Model-Aware Tokenizer Transfer

Mykola Haltiuk
Faculty of Computer Science
AGH University of Krakow
Krakow, Poland
mhaltiuk@agh.edu.pl

Aleksander Smywiński-Pohl
Faculty of Computer Science
AGH University of Krakow
Krakow, Poland
apohllo@agh.edu.pl

Abstract

Large Language Models (LLMs) are trained to support an increasing number of languages, yet their predefined tokenizers remain a bottleneck for adapting models to lower-resource or distinct-script languages. Existing tokenizer transfer methods typically rely on semantic heuristics to initialize new embeddings, ignoring higher-layer model dynamics and limiting transfer quality. We propose Model-Aware Tokenizer Transfer (MATT), a method that incorporates model internals into the tokenizer transfer process. MATT introduces an Attention Influence Modeling (AIM) objective that distills inter-token communication patterns from a source model into a target model with a new tokenizer, providing an efficient warm-up before standard language modeling. Unlike approaches that focus solely on embedding similarity, MATT leverages attention behavior to guide embedding initialization and adaptation. Experiments across diverse linguistic settings show that MATT recovers a large fraction of the original model's performance within a few GPU hours, outperforming heuristic baselines. These results demonstrate that incorporating model-level signals offers a practical and effective path toward robust tokenizer transfer in multilingual LLMs.[1]

1 Introduction

Recent advances in large language models (LLMs) have shifted attention from training monolingual models [Jiang et al., 2023, Touvron et al., 2023] to covering an increasing number of languages [Grattafiori et al., 2024, Team et al., 2025]. Such multilingual models have become valuable tools for researchers and practitioners working with lower-resource languages. They can be used directly for downstream tasks, help translate English datasets into the target language [Rybak, 2023], or act as a robust baseline for further adaptation [Ociepa et al., 2024]. Our work focuses on the last scenario: adapting a multilingual LLM to a language it already supports, by giving it a tokenizer that segments that language more efficiently.

A major practical challenge in this setting is that every pretrained model is tied to a fixed tokenizer. Alternative architectures that avoid a predefined vocabulary, such as the Byte-Latent Transformer [Pagnoni et al., 2025] or H-Net [Hwang et al., 2025], are still in the experimental stage and not yet widely adopted. Tokenizers for multilingual models are usually trained to cover many scripts at once and inevitably favor high-resource languages. As a result, languages underrepresented in the tokenizer, most acutely those with distinct alphabets such as Georgian, often receive a very limited share of the vocabulary. This mismatch leads not only to lower accuracy [Ali et al., 2024, Tamang and Bora, 2024], but also to slower processing and inference, which are vital for the end users.

One practical way to mitigate this problem is tokenizer transfer: replacing the original tokenizer of a pretrained model with a new one tailored to the target language and retraining the input and output

[1] Implementation is available at https://github.com/Goader/matt.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

embeddings [de Vries and Nissim, 2020]. Even models not explicitly trained for multilinguality usually contain some cross-lingual knowledge thanks to shared alphabets or accidental language contamination [Blevins and Zettlemoyer, 2022]. Consequently, if we can initialize the new embeddings well, much of the original performance can be recovered and used as a strong starting point for continual pretraining. At the same time, we should not expect this process to introduce entirely new linguistic knowledge, since several studies show that most of the model's knowledge is stored in the feed-forward layers [Dai et al., 2022, Geva et al., 2021, Nichani et al., 2024].

Most existing tokenizer transfer methods focus almost exclusively on the embedding layer. They construct new embeddings as linear combinations of the original ones, differing mainly in how the combination weights are computed [Minixhofer et al., 2022, Dobler and de Melo, 2023, Remy et al., 2023, 2024, Li et al., 2025]. By ignoring the higher layers, these approaches overlook how the model actually processes tokens. More recent work, such as Zero-Shot Tokenizer Transfer by Minixhofer et al. [2024], leverages the full model by training a hypernetwork with a language modeling objective to predict embeddings. While effective, this strategy is computationally demanding because language modeling requires full forward and backward passes through the model.

To overcome these limitations, we introduce **Model-Aware Tokenizer Transfer (MATT)**, a method that leverages the internal behavior of the pretrained model rather than relying only on surface semantics. At the core of MATT is the **Attention Influence Modeling (AIM)** objective.

AIM encourages the model with the new tokenizer to reproduce the inter-token interactions of the original model's attention layers. In effect, the original model acts as a teacher, while the model with the new tokenizer serves as a student that learns to match its attention patterns. This procedure distills structural knowledge about token relationships directly from the teacher, providing a richer and more informative initialization than relying on an embedding layer alone.

MATT is orthogonal to existing heuristics based on semantic similarity and can be combined with them. It acts as an efficient warm-up stage before conventional language model pretraining: MATT recovers much of the original model's performance at a fraction of the cost of language-modeling-based alternatives, though a residual gap typically remains, most visibly on generative tasks, leaving continual pretraining as the standard path to narrow it further.

We evaluate MATT by transferring the tokenizers of Gemma 3 [Team et al., 2025] and Qwen 3 [Team, 2025] models to replacement and extended versions that increase compression and expand coverage for several languages, including German, Japanese, Arabic, Swahili, and Ukrainian. Across multiple settings, MATT consistently recovers a substantial portion of the original model's performance on both generative and discriminative tasks, while requiring only a few GPU hours and outperforming heuristic-based and optimization-based transfer methods.

Our contributions can be summarized as follows:

- Attention Influence Modeling (AIM): a novel distillation objective that aligns the attention dynamics of two models with different tokenizers.
- Model-Aware Tokenizer Transfer (MATT): an efficient tokenizer transfer method that exploits model dynamics instead of relying solely on semantic relationships, outperforming heuristic and optimization-based baselines at substantially lower computational cost than language modeling objectives.
- Comprehensive evaluation: experiments across multiple languages and models demonstrating the effectiveness and efficiency of MATT.

2 Related work

Large language models and vocabulary size Large Language Models are becoming increasingly multilingual. Early open-source models focused almost exclusively on English [Jiang et al., 2023, Touvron et al., 2023, Almazrouei et al., 2023], but most recent releases include at least several languages and offer partial support for many more. This shift toward multilinguality has changed how researchers choose vocabulary size.

Studies show that larger vocabularies can improve model quality [Takase et al., 2025, Liang et al., 2023], but they also slow training and inference. As a result, most current foundation models use

vocabularies of about 100 to 250 thousand tokens, with strongly multilingual models leaning toward the upper end. This sweet spot, first popularized by XLM-RoBERTa [Conneau et al., 2020], continues in more recent models such as Gemma [Team et al., 2025], Aya Expanse [Dang et al., 2024], and even GPT-5.[2] Going beyond this range rarely pays off: performance gains are small, and efficiency drops sharply. As a result, tokenizers cannot achieve an optimal compression rate for every language, creating a need for techniques that allow efficient transfer of tokenizers to specific languages or domains without requiring very large vocabularies.

Heuristics-based embedding initialization methods When transferring a tokenizer to a new language or domain, the main challenge is initializing embeddings for tokens that did not exist in the original model. Early work on tokenizer transfer [Artetxe et al., 2020, Gogoulou et al., 2022, de Vries and Nissim, 2020] focused on proving that transfer was possible, so embedding initialization received little attention. Simple strategies were used, including random initialization, taking the mean of existing embeddings, sampling from their distribution, copying the embedding of a random token, or using token frequency as a guide.

Later research began to exploit semantic relationships between tokens. WECHSEL [Minixhofer et al., 2022] was an influential step: it trained FastText [Bojanowski et al., 2017] embeddings for the source and target languages and used a translation vocabulary to identify the closest source tokens for each new token. New embeddings were then initialized as weighted averages of these source embeddings. Several methods followed a similar direction. OFA [Liu et al., 2024] and Tik-to-Tok [Remy et al., 2023] refined the idea of using cross-lingual similarities, while Transtokenizers [Remy et al., 2024] created a token-level translation dictionary with FastAlign [Dyer et al., 2013]. Hyper-OFA [Özeren et al., 2025] went further by training a hypernetwork to map tokens from an external multilingual space into the model's embedding space, avoiding the need for simplistic linear combinations. TokAlign [Li et al., 2025] took a co-occurrence perspective, training two GloVe [Pennington et al., 2014] models on the same corpus to learn a one-to-one alignment matrix between tokens.

As LLMs became more multilingual, overlap between source and target vocabularies became an important resource. FOCUS [Dobler and de Melo, 2023] trains a FastText model on text tokenized with the target vocabulary, then initializes new embeddings as similarity-weighted averages of overlapping tokens. CLP Transfer [Ostendorff and Rehm, 2023] takes advantage of topological similarities of the latent space across model sizes within the same family: embeddings are first trained on a smaller related model and then aligned to the target model by measuring similarities with overlapping tokens.

Beyond heuristics While heuristics provide a practical starting point, they have limitations. An alternative is to train new embeddings directly by continuing language modeling with all other parameters frozen [de Vries and Nissim, 2020], but this is computationally costly.

Another approach by Minixhofer et al. [2024] trains a universal hypernetwork for a given language model by sampling tokenizers from a diverse distribution during the language modeling stage. Once the hypernetwork is trained, we can initialize embeddings for various tokenizers effortlessly, achieving a solid baseline for further continual pretraining. However, training such a hypernetwork is a compute-heavy task, requiring forward and backward passes through the whole model in every step to update the hypernetwork weights, limiting its practicality in settings where we already have a defined target tokenizer and the trained hypernetwork is not available beforehand.

More recently, Token Distillation [Dobler et al., 2026] optimizes the embedding of each new token to produce, when fed through the model, the same hidden state that the original model yields for the equivalent multi-token sequence.

Positioning Existing methods can be positioned by the signal each relies on. WECHSEL and Transtokenizers were developed for cross-lingual transfer and use an external cross-lingual signal, a bilingual dictionary or parallel corpora, which lets them target a language the source model does not know. FOCUS relies on overlap between the source and target vocabularies together with target-language text; its original evaluation includes a setting in which the overlap is restricted to numbers and punctuation, where it still performs reasonably well, so it could also be used for cross-lingual

[2] https://github.com/openai/tiktoken

transfer. Continued language modeling of the full model is the universal option, as it can introduce capabilities the model lacks, but it is expensive. Token Distillation uses only the source model's own representations and is therefore bounded by what that model already knows. MATT belongs to this last group. It distills the original model's inter-token attention behavior rather than its hidden states for equivalent token sequences, but it shares the same premise and the same limitation. We therefore position MATT as tokenizer specialization for languages the model already supports, rather than as adaptation to genuinely new languages.

3 Method

3.1 Intuition

Large Language Models generate text one token at a time. Decoder-only transformers, which form the backbone of most modern LLMs, follow the following steps: the embedding of the most recently generated token is passed through a stack of attention and feed-forward layers, and finally projected by the LM head to produce a probability distribution over the next token.

Assuming the input embedding of the last token is correct, the feed-forward layers will not damage its representation. The main source of potential distortion lies in the attention layers, where each token interacts with the context. Changing the tokenizer introduces new tokens into the context, altering these interactions and thus the internal representations that drive next-token prediction. Our goal is to train a model using a new tokenizer so that, despite these changes, its attention layers produce output embeddings similar to those generated by the original tokenizer.

3.2 Prerequisites

Consider an input string s and a tokenization function T, which produces a token sequence $T(s) = (t_1, t_2, \ldots, t_n)$ of length n.

In each attention layer,[3] the inputs are the query ($\boldsymbol{Q}$), key ($\boldsymbol{K}$), and value ($\boldsymbol{V}$) state matrices, producing the output state matrix ($\boldsymbol{O}$). Each of these can be seen as a collection of vector states for every token t_i:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q}_1 \\ \boldsymbol{q}_2 \\ \cdots \\ \boldsymbol{q}_n \end{bmatrix}_{n \times h}, \quad \boldsymbol{K} = \begin{bmatrix} \boldsymbol{k}_1 \\ \boldsymbol{k}_2 \\ \cdots \\ \boldsymbol{k}_n \end{bmatrix}_{n \times h}, \quad \boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \cdots \\ \boldsymbol{v}_n \end{bmatrix}_{n \times h}, \quad \boldsymbol{O} = \begin{bmatrix} \boldsymbol{o}_1 \\ \boldsymbol{o}_2 \\ \cdots \\ \boldsymbol{o}_n \end{bmatrix}_{n \times h}, \tag{1}$$

where h is the hidden size.

