symmetry

MDPI

Article

KFD: Selective Token Filtering and Adaptive Weighting for Efficient Knowledge Distillation

Muzaffer Kaan Yuce * and Mehmet Fatih Amasyali

Department of Computer Engineering, Yildiz Technical University, 34220 İstanbul, Türkiye; amasyali@yildiz.edu.tr

* Correspondence: kaan.yuce@yildiz.edu.tr

Abstract

Knowledge distillation (KD) transfers knowledge from large language models (LLMs) to smaller or similarly sized models in order to obtain efficient yet capable systems. However, performing distillation over all tokens is computationally expensive and may weaken the transfer signal. To address this limitation, **Knowledge-Filtered Distillation (KFD)** is introduced as a selective distillation approach in which tokens are filtered according to the divergence $KL(\mathcal{M}_2 \mid \mathcal{M}_0)$ between a teacher model ($\mathcal{M}_2$) and a base model ($\mathcal{M}_0$), while the student model ($\mathcal{M}_1$) is also derived from the same base model. Only tokens whose divergence exceeds a predefined threshold are distilled. For the selected tokens, the teacher distribution is normalized over the Top-5 predictions, whereas tokens outside this case receive a label-ranking bonus. The proposed conditional Top-5/bonus target design is shown theoretically to yield a lower label-focused target error than using only Top-5 normalization or only the bonus across all tokens. In addition, the KL and cross-entropy (CE) losses are balanced through a dynamically computed batch-level coefficient α. Experiments on multiple Turkish text datasets show that KFD consistently outperforms CE-only training, achieving higher accuracy with less data and shorter training time. KFD also outperforms entropy-based token selection methods and highlights the role of student initialization in effective knowledge transfer, thereby providing an efficient and scalable distillation framework for teacher–student models of equal size.

Keywords: knowledge distillation; token-level filtering; selective knowledge transfer; Top-k filtration; KL divergence; dynamic weighting; bonus-based regularization

1. Introduction

In the field of natural language processing, large language models (LLMs) have demonstrated remarkable success since the emergence of the Transformer architecture. These models have achieved results approaching human-level performance in tasks such as text generation, question answering, summarization, and dialogue systems [1–3]. However, these achievements come with significant challenges, including high training costs, substantial memory requirements, and long inference times [4–6]. Consequently, research has focused not only on advancing model architectures but also on improving the efficiency of training and knowledge transfer methods [7].

Knowledge distillation (KD) plays a central role in these efforts. The core idea of KD is that a large teacher model transfers not only the ground-truth labels but also its output distributions to a student model. This enables the student to benefit from the richer representational knowledge of the teacher, thereby achieving stronger generalization [8–10].

Check for updates

Received: 11 March 2026
Revised: 14 April 2026
Accepted: 15 April 2026
Published: 16 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

With the widespread adoption of deep learning-based models, this approach has attracted considerable attention in both the computer vision and natural language processing domains [11–14]. Recent works further show that KD is not restricted to model compression: parameter-efficient online distillation schemes for pretrained language models [15] and KD-based prompt tuning methods [16] illustrate how knowledge transfer can be leveraged to enable more efficient adaptation and training of large language models.

Knowledge distillation methods are generally categorized into three main axes: response-based (logit/probability transfer), feature-based (intermediate representations), and relation-based (inter-layer or inter-sample relationships). Each axis offers distinct advantages for different tasks [10,14,17]. In particular, relation-based distillation has proven effective for Transformer-based models, as it can transfer information intensively through attention heads and similarity structures [11,12,17].

For language models, various approaches to knowledge distillation have been explored across different dimensions. Sequence-level distillation methods compact the model's entire output and transfer it to the student, a strategy that has been widely applied in tasks such as machine translation [18–20]. In addition, several studies have focused on transferring intermediate layer representations to the student. These methods provide a more comprehensive transfer by leveraging not only the final output distributions but also the internal knowledge captured within the teacher's hidden layers [11,21,22].

In Transformer-based language models, task-agnostic and architecture-specific knowledge transfer has been shown to be effective when distilling attention and value projections, layer positions, and relation matrices [11,17,23]. Such architecture-aligned transfers have delivered broad improvements across multiple tasks, including text classification and question answering [12,13,21].

Token-level distillation holds particular importance in the literature. The probability distributions obtained for each token in language models carry critical information about how the model processes context. Consequently, token-based transfer has been shown to substantially enhance the performance of student models. Comparative studies indicate that, relative to sentence-level distillation, token-level distillation is especially effective in cases involving ambiguity or complex contextual dependencies [24,25].

Recently, selective or filtered knowledge distillation methods have gained increasing attention. The goal of these approaches is to apply distillation only to tokens where the teacher model exhibits uncertainty or where tokens carry high informational value, rather than uniformly across all tokens. This strategy reduces training costs while maintaining accuracy [26,27]. Moreover, comprehensive investigations have been conducted on how token selection strategies influence performance across different tasks [7]. For large language models in particular, recent work has explored selectivity both at the data and token levels: Selective Reflection Distillation (SRD) curates student-compatible training instances and schedules them via curriculum learning to improve LLM distillation efficiency [28], while AdaSPEC performs selective token-level distillation for speculative decoding by focusing on positions that are easy for a reference model, thereby improving acceptance rates without sacrificing generation quality [29]. Such methods offer promising pathways for balancing efficiency and quality in the training of large language models.

In addition, distillation techniques in which teacher and student models share the same size have also been explored. Mutual learning and online distillation approaches serve not only the purpose of model compression but also contribute to more efficient learning [30,31]. Parameter-efficient online distillation for pretrained language models further demonstrates that carefully designed update rules and lightweight adaptation modules can make distributed KD practical at scale [15]. Complementary to these developments, sparse Mixture-of-Experts (MoE) student architectures have been shown to outperform

dense students under comparable parameter budgets by activating only a subset of experts per input, achieving better accuracy–efficiency trade-offs through distillation [32]. Together, these lines of work highlight that efficient, capacity-preserving distillation remains an active and evolving research direction.

Softening the teacher distribution reveals the signal of secondary classes, thereby providing the student with a richer learning target; the impact of temperature selection on model calibration and generalization has been extensively discussed [8,18]. Given the tendency of models to produce overconfident predictions, proper calibration has been reported as a decisive factor in the success of distillation [33].

While much of the existing research has focused on smaller student models, efficient, selective, and stable distillation strategies for teacher–student pairs of equal capacity remain an open challenge. Moreover, ensuring stable token-level selection criteria in long-context settings requires striking a balance between reliable threshold calibration—across different numerical precision modes (e.g., BF16/FP32)—and model performance.

In conclusion, knowledge distillation methods in the literature have evolved from classical all-token approaches to selective and filtered techniques, and even to knowledge sharing between models of equal size. This diversity highlights that knowledge transfer is not only a tool for model compression but also a critical research direction for enhancing generalization and enabling more efficient learning [7,8,18].

In this study, a novel method called *Knowledge-Filtered Distillation (KFD)* is proposed. The main objective of KFD is to make knowledge transfer between teacher and student models more selective and efficient. Unlike conventional all-token distillation approaches, KFD applies distillation only to tokens that are considered information-critical. In this way, unnecessary computation is reduced while the student model is still encouraged to preserve essential language capabilities.

To achieve this selectivity, KFD uses token-level KL-divergence [34] values between the teacher model (M_2) and the base model (M_0). Tokens that exhibit strong disagreement in the output distributions are identified as information-critical, and distillation is applied only at these positions. As a result, less informative tokens are excluded from the transfer process, while learning is concentrated on more distinctive and informative cases. KFD is not limited to token selection alone. Through online threshold calibration, it also provides a stable selection mechanism across different precision modes. By controlling both the number of sequences and the proportion of distilled tokens during training, the method balances efficiency and accuracy. In addition, Top-k selection, dynamic α, and bonus-based target refinement allow alternative teacher signals to be transferred to the student, thereby enriching the overall learning signal.

An important contribution of KFD is its applicability to teacher and student models of the same size. Accordingly, the method is relevant not only to model compression scenarios but also to more efficient training in equal-capacity teacher–student settings. In this respect, KFD extends the use of knowledge transfer beyond its classical role as a compression tool.

The effectiveness of knowledge transfer depends not only on the distillation method itself but also on the initialization of the student model. In particular, initialization strategies based on model merging can influence the early stage of transfer from teacher to student. For example, the initial state of the student model can be modified using approaches such as linear merge [35] and KAFA-Merge [36], available in the `mergekit` tool [37] (available online: https://github.com/arcee-ai/mergekit, accessed on 14 April 2026). Such variations in the student's starting point can affect the subsequent transfer behavior.

In summary, KFD provides a selective, flexible, and efficient alternative to conventional distillation strategies. The overall process is illustrated in Figure 1. The following

sections present the methodology of KFD in detail and evaluate its effectiveness through comprehensive experiments.

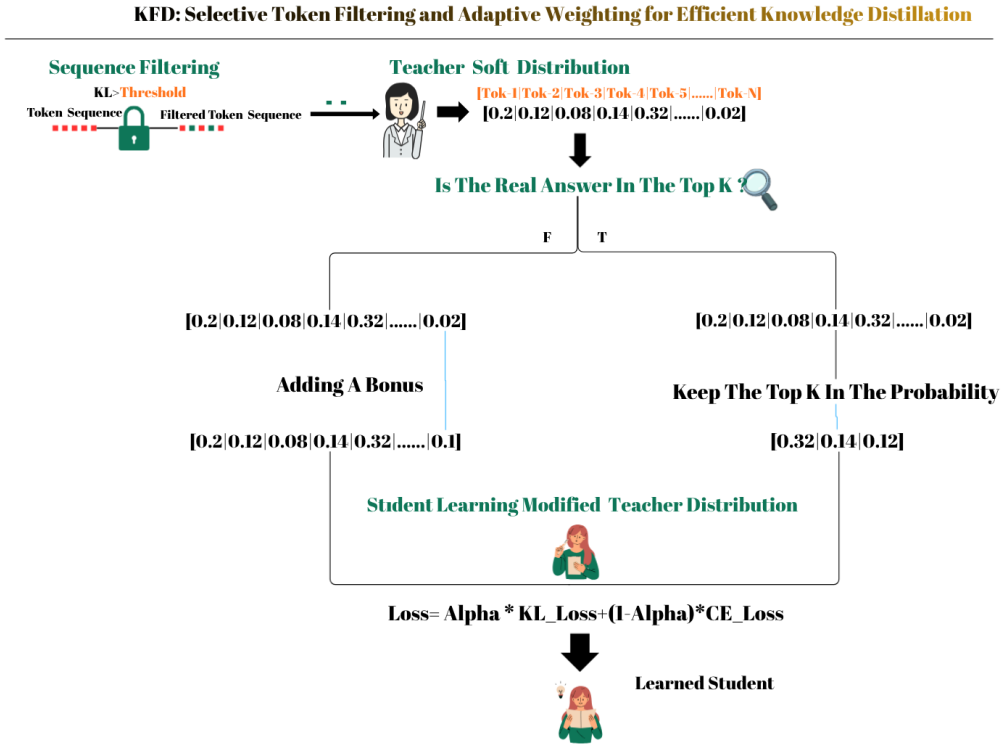


Figure 1. The sequence is filtered by KL-thresholding; the teacher target is then normalized over Top-*k* or updated with a bonus, and the student is trained with a dynamic KL + CE loss.

