Towards Precise Knowledge Distillation for Large Language Models via Knowledge Probing

Jiajun Liu[1], **Yao He**[1], **Wenjun Ke**[1,2], **Peng Wang**[*1,2], **Ziyu Shang**[1], **Guozheng Li**[1], and **Zijie Xu**[1]

[1]School of Computer Science and Engineering, Southeast University
[2]Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China
{jiajliu, heyao, kewenjun, pwang, ziyus1999, gzli, zijiexu}@seu.edu.cn

Abstract

Knowledge distillation (KD) has become a widely adopted approach for compressing large language models (LLMs). Existing distillation paradigms for LLMs can be broadly categorized into black-box and white-box approaches. Black-box methods typically transfer the chains of thought (CoTs) generated by teacher models to student models, yet they fail to convey the intrinsic knowledge embedded within teacher models. White-box methods aim to alleviate this limitation by leveraging internal representations from teacher models. However, they still face two major challenges. First, it is difficult to identify which knowledge is essential for distillation in student models. Second, it is challenging to transfer this key knowledge from teacher models accurately. To tackle these challenges, we propose **KPD**, a novel **K**nowledge **P**robing-based white-box **D**istillation framework for LLMs, which is designed to precisely distill the knowledge that the student model lacks and can be incorporated into diverse distillation approaches. Specifically, to identify critical knowledge gaps in student models, we introduce an uncertainty-based probing method that extracts tokens prone to student errors as critical knowledge. To locate the corresponding knowledge in the teacher model, we utilize integrated-gradient-based attribution to probe where such knowledge is stored and then distill the target layers. Extensive experiments across multiple datasets demonstrate that KPD boosts traditional distillation methods by 0.34%-4.56% in Rouge-L scores.

1 Introduction

Large language models (LLMs) [1, 2, 3] have brought transformative advances to a wide range of natural language processing (NLP) tasks, including text generation [4, 5], code completion [6, 7], and question answering [8, 9, 10]. As model sizes increase, LLMs demonstrate stronger generalization and reasoning capabilities. However, their ever-growing parameter scales lead to severe computational and deployment challenges, especially in resource-limited environments [11, 12]. To solve this challenge, knowledge distillation (KD) has emerged as a practical model compression paradigm, allowing a compact student model to inherit capabilities from a larger and stronger teacher model [13, 14, 15].

Current LLM distillation techniques can be grouped into two major paradigms: black-box and white-box distillation [16, 17, 18, 19]. The black-box paradigm [20, 21, 22] typically uses the teacher model to generate chains of thought (CoTs) to fine-tune the student. These methods improve student reasoning to some extent, but they fail to capture the internal knowledge structures of the teacher. On the other hand, the white-box paradigm [23, 24, 25, 26, 27] directly aligns hidden representations between the teacher and student to enable finer-grained knowledge transfer during the compression.

*Corresponding author.

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

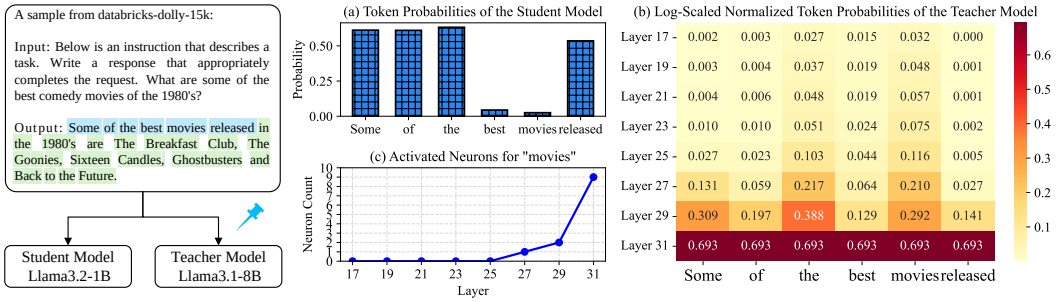


Figure 1: We use Llama 3.1-8B for the teacher model and Llama 3.2-1B for the student model to probe a sample from `databricks-dolly-15k`. (a) shows the prediction results of the student model for the first 6 tokens. (b) shows the output probability of each layer of the teacher model after layer normalization on the first 6 tokens, which reflects how confident each layer is for each token. (c) shows the number of neurons activated in each layer of the teacher model for the output of the token *movies*, which is determined by the value of the cumulative gradient greater than 0.2 during probing.

Despite their advantages, the existing white-box paradigm still suffers from two fundamental limitations that hinder precise distillation, namely, what knowledge to distill and where in the teacher this knowledge resides. **(1) Lack of knowledge awareness.** Most prior white-box methods assume that all tokens and layers contribute similarly, and thus apply uniform alignment over entire sequences and pre-selected layers. This is problematic because only a small subset of tokens typically reflects the student's real knowledge gaps. As illustrated in Figure 1(a), the student assigns exceptionally low confidence to the token *movies*, indicating high uncertainty and revealing a missing piece of domain knowledge. However, inspecting only the teacher's final-layer outputs provides an ambiguous signal: Figure 1(b) shows that late teacher layers exhibit relatively flattened confidence across tokens, which makes it hard to isolate knowledge-critical tokens from surface behaviors alone. **(2) Inaccurate knowledge transfer.** Even if a knowledge gap is identified, existing approaches usually decide where to distill by heuristically selecting layers (e.g., the last L layers or a fixed intermediate block). Such heuristics can be inaccurate because token-relevant knowledge is often layer-localized rather than uniformly distributed across the network. From Figure 1(b)–(c), intermediate teacher layers (e.g., 27 and 29) show stronger emphasis on *movies*, reflected by higher token-level focus and more activated neurons associated with this token. This mismatch suggests that distilling from arbitrary layers may miss the teacher circuits directly supporting the missing token, leading to weak or noisy supervision. Therefore, the teacher should be explicitly probed to localize knowledge-bearing layers, and neuron-level probing offers a direct way to reveal them for the targeted intermediate distillation.

Inspired by this observation, we introduce a novel **K**nowledge **P**robing-based white-box **D**istillation framework for LLMs, **KPD**, which probes both the student and teacher models to identify and distill knowledge precisely. First, to address knowledge unawareness in the student model, we propose an uncertainty-based probing mechanism that selects tokens with the highest prediction uncertainty. These tokens reveal the knowledge gaps in the student model and are used to construct a probing dataset that guides distillation. Second, to enable precise knowledge transfer, we perform attribution-guided probing on the teacher model. For each key token identified as uncertain by the student, we use integrated gradients to compute attribution scores for all neurons in each teacher layer, revealing which layers are most responsible for generating that token. Layers with high cumulative attribution are selected as knowledge-relevant layers, which likely store the missing knowledge needed by the student. To effectively transfer this knowledge, we extract the hidden representations from these layers and distill them into proportionally aligned student layers with an isometric mapping strategy. This alignment ensures that the student model receives hierarchical guidance at the appropriate depth.

We conduct extensive experiments on multiple instruction-tuned datasets and LLM families (Llama2, Llama3, and Qwen2.5) to validate the effectiveness of our framework. Results show that our KPD consistently enhances diverse KD baselines, yielding up to 4.56% improvement in Rouge-L. More experiments validate the effectiveness of our KPD in identifying knowledge gaps in students and locating the corresponding knowledge in teachers. Ablations further show that removing teacher-side probing leads to larger drops (up to 4.02% in Rouge-L) than removing student-side probing, confirming the importance of accurate layer localization in LLMs. Our contributions are as follows:

- We propose **KPD**, a novel white-box distillation framework for LLMs that performs joint probing on both student and teacher models to enable precise knowledge transfer across tasks, which can be incorporated into diverse distillation approaches to further boost them.
- We introduce a two-stage probing mechanism that first identifies the knowledge gaps in the student model via uncertainty estimation, and then locates knowledge-relevant layers in the teacher model using attribution scores based on integrated gradients for precise distillation.
- We demonstrate through extensive experiments and ablations that KPD improves various baseline distillation methods across multiple benchmarks and LLM families, achieving up to 4.56% improvement in Rouge-L and 7% improvement in accuracy across different domains.

2 Related Work

Knowledge Distillation. Knowledge distillation [28, 29, 30] is widely used for compressing LLMs [17, 18, 31, 32, 33, 34, 35, 36]. In generative LMs, methods are categorized as black- and white-box [18]. Black-box approaches fine-tune students with teacher-generated CoTs [20, 21, 22]. White-box distillation aligns teacher and student representations at the token [37, 38, 39] or sequence [40, 41] level. Subsequent work reduces exposure bias by mixing outputs [42], adds sequence-level f-divergence [43], and refines objectives via reverse KL [24], on-policy learning [23, 44, 45, 46, 47, 48, 49, 50, 51], or skewed KL off-policy updates [25]. To handle vocabulary mismatches, DSKD [52] and MultiLevelOT [53] align hidden states across tokenizers. However, prior works lack guidance on what and where to distill, which limits the distillation performance of generative language models.

Knowledge Probing. Knowledge probing analyzes how LLMs encode linguistic and factual information [54, 55]. Traditional methods train classifiers on labeled data to test whether properties such as syntax or semantics are linearly decodable from hidden states [56, 57, 58]. Recent work probes emergent capabilities truthful awareness [59], in-context knowledge [60], and safety alignment [61], showing that knowledge is sparsely distributed across layers and neurons [62, 63]. Another line treats feedforward networks (FFNs) as structured memory to localize factual knowledge [64]; integrated gradients identify knowledge neurons that strongly influence factual predictions [65]. In parallel, intervention-based methods such as amnesic probing [66] and LEACE [67] suppress access to specific properties by removing key representational directions. Building on these insights, we adopt a probing-driven distillation, where the probing guides both token selection and layer localization.

3 Methodology

3.1 Preliminary and Problem Statement

Supervised Fine-Tuning for Generative LMs. Let x and y represent the input and output sequences, respectively. Let $\mathcal{V}$ denote the vocabulary with $|\mathcal{V}|$ tokens. The output sequence up to the t-th token is denoted as $y_{<t} = (y_1, y_2, ..., y_{t-1})$, and the auto-regressive policy $p(\cdot|y_{<t}, x) \in (0,1)^{|\mathcal{V}|}$ produces a probability distribution over all tokens in $\mathcal{V}$, conditioned on the input x and previously generated output sequence $y_{<t}$. Thus, the probability of predicting the t-th token y_t is denoted as:

$$p(y_t|y_{<t}, x) = \frac{exp(z_t/\gamma)}{\sum_{i=1}^{|\mathcal{V}|} exp(z_i/\gamma)} \tag{1}$$

where z_t is the logit score of the token y_t, and γ is the temperature parameter. The model parameters can be optimized by minimizing the negative log-likelihood (NLL) loss over the ground-truth target sequence y. Given the dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$, the supervised fine-tuning loss can be defined as:

$$\mathcal{L}_{sft} = \mathbb{E}_{(x,y)\sim\mathcal{D}}[-\sum_{t=1}^{|y|} logp(y_t|y_{<t}, x)] \tag{2}$$

Knowledge Distillation for Generative LMs. We introduce the standard token-level knowledge distillation for generative LMs, aiming to minimize the scatter of the probability distributions of the outputs of the student model $\mathcal{S}$ and the teacher model $\mathcal{T}$. Given the dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$, for

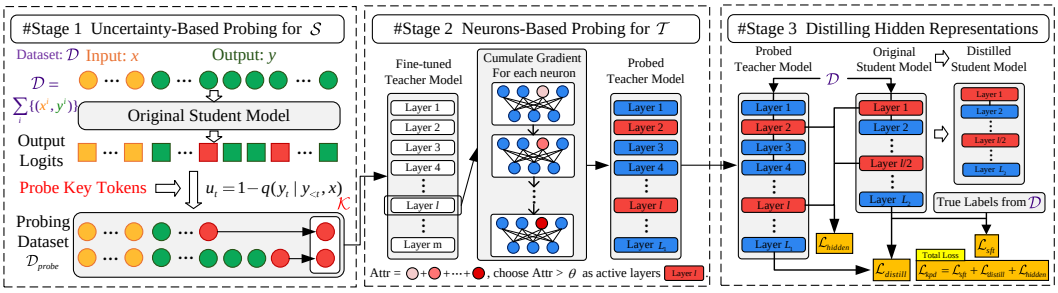


Figure 2: An overview of KPD framework.

each input-output pair $(x, y) \in \mathcal{D}$, we denote the distribution in the t-th generation step of $\mathcal{T}$ and $\mathcal{S}$ as $p(\cdot|y_{<t}, x)$ and $q(\cdot|y_{<t}, x)$, respectively. The distillation loss with Kullback-Leibler (KL) divergence is denoted as follows (more distillation losses for the generative LMs are listed in Appendix A.1):

$$\mathcal{L}_{distill} = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\sum_{t=1}^{|y|}\sum_{v\in\mathcal{V}}^{|\mathcal{V}|} p(v|y_{<t}, x) log \frac{p(v|y_{<t}, x)}{q(v|y_{<t}, x)}] \tag{3}$$

3.2 Framework

Figure 2 illustrates KPD, a three-stage distillation framework for precise knowledge transfer. First, we probe the student $\mathcal{S}$ via token-level predictive uncertainty and select highest-uncertainty tokens as key tokens. Second, we probe the teacher $\mathcal{T}$ with integrated gradients, computing neuron attributions to identify layers most responsible for those tokens. Third, we distill from these teacher layers to proportionally mapped student layers via an isometric mapping, minimizing KL between intermediate logits to teach the missing knowledge. This process allows KPD to focus distillation on what the student $\mathcal{S}$ needs and where the teacher $\mathcal{T}$ provides it, while preserving the depth-aware alignment.

