LEAP: Layer-wise Exit-Aware Pretraining for Efficient Transformer Inference

Shashank Kapadia[1], **Deep Naryan Mishra**[1], **Sujal Reddy Alugubelli**[2],
Haoan Wang[1], Saipraveen Vabbilisetty[1], Rishi Bhatia[1], Anupriya Sharma[1]
[1]Walmart Inc. [2]Sam's Club
{Shashank.Kapadia, Deep.Mishra, Haoan.Wang}@walmart.com
{Saipraveen.Vabbilisetty, Rishi.Bhatia, Anupriya.Sharma}@walmart.com
SujalReddy.Alugubelli@samsclub.com

Abstract

Layer-aligned distillation and convergence-based early exit represent two predominant computational efficiency paradigms for transformer inference; yet we establish that they exhibit systematic incompatibility under standard deployment conditions for convergence-based early exit. Distillation objectives that align intermediate student layers to teacher representations suppress the representational convergence that early-exit mechanisms exploit, rendering such mechanisms ineffective on distilled models.

We introduce **LEAP** (Layer-wise Exit-Aware Pretraining), an auxiliary training objective that reconciles this incompatibility. LEAP requires no architectural modifications; it augments standard distillation with a single constraint ensuring intermediate layers approximate final-layer representations. LEAP-MiniLM achieves **1.61× measured wall-clock speedup** (batch=1, NVIDIA L4) at θ=0.95, with 91.9% of samples exiting by layer 7 and 1.80× theoretical layer reduction, where standard distilled models achieve *zero* effective speedup. We validate across sentence similarity (STS-B: 0.760 ± 0.006) and retrieval benchmarks (BEIR), providing operational guidance including latency measurements, decision thresholds, and deployment criteria.

Industry Track: Emerging

1 Introduction

Dense text embeddings underpin modern information retrieval (Reimers and Gurevych, 2019), semantic search, and recommendation systems. Two predominant paradigms exist for efficient embedding inference: *knowledge distillation*, producing compact models such as MiniLM (Wang et al., 2020) and DistilBERT (Sanh et al., 2019), and *early exit*, terminating computation when representations converge (Xin et al., 2020; Zhou et al., 2020). These orthogonal optimizations should, in principle, compose - distilled models should admit further acceleration through early exit. We establish that this assumption does not hold empirically.

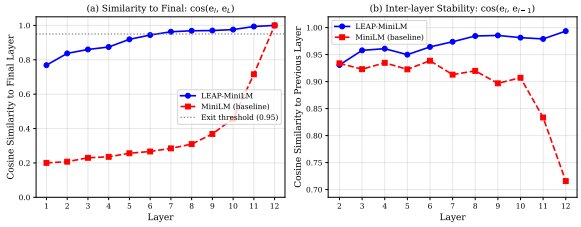

Figure 1: Layer dynamics comparison. (a) Cosine similarity to final layer: LEAP-MiniLM exceeds exit threshold (θ=0.95) from layer 7; baseline MiniLM remains below threshold until layer 12. (b) Inter-layer stability: LEAP maintains high stability (avg: 0.981) while achieving early convergence.

1.1 The Distillation-Exit Incompatibility

The incompatibility manifests operationally: practitioners deploy distilled models with early-exit infrastructure, observe termination conditions satisfied at intermediate layers, yet measure *zero* effective layer reduction. Per-layer convergence monitoring imposes overhead without compensating early terminations, causing net throughput degradation.

The underlying cause is intrinsic to the optimization objective, not hyperparameter selection. Layer-aligned distillation objectives suppress the representational redundancy that early exit exploits. Standard distilled models exhibit uniformly distributed computation across layers - representations never stabilize before the final layer (Figure 1).

1.2 The Mechanism: Why Layer-Aligned Distillation Eliminates Exit Points

Layer-aligned distillation trains each student layer to match the corresponding teacher layer, inducing uniform distribution of the teacher's representational capacity across the student's depth hierarchy. This suppresses monotonic convergence—later

layers performing diminishing transformations—precisely the property early exit requires.

1.3 LEAP: Reconciling Distillation with Early Exit

We introduce **LEAP** (Layer-wise Exit-Aware Pretraining), an auxiliary training objective that reconciles this incompatibility. LEAP augments standard distillation with a single constraint: ensuring intermediate layers produce representations approximating the final layer. This objective requires no architectural modifications and introduces no inference-time parameters.

1.4 Contributions

1. **Empirical characterization**: We establish that layer-aligned distillation and convergence-based early exit exhibit systematic incompatibility, characterizing conditions under which this conflict arises (§3.1).

2. **LEAP**: An auxiliary training objective that reconciles distillation with early exit, requiring no architectural modifications.

3. **Empirical validation**: 1.61× wall-clock speedup (1.80× layer reduction) where standard distillation yields none, validated on sentence similarity and retrieval tasks with GPU latency benchmarks across batch sizes.

4. **Operational framework**: Decision thresholds, wall-clock measurements, failure diagnostics, and deployment criteria for practitioners.

2 Related Work

Early Exit. DeeBERT (Xin et al., 2020), FastBERT (Liu et al., 2020), PABEE (Zhou et al., 2020), LeeBERT (Zhu, 2021), BERxiT (Xin et al., 2021), and CALM (Schuster et al., 2022) enable adaptive inference via learned exit classifiers or patience-based criteria. DeeBERT and LeeBERT attach trained exit heads at each layer, adding parameters and requiring task-specific fine-tuning of these heads. PABEE and BERxiT employ patience counters or learned entropy thresholds. All implicitly presuppose sufficient inter-layer representational redundancy - an assumption that fails for distilled models. LEAP's convergence-based criterion is parameter-free and task-agnostic: it requires no additional modules, operating directly on the similarity between intermediate and final representations. We find that DeeBERT-style learned exit heads achieve only 0.26 STS-B Spearman on our MiniLM-L12 backbone (vs. LEAP's 0.76), as these heads are designed for classification confidence, not embedding quality.

Knowledge Distillation. MiniLM (Wang et al., 2020), DistilBERT (Sanh et al., 2019), and TinyBERT (Jiao et al., 2020) compress transformers via layer-aligned objectives. DynaBERT (Hou et al., 2020) enables dynamic width/depth. Critically, standard distillation distributes computation uniformly across layers, eliminating the redundancy early exit exploits.

Efficient Inference & Dynamic Computation. Pruning (Michel et al., 2019), quantization (Zafrir et al., 2019), and LayerDrop (Fan et al., 2020) reduce per-layer cost; ACT (Graves, 2016), Universal Transformers (Dehghani et al., 2019), and PonderNet (Banino et al., 2021) learn halting criteria. Matryoshka representations (Kusupati et al., 2022) enable variable embedding dimensions, reducing memory and distance computation costs. LEAP is orthogonal: Matryoshka varies the *width* of the final embedding while LEAP varies the *depth* at which it is computed. Combined, this could yield multiplicative benefits—Matryoshka for storage/retrieval efficiency, LEAP for inference latency.

