PAATRA: PARAMETER-ALLOCATION-AWARE TRAINING FOR SMALL LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Small language models (SLMs) often inherit tokenizers designed for much larger model families. In our 64M-parameter GPT-2-style student using Qwen2.5's 152K-token vocabulary, 81.8% of the parameters are token embeddings. We study this vocabulary–transformer allocation with PAATRA, a pipeline that retains a frequency-ranked subset of the teacher vocabulary, distills from the teacher over that subset, and reinvests the freed parameters into transformer width, subject to an approximately fixed total budget. Distilling from Qwen2.5-0.5B on WikiText-103 with a shared training recipe, students with 47K and 20K tokens reach mean bits per byte (BPB) of 1.3452 ± 0.0064 and 1.3460 ± 0.0026 over three seeds, 4.9% below the full-vocabulary baseline (1.4147 ± 0.0025). A 10K vocabulary is slightly worse than the baseline (1.4210 ± 0.0056), and a 20K vocabulary without reinvestment is worse still (1.4322). Two results qualify this gain. First, in a single-seed learning-rate sweep, the full-vocabulary baseline tuned to 10^{-3} reaches 1.3455, matching the best 20K student. Second, small, genuine BPE/Unigram tokenizers trained with plain cross-entropy under a different protocol achieve lower BPB (1.19–1.27) than all distilled students. We therefore present PAATRA as a controlled measure of the allocation effect under a fixed recipe, for a single teacher and a single corpus. We do not claim that reduced vocabularies win under per-configuration tuning.

1 INTRODUCTION

Many small language models (SLMs) reuse a tokenizer designed for a larger sibling, including models trained by knowledge distillation. The student then inherits a vocabulary and embedding matrix sized for the teacher, regardless of its own capacity or deployment domain. The Gemma 3 270M model, designed for on-device use, places roughly 63% of its 270M parameters (about 170M) in the embedding matrix of the family's 262,144-token vocabulary, leaving about 100M for the transformer blocks.[1]

Under a fixed parameter budget, vocabulary capacity and transformer capacity compete directly. Embedding parameters grow linearly with the hidden dimension (Kd), while transformer parameters grow quadratically ($\approx 12Ld^2$). When a family scales down with the vocabulary held fixed, the embedding share $K/(K+12Ld)$ therefore rises. In our own configurations, a full-vocabulary student ($K = 151{,}665$) solves to width $d = 344$ and places 81.8% of its 63.7M parameters in embeddings. A 20K-vocabulary student solves to $d = 720$ with only 22.3% in embeddings, at nearly the same total (64.6M).

We study this redistribution with four parameter-matched students of about 64M parameters that differ only in their vocabulary–transformer split, plus an 18.5M pruning-only ablation. Each student is distilled from a Qwen2.5-0.5B teacher and evaluated on WikiText-103 in BPB. Under a shared learning rate, the 47K and 20K students improve BPB by 4.9% over the full-vocabulary baseline across three seeds. The 10K-student and pruning-only ablations do not improve. We also report two results that bound this finding. A learning-rate sweep shows that the baseline closes the gap when its

[1] Reported figures vary with whether input embeddings, the output LM head, or tied matrices are counted. Throughout, tied models (Gemma, Llama 3.2, Qwen2.5, SmolLM2) count the shared matrix once, whereas untied models (TinyLlama) count it twice. Qwen2.5-0.5B uses `vocab_size` = 151,936 in its config; our students use its 151,665 actual tokens.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

learning rate is tuned. Separately, genuine small-vocabulary tokenizers trained without distillation outperform all distilled students.

Contributions.

- *Allocation audit.* We quantify how inherited tokenizers consume the parameter budget of open SLMs and frame vocabulary size as an allocation decision under a fixed budget.
- *The PAATRA pipeline.* We combine frequency-based subset-vocabulary construction, restricted-vocabulary logit distillation, and a width solver that reinvests freed embedding parameters.
- *A controlled allocation sweep and its limits.* At a shared recipe, intermediate vocabularies improve BPB non-monotonically (three seeds per configuration), and vocabulary pruning alone does not. We also report the learning-rate sensitivity and the genuine-tokenizer baselines that qualify this result. Code, configurations, and per-run result summaries are released.

2 Related Work

Knowledge distillation for SLMs. Distillation trains a compact student to match a larger teacher's output distribution (Hinton et al., 2015). It was established for encoders by DistilBERT (Sanh et al., 2019), TinyBERT (Jiao et al., 2020), and MiniLM (Wang et al., 2020), and extended to sequence generation (Kim & Rush, 2016). Recent SLM families, such as Gemma 2, train their smallest members via distillation (Gemma Team, 2024). Throughout this line of work, the student inherits the teacher's tokenizer as a fixed precondition. We instead treat the vocabulary–transformer split as a design variable.

Vocabulary and tokenizer design. Subword tokenization (Sennrich et al., 2016; Kudo & Richardson, 2018) made vocabulary size tunable, yet families typically reuse one tokenizer across all model sizes. VocabTailor (Zhang et al., 2025) selects vocabulary components at inference time, TextPruner (Yang et al., 2022) prunes vocabulary and transformer jointly, VocabTrim (Goel et al., 2025) prunes vocabulary for speculative decoding, and PruneSLU (Do et al., 2025) combines vocabulary and structural pruning with distillation. Tao et al. (2024) derive scaling relationships showing that compute-optimal vocabulary size grows with model size, and Mota Borges (2024) report that small-model syntactic competence plateaus around 10K–15K tokens. PAATRA differs from the pruning line by reinvesting freed parameters rather than banking them as compression, and it tests reallocation under a fixed total budget. Table 1 summarizes these differences.