3.3 Uncertainty-Based Probing for the Student Model

To obtain the knowledge that the student model $\mathcal{S}$ lacks, we first identify the missing knowledge in the student model $\mathcal{S}$ via uncertainty estimation, which is used to feed back to the teacher model for targeted instruction. Given a training dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$, for each input-output pair $(x, y) \in \mathcal{D}$, where x is the input and $y = (y_1, \ldots, y_{|y|})$ is the output sequence of length $|y|$, we compute the uncertainty u_t of the student model $\mathcal{S}$ for each generated token $y_t \in y$ as follows:

$$u_t = 1 - q(y_t|y_{<t}, x) \tag{4}$$

We select the top-k_1 tokens in y with the highest uncertainty scores to form a key token set $\mathcal{K}$. This results in a probing dataset:

$$\mathcal{D}_{probe} = \{(x^i, y^i, \mathcal{K}^i)\}_{i=1}^{|\mathcal{D}|} \tag{5}$$

where $\mathcal{K}^i$ is the key token set constructed by the i-th data (x^i, y^i). The probing dataset contains the tokens with the highest prediction uncertainty of the student model, highlighting the knowledge gaps the student still needs to learn. These tokens are used to probe the teacher model in the next step for attribution-guided teacher-layer localization. The construction details of $\mathcal{D}_{probe}$ are in Appendix A.6.

3.4 Neurons-Based Probing for the Teacher Model

In the probing for the teacher model, we aim to identify where the critical knowledge missing from the student model is stored within the teacher model, using the probing dataset $\mathcal{D}_{probe}$. This enables KPD to accurately and effectively transfer such knowledge to the student model during distillation.

We begin by decomposing the transformer-based teacher model $\mathcal{T}$ into L_1 transformer block layers $\{\mathcal{T}_1, \ldots, \mathcal{T}_{L_1}\}$, where each block includes a self-attention (SA) sub-layer, a feed-forward network (FFN) sub-layer, layer normalization, and residual connections. Then, for each $(x, y, \mathcal{K}) \in \mathcal{D}_{probe}$, to assess the importance of each layer in generating a key token $y_t \in \mathcal{K}$, we construct probing inputs

that isolate the token context. Specifically, for each probing example $(x, y, \mathcal{K})$, we construct probing examples of the form $(x + y_{<t}, y_t)$ for each key token $y_t \in \mathcal{K}$. This enables us to evaluate the influence of individual FFN neurons on the prediction of y_t, inspired by the Knowledge Neurons [65]. Let $w_{l,j}$ denote the j-th neuron in the FFN of layer $\mathcal{T}_l$, and let $\overline{w}_{l,j}$ denote its original parameter value in $\mathcal{T}$. The probability of generating token y_t given input x and partial output $y_{<t}$ when neuron $w_{l,j}$ is set to $\overline{w}_{l,j}$ is as follows:

$$\mathrm{P}_{x,y_t}(\overline{w}_{l,j}) = p(y_t|x, y_{<t}, w_{l,j} = \overline{w}_{l,j}) \tag{6}$$

To estimate the attribution of neuron $w_{l,j}$ to this prediction, we apply the integrated gradient to scale $w_{l,j}$ from 0 to $\overline{w}_{l,j}$, and approximate the gradient integral $\mathrm{Attr}(w_{l,j})$ via Riemann Summation [68]:

$$\mathrm{Attr}(w_{l,j}) = \overline{w}_{l,j} \int_0^1 \frac{\partial \mathrm{P}_{x,y_t}(\alpha \overline{w}_{l,j})}{\partial w_{l,j}} \mathrm{d}\alpha \approx \frac{\overline{w}_{l,j}}{s} \sum_{k=1}^{s} \frac{\partial \mathrm{P}_{x,y_t}(\frac{k}{s}\overline{w}_{l,j})}{\partial w_{l,j}} \tag{7}$$

where s denotes the optimization step. We consider a neuron $w_{l,j}$ to be activated if $\mathrm{Attr}(w_{l,j}) > \theta$, where θ is a predefined activation threshold. The number of activated neurons in each layer $\mathcal{T}_l$, denoted N_l, reflects the contribution of the layer to the prediction of the key token. A higher N_l indicates that more key knowledge is stored within the corresponding teacher layer for distillation.

After performing this probing process over all examples in $\mathcal{D}_{probe}$, we aggregate the activation counts N_l for each layer $\mathcal{T}_l$. We then select the top-k_2 layers (excluding the last layer, since it has been used in $\mathcal{L}_{distill}$) with the highest total activations to form the key knowledge layer set $\mathcal{Q}^{\mathcal{T}} = \{\mathcal{T}_{a_1}, \mathcal{T}_{a_2}, ..., \mathcal{T}_{a_{k_2}}\}$, where $a_1, \cdots, a_{k_2}$ are the ids of the top-k_2 layers. This final set $\mathcal{Q}^{\mathcal{T}}$ indicates the teacher model layers that are most responsible for storing critical knowledge absent in the student model $\mathcal{S}$ for subsequent intermediate-layer knowledge transfer. The algorithm process for the probing is shown in Appendix A.3, and the visualization of the probing is shown in Appendix B.13.

3.5 Intermediate Representation Distillation

In order to accurately extract key knowledge from the teacher model $\mathcal{T}$ to the student model $\mathcal{S}$, we distill the layers that store key knowledge in $\mathcal{Q}^{\mathcal{T}}$ to the student model $\mathcal{S}$. Specifically, let the hidden states from the teacher model be $\mathbf{H}^{\mathcal{T}} = \{\mathbf{h}^{\mathcal{T}}_{a_1}, \ldots, \mathbf{h}^{\mathcal{T}}_{a_{k_2}}\}$ of $\mathcal{Q}^{\mathcal{T}}$. Similar to $\mathcal{T}$, we decompose the transformer-based student model $\mathcal{S}$ into L_2 transformer block layers $\{\mathcal{S}_1, \ldots, \mathcal{S}_{L_2}\}$. In order to allow the student model $\mathcal{S}$ to better learn the representations of different depth layers in the teacher model $\mathcal{T}$, we define an isometric mapping strategy to assign the corresponding layers $\mathcal{Q}^{\mathcal{S}}$ in $\mathcal{S}$ as follows:

$$\mathcal{Q}^{\mathcal{S}} = [\mathcal{S}_{c_1}, \mathcal{S}_{c_2}, ..., \mathcal{S}_{c_{k_2}}] \tag{8}$$

where $c_j = \lceil a_j / \lceil \frac{L_1}{L_2} \rceil \rceil$ for $j \in [1, k_2]$. This mapping ensures that deep teacher layers are projected proportionally to deeper student layers, preserving hierarchical semantic abstraction. Let $\mathbf{H}^{\mathcal{S}} = \{\mathbf{h}^{\mathcal{S}}_{c_1}, \ldots, \mathbf{h}^{\mathcal{S}}_{c_{k_2}}\}$ be the hidden states of the student model from $\mathcal{Q}^{\mathcal{S}}$. For the student model $\mathcal{S}$ and the teacher model $\mathcal{T}$, we map the hidden states of $\mathcal{S}$ and $\mathcal{T}$ with two $|\mathcal{V}| \times d$ linear layers $W^{\mathcal{S}}$ and $W^{\mathcal{T}}$, respectively, where d denotes the hidden size of the last dim in hidden states. Specifically, we use the output projection layer of the original models as $W^{\mathcal{S}}$ and $W^{\mathcal{T}}$ for $\mathcal{S}$ and $\mathcal{T}$, respectively, to maintain consistency with the vocabulary-level logits. Then, the output logits in the a_j-th layer of $\mathcal{T}$ and in the c_j-th layer of $\mathcal{S}$ can be denoted as $\mathbf{z}^{\mathcal{T}}_j = W^{\mathcal{T}}\mathbf{h}^{\mathcal{T}}_{a_j}$ and $\mathbf{z}^{\mathcal{S}}_j = W^{\mathcal{S}}\mathbf{h}^{\mathcal{S}}_{c_j}$, respectively. Then, the probability of predicting the t-th token in the a_j-th layer of $\mathcal{T}$ can be denoted as $p_j(y_t|y_{<t}, x)$:

$$p_j(y_t|y_{<t}, x) = \frac{exp(z^{\mathcal{T}}_{j,t}/\gamma)}{\sum_{i=1}^{|\mathcal{V}|} exp(z^{\mathcal{T}}_{j,i}/\gamma)} \tag{9}$$

where $z^{\mathcal{T}}_{j,t} \in \mathbf{z}^{\mathcal{T}}_j$ is the hidden logit score of y_t in the j-th layer of the teacher $\mathcal{T}$. Similarly, we denote the probability of predicting the t-th token in the aligned c_j-th layer of the student $\mathcal{S}$ as $q_j(y_t|y_{<t}, x)$:

$$q_j(y_t|y_{<t}, x) = \frac{exp(z^{\mathcal{S}}_{j,t}/\gamma)}{\sum_{i=1}^{|\mathcal{V}|} exp(z^{\mathcal{S}}_{j,i}/\gamma)} \tag{10}$$

where $z^{\mathcal{S}}_{j,t} \in \mathbf{z}^{\mathcal{S}}_j$ is the hidden logit score of y_t in the j-th layer of the student $\mathcal{S}$. Finally, the hidden loss is calculated as follows:

$$\mathcal{L}_{hidden} = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\frac{1}{k_2}\sum_{j=1}^{k_2}\sum_{t=1}^{|y|}\sum_{v\in\mathcal{V}}^{|\mathcal{V}|} p_j(v|y_{<t}, x) log \frac{p_j(v|y_{<t}, x)}{q_j(v|y_{<t}, x)}] \tag{11}$$

Table 1: Main results of Llama3.2-1B and Qwen2.5-1.5B with Rouge-L on domain-agnostic datasets.

