Efficient Low-Resource Language Models Using Tokenizer Transfer

Gustaf Gren
Stockholm University
gustaf.gren@ling.su.se

Murathan Kurfalı
RISE Research Institutes of Sweden
murathan.kurfali@ri.se

Abstract

Training a language model for low-resource languages is challenging due to data scarcity and computational cost. Tokenizer transfer offers a way to adapt a pre-trained model to a new tokenizer without full retraining, improving efficiency and cross-lingual applicability. To the best of our knowledge, we present the first controlled evaluation of tokenizer transfer on monolingually pretrained base models trained on language-specific corpora. We evaluate Orthogonal Mapping Pursuit (OMP) and Fast Vocabulary Transfer (FVT) across six languages and multiple fine-tuning regimes. We computed byte-normalized log-perplexity and MultiBlimp accuracy for the Goldfish model family. We use these to evaluate for target-language adaptability and source-language retention. We add monolingual or mixed finetuning to compare with only using transfer. OMP with monolingual target finetuning achieves the best target-language performance, yielding lower log-perplexity and higher MultiBlimp scores than all evaluated baselines. These include (i) a model trained only on the source language, (ii) a model trained on a smaller amount of target-language data, and (iii) the source-language model adapted via standard finetuning on the target data. The results suggest tokenizer transfer is a compute-efficient alternative for low-resource LM training: train a monolingual tokenizer for the target language, transfer it to a larger pre-trained model, and fine-tune using the target data.

Code: github.com/skogsgren/tokeneval

1 Introduction

Language models are increasingly part of daily life. By late 2025, it is estimated that one in ten people worldwide have interacted with these systems, often in casual, non-work contexts (Chatterji et al., 2025). Yet, these models remain strongly English-centric (Veselovsky et al., 2025), largely because their training data is predominantly English (Blasi et al., 2022; Joshi et al., 2020). As a result, speakers of other languages have access to less effective tools (Ranathunga and de Silva, 2022; Qin et al., 2025). One potential contributing factor is tokenization: since most tokenizers are trained for English, other languages are encoded less efficiently (Minixhofer et al., 2024), requiring more tokens than for English. Consequently, users of non-English languages face higher monetary costs for equivalent tasks and experience lower performance (Petrov et al., 2023; Ahia et al., 2023).

Retraining the model and tokenizer on data that better represents the target languages is the most direct way to improve coverage, although the optimal language distribution remains an open question. Schäfer et al. (2024) found that a 90/10 split actually improved bilingual performance compared to a 50/50 split. In any case, such retraining is costly in both monetary and environmental terms and may be infeasible when sufficient data is unavailable. Researchers have begun investigating "tokenizer transfer," a method for reusing or adapting tokenizers for other languages without retraining the full model, either by aligning the new tokenizer to the previous embeddings, or by reinitializing new embeddings. While several approaches have been proposed, a comparison between tokenizer transfer methods and more traditional finetuning approaches across multiple languages on monolingually trained models is still lacking. Most prior evaluations use multilingual LMs or within-language settings, where multilingual pre-training can mask the effect of tokenizer transfer. We therefore evaluate monolingually pretrained models under several finetuning regimes to explore the effect of tokenizer transfer across different settings.

In this work, we evaluate tokenizer transfer as a low-resource adaptation strategy for monolingually trained language models. Using the Goldfish family, we run an all-pairs study over six languages and compare two transfer methods, Fast Vocab-

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 4: Student Research Workshop, pages 639–648

ulary Transfer (FVT) and Orthogonal Matching Pursuit (OMP), against standard finetuning baselines. Concretely, we evaluate 30 source→target pairs with 10 experimental conditions for each pair, yielding comparison of 300 total configurations. We (i) measure target-language adaptation and source-language performance preservation under no finetuning, target-only finetuning, and mixed source+target finetuning, and (ii) analyze how sensitive transfer performance is to the choice of source language for each target.

Our goal is to provide practical guidance on best practices for leveraging information from high-resource languages to benefit low-resource languages, specifically by evaluating how tokenizer transfer facilitates efficient adaptation.

2 Related Work

For tokenizer transfer the main challenge is how to initialize embeddings for new tokens, i.e. tokens that exist in the new tokenizer/vocabulary but not in the original model, while maintaining original model performance. This can be achieved, for example, with multilingual static embeddings (Minixhofer et al., 2022) or by leveraging shared token spaces between models (Dobler and de Melo, 2023).