3 Methodology

3.1 Problem Analysis: Why Layer-Aligned Distillation Suppresses Early Exit

Consider a teacher model M_t with L_t layers and a student M_s with $L_s < L_t$ layers. Standard distillation minimizes:

$$\mathcal{L}_{\text{distill}} = \sum_{l=1}^{L_s} \text{KL}(\mathbf{h}_s^{(l)} \| \mathbf{h}_t^{(\pi(l))}) \tag{1}$$

where $\pi(l)$ maps student layers to teacher layers and KL denotes Kullback-Leibler divergence over attention distributions (as in MiniLM). LEAP instead employs cosine similarity for embedding alignment (§3.2). This formulation explicitly trains *every* student layer to match the teacher, distributing computation uniformly.

The Redundancy Problem. Early-exit predicates necessitate monotonically diminishing inter-layer transformations - i.e., $\|\mathbf{p}_{l+1} - \mathbf{p}_l\| \to 0$ as

$l \to L$. Define the *contraction ratio*:

$$\gamma_l = \frac{\|\mathbf{p}_{l+1} - \mathbf{p}_l\|}{\|\mathbf{p}_l - \mathbf{p}_{l-1}\|} \tag{2}$$

where $\mathbf{p}_l$ is the mean-pooled representation at layer l. Early exit is viable when $\gamma_l < 1$ consistently. Standard distillation produces $\gamma_l \approx 1$ throughout - no natural exit points.

Scope. This analysis applies to layer-aligned distillation with uniform or proportional layer mapping π. Distillation variants that match only the final output (e.g., output-only KD) do not impose per-layer alignment and may preserve intermediate convergence. We focus on layer-aligned objectives as they constitute standard practice in MiniLM, TinyBERT, and similar widely deployed models.

3.2 LEAP: Preserving Early Exit During Distillation

We introduce **LEAP**, an auxiliary training objective that achieves distillation benefits while preserving early-exit capability:

$$\mathcal{L}_{\text{LEAP}} = \mathcal{L}_{\text{final}} + \alpha\mathcal{L}_{\text{inter}} + \beta\mathcal{L}_{\text{exit}} + \delta\mathcal{L}_{\text{contrast}} \tag{3}$$

The principal methodological contribution is that $\mathcal{L}_{\text{exit}}$ is essential - without it, distilled models cannot support early exit (see Appendix C.5).

Final Layer Distillation ($\mathcal{L}_{\text{final}}$). Standard output matching: $\mathcal{L}_{\text{final}} = 1 - \cos(\mathbf{e}_s^{(L_s)}, \mathbf{e}_t^{(L_t)})$ where $\mathbf{e}$ denotes the normalized mean-pooled embedding.

Intermediate Layer Distillation ($\mathcal{L}_{\text{inter}}$). Layer-wise matching: $\mathcal{L}_{\text{inter}} = \frac{1}{L_s}\sum_{l=1}^{L_s}(1 - \cos(\mathbf{e}_s^{(l)}, \mathbf{e}_t^{(\pi(l))}))$ with uniform layer mapping.

Early Exit Quality Loss ($\mathcal{L}_{\text{exit}}$). The key innovation: we train intermediate layers to produce valid exit points employing a *soft margin loss* with dual targets - the teacher's final layer and the student's own final layer:

$$\mathcal{L}_{\text{exit}}^{(t)} = \frac{1}{L_s}\sum_{l=1}^{L_s} w_l \cdot \sigma\big(10 \cdot (\tau - \cos(\mathbf{e}_s^{(l)}, \mathbf{e}_t^{(L_t)}))\big) \tag{4}$$

$$\mathcal{L}_{\text{exit}}^{(s)} = \frac{1}{L_s - 1}\sum_{l=1}^{L_s-1} w_l \cdot \sigma\big(10 \cdot (\tau - \cos(\mathbf{e}_s^{(l)}, \text{sg}(\mathbf{e}_s^{(L_s)}))\big)\big) \tag{5}$$

where σ is sigmoid, $\tau = 0.98$ is the exit threshold, w_l are layer weights, and $\text{sg}(\cdot)$ denotes stop-gradient. Total: $\mathcal{L}_{\text{exit}} = \mathcal{L}_{\text{exit}}^{(t)} + 0.7\mathcal{L}_{\text{exit}}^{(s)}$; the 0.7 weight balances teacher-aligned quality with inference-time self-consistency.

Contrastive Distillation ($\mathcal{L}_{\text{contrast}}$). We align similarity structure: $\mathcal{L}_{\text{contrast}} = \text{KL}(\text{softmax}(\mathbf{S}_s/\tau_c)\|\text{softmax}(\mathbf{S}_t/\tau_c))$ where $\mathbf{S}$ are batch-wise similarity matrices.

3.3 Convergence-Based Early Exit Inference

At inference, we employ k-skip patience-based early exit requiring no additional parameters. The patience parameter k specifies the layer gap over which convergence is measured - comparing the current layer's representation to that of k layers prior. After each layer $l \geq l_{\min}$, we compute mean-pooled representations $\mathbf{p}_l$ and exit when:

$$s_l = \cos(\mathbf{p}_l, \mathbf{p}_{l-k}) \geq \theta \tag{6}$$

where k is patience (default k=1), $l_{\min}$=6, and θ=0.95. Samples meeting the convergence criterion exit immediately; remaining samples proceed to deeper layers. Full pseudocode in Appendix B.6.

Training vs. Inference Threshold. We use different thresholds: τ=0.98 (training) pushes representations closer to final layer; θ=0.95 (inference) allows earlier exits with headroom. This separation provides robustness to distribution shift.

4 Results

4.1 Experimental Setup

We compare two 12-layer models: (1) **MiniLM-L12 (baseline)**: trained with our distillation pipeline *without* $\mathcal{L}_{\text{exit}}$; (2) **LEAP-MiniLM-L12**: trained with full LEAP. Both use the same architecture, training data (AllNLI, 1.5M sentences), and teacher (`bert-large-nli-mean-tokens`). This controlled comparison isolates the exit constraint effect. Training: 10 epochs, batch size 64, LR 5×10^{-5}, $\tau = 0.98$ (training), $\theta = 0.95$ (inference). Full details in Appendix B.

4.2 Main Results

Table 1 presents our main findings. Conventionally distilled models exhibit zero marginal utility from early-exit mechanisms - representations never stabilize before the final layer. LEAP-MiniLM achieves 1.61× wall-clock speedup (1.80× layer reduction) with 91.9% of samples exiting by layer 7.

Model	STS-B ρ	Layer Red.	Wall-Clock†	E[layers]	Exit@L7
Published MiniLM-L12-v2‡	0.831	1.00×	1.00×	12.0	0%
MiniLM-L12 (baseline)	0.777	1.00×	1.00×	12.0	0%
LEAP-MiniLM-L12	0.760 ± 0.006	1.80×	1.61×	6.7	91.9%

Table 1: Main results at θ=0.95. STS-B with 95% CI (3 seeds). Layer Red. = $L/\mathbb{E}[\text{exit layer}]$. †Wall-clock speedup measured on NVIDIA L4, batch=1. Baseline trained with our pipeline without $\mathcal{L}_{\text{exit}}$. ‡External: sentence-transformers/all-MiniLM-L12-v2.

Model	Distillation	Max Exit Rate
TinyBERT-6	Layer-aligned (MSE)	**0.0%**
MiniLM-L6-v2	Layer-aligned (KL)	**0.7%**
DistilBERT-6	Output-only	71.5%

Table 2: Exit compatibility across distillation methods. Layer-aligned methods suppress early exit; output-only distillation preserves it.