Compute-optimal scaling. Chinchilla (Hoffmann et al., 2022) prescribes an optimal total parameter count N and token count D for a compute budget, and later work accounts for inference cost (Sardana et al., 2024). These laws treat N as a scalar. At a small scale, however, the embedding share can exceed 60% (§3), so how N is partitioned also matters.

Cross-tokenizer distillation. Approximate Likelihood Matching (Minixhofer et al., 2025), Universal Logit Distillation (Boizard et al., 2025), and MultiLevelOT (Cui et al., 2025) transfer knowledge across mismatched tokenizers. Phan et al. (2026) shows that when the student vocabulary is a subset of the teacher's, likelihoods can be computed exactly. PAATRA uses a simpler variant of this subset case: teacher logits are gathered at retained-vocabulary positions, and positions with out-of-vocabulary teacher tokens are masked from the loss.

3 Motivation: Vocabulary–Parameter Imbalance

Table 2 audits seven open SLMs from five families, spanning 270M to 3.2B parameters. Models whose vocabularies were designed for their own scale (SmolLM2 with 49K tokens, TinyLlama with 32K tokens) keep the embedding share near 12–13

We refer to embedding and LM-head parameters as *linguistic capacity* and to transformer-block parameters as *transformer capacity*. These are operational labels only: embeddings also carry

Table 1: Taxonomy of vocabulary selection and internal parameter allocation methods. "Parameters reinvested" means freed vocabulary parameters are explicitly reassigned to transformer capacity.

Method	Learned / subset vocab	Architecture changed?	Fixed budget?	Parameters reinvested?	Retrained tokenizer?	Out-of-domain eval.?
VocabTailor (Zhang et al., 2025)	Subset / selected	No	No	No	No	Not primary
TextPruner (Yang et al., 2022)	Pruned	Optional	Not required	Optional	No	Task-dependent
VocabTrim (Goel et al., 2025)	Pruned	No	No	No	No	Not primary
Scaling laws (Hoffmann et al., 2022; Tao et al., 2024)	Learned / optimized	Not prescribed	Yes, total N	No internal split	Often	Varies
Cross-tokenizer distillation (Minixhofer et al., 2025; Boizard et al., 2025; Phan et al., 2026)	Arbitrary	Student-dependent	Not required	Not prescribed	Often	Varies
PAATRA (this work)	Frequency subset	Width changed; L fixed	Yes, total N	Yes, into width	No	Preliminary (CC-News)

Table 2: Parameter audit of seven open SLMs. "Emb. %" is the share of total parameters in input embeddings plus the LM head (tied matrices counted once). Totals are computed from the public model configurations; Gemma 3 270M uses the officially stated 170M/100M split.

Model	Total params	Vocab size	Emb. %	Trans. %
SmolLM2-360M (Ben Allal et al., 2025)	362M	49,152	13.0%	87.0%
TinyLlama-1.1B (Zhang et al., 2024)	1,100M	32,000	11.9%	88.1%
Llama 3.2 3B (Grattafiori et al., 2024)	3,213M	128,256	12.3%	87.7%
Gemma 2 2B (Gemma Team, 2024)	2,614M	256,000	22.6%	77.4%
Qwen2.5-0.5B (Qwen Team, 2024)	494M	151,936	27.6%	72.4%
Llama 3.2 1B (Grattafiori et al., 2024)	1,236M	128,256	21.3%	78.7%
Gemma 3 270M (Gemma Team, 2025)	270M	262,144	**63.0%**	**37.0%**

semantics, and blocks also process lexical information. Whether inflated linguistic capacity is used is doubtful. Token frequencies are Zipfian, and restricting the Qwen2.5-0.5B teacher to its 50K or 30K most frequent WikiText-103 tokens raises its perplexity by only 1% and 11%, respectively (§6.1). The question we study is whether the displaced transformer capacity can be recovered by reallocation.

4 PAATRA METHODOLOGY

4.1 PROBLEM FORMULATION

Let N denote the student's total parameter budget, which decomposes as

$$N \;=\; N_{\text{emb}} + N_{\text{trans}}, \tag{1}$$

where N_{emb} covers the embedding matrix and LM head and N_{trans} covers everything else. For a GPT-2-style student with tied input–output embeddings, student vocabulary size K, hidden dimension d, and L layers,

$$N_{\text{emb}} = Kd, \qquad N_{\text{trans}} \approx L\,(12d^2 + 13d) + n_{\text{pos}}d, \tag{2}$$

where n_{pos} is the number of learned position embeddings (the maximum sequence length). The final layer norm's $2d$ parameters are omitted. We call N_{emb}/N the *embedding share*.