Fast Vocabulary Transfer (FVT), introduced by Gee et al. (2022), initializes each new token embedding as the mean of its shared sub-token embeddings from the teacher model's vocabulary. For tokens that are identical we use the source embedding. For tokens that appear only in the target vocabulary V_{TGT} and not in the source vocabulary V_{SRC}, FVT constructs embeddings by decomposing each new token t_i using the source tokenizer T_{SRC}. The new embedding of t_i is defined as the mean of the embeddings of the resulting source tokens:

$$E_{\text{TGT}}(t_i) = \frac{1}{|T_{\text{SRC}}(t_i)|} \sum_{t_j \in T_{\text{SRC}}(t_i)} E_{\text{SRC}}(t_j)$$

Orthogonal Mapping Projection (OMP), introduced by Goddard and Neto (2025), approximates each new token by first expressing it as a sparse combination of shared anchor tokens in the donor embedding space, then applying the same sparse coefficients to reconstruct its embedding in the base model's space. For tokens that are absent from the base vocabulary, Orthogonal Matching Pursuit (OMP) reconstructs each target embedding as a sparse linear combination of shared anchor tokens. Concretely, we approximate E^{SRC} for the new embedding $E_{\text{TGT}}(t_i)$ by:

$$E^{\text{SRC}} \approx \sum_{j \in A} \alpha_j E_j^{\text{SRC}}$$

Where $A \subseteq V_{\text{SRC}} \cap V_{\text{TGT}}$ and $|A| < k$. A is the set of anchor tokens chosen by orthogonal matching pursuit (see Goddard and Neto (2025) for the full implementation), and k being the sparsity level (higher more granularity).

Other approaches generate new embeddings instead of aligning them. Zero Shot Tokenizer Transfer (ZeTT) (Minixhofer et al., 2024) trains a small hypernetwork to predict embeddings, allowing effectively zero-shot transfer after training of the hypernetwork. Approximate Likelihood Matching (ALM) (Minixhofer et al., 2025) treats tokenizer transfer as a knowledge distillation problem, identifying comparable token chunks and minimizing differences between their likelihoods. Recently, (Yamaguchi et al., 2025) study low-resource vocabulary expansion (adding new target-language tokens to an existing model) and compare several embedding-initialization heuristics.

FVT was evaluated entirely against English testdata. OMP uses a broad set of multitask benchmarks, but evaluates only on English. ZeTT was evaluated partly for multilingual scenarios, where ZeTT retained performance on Massive Multitask Language Understanding while reducing token length by 30%. However, systematic comparison of these methods across multiple non-English languages remains unexplored.

3 Method

Our goal is to evaluate tokenizer transfer strategies across different source/target languages. To that end, we choose the Goldfish family of models (Chang et al., 2024), a set of GPT-2 sized monolingual language models. They are trained on different sizes of training data: 5MB (350 languages), 10MB (288 languages), 100MB (166 languages), and 1000MB (83 languages). Crucially, Goldfish provides monolingual models trained on language-specific corpora. This isolates tokenizer transfer from prior multilingual exposure, allowing us to rigorously test whether transfer methods work in a strict monolingual setting without any shared pretrained representations. The modest size of these

Language	Size
English	9.6MB
Swedish	9.8MB
Danish	9.8MB
Estonian	9.3MB
Turkish	10.0MB
Scottish Gaelic	9.6MB

Table 1: Monolingual dataset sizes used for finetuning

models provides flexibility to conduct an extensive all-pairs evaluation across six languages even with our limited computational resources.

For tokenizer transfer strategies we chose OMP (Goddard and Neto, 2025) and FVT (Gee et al., 2022), both methods that align shared sub-token spaces between source and target. These were chosen over approaches like ZeTT primarily due to computational constraints when comparing a large set of language pairs / models, as we in the case of ZeTT would have to train a hyper-network for each pair.

For each source→target language pair, we establish the following four baselines:

- **10MB-*tgt*:** the monolingual 10MB target-language model.
- **100MB-*src*:** the monolingual 100MB source-language model.
- **100MB-*src* (Mono FT)**: Source model finetuned on target data.
- **100MB-*src* (Mixed FT)**: Source model finetuned on combined source+target data.

We then apply **FVT** and **OMP** to transfer the 100MB source model to the target model's 10MB tokenizer. For each technique, we evaluate the transferred models in three states:

- ***OMP/FVT (No FT)*:** The source model with the transferred tokenizer of the target language.
- ***OMP/FVT (Mono FT)*:** target-only finetuning data (10MB scaled by byte-premium).
- ***OMP/FVT (Mixed FT)*:** mixed source/target data (10+10MB scaled by byte-premium).