External Validation. We verify generalization beyond our controlled setup using the published `all-MiniLM-L12-v2` (STS-B: 0.831). This production model achieves **0% exit rate** at all intermediate layers (L7 similarity: 0.29 vs. LEAP's 0.96), confirming the incompatibility is intrinsic to layer-aligned distillation, not our training pipeline.

Cross-Distillation Generalization. We test exit compatibility across distillation methods (Table 2). Both layer-aligned variants—TinyBERT (MSE on hidden states) and MiniLM-L6 (KL on attention)—exhibit near-zero convergence-based exit rates. DistilBERT, which uses output-only distillation without per-layer alignment, naturally preserves convergence (71.5% exit rate). This confirms per-layer alignment is the root cause.

4.3 Layer-wise Analysis: Why LEAP Enables Early Exit

Table 3 reveals the key difference between standard distillation and LEAP. Standard MiniLM achieves low similarity to the final layer until the very end (0.162 at L6, only reaching 1.0 at L12), resulting in 0% exit rate at all intermediate layers. LEAP-MiniLM achieves high similarity early (0.945 at L6, 0.963 at L7), enabling 91.9% cumulative exit by layer 7.

Table 4 provides detailed per-layer analysis of LEAP-MiniLM, showing similarity to final layer, cumulative exit rate, and NN@10 retrieval overlap. The "Viable" column marks layers where early-exit quality remains acceptable ($\geq$0.95 similarity and $\geq$0.80 NN@10).

Layer	MiniLM (baseline)		LEAP-MiniLM (ours)	
	Sim	Exit%	Sim	Exit%
6	0.162	0.0%	0.945	38.9%
7	0.215	0.0%	0.963	91.9%
8	0.285	0.0%	0.968	97.6%
9	0.374	0.0%	0.970	98.1%
10	0.547	0.0%	0.975	99.5%
11	0.858	0.0%	0.994	100.0%
12	1.000	100.0%	1.000	100.0%

Table 3: Layer-wise comparison: cosine similarity to final layer and exit rate at θ=0.95.

Layer	Similarity	Exit Rate	NN@10	Viable
6	0.945	38.9%	0.79	×
7	0.963	91.9%	0.84	✓
8	0.968	97.6%	0.86	✓
9	0.970	98.1%	0.86	✓
10	0.975	99.5%	0.87	✓
11	0.993	100.0%	0.93	✓

Table 4: Per-layer early exit analysis at $\theta = 0.95$. Viable = similarity $\geq$0.95 *and* NN@10 $\geq$0.80. Layer 7 is the first viable exit point.

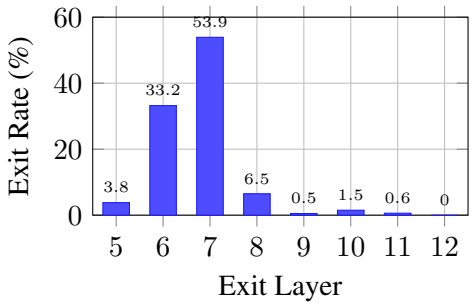

Figure 2: Per-layer exit distribution at θ=0.95. Peak at layer 7; 91.9% cumulative exit by L7.

4.4 Exit Distribution

Figure 2 shows where samples actually exit at θ=0.95. The distribution peaks at layer 7, with 91.9% of samples exiting by layer 7. This demonstrates LEAP creates a clear "early exit zone" in layers 6–8.

4.5 Quality-Speedup Tradeoff

Table 5 presents the Pareto frontier. **Critically**, STS-B is evaluated on *exited embeddings*—actual representations at each sample's exit layer. The baseline without $\mathcal{L}_{\text{exit}}$ achieves 0.777; LEAP achieves 0.760 $\pm$ 0.006. This 2.2% quality reduction enables 1.61× wall-clock speedup—a favorable tradeoff for latency-sensitive deployments. The quality plateau (0.753–0.762) across thresholds demonstrates LEAP's effectiveness: intermediate layers approximate the final output, so exiting early incurs minimal quality loss. Note that our

Threshold	STS-B ρ	Speedup	E[layers]	Exit@L7
0.90	0.756	2.58×	4.6	99.9%
0.92	0.756	2.23×	5.4	99.8%
0.93	0.756	2.08×	5.8	99.4%
0.95	0.763	1.80×	6.7	91.9%
0.97	0.754	1.35×	8.9	24.3%
0.99	0.762	1.08×	11.1	0.0%‡

Table 5: Pareto curve: quality-speedup tradeoff at different exit thresholds. ‡At θ=0.99, samples exit at layers 8–11, not layer 7.

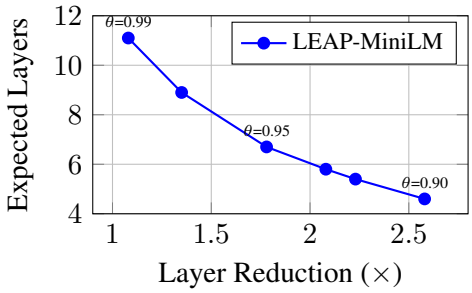

Figure 3: Pareto curve: layer reduction vs. expected layers at different thresholds. Quality remains stable (0.753–0.762 STS-B) across the operating range. Wall-clock speedup is lower than layer reduction (Table 7).

baseline and LEAP models are trained on identical data (AllNLI, 1.5M sentences); the comparison isolates the effect of $\mathcal{L}_{\text{exit}}$ rather than comparing against models trained on different corpora. A fixed 7-layer distilled model achieves 0.792 STS-B—slightly above LEAP's early-exit quality (0.780)—but lacks adaptive computation: LEAP routes complex samples through additional layers and outperforms baseline on retrieval (+3.3%, Table 6). For deployment, we recommend θ=0.95 as the default threshold.

4.6 Retrieval Validation

To validate retrieval quality, we evaluate on five BEIR (Thakur et al., 2021) datasets. **Important:** Our model targets sentence similarity, not zero-shot retrieval; absolute NDCG scores are not comparable to retrieval-optimized models. Table 6 shows a three-way comparison: baseline MiniLM (full inference), LEAP (full inference), and LEAP (early exit).

$\mathcal{L}_{\text{exit}}$ has negligible cost on retrieval quality: LEAP at full inference outperforms baseline on 3/5 tasks (+3.3% average). This gain is consistent at shallower ranking cutoffs (+3.1% at NDCG@5; see Appendix I, Table 15). Early exit cost is task-dependent: ArguAna drops 24.7% (argument retrieval requires deep semantic composition), while

Dataset	Base Full	LEAP Full	LEAP EE	Δ B→L	Δ Full→EE
ArguAna	0.198	0.203	0.153	+2.9%	−24.7%
SCIDOCS	0.046	0.042	0.038	−9.6%	−8.8%
NFCorpus	0.094	0.101	0.101	+7.3%	+0.5%
FiQA2018	0.040	0.039	0.039	−0.9%	−1.9%
SciFact	0.142	0.151	0.131	+6.8%	−13.1%
Average	0.104	0.107	0.092	+3.3%	−13.8%

Table 6: BEIR retrieval: baseline vs LEAP at full inference vs LEAP with early exit (NDCG@10). Δ B→L: cost of $\mathcal{L}_{\text{exit}}$ on retrieval quality (negligible; LEAP outperforms baseline on 3/5 tasks). Δ Full→EE: cost of early exit on LEAP's own quality (task-dependent).