2. Methodology

2.1. Overview of Knowledge-Filtered Distillation (KFD)

Knowledge-Filtered Distillation (KFD) extends classical knowledge distillation by introducing a selective learning strategy. In conventional KD, the output distribution of the teacher model is transferred to the student for every token. This leads not only to high computational cost but also to unnecessary distillation on tokens that provide limited informational value. In contrast, KFD performs knowledge transfer only on tokens for which the teacher and base models produce substantially different outputs, i.e., on tokens that carry more distinctive information. As a result, training becomes more focused and more efficient.

The KFD framework consists of three main components:

- **Base model (M0):** The common starting point of the framework. Rather than directly transferring knowledge, it is used to measure the divergence between its outputs and those of the teacher model in order to identify the tokens that are most relevant for transfer.
- **Student model (M1):** Initialized from M0 and updated during training, this model represents the final output of the KFD procedure.
- **Teacher model (M2):** Also derived from M0, this model is fine-tuned on the target task and serves as the main source of knowledge for the student.

Through this three-model structure, KFD combines selective transfer with a dynamic filtering mechanism. The knowledge provided by the teacher is first filtered through the divergence measured with respect to the base model and is then transferred to the student only on the selected tokens. Consequently, the student is trained not uniformly over all

tokens, but only on those with higher informational content. This design reduces training cost and memory usage while preserving the main benefits of knowledge transfer.

Student Initialization with KAFA-Merge

The initial state of the student model is defined before the KFD optimization stage. In addition to direct initialization from the base model, an alternative initialization obtained by KAFA-Merge is also considered in order to specify how the student starting checkpoint is constructed.

KAFA-Merge is a model-merging method for large language models that constructs a single checkpoint from multiple source models through a two-stage search procedure. In the first stage, layers with similar functionality are grouped, and group-wise mixing coefficients are determined by Bayesian optimization. In the second stage, these coefficients are refined through a linear search procedure with progressively finer layer grouping. Unlike standard linear merging, which uses a single global coefficient, KAFA-Merge assigns different mixing ratios to different layer groups and produces a merged model state from the source checkpoints.

Within the present methodology, KAFA-Merge is used only to define the initial parameters of the student model before distillation. It is not part of the token filtering rule, the teacher target construction, or the KFD loss. Its role is limited to providing an alternative student initialization within the overall KFD framework.

2.2. Mathematical Definitions

To formally define the KFD method, we first introduce the basic concepts and notations. A given input sequence

$$x = (w_1, w_2, \ldots, w_T) \tag{1}$$

consists of tokens of length T. For each token, the probability distributions of three different models are computed:

- **Base model (M0):**

$$p_t^{(M0)} = P_{\theta_{M0}}(\cdot \mid x, t) \tag{2}$$

- **Student model (M1):**

$$p_t^{(M1)} = P_{\theta_{M1}}(\cdot \mid x, t) \tag{3}$$

- **Teacher model (M2):**

$$p_t^{(M2)} = P_{\theta_{M2}}(\cdot \mid x, t) \tag{4}$$

Here, $p_t^{(M)} \in \mathbb{R}^V$ represents the probability distribution over the vocabulary of size V.

For each token, the ground-truth label is denoted as $y_t \in \{1, \ldots, V\}$. During the learning process of the student model, this label is used for the classical cross-entropy (CE) loss:

$$\mathrm{CE}(y_t, p_t^{(M1)}) = -\log p_t^{(M1)}[y_t] \tag{5}$$

In addition, to guide knowledge transfer, the similarity between the teacher and base models is measured using the *Kullback–Leibler (KL) divergence*:

$$d_t = \mathrm{KL}(p_t^{(M2)} \parallel p_t^{(M0)}) = \sum_{v=1}^{V} p_t^{(M2)}[v] \log \frac{p_t^{(M2)}[v]}{p_t^{(M0)}[v]} \tag{6}$$

These definitions establish the foundation for the selection mechanism and loss function in KFD.

2.3. Formulation of the Selection Mechanism

The key distinction of the KFD method lies in applying distillation not over all tokens, but only on those where the outputs of the teacher model and the base model diverge significantly. To achieve this, the distributional difference between the teacher model (M2) and the base model (M0) is taken into account. Tokens where the teacher diverges from the base model are considered critical for transfer, as they carry richer informational signals. Restricting distillation to such tokens reduces computational costs while making the knowledge transfer more effective.

For each token t, the predefined criterion is given by

$$d_t = \mathrm{KL}\big(p_t^{(M2)} \parallel p_t^{(M0)}\big). \tag{7}$$

Here, the value of d_t represents the degree of divergence between the teacher and base models. A large d_t indicates that the token is either more uncertain or more informative from the teacher's perspective.

The token selection process is defined in two stages:

1. **Threshold-based selection:** For each token, the computed d_t values are compared against a predefined threshold τ:

$$\mathcal{S}_\tau = \{t \mid d_t \geq \tau\}. \tag{8}$$

2. **Global budget constraint:** The number of tokens subject to distillation during training is constrained by a total budget parameter B. In this case, the selection set is defined as

$$\mathcal{S} = \text{Top-}B\big(\{d_t \mid t \in \mathcal{S}_\tau\}\big), \tag{9}$$

meaning that at most the top B tokens with the highest d_t values are selected.

As a result, KFD applies distillation only on the tokens included in the set $\mathcal{S}$. In this way, computational costs are reduced while the most informative parts are prioritized for transfer to the student model.

2.4. Post-Threshold Processing Flow

Once the thresholding step is applied, the KFD process proceeds as follows:

1. **Top-k Check and Bonus:** Let the rank of the ground-truth label in the teacher distribution be $r_t = \mathrm{rank}_{M2}(y_t)$, and let the set of the teacher's Top-k tokens be $\mathcal{T}_t = \text{Top-}k(p_t^{(M2)})$.

$$\tilde{p}_t^{(M2)}[v] = \begin{cases} p_t^{(M2)}[v]\ _{\{v \in \mathcal{T}_t\}}, & \text{if } y_t \in \mathcal{T}_t, \\ p_t^{(M2)}[v] + \dfrac{1}{r_t + 1}\ _{\{v = y_t\}}, & \text{if } y_t \notin \mathcal{T}_t. \end{cases} \tag{10}$$

2. **Determining α:** The coefficient α in the distillation loss is computed based on how many of the selected tokens in the batch have their ground-truth labels within the Top-k. The ratio of such tokens is given by:

$$\rho = \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \{r_t \leq k\}. \tag{11}$$

Subsequently, α is linearly scaled between minimum and maximum values according to this ratio:

$$\alpha = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot \rho. \tag{12}$$

As a result of this step, each selected token obtains an updated teacher distribution $\tilde{p}_t^{(M2)}$, while a dynamically adjusted batch-level α coefficient is also derived. These values are used in the computation of the final KFD loss.

3. **KFD Loss Function:** The KFD loss is defined as a weighted sum of the classical cross-entropy (CE) and KL-divergence (KD) components. For each selected token t:

$$\mathcal{L}_{\mathrm{KFD}} = \alpha \cdot \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \mathrm{KL}\Big(\tilde{p}_t^{(M2)} \parallel p_t^{(M1)}\Big) + (1-\alpha) \cdot \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \mathrm{CE}\Big(y_t, p_t^{(M1)}\Big). \tag{13}$$

Here, KL denotes the distillation component, while CE represents the standard loss computed with respect to the ground-truth labels. In this way, KFD balances efficiency and accuracy by considering the teacher distribution only for tokens deemed informationally critical.

2.5. Knowledge-Filtered Distillation Algorithm

The procedure summarized in Algorithm 1 outlines the training flow of the KFD approach in detail. During training, which progresses over each mini-batch, token-level probability distributions are first obtained from the teacher (M_2) and base (M_0) models. The divergence between these two distributions is then measured using the token-level KL-divergence metric, and tokens exceeding a predefined threshold are selected as information-critical examples where the teacher and base model differ substantially. Since threshold-based selection alone is insufficient, a global token budget is additionally applied to constrain the selection, ensuring both quality and efficiency.

Algorithm 1 KFD: Knowledge-Filtered Distillation

Require: Base model M_0, student M_1, teacher M_2
Require: Training stream $\mathcal{D}$, threshold τ, global token budget B
Require: Top-k size $k = 5$, range $(\alpha_{\min}, \alpha_{\max})$
Ensure: Trained student $\hat{M}_1$

Sample minibatch $x \sim \mathcal{D}$
Compute distributions: $p^{(M_2)} \leftarrow M_2(x)$, $p^{(M_0)} \leftarrow M_0(x)$
Tokenwise divergence: $d_t \leftarrow \mathrm{KL}(p_t^{(M_2)} \parallel p_t^{(M_0)})$
Candidate set by threshold: $\mathcal{S}_\tau \leftarrow \{\, t \mid d_t \geq \tau \,\}$
Budgeted selection: $\mathcal{S} \leftarrow \textsc{AllocateBudget}(\mathcal{S}_\tau, d, B)$

Teacher transform on each $t \in \mathcal{S}$
$r_t \leftarrow \mathrm{rank}_{M_2}(y_t)$, $\mathcal{T}_t \leftarrow \text{Top-}k(p_t^{(M_2)})$
if $r_t \leq k$ *then* keep only $\mathcal{T}_t$ and renormalize $\Rightarrow \tilde{p}_t^{(M_2)}$
else add bonus $b_t \leftarrow \frac{1}{r_t+1}$ to $\tilde{p}_t^{(M_2)}[y_t]$ and renormalize

Batch Top-k ratio: $\rho \leftarrow \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \mathbf{1}\{r_t \leq k\}$
Distillation weight: $\alpha \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\, \rho$
Student forward: $p^{(M_1)} \leftarrow M_1(x)$
KL loss on $\mathcal{S}$: $\mathcal{L}_{\mathrm{KL}} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \mathrm{KL}(\tilde{p}_t^{(M_2)} \parallel p_t^{(M_1)})$
CE loss on $\mathcal{S}$: $\mathcal{L}_{\mathrm{CE}} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} \mathrm{CE}(y_t, p_t^{(M_1)})$
Final objective: $\mathcal{L}_{\mathrm{KFD}} \leftarrow \alpha\, \mathcal{L}_{\mathrm{KL}} + (1-\alpha)\, \mathcal{L}_{\mathrm{CE}}$
Output: $\hat{M}_1 \leftarrow M_1$

For the selected tokens, the teacher's Top-k probabilities are considered. If the ground-truth label lies within this set, the teacher distribution is renormalized only over this subset, allowing the model to focus on the most reliable signals. If the ground-truth label is not in the Top-k, an additional rank-based bonus is assigned according to its position in the teacher distribution, and the distribution is updated with this information. This mechanism prevents meaningful signals with low probabilities from being disregarded.