Byte-premium scaling, calculated per language in the Goldfish project, adjusts dataset sizes to approximate equivalent content across languages by accounting for byte efficiency. All finetuning uses 750 steps, batch size 8, and learning rate 0.0001. This yields approximately 3 million tokens or $\sim$ 15% of the original 10MB Goldfish models' training budget. Learning rate and batch size are based on the ones used during pre-training for Goldfish models.

Six languages were evaluated: English, Swedish, Danish, Estonian, Turkish, and Scottish Gaelic, forming 30 source→target pairs. Data for English, Swedish, Danish, Estonian and Turkish originates from the OSCAR corpus (Ortiz Su'arez et al., 2019), a multilingual CommonCrawl-derived dataset processed by the Ungoliant pipeline (Abadji et al., 2021). Data for Scottish Gaelic is $>$ 90% made up of MADLAD (Kudugunta et al., 2023) and NLLB data (Heffernan et al., 2022; Schwenk et al., 2021)[1], all crawled from the internet. Since original Goldfish data splits were unavailable, 10MB of finetuning data was randomly sampled using the same byte-premium values. Dataset sizes are shown in Table 1. This yielded 300 total source→target/method evaluation pairs.

We evaluate models using normalized log-perplexity on the FLORES dataset (NLLB Team et al., 2024), following the original setup used in Chang et al. (2024). Perplexity measures prediction confidence, with lower values indicating better performance. While Chang et al. (2024) also evaluated on BeleBele (Bandarkar et al., 2024), a multiple-choice benchmark dataset for language understanding, preliminary testing showed near-random performance with minimal variance across models due to their small size. Despite limitations in capturing language understanding (Meister and Cotterell, 2021), perplexity remains the standard baseline metric for language models (Takahashi and Tanaka-Ishii, 2019) and has been shown to predict downstream performance when used for dataset pruning tasks (Ankner et al., 2024).

In line with Chang et al. (2024), we measure log-perplexity for a model $\mathcal{M}$ using the token probabilities $\mathcal{P}$ on the second half s_1 of every sequence while conditioning on the first half s_0. This controls for the fact that multilingual models rely on early tokens to identify the language before generating predictions. To avoid penalizing tokenizers that use many tokens to represent the same text, we report byte-normalized log-perplexity. For each sequence, we compute its log-perplexity (i.e., negative log-likelihood in log space) and normalize by

[1]See Chang et al. (2024) for full data rundown

Method	TGT NormPPL	TGT MultiBlimp	Bytes/Token
100 MB-src (*No FT*)	4.327 (std=0.764)	0.711 (std=0.147)	2.44
10 MB-tgt (*No FT*)	1.771 (std=0.103)	0.793 (std=0.125)	4.57
100 MB-src (*Mono FT*)	2.391 (std=0.316)	0.807 (std=0.144)	2.44
100 MB-src (*Mixed FT*)	2.487 (std=0.320)	0.804 (std=0.147)	2.44
FVT (*No FT*)	2.926 (std=0.214)	0.662 (std=0.140)	4.57
OMP (*No FT*)	2.668 (std=0.213)	0.652 (std=0.150)	4.57
FVT (*Mono FT*)	1.693 (std=0.096)	0.852 (std=0.131)	4.57
OMP (*Mono FT*)	**1.684 (std=0.091)**	**0.855 (std=0.140)**	4.57
FVT (*Mixed FT*)	1.805 (std=0.101)	0.815 (std=0.154)	4.57
OMP (*Mixed FT*)	1.778 (std=0.096)	0.799 (std=0.153)	4.57

Table 2: Median and standard deviation of mean normalized target perplexity (smaller is better) and MultiBlimp (larger is better) scores across all language pairs (n=30)/method, alongside average bytes per token for each method.

its UTF-8 byte count. We then take the mean over all normalized sequences:

$$\text{NormPPL}_{\mathcal{M}} = \text{mean}_s\left(\frac{-log(\mathcal{P}_{\mathcal{M}}(s_1|s_0)}{\text{Bytes}_{\text{UTF-8}}(s_1)}\right) \quad (1)$$

In the rest of the paper, we use NormPPL to refer to this byte-normalized log-perplexity.