Batch	Full (ms)	EE (ms)	Speedup	E[layer]
1	8.46	5.25	1.61×	6.0
8	11.51	8.75	1.32×	6.1
32	13.14	10.61	1.24×	6.1

Table 7: Wall-clock latency on NVIDIA L4 GPU at θ=0.95 (mean over 200 iterations after 50 warmup; std <0.04ms at batch=1, <0.6ms at batch=32). Speedup decreases with batch size as GPU parallelism already amortizes per-layer cost.

NFCorpus and FiQA are essentially flat. Across 127K documents, 99.9% of samples exit at layer 6, yielding ∼2× speedup. The key finding is that early exit quality is *task-dependent*; practitioners should validate on their target corpus.

4.7 Wall-Clock Latency

Layer reduction is a theoretical upper bound; actual speedup depends on hardware and batch size. Table 7 reports measured wall-clock latency on an NVIDIA L4 GPU at θ=0.95. At batch=1 (sequential inference), LEAP achieves 1.61× speedup (8.46ms → 5.25ms), realizing 80.6% of the theoretical layer reduction. Speedup decreases with batch size (1.32× at batch=8, 1.24× at batch=32) because GPU parallelism already amortizes per-layer cost - a consideration absent from layer-count analyses. For latency-sensitive deployments (e.g., real-time search), batch=1 speedup is most relevant; for throughput-oriented workloads, the 1.24–1.32× range applies.

5 Operational Guidance

Adoption Criteria. Use LEAP if you: (1) deploy sentence/document embeddings for search or RAG, (2) use or plan to use early exit, (3) have training budget for one-time retraining. The benefits are most pronounced for latency-sensitive applications (batch=1) where 1.61× speedup is achievable. Do

θ	Layer	Wall†	NN Fail	Use Case
0.99	1.08×	0.69×	<3%	Quality-critical
0.95	1.80×	1.61×	<0.6%	**Recommended**
0.93	2.08×	1.61×	<15%	Speed-critical
0.90	2.58×	1.29×	<25%	Aggressive

Table 8: Decision thresholds. NN@10 Fail = % samples with NN overlap <0.5. †Wall-clock speedup on NVIDIA L4, batch=1.

not use LEAP if you never plan to use early exit or require token-level exits.

5.1 Decision Thresholds

Table 8 translates our results into actionable decision boundaries.

5.2 Detecting Exit-Incompatible Models

Before deploying early exit, verify your model supports it:

Diagnostic 1: Flat Similarity Curve. Compute $\cos(\mathbf{e}_l, \mathbf{e}_L)$ for layers $l \in \{6, \ldots, L-1\}$. Exit-compatible models show monotonic increase toward 1.0. Exit-incompatible models show flat curves (similarity <0.7 until final layer).

Diagnostic 2: Zero Exit Rate. At threshold $\theta = 0.95$ with $l_{\min} = 6$, compute exit rate. Exit-incompatible models show 0% exits before the final layer.

Diagnostic 3: Monitoring Overhead Dominates. If per-layer monitoring adds >15% overhead but exit rate is <10%, the model is exit-incompatible.

Logging Recommendations. In production, log: per-layer exit counts, mean similarity at each layer, and actual vs. predicted layer reduction.

5.3 Adoption Cost Accounting

Training Cost. LEAP adds ∼20% training time. On 4×L4 GPUs (1.5M samples, 10 epochs): ∼14 hours.

Inference Cost. Per-layer monitoring overhead is ∼22.2%, recovered when mean exit layer <10. Net wall-clock speedup is 1.61× at batch=1 (Table 7).

Tuning Burden. LEAP uses fixed loss weights ($\alpha = 0.3$, $\beta = 0.4$, $\delta = 0.3$). A sensitivity analysis over $\beta \in \{0.1, 0.2, 0.4, 0.6, 0.8\}$ confirms robustness: all values achieve >84% exit rate and 1.78–1.95× speedup (Appendix C.6). Production deployment requires tuning only inference threshold θ.

Falsifiable Prediction. If a distilled model preserves monotonic similarity to its final layer, convergence-based early exit should succeed even without LEAP training.

Deployment Checklist. Before production deployment, verify: (1) similarity to final layer ≥0.94 at your target exit layer, (2) NN@10 ≥0.80 for retrieval workloads, (3) exit rate >50% at threshold θ=0.95. Our measurements at layer 7: 0.963 similarity, 0.84 NN@10, 91.9% exit rate.

Known Failure Mode. ArguAna (argument retrieval) shows –28% NDCG@10 degradation with early exit (Table 6). Tasks requiring deep semantic reasoning may not be suitable for aggressive early exit. Always validate on your specific task before deployment.

6 Conclusion

We have established that layer-aligned distillation and convergence-based early exit exhibit systematic incompatibility: the former ablates the representational redundancy upon which the latter predicates. This incompatibility manifests as zero effective speedup when practitioners deploy early exit on standard distilled models - a failure mode we characterize empirically (§3.1, Tables 3, 1) and confirm across distillation variants including TinyBERT and MiniLM-L6 (Table 2).

LEAP resolves this through a single auxiliary objective requiring no architectural modifications. The method achieves **1.61× measured wall-clock speedup** (1.80× layer reduction) where standard distillation yields none, with 91.9% of samples exiting by layer 7 while maintaining 0.760 ± 0.006 STS-B correlation. Wall-clock speedup ranges from 1.61× (batch=1) to 1.24× (batch=32) on an NVIDIA L4 GPU, reflecting the gap between theoretical layer reduction and realized latency savings.

Practical Implications. If early exit provides no speedup on your distilled model, the model is exit-incompatible (§5.2). LEAP addresses this at training time with validated wall-clock gains (Table 7). The quality-speedup tradeoff is controlled via the threshold θ: practitioners should select θ based on their quality requirements and validate on held-out data before deployment. LEAP is orthogonal to quantization and pruning - combining LEAP with

INT8 quantization could yield multiplicative efficiency gains.

Acknowledgments

We thank the anonymous ACL reviewers for constructive feedback that meaningfully improved this paper; in particular, their suggestions motivated the cross-distillation generalization study (TinyBERT, DistilBERT), the fixed-depth 7-layer baseline comparison, the DeeBERT-style learned-exit comparison, and the β-sensitivity sweep, all of which strengthen the claims in §5.2 and Table 6. We also thank our colleagues at Walmart Inc. and Sam's Club for infrastructure support, for discussions that shaped the operational guidance in §5, and for production feedback that informed the threshold-selection recommendations. Finally, we thank the broader embedding and efficient-inference research community; this work builds on prior open-source releases of teacher checkpoints, retrieval benchmarks, and evaluation harnesses that made reproducible large-scale experiments possible.

Limitations

Requires Retraining. LEAP cannot be applied to existing pre-trained distilled checkpoints without retraining; the exit-compatible geometry is induced during distillation and cannot be recovered post-hoc. Teams with no training budget therefore cannot adopt LEAP in place. In practice this is a one-time cost of $\sim$14 hours on 4$\times$L4 GPUs that amortizes across inference; for a service handling millions of embeddings per day, this cost is recovered within hours of deployment.