Llama3.1-8B → Llama3.2-1B							Qwen2.5-7B → Qwen2.5-1.5B						
Methods	**Dolly**	**SelfInst**	**S-NI**	**UnNI**	**Vicuna**	**Avg.**	**Methods**	**Dolly**	**SelfInst**	**S-NI**	**UnNI**	**Vicuna**	**Avg.**
Teacher Model	31.21	25.17	38.17	37.72	19.82	30.42	Teacher Model	31.31	27.27	44.17	40.04	24.85	33.53
SFT	23.94	14.73	28.63	27.01	15.92	22.05	SFT	23.64	20.13	38.70	33.75	17.49	26.74
KD	25.72	15.19	29.06	28.19	16.19	22.87	KD	23.20	18.39	37.19	31.75	19.59	26.02
w/ KPD	**26.43**	**16.57**	**30.12**	**29.47**	**17.33**	**23.98**	**w/ KPD**	**27.76**	**19.14**	**38.93**	**32.25**	**20.26**	**27.67**
RKD	24.21	17.48	33.92	34.48	18.06	25.63	RKD	25.24	17.25	35.65	35.58	20.87	26.92
w/ KPD	**25.54**	**17.97**	**34.49**	**36.19**	**19.44**	**26.72**	**w/ KPD**	**27.54**	**19.97**	**36.49**	**36.19**	**21.44**	**28.32**
SeqKD	26.47	16.08	30.57	31.06	16.42	24.12	SeqKD	28.13	19.63	36.30	34.81	19.41	27.66
w/ KPD	**28.77**	**16.74**	**31.26**	**32.22**	**17.91**	**25.38**	**w/ KPD**	**29.11**	**20.38**	**37.77**	**36.47**	**20.37**	**28.82**
ImitKD	25.73	15.81	26.71	27.31	17.95	22.70	ImitKD	24.58	17.67	34.41	33.56	18.82	25.81
w/ KPD	**26.15**	**17.15**	**28.45**	**28.79**	**18.11**	**23.73**	**w/ KPD**	**26.69**	**18.42**	**35.65**	**34.48**	**19.82**	**27.01**
GKD	26.95	16.60	31.76	30.61	17.06	24.60	GKD	29.28	22.33	42.16	37.19	22.57	30.71
w/ KPD	**27.99**	**17.00**	**32.33**	**31.56**	**17.42**	**25.26**	**w/ KPD**	**30.12**	**23.60**	**42.32**	**38.70**	**22.96**	**31.54**
MiniLLM	28.25	18.10	34.36	33.41	17.61	26.35	MiniLLM	30.24	23.12	42.33	39.59	23.79	31.81
w/ KPD	**28.67**	**18.80**	**34.80**	**33.95**	**18.14**	**26.87**	**w/ KPD**	**31.01**	**23.94**	**42.90**	**40.02**	**24.13**	**32.40**
Distillm	28.33	19.23	37.60	35.65	18.24	27.81	Distillm	29.38	23.84	41.30	38.96	21.61	31.02
w/ KPD	**28.79**	**19.58**	**37.96**	**36.02**	**18.86**	**28.24**	**w/ KPD**	**29.94**	**24.27**	**41.91**	**39.62**	**22.28**	**31.61**
Distillm-2	28.42	19.32	37.68	35.74	18.41	27.91	Distillm-2	29.43	23.99	41.54	39.11	21.86	31.19
w/ KPD	**28.86**	**19.59**	**38.04**	**36.11**	**18.94**	**28.31**	**w/ KPD**	**29.98**	**24.52**	**42.06**	**39.97**	**22.34**	**31.77**
ABKD	28.50	19.28	37.59	35.83	18.42	27.92	ABKD	29.62	24.16	41.63	39.22	21.94	31.31
w/ KPD	**28.92**	**19.52**	**37.99**	**36.14**	**18.88**	**28.29**	**w/ KPD**	**30.06**	**24.60**	**42.14**	**40.06**	**22.43**	**31.86**

The final total loss $\mathcal{L}_{kpd}$ is calculated as follows:

$$\mathcal{L}_{kpd} = \mathcal{L}_{sft} + \mathcal{L}_{distill} + \mathcal{L}_{hidden} \tag{12}$$

4 Experiments

4.1 Experimental Settings

Datasets. We conduct experiments on domain-agnostic and domain-specific datasets. For domain-agnostic datasets, we distill on Databricks-Dolly-15k (**Dolly**) [69], and test on **Dolly**, Self-Instruct (**SelfInst**) [70], Vicuna Evaluation (**Vicuna**) [71], Super-Natural Instructions (**S-NI**) [72], and Unnatural Instructions (**UnNI**) [73]. For domain-specific datasets, we proceed as follows: (1) **Medical**: we distill on GenMedGPT [74] and evaluate on MMLU [75] medical tasks: clinical knowledge (**CK**), professional medicine (**PM**), and college medicine (**CM**). (2) **Legal**: we distill on Lawyer-Instruct [76] and USTerms [77] and evaluate on MMLU law tasks: international law (**IL**) and professional law (**PL**). More details and an introduction to the datasets are listed in Appendix A.2.

Models. For student models, we choose TinyLlama-1.1B [78], Llama3.2-1B [79], and Qwen2.5-1.5B [80]. For teacher models, we choose Llama2-7B [81], Llama3.1-8B [79], and Qwen2.5-7B [80]. We conduct three distillation experiments: from Llama2-7B to TinyLlama-1.1B, from Llama3.1-8B to Llama3.2-1B, and from Qwen2.5-7B to Qwen2.5-1.5B across different representative model families.

Baselines. We apply our KPD framework to nine distillation methods for LLMs under the same setting: KD [29], RKD [24], SeqKD [40], ImitKD [42], GKD [23], MiniLLM [24], Distillm [25], Distillm-2 [33], and ABKD [34]. More details of the distillation baselines are listed in Appendix A.4.

Settings. When performing distillation on each dataset, we first fine-tune the teacher model and then distill the student model by the fine-tuned teacher model. We evaluate all methods with Rouge-L and GPT-4 scores on domain-agnostic datasets, and accuracy scores on domain-specific datasets. For each baseline, we evaluate both its original method and the method augmented with our KPD. All the results are the average of 5 runs with random seeds. More details of the settings are in Appendix A.5. Our code is available at https://github.com/ljj-007/KPD.

Table 2: Main results of Llama3.2-1B and Qwen2.5-1.5B with accuracy on domain-specific datasets.

Llama3.1-8B → Llama3.2-1B							Qwen2.5-7B → Qwen2.5-1.5B						
Methods	**CK**	**PM**	**CM**	**PL**	**IL**	**Avg.**	**Methods**	**CK**	**PM**	**CM**	**PL**	**IL**	**Avg.**
Teacher Model	0.59	0.78	0.57	0.44	0.63	0.60	Teacher Model	0.73	0.79	0.65	0.55	0.82	0.71
SFT	0.33	0.24	0.31	0.22	0.18	0.26	SFT	0.65	0.53	0.57	0.43	0.49	0.53
KD	0.36	0.26	0.33	0.27	0.25	0.29	KD	0.65	0.60	0.60	0.44	0.61	0.58
w/ KPD	**0.41**	**0.31**	**0.36**	**0.31**	**0.29**	**0.34**	**w/ KPD**	**0.68**	**0.63**	**0.62**	**0.46**	**0.65**	**0.61**
RKD	0.38	0.31	0.35	0.28	0.27	0.32	RKD	0.63	0.66	0.60	0.47	0.66	0.60
w/ KPD	**0.44**	**0.34**	**0.40**	**0.30**	**0.31**	**0.36**	**w/ KPD**	**0.74**	**0.69**	**0.63**	**0.50**	**0.72**	**0.66**
SeqKD	0.35	0.29	0.34	0.29	0.26	0.31	SeqKD	0.72	0.61	0.62	0.47	0.65	0.61
w/ KPD	**0.38**	**0.33**	**0.36**	**0.33**	**0.31**	**0.34**	**w/ KPD**	**0.73**	**0.70**	**0.68**	**0.49**	**0.67**	**0.65**
ImitKD	0.38	0.34	0.35	0.26	0.25	0.32	ImitKD	0.64	0.63	0.61	0.48	0.58	0.59
w/ KPD	**0.45**	**0.35**	**0.39**	**0.29**	**0.29**	**0.35**	**w/ KPD**	**0.70**	**0.66**	**0.68**	**0.51**	**0.65**	**0.64**
GKD	0.39	0.29	0.32	0.29	0.27	0.31	GKD	0.66	0.64	0.62	0.51	0.61	0.61
w/ KPD	**0.43**	**0.33**	**0.35**	**0.33**	**0.31**	**0.35**	**w/ KPD**	**0.72**	**0.69**	**0.71**	**0.53**	**0.68**	**0.67**
MiniLLM	0.37	0.28	0.33	0.27	0.24	0.30	MiniLLM	0.71	0.65	0.62	0.46	0.63	0.61
w/ KPD	**0.40**	**0.34**	**0.38**	**0.33**	**0.34**	**0.36**	**w/ KPD**	**0.74**	**0.71**	**0.72**	**0.49**	**0.73**	**0.68**
Distillm	0.41	0.31	0.35	0.26	0.26	0.32	Distillm	0.74	0.61	0.60	0.48	0.62	0.61
w/ KPD	**0.46**	**0.35**	**0.37**	**0.31**	**0.32**	**0.36**	**w/ KPD**	**0.77**	**0.68**	**0.65**	**0.51**	**0.74**	**0.67**
Distillm-2	0.43	0.28	0.36	0.28	0.28	0.33	Distillm-2	0.72	0.68	0.61	0.50	0.63	0.63
w/ KPD	**0.49**	**0.34**	**0.41**	**0.34**	**0.35**	**0.39**	**w/ KPD**	**0.75**	**0.70**	**0.64**	**0.52**	**0.72**	**0.67**
ABKD	0.42	0.32	0.34	0.28	0.26	0.32	ABKD	0.72	0.68	0.64	0.49	0.68	0.64
w/ KPD	**0.48**	**0.36**	**0.38**	**0.35**	**0.31**	**0.38**	**w/ KPD**	**0.74**	**0.71**	**0.66**	**0.52**	**0.73**	**0.67**

4.2 Main Results

Domain-Agnostic Results. To test KPD on domain-agnostic settings, we perform knowledge distillation from Llama3.1-8B to Llama3.2-1B, Qwen2.5-7B to Qwen2.5-1.5B in Table 1 (Llama2-7B to TinyLlama in Appendix B.1). Our KPD method consistently improves baselines like KD, RKD, and SeqKD across models and datasets. For instance, on Llama3.2-1B, KPD boosts SeqKD by +1.26↑ and KD by +1.12↑. Similar improvements are seen on Qwen2.5-1.5B, with KD increasing by +1.65↑. The gains from KPD diminish as the base model improves, suggesting it is most beneficial for weaker models. KPD also shows strong generalization, improving performance across tasks even when trained on Dolly data alone, with notable boosts on Llama3.2-1B and Qwen2.5-1.5B (e.g., SeqKD increases by +2.30↑ on Dolly). These results confirm that KPD enhances distillation by capturing transferable knowledge across different models without the need for dataset-specific tuning. Moreover, we test with GPT-4 scores. For each dataset, we sample 100 samples. For each input, we generate two answers from the baseline without KPD and from the baseline augmented with KPD, and submit the pair to `gpt-4-turbo` to choose the better output. Figure 3 shows that KPD is preferred in the majority of cases. On Llama3.2-1B and Qwen2.5-1.5B, KPD wins for most baselines, where ties are small (about 6-14%), and losses fall in the 27-40% range, indicating that KPD improves performance across baselines.

Domain-Specific Results. To test KPD on domain-specific settings, we perform distillation on the domain-specific datasets. Table 2 reports accuracy on medical (CK, PM, CM) and legal (PL, IL) datasets of Llama3.2-1B and Qwen2.5-1.5B (results of TinyLlama in Appendix B.1). Across all baselines, adding KPD raises the Avg. scores, with typical gains of 2%-4% on Llama3.2-1B and 3%-6% on Qwen2.5-1.5B, and the largest increases often occur on knowledge-intensive subsets such as PM and IL. This demonstrates that our approach also performs exceptionally well on the domain-specific tasks.

4.3 Ablation Results

We ablate four representative baselines: two classic (KD, SeqKD) and two LLM-designed (GKD, Distillm) on Llama3.2-1B in Table 3 and Qwen2.5-1.5B in Appendix B.2. **w/o student-probe** replaces uncertainty tokens with random ones, and **w/o teacher-probe** replaces layer selection with

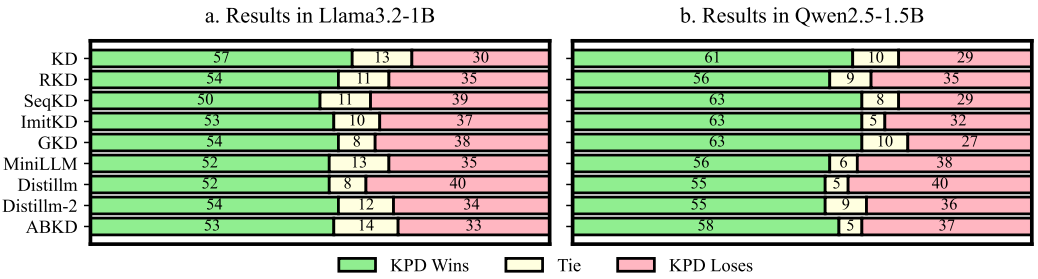


Figure 3: Main results of Llama3.2-1B and Qwen2.5-1.5B with GPT-4 scores on one hundred random samples of the domain-agnostic datasets.

Table 3: Ablation results on Llama3.2-1B (Rouge-L).

