending on prompt length. Importantly, this is achieved without compromising model quality on general or in-domain tasks. In additional studies, we find that the model learns to emit the new tokens in ≈98% of cases for sequences longer than 15 words, confirming that the additional vocabulary is effectively embraced rather than ignored.

References

Anton Alexandrov, Veselin Raychev, Mark Mueller, Ce Zhang, Martin T. Vechev, and Kristina Toutanova. Mitigating catastrophic forgetting in language transfer via model merging. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Associa-*

tion for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024, pages 17167–17186. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-EMNLP.1000. URL `https://doi.org/10.18653/v1/2024.findings-emnlp.1000`.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. On the cross-lingual transferability of monolingual representations. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.421. URL `https://aclanthology.org/2020.acl-main.421/`.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=4WnqRR915j`.

Abhinand Balachandran. Tamil-llama: A new tamil language model based on llama 2. *CoRR*, abs/2311.05845, 2023. doi: 10.48550/ARXIV.2311.05845. URL `https://doi.org/10.48550/arXiv.2311.05845`.

Zeming Chen, Alejandro Hernández-Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, Alexandre Sallinen, Alireza Sakhaeirad, Vinitra Swamy, Igor Krawczuk, Deniz Bayazit, Axel Marmet, Syrielle Montariol, Mary-Anne Hartley, Martin Jaggi, and Antoine Bosselut. MEDITRON-70B: scaling medical pretraining for large language models. *CoRR*, abs/2311.16079, 2023. doi: 10.48550/ARXIV.2311.16079. URL `https://doi.org/10.48550/arXiv.2311.16079`.

ChangSu Choi, Yongbin Jeong, Seoyoon Park, Inho Won, HyeonSeok Lim, Sangmin Kim, Yejee Kang, Chanhyuk Yoon, Jaewan Park, Yiseul Lee, Hyejin Lee, Younggyun Hahm, Hansaem Kim, and Kyungtae Lim. Optimizing language augmentation for multilingual large language models: A case study on korean. In Nicoletta Calzolari, Min-Yen Kan, Véronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 12514–12526. ELRA and ICCL, 2024. URL `https://aclanthology.org/2024.lrec-main.1095`.

Zoltan Csaki, Pian Pawakapan, Urmish Thakker, and Qiantong Xu. Efficiently adapting pretrained language models to new languages. *CoRR*, abs/2311.05741, 2023. doi: 10.48550/ARXIV.2311.05741. URL `https://doi.org/10.48550/arXiv.2311.05741`.

Yiming Cui, Ziqing Yang, and Xin Yao. Efficient and effective text encoding for chinese llama and alpaca. *CoRR*, abs/2304.08177, 2023. doi: 10.48550/ARXIV.2304.08177. URL `https://doi.org/10.48550/arXiv.2304.08177`.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. Getting the most out of your tokenizer for pre-training and domain adaptation. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=ZFYBnLljtT`.

Severino Da Dalt, Joan Llop, Irene Baucells, Marc Pàmies, Yishi Xu, Aitor Gonzalez-Agirre, and Marta Villegas. FLOR: on the effectiveness of language adaptation. In Nicoletta Calzolari, Min-Yen Kan, Véronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 7377–7388. ELRA and ICCL, 2024. URL `https://aclanthology.org/2024.lrec-main.650`.

Konstantin Dobler and Gerard de Melo. Language adaptation on a tight academic compute budget: Tokenizer swapping works and pure bfloat16 is enough. *CoRR*, abs/2408.15793, 2024. doi: 10.48550/ARXIV.2408.15793. URL `https://doi.org/10.48550/arXiv.2408.15793`.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. Continual pre-training for cross-lingual LLM adaptation: Enhancing japanese language capabilities. *CoRR*, abs/2404.17790, 2024. doi: 10.48550/ARXIV.2404.17790. URL https://doi.org/10.48550/arXiv.2404.17790.