In the final stage, a dynamic batch-level coefficient α is computed, and the student model (M_1) is updated using the KFD loss, which jointly optimizes the classical cross-entropy loss and the KL-divergence loss. In this way, KFD intensifies the learning signal by focusing only on critical tokens, reduces unnecessary computation, and enables achieving high accuracy within shorter training times.

3. Comparative Analysis of Knowledge-Filtered Distillation from a Target-Error Perspective

The design of KFD is guided by two complementary objectives. The first objective concerns the *data elimination/filtering* mechanism: by selecting token instances that carry the strongest learning signal—i.e., more discriminative and less noisy tokens (e.g., those emphasized by the more discriminative model $\mathcal{M}_2$)—KFD aims to reach the same or higher performance with **less data** and thus in **shorter training time**. In other words, filtering reduces the "inefficient" portion of learning and redirects the student's capacity toward more informative examples.

In this section, we focus on the second objective: for the examples that pass the filter, we analyze why KFD's *conditional target construction* (in-Top-k $\rightarrow$ Top-k renormalization, out-of-Top-k $\rightarrow$ bonus) is preferable to two ablations from a *target error* perspective:

- **Always-TopK:** applying Top-k truncation and renormalization to every example.
- **Always-Bonus:** applying the bonus-augmented target to every example.

Our goal is to quantify, given the availability of the ground-truth label y, how well the constructed target distribution preserves label-focused supervision, and to theoretically demonstrate that KFD avoids the "failure regimes" of the two extreme strategies (e.g., completely discarding the true label for out-of-Top-k examples, or unnecessarily diluting the learning signal for in-Top-k examples).

3.1. Definitions and Notation

Let a token instance be $z = (x, t, y)$, where $y \in \{1, \ldots, V\}$ denotes the ground-truth label. Let $q(\cdot) \in \Delta^{V-1}$ denote the teacher distribution (for a single token position).

3.1.1. Top-k Set and Top-k Target

Let $T = \text{Top-}k(q)$ be the indices of the k largest probabilities under q. The Top-k truncated and renormalized target is defined as

$$q_i^{(k)} = \begin{cases} \frac{q_i}{Z}, & i \in T, \\ 0, & i \notin T, \end{cases} \qquad Z = \sum_{j \in T} q_j \leq 1. \tag{14}$$

In particular, if $y \notin T$, then $q_y^{(k)} = 0$.

3.1.2. Bonus-Augmented Target

The bonus target is defined over the full vocabulary as

$$\tilde{q}_i = \frac{q_i + b\,\mathbf{1}\{i = y\}}{1 + b}, \qquad b > 0. \tag{15}$$

Hence

$$\tilde{q}_y = \frac{q_y + b}{1 + b} \;\geq\; \frac{b}{1 + b}. \tag{16}$$

When using a rank-based bonus in KFD, $b = \frac{1}{r+1}$ with $r = \mathrm{rank}_q(y)$, which implies

$$\tilde{q}_y \;\geq\; \frac{1}{r + 2}. \tag{17}$$

3.1.3. Three Methods (Target Construction)

$$r^{\mathrm{KFD}}(q, y) = \begin{cases} q^{(k)}, & y \in T, \\ \tilde{q}, & y \notin T, \end{cases} \tag{18}$$

$$r^{\mathrm{AT}}(q, y) = q^{(k)} \quad \text{(Always-TopK)}, \tag{19}$$

$$r^{\mathrm{AB}}(q, y) = \tilde{q} \quad \text{(Always-Bonus)}. \tag{20}$$

3.1.4. Target Error

We measure target error with respect to the one-hot ground-truth target using the cross-entropy

$$\mathcal{E}(r; y) := \mathrm{CE}(\mathbf{e}_y, r) = -\log r_y, \tag{21}$$

where $\mathbf{e}_y$ denotes the one-hot vector for class y.

This definition captures the label-focused quality of the target distribution: increasing the target mass on the correct label y decreases the target error.

3.2. Theorems: Why Knowledge-Filtered Distillation Outperforms the Ablations

Theorem 1 (In the out-of-Top-k regime, KFD yields lower target error than Always-TopK)**.** *Let $y \notin T$. Then*

$$\mathcal{E}\left(r^{\mathrm{KFD}}; y\right) \;\leq\; \mathcal{E}\left(r^{\mathrm{AT}}; y\right), \tag{22}$$

and moreover the following quantitative comparison holds:

$$\mathcal{E}\left(r^{\mathrm{AT}}; y\right) = \infty, \qquad \mathcal{E}\left(r^{\mathrm{KFD}}; y\right) \;\leq\; \log\Big(\frac{1 + b}{b}\Big). \tag{23}$$

Therefore, as b increases (or, under a rank-based bonus, as r decreases), KFD's target error decreases relative to the "zeroing-out" behavior of Top-k truncation.

(Proof: Appendix A, Theorem 1).

3.2.1. Short Intuition (Why Is Knowledge-Filtered Distillation Better?)

When $y \notin T$, Always-TopK sets $r_y^{\mathrm{AT}} = q_y^{(k)} = 0$, which yields infinite cross-entropy with respect to the one-hot target. In contrast, KFD guarantees $r_y^{\mathrm{KFD}} = \tilde{q}_y \geq \frac{b}{1+b} > 0$ via the bonus, thereby preserving a nonzero label signal.

Lemma 1 (If Top-k renormalization increases the mass on y, the target error decreases)**.** *Let $y \in T$ and define the target error as*

$$\mathcal{E}(r; y) := \mathrm{CE}(\mathbf{e}_y, r) = -\log r_y. \tag{24}$$

If

$$q_y^{(k)} \geq \tilde{q}_y, \tag{25}$$

then

$$\mathcal{E}\left(q^{(k)}; y\right) \;\leq\; \mathcal{E}\left(\tilde{q}; y\right). \tag{26}$$

Theorem 2 (In the in-Top-k regime, KFD has a focusing advantage over Always-Bonus)**.** *Let $y \in T$. In this case, KFD uses $q^{(k)}$ whereas Always-Bonus uses $\tilde{q}$. Under Lemma 1, KFD achieves a smaller target error:*

$$\mathcal{E}\left(r^{\mathrm{KFD}}; y\right) \;\leq\; \mathcal{E}\left(r^{\mathrm{AB}}; y\right). \tag{27}$$

(Proof: Appendix A, Theorem 2).

3.2.2. Why Is Knowledge-Filtered Distillation Better?

The regime $y \in T$ corresponds to cases where the teacher already assigns high probability to the correct label. Here, KFD concentrates and renormalizes the target mass within Top-k, often increasing the mass on y and thus reducing the label-focused target error. Always-Bonus, by applying a mixture-style target everywhere, may forgo this focusing effect and can dilute the learning signal even in reliable examples.

3.3. Overall Summary

Under the target error metric $\mathcal{E}(r; y) = \mathrm{CE}(\mathbf{e}_y, r)$, the key advantage of KFD is twofold: (i) in the out-of-Top-k regime, it prevents Always-TopK from fully discarding the ground-truth signal by ensuring a nonzero target mass on y via the bonus (Theorem 1); and (ii) in the in-Top-k regime, it concentrates the target on the Top-k set and can allocate more mass to the correct label, yielding a more focused signal than Always-Bonus under Lemma 1 (Theorem 2). Overall, KFD can be viewed as a conditional target construction that mitigates the "wrong target in the wrong regime" failure modes of the two extreme strategies.

4. Experiments

This section presents the experimental studies conducted to evaluate the proposed KFD method. The results are reported across different large model types and datasets. Model performance is measured on a held-out test set using the **next-token prediction** accuracy. The best results are highlighted in green.

The experiments are conducted on both 100-document and 1000-document subsets. The smaller 100-document subset is chosen to observe the behavior of the method under limited data conditions and to test the data efficiency of KFD. In contrast, the larger 1000-document subset is used to demonstrate the scalability of the method and its performance improvements when more data is available. These two different data sizes enable a comparative evaluation of KFD's effectiveness in both small- and large-scale scenarios. In the tables, these two data settings are denoted as V1 (100 documents) and V2 (1000 documents).

4.1. Construction of M0, M1, and M2 Models

The M0, M1, and M2 models share the same architecture. All models are based on the CosmosGPT architecture [38] and are trained in two different sizes: Medium (350 M) and Large (750 M).

- **M0 model:** Obtained through pretraining with 10 GB of CulturaX data [39].
- **M1 model:** Trained further on top of M0 using a combined dataset of 20 GB from Cosmopedia (Stanford) [40], OpenOrca [41], and Tinystrange-textbook [42].
- **M2 model:** Trained further on top of M0 using 20 GB of Cosmopedia (Stories) [40] data.

The overall process is illustrated in Figure 2, while the training times are reported in Table 1.