For a fixed corpus and budget N, there may exist a vocabulary size K^* at which the benefit of extra transformer capacity balances the loss of lexical coverage. Above it, additional vocabulary rows return less than the transformer capacity they displace; below it, coverage loss dominates. The hypothesis predicts a non-monotonic BPB curve over K at fixed N. We test it using a teacher-only screen and a trained, parameter-matched sweep on a single corpus. The sweep is an allocation measurement for that corpus, not a universal estimate of K^*.

4.2 Domain-Specific Vocabulary Construction

Given a target corpus $\mathcal{D}$ and a teacher tokenizer with vocabulary V_T, we tokenize $\mathcal{D}$ with the teacher tokenizer, count occurrences $f_{\mathcal{D}}(t)$ for each $t \in V_T$, and retain the k most frequent tokens together with the teacher's special tokens $\mathcal{S}_T$:

$$V_K \;=\; \mathrm{top}_k\big(V_T;\, f_{\mathcal{D}}\big) \,\cup\, \mathcal{S}_T, \qquad K = |V_K|. \tag{3}$$

Configuration names (K50K, K20K, K10K) refer to the requested k. The resulting size K includes special tokens outside the top-k set; for example, $k = 20{,}000$ gives $K = 20{,}002$. Token IDs are remapped contiguously, so $V_K \subseteq V_T$ and no learned vocabulary projection is needed. Candidate sizes are first screened cheaply by measuring the *teacher's* perplexity when its predictions are restricted to V_K, with no training (§6.1). In this work, we train all candidates and use the screen as a corroborating diagnostic rather than as an automatic selection rule.

4.3 Cross-Tokenizer Student Training

The corpus is tokenized once with the teacher tokenizer. The map $t2s : V_T \to V_K$ converts teacher IDs to student IDs, and $s2t : V_K \to V_T$ maps back. A teacher token outside V_K has no student ID. In the student's *input*, it is replaced by a surrogate, student ID 0 (the retained token with the smallest teacher ID). As a *target*, it is excluded from both loss terms by a binary mask $m_i = \mathbb{1}[y_i \in V_K]$. Let $z^T \in \mathbb{R}^{B \times n_{\mathrm{ctx}} \times |V_T|}$ be the teacher logits for a batch of B sequences of length n_{ctx}. At each position, we gather the teacher's logits of the student's vocabulary,

$$z^{T\downarrow}_{b,j,k} \;=\; z^{T}_{b,j,\, s2t(k)}, \qquad k = 1, \ldots, K, \tag{4}$$

so that a softmax over $z^{T\downarrow}$ is exactly the teacher distribution renormalized over V_K. The student is trained on the masked average over positions i of

$$\mathcal{L}_i \;=\; \alpha\, \tau^2\, \mathrm{KL}\big(\sigma_\tau(z_i^{T\downarrow}) \,\big\|\, \sigma_\tau(z_i^S)\big) \;+\; (1-\alpha)\, \mathrm{CE}(z_i^S, y_i), \tag{5}$$

where σ_τ is the temperature-τ softmax, z^S are the student logits, y_i is the next token, and α weights the two terms ($\tau = 2$, $\alpha = 0.7$). The distillation target is exact for retained positions. Renormalizing over V_K raises the teacher's log-likelihood of a retained token by $-\log Z_{V_K}(c) \geq 0$, where $Z_{V_K}(c)$ is the teacher mass on V_K (Appendix A). Because surrogate IDs alter the student's context, the resulting student scores are not the likelihoods of raw text under a genuine reduced tokenizer.

4.4 Parameter Reinvestment

Shrinking the vocabulary from $|V_T|$ to K frees $(|V_T| - K)d$ embedding parameters at fixed d. PAATRA reinvests them in width, holding depth L fixed:

$$d_K \;=\; \arg \min_{d \in \{H, 2H, 3H, \ldots\}} \big|\, N - Kd - L(12d^2 + 13d) - n_{\mathrm{pos}} d \,\big|, \tag{6}$$

where H is the number of attention heads. The solver uses the exact count, including the final layer norm. Width enlarges every attention and feed-forward block at once, and holding L fixed keeps depth constant across the family, so configurations differ only in (K, d). We report one depth-matched control (§6.5). A systematic width-versus-depth study remains future work.

Algorithm 1 assembles the components. The screen (lines 2–4) needs only teacher inference; the training loops (lines 6–16) run once per configuration. The algorithm returns the whole family; selecting a single configuration would require a held-out validation split, which this study does not use.

5 Experimental Setup

5.1 Configurations

The frozen teacher is Qwen2.5-0.5B (Qwen Team, 2024). Table 3 lists the five students. Full keeps the complete teacher vocabulary. **K50K**, **K20K**, and **K10K** solve Eq. equation 6 for the width that