Leonidas Gee, Andrea Zugarini, Leonardo Rigutini, and Paolo Torroni. Fast vocabulary transfer for language model compression. In Yunyao Li and Angeliki Lazaridou, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: EMNLP 2022 - Industry Track, Abu Dhabi, UAE, December 7 - 11, 2022*, pages 409–416. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.EMNLP-INDUSTRY.41. URL https://doi.org/10.18653/v1/2022.emnlp-industry.41.

Shuhao Gu, Mengdi Zhao, Bowen Zhang, Liangdong Wang, Jijie Li, and Guang Liu. Retok: Replacing tokenizer to enhance representation efficiency in large language model. *CoRR*, abs/2410.04335, 2024. doi: 10.48550/ARXIV.2410.04335. URL https://doi.org/10.48550/arXiv.2410.04335.

Christian Herold, Michael Kozielski, Tala Bazazo, Pavel Petrushkov, Yannick Versley, Seyyed Hadi Hashemi, Patrycja Cieplicka, Dominika Basaj, and Shahram Khadivi. Domain adaptation of foundation LLMs for e-commerce. In Georg Rehm and Yunyao Li, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1039–1049, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-288-6. doi: 10.18653/v1/2025.acl-industry.74. URL https://aclanthology.org/2025.acl-industry.74/.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019. URL http://proceedings.mlr.press/v97/houlsby19a.html.

Erik Johannes Husom, Arda Goknil, Merve Astekin, Lwin Khin Shar, Andre Kåsen, Sagar Sen, Benedikt Andreas Mithassel, and Ahmet Soylu. Sustainable LLM inference for edge AI: evaluating quantized llms for energy efficiency, output accuracy, and inference latency. *CoRR*, abs/2504.03360, 2025. doi: 10.48550/ARXIV.2504.03360. URL https://doi.org/10.48550/arXiv.2504.03360.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In Jason Flinn, Margo I. Seltzer, Peter Druschel, Antoine Kaufmann, and Jonathan Mace, editors, *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M. Rush. Block pruning for faster transformers. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 10619–10629. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.829. URL https://doi.org/10.18653/v1/2021.emnlp-main.829.

Celio Larcher, Marcos Piau, Paulo Finardi, Pedro Gengo, Piero Esposito, and Vinicius F. Carida. Cabrita: closing the gap for foreign languages. *CoRR*, abs/2308.11878, 2023. doi: 10.48550/ARXIV.2308.11878. URL https://doi.org/10.48550/arXiv.2308.11878.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.243. URL https://doi.org/10.18653/v1/2021.emnlp-main.243.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Baohao Liao, Christian Herold, Shahram Khadivi, and Christof Monz. IKUN for WMT24 general MT task: Llms are here for multilingual machine translation. In Barry Haddow, Tom Kocmi, Philipp Koehn, and Christof Monz, editors, *Proceedings of the Ninth Conference on Machine Translation, WMT 2024, Miami, FL, USA, November 15-16, 2024*, pages 263–269. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.WMT-1.19. URL https://doi.org/10.18653/v1/2024.wmt-1.19.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André F. T. Martins, and Hinrich Schütze. Mala-500: Massive language adaptation of large language models. *CoRR*, abs/2401.13303, 2024. doi: 10.48550/ARXIV.2401.13303. URL https://doi.org/10.48550/arXiv.2401.13303.

Chengyuan Liu, Shihang Wang, Lizhi Qing, Kun Kuang, Yangyang Kang, Changlong Sun, and Fei Wu. Gold panning in vocabulary: An adaptive method for vocabulary expansion of domain-specific llms. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 7442–7459. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.EMNLP-MAIN.424. URL https://doi.org/10.18653/v1/2024.emnlp-main.424.

Vladislav D. Mosin, Igor Samenko, Borislav Kozlovskii, Alexey Tikhonov, and Ivan P. Yamshchikov. Fine-tuning transformers: Vocabulary transfer. *Artif. Intell.*, 317:103860, 2023. doi: 10.1016/J.ARTINT.2023.103860. URL https://doi.org/10.1016/j.artint.2023.103860.