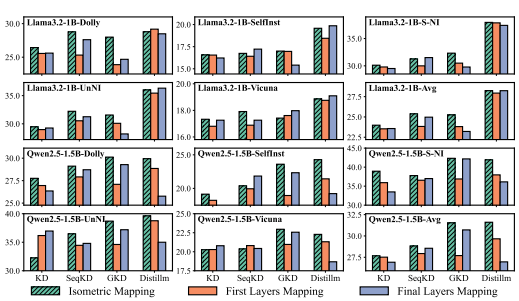
Methods	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg.
Llama3.1-8B → Llama3.2-1B						
KD	**26.43**	**16.57**	**30.12**	**29.47**	**17.33**	**23.98**
w/o student-probe	25.61	15.30	27.79	28.85	16.11	22.73
w/o teacher-probe	24.30	13.88	26.12	27.18	14.88	21.27
SeqKD	**28.77**	**16.74**	**31.26**	**32.22**	**17.91**	**25.38**
w/o student-probe	26.80	15.53	29.08	30.97	14.53	22.58
w/o teacher-probe	25.63	13.93	27.98	29.86	13.11	21.30
GKD	**27.99**	**17.00**	**32.33**	**31.56**	**17.42**	**25.26**
w/o student-probe	24.88	16.89	30.19	29.63	16.65	23.65
w/o teacher-probe	23.46	15.23	28.67	28.11	15.32	22.16
Distillm	**28.79**	**19.58**	**37.96**	**36.02**	**18.86**	**28.24**
w/o student-probe	28.29	18.08	35.56	34.62	18.26	26.96
w/o teacher-probe	27.18	16.75	34.22	33.50	16.84	25.70

random layers. We observe that: (1) Removing either probe consistently hurts: on Llama3.2-1B, KD drops 23.98→21.27 (2.71↓) and SeqKD 25.38→21.30 (4.08↓). (2) Teacher probing matters more: on Qwen2.5-1.5B with GKD the drop is 1.78↓ (w/o student-probe) vs. 2.30↓ (w/o teacher-probe) and on Llama3.2-1B with SeqKD, 2.80↓ vs. 4.08↓. It indicates that aligning uncertain tokens with high-attribution teacher layers is essential for precise knowledge transfer during distillation processes.

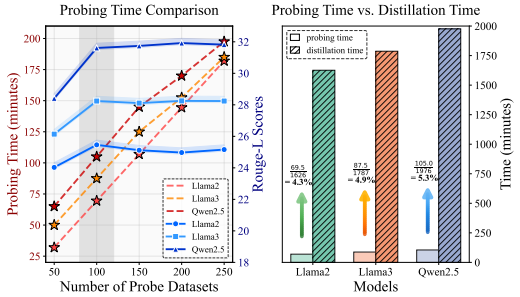
4.4 Analysis Results

Layer Mapping Strategy. To investigate the impact of layer mapping during distillation, we evaluate three mapping strategies: isometric mapping, front-layer mapping, and final-layer mapping. We control the layer number k_2 to 3, and the results are shown in Figure 4a. First, isometric mapping consistently achieves the best performance on average across all methods and both model families. It shows that proportionally distributing knowledge across student layers facilitates more effective knowledge transfer and deeper representation alignment. Second, final-layer mapping generally performs better than front-layer mapping (e.g., 28.55↑ vs. 27.94↓ in seqKD of Qwen2.5-1.5B), showing that injecting knowledge into deeper student layers is more effective. However, it still underperforms isometric mapping, suggesting that isometric mapping balances knowledge in layers.

Probing Efficiency. To study the efficiency of probing, first, we analyze the scale of probing datasets to find the best probing scale. As shown in Figure 4b (left), probing time increases roughly linearly with the number of probe examples, while performance saturates once the probe set reaches 100. Concretely, Rouge-L at 100 vs. 250 examples changes only marginally (0.2-0.3 across all model families), indicating that a small scale of probing dataset (100) already achieves near-optimal gains. Second, we analyze the probing costs during the whole distillation process. Figure 4b (right) compares the extra probing time (the scale of probing datasets is 100) with the average distillation time on domain-agnostic datasets. The additional probing cost is modest across model families, constituting only 4.3% (Llama2), 4.9% (Llama3), and 5.3% (Qwen2.5) of total wall-clock time. Thus, the probing introduces only negligible additional training overhead while preserving performance.

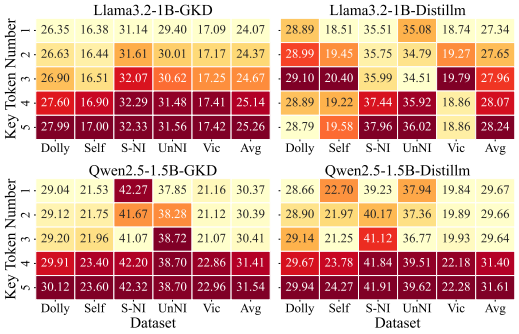


(a) Results of three mapping strategies: (1) **Isometric Mapping** aligns student layers proportionally with selected teacher layers; (2) **Front-Layer Mapping** uses only the first several student layers; (3) **Final-Layer Mapping** utilizes only the last several student layers.

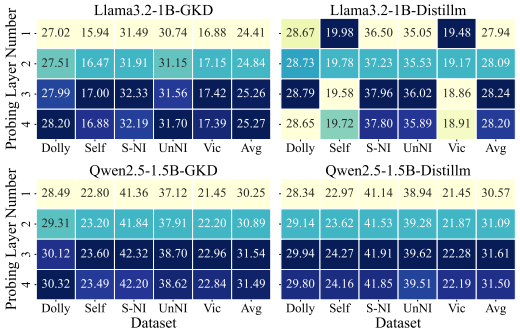


(b) Analysis of probing efficiency: (1) Probing Scale shows how the number of probing examples affects probing time and Rouge-L performance; (2) Time Comparison compares the additional probing time with the average distillation time across different LLM families.

Figure 4: Analysis of layer mapping strategies and probing efficiency.



(a) Results of probing token number k_1 from 1 to 5. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

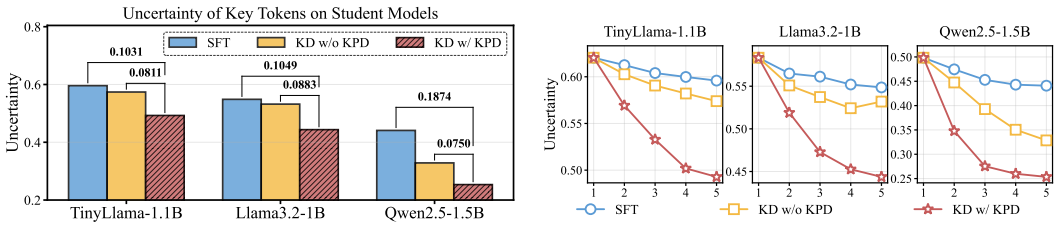


(b) Results of probing layer number k_2 from 1 to 4. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

Figure 5: Analysis of the effect of probing token number k_1 and probing layer number k_2.

Number of Probing Tokens k_1. To study the impact of the scale of probing tokens, we adjust the number of key tokens k_1 by varying it from 1 to 5, and results are shown in Figure 5a and Appendix B.4. Performance generally improves with more key tokens probed. For example, in GKD with Llama3.2-1B, the average score increases from 24.07 to 25.26 (k_1=1 to 5), indicating that uncertain tokens carry valuable knowledge. This trend suggests that expanding the probing set increases the coverage of missing knowledge cues, thereby providing denser and more informative supervision signals during distillation. However, performance saturates around $k_1 = 4$ or 5, with occasional drops (e.g., in Distillm on Dolly and Vicuna), suggesting over-probing is not beneficial for distillation. A plausible reason is that adding lower-uncertainty tokens introduces noisy or redundant constraints, which may dilute the gradients on truly knowledge-critical tokens. In practice, setting k_1 to 4-5 achieves a trade-off between effectiveness and stability during the distillation of LLMs.

Number of Probing Layers k_2. To understand the influence of the number of probed teacher layers, we explore various values of k_2 ranging from 1 to 4 as shown in Figure 5b and Appendix B.5. At the early stage, increasing k_2 leads to noticeable performance gains, as more teacher layers contribute complementary knowledge signals. For example, in GKD with Llama3.2-1B, the average score improves from 24.41 ($k_2 = 1$) to 25.26 ($k_2 = 3$). This indicates that aggregating information from multiple layers enhances the richness and robustness of knowledge transfer. However, further increasing k_2 beyond 3 offers marginal or even negative returns. For instance, Distillm with Llama3.2-1B peaks at 28.24 when $k_2 = 3$, but slightly drops to 28.20 at $k_2 = 4$. This suggests that including too many layers may introduce overlapping or less-relevant representations, potentially diluting critical knowledge during distillation. Therefore, a moderate number of teacher layers ($k_2 = 3$) offers a good balance between knowledge diversity and transfer quality during the operation of our KPD.



(a) Results of the average uncertainty of key tokens in probing datasets across different student models.

(b) Results of the average uncertainty of key tokens in probing datasets during training epochs (the x-axis).

Figure 6: Analysis of the uncertainty of key tokens in the probing datasets.

Table 4: Intervention analysis on teacher layers. We report the average log-probability drop on key tokens and the relative degradation compared with random layers.

Intervention Target	Avg. Log-Prob Drop	Relative Degradation
Top-k_2 selected layers (ours)	**0.47**	**4.3**×
Random layers	0.11	1.0×
Low-activation layers	0.16	1.5×

Uncertainty Reduction of Key Tokens. To test whether KPD targets the weakest knowledge of the student model, we measure the average uncertainty of key tokens in the probing datasets with and without KPD. As shown in Figure 6a, KPD consistently lowers uncertainty across students. For example, KPD reduces uncertainty by 0.1031–0.1874 compared with SFT and by 0.0750–0.0883 compared with vanilla KD. Notably, the uncertainty drop is larger when compared to vanilla KD, suggesting that merely mimicking teacher outputs is insufficient to reliably resolve the hardest tokens of the student without explicit gap-aware probing. As shown in Figure 6b, across all model series, KD augmented with KPD reduces the average uncertainty more quickly and converges to a consistently lower final level than both SFT and vanilla KD. This faster reduction implies that KPD provides earlier and stronger corrective signals on knowledge-critical tokens. These reductions indicate that KPD concentrates supervision on the tokens the student initially finds most uncertain, yielding precise knowledge distillation for LLMs.

Intervention Analysis on Teacher-Layer Relevance. KPD selects teacher layers according to the number of activated neurons identified by integrated gradients. A natural question is whether these selected layers are genuinely more relevant to the key tokens than randomly chosen or low-activation layers. To answer this question, we conduct an intervention-style analysis on the teacher model. For each probing example, we compare three intervention targets: (i) the top-k_2 layers selected by KPD, (ii) randomly selected layers with the same number of layers, and (iii) low-activation layers with the smallest attribution-based activation counts. We then intervene on these layers and measure the average drop in the log-probability of the teacher on the key tokens. A larger drop indicates that the intervened layers are more important for supporting the target-token prediction. As shown in Table 4, intervening on the layers selected by KPD causes a much larger drop in the teacher key-token log-probability than intervening on random or low-activation layers. This indicates that the selected layers are not arbitrary high-activity layers, but are substantially more relevant to the prediction of the student-uncertain tokens. Together with the ablation results in Appendix B.2, this provides direct empirical evidence that teacher-side probing identifies the depth-sensitive knowledge sources.

5 Conclusion

In this work, we propose KPD, a white-box distillation framework for LLMs that enables precise knowledge transfer from specific layers of the teacher model to the student model. By identifying weak predictions and distilling from knowledge-relevant teacher layers, KPD achieves more effective representation alignment. Experiments on various datasets show consistent improvements over strong baselines, highlighting the value of knowledge-aware and structure-guided distillation for LLMs.

Acknowledgements

This work was supported by National Science Foundation of China (Grant Nos.62376057), SEU Innovation Capability Enhancement Plan for Doctoral Students (No. CXJH_SEU 25131), the Fundamental Research Funds for the Central Universities (2242025K30024), the Start-up Research Fund of Southeast University (RF1028623234), and the Big Data Computing Center of Southeast University. All opinions are of the authors and do not reflect the view of sponsors.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

[4] Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, et al. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*, 2024.

[5] Mushui Liu, Yuhang Ma, Zhen Yang, Jun Dan, Yunlong Yu, Zeng Zhao, Zhipeng Hu, Bai Liu, and Changjie Fan. Llm4gen: Leveraging semantic representation of llms for text-to-image generation. In *AAAI*, 2025.

[6] Aryaz Eghbali and Michael Pradel. De-hallucinator: Iterative grounding for llm-based code completion. *arXiv preprint arXiv:2401.01701*, 2024.

[7] Chong Wang, Kaifeng Huang, Jian Zhang, Yebo Feng, Lyuye Zhang, Yang Liu, and Xin Peng. How and why llms use deprecated apis in code completion? an empirical study. *arXiv preprint arXiv:2406.09834*, 2024.

[8] Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools. In *NeurIPS*, 2023.

[9] Joao Monteiro, Pierre-Andre Noel, Etienne Marcotte, Sai Rajeswar Mudumba, Valentina Zantedeschi, David Vazquez, Nicolas Chapados, Chris Pal, and Perouz Taslakian. Repliqa: A question-answering dataset for benchmarking llms on unseen reference content. In *NeurIPS*, 2024.