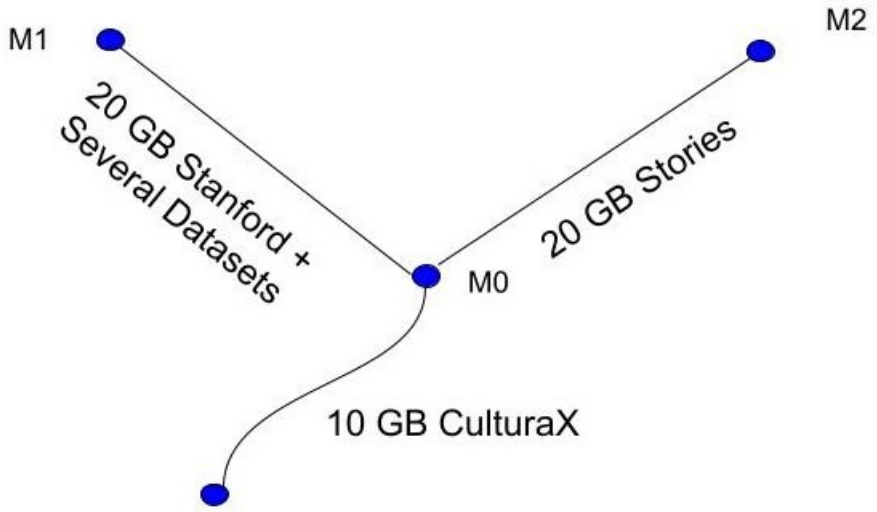


Figure 2. Training process of M0, M1, and M2 models.

Table 1. Approximate Training Times of M0, M1, and M2.

Model	Medium	Large	Description
M0	∼20 h	∼30 h	Pretrain
M1	∼40 h	∼60 h	Finetune on M0
M2	∼40 h	∼60 h	Finetune on M0

4.2. Determination of the Threshold Value

The threshold value (τ) used in the KFD method is determined from the empirical KL-divergence distribution between the teacher (M2) and the base (M0) models. As shown in Figure 3, most tokens are concentrated in a narrow low-divergence region, whereas the more informative disagreement signals are located in the upper part of the distribution. Very small threshold values retain too many low-divergence tokens and increase computational cost, while excessively large threshold values remove too much of the informative region and reduce signal diversity.

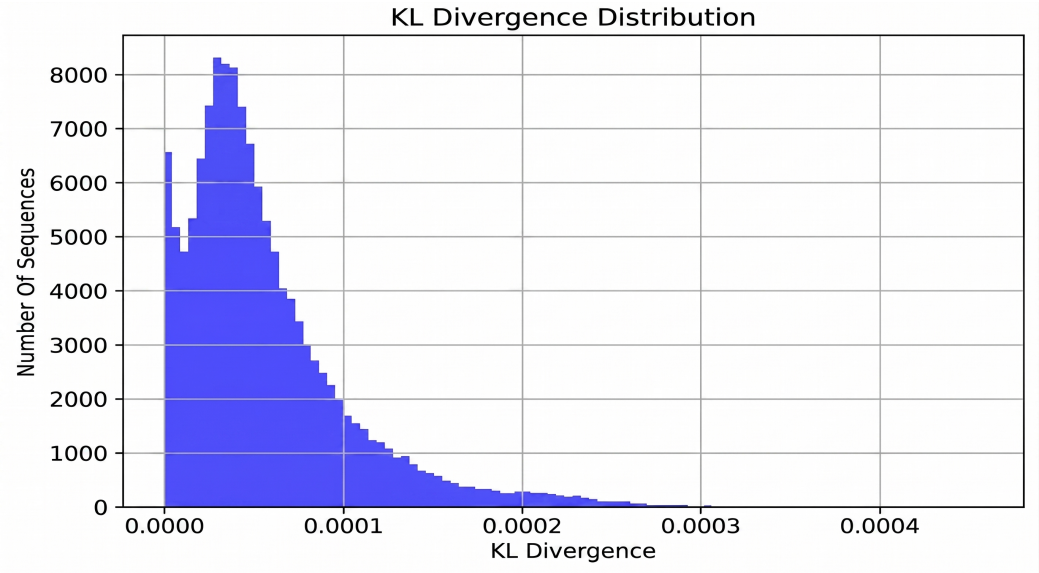


Figure 3. Distribution of KL-divergence values.

In this study, thresholding is defined as a percentile-based filtering step. Specifically, the lower 80% of token-level KL-divergence values are excluded, and only the upper 20%

are retained for distillation. Under this definition, the numerical value $\tau = 0.00008$ is the empirical cutoff corresponding to that 80/20 split for the KL-distribution shown in Figure 3. This choice shifts training away from the dense low-divergence majority and toward the higher-divergence subset, yielding a more selective and information-focused transfer process.

4.3. Size of the Selected Sequences

Table 2 shows the amount of data scanned to obtain the specified number of sequences. Table 3 presents the total number of sequences corresponding to the raw data sizes.

Table 2. Amount of Data Scanned to Obtain Filtered Sequences.

Scanned Data	Filtered Sequence Count
∼42 MB	1,500,000
∼336 MB	12,000,000
∼672 MB	24,000,000
∼1 GB	36,000,000

Table 3. Total Number of Sequences for Raw Data Sizes.

Raw Data Size	Total Sequence Count
1 GB	107,346,501
20 GB	2,146,930,020

4.4. Selection of Hyperparameters

The effectiveness of the KFD method critically depends on the appropriate choice of hyperparameters. In this context, three parameters stand out in particular: the *Top-k* value, the coefficient α, and the *rank bonus* term. The Top-*k* value determines which candidates from the teacher distribution are transferred to the student; the coefficient α controls the balance between the KL-divergence and the cross-entropy loss; and the bonus term is added to strengthen the learning signal in cases where the ground-truth label appears at lower ranks. To evaluate the effects of these parameters and to achieve consistent results, experiments were conducted on a training dataset of 1.5 M sequences.

4.4.1. Top-*k* Value

Table 4 presents the test results obtained after training with different Top-*k* values.

Table 4. Results obtained with different Top-*k* values. Bold and green values indicate the best results in each block.

Model	Test Data	Top-1	Top-5
KFD-Top1		50.02%	71.43%
KFD-Top3		50.21%	71.82%
KFD-Top5	**V1**	**50.28%**	**71.99%**
KFD-Top7		50.24%	71.96%
KFD-Top10		50.26%	71.99%
KFD-Top1		49.27%	70.46%
KFD-Top3		49.47%	70.88%
KFD-Top5	**V2**	**49.52%**	**71.04%**
KFD-Top7		49.50%	71.03%
KFD-Top10		49.51%	71.04%

4.4.2. Dynamic Determination of Bonus and α Parameters

Two critical hyperparameters in the KFD method are the *bonus* and α values. The *bonus* represents an additional weight applied to reinforce low-probability but relevant teacher predictions, while the coefficient α determines the balance between the cross-entropy (CE) and KL-divergence (KD) components in the overall loss.

In the experimental setup, different scenarios were evaluated for the use of the *bonus* and α parameters. First, settings in which both parameters were kept fixed were examined. Next, cases were tested where the *bonus* was averaged while α remained fixed. Conversely, settings in which the *bonus* remained fixed but α was updated dynamically were also considered. In addition, cases where the *bonus* was averaged and α was dynamic were evaluated separately. Finally, both parameters were updated dynamically.

Table 5 reports the results obtained with these fixed and dynamic configurations. Rows highlighted in bold correspond to the setting with dynamic *bonus* and dynamic α.

Table 5. Results obtained when the *bonus* and α parameters are set as fixed or dynamic. Bold and green values indicate the best results in each test block.

Model	Test Data	Top-1	Top-5
KFD-NoBonus	**V1**	50.22%	71.92%
KFD-01Bonus-70-30		50.33%	72.02%
KFD-AvgBonus-70-30		50.20%	71.86%
KFD-01Bonus-D.Alpha		50.31%	71.96%
KFD-AvgBonus-D.Alpha		50.19%	71.87%
KFD-D.Bonus-D.Alpha		**50.38%**	**72.10%**
KFD-NoBonus	**V2**	49.48%	70.95%
KFD-01Bonus-70-30		49.51%	71.00%
KFD-AvgBonus-70-30		49.42%	70.90%
KFD-01Bonus-D.Alpha		49.52%	71.01%
KFD-AvgBonus-D.Alpha		49.43%	70.89%
KFD-D.Bonus-D.Alpha		**49.55%**	**71.09%**

In addition to the dynamic-update setting, the functional form of the *bonus* term was also examined. For this purpose, the reciprocal rank-based *bonus* used in KFD, $b = 1/(r+1)$, was compared with an exponential *bonus* formulation under the same V1 setting and training budgets. Table 6 presents the corresponding Top-1 and Top-5 results for the 12 M, 24 M, and 36 M configurations.

Table 6. Comparison of linear and exponential bonus formulations on V1. Bold values indicate the better result within each row.

Model	Linear Top-1	Linear Top-5	Exponential Top-1	Exponential Top-5
KFD-12M	**54.68%**	**76.15%**	54.03%	75.23%
KFD-24M	**55.27%**	**76.65%**	54.72%	75.88%
KFD-36M	**55.82%**	**76.98%**	55.22%	76.43%

4.5. Knowledge-Filtered Distillation

After determining the parameters for KFD, this section presents the results of the actual comparative experiments.

First, to demonstrate the fundamental effectiveness of the proposed method, KFD is compared against models trained solely with cross-entropy (*CE-only*). This comparison is critical for understanding why KFD favors selective transfer over classical full-data training. The results for the medium models on the V1 and V2 test sets are shown in Figures 4 and 5,

respectively, while the results for the large models are shown in Figures 6 and 7, respectively. To demonstrate the overall reliability and consistency of the proposed approach, the KAFA-12M configuration was trained five times with different random seeds, and the mean and standard deviation values are reported using the "±" notation. The results for the medium and large models are shown in Table 7. Although repeated only for this representative setup, the purpose of this analysis was to indicate the general stability of the KFD framework rather than to focus on a single model instance.

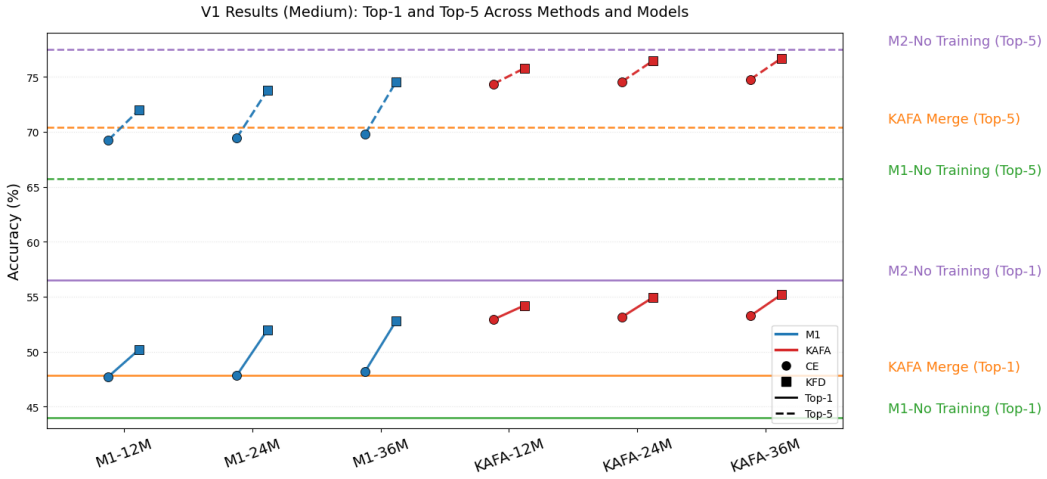


Figure 4. Performance of the medium model on the V1 test set.