Algorithm 1 PAATRA training procedure

Require: teacher M_T with vocabulary V_T; corpus $\mathcal{D}$; budget N; candidate sizes $\mathcal{K}$
Ensure: trained student family $\{M_S^K : K \in \mathcal{K}_{\text{train}}\}$
Tokenize $\mathcal{D}$ with the teacher tokenizer; compute frequencies $f_{\mathcal{D}}(t)$ for $t \in V_T$
for all $K \in \mathcal{K}$ **do** ▷ diagnostic screen: teacher inference only
    Build V_K via Eq. 3; evaluate teacher perplexity restricted to V_K
end for
Select $\mathcal{K}_{\text{train}} \subseteq \mathcal{K}$ from the perplexity–K curve ▷ here $\mathcal{K}_{\text{train}} = \mathcal{K}$
for all $K \in \mathcal{K}_{\text{train}}$ **do**
    $d_K \leftarrow$ solve Eq. 6 for budget N ▷ reinvest freed parameters
    Initialize student M_S^K with vocabulary V_K, width d_K, depth L
    Build index maps $s2t : V_K \to V_T$ and $t2s : V_T \to V_K$ (surrogate ID 0 for $t \notin V_K$)
    for all batches $x \sim \mathcal{D}$ (teacher token IDs) **do**
        Loss mask $m_i \leftarrow \mathbb{1}[x_{i+1} \in V_K]$ ▷ mask out-of-vocabulary targets
        $z^T \leftarrow M_T(x)$; $z^{T\downarrow} \leftarrow \text{gather}(z^T, s2t)$ ▷ Eq. 4
        $z^S \leftarrow M_S^K(t2s(x))$ ▷ student sees remapped IDs
        Update M_S^K by descending the masked mean of Eq. 5
    end for
end for
return $\{M_S^K\}$

Table 3: Student configurations (target 64M $\pm 1\%$). Ablation_20K keeps Full's width.

Config	K	d	Params	Emb.%
Full	151,665	344	63.7M	81.8
K50K	47,181	608	64.5M	44.4
K20K	20,002	720	64.6M	22.3
K10K	10,002	760	63.5M	12.0
Ablation_20K	20,002	344	18.5M	37.3

Table 4: Teacher-only screen: perplexity of the frozen teacher restricted to K tokens (ratio to the unrestricted teacher).

K	PPL	Ratio	Regime
50K	20.49	1.01×	Near baseline
30K	22.31	1.11×	Minor
20K	29.17	1.44×	Transition
10K	58.13	2.88×	Degraded
5K	135.28	6.70×	Severe

consumes the freed parameters. The 80K-row training subset contains only about 47.2K distinct teacher tokens, so the requested $k = 50{,}000$ is capped, and K50K ($K = 47{,}181$) retains essentially every token type observed in training. Ablation_20K uses the K20K vocabulary with Full's width ($d = 344$, 18.5M parameters) as a pruning-only control. All four parameter-matched students lie within $\pm 1\%$ of the 64M target.

5.2 Data and training

We use WikiText-103-raw-v1 (Merity et al., 2017). The training data are the first 80,000 rows of the training split with at least 20 characters, tokenized with the teacher tokenizer, and chunked into non-overlapping 512-token sequences. Students are GPT-2 decoders (Radford et al., 2019) with tied embeddings, $L = 8$, $H = 8$, and $n_{\text{pos}} = 512$. Training uses AdamW (weight decay 10^{-2}), peak learning rate 3×10^{-4} with 500 warmup steps and cosine decay, batch size 4, gradient clipping at 1.0, and 15,000 steps (30.7M training tokens). The primary sweep uses seed 42, and all four parameter-matched configurations are repeated with seeds 1 and 2. The student `GPT2Config` sets `bos_token_id` and `eos_token_id` to `None`, because GPT-2's default of 50,256 is out of range for vocabularies below 50,257.

5.3 Evaluation

All comparisons use bits per byte on the full WikiText-103 test split (1,293,436 UTF-8 bytes):

$$\text{BPB} = \frac{\text{NLL}_{\text{sum}}}{n_{\text{bytes}} \cdot \ln 2}, \tag{7}$$

Table 5: BPB on the full WikiText-103 test split under the shared recipe (peak LR 3×10^{-4}), computed on the same raw test bytes with the surrogate-ID protocol of §5. Mean $\pm$ sample sd over seeds 42, 1, and 2. Δ is the relative change versus the mean Full. Per-seed values are in Table 8.

Config	K	Params	BPB ↓	Δ vs. Full	Seeds
K50K	47,181	64.5M	$\mathbf{1.3452 \pm 0.0064}$	−4.9%	3
K20K	20,002	64.6M	$\mathbf{1.3460 \pm 0.0026}$	−4.9%	3
K10K	10,002	63.5M	1.4210 ± 0.0056	+0.4%	3
Full	151,665	63.7M	1.4147 ± 0.0025	—	3
Ablation_20K	20,002	18.5M	1.4322	+1.2%	1

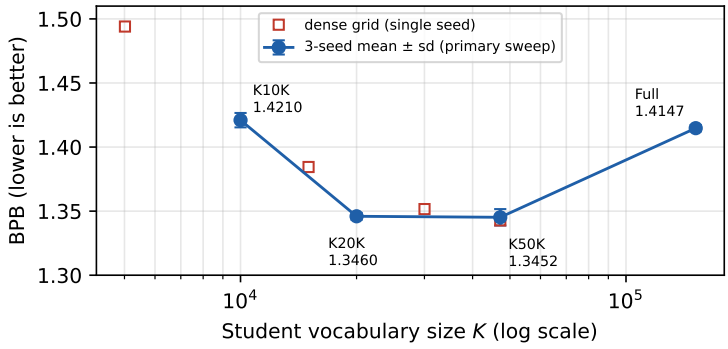

Figure 1: BPB versus student vocabulary size at ≈64M parameters under the shared recipe. Filled markers show 3-seed means $\pm$ sd; hollow squares show single-seed dense-grid runs (§6.5).

where $\mathrm{NLL}_{\mathrm{sum}}$ is the negative log-likelihood in nats summed over all scored test positions (non-overlapping 512-token windows) and n_{bytes} is the number of raw test bytes. All students score the same teacher-tokenized positions. For reduced vocabularies, out-of-vocabulary test tokens are mapped to a surrogate ID and scored accordingly. BPB is therefore comparable across our students within this protocol, but it is not an exact raw-text likelihood under a genuine reduced tokenizer. The surrogate effect is expected to be small when coverage is high (K50K, K20K) and larger for K10K. We report the sample standard deviation over seeds.