Nandini Mundra, Aditya Nanda Kishore, Raj Dabre, Ratish Puduppully, Anoop Kunchukuttan, and Mitesh M. Khapra. An empirical comparison of vocabulary expansion and initialization approaches for language models. *CoRR*, abs/2407.05841, 2024. doi: 10.48550/ARXIV.2407.05841. URL https://doi.org/10.48550/arXiv.2407.05841.

Itay Nakash, Nitay Calderon, Eyal Ben-David, Elad Hoffer, and Roi Reichart. Adaptivocab: Enhancing LLM efficiency in focused domains through lightweight vocabulary adaptation. *CoRR*, abs/2503.19693, 2025. doi: 10.48550/ARXIV.2503.19693. URL https://doi.org/10.48550/arXiv.2503.19693.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on GPU clusters using megatron-lm. In Bronis R. de Supinski, Mary W. Hall, and Todd Gamblin, editors, *International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2021, St. Louis, Missouri, USA, November 14-19, 2021*, page 58. ACM, 2021. doi: 10.1145/3458817.3476209. URL https://doi.org/10.1145/3458817.3476209.

Xuan-Phi Nguyen, Wenxuan Zhang, Xin Li, Mahani Aljunied, Qingyu Tan, Liying Cheng, Guanzheng Chen, Yue Deng, Sen Yang, Chaoqun Liu, Hang Zhang, and Lidong Bing. Seallms - large language models for southeast asia. *CoRR*, abs/2312.00738, 2023. doi: 10.48550/ARXIV.2312.00738. URL https://doi.org/10.48550/arXiv.2312.00738.

Malte Ostendorff and Georg Rehm. Efficient language model training through cross-lingual and progressive transfer learning. *CoRR*, abs/2301.09626, 2023. doi: 10.48550/ARXIV.2301.09626. URL https://doi.org/10.48550/arXiv.2301.09626.

Chester Palen-Michel, Ruixiang Wang, Yipeng Zhang, David Yu, Canran Xu, and Zhe Wu. Investigating LLM applications in e-commerce. *CoRR*, abs/2408.12779, 2024. doi: 10.48550/ARXIV.2408.12779. URL https://doi.org/10.48550/arXiv.2408.12779.

Bo Peng, Xinyi Ling, Ziru Chen, Huan Sun, and Xia Ning. ecellm: Generalizing large language models for e-commerce from large-scale, high-quality instruction data. *CoRR*, abs/2402.08831, 2024. doi: 10.48550/ARXIV.2402.08831. URL https://doi.org/10.48550/arXiv.2402.08831.

Kunat Pipatanakul, Phatrasek Jirabovonvisut, Potsawee Manakul, Sittipong Sripaisarnmongkol, Ruangsak Patomwong, Pathomporn Chokchainant, and Kasima Tharnpipitchai. Typhoon: Thai large language models. *CoRR*, abs/2312.13951, 2023. doi: 10.48550/ARXIV.2312.13951. URL https://doi.org/10.48550/arXiv.2312.13951.

Nina Pörner, Ulli Waltinger, and Hinrich Schütze. Inexpensive domain adaptation of pretrained language models: Case studies on biomedical NER and covid-19 QA. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1482–1490. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.FINDINGS-EMNLP.134. URL https://doi.org/10.18653/v1/2020.findings-emnlp.134.

Ishrak Jahan Ratul, Yuxiao Zhou, and Kecheng Yang. Accelerating deep learning inference: A comparative analysis of modern acceleration frameworks. *Electronics*, 14(15), 2025. ISSN 2079-9292. doi: 10.3390/electronics14152977. URL https://www.mdpi.com/2079-9292/14/15/2977.

François Remy, Pieter Delobelle, Hayastan Avetisyan, Alfiya Khabibullina, Miryam de Lhoneux, and Thomas Demeester. Trans-tokenization and cross-lingual vocabulary transfers: Language adaptation of llms for low-resource NLP. *CoRR*, abs/2408.04303, 2024. doi: 10.48550/ARXIV.2408.04303. URL https://doi.org/10.48550/arXiv.2408.04303.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950, 2023. doi: 10.48550/ARXIV.2308.12950. URL https://doi.org/10.48550/arXiv.2308.12950.