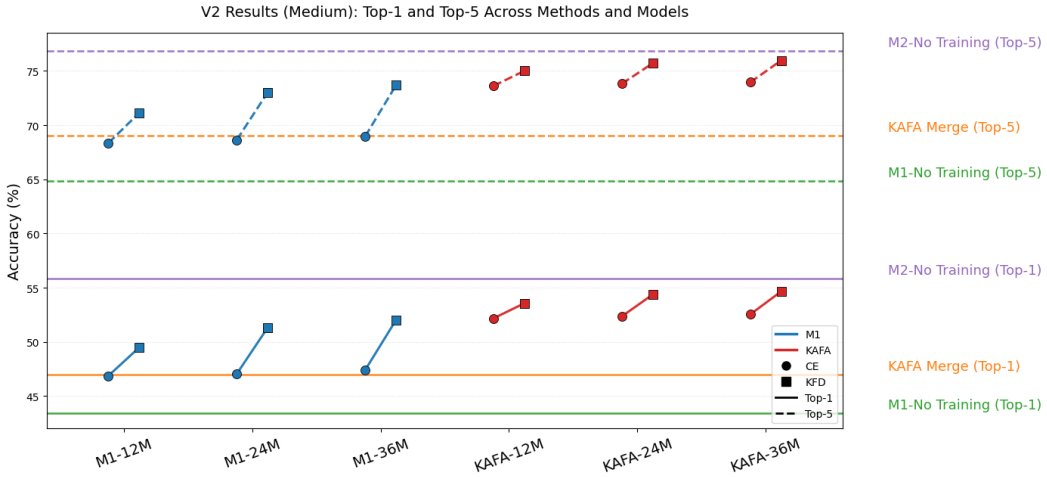


Figure 5. Performance of the medium model on the V2 test set.

Table 7. KAFA-12M configuration was trained five times with different random seeds, and the mean and standard deviation values are reported using the "±" notation.

Model	Test Data	CE-Top1	CE-Top5	KFD-Top1	KFD-Top5
KAFA-12M-Medium	**V1**	52.93% ± 0.05	74.36% ± 0.03	54.20% ± 0.06	75.79% ± 0.05
KAFA-12M-Large		54.01% ± 0.06	75.64% ± 0.08	55.44% ± 0.07	76.96% ± 0.05
KAFA-12M-Medium	**V2**	52.17% ± 0.03	73.62% ± 0.02	53.56% ± 0.03	75.00% ± 0.02
KAFA-12M-Large		53.24% ± 0.02	74.75% ± 0.01	54.68% ± 0.02	76.15% ± 0.03

In addition, the widely used *entropy-based token selection* approach from the literature is included as a baseline. In this way, the advantage of KFD's $\mathrm{KL}(M_2 \parallel M_0)$-based selection strategy over simple entropy-thresholding methods is revealed. The results of

these comparisons are reported in Table 8, where KFD clearly demonstrates superiority in terms of both accuracy and data efficiency.

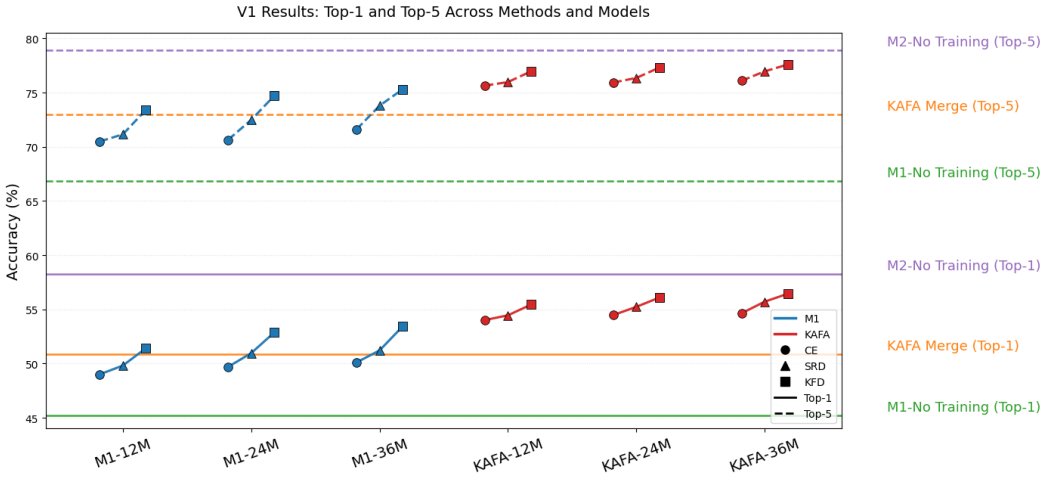


Figure 6. Performance of the large model on the V1 test set.

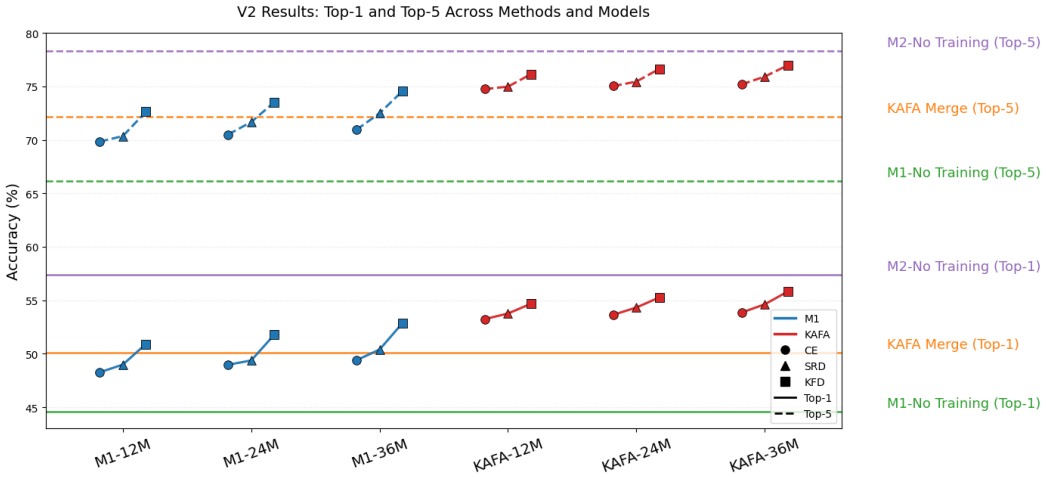


Figure 7. Performance of the large model on the V2 test set.

Table 8. Medium models: Comparison of KFD's $\mathrm{KL}(M_2 \parallel M_0)$-based token selection with the standard entropy-based token selection baseline. Bold and green values indicate the best results in each test block.

Model	Test Data	Entropy-Top1	Entropy-Top5	KFD-Top1	KFD-Top5
KAFA-12M	**V1**	53.81%	75.22%	54.20%	75.79%
KAFA-24M		54.24%	75.66%	54.92%	76.46%
KAFA-36M		54.44%	75.96%	**55.18%**	**76.68%**
KAFA-12M	**V2**	52.85%	74.33%	53.56%	75.00%
KAFA-24M		53.35%	74.80%	54.37%	75.71%
KAFA-36M		53.74%	75.11%	**54.65%**	**75.96%**

When interpreting the results, it is also important to clarify the meaning of the models listed in the table: *M1 No Training* refers to the baseline student model without any additional training; *M1+D2 (Full)* denotes the student retrained directly on the full dataset used for training the teacher; *M2 No Training* indicates the teacher model itself; *Linear Merge* and *KAFA-Merge* represent combined models obtained by applying linear and KAFA-based merging techniques to M1 and M2, respectively [36]. For KFD, different variants are reported according to the number of training sequences (12M, 24M, and 36M tokens). This

allows us to evaluate both the impact of training data size on KFD's performance and the scalability of the method.

In these comparisons, the **teacher model** (M_2) serves as an important reference point. Since the highest attainable performance bound for the proposed method is the teacher model itself, the primary goal of KFD is to enable the student to approach this upper bound accuracy as closely as possible.

4.5.1. Alternative Evaluation Metrics

To provide a more comprehensive understanding of KFD's impact, the models were evaluated not only with accuracy scores (Top-1/Top-5) but also with additional metrics. In this section, we report two complementary evaluations: **HellaSwag accuracy** and **perplexity**.

- **HellaSwag Results:** The HellaSwag dataset [43] measures models' performance on reasoning and natural language inference. The goal here is to examine the effect of KFD and alternative training/merging strategies on logic-oriented tasks. In this study, we used the Turkish-translated version, Hellaswag-tr [44], and followed the evaluation methodology described in LLMTurkishLeaderboard-v0.2 [45].
- **Perplexity Results:** Perplexity is a standard metric that evaluates how well a language model captures the probability distribution of sequences, reflecting the average uncertainty of its predictions. Lower perplexity indicates that the model can more accurately predict word sequences in the test set. In our experiments, perplexity was computed for both V1 and V2 test scenarios. Of particular interest are the *KAFA-NonFiltered-20GB* and *KAFA-NonFiltered-Large-20GB* experiments, where KFD's filtering mechanism was disabled and CE+KL losses were applied over all tokens. This comparison highlights the contribution and necessity of the proposed selective filtering.

The combined results for HellaSwag accuracy and perplexity on the V1 and V2 test sets are presented in Table 9.

Table 9. Combined perplexity results on V1/V2 and HellaSwag accuracy. Lower perplexity is better; higher HellaSwag accuracy is better. Green highlights the best result in each column.