Additionally, we evaluate our models using the MultiBlimp benchmark (Jumelet et al., 2025), which leverages Universal Dependencies (Nivre et al., 2016) and UniMorph (Batsuren et al., 2022) to create a multilingual benchmark of linguistic minimal pairs for two types of subject-verb agreement. Each pair has one correct interpretation S^+ and one incorrect interpretation S^-. In English, as an example, we could have s^+ = *"These wolf packs have flourished"* and s^- = *"These wolf packs 've flourished."* For each model $\mathcal{M}$ we can calculate MultiBlimp accuracy for a dataset $\mathcal{D}$ by going through each pair S, first calculating the model probabilities $\mathcal{P}_{\mathcal{M}}$ for S^+ and S^-, taking the largest probability as the model's guess, and then getting the mean of correct lines:

$$Acc(\mathcal{M};\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{s\in\mathcal{D}} \mathbb{1}\left[\mathcal{P}_{\mathcal{M}}(S^+) > \mathcal{P}_{\mathcal{M}}(S^-)\right] \quad (2)$$

All experiments were conducted on a single Titan X GPU. Tokenizer transfer required approximately 30 minutes total for all pairs, with OMP and FVT taking about a minute each for one language pair. Finetuning for 750 steps required approximately 15 minutes per model.

4 Results

OMP with monolingual target finetuning achieved the lowest median target NormPPL (1.684) and highest median target MultiBlimp accuracy (0.855), outperforming both the 100MB source baseline (TGT PPL 4.327, TGT MultiBlimp 0.711) and the 10MB target baseline (TGT PPL 1.771, TGT MultiBlimp 0.793). FVT with target finetuning performed comparably at TGT PPL 1.693, TGT MultiBlimp 0.852. Without finetuning, OMP achieved median target NormPPL within 0.4 of the mono-finetuned 100MB source model while requiring substantially less computation and also providing improved tokenization efficiency (4.57 vs 2.44 bytes/token). However, OMP without finetuning degraded median MultiBlimp accuracy 0.06 below the 100MB source baseline. Full target NormPPL results are in Table 2.

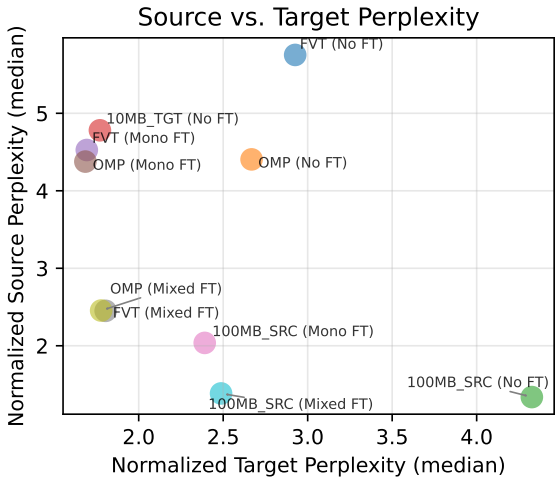

Figure 1: Normalized source vs. target NormPPL for each method. Each point represents the median performance across all language pairs for source/target respectively.

4.1 Effect on the source language

Examining source and target NormPPL jointly reveals a trade-off: tokenizer transfer methods

Method	SRC+TGT NormPPL	SRC+TGT MultiBlimp
100 MB-src (*No FT*)	5.553 (std=0.765)	1.674 (std=0.155)
10 MB-tgt (*No FT*)	6.513 (std=0.596)	1.610 (std=0.190)
100 MB-src (*Mono FT*)	4.457 (std=0.351)	1.671 (std=0.169)
100 MB-src (*Mixed FT*)	**3.812 (std=0.330)**	**1.734 (std=0.148)**
FVT (*No FT*)	8.465 (std=0.908)	1.497 (std=0.188)
OMP (*No FT*)	7.051 (std=0.644)	1.507 (std=0.208)
FVT (*Mono FT*)	6.194 (std=0.674)	1.633 (std=0.185)
OMP (*Mono FT*)	6.064 (std=0.585)	1.611 (std=0.196)
FVT (*Mixed FT*)	4.260 (std=0.336)	1.675 (std=0.179)
OMP (*Mixed FT*)	4.212 (std=0.331)	1.667 (std=0.181)

Table 3: Median and standard deviation of mean normalized source + target perplexity (smaller is better) and MultiBlimp (larger is better) scores across all language pairs (n=30)/method.

achieve strong target performance but degrade source-language capability more than finetuned source models. Figure 1 illustrates this relationship, showing how transfer methods reduce target NormPPL relative to the un-transferred source baseline, but at the cost of increased source NormPPL. The corresponding figure for MultiBlimp is available in Figure 2, and it differs in that the non-finetuned OMP/FVT performs *worse* than any other method, including both baselines. For the finetuned OMP/FVT methods we see similar tradeoffs between target accuracy and source, having diminished source performance. For combined source+target performance (Table 3), the 100MB source model with mixed finetuning achieves the best overall score.