Vin Sachidananda, Jason S. Kessler, and Yi'an Lai. Efficient domain adaptation of language models via adaptive tokenization. In Nafise Sadat Moosavi, Iryna Gurevych, Angela Fan, Thomas Wolf, Yufang Hou, Ana Marasovic, and Sujith Ravi, editors, *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, SustaiNLP@EMNLP 2021, Virtual, November 10, 2021*, pages 155–165. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.SUSTAINLP-1.16. URL https://doi.org/10.18653/v1/2021.sustainlp-1.16.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Jiu Sha, Mengxiao Zhu, Chong Feng, and Yuming Shang. Veef-multi-llm: Effective vocabulary expansion and parameter efficient finetuning towards multilingual large language models. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 7963–7981. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.533/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. URL http://arxiv.org/abs/1909.08053.

Wen Tai, H. T. Kung, Xin Dong, Marcus Z. Comiter, and Chang-Fu Kuo. exbert: Extending pre-trained models with domain-specific vocabulary under constrained training resources. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1433–1439. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.FINDINGS-EMNLP.129. URL https://doi.org/10.18653/v1/2020.findings-emnlp.129.

Atula Tejaswi, Nilesh Gupta, and Eunsol Choi. Exploring design choices for building language-specific llms. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 10485–10500. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-EMNLP.614. URL https://doi.org/10.18653/v1/2024.findings-emnlp.614.

David Thulke, Yingbo Gao, Petrus Pelser, Rein Brune, Rricha Jalota, Floris Fok, Michael Ramos, Ian van Wyk, Abdallah Nasir, Hayden Goldstein, Taylor Tragemann, Katie Nguyen, Ariana Fowler, Andrew Stanco, Jon Gabriel, Jordan Taylor, Dean Moro, Evgenii Tsymbalov, Juliette de Waal, Evgeny Matusov, Mudar Yaghi, Mohammad Shihadah, Hermann Ney, Christian Dugast, Jonathan Dotan, and Daniel Erasmus. Climategpt: Towards AI synthesizing interdisciplinary research on climate change. *CoRR*, abs/2401.09646, 2024. doi: 10.48550/ARXIV.2401.09646. URL https://doi.org/10.48550/arXiv.2401.09646.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019. URL http://arxiv.org/abs/1910.03771.

Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR, 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Benfeng Xu, Chunxu Zhao, Wenbin Jiang, PengFei Zhu, Songtai Dai, Chao Pang, Zhuo Sun, Shuohuan Wang, and Yu Sun. Retrieval-augmented domain adaptation of language models. In Burcu Can, Maximilian Mozes, Samuel Cahyawijaya, Naomi Saphra, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Chen Zhao, Isabelle Augenstein, Anna Rogers, Kyunghyun Cho, Edward Grefenstette, and Lena Voita, editors, *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 54–64, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.repl4nlp-1.5. URL `https://aclanthology.org/2023.repl4nlp-1.5/`.

Atsuki Yamaguchi, Terufumi Morishita, Aline Villavicencio, and Nikolaos Aletras. Elchat: Adapting chat language models using only target unlabeled language data. *arXiv preprint arXiv:2412.11704*, 2024a.

Atsuki Yamaguchi, Aline Villavicencio, and Nikolaos Aletras. An empirical study on cross-lingual vocabulary adaptation for efficient language model inference. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 6760–6785. Association for Computational Linguistics, 2024b. doi: 10.18653/V1/2024.FINDINGS-EMNLP.396. URL `https://doi.org/10.18653/v1/2024.findings-emnlp.396`.

Atsuki Yamaguchi, Aline Villavicencio, and Nikolaos Aletras. How can we effectively expand the vocabulary of llms with 0.01 gb of target language text? *arXiv preprint arXiv:2406.11477*, 2024c.