Model	Perp (V1)	Perp (V2)	HellaSwag
M1	19.89	21.55	26.11%
M2	7.88	8.35	26.18%
KAFA-12M	9.02	9.60	26.01%
KAFA-24M	8.58	9.09	26.51%
KAFA-36M	8.34	8.92	**26.87%**
KAFA-NonFiltered-20GB	**7.77**	**8.24**	26.43%
M1-Large	17.95	19.43	26.89%
M2-Large	7.05	7.46	26.80%
KAFA-Large-12M	8.30	8.80	27.05%
KAFA-Large-24M	8.00	8.48	27.14%
KAFA-Large-36M	7.85	8.29	**27.26%**
KAFA-NonFiltered-Large-20GB	**6.86**	**7.28**	26.93%

4.5.2. Matched-Time Comparison with CE Training

An additional comparison was conducted to evaluate CE-only training and KFD under a matched training-time setting. For this purpose, CE-only training was also run with an extended training duration, and the resulting performance was recorded together with the corresponding KFD results under the same experimental configuration.

Figure 8 presents this comparison for the Large model on the V1 setting. The figure includes the results of CE-only training, extended CE-only training, and KFD obtained under the same setup.

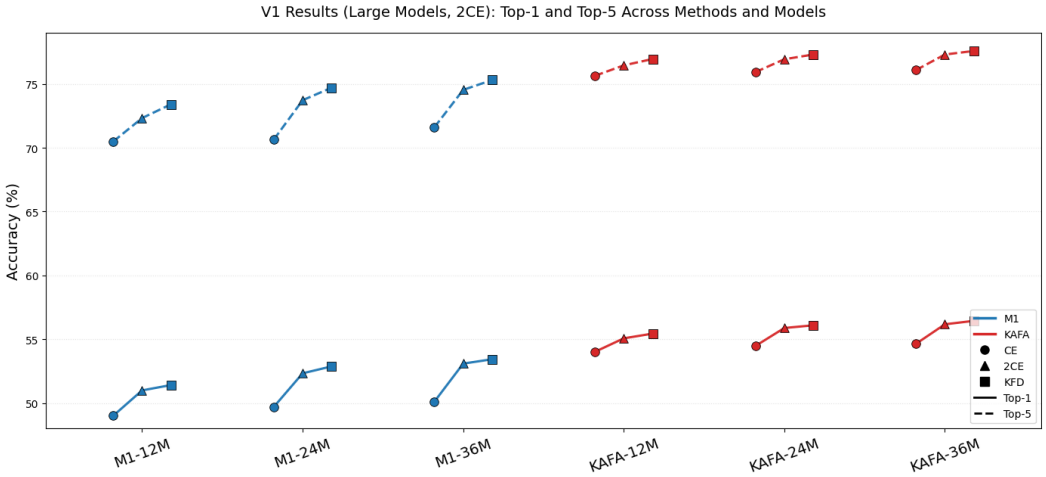


Figure 8. Matched-time comparison of CE-only training, extended CE-only training, and KFD for the Large model on V1.

4.5.3. Additional Experiments on an English Setup

An additional experiment was conducted to examine the behavior of KFD in an English-language setting. For this purpose, the `openai/gpt-2` [46] model was used, and the teacher–student setup was constructed using the English *Stories* and *Stanford* datasets. After training, the models were evaluated on the V1 English test set using Top-1, Top-5, and HellaSwag accuracy.

Table 10 presents the corresponding results obtained in this English setup.

Table 10. Performance on the V1 test set in the English setup.

Model	Test Data	Top-1	Top-5	HellaSwag
M0	**V1**	40.34%	62.12%	28.24
M1	**V1**	43.56%	67.01%	29.66
M2	**V1**	53.29%	78.05%	29.39
M1-100M-CE	**V1**	49.63%	73.97%	30.51
M1-100M-SRD	**V1**	50.34%	75.12%	30.87
M1-100M-KFD	**V1**	51.71%	76.21%	31.14

4.5.4. Student–Teacher Role Reversal

In the previous experiments, the M1-Stanford model was used as the student, while the M2-Stories model served as the teacher. In this section, the roles are reversed: M2-Stories is assigned as the student and M1-Stanford as the teacher, as illustrated in Figure 9. This setup allows us to examine whether the proposed method yields consistent performance across different student–teacher configurations.

The tables first report the performance of the models without any additional training, followed by the results of KFD under different sequence lengths. Furthermore, the "Non-Filtered-20GB" condition corresponds to training with KFD applied on all tokens without any filtering, where both CE and KL losses are computed across the entire dataset.

The results for the medium models on the V1 and V2 test sets are shown in Figures 10 and 11, respectively, while the results for the large models are shown in Figures 12 and 13, respectively.

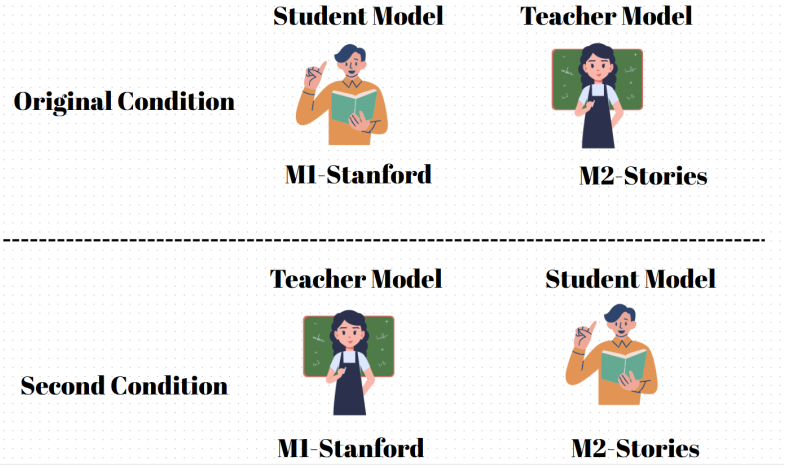


Figure 9. In the *Original Condition* (**top**), M1-Stanford is the student and M2-Stories the teacher. In the *Reversed Condition* (**bottom**), the roles are swapped, with M1-Stanford acting as the teacher and M2-Stories as the student.

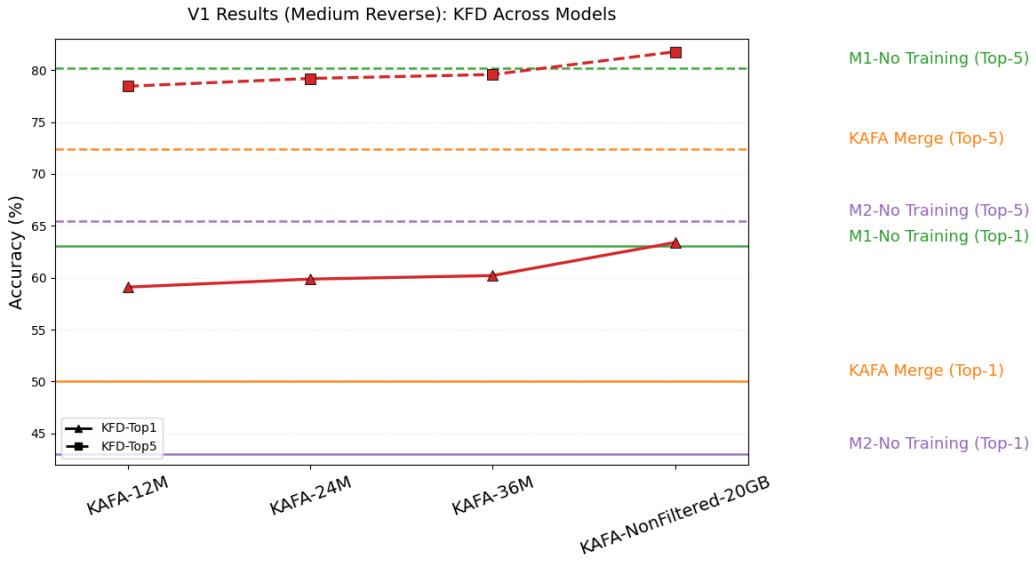


Figure 10. Performance of the medium model under the reverse setting on the V1 test set.

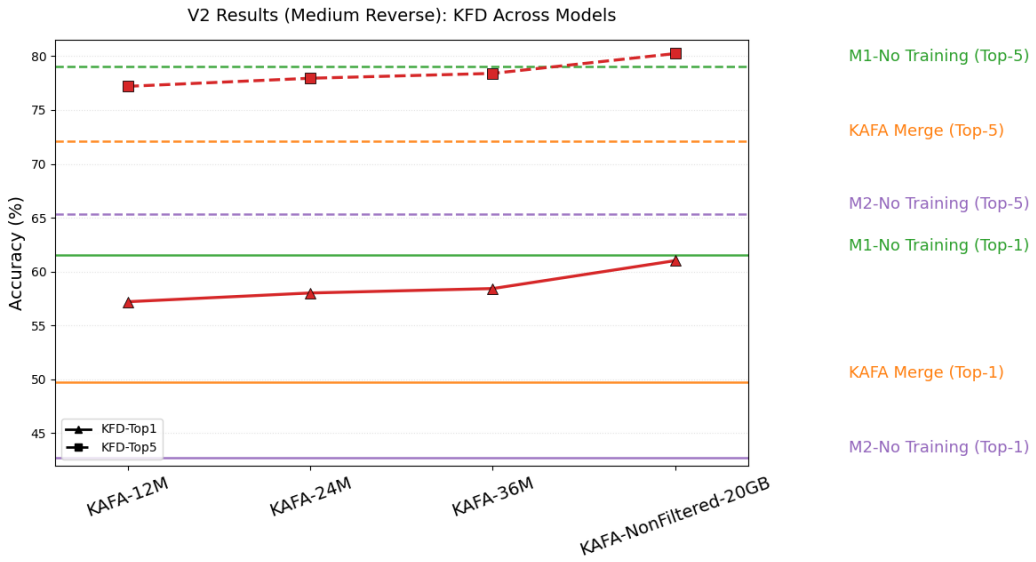


Figure 11. Performance of the medium model under the reverse setting on the V2 test set.