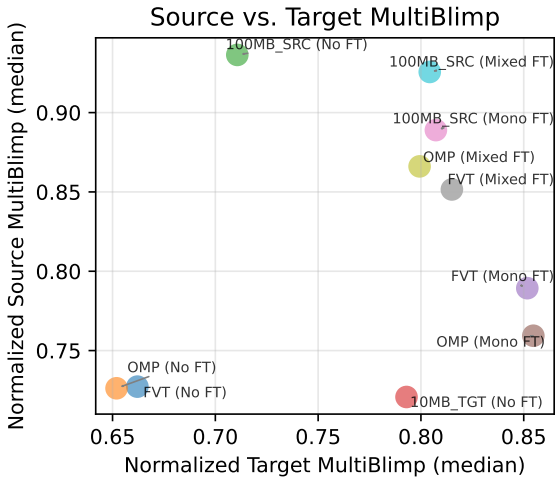

Figure 2: Normalized source vs. target MultiBlimp accuracy for each method. Each point represents the median performance across all language pairs for source/target respectively.

4.2 Effect of the Source–Target Language Pair

While aggregate results (Table 2 and 3) show consistent trends for tokenizer transfer across all evaluated settings, we also investigate whether the choice of source language has a substantial impact on target-language performance.

Table 4 summarizes these results. For each target language, it reports the baseline performance (*100mb-src*). For each transfer method, it also provides the median and standard deviation of NormPPL under the monolingual finetuning variant (*OMP/FVT (Mono FT)*) across all sources, along with the best- and worst-performing source languages. Figure 3 complements this summary by showing the full per-source breakdown across conditions for each target.

Overall, the effect of the source language is limited once adaptation is applied. Whereas the baseline models show substantial differences (median NormPPL ranges from 2.82 for English up to 4.80 for Turkish), after tokenizer transfer followed by finetuning on the target language, the spread across sources becomes small. In particular, for Estonian, Turkish, and Scottish Gaelic the variation across sources is negligible (Std ≤ 0.01 for both FVT and OMP), indicating that performance is largely source-invariant in these targets. English also shows only minor variation (Std $= 0.03$ under FVT and 0.01 under OMP). The largest source sensitivity is observed for Swedish and Danish (Std ≈ 0.04–0.06), although even here the differences are modest in absolute terms. Finally, OMP is consistently as good as or slightly better than FVT in median PPL across all targets in Table 4.

5 Discussion

This study addresses (1) how tokenizer transfer methods compare to traditional finetuning for

	Baseline	FVT			OMP		
Target	Median	Median (Std)	Best	Worst	Median (Std)	Best	Worst
ENG	2.822	1.676 (0.026)	GLA (1.647)	EST (1.712)	1.676 (0.015)	GLA (1.660)	TUR (1.701)
SWE	4.199	1.780 (0.046)	DAN (1.669)	TUR (1.792)	1.753 (0.038)	DAN (1.668)	GLA (1.772)
DAN	4.037	1.777 (0.056)	SWE (1.636)	TUR (1.785)	1.747 (0.042)	SWE (1.647)	GLA (1.758)
EST	4.505	1.879 (0.003)	SWE(1.873)	GLA (1.883)	1.853 (0.005)	SWE (1.846)	GLA (1.859)
TUR	4.796	1.589 (0.003)	SWE (1.583)	DAN (1.591)	1.572 (0.007)	EST (1.560)	ENG (1.580)
GLA	4.410	1.672 (0.005)	ENG (1.660)	EST (1.674)	1.649 (0.003)	ENG (1.642)	TUR (1.650)

Table 4: Cross-lingual median **NormPPL** (lower is better) for each target language under FVT and OMP. Baseline reports the performance of the 100MB-*src* model on the target language without tokenizer transfer. For each method, we take each source language's target NormPPL in the *OMP/FVT (Mono FT)* setting and report the median NormPPL (standard deviation) **across source languages**, along with the source language achieving the **best** and **worst** scores.

target-language adaptation, and (2) what trade-offs exist between target and source-language performance. Importantly, we focus only on *monolingually pretrained* source models, unlike the majority of prior work, which studies transfer within the same language or from multilingual LMs (Goddard and Neto, 2025; Minixhofer et al., 2024, 2025). These models are not trained to share representations with the target language we want to adapt to, making cross-language adaptation even more difficult.