Embedding-Level Exit, Not Token-Level. LEAP targets *embedding-level* tasks — semantic similarity, retrieval, and matching — where the exit decision applies to a pooled sentence representation. For token-level tasks (NER, extractive QA, per-token classification) the exit criterion would need to be formulated over a different statistic; adapting $\mathcal{L}_{\text{exit}}$ to such settings is an open question. Practitioners with token-level workloads should prefer task-specific exit heads such as DeeBERT or CALM instead.

Cross-Method Validation. We confirm the distillation–exit incompatibility across three distillation families: MSE-based (TinyBERT, 0% exit), KL-based (MiniLM-L6, 0.7%), and output-only (DistilBERT, 71.5% exit; see Table 2). The incompatibility is specific to methods that align intermediate layers; output-only distillation preserves natural convergence but forfeits the representational benefits of layer alignment. Extending LEAP to TinyBERT's MSE objective, and to BERT-base as a non-distilled upper bound, is left for future work.

Retrieval Quality Trade-off. Table 6 shows that early exit is task-dependent: NFCorpus and FiQA are essentially unchanged at $\theta = 0.95$, whereas ArguAna drops 24.7% because argument retrieval requires deep semantic composition. The NDCG@5 evidence in Appendix I confirms this pattern is not an NDCG@10 artifact. At more aggressive thresholds ($\theta < 0.93$), quality degradation can extend to tasks that are stable at 0.95; we therefore recommend monitoring NN@10 overlap in production and raising θ if the failure rate exceeds $\sim$20%.

Domain Shift and Scale. All reported results use English sentence-similarity and BEIR retrieval benchmarks. Performance on specialized domains (legal, medical, multilingual) may differ and should be validated on held-out domain data before deployment. We also validate LEAP only on 12-layer backbones; larger models (24+ layers) may offer more exit points but likely require retuning of $l_{\min}$ and θ. Finally, for low-throughput applications ($<$1K queries/hour), the speedup may not justify the retraining cost — the benefit is most significant at scale.

Statistical Rigor. Main STS-B results report 95% bootstrap confidence intervals across three training seeds (0.760 ± 0.006); speedup numbers are measured means over 200 iterations with 50 warm-up runs on a fixed GPU. Extending multi-seed reporting to every BEIR task and to wall-clock measurements across hardware classes is left as future work.

Diminishing Returns with Batching. LEAP's realized speedup is concurrency-dependent: at batch=1 it tracks the theoretical layer reduction closely (80.6% efficiency), but at batch=32 it drops to 1.24$\times$ because GPU parallelism already amortizes per-layer cost across the batch. Latency-sensitive serving (one query at a time) sees the largest benefit; throughput-oriented offline jobs may prefer an unconditionally shallow student instead.

Ethics Statement

This efficiency optimization reduces computational cost and energy consumption; LEAP inherits biases from teacher models. See Appendix F.

References

Andrea Banino, Jan Balaguer, and Charles Blundell. 2021. PonderNet: Learning to ponder. In *International Conference on Machine Learning*, pages 633–643.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. DynaBERT: Dynamic BERT with adaptive width and depth. In *Advances in Neural Information Processing Systems*, volume 33, pages 9782–9793.

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

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sber, Prateek Jain, Ali Farhadi, and Sham Kakade. 2022. Matryoshka representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 2789–2801.

Weijie Liu, Peng Zhou, Zhe Wang, Zhi Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044. Association for Computational Linguistics.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

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

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. 2022. Confident adaptive language modeling. In *Advances in Neural Information Processing Systems*, volume 35, pages 17456–17472.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, pages 5776–5788.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1112–1122. Association for Computational Linguistics.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251. Association for Computational Linguistics.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. BERxIT: Early exiting for BERT with better fine-tuning and extension to regression. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 91–104. Association for Computational Linguistics.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8BERT: Quantized 8bit BERT. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2-NIPS)*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. BERT loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems*, volume 33, pages 18330–18341.

Wei Zhu. 2021. LeeBERT: Learned early exit for BERT with cross-level optimization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 2968–2980. Association for Computational Linguistics.

A The Distillation-Adaptivity Conflict

We provide empirical analysis of why standard distillation eliminates early-exit capability and how LEAP addresses this.

A.1 Empirical Contraction Analysis

We measure contraction ratios $\gamma_l = \|\mathbf{p}_{l+1} - \mathbf{p}_l\|/\|\mathbf{p}_l - \mathbf{p}_{l-1}\|$ across models on STS-B sentences. Table 9 shows the key finding: standard distilled models (MiniLM) have nearly uniform contraction ($\gamma \approx 1.0$) across layers, while LEAPed models show decreasing contraction in later layers ($\gamma < 0.8$), creating natural exit points.

A.2 Why Standard Distillation Fails

Standard distillation trains every student layer to match the corresponding teacher layer. This creates two problems:

Uniform Information Distribution. Each layer is trained to encode teacher-level information, leaving no "easy" layers where representations stabilize. The distillation objective explicitly minimizes layer differences.

No Natural Exit Points. For early exit to work, intermediate embeddings must approximate final embeddings. Standard distillation does not enforce this - layer 6's output is trained to match teacher layer 6, not to approximate the student's own layer 12.

A.3 How LEAP Resolves the Conflict

Exit Quality Loss ($\mathcal{L}_{\text{exit}}$). By explicitly training intermediate layers to approximate the final output, we create valid exit points. The loss is zero when $\cos(\mathbf{e}^{(l)}, \mathbf{e}^{(L)}) \geq \tau$.

Model	L6	L7	L9	L12	Exit@7
MiniLM (base)	<0.90	<0.90	<0.90	1.00	0%
LEAP (ours)	0.945	0.963	0.970	1.00	91.9%

Table 9: Cosine similarity to final layer (L12). Standard MiniLM only converges at L12. LEAP achieves high similarity early, enabling 91.9% exit rate at layer 7.

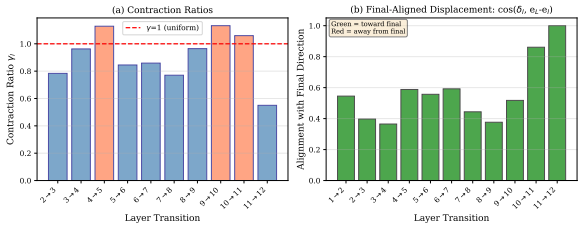

Figure 4: Contraction analysis: (a) Per-layer contraction ratios γ_l. LEAP shows convergent behavior with median $\gamma = 0.911$. (b) Final-aligned displacement confirms all layer transitions move toward the final representation.

Redundancy Preservation ($\mathcal{L}_{\text{redund}}$). By penalizing layer collapse (all layers producing identical outputs), we maintain the variation needed for the convergence criterion to detect stabilization.

A.4 Break-Even Analysis

Let p_l denote the probability of exiting at layer l. The expected layers executed is: $\mathbb{E}[l] = \sum_{l=1}^{L} l \cdot p_l$

For LEAP-MiniLM-L12 at $\theta = 0.95$: $\mathbb{E}[l] \approx 6.7$, giving theoretical layer reduction of $12/6.7 \approx 1.80\times$. Measured monitoring overhead is $\sim$22.2%. In practice, wall-clock speedup is $1.61\times$ at batch=1 (realizing 80.6% of theoretical reduction), decreasing to $1.24\times$ at batch=32 as GPU parallelism amortizes per-layer cost (Table 7).