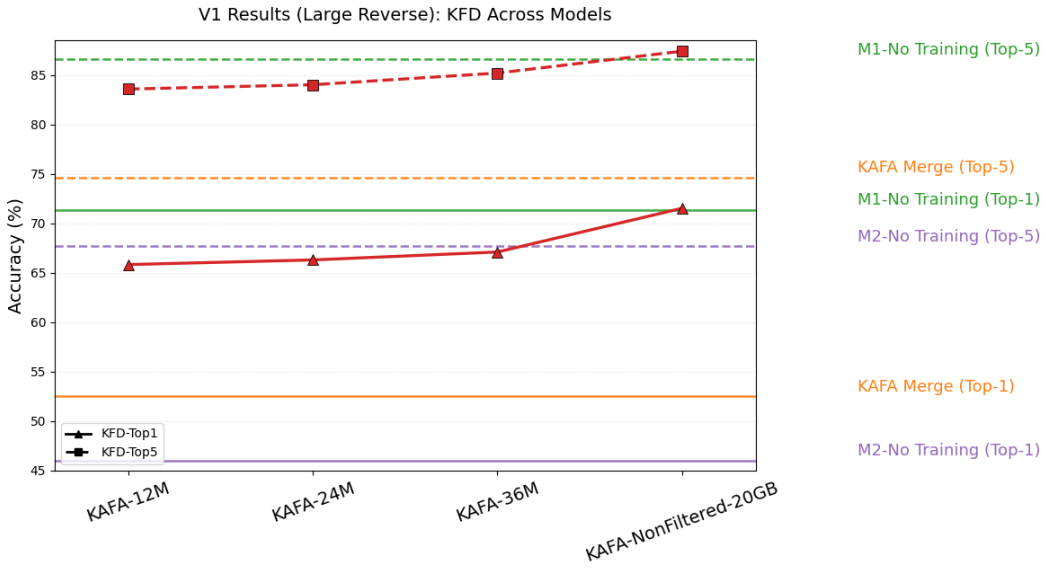


Figure 12. Performance of the large model under the reverse setting on the V1 test set.



Figure 13. Performance of the large model under the reverse setting on the V2 test set.

4.5.5. Effect of Different Threshold Values

In this section, we analyze the impact of the threshold parameter in Algorithm 1 by comparing multiple values ranging from 0.00005 to 0.0002. While the main experiments were conducted using 0.00008 and 0.0001, additional runs with smaller-scale settings (1 M sequences) were performed to illustrate how the threshold directly affects training efficiency and scalability. This small-scale analysis provides complementary insights into the computational cost of KFD and CE models under varying thresholds.

Table 11 summarizes the Top-1 and Top-5 accuracies for both CE and KFD across the two primary thresholds (0.00008 and 0.0001), while Table 12 extends this analysis by reporting detailed training times across multiple sequence sizes. As observed, when the threshold increases from 0.00008 to 0.0001, both computation time and the amount of processed data increase significantly. For instance, obtaining 36 M sequences requires scanning approximately 1 GB of text data at 0.00008, compared to nearly 1.5 GB at 0.0001. The 1 M-level experiments further confirm that this trade-off between accuracy and efficiency holds consistently across smaller-scale setups as well, emphasizing that threshold tuning is a crucial factor in balancing performance and cost.

Table 11. Performance comparison of large models under different threshold values.

Model	Test Data	Threshold	CE Top1	CE Top5	KFD Top1	KFD Top5
KAFA-12M	**V1**	0.00008	54.01%	75.64%	55.44%	76.96%
KAFA-12M		0.0001	54.61%	76.10%	55.92%	77.25%
KAFA-36M		0.00008	54.64%	76.09%	56.44%	77.59%
KAFA-36M		0.0001	55.21%	76.58%	57.15%	78.02%
KAFA-12M	**V2**	0.00008	53.24%	74.75%	54.68%	76.15%
KAFA-12M		0.0001	53.80%	74.25%	55.30%	76.65%
KAFA-36M		0.00008	53.85%	75.19%	55.82%	76.98%
KAFA-36M		0.0001	54.63%	75.61%	56.53%	77.49%

Table 12. Training times across different sequence sizes and threshold values. While 0.00008 and 0.0001 are used in the main experiments, additional 1 M-scale results are included to illustrate the computational cost and efficiency trends across smaller-scale setups. Both CE-only and KFD settings are reported for comparison.

Sequence Size	CE-Only	SRD	KFD	Threshold
1 M	∼3 min	∼5 min	∼5 min	0.00005
1 M	∼5 min	∼7 min	∼6 min	0.00008
1 M	∼8 min	∼13 min	∼12 min	0.0001
1 M	∼3 h	∼7 h	∼6 h	0.0002
12 M	∼30 min	∼1 h 30 min	∼1 h	0.00008
36 M	∼1 h 40 min	∼3 h 30 min	∼2 h 45 min	0.00008
12 M	∼1 h	∼2 h 30 min	∼2 h	0.0001
36 M	∼3 h	∼6 h	∼5 h	0.0001

5. Why Knowledge-Filtered Distillation Provides an Effective Selective Distillation Framework

Instead of applying knowledge transfer uniformly across all tokens, KFD restricts distillation to regions where the outputs of the teacher (M_2) and the base model (M_0) diverge more substantially. This selectivity is implemented through a KL($M_2 \parallel M_0$)-based threshold, which directs computation toward more informative regions and reduces unnecessary distillation cost.

The position of the correct label within the teacher's distribution determines how the target is constructed. If the correct label lies within the teacher's Top-5 predictions, only these candidates are retained and renormalized, producing a sharper teacher target. If the correct label falls outside the Top-5, a ranking bonus is added so that the correct signal is not completely discarded. In this way, low-probability but correct signals can still be preserved in uncertain cases.

Rather than using fixed distillation weights, KFD adapts them according to the degree of similarity or divergence between the teacher and base model outputs. A dynamic α coefficient, determined by the Top-5 ratio, places relatively greater emphasis on cross-entropy when the outputs are more similar and greater emphasis on KL divergence when they differ more strongly. This mechanism provides a data-dependent balance between the two loss components and reduces reliance on a single fixed weighting scheme.

Controlling training cost is also an important part of the KFD design. By defining a global token budget, the total amount of transferred supervision is constrained from the outset, allowing systematic control over the trade-off among data usage, time, and accuracy. Within the reported experiments, KFD was also applied to teacher–student pairs of the same capacity, where selective transfer over a subset of the data produced competitive results relative to full-data CE training.

Student initialization also influences the observed transfer behavior. In the reported experiments, students initialized through Linear Merge or KAFA-Merge provided additional settings for examining KFD under different starting points.

Finally, KFD was compared not only with direct cross-entropy (*CE-only*) training but also with entropy-based token selection approaches. While entropy thresholding provides a general heuristic for identifying informative regions, KFD uses the divergence between the teacher and base distributions as its selection signal. In the reported experiments, this KL-based selection yielded stronger results in the evaluated settings.

Overall, the reported findings suggest that KFD provides a selective distillation framework that combines token filtering, conditional target construction, dynamic loss weighting, and explicit budget control within a single training procedure.

6. Conclusions

This study introduced *Knowledge-Filtered Distillation (KFD)* as a selective and efficient knowledge transfer framework for teacher–student pairs of the same capacity. The study specifically aimed to evaluate whether selective token-level knowledge transfer, combined with adaptive target shaping and dynamic loss weighting, can reduce training cost and data usage while maintaining performance close to full training. KFD is built on four main components: KL-based token filtering, Top-5 target preservation, a rank-based bonus, and a dynamic α that balances KL and CE losses.

Experimental results showed that this selective design yields consistent gains across both the Medium and Large settings. Threshold-based filtering with $\tau = 0.00008$ provided a favorable balance between retained diversity and training cost, while $k = 5$ produced the most balanced Top-1/Top-5 performance. In the main comparisons, the best Medium configuration reached 55.18% Top-1 and 76.68% Top-5 on V1, whereas the best Large configuration reached 56.44% and 77.59%, respectively. These results indicate that KFD can approach or surpass CE-only training while using less data and shorter training time.

KFD also remained effective under different student initializations, including KAFA-Merge. Additional experiments in an English `openai/gpt-2` setup further supported the effectiveness of the method beyond the Turkish setting. A matched-time comparison with extended CE-only training was also included, and KFD remained competitive under the same training-time budget. Regarding perplexity, KFD consistently improved upon the initial student models and remained competitive, although the lowest perplexity values were obtained by the non-filtered CE + KL variants. This indicates that full-token training remains more directly aligned with minimizing next-token prediction error over the full distribution, whereas KFD is preferable when the goal is to achieve strong downstream performance with reduced training cost and data usage.

Author Contributions: Conceptualization, M.K.Y. and M.F.A.; Methodology, M.K.Y. and M.F.A.; Software, M.K.Y.; Writing—original draft, M.K.Y.; Writing—review and editing, M.F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the Scientific and Technological Research Council of Turkey (TUBITAK) under Grant No: 124E055. This work has been funded by Yildiz Technical University Scientific Research Projects Coordination Unit under project number FDK-2024-6421.

Data Availability Statement: The datasets used in this study are publicly available from the sources cited in the manuscript. The code and additional implementation details used to support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: During the preparation of this work the authors used ChatGPT (OpenAI, GPT-5.4 Thinking) in order to improve language and readability. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Proofs

Appendix A.1. Proof of Theorem 1

Proof. Assume $y \notin T$. Then

$$r^{\mathrm{AT}}(q, y) = q^{(k)}, \qquad r^{\mathrm{KFD}}(q, y) = \tilde{q}. \tag{A1}$$

By definition of the Top-k target, when $y \notin T$ we have

$$q_y^{(k)} = 0. \tag{A2}$$

The target error is defined as

$$\mathcal{E}(r; y) = \mathrm{CE}(\mathbf{e}_y, r) = -\log r_y. \tag{A3}$$

Always-TopK

Since $r_y^{\mathrm{AT}} = q_y^{(k)} = 0$, it follows that

$$\mathcal{E}\left(r^{\mathrm{AT}}; y\right) = -\log 0 = \infty. \tag{A4}$$

KFD

For the bonus target,

$$\tilde{q}_y = \frac{q_y + b}{1 + b} \;\geq\; \frac{b}{1 + b}. \tag{A5}$$

Therefore,

$$\mathcal{E}\left(r^{\mathrm{KFD}}; y\right) = \mathcal{E}(\tilde{q}; y) = -\log \tilde{q}_y \leq -\log\Big(\frac{b}{1 + b}\Big) = \log\Big(\frac{1 + b}{b}\Big). \tag{A6}$$

Hence,

$$\mathcal{E}\left(r^{\mathrm{KFD}}; y\right) \;\leq\; \mathcal{E}\left(r^{\mathrm{AT}}; y\right), \tag{A7}$$

which proves the theorem. □

Appendix A.2. Proof of Theorem 2

Proof. Under the assumption $y \in T$, the KFD and Always-Bonus targets are

$$r^{\mathrm{KFD}}(q, y) = q^{(k)}, \qquad r^{\mathrm{AB}}(q, y) = \tilde{q}. \tag{A8}$$