For (1), OMP with monolingual target finetuning achieved the strongest target-language performance (1.684 NormPPL, 0.855 MultiBlimp accuracy), outperforming both the 10MB target baseline and the 100MB source model finetuned on the target language (Mono FT). This aligns with the findings of Goddard and Neto (2025), that sparse anchor-based reconstruction preserves semantic structure. Without finetuning, OMP and FVT performed similar to that of a mono-finetuned source model with respect to NormPPL, suggesting the transfer process itself may provide adaptation, though both approaches suffered in regards to MultiBlimp accuracy compared to baselines.

For source-language performance preservation, results highlighted a trade-off: OMP and FVT optimize target performance at the expense of source-language capability. The 100MB source model with mixed finetuning achieved the best combined source+target NormPPL, suggesting that for bilingual use cases, traditional finetuning remains superior. This was especially notable in the MultiBlimp accuracy, where the transfer methods without finetuning degraded both source performance and failed to match the baseline target accuracy. The mixed finetuning condition partially mitigated this trade-off for FVT and OMP, but it lagged behind the 100MB source model with mixed finetuning.

For (2), we find that source-language choice is *often* of limited importance after adaptation, but not irrelevant. Under *OMP/FVT (Mono FT)*, the standard deviation of target NormPPL across source languages is negligible for several targets (EST: 0.003/0.005 for FVT/OMP; TUR: 0.003/0.007; GLA: 0.005/0.003) and remains small for ENG (0.026/0.015). The largest source sensitivity occurs for SWE and DAN (SWE: 0.046/0.038; DAN: 0.056/0.042), indicating that source effects persist for some targets even after finetuning.

Taken together, these results suggest that tokenizer transfer can make adaptation *less* sensitive to source-language choice and, in our setting, yields stronger target-language gains than finetuning alone, at least within our Latin-script experiments. Practically, this implies that a readily available high-resource model in the same script can often serve as a reasonable source even without selecting a linguistically similar "parent" model, provided there is sufficient script/token overlap. At the same time, SWE and DAN (the closest pair in our set) show the clearest source effects: in the *OMP/FVT (Mono FT)* setting, SWE is the best source for adapting to DAN, improving target NormPPL by ≈0.10 over the next-best source; conversely, DAN is the best source for adapting to SWE, with an ≈0.08 advantage over the second best source language (Figure 3). This pattern suggests that even when overall variance is small, selecting a particularly well-matched source can yield measurable gains for some targets.

Overall, our findings suggest a practical recipe for low-resource language model training, in that we can train a better language model for a language in a low resource 10MB scenario by: i) training a tokenizer on that 10MB; ii) transferring that tokenizer