Attention is computed as:

$$\boldsymbol{O} = \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} = \boldsymbol{A}\boldsymbol{V}, \tag{2}$$

where $\boldsymbol{A}$ is the attention matrix of shape $n \times n$, that contains weights with which the value states are aggregated into the output state.

We can break down the final matrix multiplication $\boldsymbol{A}\boldsymbol{V}$ into a chain of value states ($\boldsymbol{V}$) averages for each token, weighted by the attention matrix $\boldsymbol{A}$. The output state for the token t_i would then look the following way:

$$\boldsymbol{o}_i = \text{softmax}\left(\frac{\boldsymbol{q}_i\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} = \boldsymbol{A}_{i,:}\boldsymbol{V} = \sum_{j=1}^{n} \boldsymbol{A}_{i,j}\boldsymbol{v}_j = \sum_{j=1}^{n} \boldsymbol{v}^*_{i,j}, \tag{3}$$

where $\boldsymbol{v}^*_{i,j} = \boldsymbol{A}_{i,j}\boldsymbol{v}_j$ is a weighted value state for the token t_j given the query token t_i.

[3]While we proceed with a single-head definition, it is directly applicable to multi-head, multi-query, or grouped-query attention variants.

3.3 Segment-level interpretation of attention

To compare attention outputs across different tokenizers, we introduce a segmentation function S that splits the input string s into segments $(s_1, s_2, \ldots, s_m)$ while respecting a set of tokenization functions $\mathcal{T}$:

$$\begin{aligned} S(s;\, \mathcal{T}) &= (s_1, s_2, \ldots, s_m), \text{ such that} \\ \forall\, T \in \mathcal{T}:\; T(s_1) \circ T(s_2) \circ \cdots \circ T(s_m) &= T(s), \end{aligned} \tag{4}$$

where $\circ$ is a concatenation operator. This ensures that no segment boundary lies within any token produced by any tokenization function in $\mathcal{T}$.

The most intuitive approach is a function that splits the input string into words, and the rest of the section is explained in relation to this function. However, for some languages, word segmentation can be ambiguous; thus, in practice, we define our segmentation function to always choose segments of minimal length that still satisfy the above condition (see Appendix B for the algorithm).

Given S, we define weighted value states for a segment s_k with respect to a query token t_i:

$$\mathfrak{s}_{i,k} = \sum_{\{j\,:\,t_j \in T(s_k)\}} \boldsymbol{v}^*_{i,j}. \tag{5}$$

The output state for token t_i can then be expressed as a sum over segments:

$$\boldsymbol{o}_i = \sum_{j=1}^{n} \boldsymbol{v}^*_{i,j} = \sum_{j=1}^{m} \mathfrak{s}_{i,j}. \tag{6}$$

To move from token-level to segment-level interpretation, we replace individual query tokens with segment representations. Since the output state of each token is designed to predict the next token, it is natural to require that the output state of a segment should similarly carry enough information to predict the next segment. Because the language modeling head still operates at the token level, we approximate "predicting the next segment" by predicting the first token of that next segment.

Consider a segment s_i whose tokens are $T(s_i) = (t_a, t_{a+1}, \ldots, t_b)$. The final token t_b produces the output state used to generate the next token t_{b+1}, which begins the following segment s_{i+1}. We therefore define a function ℓ_T that maps a segment index to the index of its last token:

$$\ell_T(i) = b. \tag{7}$$

The query state of segment s_i is set equal to the query state of its last token:

$$\mathfrak{q}_i = \boldsymbol{q}_{\ell_T(i)} = \boldsymbol{q}_b, \tag{8}$$

and the output state of the segment is computed from this query state:

$$\mathfrak{o}_i = \text{softmax}\left(\frac{\mathfrak{q}_i \boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} = \text{softmax}\left(\frac{\boldsymbol{q}_{\ell_T(i)} \boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} = \boldsymbol{o}_{\ell_T(i)} = \sum_{j=1}^{m} \mathfrak{s}_{\ell_T(i),j}. \tag{9}$$

3.4 Attention Influence Modeling

As described in Section 3.1, our goal is to train the model with a new tokenizer T' so that its output states match those of the original model with tokenizer T.

Since we can enforce a common segmentation function S, we approximate this by requiring the new model to produce the same segment-level outputs $\mathfrak{o}'_i$ as the old ones – $\mathfrak{o}_i$. A more detailed objective also matches the weighted value states $\mathfrak{s}_{\ell_T(i),j}$ and $\mathfrak{s}'_{\ell_{T'}(i),j}$ of every segment s_j for each query segment s_i, with the causal constraint $j \leq i$.

Given the above, we define the **Attention Influence Modeling** objectives (normal and simplified):

$$\mathcal{L}_{AIM} = \frac{2}{m(m+1)} \sum_{i=1}^{m} \sum_{j=1}^{i} \mathcal{L}^*(\mathfrak{s}_{\ell_T(i),j}, \mathfrak{s}'_{\ell_{T'}(i),j}), \tag{10}$$

$$\mathcal{L}_{AIM^*} = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}^*(\mathfrak{o}_i, \mathfrak{o}'_i), \tag{11}$$

where $\mathcal{L}^*(\boldsymbol{x}, \boldsymbol{y})$ can be any loss function, that brings $\boldsymbol{x}$ and $\boldsymbol{y}$ closer. In Section 4, we use the MSE loss.

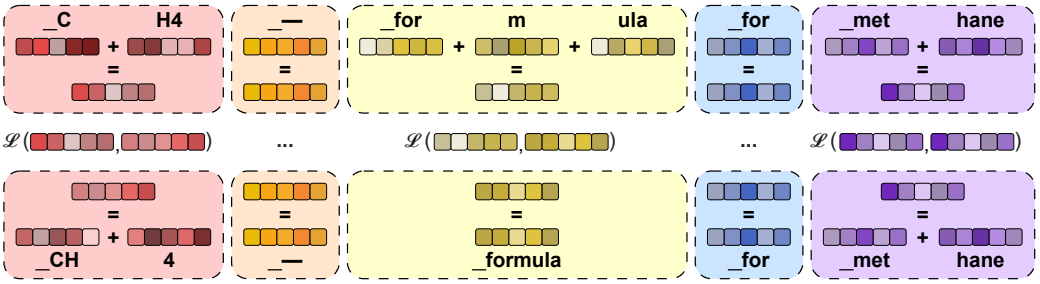


Figure 1: Attention Influence Modeling (AIM) objective with word segmentation. For each input, the weighted value vectors $\boldsymbol{v}^*_{i,j}$ of the original tokens t_j are aggregated into segment-level vectors $\mathfrak{s}_{i,k}$ according to a chosen word-segmentation function. The model trained with the new tokenizer produces its own segment representations $\mathfrak{s}'_{i,k}$. The AIM objective encourages these new segment representations to stay close to the segment representations $\mathfrak{s}_{i,k}$ computed from the model using the old tokenizer. All this happens with respect to the query state $\mathfrak{q}_5$ of the 5th segment (`_methane`), which is equal to the query state of its last token – `hane`.

Figure 1 illustrates an example of applying AIM to the text *CH4 – formula for methane*. In this case, we use a word segmentation function together with different tokenization functions for a given query state $\mathfrak{q}_5$, where the segment s_5 corresponds to `_methane`.

Appendix K complements Figure 1 with a sequence-level view of the same alignment, plotting the weighted value states $\boldsymbol{v}^*_{i,j}$ across the full causal attention matrix with segments highlighted; these segment-level representations are matched and optimized to be equal under the $\mathcal{L}_{AIM}$ loss.

3.5 Technical details

During training, the model with the old tokenizer T is kept frozen. The model with the new tokenizer T' has all layers frozen except the input embeddings. As a small modification to the basic training setup, we partially freeze the embedding matrix: tokens that are shared between the old and new tokenizers are initialized from the original model and kept fixed, while only the embeddings of new, non-overlapping tokens are updated during training (see Appendix E).

To speed up convergence, we initialize new embeddings using FOCUS [Dobler and de Melo, 2023]. We train with AdamW [Loshchilov and Hutter, 2017], a constant learning rate of 1×10^{-4}, and no weight decay. However, it should be noted that we have not performed extensive hyperparameter tuning, so using learning rate scheduling, adapting the learning rate, weight decay, and other hyperparameters may yield significantly better results.

MATT offers a key advantage over standard language modeling with frozen non-embedding parameters: greater efficiency. Since AIM is defined at the attention-layer level, we can decide how much of the model to include in the tokenizer transfer by selecting the layer depth at which AIM is applied. Specifically, by choosing a value of n, we take only the first n layers into account. This allows us to balance efficiency and performance.

We ablate the choice of MATT target layer in Appendix E and find a consistent pattern: performance climbs through the early layers, plateaus across the middle of the model, and slightly degrades on the final layers – mirroring the trend reported for Token Distillation [Dobler et al., 2026]. The plateau aligns with the formation of coherent word-level representations as tokens are detokenized in early and middle layers [Kaplan et al., 2024]. Because training time and VRAM grow linearly with target depth, we recommend a layer near the start of this plateau, empirically around the first quarter to first third of the model (e.g. layer 12 of 48 for Gemma 3 12B PT as in Section 4.1).

We supervise attention for two reasons. First, the attention layers are where we expect a tokenizer change to distort the model (see Section 3.1). Second, AIM constrains how every query segment

attends to each preceding segment, a number of constraints quadratic in the number of segments, whereas a hidden-state objective, as in Token Distillation, constrains at most one state per position. We therefore expect AIM to provide a richer training signal per document. While MATT outperforms Token Distillation on average in Section 4, the two methods also differ in target depth and in whether tokens are optimized separately or jointly, so these results do not isolate this design choice.

Since only input embeddings are trained with AIM, tied input–output embeddings are advantageous, as the tuned input embeddings can be reused in the LM head. For models without tied embeddings, we combine AIM with an NTP loss to ensure proper optimization of the output embeddings, at a slight cost in training efficiency; this combined variant is used *only* in the untied-embeddings setting (Appendix F). Throughout the rest of the paper, "MATT" refers to the AIM objective alone.

4 Experiments

We conducted a series of experiments across different languages, model families, and scales to evaluate the effectiveness of the MATT method compared to existing heuristic- and optimization-based approaches. In each experiment, we first trained a tokenizer with a higher compression rate than the original one, replaced or extended the base tokenizer with it (see Appendix C), and then applied tokenizer transfer to the result. We chose Ukrainian because its Cyrillic script is not well represented in the vocabularies of major LLMs, while the language is included in multiple benchmarks, allowing us to inspect the method's performance in different scenarios. For the multilingual setting we adapt to four typologically diverse target languages – Arabic, German, Japanese, and Swahili, which vary in resource availability, writing system, and language family.

Additional experiments, including convergence speed tests (Appendix D) and ablation studies (Appendix E), are presented to complement the main results.

4.1 Main results

Our primary evaluation uses the Gemma 3 12B PT model [Team et al., 2025]. We replaced its default tokenizer with a fixed-size replacement tokenizer that reassigns over 80,000 vocabulary entries to improve Ukrainian coverage, raising the compression rate from 2.98 to 4.44. This increase translates to an almost 50% speedup during inference. We compare the following methods:

- **WECHSEL** – transfer using the English–Ukrainian vocabulary from the official implementation.[4]
- **Transtokenizers** – token alignment via FastAlign using parallel corpora (OpenSubtitles [Lison and Tiedemann, 2016] and NLLB [NLLB-Team et al., 2022]) and the official `transtokenizers`[5] toolkit.
- **TokAlign** – GloVe embeddings trained on 2 million Ukrainian documents (approximately 1.86 billion Gemma tokens) from the Kobza corpus [Haltiuk and Smywiński-Pohl, 2025], used to create a one-to-one alignment matrix with the official implementation.[6]
- **FOCUS** – FastText embeddings trained on the same data as TokAlign, with initialization performed via the `deepfocus`[7] package.
- **NTP** – initialized with one of the above methods, and trained using the Next Token Prediction (NTP) objective with non-embedding layers frozen. We compare several versions of this baseline corresponding to 50%, 100%, and 150% of the training budget dedicated to MATT.
- **Token Distillation** – trained using the `token-distillation`[8] package. The number of mined examples per token was set to match or exceed MATT's compute budget.
- **MATT** – initialized with FOCUS embeddings and trained on around 240 million Ukrainian tokens from Kobza using the AIM objective with MSE loss on the 12th layer out of 48, with original embeddings frozen and all other hyperparameters unchanged (see Section 3.5).