6 Results

6.1 Teacher-Only Vocabulary Screen

Table 4 shows that restricting the teacher's vocabulary costs little at 50K and 30K tokens (1–11% higher perplexity), 44% at 20K, and far more at 10K and below (Figure 2, Appendix B). The screen requires only teacher inference and places the three reduced vocabularies into three regimes: near-baseline (47K), transition (20K), and degraded (10K). It diagnoses coverage loss for this teacher and corpus. It does not predict student BPB.

6.2 Parameter-Matched BPB Sweep

Table 5 and Figure 1 report the sweep. Under the shared recipe, K50K and K20K each improve mean BPB by 4.9% over Full. The improvement holds across all seeds: paired per-seed gains are 4.3–5.5% for K50K and 4.5–5.1% for K20K. K10K is slightly worse than Full (+0.4% on average, worse on two of three seeds) despite having the widest transformer (88% of the budget). The two intermediate settings therefore beat both extremes, consistent with the allocation hypothesis at this recipe. K50K and K20K are indistinguishable, and the single-seed dense grid (5K–47K) is consistent with a broad basin between roughly 20K and 47K. Because K50K keeps essentially every token type seen during

Table 6: BPB versus peak learning rate (seed 42; architecture, data, and 15,000-step schedule fixed). The 3×10^{-4} column is an independent rerun of the primary recipe; it differs from the seed-42 values of Table 8 by at most 0.0011. Bold marks each row's best.

Config	$\mathbf{1 \times 10^{-4}}$	$\mathbf{3 \times 10^{-4}}$	$\mathbf{6 \times 10^{-4}}$	$\mathbf{1 \times 10^{-3}}$
Full	1.5897	1.4161	1.3568	**1.3455**
K20K	1.4230	**1.3448**	1.3620	1.4213
K10K	1.4642	**1.4160**	1.4506	1.5008

Table 7: Genuine-tokenizer CE-only baselines under a different protocol from the main sweep (Appendix C). Lower BPB is better.

Tokenizer	**Vocab**	**Params**	**Steps**	**BPB**
BPE	10K	57.4M	15K	1.2258
BPE	20K	51.4M	15K	1.2015
BPE	50K	41.9M	15K	1.1862
Unigram	10K	57.4M	15K	1.2705
Unigram	20K	51.4M	15K	1.2366
Unigram	50K	41.9M	15K	1.2285

training, its gain over Full mostly reflects the removal of about 104K embedding rows for tokens that never occur in the training data.

6.3 Reinvestment Ablation

Ablation_20K shares K20K's vocabulary but keeps $d = 344$. It scores BPB 1.4322, 1.2% worse than the mean Full and 6.4% worse than the mean K20K. Vocabulary pruning alone, therefore, does not reproduce the K20K result. Because the ablation also has far fewer parameters, it does not isolate reinvestment as the only mechanism.

6.4 Learning-Rate Sensitivity

Table 6 shows that the comparison depends on the learning rate. Full improves steadily with larger learning rates and reaches 1.3455 at 10^{-3}, the largest value tried. K20K and K10K peak at 3×10^{-4} and degrade above it. With each configuration tuned over this grid, Full (1.3455) and K20K (1.3448) are indistinguishable, and Full may improve further at higher rates. The 4.9% gain in Table 5 is therefore established only under a shared recipe that happens to be near-optimal for the reduced-vocabulary students. Whether reallocation helps under per-configuration tuning remains open and needs multi-seed sweeps that extend beyond 10^{-3}. One plausible reading is that the embedding-dominated Full model, whose rare-token rows receive sparse updates, prefers larger steps. We have not tested this.

6.5 Genuine-Tokenizer Baselines and Additional Checks

To separate the surrogate-ID protocol from a real reduced tokenizer, we trained six students directly on their own BPE or Unigram tokenizations (10K, 20K, and 50K tokens) with plain cross-entropy and no distillation. These runs follow a *different* protocol from the main sweep (Appendix C). They use the full WikiText-103 training split, a batch size of 8, weight decay of 0.1, and a constant learning rate of 3×10^{-4}. A width solver that counted the tied embedding twice gave them 57.4M, 51.4M, and 41.9M parameters. They reach BPB 1.186–1.270 (Table 7), lower than every distilled PAATRA student, even with fewer parameters and no teacher. The comparison is confounded by these protocol differences and by slightly different scored bytes. However, it indicates that the teacher-tokenized surrogate-ID pipeline is not a competitive approach for training such small models. We therefore read our sweep as a measurement of allocation *within* that pipeline.