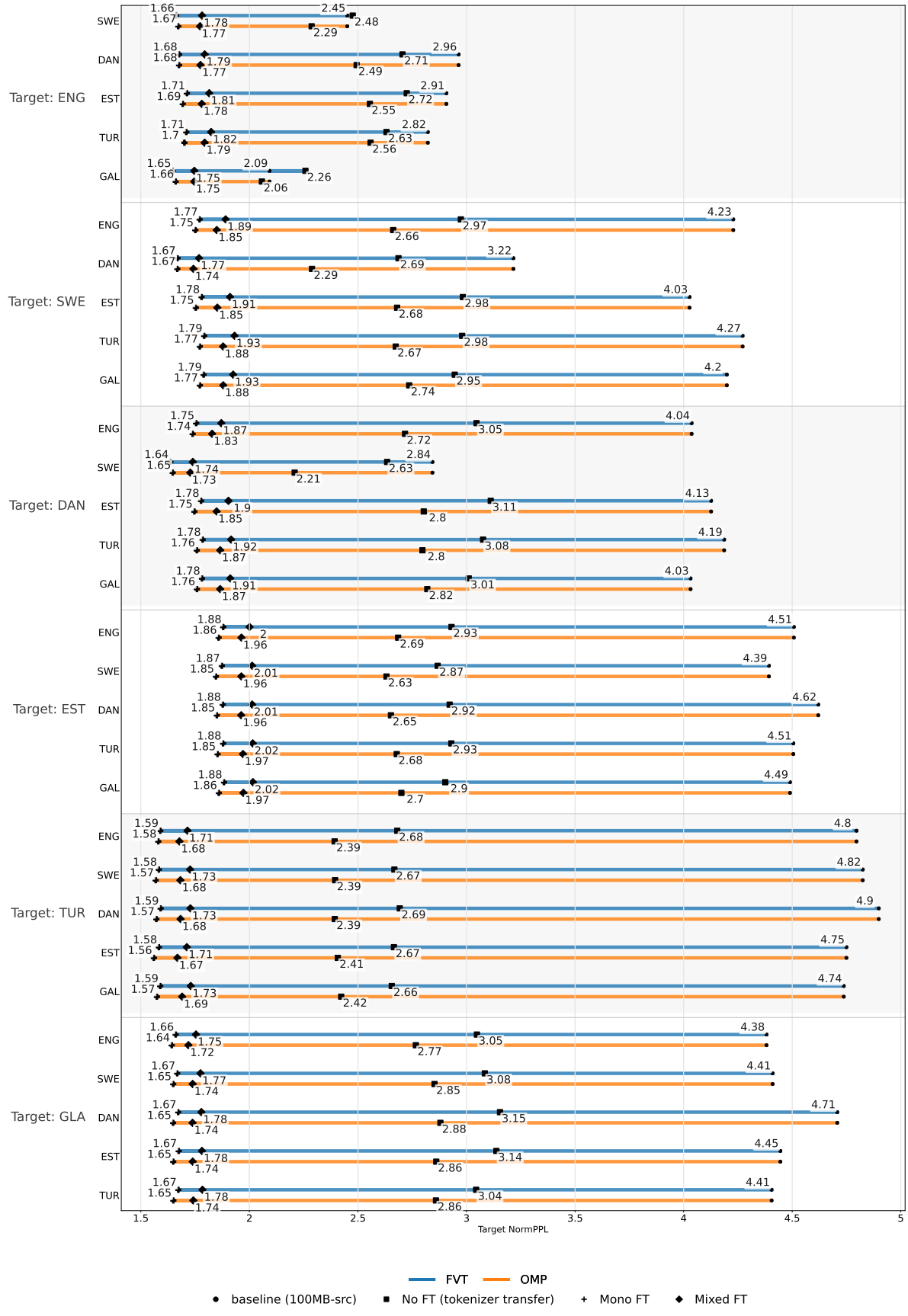

Figure 3: NormPPL (lower is better) for each target language across source languages, showing the progression from the *100MB-src* baseline through *OMP/FVT (No FT)*, *OMP/FVT (Mono FT)*, and *OMP/FVT (Mixed FT)*.

to a larger pre-trained model, and iii) fine-tune the model using our 10MB data for significantly fewer steps than would be required for a monolingual model.

Future work could explore: (i) transfer between distant language pairs using multilingually trained tokenizers, (ii) scaling behavior across model sizes to determine if source degradation diminishes with capacity, and (iii) evaluate on reasoning/natural language benchmarks using larger models.

6 Conclusion

We present an all-pair evaluation of OMP, FVT tokenizer transfer methods compared to traditional finetuning approaches for monolingually trained low-resource language models. Across six languages, we show that tokenizer transfer combined with target-language finetuning outperforms small monolingual models trained from scratch, while requiring a fraction of the compute. OMP delivers the strongest target-language results, though at the cost of degraded source-language performance, highlighting a clear adaptability/retention trade-off. Our findings suggest tokenizer transfer as a practical and compute-efficient strategy for extending pretrained language models to new low-resource languages, provided that token overlap is sufficient and bilingual performance is not the primary goal.

Limitations

This study has several limitations. First, our evaluation covers only six languages, all using the Latin script. Early experiments with Greek and Arabic were excluded due to extreme tokenizer mismatch (up to $\sim$90% unknown tokens), indicating that both FVT and OMP do not apply straightforwardly when source and target scripts are mismatched and token overlap is low. Second, we focus on relatively small models (Goldfish at 10MB/100MB), and the extent to which these findings generalize should be verified on larger models. Similarly, we are using data in effectively a simulated low-resource scenario. It could be the case that this has unforeseen interaction effects. For example, Kreutzer et al. (2022) discovered that low-resource languages have more noise than high-resource languages for datasets derived from crawled internet sources. So even though we are using the same amount of data as a low-resource scenario, the *quality* of that data might be different and not accurately reflect a true low-resource scenario. Third, most settings were evaluated with a single run due to resource constraints; future work should repeat a representative subset with multiple random seeds and report mean$\pm$std. Finally, results rely on NormPPL and MultiBlimp; future work should assess downstream generation capabilities, though this is challenging for models of this size.

Ethical Considerations

The use of Common Crawl data for the pre-training and finetuning of models risks further emphasizing harmful bias in the training data. Also, the environmental cost of model training and finetuning, while reduced through transfer methods, remains non-negligible. The carbon footprint of 300 experimental runs across multiple finetuning conditions was not measured but should be considered in future work.