[10] Zijie Xu, Wenjun Ke, Peng Wang, Guozheng Li, Qingjian Ni, Jiajun Liu, Ziyu Shang, and Jing Zhou. Benchmarking and enhancing rule knowledge-driven reasoning of large language models. In *AAAI*, 2026.

[11] Hongbo Li and Lingjie Duan. Theory of mixture-of-experts for mobile edge computing. In *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2025.

[12] Lyudong Jin, Yanning Zhang, Yanhan Li, Shurong Wang, Howard H Yang, Jian Wu, and Meng Zhang. Moe2: Optimizing collaborative inference for edge large language models. *arXiv preprint arXiv:2501.09410*, 2025.

[13] Yichuan Li, Xiyao Ma, Sixing Lu, Kyumin Lee, Xiaohu Liu, and Chenlei Guo. Mend: Meta demonstration distillation for efficient and effective in-context learning. In *ICLR*, 2024.

[14] Mara Finkelstein and Markus Freitag. Mbr and qe finetuning: Training-time distillation of the best and most expensive decoding methods. In *ICLR*, 2024.

[15] Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. Da-kd: Difficulty-aware knowledge distillation for efficient large language models. In *ICML*, 2025.

[16] Kaituo Feng, Changsheng Li, Xiaolu Zhang, Jun Zhou, Ye Yuan, and Guoren Wang. Keypoint-based progressive chain-of-thought distillation for llms. In *ICML*, 2024.

[17] Qihuang Zhong, Liang Ding, Li Shen, Juhua Liu, Bo Du, and Dacheng Tao. Revisiting knowledge distillation for autoregressive language models. In *ACL*, 2024.

[18] Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

[19] Wei Wang, Zhaowei Li, Qi Xu, Yiqing Cai, Hang Song, Qi Qi, Ran Zhou, Zhida Huang, Tao Wang, and Li Xiao. Qcrd: Quality-guided contrastive rationale distillation for large language models. In *EMNLP*, 2025.

[20] Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of ACL*, 2023.

[21] Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In *ACL*, 2023.

[22] Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

[23] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *ICLR*, 2024.

[24] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *ICLR*, 2024.

[25] Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. In *ICML*, 2024.

[26] Ziqi Zhang, Ali Shahin Shamsabadi, Hanxiao Lu, Yifeng Cai, and Hamed Haddadi. Membership and memorization in llm knowledge distillation. In *EMNLP*, 2025.

[27] Faisal Hamman, Pasan Dissanayake, Yanjun Fu, and Sanghamitra Dutta. Few-shot knowledge distillation of llms with counterfactual explanations. In *NeurIPS*, 2025.

[28] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, 2006.

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

[30] Chengming Hu, Xuan Li, Dan Liu, Haolun Wu, Xi Chen, Ju Wang, and Xue Liu. Teacher-student architecture for knowledge distillation: A survey. *arXiv preprint arXiv:2308.04268*, 2023.

[31] Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. Taid: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. In *ICLR*, 2025.

[32] Lin Sun, Guangxiang Zhao, Xiaoqi Jian, Yuhan Wu, Weihong Lin, Yongfu Zhu, Linglin Zhang, Jinzhu Wu, Junfeng Ran, Sai-er Hu, et al. Tinyr1-32b-preview: Boosting accuracy with branch-merge distillation. *arXiv preprint arXiv:2503.04872*, 2025.

[33] Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms. In *ICML*, 2025.

[34] Guanghui Wang, Zhiyong Yang, Zitai Wang, Shi Wang, Qianqian Xu, and Qingming Huang. Abkd: Pursuing a proper allocation of the probability mass in knowledge distillation via α-β-divergence. In *ICML*, 2025.

[35] Tianyu Peng and Jiajun Zhang. Enhancing knowledge distillation of large language models through efficient multi-modal distribution alignment. In *COLING*, 2025.

[36] Jiajun Liu, Yao He, Wenjun Ke, Peng Wang, Ziyu Shang, Guozheng Li, and Zijie Xu. Balanced knowledge distillation for large language models with mix-of-experts. In *AAAI*, 2026.

[37] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[38] Yeongmin Kim, Donghyeok Shin, Mina Kang, Byeonghu Na, and Il-Chul Moon. Distillation of large language models via concrete score matching. *arXiv preprint arXiv:2509.25837*, 2025.

[39] Thinh On, Hengzhi Pei, Leonard Lausen, and George Karypis. Knowledge distillation for large language models through residual learning. In *ICLR*, 2026.

[40] Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *EMNLP*, 2016.

[41] Jingxuan Wei, Linzhuang Sun, Yichong Leng, Xu Tan, Bihui Yu, and Ruifeng Guo. Sentence-level or token-level? a comprehensive study on knowledge distillation. In *IJCAI*, 2024.

[42] Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *EMNLP*, 2020.

[43] Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. f-divergence minimization for sequence-level knowledge distillation. In *ACL*, 2023.

[44] Zhen Tan, Chengshuai Zhao, Song Wang, Jundong Li, Tianlong Chen, et al. Probing to refine: Reinforcement distillation of llm reasoners via explanatory inversion. In *ICLR*, 2026.

[45] Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Yuanheng Zhu, and Dongbin Zhao. Revisiting on-policy distillation: Empirical failure modes and simple fixes. *arXiv preprint arXiv:2603.25562*, 2026.

[46] Wenhong Zhu, Ruobing Xie, Rui Wang, and Pengfei Liu. Hybrid policy distillation for llms. *arXiv preprint arXiv:2604.20244*, 2026.

[47] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

[48] Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr. *arXiv preprint arXiv:2604.03128*, 2026.

[49] Jongwoo Ko, Sara Abdali, Young Jin Kim, Tianyi Chen, and Pashmina Cameron. Scaling reasoning efficiently via relaxed on-policy distillation. *arXiv preprint arXiv:2603.11137*, 2026.

[50] Xinsen Zhang, Zhenkai Ding, Tianjun Pan, Run Yang, Chun Kang, Xue Xiong, and Jingnan Gu. Opsdl: On-policy self-distillation for long-context language models. *arXiv preprint arXiv:2604.17535*, 2026.

[51] Zhaoyang Zhang, Shuli Jiang, Yantao Shen, Yuting Zhang, Dhananjay Ram, Shuo Yang, Zhuowen Tu, Wei Xia, and Stefano Soatto. Reinforcement-aware knowledge distillation for llm reasoning. *arXiv preprint arXiv:2602.22495*, 2026.

[52] Songming Zhang, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu. Dual-space knowledge distillation for large language models. In *EMNLP*, 2024.

[53] Xiao Cui, Mo Zhu, Yulei Qin, Liang Xie, Wengang Zhou, and Houqiang Li. Multi-level optimal transport for universal cross-tokenizer knowledge distillation on language models. In *AAAI*, 2025.

[54] Paul Youssef, Osman Koraş, Meijie Li, Jörg Schlötterer, and Christin Seifert. Give me the facts! a survey on factual knowledge probing in pre-trained language models. In *Findings of EMNLP*, 2023.

[55] Tianjie Ju, Zhenyu Shao, Bowen Wang, Yujia Chen, Zhuosheng Zhang, Hao Fei, Mong-Li Lee, Wynne Hsu, Sufeng Duan, and Gongshen Liu. Probing then editing response personality of large language models. *arXiv preprint arXiv:2504.10227*, 2025.

[56] Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *ACL*, 2018.

[57] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*, 2019.

[58] John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *EMNLP*, 2019.

[59] Amos Azaria and Tom Mitchell. The internal state of an llm knows when it's lying. In *Findings of EMNLP*, 2023.

[60] Tianjie Ju, Weiwei Sun, Wei Du, Xinwei Yuan, Zhaochun Ren, and Gongshen Liu. How large language models encode context knowledge? a layer-wise probing study. In *COLING*, 2024.

[61] Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. Uncovering safety risks of large language models through concept activation vector. In *NeurIPS*, 2024.

[62] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 2022.

[63] Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 2024.

[64] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *EMNLP*, 2021.

[65] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *ACL*, 2022.

[66] Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. In *ACL*, 2021.

[67] Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. Leace: Perfect linear concept erasure in closed form. In *NeurIPS*, 2023.

[68] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017.

[69] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023.

[70] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *ACL*, 2023.

[71] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

[72] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *EMNLP*, 2022.

[73] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. In *ACL*, 2023.

[74] Yunxiang Li, Zihan Li, Kai Zhang, Ruilong Dan, Steve Jiang, and You Zhang. Chatdoctor: A medical chat model fine-tuned on a large language model meta-ai (llama) using medical domain knowledge. *Cureus*, 15(6), 2023.

[75] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021.

[76] Alignment-Lab-AI. Lawyer-instruct. Hugging Face Datasets, 2024.

[77] Ilias Chalkidis, Nicolas Garneau, Catalina Goanta, Daniel Katz, and Anders Søgaard. LeXFiles and LegalLAMA: Facilitating English multinational legal language model development. In *ACL*, 2023.

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

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

[80] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[81] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[82] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[83] Siyue Wu, Hongzhan Chen, Xiaojun Quan, Qifan Wang, and Rui Wang. Ad-kd: Attribution-driven knowledge distillation for language model compression. In *ACL*, 2023.

[84] Tianyu Huang, Weisheng Dong, Fangfang Wu, Xin Li, and Guangming Shi. Uncertainty-driven knowledge distillation for language model compression. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2850–2858, 2023.

[85] Guodong Xu, Ziwei Liu, and Chen Change Loy. Computation-efficient knowledge distillation via uncertainty-aware mixup. *Pattern recognition*, 138:109338, 2023.

[86] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[87] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009.

[88] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[89] Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Less is more: task-aware layer-wise distillation for language model compression. In *ICML*, 2023.

Appendix Contents

A Methodological and Experimental Details **17**
- A.1 Details of Distillation Losses for LLMs . . . 17
- A.2 Details of Datasets . . . 17
- A.3 Details of Knowledge Probing Algorithm . . . 18
- A.4 Details of Baselines . . . 19
- A.5 Details of Experimental Parameters . . . 20
- A.6 Details of Probing Datasets . . . 20

B Additional Experiments and Analysis **21**
- B.1 Main Experiments on TinyLlama-1.1B . . . 21
- B.2 Ablation Experiments on Qwen2.5-1.5B . . . 22
- B.3 Experiments on Larger Teacher Models . . . 23
- B.4 Analysis of Number of Probing Tokens k_1 . . . 24
- B.5 Analysis of Number of Probing Layers k_2 . . . 24
- B.6 Analysis of Different Distillation Scenarios . . . 25
- B.7 Analysis of Computational Overhead . . . 27
- B.8 Analysis of Different Attribution and Uncertainty-Guided Methods . . . 27
- B.9 Analysis of Different Plug-In-Style KD Methods . . . 30
- B.10 Analysis of Result Significance . . . 31
- B.11 Analysis of Alternative Uncertainty Estimators and Attribution Methods . . . 32
- B.12 Analysis of Significance and Robustness of GPT-4 Pairwise Evaluation . . . 33
- B.13 Visualization of Knowledge Probing in Teacher Models . . . 34
- B.14 Case Study . . . 36
- B.15 Discussion on Target-Aware Uncertainty . . . 39
- B.16 Discussion on Offline Probing and Optimization Stability . . . 39
- B.17 Analysis on the Scalability of the Probing Procedure . . . 40
- B.18 Analysis on Heterogeneous Teacher-Student Distillation . . . 40
- B.19 Sensitivity to the Activation Threshold . . . 41
- B.20 Rationale of Isometric Layer Mapping . . . 41

C Theoretical Analysis and Discussion **41**
- C.1 Analysis of Student-Side Uncertainty Probing . . . 42
- C.2 Analysis of Teacher-Side IG Attribution and Layer Selection . . . 44
- C.3 Analysis of Alignment Mapping and Intermediate-Layer Distillation Objective . . . 46

D Limitations **50**

A Methodological and Experimental Details

A.1 Details of Distillation Losses for LLMs

We introduce the common distillation loss calculation methods for generative language models (LMs). We divide them into **token-level** and **sequence-level** methods.

Token-level Distillation. Given the dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$ and the vocabulary $\mathcal{V}$, for each input-output pair $(x, y) \in \mathcal{D}$, we denote $p(\cdot|y_{<t}, x)$ and $q(\cdot|y_{<t}, x)$ as the probability distribution for each generative token of the teacher model and the student model, respectively. Common token-level distillation losses are shown in Table 5, where $\mathrm{KL}(\cdot||\cdot)$ and $\mathrm{JSD}(\cdot||\cdot)$ between two probability distributions $p(\cdot)$ and $q(\cdot)$ can be defined as follows:

$$\mathrm{KL}(p(\cdot)||q(\cdot)) = \sum_{v\in\mathcal{V}} p(v) log \frac{p(v)}{q(v)} \tag{13}$$

$$\mathrm{JSD}(p(\cdot)\|q(\cdot)) = \frac{1}{2}\mathrm{KL}(p(\cdot)\|m(\cdot)) + \frac{1}{2}\mathrm{KL}(q(\cdot)\|m(\cdot)) \tag{14}$$

where $m(\cdot) = \frac{1}{2}\left(p(\cdot) + q(\cdot)\right)$.