A depth-matched K20K control with $L = 16$ and $d = 528$ (64.5M parameters) reaches BPB 1.3561, versus 1.3460 for the $L = 8$ K20K. A single-seed dense grid at K5K/K15K/K30K/K75K (capped at

47,194 tokens) gives 1.4940/1.3844/1.3516/1.3427 (Figure 1). On a fixed 2.54 MB CC-News sample (single seed; vocabularies still selected on WikiText), Full, K50K, K20K, and K10K reach 2.0743, 1.9617, 1.8510, and 2.0143. Surrogate IDs are more frequent out-of-domain, so this ordering is only indicative. Details, losses, and CPU runtime measurements are in Appendix B.

7 Discussion and Limitations

Under one fixed training recipe, moving parameters from rarely used embedding rows into transformer width lowers BPB by about 5% on WikiText-103. The effect is non-monotonic in K: coverage loss erases the gain by 10K tokens, as the teacher-only screen anticipates, and pruning without reinvestment does not help. Stable training curves (Appendix D) make optimization failure an unlikely explanation for K10K. The per-seed consistency of the gain makes it unlikely to be seed noise.

The learning-rate sweep shows that a tuned full-vocabulary baseline matches the best reallocated student, so we do not claim that reallocation dominates under tuned training. Genuine small tokenizers trained without distillation achieve lower BPB than all distilled students, so we do not recommend the surrogate-ID pipeline as a training recipe. Reallocation does not reduce analytical compute per token at a fixed parameter budget (Table 10).

All results use one corpus (WikiText-103), one teacher, one training length, and GPT-2-style students below 100M parameters. The vocabulary is selected from the same corpus used for training. All selection and evaluation use the test split; there is no separate validation split. Surrogate IDs alter the student's context and are scored at evaluation. We do not report per-K out-of-vocabulary rates, which may affect K10K and the CC-News results the most. The learning-rate sweep, dense grid, depth control, ablation, and out-of-domain results are single-seed. Downstream task performance is untested.

8 Conclusion

We presented PAATRA, which treats the vocabulary–transformer split of a small language model as an explicit design variable. At a fixed parameter budget and shared training recipe, replacing a 152K-token inherited vocabulary with a 20K–47K frequency-selected one and reinvesting the saved parameters in width lowers WikiText-103 BPB by 4.9% across three seeds. A 10K vocabulary and pruning without reinvestment do not help. The gain vanishes when the full-vocabulary baseline's learning rate is tuned, and genuinely small tokenizers outperform the distilled pipeline. The allocation effect is therefore real under a fixed recipe but not yet shown to survive tuned training. The next steps are multi-seed learning-rate sweeps for every configuration, genuine-tokenizer training under a matched protocol, and additional corpora and teachers.

AI Use Statement

We used generative AI tools to assist with writing and editing; retrieval and discovery of related work; experiment planning, debugging, and reproducibility packaging; drafting portions of the paper; and formulating and checking mathematical claims. All AI-assisted text, code, references, analyses, experimental results, and mathematical statements were reviewed and verified by the authors. The authors take full responsibility for the final manuscript, claims, experiments, and released artifacts.

Ethics Statement

This work uses publicly available WikiText-103 data and a publicly released pretrained language model. It involves no human participants, human-subject experiments, collection of personal or sensitive information, or individual-level decision-making. The released code and results contain no author-identifying or private data. Reduced vocabularies may lower coverage of rare names, dialects, languages, or specialized terminology in the tail of the training distribution. We did not develop or evaluate the method for harmful, discriminatory, surveillance, or security-sensitive applications. We acknowledge the computational and environmental costs of model training and limited the

experiments to the fixed 15,000-step schedules reported in the paper. The authors have reviewed and will comply with the ICLR Code of Ethics and the applicable licenses and terms for the datasets, models, and software used.

Reproducibility Statement

Full code, pretraining scripts, and configurations are available in the anonymized repository: `https://anonymous.4open.science/r/PAATRA-1970/`.

The seed-42 primary-sweep values and the teacher-screen perplexities are provided as a transcription of the reported numbers, because their raw evaluation files are not bundled. Model checkpoints are not bundled but can be regenerated with the provided commands. Hyperparameters, seeds, and the evaluation protocol are given in §5.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. SmolLM2: When smol goes big—data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

Nicolas Boizard, Kevin El Haddad, Céline Hudelot, and Pierre Colombo. Towards cross-tokenizer distillation: The universal logit distillation loss for LLMs. *Transactions on Machine Learning Research*, 2025.

Xiao Cui, Mo Zhu, Yulei Qin, Liang Xie, Wengang Zhou, and Houqiang Li. Multi-level optimal transport for universal cross-tokenizer knowledge distillation on language models. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, pp. 23724–23732, 2025.

Truong Do, Minh-Phuong Nguyen, and Le-Minh Nguyen. PruneSLU: Efficient on-device spoken language understanding through vocabulary and structural pruning. In *Proceedings of Interspeech 2025*, pp. 1738–1742, 2025. doi: 10.21437/Interspeech.2025-1893.