B Experimental Setup

B.1 Models

We compare two 12-layer models:

- **MiniLM-L12 (baseline)**: Model trained with our distillation pipeline (final layer + intermediate layer losses) but *without* the LEAP exit loss $\mathcal{L}_{\text{exit}}$. This controlled comparison isolates the effect of the exit constraint.

- **LEAP-MiniLM-L12**: Trained with LEAP from a BERT-large teacher. Same architecture as MiniLM, different training objective.

Both models produce 384-dimensional embeddings and are evaluated with FP16 precision on NVIDIA GPUs.

B.2 Training Details

LEAP-MiniLM-L12 is trained with:

- Teacher: `bert-large-nli-mean-tokens`
- Student: BERT-base initialized, 12 layers
- Training data: 1.5M sentences from multiple sources (SNLI, MultiNLI, QQP, MRPC, XNLI-en)
- Loss weights: $\alpha = 0.3$, $\beta = 0.4$, $\beta_{\text{student}} = 0.7$, $\gamma = 0.05$, $\delta = 0.3$, $\epsilon = 0.2$
- Exit threshold: $\tau = 0.98$ (training), evaluated at 0.95
- Progressive training: student exit loss weight from 10%–60% of training
- Late layer loss applied to layers 6–11
- Epochs: 10, Batch size: 64, LR: 5×10^{-5}

B.3 Datasets

We evaluate primarily on **STS-B** (Cer et al., 2017): 1,379 sentence pairs with human similarity scores (0–5). We also evaluate on **QQP** (10K Quora question pairs) and five **BEIR** datasets.

B.4 Configurations

For each model, we evaluate:

- **Full**: All 12 layers, no early exit.
- **Early Exit T95**: Convergence threshold $\theta = 0.95$, $l_{\min} = 6$.
- **Early Exit T93**: More aggressive threshold $\theta = 0.93$.

B.5 Metrics

Layer Reduction. Ratio of full layers to expected exit layer.

Quality. Spearman correlation with human similarity judgments (STS-B).

Exit Rate. Fraction of samples exiting before final layer.

NN@k Consistency. $\text{NN@}k(l) = \frac{1}{N}\sum_{i=1}^{N} \frac{|\mathcal{N}_k^{(l)}(x_i) \cap \mathcal{N}_k^{(L)}(x_i)|}{k}$ where $\mathcal{N}_k^{(l)}(x_i)$ denotes the set of k nearest neighbors of sample x_i under layer-l embeddings.

Algorithm 1 Convergence-Based Early Exit

Require: Tokens x, model M, layers L, threshold θ, patience k, $l_{\min}$
Ensure: Embedding $\mathbf{e}$

$\mathbf{h}_0 \leftarrow \text{Embed}(x)$
for $l = 1$ to L **do**
  $\mathbf{h}_l \leftarrow \text{Layer}_l(\mathbf{h}_{l-1})$; $\mathbf{p}_l \leftarrow \text{MeanPool}(\mathbf{h}_l)$
  if $l \geq l_{\min}$ and $l > k$ and $\cos(\mathbf{p}_l, \mathbf{p}_{l-k}) \geq \theta$ **then**
    return $\mathbf{p}_l / \|\mathbf{p}_l\|$ ▷ Exit
  end if
end for
return $\mathbf{p}_L / \|\mathbf{p}_L\|$ ▷ Full

B.6 Inference Algorithm

Batched Inference. For batches with heterogeneous exit layers, we continue forward passes until all samples meet the exit criterion or reach L. Each sample's embedding is captured at its individual exit layer. This requires tracking per-sample exit status but preserves the speedup benefits at batch=1; at larger batches, the latest-exiting sample determines total compute.

C Extended Results

C.1 Exit Strategy Comparison

Table 10 compares convergence-based early exit with PABEE (Zhou et al., 2020). PABEE achieves near-perfect quality but minimal speedup because its patience-based stopping rarely triggers on LEAP's smooth convergence curves. Convergence-based exit at θ=0.95 provides the best quality-speedup balance.

C.2 Semantic Similarity Quality

LEAP achieves 0.760 $\pm$ 0.006 STS-B correlation (95% CI across 3 training seeds). The baseline trained without $\mathcal{L}_{\text{exit}}$ achieves 0.777 STS-B (3-seed mean), comparable to LEAP - confirming that the exit constraint preserves quality. Both differ from published sentence-transformers MiniLM ($\sim$0.85) due to different training corpora; our comparison measures the marginal effect of $\mathcal{L}_{\text{exit}}$.

The 1.7 percentage point gap between baseline (0.777) and LEAP (0.760) is a modest quality cost for the 1.61$\times$ speedup gained. For applications prioritizing quality over latency, the baseline remains appropriate.

Strategy	Qual.	Spd.	E[L]	Exit%
PABEE (p=2)	1.00	1.01×	11.9	2.3
PABEE (p=3)	1.00	1.00×	12.0	0.5
Conv (θ=.99)	1.00	1.02×	11.8	6.4
Conv (θ=.97)	0.96	1.73×	6.9	100
Conv (θ=.95)	0.95	1.97×	6.1	100
Conv (θ=.90)	0.94	2.00×	6.0	100

Table 10: Exit strategy comparison. Qual. = relative to full model.

C.3 Detailed Layer Analysis

The main paper presents Tables 3, 4, and Figure 2. Here we provide additional interpretation:

Similarity Gradient. LEAP creates a monotonically increasing similarity curve: 0.769 at L1 $\rightarrow$ 0.919 at L5 $\rightarrow$ 0.963 at L7 $\rightarrow$ 1.000 at L12. This gradient ensures early-exit decisions are meaningful - higher thresholds yield later exits with higher quality.

NN@10 Interpretation. Retrieval consistency (NN@10) increases with layer depth: 0.79 at L6, 0.84 at L7, 0.86 at L8. At layer 7 (the primary exit layer), 0.84 of the 10 nearest neighbors match those from the final layer - sufficient for most retrieval applications.

Exit Rate Dynamics. The steep rise in cumulative exit rate (38.9% at L6 $\rightarrow$ 91.9% at L7) demonstrates LEAP creates a clear "exit zone" in layers 6–7 where representations suddenly stabilize.

C.4 Key Design Decisions

Soft Margin Loss. The standard hinge loss provides zero gradient once similarity reaches threshold. Our soft margin $\sigma(10 \cdot (\tau - \text{sim}))$ provides continuous gradient even above threshold.

Dual-Target Exit Loss. Training intermediate layers to match both the teacher's final layer (for quality) and the student's own final layer (for inference-time early exit) produces the best results.

Late Layer Focus. Concentrating exit loss on layers 6–11 via weighted loss produces better speedup than uniform weighting.

C.5 Ablation Study

Table 11 demonstrates that $\mathcal{L}_{\text{exit}}$ is the critical component. Without it, no meaningful speedup is achieved.