Lemma 1 defines the target error as

$$\mathcal{E}(r; y) = \mathrm{CE}(\mathbf{e}_y, r) = -\log r_y, \tag{A9}$$

and states that under $q_y^{(k)} \geq \tilde{q}_y$ we have

$$\mathcal{E}\left(q^{(k)}; y\right) \leq \mathcal{E}\left(\tilde{q}; y\right). \tag{A10}$$

Hence,

$$\mathcal{E}\left(r^{\mathrm{KFD}}; y\right) = \mathcal{E}\left(q^{(k)}; y\right) \tag{A11}$$

$$\leq \mathcal{E}\left(\tilde{q}; y\right) \tag{A12}$$

$$= \mathcal{E}\left(r^{\mathrm{AB}}; y\right), \tag{A13}$$

which proves the theorem. □

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*; NeurIPS: San Diego, CA, USA, 2017; pp. 5998–6008. Available online: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (accessed on 14 April 2026).
2. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the NAACL-HLT*; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; pp. 4171–4186. [CrossRef]
3. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. In *Advances in Neural Information Processing Systems (NeurIPS)*; NeurIPS: San Diego, CA, USA, 2020; pp. 1877–1901. Available online: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (accessed on 14 April 2026).
4. Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; Casas, D.D.L.; Hendricks, L.A.; Welbl, J.; Clark, A.; et al. Training Compute-Optimal Large Language Models. *arXiv* **2022**, arXiv:2203.15556. [CrossRef]
5. Zhuang, B.; Liu, J.; Pan, Z.; He, H.; Weng, Y.; Shen, C. A Survey on Efficient Training of Transformers. *arXiv* **2023**, arXiv:2302.01107. [CrossRef]
6. Xu, C.; McAuley, J. A Survey on Model Compression and Acceleration for Pretrained Language Models. *arXiv* **2022**, arXiv:2202.07105. [CrossRef]
7. Xu, X.; Li, M.; Tao, C.; Shen, T.; Cheng, R.; Li, J.; Xu, C.; Tao, D.; Zhou, T. A Survey on Knowledge Distillation of Large Language Models. *arXiv* **2024**, arXiv:2402.13116.
8. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531. [CrossRef]
9. Buciluǎ, C.; Caruana, R.; Niculescu-Mizil, A. Model Compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*; ACM: New York, NY, USA, 2006; pp. 535–541. [CrossRef]
10. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. FitNets: Hints for Thin Deep Nets. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015. Available online: https://arxiv.org/abs/1412.6550 (accessed on 14 April 2026).
11. Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; Liu, Q. TinyBERT: Distilling BERT for Natural Language Understanding. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020, Online, 16–20 November 2020; pp. 4163–4174. [CrossRef]
12. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. *arXiv* **2019**, arXiv:1910.01108.
13. Turc, I.; Chang, M.-W.; Lee, K.; Toutanova, K. Well-Read Students Learn Better: On the Importance of Pre-Training Compact Models. *arXiv* **2019**, arXiv:1908.08962.
14. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]
15. Wang, Y.; Wang, J.; Zhang, X. Parameter-Efficient Online Knowledge Distillation for Pretrained Language Models. *Expert Syst. Appl.* **2025**, *265*, 126040. [CrossRef]
16. Won, H.-S.; Choi, J.-Y.; Zaman, N.; Aliyeva, D.; Kim, K.-M. P-Distill: Efficient and Effective Prompt Tuning Using Knowledge Distillation. *Appl. Sci.* **2025**, *15*, 2420. [CrossRef]
17. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *arXiv* **2020**, arXiv:2002.10957.
18. Kim, Y.; Rush, A.M. Sequence-Level Knowledge Distillation. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, TX, USA, 1–4 November 2016; pp. 1317–1327. [CrossRef]

19. Freitag, M.; Al-Onaizan, Y.; Sankaran, B. Ensemble Distillation for Neural Machine Translation. *arXiv* **2017**, arXiv:1702.01802. [CrossRef]
20. Tan, X.; Ren, Y.; He, D.; Qin, T.; Zhao, Z.; Liu, T.-Y. Multilingual Neural Machine Translation with Knowledge Distillation. *arXiv* **2019**, arXiv:1902.10461. [CrossRef]
21. Sun, S.; Cheng, Y.; Gan, Z.; Liu, J. Patient Knowledge Distillation for BERT Model Compression. *arXiv* **2019**, arXiv:1908.09355. [CrossRef]
22. Sajjad, H.; Dalvi, F.; Durrani, N.; Nakov, P. On the Effect of Dropping Layers of Pre-trained Transformer Models. *arXiv* **2020**, arXiv:2004.03844. [CrossRef]
23. Sun, Z.; Yu, H.; Song, X.; Liu, R.; Yang, Y.; Zhou, D. MobileBERT: A Compact Task-Agnostic BERT for Resource-Limited Devices. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), Virtual, 5–10 July 2020; pp. 2158–2170. [CrossRef]
24. Tang, R.; Lu, Y.; Liu, L.; Mou, L.; Vechtomova, O.; Lin, J. Distilling Task-Specific Knowledge from BERT into Simple Neural Networks. *arXiv* **2019**, arXiv:1903.12136. [CrossRef]
25. Liao, B.; Wang, Y.; Yang, B.; Gao, X.; Xia, F. An Empirical Study of Knowledge Distillation for Code Understanding Tasks. *arXiv* **2025**, arXiv:2508.15423. [CrossRef]
26. Wang, F.; Yan, J.; Meng, F.; Zhou, J. Selective Knowledge Distillation for Neural Machine Translation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP), Online, 1–6 August 2021; pp. 6456–6466. [CrossRef]
27. Guo, Z.; Li, Y.; Zhang, X.; Liu, Q. Leveraging Logit Uncertainty for Better Knowledge Distillation. *Sci. Rep.* **2024**, *14*, 31249. [CrossRef]
28. Liu, L.; Zhang, M. Less Is More: Selective Reflection for Compatible and Efficient Knowledge Distillation in Large Language Models. *arXiv* **2025**, arXiv:2508.06135. [CrossRef]
29. Hu, Y.; Guo, J.; Feng, X.; Zhao, T. AdaSPEC: Selective Knowledge Distillation for Efficient Speculative Decoders. In *Advances in Neural Information Processing Systems (NeurIPS)*; NeurIPS: San Diego, CA, USA, 2025.
30. Zhang, Y.; Xiang, T.; Hospedales, T.M.; Lu, H. Deep Mutual Learning. *arXiv* **2018**, arXiv:1706.00384.
31. Anil, R.; Pereyra, G.; Passos, A.; Ormándi, R.; Dahl, G.E.; Hinton, G.E. Large Scale Distributed Neural Network Training through Online Distillation. *arXiv* **2018**, arXiv:1804.03235.
32. Ryu, J.; Kim, M.; Kim, J. Sparse MoE Students for Efficient Knowledge Distillation. *IEEE Access* **2025**, *13*, 187373–187382. . [CrossRef]
33. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On Calibration of Modern Neural Networks. *arXiv* **2017**, arXiv:1706.04599. [CrossRef]
34. Kullback, S.; Leibler, R.A. On Information and Sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86. [CrossRef]
35. Wortsman, M.; Ilharco, G.; Gadre, S.Y.; Roelofs, R.; Gontijo-Lopes, R.; Morcos, A.S.; Namkoong, H.; Farhadi, A.; Carmon, Y.; Kornblith, S.; et al. Model Soups: Averaging Weights of Multiple Fine-Tuned Models Improves Accuracy without Increasing Inference Time. *arXiv* **2022**, arXiv:2203.05482. [CrossRef]
36. Yüce, M.K.; Amasyalı, M.F. KAFA-Merge: Model Merging Method with Layer-Based Bayes+Linear Search. *IEEE Access* **2025**, *13*, 108743–108755. [CrossRef]
37. Goddard, C.; Siriwardhana, S.; Ehghaghi, M.; Meyers, L.; Karpukhin, V.; Benedict, B.; McQuade, M.; Solawetz, J. Arcee's MergeKit: A Toolkit for Merging Large Language Models. In Proceedings of the EMNLP: Industry Track, Miami, FL, USA, 12–16 November 2024; pp. 477–485. Available online: https://aclanthology.org/2024.emnlp-industry.36/ (accessed on 14 April 2026).
38. Kesgin, H.T.; Yüce, M.K.; Doğan, E.; Uzun, M.E.; Uz, A.; Seyrek, H.E.; Zeer, A.; Amasyalı, M.F. Introducing cosmosGPT: Monolingual Training for Turkish Language Models. *arXiv* **2024**, arXiv:2404.17336. [CrossRef]
39. Nguyen, T.; Nguyen, C.V.; Lai, V.D.; Man, H.; Ngo, N.T.; Dernoncourt, F.; Rossi, R.A.; Nguyen, T.H. CulturaX: A Cleaned, Enormous, and Multilingual Dataset for Large Language Models in 167 Languages. *arXiv* **2023**, arXiv:2309.09400. [CrossRef]
40. Ben Allal, L.; Lozhkov, A.; Penedo, G.; Wolf, T.; von Werra, L. Cosmopedia. Available online: https://huggingface.co/datasets/HuggingFaceTB/cosmopedia (accessed on 14 April 2026).
41. Lian, W.; Goodson, B.G.; Pentland, E.; Cook, A.; Vong, C.; Teknium. OpenOrca: An Open Dataset of GPT Augmented FLAN Reasoning Traces. Available online: https://huggingface.co/datasets/Open-Orca/OpenOrca (accessed on 14 April 2026).
42. Pham, N. Tiny-Strange-Textbooks (Revision 6f304f1). Available online: https://huggingface.co/datasets/nampdn-ai/tiny-strange-textbooks (accessed on 14 April 2026).
43. Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; Choi, Y. HellaSwag: Can a Machine Really Finish Your Sentence? In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, 28 July–2 August 2019; pp. 4791–4800. [CrossRef]
44. Alhajar, M. Hellaswag Turkish v0.2. Available online: https://huggingface.co/datasets/malhajar/hellaswag_tr-v0.2 (accessed on 14 April 2026).

45. Alhajar, M. Open LLM Turkish Leaderboard v0.2. Available online: https://huggingface.co/spaces/malhajar/OpenLLMTurkishLeaderboard (accessed on 14 April 2026).
46. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models are Unsupervised Multitask Learners. OpenAI Blog. 2019. Available online: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf (accessed on 14 April 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.