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Raghavv Goel, Sudhanshu Agrawal, Mukul Gagrani, Junyoung Park, Yifan Zao, He Zhang, Tian Liu, Yiping Yang, Xin Yuan, Jiuyan Lu, Chris Lott, and Mingu Lee. VOCABTRIM: Vocabulary pruning for efficient speculative decoding in LLMs. *arXiv preprint arXiv:2506.22694*, 2025.

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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4163–4174, 2020.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, 2016.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, 2018.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. Universal cross-tokenizer distillation via approximate likelihood matching. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Rafael Braga Medeiros Mota Borges. *Tokenization Matters: Training Your Tokenizer Right—Testing the Impact of Tokenization on Language Modelling with (Small) Transformers*. Bachelor's thesis, Delft University of Technology, 2024.

Buu Phan, Ashish Khisti, and Karen Ullrich. Cross-tokenizer likelihood scoring algorithms for language model distillation. In *International Conference on Learning Representations*, 2026.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Nikhil Sardana, Jacob Portes, Sasha Doubov, and Jonathan Frankle. Beyond chinchilla-optimal: Accounting for inference in language model scaling laws. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 43445–43460, 2024.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1715–1725, 2016.

Chaofan Tao, Qian Liu, Longxu Dou, Niklas Muennighoff, Zhongwei Wan, Ping Luo, Min Lin, and Ngai Wong. Scaling laws with vocabulary: Larger models deserve larger vocabularies. In *Advances in Neural Information Processing Systems*, volume 37, pp. 114147–114179, 2024.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Ziqing Yang, Yiming Cui, and Zhigang Chen. TextPruner: A model pruning toolkit for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 35–43, 2022.

Hanling Zhang, Yayu Zhou, Tongcheng Fang, Zhihang Yuan, Guohao Dai, Wanli Ouyang, and Yu Wang. VocabTailor: Dynamic vocabulary selection for downstream tasks in small language models. *arXiv preprint arXiv:2508.15229*, 2025.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

A Restricted-Vocabulary Likelihood

For a context c and teacher next-token distribution $p_T(\cdot \mid c)$, let $Z_{V_K}(c) = \sum_{v \in V_K} p_T(v \mid c) \in (0, 1]$. The restricted teacher distribution is

$$p_T^{V_K}(v \mid c) = \frac{p_T(v \mid c)}{Z_{V_K}(c)}, \qquad v \in V_K, \tag{8}$$

and for a retained target $y \in V_K$,

$$\log p_T^{V_K}(y \mid c) - \log p_T(y \mid c) = -\log Z_{V_K}(c) \geq 0. \tag{9}$$

Summed over retained positions $\mathcal{I}$, the teacher's gap is $\sum_{i \in \mathcal{I}} -\log Z_{V_K}(c_i)$. In BPB units, divide by $n_{\text{bytes}} \ln 2$. No bound based on coverage alone exists without a lower bound on Z_{V_K}. This identity concerns the *teacher* at retained positions. It neither bounds nor corrects the reported *student* BPB, which also scores out-of-vocabulary positions as the surrogate ID and conditions on contexts containing surrogate IDs. Exact comparability would require a genuine reduced-tokenizer segmentation or an explicit cross-tokenizer likelihood computation (Phan et al., 2026).

B Additional Results

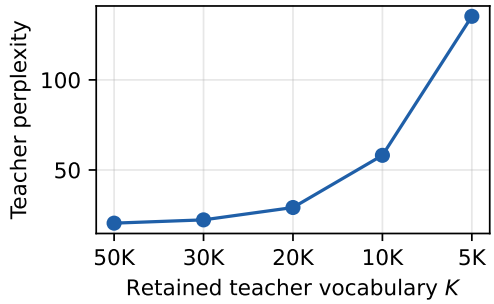

Figure 2: Teacher-only vocabulary screen on WikiText-103 (values of Table 4). Perplexity changes little at 50K–30K retained tokens and rises steeply below 20K.

Per-seed results. Table 8 lists every run behind Table 5. Seeds 1 and 2 of K50K and K10K come from the replication runs in the released package.

Table 8: Per-seed BPB under the shared recipe (WikiText-103 test).

Config	Seed 42	Seed 1	Seed 2	Mean $\pm$ sd
Full	1.4165	1.4119	1.4158	1.4147 ± 0.0025
K50K	1.3384	1.3511	1.3461	1.3452 ± 0.0064
K20K	1.3438	1.3489	1.3452	1.3460 ± 0.0026
K10K	1.4149	1.4220	1.4260	1.4210 ± 0.0056

Depth control and dense grid. Table 9 lists the single-seed runs of §6.5. These runs used a different vocabulary build that adds 14 rather than 2 non-frequency-ranked tokens (for example, $K = 20{,}015$ for the depth control). For K75K, the request for 75,000 tokens is capped by the roughly 47.2K token types present in the training subset, giving $K = 47{,}194$ in this build. All final bundles match their training logs and architecture metadata.

Learning-rate sweep losses. For K10K, the final total/KD/CE losses at learning rates 10^{-4}, 3×10^{-4}, 6×10^{-4}, and 10^{-3} are 2.523/2.079/3.558, 2.245/1.790/3.305, 2.355/1.903/3.411, and 2.550/2.102/3.596. All four runs have 63,520,800 parameters ($K = 10{,}002$, $d = 760$, $L = 8$).