[4] https://github.com/CPJKU/wechsel
[5] https://github.com/LAGoM-NLP/transtokenizer
[6] https://github.com/ZNLP/TokAlign
[7] https://github.com/konstantinjdobler/focus
[8] https://github.com/konstantinjdobler/token-distillation

All evaluations in this section are restricted to Ukrainian. Specifically, we use the Ukrainian subset of Belebele [Bandarkar et al., 2024], Global MMLU [Singh et al., 2025], and XL-Sum [Hasan et al., 2021]; for translation we use both the en→uk and uk→en directions of Long FLORES [Paniv, 2025], a document-level extension of FLORES [NLLB-Team et al., 2022, Goyal et al., 2021, Guzmán et al., 2019], and the en→uk direction of WMT24++ [Deutsch et al., 2025]. Including both Long FLORES directions probes generative performance into and out of the target language. Evaluation is performed with the `lm-evaluation-harness` framework [Gao et al., 2024] with a 3-shot prompt; the full protocol is given in Appendix I. Long FLORES and WMT are evaluated under sampling at temperature 1.0, and Table 14 in that appendix reports them under greedy decoding.

Table 1: Performance of Gemma 3 12B PT model with different tokenizer transfer methods on Belebele and Global MMLU (accuracy, %), Long FLORES, WMT, and XL-Sum (BLEU). "Avg Disc" averages Belebele and Global MMLU; "Avg Gen" averages Long FLORES (both directions), WMT, and XL-Sum.

Model	Training Time	Belebele	Global MMLU	Long FLORES en→uk	Long FLORES uk→en	WMT	XL-Sum	Avg Disc	Avg Gen
Gemma 3 12B PT	-	89.33	67.03	14.36	26.32	3.52	6.52	78.18	12.68
Heuristics									
WECHSEL	-	22.67	24.61	0.00	0.00	0.00	0.00	23.64	0.00
Transtokenizers	-	61.89	46.03	0.04	0.10	0.09	0.02	53.96	0.06
TokAlign	-	31.44	32.98	0.00	0.31	0.00	0.01	32.21	0.08
FOCUS	-	48.78	37.14	1.01	6.19	0.88	0.20	42.96	2.07
Optimization Based									
Transtokenizers w/ NTP	3h 30m	82.44	59.02	3.64	1.24	0.88	4.06	70.73	2.46
Transtokenizers w/ NTP	7h 00m	85.22	59.83	4.63	2.17	0.95	4.80	72.53	3.14
Transtokenizers w/ NTP	10h 30m	85.67	59.38	5.13	2.07	0.96	4.80	72.53	3.24
FOCUS w/ NTP	3h 30m	85.44	57.38	3.51	20.26	2.13	4.32	71.41	7.56
FOCUS w/ NTP	7h 00m	87.00	60.55	4.32	23.00	2.51	5.04	73.78	8.72
FOCUS w/ NTP	10h 30m	87.44	60.57	4.34	21.73	2.60	5.16	74.01	8.46
Token Distillation	5h 30m	82.00	58.10	0.00	21.54	0.41	3.71	70.05	6.42
Token Distillation	8h 20m	82.89	59.23	0.00	21.92	0.72	4.06	71.06	6.68
MATT	7h 00m	**89.56**	**64.98**	**8.70**	**27.89**	**4.71**	**5.95**	**77.27**	**11.81**

Table 1 shows a clear advantage of MATT over all other methods. While heuristic-based approaches such as FOCUS and Transtokenizers can regain up to about 70% of the original model's accuracy on discriminative tasks, they reach no more than about 16% of the original generative performance.

Optimization-based approaches show substantially stronger recovery. NTP's discriminative results are similar across heuristic initializations, while its generative results depend strongly on them (Avg Gen up to 3.24 with Transtokenizers and 8.72 with FOCUS). In its best configuration, it recovers nearly 95% of the original model's discriminative performance and around 70% of its generative capabilities. Token Distillation comes close to NTP on the discriminative side (around 91%) but its generative behavior is sharply asymmetric: it largely preserves uk→en translation (21.92 BLEU vs 26.32), yet collapses to near-zero on en→uk and on WMT (en→uk). MATT, by contrast, achieves over 90% of the original generative performance with balanced quality across both translation directions, while keeping discriminative accuracy close to the unmodified model. These results demonstrate the advantage of the model-aware approach to tokenizer transfer, particularly given the modest compute budget required. Under greedy decoding (Table 14), these generative gaps shrink considerably and Token Distillation no longer collapses on en→uk (28.11 BLEU out of 31.47), while MATT remains the best transfer method on all three translation tasks.

All optimization-based methods saturate quickly. NTP shows minimal improvement between 100% and 150% of MATT's compute budget (7h vs 10.5h), and MATT-level performance does not appear attainable within a reasonable budget. MATT also saturates (Appendix D), but at a substantially higher level; further gains likely require unfreezing the model's layers, since the over 80,000 reassigned vocabulary entries meaningfully alter model dynamics. We evaluate continual pretraining on one model and language in Appendix G; a systematic comparison across models and languages is left for future work.

4.2 Multilingual results

In the multilingual setting, we apply MATT independently per target language to two open mid-scale models, Gemma 3 4B PT [Team et al., 2025] and Qwen 3 4B [Team, 2025], on four typologically diverse target languages: Arabic, German, Japanese, and Swahili. For each (model, target language) pair we construct a per-language extended tokenizer that improves compression for the target language while preserving the original tokenizer's coverage of other languages, and we evaluate MATT, FOCUS, and Token Distillation against the unmodified Baseline. Per-language vocabulary sizes and compression rates are reported in Appendix H (Table 11); the extended tokenizers improve compression on the target language by $1.2\times$ to $1.6\times$ over the base. Training data is drawn from HPLT 2.0 Cleaned [Burchell et al., 2025], with approximately 250 million tokens per language. Roughly following the rule of thumb in Section 3.5, AIM is applied at the 12th layer for Gemma 3 4B PT (out of 34) and the 16th layer for Qwen 3 4B (out of 36); all other hyperparameters match Section 4.1. Token Distillation training was configured to match or exceed MATT's per-language compute budget on the same H100 (see the Time column of Table 12), so MATT's lead over Token Distillation cannot be attributed to having received more training time. Performance is reported on the target-language subsets of Belebele, Global MMLU, MMMLU [Hendrycks et al., 2020], and XL-Sum, and on the en→x and x→en directions of Long FLORES, following the protocol in Appendix I.

Table 2: Multilingual results, averaged across four target languages (Arabic, German, Japanese, Swahili). Each task column is the mean of the per-language values from Appendix H; Belebele, Global MMLU, and MMMLU report accuracy (%); Long FLORES (both directions) and XL-Sum report BLEU. "Avg Disc" averages the three discriminative metrics; "Avg Gen" the three generative ones. XL-Sum has no German subset, so its mean is over Arabic, Japanese, and Swahili.

Method	Belebele	Global MMLU	MMMLU	Long FLORES		XL-Sum	Avg	Avg
				en→x	x→en		Disc	Gen
Gemma 3 4B PT								
Baseline	70.53	49.12	49.26	7.93	20.22	4.14	56.30	10.76
FOCUS	43.33	37.30	36.63	1.93	8.39	0.18	39.09	3.50
Token Distillation	57.39	43.04	43.52	1.31	12.87	1.09	47.98	5.09
MATT	**66.03**	**47.20**	**46.84**	**3.94**	**13.22**	**1.27**	**53.36**	**6.14**
Qwen 3 4B								
Baseline	73.47	54.29	53.80	10.02	28.29	1.93	60.52	13.41
FOCUS	41.94	34.48	34.73	4.77	9.49	0.23	37.05	4.83
Token Distillation	57.22	42.86	42.68	5.11	17.74	0.04	47.59	7.63
MATT	**66.39**	**46.26**	**47.11**	**5.20**	**20.68**	**0.90**	**53.25**	**8.93**

Table 2 reports the means across the four target languages per model family; per-language detail is in Appendix H (Table 12). At the aggregate level, MATT outperforms FOCUS and Token Distillation on every metric in both family blocks, including both translation directions of Long FLORES and XL-Sum BLEU. Token Distillation slots between FOCUS and MATT, recovering most of FOCUS's discriminative and generative gap to MATT, but does not match MATT's overall recovery in either family.

Looking at the per-language detail, MATT wins Avg Disc in 7 of the 8 (family $\times$ language) sub-blocks; the only exception is Qwen 3 4B on Japanese, where Token Distillation is ahead (55.44 vs 53.12). MATT wins Avg Gen in 6 of the 8 sub-blocks; Token Distillation overtakes MATT on Avg Gen for Gemma 3 4B PT on Japanese and for Qwen 3 4B on German, in both cases primarily through stronger Long FLORES translation. MATT's Avg Disc lead over Token Distillation is largest on Arabic and Swahili and narrows on German and Japanese, the two languages on which both base models score highest. A striking failure case is Qwen 3 4B on Swahili: Token Distillation degrades *below* the FOCUS heuristic on Belebele (25.89 vs 33.78), suggesting that hidden-state matching may be brittle when the teacher's representations for the target language are themselves weak.

A general observation about MATT in this setting is that it recovers what the teacher already knows, but cannot induce capabilities the teacher lacks; as a self-distillation method, its performance is effectively upper-bounded by the original model. This is most visible on Qwen 3 4B for Swahili, where the base model itself performs far worse than on the other languages (Belebele 44.89, Global MMLU 36.41). MATT improves Belebele from FOCUS's 33.78 to 41.67 and Global MMLU from

30.99 to 33.69, narrowing the gap to the teacher but unable to surpass it. The same upper-bound argument applies to languages outside the teacher's pretraining mix entirely; cross-lingual transfer to such languages is beyond the scope of MATT.

5 Conclusion

In this work, we introduced MATT, a model-aware method for tokenizer transfer that leverages the internal dynamics of LLMs. We applied MATT to adapt the tokenizers of Gemma 3 and Qwen 3 models across multiple languages and settings, demonstrating that it consistently recovers a large portion of the original model's capabilities while requiring only a few GPU hours of training. Unlike heuristic-based methods that rely solely on the embedding layer, MATT refines token representations with direct feedback from the model, thanks to the novel Attention Influence Modeling (AIM) objective, allowing it to bridge the performance gap caused by tokenizer changes more effectively.

Our experiments highlight this advantage most clearly in the transfer of the 12 billion-parameter Gemma 3 model to Ukrainian. With the replacement tokenizer reassigning over 80,000 vocabulary entries, MATT achieves an average score of 77.27 out of the original 78.18 on the discriminative tasks and 11.81 out of 12.68 on the generative ones, outperforming both heuristic and optimization-based baselines. This substantial improvement underscores the value of incorporating model dynamics into tokenizer transfer and shows that high performance can be retained at a compute budget where NTP falls short, even with 50% more training time.

Acknowledgments and Disclosure of Funding

The research presented in this paper was financed from the funds assigned by Polish Ministry of Science and Higher Education to AGH University of Krakow.

We would like to express our gratitude to the startup Comand.AI for compute support, without which MATT would not have been created.

We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2025/018285.

We acknowledge the EuroHPC Joint Undertaking for awarding us access to MareNostrum 5 hosted by the Barcelona Supercomputing Center, Spain (project EHPC-DEV-2026D05-089).

We would also like to thank the Area Chair and reviewers for helpful discussions.