Acknowledgments

We gratefully acknowledge support from the Swedish Research Council (grant no. 2022-02909). We thank Joakim Nivre for his helpful guidance during the conceptualization of this work. We also thank the anonymous reviewers for valuable feedback that helped refine and improve the paper.

References

Julien Abadji, Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2021. Ungoliant: An optimized pipeline for the generation of a very large-scale multilingual web corpus.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R Mortensen, Noah A Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9904–9923.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L Leavitt, and Mansheej Paul. 2024. Perplexed by perplexity: Perplexity-based pruning with small reference models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775,

Bangkok, Thailand. Association for Computational Linguistics.

Khuyagbaatar Batsuren, Omer Goldman, Salam Khalifa, Nizar Habash, Witold Kieraś, Gábor Bella, Brian Leonard, Garrett Nicolai, Kyle Gorman, Yustinus Ghanggo Ate, Maria Ryskina, Sabrina Mielke, Elena Budianskaya, Charbel El-Khaissi, Tiago Pimentel, Michael Gasser, William Abbott Lane, Mohit Raj, Matt Coler, and 76 others. 2022. UniMorph 4.0: Universal Morphology. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 840–855, Marseille, France. European Language Resources Association.

Damian Blasi, Antonios Anastasopoulos, and Graham Neubig. 2022. Systematic inequalities in language technology performance across the world's languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5486–5505, Dublin, Ireland. Association for Computational Linguistics.

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K. Bergen. 2024. Goldfish: Monolingual language models for 350 languages. *Preprint*, arXiv:2408.10441.

Aaron Chatterji, Thomas Cunningham, David J Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. 2025. How people use chatgpt. Working Paper 34255, National Bureau of Economic Research.

Konstantin Dobler and Gerard de Melo. 2023. Focus: Effective embedding initialization for monolingual specialization of multilingual models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, page 13440–13454. Association for Computational Linguistics.

Leonidas Gee, Andrea Zugarini, Leonardo Rigutini, and Paolo Torroni. 2022. Fast vocabulary transfer for language model compression. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 409–416.

Charles Goddard and Fernando Fernandes Neto. 2025. Training-free tokenizer transplantation via orthogonal matching pursuit. *Preprint*, arXiv:2506.06607.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages. *Preprint*, arXiv:2205.12654.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. 2025. Multiblimp 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Preprint*, arXiv:2504.02768.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, and 33 others. 2022. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Christopher A. Choquette-Choo, Katherine Lee, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, and Orhan Firat. 2023. Madlad-400: A multilingual and document-level large audited dataset. *Preprint*, arXiv:2309.04662.

Clara Meister and Ryan Cotterell. 2021. Language model evaluation beyond perplexity. *Preprint*, arXiv:2106.00085.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsaz. 2022. Wechsel: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 3992–4006. Association for Computational Linguistics.

Benjamin Minixhofer, Edoardo Maria Ponti, and Ivan Vulić. 2024. Zero-shot tokenizer transfer. *Advances in Neural Information Processing Systems*, 37:46791–46818.

Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. 2025. Universal cross-tokenizer distillation via approximate likelihood matching. *Preprint*, arXiv:2503.20083.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Pedro Javier Ortiz Su'arez, Benoit Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. *Advances in neural information processing systems*, 36:36963–36990.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S. Yu. 2025. A survey of multilingual large language models. *Patterns*, 6(1):101118.

Surangika Ranathunga and Nisansa de Silva. 2022. Some languages are more equal than others: Probing deeper into the linguistic disparity in the nlp world. *Preprint*, arXiv:2210.08523.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500, Online. Association for Computational Linguistics.

Anton Schäfer, Shauli Ravfogel, Thomas Hofmann, Tiago Pimentel, and Imanol Schlag. 2024. The role of language imbalance in cross-lingual generalisation: Insights from cloned language experiments. *Preprint*, arXiv:2404.07982.

Shuntaro Takahashi and Kumiko Tanaka-Ishii. 2019. Evaluating computational language models with scaling properties of natural language. *Computational Linguistics*, 45(3):481–513.

Veniamin Veselovsky, Berke Argin, Benedikt Stroebl, Chris Wendler, Robert West, James Evans, Thomas L. Griffiths, and Arvind Narayanan. 2025. Localized cultural knowledge is conserved and controllable in large language models. *Preprint*, arXiv:2504.10191.

Atsuki Yamaguchi, Aline Villavicencio, and Nikolaos Aletras. 2025. How can we effectively expand the vocabulary of llms with 0.01gb of target language text? *Computational Linguistics*, pages 1–40.