Table 9: Dense-grid and depth-control runs (single seed, shared recipe). Losses are at step 15,000.

Run	K	d	L	Params	Total	KD	CE	BPB
K5K	5,014	784	8	63.42M	2.269	1.886	3.162	1.4940
K15K	15,014	736	8	63.51M	2.242	1.750	3.389	1.3844
K30K	30,014	672	8	63.94M	2.221	1.683	3.474	1.3516
K75K	47,194	608	8	64.56M	2.252	1.698	3.545	1.3427
K20K, $L = 16$	20,015	528	16	64.48M	2.297	1.780	3.504	1.3561

Out-of-domain diagnostic. We evaluated the existing Full, K50K, K20K, and K10K checkpoints on a fixed 2,544,280-byte CC-News sample, obtaining BPB 2.0743, 1.9617, 1.8510, and 2.0143. Planned GitHub Code and PubMed evaluations could not be run because their dataset downloads failed.

CPU runtime and analytical compute. Table 10 reports CPU inference measurements for the existing checkpoints. They use PyTorch, sequence length 512, and three timed forward passes after one warmup pass. Latency is per token at a batch size of 1, and throughput is measured at a batch size of 16. Analytical compute counts 2 FLOPs per multiply-accumulate: $2 \cdot 12Ld^2$ for the blocks, $2 \cdot 2Ln_{\text{ctx}}d$ for attention scores and values at full context, and $2Kd$ for the tied output projection. At a fixed parameter budget, reduced vocabularies lower the output-projection cost but raise block cost, so total compute per token is slightly *higher* than for Full. CPU latency at a batch size of 1 also increases with width. GPU measurements were not collected. The pruning-only ablation is fastest because it is 3.5× smaller; it is not a fixed-budget comparison.

Table 10: CPU inference measurements and analytical compute per token.

Config	CPU ms/token (B=1)	CPU tok/s (B=16)	MFLOPs/token
Full	0.084	13,626	132.7
K50K	0.114	13,930	138.3
K20K	0.122	16,771	140.1
K10K	0.127	14,493	138.6
Ablation_20K	0.076	35,867	42.1

C Genuine-Tokenizer Baselines

We trained BPE and Unigram tokenizers (HuggingFace `tokenizers`; four special tokens; BPE minimum frequency 2) with 10K, 20K, and 50K tokens. We then trained tied GPT-2 students ($L = 8$, $H = 8$, context 512) with cross-entropy only for 15,000 steps. The setup differs from the main sweep in five ways. (i) The training data are all rows of the WikiText-103 training split with at least 20 characters, not the first 80,000. (ii) Batch size is 8, so these runs see twice as many tokens. (iii) Weight decay is 0.1. (iv) The learning rate is a constant 3×10^{-4}, with no warmup or decay. (v) The width solver counted the tied embedding matrix twice, so the models have 57.4M, 51.4M, and 41.9M parameters instead of ≈64M, and they shrink as the vocabulary grows. Evaluation scores non-overlapping 512-token chunks of the tokenized test split, covering 1,287,656 bytes rather than the 1,293,436 used elsewhere. With both families, BPB decreases as vocabulary grows, even though parameters shrink, and BPE outperforms Unigram at each size. A matched-protocol comparison against PAATRA remains to be done.

D Training Curves and Loss Details

Table 11 and Figure 3 report the primary-sweep training losses. Total loss equals $0.7 \cdot \text{KD} + 0.3 \cdot \text{CE}$, where KD includes the τ^2 factor. CE and KD are computed over different vocabularies (V_K) and are therefore not comparable across rows. The curves decrease at every 1,000-step logging interval without divergence or spikes. We observed no optimization instability at any embedding share, including K10K's 12%.

Table 11: Training losses at step 15,000 for the primary-sweep runs. Training times are wall-clock.

Config	Total	KD	CE	Train time
Full	2.526	1.927	3.923	103 min
K50K	2.247	1.689	3.551	85 min
K20K	2.222	1.709	3.419	80 min
K10K	2.244	1.790	3.303	77 min
Ablation_20K	2.604	2.084	3.819	50 min

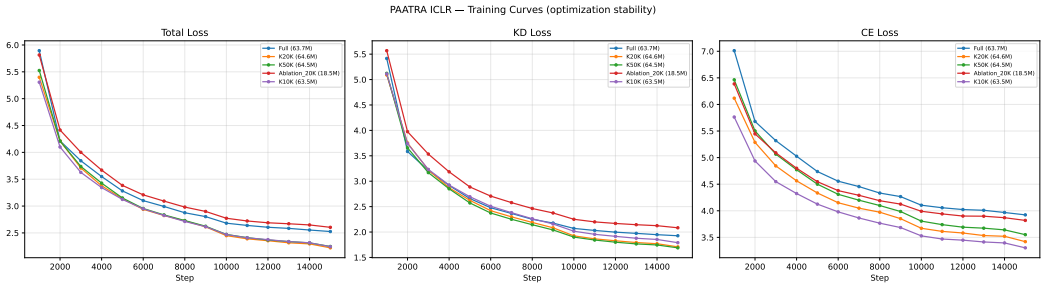

Figure 3: Training-loss trajectories (total, KD, and CE components) for the five primary-sweep configurations, logged every 1,000 steps.