The authors declare no competing interests.

References

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, Charvi Jain, Alexander Weber, Lena Jurkschat, Hammam Abdelwahab, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Samuel Weinbach, Rafet Sifa, Stefan Kesselheim, and Nicolas Flores-Herr. Tokenizer choice for LLM training: Negligible or crucial? In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.247. URL `https://aclanthology.org/2024.findings-naacl.247/`.

Loubna Ben Allal, Lewis Tunstall, Nouamane Tazi, Elie Bakouch, Ed Beeching, Carlos Miguel Patiño, Clémentine Fourrier, Thibaud Frere, Anton Lozhkov, Colin Raffel, Leandro von Werra, and Thomas Wolf. The smol training playbook: The secrets to building world-class llms, 2025.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023.

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/.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.44. URL https://aclanthology.org/2024.acl-long.44/.

Francesco Bertolotti and Walter Cazzola. By tying embeddings you are assuming the distributional hypothesis. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=yyYMAprcAR.

Terra Blevins and Luke Zettlemoyer. Language contamination helps explains the cross-lingual capabilities of English pretrained models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3563–3574, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.233. URL https://aclanthology.org/2022.emnlp-main.233/.

Bohdan Didenko. Post #9194 on Telegram Channel Zaduha. https://t.me/zaduha/9194, June 2025. [Online; accessed 26 June 2025].

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.

Laurie Burchell, Ona de Gibert, Nikolay Arefyev, Mikko Aulamo, Marta Bañón, Mariia Fedorova, Liane Guillou, Barry Haddow, Jan Hajič, Erik Henriksson, et al. An expanded massive multilingual dataset for high-performance language technologies. *arXiv preprint arXiv:2503.10267*, 2025.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 794–803. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/chen18a.html.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581. URL https://aclanthology.org/2022.acl-long.581/.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, et al. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *arXiv preprint arXiv:2412.04261*, 2024.

Wietse de Vries and Malvina Nissim. As good as new. how to successfully recycle english gpt-2 to make models for other languages. *arXiv preprint arXiv:2012.05628*, 2020.

Daniel Deutsch, Eleftheria Briakou, Isaac Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, Shruti Rijhwani, Parker Riley, Elizabeth Salesky, Firas Trabelsi, Stephanie Winkler, Biao Zhang, and Markus Freitag. WMT24++: Expanding the Language Coverage of WMT24 to 55 Languages & Dialects, 2025. URL https://arxiv.org/abs/2502.12404.

Bohdan Didenko. Writing-system-level tokenizer adaptation for byte-level bpe. *arXiv preprint arXiv:2608.00582*, 2026.

Konstantin Dobler and Gerard de Melo. FOCUS: Effective embedding initialization for monolingual specialization of multilingual models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13440–13454, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.829. URL https://aclanthology.org/2023.emnlp-main.829/.

Konstantin Dobler, Desmond Elliott, and Gerard de Melo. Token distillation: Attention-aware input embeddings for new tokens. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=n20ml5nGEo.

Chris Dyer, Victor Chahuneau, and Noah A Smith. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 644–648, 2013.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories, 2021. URL https://arxiv.org/abs/2012.14913.

Evangelia Gogoulou, Ariel Ekgren, Tim Isbister, and Magnus Sahlgren. Cross-lingual transfer of monolingual models. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 948–955, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.100/.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dirk Groeneveld, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 15789–15809, 2024.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. Two new evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. 2019.

Mykola Haltiuk and Aleksander Smywiński-Pohl. On the path to make Ukrainian a high-resource language. In Mariana Romanyshyn, editor, *Proceedings of the Fourth Ukrainian Natural Language Processing Workshop (UNLP 2025)*, pages 120–130, Vienna, Austria (online), July 2025. Association for Computational Linguistics. ISBN 979-8-89176-269-5. doi: 10.18653/v1/2025.unlp-1.14. URL https://aclanthology.org/2025.unlp-1.14/.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.413. URL https://aclanthology.org/2021.findings-acl.413/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. *arXiv preprint arXiv:2507.07955*, 2025.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Guy Kaplan, Matanel Oren, Yuval Reif, and Roy Schwartz. From tokens to words: On the inner lexicon of llms. *arXiv preprint arXiv:2410.05864*, 2024.

Chong Li, Jiajun Zhang, and Chengqing Zong. TokAlign: Efficient vocabulary adaptation via token alignment. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4109–4126, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.207. URL https://aclanthology.org/2025.acl-long.207/.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. XLM-V: Overcoming the vocabulary bottleneck in multilingual masked language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.813. URL https://aclanthology.org/2023.emnlp-main.813/.

Pierre Lison and Jörg Tiedemann. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1147/.

Yihong Liu, Peiqin Lin, Mingyang Wang, and Hinrich Schuetze. OFA: A framework of initializing unseen subword embeddings for efficient large-scale multilingual continued pretraining. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1067–1097, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.68. URL https://aclanthology.org/2024.findings-naacl.68/.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL http://arxiv.org/abs/1711.05101.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsaz. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3992–4006, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.293. URL https://aclanthology.org/2022.naacl-main.293.

Benjamin Minixhofer, Edoardo M. Ponti, and Ivan Vulić. Zero-shot tokenizer transfer. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 46791–46818. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/532ce4fcf853023c4cf2ac38cbc5d002-Paper-Conference.pdf.

Eshaan Nichani, Jason D. Lee, and Alberto Bietti. Understanding factual recall in transformers via associative memories, 2024. URL https://arxiv.org/abs/2412.06538.

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, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. No language left behind: Scaling human-centered machine translation. 2022.

Krzysztof Ociepa, Łukasz Flis, Krzysztof Wróbel, Adrian Gwoździej, and Remigiusz Kinas. Bielik 7b v0.1: A polish language model – development, insights, and evaluation, 2024. URL https://arxiv.org/abs/2410.18565.

Malte Ostendorff and Georg Rehm. Efficient language model training through cross-lingual and progressive transfer learning. *arXiv preprint arXiv:2301.09626*, 2023.

Enes Özeren, Yihong Liu, and Hinrich Schuetze. HYPEROFA: Expanding LLM vocabulary to new languages via hypernetwork-based embedding initialization. In Jin Zhao, Mingyang Wang, and Zhu Liu, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 79–96, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-254-1. doi: 10.18653/v1/2025.acl-srw.6. URL https://aclanthology.org/2025.acl-srw.6/.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. Byte latent transformer: Patches scale better than tokens. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9238–9258, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.453. URL https://aclanthology.org/2025.acl-long.453/.

Yurii Paniv. Isolating llm performance gains in pre-training versus instruction-tuning for mid-resource languages: The ukrainian benchmark study. In *Proceedings of the 15th International Conference on Recent Advances in Natural Language Processing - Natural Language Processing in the Generative AI era*, pages 876–883, Varna, Bulgaria, September 2025. INCOMA Ltd., Shoumen, Bulgaria. URL https://aclanthology.org/2025.ranlp-1.100.

Yurii Paniv, Bohdan Didenko, Mykola Haltiuk, Vladyslav Humennyy, Andrian Kravchenko, Roman Kyslyi, Viktoriia Makovska, Artem Orlovskyi, Bohdan Ruban, Maksym-Yurii Rudko, Anastasiia Senyk, Nazarii Drushchak, Dmytro Chaplynskyi, and Mariana Romanyshyn. Data-Efficient Adaptation of Multilingual LLMs to Ukrainian. In *Proceedings of the Fifth Ukrainian Natural Language Processing Conference (UNLP)*, 2026. URL https://unlp.org.ua/wp-content/uploads/2026/05/data-efficient-adaptation-of-multilingual-llms-to-ukrainian.pdf. Preprint.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162/.

Taido Purason, Pavel Chizhov, Ivan P. Yamshchikov, and Mark Fishel. Teaching old tokenizers new words: Efficient tokenizer adaptation for pretrained models. In Vera Demberg, Kentaro Inui, and Lluís Marquez, editors, *Findings of the Association for Computational Linguistics: EACL 2026*, pages 6492–6516, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.341. URL https://aclanthology.org/2026.findings-eacl.341/.

François Remy, Pieter Delobelle, Bettina Berendt, Kris Demuynck, and Thomas Demeester. Tik-to-tok: Translating language models one token at a time: An embedding initialization strategy for efficient language adaptation. *arXiv preprint arXiv:2310.03477*, 2023.

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. *arXiv preprint arXiv:2408.04303*, 2024.

Piotr Rybak. Maupqa: Massive automatically-created polish question answering dataset. *arXiv preprint arXiv:2305.05486*, 2023.

Shivalika Singh, Angelika Romanou, Clémentine Fourrier, David Ifeoluwa Adelani, Jian Gang Ngui, Daniel Vila-Suero, Peerat Limkonchotiwat, Kelly Marchisio, Wei Qi Leong, Yosephine Susanto, Raymond Ng, Shayne Longpre, Sebastian Ruder, Wei-Yin Ko, Antoine Bosselut, Alice Oh, Andre Martins, Leshem Choshen, Daphne Ippolito, Enzo Ferrante, Marzieh Fadaee, Beyza Ermis, and Sara Hooker. Global MMLU: Understanding and addressing cultural and linguistic biases in multilingual evaluation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18761–18799, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.919. URL https://aclanthology.org/2025.acl-long.919/.

Sho Takase, Ryokan Ri, Shun Kiyono, and Takuya Kato. Large vocabulary size improves large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1015–1026, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.57. URL https://aclanthology.org/2025.findings-acl.57/.

S. Tamang and D. J. Bora. Evaluating tokenizer performance of large language models across official indian languages, 2024. URL https://arxiv.org/abs/2411.12240.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024a.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024b.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Qwen Team. Qwen3 technical report, 2025. URL `https://arxiv.org/abs/2505.09388`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

A Limitations

The first limitation is that MATT only directly optimizes input embeddings. When input and output embeddings are tied, AIM's updates do propagate into the LM head, but the output embeddings still receive no *direct* supervision toward producing good next-token distributions; this likely contributes to the residual gap on generative tasks observed throughout our experiments. For models with untied embeddings the issue is more pronounced, and we show in Appendix F that adding an NTP loss to the AIM objective recovers most of that gap, at a slight cost in training efficiency.

Second, MATT is a self-distillation method, so its performance is effectively upper-bounded by the original model and it cannot introduce capabilities the teacher lacks. It therefore does not replace continual pretraining, which we evaluate in Appendix G.

A further limitation is the need for an additional forward pass during optimization through the model using the original tokenizer to obtain targets for the AIM objective. Although this adds computational overhead, the cost remains lower than a full forward pass through the entire model, as the target layer is positioned roughly one-third of the way through the network.

We have also not tested MATT on encoder-only architectures. In principle, applying it to such models would only require removing the causal constraint in the AIM definition.

Furthermore, we did not evaluate on truly low-resource languages, where benchmark coverage is too sparse for meaningful evaluation; Swahili is the closest proxy in our suite.

BLEU rewards surface n-gram overlap and penalizes equally valid paraphrases, so it does not fully reflect translation quality. All results are single runs without error bars, as we prioritized coverage of settings over repetition, so small differences should not be over-interpreted.

Finally, we do not include a comparison to ZeTT. The authors of Token Distillation use existing ZeTT hypernetworks on a subset of compared models, and show that Token Distillation performs consistently better across a range of different settings. We therefore consider it a sufficient baseline. Comparing against Token Distillation is possible in every setting, and we use it as our baseline in all main body experiments (monolingual and multilingual). The models we used in our experiments (Gemma 3 and Qwen 3 families) do not have any existing ZeTT hypernetworks we could use to cheaply add it as a baseline. Training one from scratch is expensive.

B Segmentation algorithm

Instead of relying on word-based segmentation, we use an offset-based segmentation strategy. Designing a consistent word segmentation across tokenizers is challenging because tokenizers often differ in normalization rules, pre-tokenization steps, language coverage, etc. These differences make it difficult to ensure that segment boundaries match at the word level.

The offset-based method addresses this by operating directly on character offsets in the original text. Given two different tokenizations of the same string, along with the start and end positions of each token, the algorithm searches for all possible split positions that never cut through the middle of any token (see Figure 2).