Ziqing Yang, Zihang Xu, Yiming Cui, Baoxin Wang, Min Lin, Dayong Wu, and Zhigang Chen. CINO: A chinese minority pre-trained language model. In Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, and Seung-Hoon Na, editors, *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 3937–3949. International Committee on Computational Linguistics, 2022. URL `https://aclanthology.org/2022.coling-1.346`.

Yunzhi Yao, Shaohan Huang, Wenhui Wang, Li Dong, and Furu Wei. Adapt-and-distill: Developing small, fast and effective pretrained language models for domains. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 460–470. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.FINDINGS-ACL.40. URL `https://doi.org/10.18653/v1/2021.findings-acl.40`.

Rong Zhang, Revanth Gangi Reddy, Md. Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avirup Sil, and Todd Ward. Multi-stage pre-training for low-resource domain adaptation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5461–5468. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.440. URL `https://doi.org/10.18653/v1/2020.emnlp-main.440`.

A Technical Appendices and Supplementary Material

A.1 Limitations

In this study we focus on a single domain, namely e-commerce, since this is the domain of our in-house downstream tasks. Potentially, the behavior of our approach might be different for different domains. Furthermore, we focus on a single model family, namely the Llama-3 models, since they are the most relevant for us at this time. The behavior of other model families might be different in terms of potential savings or the change in model quality after the tokenizer extension.

A.2 Related Work - Continuation

LLM Efficiency

Efficient LLM inference can leverage a spectrum of model compression and system-level optimizations. Model quantization reduces weight/activation precision (e.g., 8-bit or 4-bit) to shrink memory and computation, achieving significant throughput gains with negligible accuracy loss Xiao et al. [2023]. Similarly, model pruning eliminates redundant parameters or entire sub-modules (e.g., attention heads or layers), yielding substantially smaller and faster models with minimal performance drop Lagunas et al. [2021]. Knowledge distillation trains compact student models to mimic large teacher models, retaining most of the original accuracy with far fewer parameters and faster runtime Sanh et al. [2019]. Beyond model-centric methods, compilation frameworks (e.g., NVIDIA TensorRT and Apache TVM) convert neural network graphs into optimized executables via kernel fusion, mixed-precision and other low-level optimizations, greatly improving throughput on target hardware Ratul et al. [2025]. Finally, edge deployment of LLMs has emerged, using aggressive compression and efficient runtimes to run models on resource-constrained devices Husom et al. [2025].

Our approach of vocabulary extension is fully orthogonal to existing efficiency-related optimizations. It can be used on top of any highly efficient deployment to further reduce cost and latency.

Domain Adaptation

Large pretrained LLMs often struggle with domain-specific tasks, motivating targeted domain adaptation Lewkowycz et al. [2022], Chen et al. [2023], Rozière et al. [2023]. Continuing pretraining on in-domain text or fine-tuning the entire model can substantially boost performance on domain-specific tasks Azerbayev et al. [2024], Shao et al. [2024], Thulke et al. [2024], but performing these is quite expensive. To improve efficiency, parameter-efficient methods have emerged: adapter-based fine-tuning inserts small trainable modules into the network while keeping most weights frozen Houlsby et al. [2019], and prompt tuning optimizes a few soft prompt vectors with the model fixed Lester et al. [2021]. Retrieval-based approaches further inject relevant domain knowledge at inference time, enabling LLMs to leverage external in-domain data on the fly instead of exhaustive retraining Xu et al. [2023].

A.3 Detailed Methodology

A.3.1 Training a Domain-Specific Tokenizer

The first step involves the training of a new in-domain tokenizer model on some domain-specific dataset. It is important that we choose sufficient training data representative of the target domain, ideally encompassing all potential downstream tasks. For the tokenizer, we only care about the vocabulary, we can discard the list of merge operations. Since we expect a significant vocabulary overlap between the original tokenizer and the in-domain one, we should set the target token size for training to a higher number than the number of tokens we want to add to the original tokenizer.