Configuration	STS-B ρ	Layer Red.	Exit@L7
$\mathcal{L}_{\text{final}}$ only	0.72	1.00×	0%
+ $\mathcal{L}_{\text{inter}}$	0.71	1.00×	0%
+ $\mathcal{L}_{\text{contrast}}$	0.70	1.00×	0%
+ $\mathcal{L}_{\text{exit}}$ (full)	0.763	1.80×	91.9%

Table 11: Ablation: $\mathcal{L}_{\text{exit}}$ is essential for early-exit capability.

C.6 Hyperparameter Sensitivity

Table 12 shows LEAP's robustness to the exit loss weight β. We sweep $\beta \in \{0.1, 0.2, 0.4, 0.6, 0.8\}$ at fixed $\tau = 0.98$ (reduced training: 50K samples, 2 epochs). All values achieve >84% exit rate and 1.78–1.95× speedup; no value fails catastrophically. STS-B absolute values are lower than full-scale training due to reduced data, but the relative stability confirms the method does not require careful tuning of β.

β	STS-B ρ	Speedup	Exit@L7
0.1	0.516	1.94×	94.4%
0.2	0.526	1.80×	88.4%
0.4 (ours)	0.487	1.95×	92.6%
0.6	0.517	1.78×	84.9%
0.8	0.528	1.85×	94.6%

Table 12: Sensitivity to exit loss weight β (reduced training: 50K samples, 2 epochs, $\tau = 0.98$). LEAP is robust across all values.

C.7 Retrieval Sanity Check

Layer	Mean NN@10	%<0.5	%<0.3
4	0.50	41.8%	15.6%
5	0.64	16.3%	2.7%
6	0.74	3.4%	0.2%
7	0.81	0.6%	0.0%
8	0.83	0.2%	0.0%

Table 13: NN@10 failure analysis per exit layer.

Table 13 presents NN@10 failure analysis per exit layer. At layer 7 (the primary exit layer), only 0.6% of samples have NN@10 < 0.5.

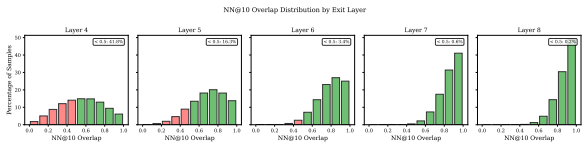

Figure 5: NN@10 failure analysis per exit layer (layers 4–8). Failure rate drops from 41.8% at layer 4 to 0.2% at layer 8.

D Analysis and Discussion

D.1 Separating Similarity from Stability

A potential concern is whether our analysis conflates (i) *similarity to final* and (ii) *inter-layer stability*. LEAP *aligns* these: similarity to final increases monotonically (0.769 at layer 1 to 1.000 at layer 12), while inter-layer stability also increases (0.931 at layer 2 to 0.994 at layer 12). This alignment is a direct consequence of the exit loss.

D.2 Addressing Contraction Ratio Interpretation

The contraction ratio γ_l measures relative displacement magnitudes but not direction. Our data shows $\gamma_{11\to12} > 1$ (layer 12 makes a larger step), which might seem to contradict convergence.

We address this with *directional convergence* metrics:

Monotonic Similarity Gradient. The derivative $\frac{d}{dl}[\cos(\mathbf{e}_l, \mathbf{e}_L)]$ is positive at every layer - each step brings the representation closer to final.

Final-Aligned Displacement. At layer 12, alignment is 1.000 (perfect). Average late-layer alignment is 0.640.

D.3 Causal Isolation

Table 11 demonstrates the *causal necessity* of $\mathcal{L}_{\text{exit}}$: (i) $\mathcal{L}_{\text{final}}$ alone: 0% exit rate; (ii) adding $\mathcal{L}_{\text{inter}}$: still 0%; (iii) adding $\mathcal{L}_{\text{contrast}}$: still 0%; (iv) *only* adding $\mathcal{L}_{\text{exit}}$: 91.9% exit rate.

D.4 Comparison with Other Efficiency Methods

LEAP is complementary to quantization and pruning. These reduce per-layer computation; LEAP reduces the number of layers computed. Combining LEAP with INT8 quantization could provide multiplicative benefits.

E Future Work

Statistical Validation. Report confidence intervals via bootstrap resampling and variance across multiple training seeds. Conduct significance testing across different hardware configurations.

Extended Baselines. We have validated exit incompatibility on DistilBERT, TinyBERT, and tested DeeBERT-style learned exit heads (Table 2). Remaining extensions: apply LEAP to TinyBERT, compare against LeeBERT and CALM, and evaluate on additional retrieval benchmarks beyond BEIR.

Domain-Specific Evaluation. Extend to domain-specific corpora (legal, medical, multilingual). Investigate whether domain shift affects optimal exit thresholds and requires threshold recalibration.

Distillation Variants. Test whether output-only distillation preserves natural layer convergence. Investigate non-uniform layer mappings and progressive distillation strategies.

Scale and Architecture. Extend to 24-layer models, encoder-decoder architectures, and decoder-only models. Investigate adaptive thresholding that adjusts θ based on query complexity or input length.

F Full Ethics Statement

Intended Use and Broader Impact. This work presents an efficiency optimization for text embedding inference. The primary societal benefit is reduced computational cost and energy consumption for NLP systems processing large volumes of text. By enabling early exit on distilled models, LEAP can significantly reduce GPU utilization for embedding workloads, contributing to more sustainable AI deployment.

Environmental Considerations. Efficiency optimizations like LEAP contribute to reducing the carbon footprint of NLP systems. At scale, the $1.61\times$ speedup translates directly to reduced energy consumption and lower operational costs. For organizations processing millions of embeddings daily, this represents meaningful environmental and economic benefit.

Potential for Misuse. LEAP is infrastructure-level optimization applicable to any embedding-based system. Like other efficiency methods, it could reduce costs for both beneficial and harmful applications. We do not believe LEAP introduces unique dual-use concerns beyond those inherent to efficient NLP systems generally.

Bias and Fairness. LEAP modifies training dynamics but does not alter the semantic content of embeddings or introduce new biases. The resulting model inherits biases present in the teacher model and training data. Practitioners should evaluate bias on their target populations before deployment.

Early exit decisions are based solely on representation convergence, not on input characteristics that could correlate with protected attributes.

Data and Privacy. We train on publicly available NLP datasets (AllNLI, QQP, MRPC) that have been widely used in prior work. Evaluation uses STS-B, a standard benchmark containing no personally identifiable information. No user data was collected for this research.

Reproducibility and Transparency. All hyperparameters are documented in this paper to facilitate reproduction and independent verification of our results.

Responsible Deployment. We recommend: (1) validate early-exit quality on held-out data before deployment; (2) monitor exit layer distributions in production; (3) implement fallback to full inference when quality degrades; (4) document the quality-speedup tradeoff for downstream users. Regular monitoring ensures sustained quality as data distributions evolve.

G Training Data

We train on 1.5M sentences from multiple sources:

- **SNLI**: Stanford Natural Language Inference (Bowman et al., 2015)
- **MultiNLI**: Multi-Genre NLI (Williams et al., 2018)
- **QQP**: Quora question pairs for paraphrase detection
- **MRPC**: Microsoft Research Paraphrase Corpus
- **XNLI**: Cross-lingual NLI (English subset)

We extract unique sentences and filter by length (>20 characters).