This approach is universal: such a segmentation always exists, even if the worst case reduces to a single segment spanning the entire input. It can also lead to more precise alignments because the target tokenizer may break a word into several sub-tokens. By working with character offsets, we can introduce mid-word segment boundaries whenever they yield a better match.

For example, consider the sentence *CH4 is a formula for methane*. Suppose the original tokenizer produces the tokens `_for`, `m`, and `ula` for the word *formula*, while the new tokenizer produces `_form` and `ula`. A word-level strategy would force alignment at the whole-word boundary, but an offset-based method can instead match `_for` and `m` with `_form`, and `ula` with `ula`, which more closely respects both tokenizations.

Algorithm 1 provides detailed pseudocode for implementation.

Algorithm 1 Offset–Based Segmentation

Input: teacher offsets O_t, student offsets O_s
Output: teacher segment ids S_t, student segment ids S_s

{initialize outputs and counters}
$S_t \leftarrow [\,], S_s \leftarrow [\,]$
$e \leftarrow -1$ {current end}
$k \leftarrow -1$ {current segment id}

{iterate until both queues empty}
while $O_t \neq \emptyset$ **or** $O_s \neq \emptyset$ **do**
  {if one side empty, label all remaining tokens with current segment id}
  if $O_t = \emptyset$ **then**
    for each o in O_s **do**
      Append k to S_s
    end for
    break
  else if $O_s = \emptyset$ **then**
    for each o in O_t **do**
      Append k to S_t
    end for
    break
  end if
  {peek next offsets}
  $(t_s, t_e) \leftarrow \text{peek}(O_t)$
  $(s_s, s_e) \leftarrow \text{peek}(O_s)$
  {continue with the same segment if overlap}
  if $t_s < e$ **then**
    Append k to S_t, pop(O_t)
    $e \leftarrow \max(e, t_e)$
  else if $s_s < e$ **then**
    Append k to S_s, pop(O_s)
    $e \leftarrow \max(e, s_e)$
  else
    {else start a new segment}
    $k \leftarrow k + 1$
    Append k to S_t and S_s, pop(O_t), pop(O_s)
    $e \leftarrow \max(t_e, s_e)$
  end if
end while

return (S_t, S_s)

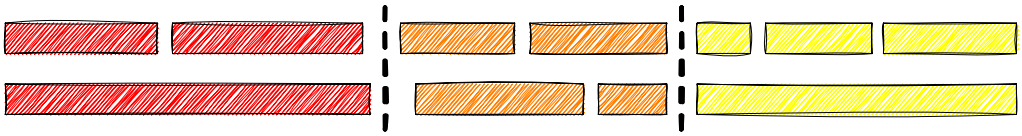


Figure 2: Offset-based segmentation algorithm visualization.

C Tokenizer construction

Our experiments use two procedures. The Ukrainian tokenizer in Section 4.1 replaces part of the base vocabulary and keeps its size fixed, while the per-language tokenizers in Section 4.2 extend it. Both base tokenizers are accessed through the Hugging Face Transformers interface, which represents them as byte-level BPE models, so segmentation is governed by an ordered list of merge rules.

Replacement The construction of Ukrainian tokenizers follows the approach of Didenko [2026]. The vocabulary size stays fixed, Cyrillic entries and entries from other scripts are overwritten with tokens trained on Ukrainian text, and the merge list is rebuilt accordingly. Removing the target-script entries also removes the merge rules that reference them, the target-script region is reconstructed from the byte-level pieces that remain, and each added token receives a merge rule matching how the new tokenizer segments it. All other entries keep their original identifiers. We use the Lapa LLM [Paniv et al., 2026] tokenizer[9] for the main results, and the Tereshchenko Blue tokenizer[10] for ablations.

Extension We train a target-language BPE tokenizer of 100,000 entries on 1 million HPLT 2.0 documents, using SentencePiece for Gemma and the Hugging Face BPE trainer for Qwen. We append each of its entries not already present after the base vocabulary, leaving base token identifiers untouched, and for every appended entry add merge rules for the split points whose two halves are already in the vocabulary. These rules are appended at the end of the ordered list, so all base rules keep their rank and take precedence. Appended rules can still apply once base merges are exhausted, so text in other languages may be segmented slightly differently. The procedure is deliberately simple, and some appended entries are never produced by the tokenizer, so we advise using more careful extension methods [Purason et al., 2026] in practice.

D Convergence speed

This appendix reports a convergence experiment with Gemma 3 4B PT trained on HPLT 2.0 Cleaned using a single multilingual extended tokenizer covering Arabic, German, English, Japanese, and Swahili; its compression statistics are reported in Table 3. We save model checkpoints every 3,000 training steps; the 250k-step row of Table 4 corresponds to the fully trained model, and the earlier checkpoints allow us to observe how quickly the embeddings adapt to the new tokenizer and to evaluate whether training can be substantially shortened.

Table 3: Comparison of the original and the multilingual extended Gemma tokenizer used in this appendix. Compression rate is the average number of characters represented by a single token (higher is better).

Tokenizer	**Vocabulary Size**	**Compression Rate**				
		ar	**de**	**en**	**ja**	**sw**
original	262,145	2.85	3.97	4.32	1.68	2.98
extended	387,980	**3.91**	**4.50**	**4.34**	**2.13**	**4.25**

The AIM objective provides a rich learning signal for tuning the embeddings. Using a mean squared error (MSE) loss, the number of value pairs contributing to the objective is proportional to the product of the head dimensionality, the number of attention heads, the number of possible segment pairs,

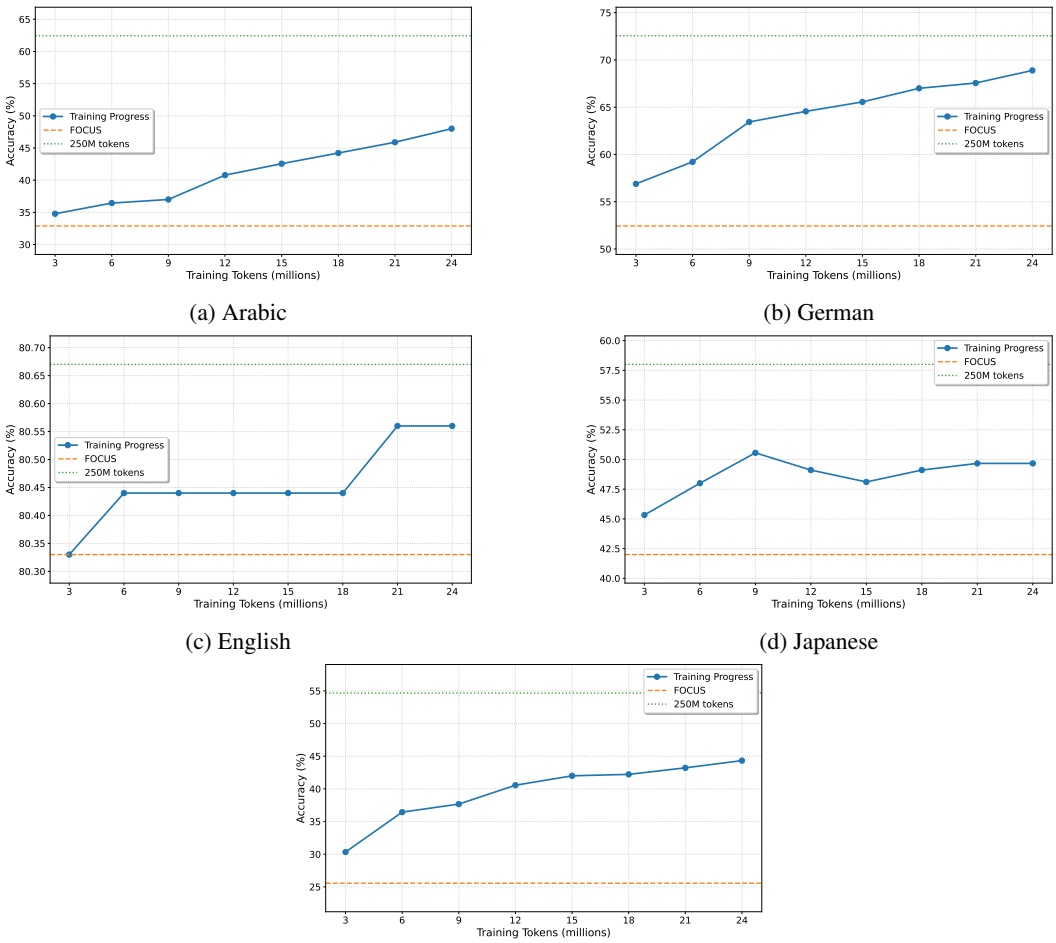
[9] https://huggingface.co/lapa-llm/tokenizer

[10] https://huggingface.co/transhumanist-already-exists/tereshchenkoblue-tokenizer

and the batch size. In the configuration used here – four documents per batch, each truncated to 256 tokens – this amounts to hundreds of millions of pairs at every step of the training.

Table 4 presents the results for the first eight checkpoints on the Belebele benchmark across all tested languages, while Figure 3 provides a visual view of the same trends.

The data show that more than 50% of the final performance gains can be achieved in under 10% of the total training steps, corresponding to fewer than five million tokens per language. This indicates that training time could be cut dramatically with only a minor loss in accuracy, especially if an adaptive data selection strategy is used to prioritize documents that contain a higher proportion of previously unseen tokens.



(a) Arabic

(b) German

(c) English

(d) Japanese

(e) Swahili

Figure 3: Accuracy on the Belebele benchmark over training tokens for five languages, with horizontal lines marking FOCUS initialization and full performance.

E Ablation studies

We apply the MATT method with AIM objective, transferring Gemma 3 4B PT to a Ukrainian-centric tokenizer (Tereshchenko Blue) by Bohdan Didenko [2025], which increases the compression rate by around 50%. We conduct several ablation studies to determine the optimal training configuration, evaluating performance on the Belebele and Global MMLU benchmarks in the same setting as in Section 4. We report the results in Table 5 and describe our insights below.

AIM objective on all layers deteriorates both efficiency and performance compared to only the last one. Since AIM is defined for a single attention layer, we can apply it many times to any

Table 4: Performance on the Belebele benchmark during early training of Gemma 3 4B PT with Model-Aware Tokenizer Transfer, showing rapid gains within the first 10% of steps compared to the full run.

		Belebele				
Steps #	**Tokens #**	**ar**	**de**	**en**	**ja**	**sw**
0k	0M	32.89	52.44	80.33	42.00	25.56
3k	3M	34.78	56.89	80.33	45.33	30.33
6k	6M	36.44	59.22	80.44	48.00	36.44
9k	9M	37.00	63.44	80.44	50.56	37.67
12k	12M	40.78	64.56	80.44	49.11	40.56
15k	15M	42.56	65.56	80.44	48.11	42.00
18k	18M	44.22	67.00	80.44	49.11	42.22
21k	21M	45.89	67.56	80.56	49.67	43.22
24k	24M	48.00	68.89	80.56	49.67	44.33
250k	250M	62.44	72.56	80.67	58.00	54.67

Table 5: Ablation studies of MATT configurations. **3** and **5** denote the number of layers used for AIM objective.

	VRAM (GiB)		**Time**		**Belebele**		**Global MMLU**	
	3	**5**	**3**	**5**	**3**	**5**	**3**	**5**
All Layers vs. Last Layer								
all layers	17.3	26.5	1h 33m	2h 19m	32.56	35.11	28.63	29.00
last layer	**9.1**	**10.1**	**0h 56m**	**1h 04m**	**37.22**	**60.11**	**29.95**	**34.80**
Initialization Method								
WECHSEL	**9.1**	**10.1**	**0h 52m**	**1h 04m**	42.22	59.78	29.82	33.56
FOCUS	**9.1**	**10.1**	0h 55m	1h 06m	37.11	**60.89**	30.10	34.72
Transtokenizers	**9.1**	**10.1**	0h 55m	**1h 04m**	**52.44**	**60.89**	**32.43**	**34.75**

subset of layers. To balance efficiency and accuracy, we compare defining the AIM objective over all layers up to a target depth n against using only the final n-th layer. Results in Table 5 show that restricting AIM to the last layer requires much less VRAM and training time, while also delivering better downstream performance.