A.3.2 Extending the Original Tokenizer

We initialize the extended tokenizer with the original tokenizer. We then traverse the vocabulary of the in-domain tokenizer in the default order. For each token, we tokenize it using the current extended tokenizer. If the result is a single token, this means this token is already in the vocabulary of the extended tokenizer and we can continue without doing anything. However, if the token is split into two tokens, we append the tuple of the two tokens at the end of the list of merge operations of the extended tokenizer and also add the token into the model vocabulary. We want to point out, that following this approach, we will never encounter the case where the token is split into three or more tokens, because all except one of the corresponding merge operations will already have been added to the extended tokenizer in an earlier step. We repeat the above until we have added a fixed amount of new tokens to the tokenizer. The detailed algorithm is outlined in Algorithm 1.

Since we append the new merge operations always to the end of the list, the first part of the final list of merge operations will be identical to that of the original tokenizer. The merges that are appended to the list (see Algorithm 1 line 14) can only reduce (or leave unchanged) token counts because earlier merges are unaffected by later ones. That means, contrary to existing approaches, we will never have the situation where our extended tokenizer results in a worse tokenization than the original one.

A.3.3 Embedding Initialization

Since we are extending the LLM vocabulary, we also need to initialize the newly added embedding and projection vectors in the LLM. The baseline method comprises of using random initialization, but previous work has shown that better performance can be achieved when initializing the new token-embedding as the average of the embeddings of the existing tokens that the new token is composed of Yao et al. [2021]. Therefore we follow this approach of average initialization. The detailed algorithm for embedding initialization is outlined in Algorithm 2.

A.3.4 Continuous Training

To adapt the tokenizer-extended LLM to the new extended vocabulary, we need to continuously train the model on some data. In our setting, we start with a model that is already adapted towards the e-commerce domain, so we just sample training data from the same distribution that was used to adapt the model to the domain in the first place. We perform full training of all model parameters, not freezing any part of the network.

A.3.5 Final Evaluation

We need to evaluate the final tokenizer-extended LLM in several aspects:

- What is the inference speedup on in-domain downstream tasks?
- How has the model quality changed?
- To what extent is the model utilizing the new tokenizations vs the old one?

To measure the model quality, we utilize the same set of public and e-commerce specific benchmarks as in Herold et al. [2025]. To measure the inference speed and utilization of new tokens, we design a set of experiments utilizing a comprehensive set of in-house downstream tasks and LLM deployment via vLLM Kwon et al. [2023].

Algorithm 1: Extend base BPE Tokenizer with in-domain tokens

Data: T_b: base tokenizer, T_d: in-domain tokenizer, N: number of tokens to add
Result: T_{ext}: extended tokenizer

$V_b \leftarrow$ vocab of T_b;
$M_b \leftarrow$ merges of T_b;
$V_d \leftarrow$ vocab of T_d, sorted by descending frequency;
$i \leftarrow 0$;
foreach $t \in V_d$ **do**
    if $t \in V_b$ ***or*** *T_b's pre-tokenizer splits t* **then**
        continue;
    $enc \leftarrow T_b$.encode(t);
    if $len(enc) \neq 2$ **then**
        continue;
    $[l, r] \leftarrow T_b$.encode(t);
    $m \leftarrow$ merge of l and r;
    if $m \notin M_b$ **then**
        append m to M_b;
    add t to V_b with new ID;
    $i \leftarrow i + 1$;
    if $i \geq N$ **then**
        break;
$T_{ext} \leftarrow T_b$;
return updated tokenizer T_{ext};

Algorithm 2: Initialize the in-domain token embeddings.

Data: T_b: base tokenizer, T_d: in-domain tokenizer, E_b: base token embeddings
Result: E_d: in-domain token embeddings

$V_b \leftarrow$ vocab of T_b;
$V_d \leftarrow$ vocab of T_d;
$E_d \leftarrow \{\}$;
foreach $t \in V_d$ **do**
    if $t \in V_b$ **then**
        continue;
    $e \leftarrow 0$;
    $t_b \leftarrow T_b$.encode(t);
    foreach $t_{old} \in t_b$ **do**
        $e \leftarrow e + E_b[t_{old}]$;
    $e \leftarrow e/\text{len}(t_b)$;
    add e to E_d;
return E_d;