Table 5: Token-level distillation losses for generative LMs.

Distillation Methods	Formula
Forward KL Divergence [29]	$\mathcal{L}_{\mathrm{FKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\|y\|} \mathrm{KL}(p(\cdot\|y_{<t}, x)\|q(\cdot\|y_{<t}, x))\right]$
Reverse KL Divergence [24]	$\mathcal{L}_{\mathrm{RKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\|y\|} \mathrm{KL}(q(\cdot\|y_{<t}, x)\|p(\cdot\|y_{<t}, x))\right]$
Jensen-Shannon Divergence [23]	$\mathcal{L}_{\mathrm{JS}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\|y\|} \mathrm{JSD}(p(\cdot\|y_{<t}, x)\|q(\cdot\|y_{<t}, x))\right]$
Skew (Reverse) KL Divergence [25]	$\mathcal{L}_{\mathrm{SKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\|y\|} \mathrm{KL}(p(\cdot\|y_{<t}, x)\|\alpha p(\cdot\|y_{<t}, x) + (1-\alpha) q(\cdot\|y_{<t}, x))\right]$

Sequence-Level Distillation. Given the dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$ and the vocabulary $\mathcal{V}$, for each input-output pair $(x, y) \in \mathcal{D}$, we denote $p(y_t|y_{<t}, x)$ and $q(y_t|y_{<t}, x)$ as the conditional probabilities assigned to the t-th token $y_t \in y$ by the teacher and student models, respectively. In SeqKD, the teacher model generates target sequences, which are then used to supervise the student model. The sequence-level distillation loss is defined as:

$$\mathcal{L}_{\mathrm{SeqKD}} = -\mathbb{E}_{(x,y)\sim\mathcal{D}}[\sum_{t=1}^{|y|} \log q(y_t|y_{<t}, x)] \tag{15}$$

A.2 Details of Datasets

Domain-Agnostic Datasets. For domain-agnostic datasets, we conduct experiments on five datasets: Databricks-Dolly-15k (**Dolly**) [69], Self-Instruct (**SelfInst**) [70], Vicuna Evaluation (**Vicuna**) [71], Super-Natural Instructions (**S-NI**) [72], and Unnatural Instructions (**UnNI**) [73]. Specifically, we use 11K samples from **Dolly** for training, 1K for validation, and 500 for testing. Additionally, to further assess the generalization ability, we evaluate on **SelfInst**, **Vicuna**, **S-NI**, and **UnNI**. The details are listed as follows:

- **Databricks-Dolly-15k** [69]: Databricks-Dolly-15k is an open-source instruction-following dataset that consists of 15K examples covering a diverse set of task categories such as brainstorming, classification, closed and open-ended QA, generation, information extraction, and summarization. The instructions were manually created by thousands of contributors, following behavioral task types originally outlined in [82].
- **Self-Instruct** [70]: Self-Instruct is a large-scale instruction-following dataset generated using a bootstrapping framework, where a base language model is prompted to synthesize diverse instructional data. The dataset comprises 52K unique instructions and 82K corresponding input-output pairs, including 252 expert-curated tasks targeting practical

Algorithm 1 Knowledge Probing

Input: dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{|\mathcal{D}|}$, student model $\mathcal{S}$, teacher model $\mathcal{T}$, thresholds k_1, k_2, θ
Output: probing set $\mathcal{D}_{\text{probe}}$, activated teacher layers $\mathcal{Q}^{\mathcal{T}}$
Initialize $\mathcal{D}_{\text{probe}} \leftarrow \emptyset$
/* Step 1: Uncertainty-based probing for student */
for $i = 1, \ldots, |\mathcal{D}|$ **do**
  Get (x^i, y^i) from $\mathcal{D}$
  for $t = 1, \ldots, |y^i|$ **do**
    Update $u_t^i \leftarrow 1 - q\big(y_t^i \mid y_{<t}^i, x^i\big)$
  end for
  Update $\mathcal{K}^i \leftarrow TopK_{k_1}\big(\{u_t^i\}_{t=1}^{|y^i|}\big)$
  Update $\mathcal{D}_{\text{probe}} \leftarrow \mathcal{D}_{\text{probe}} \cup \{(x^i, y^i, \mathcal{K}^i)\}$
end for
/* Step 2: Neuron-based probing for teacher */
Initialize $N_l \leftarrow 0,\ \forall\, l \in \{1, \ldots, L^{\mathcal{T}}\}$
for each $(x^i, y^i, \mathcal{K}^i) \in \mathcal{D}_{\text{probe}}$ **do**
  for each $t \in \mathcal{K}^i$ **do**
    for $l = 1, \ldots, L^{\mathcal{T}}$ **do**
      for each $w_{l,j} \in \mathrm{FFN}(L_l)$ **do**
        Compute $\mathrm{Attr}(w_{l,j})$ via integrated gradients
        if $\mathrm{Attr}(w_{l,j}) > \theta$ **then**
          Update $N_l \leftarrow N_l + 1$
        end if
      end for
    end for
  end for
end for
Update $\mathcal{Q}^{\mathcal{T}} \leftarrow TopK_{k_2}\big(\{N_l\}_{l=1}^{L^{\mathcal{T}}}\big)$
return $\mathcal{D}_{\text{probe}},\ \mathcal{Q}^{\mathcal{T}}$

applications, along with 50K examples collected from public resources for benchmarking purposes.

- **Vicuna** [71]: The whole Vicuna dataset contains 70K samples of user-shared conversations. In our experiments, following the settings of [24], we choose 80 challenging questions to evaluate the model performance.
- **Super-Natural Instructions** [72]: Super-Natural Instructions is a comprehensive benchmark designed to evaluate instruction-following capabilities across 1,616 diverse NLP tasks, each paired with expert-written instructions. The dataset spans 76 distinct task types, with its test set comprising 9K samples drawn from 119 representative tasks.
- **Unnatural Instructions** [73]: Unnatural Instructions is a large-scale instruction dataset consisting of 240K examples generated primarily by AI models with minimal human involvement. The core evaluation subset includes 60K samples.

Domain-Specific Datasets. For the **Medical** track, we distill on **GenMedGPT** [74] (115K), and evaluate on **MMLU** [75] medical subsets: clinical knowledge (**CK**; 265 questions), professional medicine (**PM**; 272 questions), and college medicine (**CM**; 173 questions). For the **Legal** track, we distill on **Lawyer-Instruct** [76] (9.2K instruction response pairs) and **USTerms** [77] (5.8K terminology instances across seven labels) and evaluate on **MMLU** law subsets: international law (**IL**; 140 questions) and professional law (**PL**; 1.5K questions).

A.3 Details of Knowledge Probing Algorithm

We design a two-stage knowledge probing algorithm to identify critical knowledge gaps between the student and teacher models. The details of our knowledge probing are shown in Algorithm 1. In the first stage (lines 2–7), we estimate token-level uncertainty from the student model and select the top-k_1

Table 6: Training and distillation hyperparameters for the teacher model, student model, and our KPD framework.

Settings	Teacher Model	Student Model	Distillation
Common Parameters			
Epoch	10	10	10
Learning Rate	5e-5	5e-5	5e-4
Batch Size	8	8	4
LR Scheduler	Cosine	Cosine	Cosine
Fine-Tuning Method	LoRA	LoRA	LoRA
LoRA Rank	256	256	256
LoRA Alpha	64	64	64
LoRA Dropout	0.1	0.1	0.1
Weight Decay	1e-2	1e-2	1e-2
Clip Grad	1	1	1
Max Input Size	512	512	512
Top-k	0	0	0
Top-p	1	1	1
Temperature γ	1	1	1
Parameters of our KPD			
The Number of Key Tokens k_1	–	–	5
The Number of Teacher Layers k_2	–	–	3
Probing Steps s	–	–	20
Activation Threshold θ	–	–	0.2

most uncertain tokens as key targets. In the second stage (lines 10–19), we use integrated gradients to compute attribution scores for feedforward neurons in the teacher model, measuring their contribution to the prediction of each key token. The layers with the highest number of activated neurons, that is, those with attribution scores above a threshold, are selected as the top-k_2 knowledge-relevant teacher layers for distillation.

A.4 Details of Baselines

We apply our method to the following large model distillation methods to demonstrate the effectiveness and generalizability of our approach in our main experiments: (a) Three common distillation methods: **KD** [29], **RKD** [24], and **SeqKD** [40]. (b) Six LLM-designed methods: **ImitKD** [42], **GKD** [23], **MiniLLM** [24], **Distillm** [25], **Distillm-2** [33], and **ABKD** [34]. The details are listed as follows:

- **KD** [29]: The most traditional knowledge distillation method for LLMs involves minimizing the Kullback-Leibler (KL) divergence between the output distributions of a large teacher model and a smaller student model. Specifically, the student is trained to match the teacher's softmax probabilities over the vocabulary at each generation step, enabling the transfer of output-level knowledge without accessing ground-truth labels.
- **RKD** [24]: Reverse KL distillation (RKD) trains the student to produce confident predictions aligned with the teacher by minimizing the divergence from the student outputs to the teacher distribution, encouraging sharper outputs when the teacher is uncertain.
- **SeqKD** [40]: Sequence-level knowledge distillation (SeqKD) trains the student model to imitate full output sequences generated by the teacher, rather than matching token-level distributions, enabling simpler decoding strategies like greedy search while maintaining strong performance.
- **ImitKD** [42]: ImitKD views knowledge distillation as an imitation learning problem, training the student to mimic the teacher sequence-generation behavior while mitigating exposure bias, leading to better generation quality and faster inference compared to standard SeqKD.
- **GKD** [23]: GKD addresses the distribution mismatch in sequence-level distillation by training the student on its own generated outputs, using teacher feedback as supervision.

```
{
        "id": 5,
        "instruction": "Why are the Georgia Bulldogs back to back National Champions in Football?",
        "prompt": "Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back-to-back National Champions in Football?\n\n### Response:\n",
        "input": "",
        "output": "Coach Kirby Smart brought to Georgia what he learned as a longtime assistant with Nick Saban at Alabama and made his own process for Championship success. Smart has recruited the top high school players in the country to play for the Bulldogs and he has created a culture of brotherhood and accountability. The Georgia Bulldogs are now the standard of excellence when it comes to College Football."
}
```


Figure 7: An instruction-response pair.

It also supports flexible loss designs and naturally integrates with reinforcement learning fine-tuning.

- **MiniLLM** [24]: MiniLLM distills open-source white-box LLMs into smaller models by optimizing a reverse KL divergence objective, which better suits generative tasks by avoiding overestimation of low-probability tokens. It yields more accurate, well-calibrated, and efficient student models across various sizes.
- **Distillm** [25]: Distillm is an efficient distillation framework for autoregressive language models, combining a novel skew KL divergence loss with an adaptive off-policy training strategy, achieving strong performance with significantly reduced computational cost.
- **Distillm-2** [33]: DistillM-2 is a contrastive knowledge-distillation framework that boosts large-language-model compression by explicitly raising the probability of teacher outputs while lowering that of student outputs through asymmetric SKL/SRKL losses and curated data sampling, leading to tighter teacher-student alignment.
- **ABKD** [34]: ABKD introduces an $\alpha - \beta$ divergence loss that smoothly interpolates between forward and reverse KL, striking a principled balance between hardness- and confidence-concentration in knowledge distillation.

A.5 Details of Experimental Parameters

We implement our code with PyTorch and apply our framework for all baselines with 8 NVIDIA A100 GPUs. **All experiments are the average of 5 runs under different random seeds.** Table 6 summarizes the training and distillation hyperparameters used for the teacher model, the student model, and our proposed KPD framework. We adopt consistent configurations across models as shown in Table 6. The distillation stage employs a higher learning rate and smaller batch size for stability and efficiency. Additionally, we introduce four probing-specific parameters: the number of key tokens k_1, the number of teacher layers k_2, the probing steps s, and the activation threshold θ to control token selection, layer filtering, and attribution steps in our KPD framework. Following prior work [65, 60], we set s to 20 and θ to 0.2 to ensure that accurately activated neurons can be identified. For k_1 and k_2, we set them to 5 and 3, respectively, and study their effect in Experiments 4.4.