FOCUS and Transtokenizers perform similarly on higher layers, while WECHSEL underperforms. Because MATT is independent of the embedding initialization method, different starting points can be tested. We compare WECHSEL, FOCUS, and Transtokenizers. FOCUS and Transtokenizers perform similarly on higher layers, while WECHSEL lags behind (see Table 5). Although Transtokenizers occasionally achieves the best scores, in other experiments, we find FOCUS to be more stable across models, and therefore make it our default choice.

The Transtokenizers method may have an upper hand due to its better utilization of English embeddings, as it learns an English-Ukrainian token-level dictionary from parallel corpora and utilizes it to transfer embeddings from English tokens to their Ukrainian counterparts. Whereas FOCUS utilizes the tokens' overlap to train a FastText model over it, and although it contains English tokens as well, the FastText training corpus contains little data that encompasses both English and Ukrainian text in the same document. This could potentially limit FOCUS to pay attention mostly to Ukrainian overlapped tokens, given the limited usability of English tokens' embeddings.

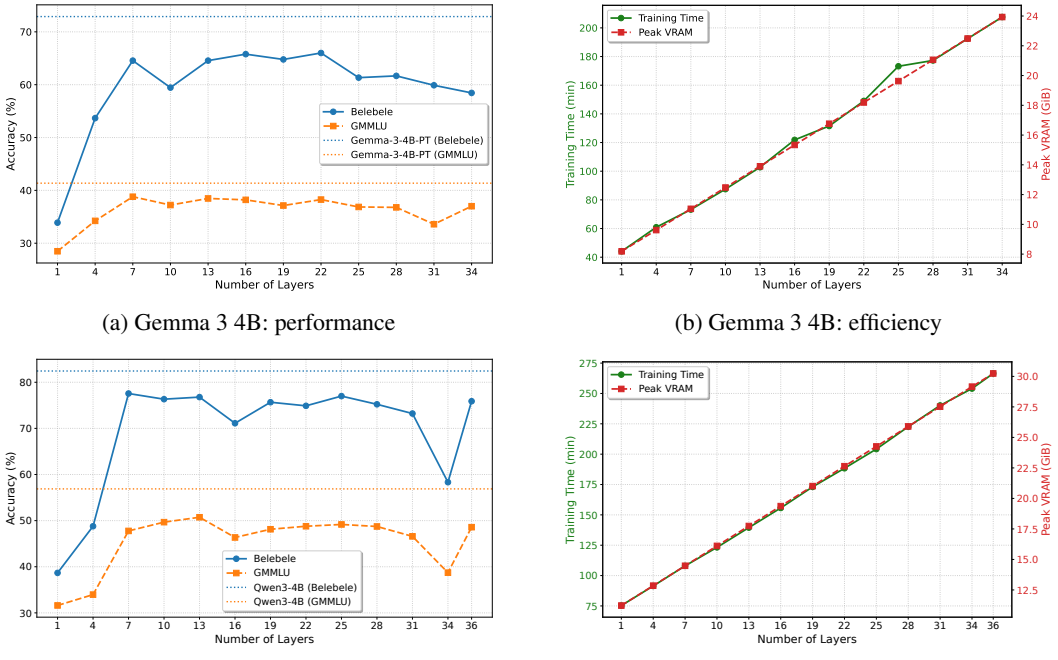
MATT, in the way it uses existing embeddings, is conceptually closer to FOCUS than Transtokenizers, as it models inter-token communication of original tokens, focusing predominantly on Ukrainian ones. As denoted in Appendix D, MATT quickly converges, requiring little data to recover a large part of the original model's performance. This means that differences in the initialization (about 11

percentage points on Avg Disc for Gemma 3 12B PT; see Table 1) are evened out during training. This can also be seen with WECHSEL, which performs considerably worse compared to FOCUS or Transtokenizers (Table 1), but achieves only slightly worse results after a round of MATT training, especially on higher layers (see Table 5). The final observation is presented in Table 1, where we see that even the NTP baseline starting from the Transtokenizers initialization, which is initially better, achieves similar discriminative performance to the NTP over FOCUS.

Minor efficiency differences in Table 5 are likely due to external factors such as checkpointing overhead.

AIM on higher layers leads to better results, but saturates within the first third of the model's depth. MATT allows selecting how deep into the model the AIM objective is applied, creating a natural trade-off between efficiency and accuracy. We evaluate different target depths on Gemma 3 4B with Ukrainian and on Qwen 3 4B with Arabic (Figure 4). On both models, performance rises steeply over the first layers and reaches a plateau at around layer 7, roughly one fifth of the total depth, with no consistent gain from going further. On Gemma 3 4B, performance also declines over the final layers, and on Qwen 3 4B there is an isolated drop at 34 layers that we did not investigate. In contrast, memory consumption and training time continue to grow almost linearly with the number of layers, highlighting the cost of deeper alignment.

This prompts us to recommend choosing layers from the first quarter to the first third of the model's depth to consistently land on the performance plateau, while keeping the computational requirements low. Both sweeps use a smaller training set of 400k documents for speed, so their values are not directly comparable to our main results.



(a) Gemma 3 4B: performance

(b) Gemma 3 4B: efficiency

(c) Qwen 3 4B: performance

(d) Qwen 3 4B: efficiency

Figure 4: Effect of applying the AIM objective to different numbers of layers on Gemma 3 4B (Ukrainian, top) and Qwen 3 4B (Arabic, bottom). The plots show the trade-off between model performance (left) and computational efficiency (right) as the application depth increases. Absolute training times are not comparable between the two rows, as the sweeps ran on different clusters.

Unfreezing shared embeddings weakens performance and efficiency. MATT keeps the embeddings of shared tokens fixed at their original values and only trains the new ones. This protects what the model already knows, but it also prevents shared embeddings from adapting to the target language (e.g., repurposing English tokens for Swahili, since both use Latin script). To test this trade-off, we

repeat the Gemma 3 4B PT runs from Section 4.2 with all input embeddings trainable, changing nothing else.

Table 6 shows that training all embeddings lowers the average score by 1.42 percentage points, while also increasing the training time by about 30%. It also tends to harm other languages. Frozen embeddings score higher in 10 of the 12 Belebele comparisons in Table 7, with the largest differences for the Japanese model, and a similar pattern holds on Global MMLU and MMMLU.

Table 6: Gemma 3 4B PT with frozen shared embeddings versus all input embeddings trained, evaluated on the target language.

	Belebele		**Global MMLU**		**MMMLU**		**Avg Disc**	
	Frozen	**All**	**Frozen**	**All**	**Frozen**	**All**	**Frozen**	**All**
Arabic	**66.67**	63.89	**46.35**	45.79	**45.68**	45.63	**52.90**	51.77
German	**73.56**	71.00	**52.07**	51.95	**52.35**	51.84	**59.33**	58.26
Japanese	**64.33**	59.11	**46.75**	44.85	**46.41**	45.81	**52.50**	49.92
Swahili	**59.56**	58.67	**43.63**	42.66	**42.92**	42.10	**48.70**	47.81
Average	**66.03**	63.17	**47.20**	46.31	**46.84**	46.34	**53.36**	51.94

Table 7: Belebele accuracy of each adapted model on the other languages, with base Gemma 3 4B PT for reference.

Adapted to	**Arabic**		**German**		**Japanese**		**Swahili**	
	Frozen	**All**	**Frozen**	**All**	**Frozen**	**All**	**Frozen**	**All**
Base	69.67		75.67		69.89		66.89	
Arabic	–		**75.67**	75.11	**68.22**	68.00	**64.67**	61.56
German	**69.78**	69.56	–		**68.67**	68.00	64.22	**64.67**
Japanese	**69.33**	65.56	**75.56**	70.33	–		**64.78**	54.89
Swahili	69.44	**69.78**	**75.33**	75.22	**68.56**	67.56	–	

F Performance on models with untied embeddings

The use of tied embeddings varies greatly both between the model families and model sizes. For example, Llama 3.2 1B and Llama 3.2 3B [Grattafiori et al., 2024] both utilize tied embeddings to reduce the number of parameters, whereas a larger Llama 3.1 8B does not. In contrast, the Gemma family [Team et al., 2024a,b, 2025] consistently uses tied embeddings across all sizes.

A significant amount of model pretraining research conducts very little to no experimentation on the effects of embedding tying [Jiang et al., 2023, Touvron et al., 2023, Groeneveld et al., 2024, Team, 2025]. And those that do [Bai et al., 2023] offer a limited explanation of the reasons behind their choice, referring to preliminary results that are not reported in the paper.

More recent research suggests that embedding tying is more effective both from a theoretical standpoint [Bertolotti and Cazzola, 2024] and in achieving lower validation loss and better performance on downstream tasks [Allal et al., 2025]. This leads us to believe that the share of models with tied embeddings may increase in the coming years, making our method even more relevant.

We conducted additional experiments on Qwen 3 8B [Team, 2025], which does not tie embeddings. The presented experiment uses a single multilingual extended Qwen tokenizer covering Arabic, German, English, Japanese, and Swahili (i.e., one tokenizer trained on a multilingual mixture rather than the per-language tokenizers used in Section 4.2); its compression statistics are reported in Table 8.

We train on a limited amount of data relative to the original experimental setting due to resource constraints. The AIM objective used in MATT is enhanced by incorporating an NTP loss, similar to Dobler et al. [2026], to appropriately handle output embeddings in a model with untied embeddings. The losses are aggregated by aligning their scales, rather than by using the norm of the gradients

Table 8: Comparison of the original and the multilingual extended Qwen tokenizer used in this appendix. Compression rate is the average number of characters represented by a single token (higher is better).

Tokenizer	Vocabulary Size	Compression Rate				
		ar	de	en	ja	sw
original	151,669	2.60	3.47	4.36	1.49	2.58
extended	298,833	**3.92**	**4.49**	**4.42**	**2.29**	**4.23**

[Chen et al., 2018], which leaves room for further improvement. We apply AIM on the 16th layer out of 36.

Table 9: Benchmark results for transferring original tokenizers to their extended versions across five languages (Arabic, German, English, Japanese, Swahili) for Qwen 3 8B with untied embeddings.

	Time	Belebele					MMMLU					Global MMLU					Avg
Model		ar	de	en	ja	sw	ar	de	en	ja	sw	ar	de	en	ja	sw	
Qwen 3 8B	-	86.11	91.56	92.56	84.00	60.22	61.91	68.25	74.56	65.77	41.20	66.25	71.25	77.75	66.50	45.25	70.21
FOCUS	-	37.33	60.44	90.78	41.67	29.22	32.62	43.90	70.78	37.16	30.07	30.75	45.00	**72.50**	34.25	30.50	45.80
FOCUS w/ NTP	4h 10m	57.44	79.89	89.67	51.11	35.00	38.35	47.57	68.60	38.95	31.53	38.25	51.25	**72.50**	40.25	29.75	51.34
MATT w/ NTP	1h 40m	50.78	63.33	84.33	38.00	30.00	45.43	57.38	72.25	45.19	32.46	45.25	56.75	**72.50**	41.00	**32.00**	51.11
MATT w/ NTP	**3h 20m**	**77.78**	**85.44**	**91.67**	**64.33**	**38.00**	**48.56**	**59.36**	**72.31**	**46.71**	**33.18**	**50.75**	**58.25**	71.75	**45.50**	31.50	**58.34**

MATT, with NTP loss and unfrozen embeddings, surpasses both the FOCUS and NTP baselines by a considerable margin. It achieved the performance of the NTP baseline with approximately 40% of the compute budget allocated to it, underlining the importance of a rich signal coming from the AIM objective. Further training only improved the results, widening the gap between MATT and baselines.

Although training is slower than with tied embeddings, MATT still outperforms both heuristic and optimization-based baselines even in this setting. Considering that the training runs were shorter than those presented in Section 4.2, we can expect further growth in performance.