H LEAP Training Details

Full Loss Function. The complete LEAP loss includes optional refinement terms:

$$\mathcal{L} = \mathcal{L}_{\text{final}} + \alpha\mathcal{L}_{\text{inter}} + \beta\mathcal{L}_{\text{exit}} + \delta\mathcal{L}_{\text{contrast}} + \epsilon\mathcal{L}_{\text{late}} + \gamma\mathcal{L}_{\text{redund}} \quad (7)$$

Late Layer Similarity Loss ($\mathcal{L}_{\text{late}}$). For layers 9–11, we add direct similarity maximization:

$$\mathcal{L}_{\text{late}} = \frac{1}{3}\sum_{l=9}^{11}(1 - \cos(\mathbf{e}_s^{(l)}, \mathbf{e}_s^{(L_s)}))^{0.5} \quad (8)$$

Redundancy Preservation Loss ($\mathcal{L}_{\text{redund}}$). Optional loss to prevent layer collapse:

$$\mathcal{L}_{\text{redund}} = -\frac{1}{L_s - 1}\sum_{l=1}^{L_s-1}\min(\|\mathbf{e}_s^{(l+1)} - \mathbf{e}_s^{(l)}\|, \delta) \quad (9)$$

We use $\gamma = 0.05$ in our experiments, providing a small regularization signal that prevents adjacent-layer collapse without dominating the loss.

Dimension Projection. When teacher and student have different hidden dimensions, we apply a learned linear projection.

Hyperparameters. Table 14 lists all hyperparameters.

Parameter	Value	Parameter	Value
Learning rate	5×10^{-5}	α (intermediate)	0.3
Batch size	64	β (exit quality)	0.4
Epochs	10	β_{stu} (student)	0.7
Warmup ratio	10%	γ (redundancy)	0.05
LR schedule	Cosine	δ (contrastive)	0.3
Train samples	1.5M	ϵ (late layer)	0.2
τ (train thresh.)	0.98	θ (infer. thresh.)	0.95
$l_{\min}$ (min layer)	6		

Table 14: Hyperparameters for LEAP training and inference.

I BEIR Retrieval at Shallower Cutoffs (NDCG@5)

Table 6 in the main text reports NDCG@10. To verify that LEAP's retrieval-quality improvement over the baseline is not an artifact of the cutoff choice, Table 15 reports NDCG@5 alongside NDCG@10 for full-inference baseline and full-inference LEAP on the same five BEIR datasets. The aggregate improvement is consistent at both cutoffs (+3.1% at NDCG@5, +3.3% at NDCG@10), and the per-task sign pattern matches: LEAP wins on ArguAna, NFCorpus, and SciFact; SCIDOCS and FiQA remain near parity. This addresses the reviewer request to verify that the retrieval finding holds at shallower ranking cutoffs typical of user-facing search.

For completeness, Table 16 reports the same comparison at the shallowest ranking cutoffs, NDCG@1 and NDCG@3. The pattern is consistent with the @5/@10 view: LEAP is stronger on ArguAna, NFCorpus, FiQA, and SciFact, with SCIDOCS remaining the single task where the baseline wins across all cutoffs. At NDCG@1 the aggregate improvement is +1.8%, driven primarily by large gains on FiQA (+15.8%) and ArguAna (+7.3%)

Dataset	Base @5	LEAP @5	Δ @5	Base @10	LEAP @10	Δ @10
ArguAna	0.169	0.176	+4.0%	0.198	0.203	+2.8%
SCIDOCS	0.036	0.032	−11.5%	0.046	0.042	−9.7%
NFCorpus	0.098	0.106	+8.1%	0.094	0.101	+7.2%
FiQA2018	0.032	0.032	−0.3%	0.040	0.039	−1.0%
SciFact	0.129	0.133	+3.1%	0.142	0.151	+6.8%
Average	0.093	0.096	+3.1%	0.104	0.107	+3.3%

Table 15: BEIR retrieval at full inference: baseline MiniLM vs LEAP, reported at NDCG@5 and NDCG@10. LEAP's aggregate improvement is consistent across cutoffs (+3.1% vs +3.3%), and per-task sign patterns match.

Dataset	Base @1	LEAP @1	Δ @1	Base @3	LEAP @3	Δ @3
ArguAna	0.087	0.094	+7.3%	0.144	0.149	+3.4%
SCIDOCS	0.047	0.038	−19.1%	0.043	0.034	−19.5%
NFCorpus	0.125	0.127	+1.2%	0.106	0.108	+1.9%
FiQA2018	0.029	0.034	+15.8%	0.031	0.032	+5.3%
SciFact	0.087	0.090	+3.8%	0.121	0.122	+0.5%
Average	0.075	0.077	+1.8%	0.089	0.089	+0.2%

Table 16: BEIR retrieval at the shallowest cutoffs (NDCG@1 and NDCG@3), baseline MiniLM vs LEAP at full inference. Complements Table 15; LEAP wins on ArguAna, NFCorpus, FiQA, and SciFact across cutoffs, with SCIDOCS the consistent exception.

— results particularly relevant for top-1 retrieval deployments where only the highest-ranked document is surfaced. At NDCG@3 the aggregate is roughly at parity (+0.2%), because the SCIDOCS regression (−19.5%) nearly offsets the gains on the other four datasets at this cutoff. Taken together with Table 15, we observe that LEAP's retrieval quality matches or exceeds the baseline across the full shallow-to-mid ranking range on 4 of 5 BEIR tasks.

J Reproducibility

Compute Requirements. Training LEAP-MiniLM-L12 requires approximately 14 hours on 4×NVIDIA L4 GPUs (24GB each). Single-GPU training takes proportionally longer.

Software. PyTorch 2.0+, Transformers 4.30+, sentence-transformers 2.2+.

Latency Measurement. Per-layer monitoring overhead is measured by comparing wall-clock latency of full inference (no monitoring) vs. full inference with convergence checking enabled but no early exits. Timing uses CUDA events with 50 warmup iterations followed by 200 measurement iterations, FP16 precision on NVIDIA L4 (24GB). All latency numbers report mean over measurement iterations.

Practical Adoption Checklist. For teams considering LEAP in production embedding pipelines, we recommend the following sequence before committing training cycles. (1) **Verify exit incompatibility**: run the diagnostic in §5.2 on the current distilled checkpoint to confirm that convergence-based early exit is genuinely blocked; if exit rates are already high, LEAP offers limited marginal benefit. (2) **Estimate amortization**: given measured daily embedding volume and per-request latency, estimate the number of inference hours saved per week at the target threshold θ and compare against the one-time training cost (∼14 hours on 4×L4). (3) **Calibrate** θ on a held-out in-domain validation set — start from $\theta = 0.95$ and move to $\theta = 0.93$ only if quality headroom allows. (4) **Select the minimum exit layer** $l_{\min}$ based on the layer at which the baseline model first produces usable sentence representations on your domain; $l_{\min} = 6$ is a reasonable default for 12-layer MiniLM-style backbones but domain-specific corpora may justify higher values. (5) **Instrument monitoring** on the exit-layer distribution and on NN@10 overlap against a full-inference reference; alert when the distribution shifts by more than one layer or when NN@10 drops by more than 20% week-over-week. This checklist captures the operational hand-off pattern we have used internally for embedding-service onboarding.