A.6 Details of Probing Datasets

We provide a detailed explanation of the probing dataset. We illustrate the knowledge probing process with an example from the `databricks-dolly-15k` (**Dolly**) dataset. Given an instruction-response pair in Figure 7. We then construct a probing dataset that queries the student model with truncated prompts to isolate each uncertain token, as shown in Figure 8. The set of $\mathcal{K}$ contains the key tokens `[Championship, process, brother, Coach, longtime]`. Using the above probing data, we activate neurons in the teacher model via integrated gradients and extract attribution scores. We count the number of activated neurons per layer and select the top layers with the highest activations as knowledge-relevant layers for distillation. In our experiments, due to the excessive time and memory

```
{

  "id": 5,
  "probes": [
    {
      "prompt": "<|begin_of_text|>Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back to back National Champions in Football?\n\n### Response:\nCoach Kirby Smart brought to Georgia what he learned as a longtime assistant with Nick Saban at Alabama and made his own process for",
      "key_token": "Championship"
    },
    {
      "prompt": "<|begin_of_text|>Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back to back National Champions in Football?\n\n### Response:\nCoach Kirby Smart brought to Georgia what he learned as a longtime assistant with Nick Saban at Alabama and made his own",
      "key_token": "process"
    },
    {
      "prompt": "<|begin_of_text|>Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back to back National Champions in Football?\n\n### Response:\nCoach Kirby Smart brought to Georgia what he learned as a longtime assistant with Nick Saban at Alabama and made his own process for Championship success. Smart has recruited the top high school players in the country to play for the Bulldogs and he has created a culture of",
      "key_token": "brother"
    },
    {
      "prompt": "<|begin_of_text|>Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back to back National Champions in Football?\n\n### Response:\n",
      "key_token": "Coach"
    },
    {
      "prompt": "<|begin_of_text|>Below is an instruction that describes a task. Write a response that appropriately completes the request.\n\n### Instruction:\nWhy are the Georgia Bulldogs back to back National Champions in Football?\n\n### Response:\nCoach Kirby Smart brought to Georgia what he learned as a",
      "key_token": "longtime"
    }
  ]
}
```


Figure 8: An example of probing datasets.

cost of probing with the full `databricks-dolly-15k` dataset, we construct three probe datasets by randomly sampling 500 entries from the `databricks-dolly-15k` dataset for the three student models TinyLlama, Llama3.2-1b, and Qwen2.5-1.5B, respectively. Then, we use these three probe datasets to probe the three teacher models Llama2-7B, Llama3.1-8B, and Qwen2.5-7B, respectively, and select the layer with the most activated neurons for distillation.

B Additional Experiments and Analysis

B.1 Main Experiments on TinyLlama-1.1B

Tables 7 and 8 report the main results of TinyLlama-1.1B on domain-agnostic (Rouge-L) and domain-specific (accuracy) benchmarks, respectively.

From Table 7, we observe that KPD consistently improves TinyLlama-1.1B across all nine KD backbones on domain-agnostic instruction benchmarks, indicating that uncertainty-guided token selection is broadly compatible with heterogeneous distillation objectives rather than being tailored to a particular recipe. Averaged over all backbones, KPD yields a +0.94 Rouge-L gain on average, with the largest mean improvement appearing on the in-domain Dolly set (+1.66), while still providing non-trivial gains on out-of-domain evaluations such as UnNI and Vicuna (both around +0.71 on average). The magnitude of improvements also correlates with the strength of the underlying distillation method: weaker or less LLM-specialized baselines (e.g., RKD and ImitKD) benefit more (e.g., Avg. +1.55 and +1.66, respectively), whereas strong generation-centric distillers (e.g., GKD, Distillm, Distillm-2, ABKD) show smaller yet consistent gains (typically +0.43 to +0.66 on Avg.), suggesting that KPD mainly contributes complementary "where-to-distill" supervision by prioritizing hard and uncertain

Table 7: Main results of TinyLlama-1.1B with Rouge-L scores on domain-agnostic datasets.

Methods	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg.
Llama2-7B → TinyLlama-1.1B						
Teacher Model	29.96	19.91	33.46	32.46	17.45	26.65
SFT	20.58	15.00	28.12	24.68	13.84	20.44
KD	20.28	15.23	27.68	24.38	14.79	20.47
w/ KPD	**22.94**	**16.17**	**28.01**	**25.30**	**14.96**	**21.48**
RKD	20.99	16.92	30.48	31.84	16.30	23.31
w/ KPD	**25.48**	**17.18**	**31.01**	**32.30**	**18.33**	**24.86**
SeqKD	21.05	15.96	30.81	30.88	15.60	22.86
w/ KPD	**23.88**	**16.50**	**31.94**	**31.34**	**15.67**	**23.87**
ImitKD	21.17	13.83	24.64	23.53	14.85	19.60
w/ KPD	**23.95**	**14.61**	**26.07**	**25.89**	**15.76**	**21.26**
GKD	23.47	17.05	31.14	29.24	15.57	23.30
w/ KPD	**23.55**	**17.97**	**31.98**	**29.31**	**15.85**	**23.73**
MiniLLM	23.10	15.10	29.14	28.14	14.25	22.17
w/ KPD	**24.13**	**16.21**	**30.15**	**29.41**	**15.12**	**23.17**
Distillm	25.82	18.49	32.65	31.46	16.06	24.89
w/ KPD	**26.18**	**19.34**	**33.26**	**31.88**	**16.64**	**25.46**
Distillm-2	26.44	18.47	32.74	31.75	16.13	25.11
w/ KPD	**26.86**	**19.57**	**33.39**	**31.91**	**16.84**	**25.72**
ABKD	26.52	18.38	32.66	31.83	16.22	25.12
w/ KPD	**26.79**	**19.64**	**33.29**	**32.14**	**17.03**	**25.78**

tokens instead of duplicating what these methods already optimize. Importantly, KPD also reduces the gap to the teacher: for example, the best augmented variants (e.g., ABKD w/ KPD and Distillm-2 w/ KPD) approach the teacher's Avg. Rouge-L, demonstrating that the added probing signal helps the student absorb additional teacher knowledge even under a large capacity gap (Llama2-7B → TinyLlama-1.1B).

Table 8 further shows that the benefits extend to domain-specific medical and legal evaluations, where improvements are measured by accuracy rather than n-gram overlap. Across all backbones, KPD increases the overall accuracy of Avg by about +0.027 absolute on average, and the gains concentrate on the most knowledge-intensive law subsets: PL improves by about +0.038 and IL by about +0.052 on average, which is considerably higher than the average improvements in the medical subsets (CK, PM, and CM). This pattern is consistent with the motivation of KPD: uncertainty tends to highlight long-tail entities and terminology and brittle decision points, while attribution-guided teacher layer selection provides higher-quality targets for transferring specialized domain knowledge. Concretely, several methods exhibit substantial boosts on law tasks (e.g., ImitKD improves from 0.25→0.31 on PL and 0.20→0.28 on IL), and the best augmented setting (e.g., Distillm-2 w/ KPD) reaches 0.33 Avg., narrowing the teacher-student gap (teacher 0.35 Avg.) and indicating that KPD is particularly effective at transferring professional knowledge beyond general instruction-following skills. Overall, the results indicate that KPD yields the most pronounced improvements when evaluation hinges on precise domain decisions, rather than surface-form similarity. The consistently larger gains on PL and IL suggest that KPD is especially beneficial for transferring sparse, long-tail legal concepts that are hard to recover from generic instruction tuning alone. Meanwhile, the smaller but stable improvements on medical subsets imply that KPD generalizes across domains and does not rely on domain-specific heuristics. Taken together, these findings support KPD as a practical plug-in for enhancing professional knowledge of the student model while preserving their general capability.

B.2 Ablation Experiments on Qwen2.5-1.5B

The ablation results of Qwen2.5-1.5B are shown in Table 9. Table 9 validates the necessity of both components in KPD—student-side uncertainty probing and teacher-side attribution probing—under

Table 8: Main results of TinyLlama-1.1B with accuracy scores on domain-specific datasets.

Methods	CK	PM	CM	PL	IL	Avg.
Llama2-7B → TinyLlama-1.1B						
Teacher Model	0.35	0.34	0.38	0.35	0.32	0.35
SFT	0.26	0.24	0.26	0.22	0.18	0.23
KD	0.30	0.28	0.33	0.24	0.22	0.27
w/ KPD	**0.31**	**0.30**	**0.34**	**0.26**	**0.26**	**0.29**
RKD	0.29	0.27	0.32	0.23	0.21	0.26
w/ KPD	**0.31**	**0.30**	**0.34**	**0.25**	**0.26**	**0.29**
SeqKD	0.30	0.29	0.33	0.25	0.22	0.28
w/ KPD	**0.32**	**0.31**	**0.34**	**0.27**	**0.27**	**0.30**
ImitKD	0.30	0.28	0.32	0.25	0.20	0.27
w/ KPD	**0.32**	**0.30**	**0.33**	**0.31**	**0.28**	**0.31**
GKD	0.32	0.29	0.34	0.27	0.23	0.29
w/ KPD	**0.34**	**0.32**	**0.36**	**0.33**	**0.28**	**0.32**
MiniLLM	0.31	0.29	0.33	0.26	0.21	0.30
w/ KPD	**0.32**	**0.31**	**0.34**	**0.29**	**0.26**	**0.32**
Distillm	0.33	0.31	0.34	0.26	0.22	0.29
w/ KPD	**0.34**	**0.32**	**0.35**	**0.31**	**0.27**	**0.32**
Distillm-2	0.34	0.30	0.35	0.28	0.23	0.30
w/ KPD	**0.35**	**0.32**	**0.36**	**0.32**	**0.28**	**0.33**
ABKD	0.30	0.28	0.32	0.26	0.22	0.28
w/ KPD	**0.31**	**0.30**	**0.33**	**0.30**	**0.27**	**0.30**

the Qwen2.5-7B → Qwen2.5-1.5B setting. Removing either probe consistently degrades performance across all evaluated backbones and datasets, but the teacher-probe is systematically more critical: compared to the full KPD-augmented pipeline, w/o student-probe typically drops Avg. Rouge-L by 0.80-2.43 (e.g., KD 27.67→26.50, SeqKD 28.82→28.02, GKD 31.54→29.76, Distillm 31.61→29.18), whereas w/o teacher-probe causes substantially larger degradations of 2.23–4.02 (e.g., KD 27.67→24.88, SeqKD 28.82→26.59, GKD 31.54→28.24, Distillm 31.61→27.59). This indicates that correctly identifying knowledge-relevant teacher layers (rather than randomly sampling layers) is the dominant factor for effective transfer, especially on challenging instruction distributions where deeper reasoning and knowledge composition is needed; accordingly, the largest drops under w/o teacher-probe often appear on knowledge-intensive sets such as S-NI (e.g., Distillm 41.91→37.51) and also on broader generalization tests like Vicuna (e.g., KD 20.26→17.43). At the same time, the non-trivial losses under w/o student-probe confirm that uncertainty-based token prioritization is not merely an auxiliary heuristic: it provides a consistent "focus mechanism" that reduces wasted distillation budget on easy tokens and complements teacher probing by aligning the distilled signal with the student's actual knowledge gaps.

B.3 Experiments on Larger Teacher Models

This subsection analyzes the scalability of KPD when the teacher model size increases. Due to limited computational resources, our main experiments in the paper focus on teacher models in the 7-8B range, where we already cover three different model families: Llama2-7B, Llama3.1-8B, and Qwen2.5-7B. These teachers are distilled into TinyLlama-1.1B, Llama3.2-1B, and Qwen2.5-1.5B, respectively, across both domain-agnostic and domain-specific datasets, and consistently show that KPD can be integrated into diverse KD baselines to improve the student.

To further investigate whether KPD still improves performance with a larger teacher, we additionally conduct a new experiment using Llama2-13B [81] as the teacher and TinyLlama-1.1B [78] as the student. We follow the same distillation setup as in the paper and evaluate on five instruction-following benchmarks (Dolly, SelfInst, S-NI, UnNI, and Vicuna), comparing KD and KD+KPD and reporting both per-dataset scores and the overall average as shown in Table 10. From Table 10, we observe

Table 9: Ablation results of Qwen2.5-1.5B with Rouge-L scores (%).