G Post-transfer continual pretraining

Transfer can be followed by continual pretraining, so we test whether the choice of transfer method still matters afterwards. We continue four Gemma 3 4B PT checkpoints transferred to the same Arabic tokenizer: MATT, Token Distillation, FOCUS, and FOCUS followed by embeddings-only NTP. Each is trained for about 1B tokens of Arabic HPLT under an identical recipe with all language-model parameters unfrozen (learning rate 10^{-5}, sequence length 8192, about 40 H100-hours per arm), and evaluated at about 200M, 500M and 1B tokens. Table 10 reports the results. Long FLORES uses greedy decoding here, unlike Table 12 (see Appendix I).

On the discriminative benchmarks, differences between methods shrink but do not disappear. MATT leads Global MMLU at every budget and Avg Disc at every budget except 500M, where Token Distillation is ahead. Its Avg Disc lead over Token Distillation falls from 8.35 points before continual pretraining to 0.82 at 1B, and Token Distillation leads MMMLU at 1B.

On translation, ar→en follows the same pattern, with all methods between 38.93 and 40.43 BLEU at 1B. On en→ar, continual pretraining under this recipe does not uniformly help. MATT falls from 20.72 to 18.46 BLEU and FOCUS from 18.16 to 15.17, while Token Distillation rises from 14.49 to 19.01 and finishes ahead of MATT at 1B, and NTP rises from 15.37 to 16.49.

MATT's clearest benefit here is at low budgets. MATT followed by 200M tokens, about a third of the compute of FOCUS followed by 1B tokens including transfer, matches or exceeds it on every discriminative benchmark and in both translation directions. Advantages over Token Distillation and NTP are smaller, and those methods also perform well at higher budgets. These results come from a single model, language and seed. Generally, it is expected that under a large enough continual pretraining budget the gap between tokenizer transfer methods gradually disappears, yet the cost to achieve the same result may be considerably different.

Table 10: Continual pretraining of Gemma 3 4B PT on Arabic. Belebele, Global MMLU, and MMMLU report accuracy (%); Long FLORES reports BLEU under greedy decoding. Budgets are approximate token counts. "Avg Disc" averages the three discriminative metrics. Bold marks the best method per column within each budget.

Method	Belebele	Global MMLU	MMMLU	Long FLORES		Avg Disc
				en→ar	ar→en	
Baseline	69.67	48.40	48.74	20.89	42.33	55.60
Before continual pretraining						
MATT	**66.67**	**46.35**	**45.68**	**20.72**	**39.78**	**52.90**
Token Distillation	52.11	40.26	41.29	14.49	33.05	44.55
FOCUS	36.44	32.50	31.61	18.16	14.27	33.52
NTP	61.00	41.55	41.30	15.37	34.88	47.95
200M tokens						
MATT	**69.78**	**46.08**	**45.46**	**16.67**	**39.71**	**53.77**
Token Distillation	65.33	45.44	45.33	16.48	**39.71**	52.03
FOCUS	61.89	40.65	39.97	9.59	33.42	47.50
NTP	67.44	45.23	45.11	15.12	36.95	52.59
500M tokens						
MATT	65.56	**46.50**	45.24	18.11	39.95	52.43
Token Distillation	**66.11**	45.98	**46.35**	**18.38**	**40.32**	**52.81**
FOCUS	62.00	43.30	42.23	13.75	37.31	49.18
NTP	65.78	45.85	45.47	16.61	38.94	52.37
1B tokens						
MATT	**69.44**	**46.96**	45.91	18.46	**40.43**	**54.10**
Token Distillation	66.44	46.38	**47.01**	**19.01**	40.03	53.28
FOCUS	60.11	43.50	42.14	15.17	39.09	48.58
NTP	65.44	45.62	45.37	16.49	38.93	52.14

H Per-language multilingual results

Per-language details for the multilingual setting in Section 4.2. Table 11 reports the per-language extended tokenizer statistics: original and extended vocabulary sizes, and the compression rate (average number of characters per token) measured on a held-out HPLT 2.0 sample for each target language. Table 12 reports the per-language adaptation results aggregated to means in Table 2; training-time wall-clocks (single H100) appear in the Time column, where Token Distillation runs were configured to match or exceed MATT's per-language compute budget. XL-Sum has no German subset (denoted "–").

Table 11: Per-language extended tokenizer statistics. Compression rate is the average number of characters represented by a single token (higher is better), measured on a held-out HPLT 2.0 sample of the target language.

		Original		Extended	
Family	**Lang**	**Vocab**	**Compr.**	**Vocab**	**Compr.**
Gemma 3 4B PT	ar	262,145	2.85	344,152	**4.15**
	de	262,145	3.97	325,863	**4.65**
	ja	262,145	1.68	338,663	**2.22**
	sw	262,145	2.98	331,378	**3.82**
Qwen 3 4B	ar	151,669	2.60	241,160	**4.19**
	de	151,669	3.47	229,814	**4.58**
	ja	151,669	1.49	239,381	**2.41**
	sw	151,669	2.58	230,444	**3.54**

A note on the Japanese generative metrics: the original Gemma 3 4B PT and Qwen 3 4B already score under 1.5 BLEU on Long FLORES en→ja (0.20 and 1.20) and below 0.5 BLEU on XL-Sum (0.39 and 0.37), so cell-level differences between FOCUS, Token Distillation, and MATT in these columns sit within BLEU noise on a near-zero scale rather than reflecting meaningful differences in generation quality. The Long FLORES x→en direction, where the model generates English, is comparatively well-behaved (10.97 and 24.45 BLEU on the Gemma and Qwen baselines respectively), and the same caution in milder form applies to other low-baseline cells, e.g. XL-Sum on Swahili for Qwen 3 4B (2.05).

Table 12: Per-language multilingual results for Gemma 3 4B PT and Qwen 3 4B with their tokenizers extended per language, evaluated on the target language only. Training time is wall-clock on a single H100.

Method	Training Time	Belebele	Global MMLU	MMMLU	Long FLORES en→x	Long FLORES x→en	XL-Sum	Avg Disc	Avg Gen
Gemma 3 4B PT									
Arabic									
Baseline	–	69.67	48.40	48.74	5.07	20.21	5.12	55.60	10.13
FOCUS	–	36.44	32.50	31.61	0.14	4.60	0.21	33.52	1.65
Token Distillation	7h 29m	52.11	40.26	41.29	0.01	11.40	**1.42**	44.55	4.28
MATT	5h 02m	**66.67**	**46.35**	**45.68**	**0.88**	**12.87**	0.92	**52.90**	**4.89**
German									
Baseline	–	75.67	53.33	53.55	15.55	24.46	–	60.85	20.01
FOCUS	–	60.67	43.04	42.75	6.25	16.29	–	48.82	11.27
Token Distillation	6h 48m	66.78	48.71	48.64	4.95	**20.49**	–	54.71	12.72
MATT	5h 39m	**73.56**	**52.07**	**52.35**	**9.26**	18.29	–	**59.33**	**13.77**
Japanese									
Baseline	–	69.89	49.57	50.39	0.20	10.97	0.39	56.62	3.85
FOCUS	–	44.56	39.19	38.76	0.00	5.66	0.02	40.83	1.89
Token Distillation	6h 31m	59.56	43.65	44.21	**0.02**	**7.63**	**0.08**	49.14	**2.58**
MATT	5h 21m	**64.33**	**46.75**	**46.41**	0.01	6.49	0.01	**52.50**	2.17
Swahili									
Baseline	–	66.89	45.16	44.35	10.88	25.23	6.90	52.13	14.34
FOCUS	–	31.67	34.46	33.39	1.33	7.01	0.32	33.17	2.88
Token Distillation	8h 07m	51.11	39.53	39.93	0.26	11.95	1.76	43.52	4.66
MATT	5h 21m	**59.56**	**43.63**	**42.92**	**5.63**	**15.24**	**2.88**	**48.70**	**7.92**
Qwen 3 4B									
Arabic									
Baseline	–	82.44	56.88	56.15	9.95	35.23	3.36	65.16	16.18
FOCUS	–	38.33	31.48	31.81	2.62	8.61	0.40	33.88	3.88
Token Distillation	7h 12m	60.67	40.98	41.63	0.07	21.61	0.12	47.76	7.27
MATT	6h 35m	**75.22**	**48.25**	**49.26**	**3.07**	**25.48**	**2.26**	**57.58**	**10.27**
German									
Baseline	–	86.22	63.05	62.86	25.83	40.86	–	70.71	33.34
FOCUS	–	55.22	40.68	40.87	14.26	18.65	–	45.59	16.45
Token Distillation	7h 35m	74.44	51.65	52.08	**20.29**	**33.05**	–	59.39	**26.67**
MATT	7h 45m	**82.78**	**56.86**	**57.93**	15.30	29.99	–	**65.86**	22.65
Japanese									
Baseline	–	80.33	60.81	59.45	1.20	24.45	0.37	66.86	8.67
FOCUS	–	40.44	34.76	35.09	**0.21**	7.22	0.02	36.76	2.49
Token Distillation	7h 35m	**67.89**	**49.44**	**49.00**	0.00	15.96	0.00	**55.44**	5.32
MATT	6h 30m	65.89	46.23	47.24	0.11	**17.36**	**0.10**	53.12	**5.86**
Swahili									
Baseline	–	44.89	36.41	36.72	3.10	12.60	2.05	39.34	5.91
FOCUS	–	33.78	30.99	31.14	1.98	3.47	0.25	31.97	1.90
Token Distillation	8h 52m	25.89	29.38	28.01	0.09	0.35	0.00	27.76	0.15
MATT	7h 39m	**41.67**	**33.69**	**34.01**	**2.34**	**9.88**	**0.35**	**36.46**	**4.19**

I Evaluation protocol

All evaluations use lm-evaluation-harness [Gao et al., 2024] with the vLLM backend, a 3-shot prompt with exemplars separated by a blank line, and, for Gemma, an automatically prepended BOS token. Every system within a table is scored on the same tasks, prompts, few-shot construction and documents. Table 13 lists the tasks. Model outputs are not filtered or post-processed before scoring. BLEU is corpus-level, computed with sacreBLEU 2.5.1 under its default tokenizer.

Table 13: Evaluation tasks. Long FLORES is evaluated in both directions. Exemplars are drawn from the few-shot split, excluding the evaluated document where the two splits coincide.

Task	Documents	Few-shot split	Decoding	Stop
Belebele	900	test	likelihood	–
Global MMLU	14,042	dev	likelihood	–
MMMLU	14,042	default	likelihood	–
Long FLORES	281	dev	sampled, $T = 1.0$	newline
WMT24++ (en→uk)	998	test	sampled, $T = 1.0$	none
XL-Sum	4,689–5,399	train	greedy	blank line

Discriminative tasks. Belebele, Global MMLU and MMMLU are scored by loglikelihood over the answer options. Global MMLU uses the full test set of 14,042 questions, except for Table 9, which uses its 400-question Lite version.

Generative tasks. Long FLORES items are complete article segments of about four sentences, with prompts of the form `English text: {source}\nArabic text:`. Generation is limited to 256 tokens. Long FLORES and WMT24++ in Tables 1, 2 and 12 use vLLM's default sampling parameters (temperature 1.0, top-p 1.0, no top-k), and XL-Sum uses greedy decoding. Tables 14 and 15 and the continual pretraining results in Appendix G use greedy decoding throughout.

Table 14 reports the translation tasks of Table 1 under greedy decoding. MATT remains the best transfer method on all three tasks, as in Table 1. Table 15 reports Long FLORES on the models of Table 12 under greedy decoding. MATT is the best transfer method on both mean directions in both families and in 15 of the 16 per-language columns; the exception is Qwen 3 4B on en→de, where Token Distillation leads.

Table 14: Translation performance of Gemma 3 12B PT with selected tokenizer transfer methods under greedy decoding, BLEU.

	Long FLORES		
Method	**en→uk**	**uk→en**	**WMT**
Gemma 3 12B PT	31.47	45.30	6.26
FOCUS	29.70	19.55	5.00
FOCUS w/ NTP (10h 30m)	23.23	42.65	5.37
Token Distillation (8h 20m)	28.11	40.27	5.83
MATT	**30.36**	**44.66**	**8.93**

Software versions. The evaluation pipeline uses lm-evaluation-harness v0.4.8, vLLM 0.8.5 and Transformers 4.55.0, loading models in bfloat16 with `max_model_len=8192`.

Table 15: Long FLORES on the models of Table 12 under greedy decoding, BLEU.

Method	Arabic		German		Japanese		Swahili		Mean	
	en→x	x→en	en→x	x→en	en→x	x→en	en→x	x→en	en→x	x→en
Gemma 3 4B PT										
Baseline	20.89	42.33	34.69	45.33	1.89	28.69	26.80	44.31	21.07	40.17
FOCUS	18.16	14.27	32.64	35.92	0.39	17.39	20.82	18.55	18.00	21.53
Token Dist.	14.49	33.05	33.45	43.82	1.06	24.87	21.94	38.46	17.74	35.05
MATT	**20.72**	**39.78**	**34.60**	**44.72**	**1.46**	**24.92**	**26.81**	**43.37**	**20.90**	**38.20**
Qwen 3 4B										
Baseline	14.39	37.17	28.85	42.54	0.76	26.82	1.60	14.47	11.40	30.25
FOCUS	13.15	10.26	21.43	22.87	1.46	9.04	0.88	3.46	9.23	11.41
Token Dist.	3.15	22.66	**26.93**	34.46	0.01	17.65	0.13	0.20	7.56	18.74
MATT	**14.25**	**31.38**	21.38	**38.54**	**1.63**	**19.42**	**1.27**	**10.72**	**9.63**	**25.02**

J Licenses of used assets

All assets used in this work are publicly released by their original authors and used in accordance with their stated licenses. Table 16 lists the main assets together with their licenses and the URLs where the licenses are declared, so that the reader can verify the license used.

Table 16: Licenses of assets used in this work. Links point to the page where each license is stated.

Asset	License	Source
Base models		
Gemma 3 (4B / 12B PT)	Gemma Terms of Use	ai.google.dev/gemma/terms
Qwen 3 (4B / 8B)	Apache 2.0	huggingface.co/Qwen/Qwen3-4B
Training data		
Kobza	CC0 1.0	huggingface.co/datasets/Goader/kobza
HPLT 2.0 Cleaned	CC0 1.0	huggingface.co/datasets/HPLT/HPLT2.0_cleaned
OpenSubtitles (OPUS)	Unspecified (research use)	opus.nlpl.eu/OpenSubtitles.php
NLLB	CC-BY-SA 4.0	huggingface.co/datasets/openlanguagedata/flores_plus
Evaluation benchmarks		
Belebele	CC-BY-SA 4.0	huggingface.co/datasets/facebook/belebele
Global MMLU	Apache 2.0	huggingface.co/datasets/CohereLabs/Global-MMLU
Global MMLU Lite	Apache 2.0	huggingface.co/datasets/CohereLabs/Global-MMLU-Lite
MMMLU (OpenAI)	MIT	huggingface.co/datasets/openai/MMMLU
FLORES / Long FLORES	CC-BY-SA 4.0	huggingface.co/datasets/openlanguagedata/flores_plus
WMT24++	Apache 2.0	huggingface.co/datasets/google/wmt24pp
XL-Sum	CC-BY-NC-SA 4.0	huggingface.co/datasets/csebuetnlp/xlsum
Frameworks and baseline implementations		
`lm-evaluation-harness`	MIT	github.com/EleutherAI/lm-evaluation-harness
WECHSEL	MIT	github.com/CPJKU/wechsel
`transtokenizers`	MIT	github.com/LAGoM-NLP/transtokenizer
TokAlign	MIT	github.com/ZNLP/TokAlign
FOCUS / `deepfocus`	MIT	github.com/konstantinjdobler/focus
`token-distillation`	MIT	github.com/konstantinjdobler/token-distillation

K Token-level AIM visualization

This appendix accompanies Section 3; it complements Figure 1 with a sequence-level view of the same alignment. Notation matches Section 3.

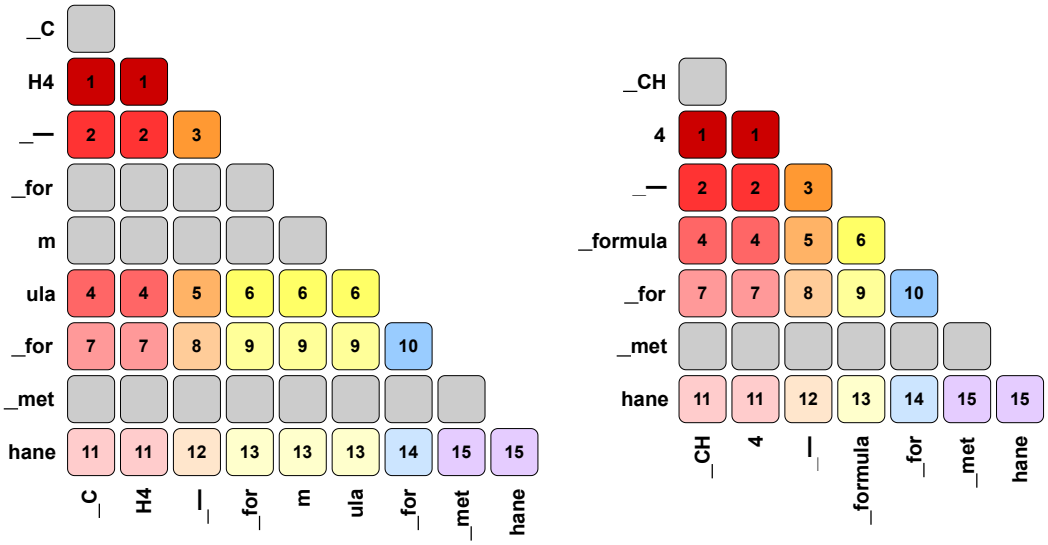


Figure 5: Sequence-level visualization of the AIM alignment between teacher and student models for the same text as Figure 1. Each cell of the (causally masked, hence triangular) attention matrix is the weighted value state $\boldsymbol{v}^*_{i,j}$ for query token t_i (rows) and key/value token t_j (columns); the left matrix uses the original tokenizer T and the right matrix the new tokenizer T'. Numbers and matching colors within a matrix identify tokens that are aggregated into the same segment-level state $\mathfrak{s}_{i,k}$, and numbers and colors across the two matrices indicate corresponding pairs $\mathfrak{s}_{\ell_T(i),j}$ and $\mathfrak{s}'_{\ell_{T'}(i),j}$ that are matched by the loss $\mathcal{L}^*$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state three contributions: (i) the AIM distillation objective (Section 3); (ii) the MATT tokenizer-transfer method, which we show outperforms heuristic and optimization-based baselines on average (Tables 1, 2, 12); and (iii) an evaluation across two model families (Gemma 3, Qwen 3) and five target languages (Ukrainian in Section 4.1; Arabic, German, Japanese, Swahili in Section 4.2 and Appendix H). Section 1 explicitly acknowledges the residual performance gap to the teacher on generative tasks, and Appendix A bounds the scope further (preference for tied embeddings, no truly low-resource languages, no encoder-only architectures), while Appendix G covers continual pretraining on one model and language.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix A discusses (i) the method's preference for tied input/output embeddings (with a mitigation for the untied case in Appendix F), (ii) the upper bound imposed by the original model, which means MATT does not replace continual pretraining, (iii) the extra forward pass through the teacher needed for AIM targets, and (iv) the lack of evaluation on encoder-only architectures.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper presents a methodological/empirical contribution: Section 3 derives the AIM and AIM* objectives from a segment-level reformulation of attention (Eqs. 1–11) but does not state formal theorems that would require proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 fully specifies the AIM/AIM* objectives, the segmentation procedure (with the algorithm in Appendix B), the layer-selection rule, and the partial embedding-freeze scheme; Section 3.5 reports the optimizer (AdamW), learning rate, and FOCUS-based initialization; Section 4 lists every model (Gemma 3 12B/4B PT, Qwen 3 4B, Qwen 3 8B for Appendix F), every dataset (Kobza, HPLT 2.0, Belebele, Global MMLU, MMMLU, Long FLORES, WMT24++, XL-Sum), the training-token budget, the evaluation framework (`lm-evaluation-harness`, 3-shot), and links to all baseline implementations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The implementation of MATT is available at `https://github.com/Goader/matt` under the Apache 2.0 license. The optimizer, learning rate, FOCUS-based initialization, partial embedding-freeze scheme, target-layer choice, and per-experiment training-token budgets are reported in Section 3.5 and Section 4, and the evaluation protocol in Appendix I. Evaluation is performed with the public `lm-evaluation-harness`, and all training data and benchmarks are publicly released (see Appendix J).

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 3.5 reports the optimizer (AdamW), constant learning rate (1×10^{-4}), no weight decay, the partial embedding-freeze scheme, and FOCUS-based initialization,

and notes that hyperparameters were not extensively tuned. Section 4.1 states the training-token budget (≈240M Ukrainian tokens from Kobza) and the AIM target layer (12th, for Gemma 3 12B PT). Section 4.2 reports ≈250M tokens per language from HPLT 2.0, with AIM applied at the 12th layer for Gemma 3 4B PT and the 16th layer for Qwen 3 4B. Evaluation uses `lm-evaluation-harness` with 3-shot prompts on the standard public splits of each benchmark.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We report single-run results without error bars. Each MATT run takes several hours on a single H100 (5–8h for the 4B configurations; 7h for Gemma 3 12B PT; see Table 1, Appendix H, and Appendix F), and across all baselines, model sizes, and languages reported, multi-seed sweeps were not affordable within our compute budget. The averaged results of MATT rank above heuristic and optimization-based baselines across two model families (Gemma 3, Qwen 3), three model sizes (4B, 8B, 12B), and five target languages (de, ja, ar, sw, uk), which provides implicit robustness evidence.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Each individual transfer run uses a single H100 GPU; jobs were dispatched to an HPC cluster, with local development and early experiments performed on a single RTX 5090. Per-run wall-clock training time is reported in Table 1 for the Ukrainian setting (3h 30m–10h 30m for the optimization-based baselines, 5h 30m and 8h 20m for

Token Distillation, 7h for MATT) and in Table 9 for the Qwen 3 8B untied-embeddings experiment (1h 40m–4h 10m). Per-run training times for the multilingual sweep are reported in Appendix H. Continual pretraining takes about 40 H100-hours per arm (Appendix G). Total project compute, including preliminary and discarded runs, is approximately 1500–2500 H100-hours.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses publicly released model weights (Gemma 3, Qwen 3) and publicly released datasets (Kobza, HPLT 2.0, FLORES, WMT24++, XL-Sum, Belebele, Global MMLU, MMMLU) under their respective licenses and involves no human subjects or crowdsourcing.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 1 motivates MATT in terms of broader access to LLMs for lower-resource and distinct-script languages and faster, cheaper inference for end users — the intended positive impacts of the work. Negative impacts are not separately discussed because MATT introduces no new model capabilities; it is a cheaper transfer mechanism for already-released base models, and any dual-use risks (e.g. disinformation or automated abuse in additional languages) inherit from those base models rather than from MATT itself.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases a method for transferring tokenizers; it does not release a new high-risk pre-trained model, a new dataset scraped from the web, or any other artifact that introduces misuse risk beyond that of the underlying base models, which are already publicly released by their original authors under their own licenses.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets used in this work — base models, training data, evaluation benchmarks, and baseline implementations — are cited at first use and are publicly released by their original authors. Appendix J (Table 16) lists the main assets together with their license names and the URL where the license is stated; all assets are used in accordance with their licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces no new datasets or trained model checkpoints as released assets. The MATT implementation is released at `https://github.com/Goader/matt` under the Apache 2.0 license and documented in its README.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work involves no crowdsourcing and no research with human subjects; all training data and benchmarks are pre-existing, publicly available corpora.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work involves no human subjects, so no IRB or equivalent review was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as a component of the core method development. The method, experimental design, and analysis were created manually by the authors. LLMs were used only for writing assistance, editing and grammar correction, which does not require declaration per the NeurIPS LLM policy.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.