Methods	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg.
Qwen2.5-7B → Qwen2.5-1.5B						
KD	**27.76**	**19.14**	**38.93**	**32.25**	**20.26**	**27.67**
w/o student-probe	26.31	18.14	37.14	31.48	19.42	26.50
w/o teacher-probe	25.08	16.58	35.27	30.05	17.43	24.88
SeqKD	**29.11**	**20.38**	**37.77**	**36.47**	**20.37**	**28.82**
w/o student-probe	27.15	20.14	36.26	36.46	20.12	28.02
w/o teacher-probe	25.80	18.86	34.35	35.22	18.74	26.59
GKD	**30.12**	**23.60**	**42.32**	**38.70**	**22.96**	**31.54**
w/o student-probe	28.14	21.41	40.61	37.16	21.47	29.76
w/o teacher-probe	26.68	19.62	38.62	36.03	20.24	28.24
Distillm	**29.94**	**24.27**	**41.91**	**39.62**	**22.28**	**31.61**
w/o student-probe	28.27	20.64	38.86	37.69	20.46	29.18
w/o teacher-probe	26.59	18.73	37.51	35.90	19.23	27.59

Table 10: Distillation from Llama2-13B to TinyLlama-1.1B on instruction-following benchmarks (Rouge-L).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
Llama2-13B to TinyLlama-1.1B						
KD	21.37	13.82	27.46	25.93	15.24	20.76
KD + KPD	**23.72**	**15.04**	**29.11**	**27.58**	**16.83**	**22.46**

that, even when the teacher is scaled up from 7B to 13B parameters, KPD consistently improves over vanilla KD on all benchmarks. The absolute gains range from +1.22 to +2.35 Rouge-L across datasets (e.g., Dolly: 21.37 → 23.72), with an overall average improvement of +1.70 Rouge-L. This is in line with the gains we see for Llama2-7B, Llama3.1-8B, and Qwen2.5-7B in the main paper, suggesting that our probing-based "what and where to distill" mechanism remains effective even when the teacher has a more complex representation space.

B.4 Analysis of Number of Probing Tokens k_1

The complete analysis of the number of probing tokens k_1 is shown in Figure 9 in KD, SeqKD, GKD, and Distillm. Figure 9 shows that KPD is moderately sensitive to the number of probed key tokens, but follows a clear and stable trend across KD, SeqKD, GKD, and Distillm: increasing k_1 from very small values yields consistent improvements because more uncertain tokens expose additional student knowledge gaps that can be corrected via targeted probing and intermediate distillation. However, the gains typically saturate around $k_1 = 4 \sim 5$, and in a few cases, we observe slight regressions on specific datasets (e.g., Distillm on Dolly/Vicuna), suggesting that probing an excessive number of tokens is not beneficial. Overall, these results indicate that KPD does not rely on a finely tuned k_1: a small set of high-uncertainty tokens is sufficient to obtain most of the benefit, and $k_1 = 4 \sim 5$ provides a practical trade-off between effectiveness and stability.

B.5 Analysis of Number of Probing Layers k_2

The complete analysis of the number of probing layers k_2 is shown in Figure 10 in KD, SeqKD, GKD, and Distillm. Figure 10 analyzes the effect of the number of selected teacher layers and demonstrates a similar "early gain, then diminishing returns" pattern. When k_2 increases from 1 to 2-3, performance improves noticeably, which is consistent with the intuition that aggregating signals from multiple knowledge-bearing layers enriches the transferred representation and makes the distillation targets more robust to layer-specific distillation. Beyond $k_2 = 3$, further increasing the number of layers offers marginal improvements or even slight degradation for some backbones (e.g., Distillm), implying that remembering too many layers may introduce redundant or weakly relevant

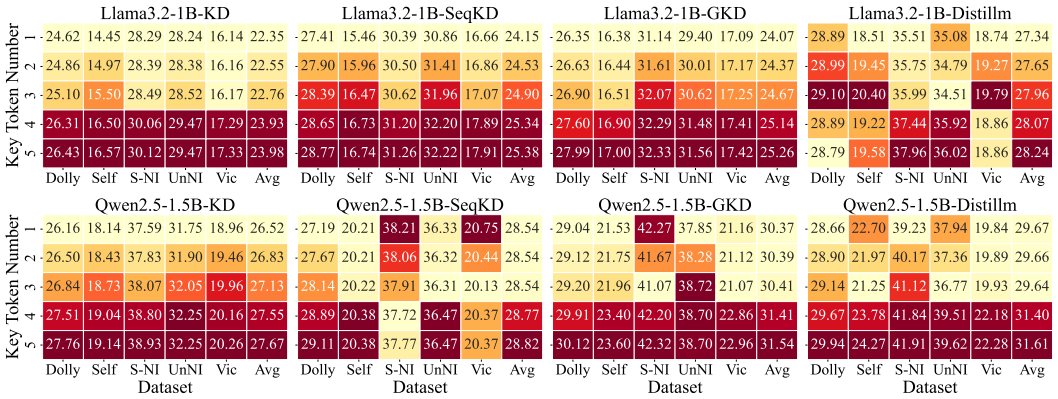


Figure 9: Complete results of probing token number k_1 from 1 to 5. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

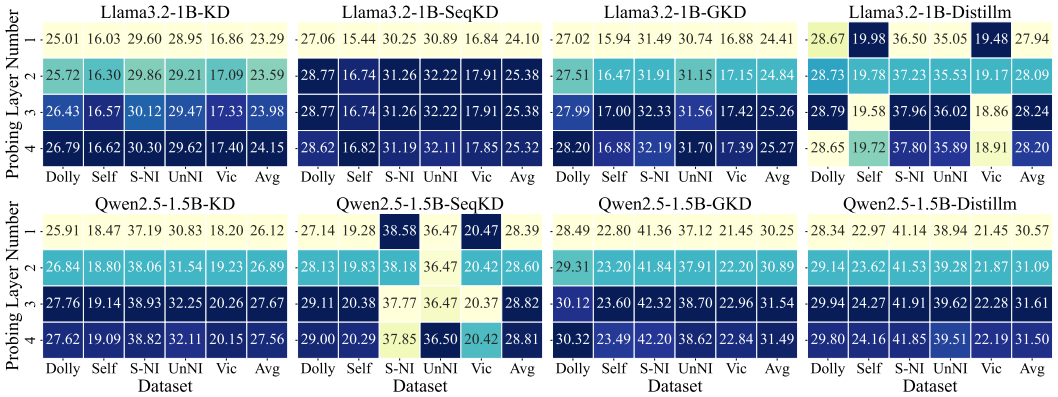


Figure 10: Results of probing layer number k_2 from 1 to 4. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

features that complicate optimization and reduce the sharpness of "where-to-distill" supervision. Therefore, $k_2 = 3$ is a robust default that captures most of the benefit while avoiding unnecessary complexity.

B.6 Analysis of Different Distillation Scenarios

Knowledge Distillation performance in LLMs can vary under different scenarios, e.g., teacher-student architectures, capacity gaps, and low-resource regimes. In this section, we summarize the behavior and generalizability of KPD across a wide range of KD scenarios covered in our experiments.

Architectures and Capacity Gaps (Three LLM Families, Three Teacher-Student Pairs). Our main experiments span three teacher-student pairs drawn from three LLM families, covering diverse architectures and compression ratios: (i) Llama2-7B $\rightarrow$ TinyLlama-1.1B ($\sim$6$\times$), (ii) Llama3.1-8B $\rightarrow$ Llama3.2-1B ($\sim$8$\times$), (iii) Qwen2.5-7B $\rightarrow$ Qwen2.5-1.5B ($\sim$4-5$\times$). These models differ in architecture, depth, width, and pretraining recipes (e.g., Llama vs. Qwen families), and they cover both very small students (1.1B) and moderately sized students (1.5B). Across all three pairs, KPD is applied as a plug-in to 9 diverse KD methods (KD, RKD, SeqKD, ImitKD, GKD, MiniLLM, Distillm, Distillm-2, and ABKD). As summarized in the main experiments of Table 1, KPD consistently improves every baseline across these settings, with gains of up to 4.56 Rouge-L on domain-agnostic instruction-tuning benchmarks. In particular, for the KD backbone, the average Rouge-L increases from 22.87 $\rightarrow$ 23.98 on the Llama3 pair and from 26.02 $\rightarrow$ 27.67 on the Qwen2.5 pair. For the stronger Distillm backbone, KPD still pushes the average Rouge-L from 27.81 $\rightarrow$ 28.24 (Llama3) and 31.02 $\rightarrow$ 31.61 (Qwen2.5). This indicates that KPD remains effective both when the teacher-student

Table 11: Low-resource distillation on Dolly (20% training data, Llama3.1-8B → Llama3.2-1B, Rouge-L).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
Llama3.1-8B → Llama3.2-1B						
SFT	22.13	13.57	27.05	25.46	15.08	20.66
KD	23.39	14.06	27.83	26.68	15.63	21.52
KD+KPD	**24.54**	**15.17**	**28.68**	**27.95**	**16.44**	**22.56**

Table 12: Low-resource distillation on Dolly (20% training data, Qwen2.5-7B → Qwen2.5-1.5B, Rouge-L).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
Qwen2.5-7B → Qwen2.5-1.5B						
SFT	23.33	14.37	28.44	26.68	16.24	21.81
KD	24.47	14.85	29.13	27.94	16.80	22.64
KD+KPD	**25.63**	**15.86**	**29.96**	**29.15**	**17.72**	**23.66**

gap is large and when the underlying KD recipe is already strong. Overall, the relative improvements of KPD are stable across different architectures and capacity gaps rather than being tailored to a single model pair.

Different Task and Data Regimes (Domain-Agnostic vs. Domain-Specific). Beyond general instruction-following benchmarks (Dolly, SelfInst, S-NI, UnNI, Vicuna), we also evaluate KPD in domain-specific medical and legal scenarios by distilling on GenMedGPT, Lawyer-Instruct, and USTerms and testing on MMLU medical and law subsets. These settings differ substantially in style, label space, and difficulty from general instruction-tuning. Concretely, as shown in Table 2 in the main experiments, for Llama3.1-8B → Llama3.2-1B, on medical (CK, PM, and CM) and legal (PL and IL) tasks, the average accuracy of the student improves from 0.26 (SFT) to 0.29 (KD), and further to 0.34 (KD+KPD). On individual tasks such as professional medicine (PM), accuracy increases from 0.24 (SFT) to 0.26 (KD) and 0.31 (KD+KPD). On clinical knowledge (CK), it improves from 0.33 (SFT) → 0.36 (KD) → 0.41 (KD+KPD). For Qwen2.5-7B → Qwen2.5-1.5B, the average accuracy on these medical and legal tasks similarly increases from 0.53 (SFT) to 0.58 (KD) and 0.61 (KD+KPD), with consistent gains (0.02-0.04 absolute) across all five sub-tasks. These results show that KPD not only helps on domain-agnostic instruction-following, but also provides robust improvements in specialized medical and legal domains across different model families.

Low-resource distillation regime. To directly assess the robustness of KPD under data scarcity, we conduct an additional experiment under a low-resource distillation regime. Specifically, for both the Llama3.1-8B → Llama3.2-1B and Qwen2.5-7B → Qwen2.5-1.5B pairs, we randomly sample 20% of the original Dolly training set to construct a reduced training corpus, and then train SFT, KD, and KD+KPD on this 20% subset while keeping all other training hyperparameters unchanged. We evaluate the resulting students on the same instruction-following benchmarks as in the main paper (Dolly, SelfInst, S-NI, UnNI, Vicuna). The Rouge-L results are summarized in Tables 11 and 12. We observe the following: (i) As expected, all methods degrade compared to the full-data setting, since they now see only 20% of the training examples. For example, under the low-resource regime, the average Rouge-L of SFT is 20.66 for Llama3.2-1B (Table 11) and 21.81 for Qwen2.5-1.5B (Table 12), both lower than their corresponding full-data counterparts reported in the main paper. (ii) Importantly, KPD consistently improves over the underlying KD baseline on every evaluation set, even in this low-resource regime. For Llama3.1-8B → Llama3.2-1B, the average Rouge-L increases from 21.52 (KD) to 22.56 (KD+KPD). For Qwen2.5-7B → Qwen2.5-1.5B, the average Rouge-L similarly increases from 22.64 (KD) to 23.66 (KD+KPD). (iii) Compared to SFT, KD+KPD provides even larger gains under 20% data. For Llama3.2-1B, the average Rouge-L improves from 20.66 (SFT) to 22.56 (KD+KPD). For Qwen2.5-1.5B, it improves from 21.81 (SFT) to 23.66 (KD+KPD). This shows that the combination of KD and KPD remains clearly beneficial even when the data budget is significantly reduced. These results indicate that the benefits of KPD are not limited to high-resource

Table 13: Conceptual comparison between KPD and representative attribution, uncertainty-guided KD methods. IG denotes Integrated Gradients. U-D KD and U-A Mixup denote Uncertainty-Driven KD and Uncertainty-Aware Mixup, respectively.

Method	Attribution	Uncertainty	Distillation Strategy	Scope
AD-KD	Teacher IG, token weights	–	Final logits + rationales	Encoder LMs (e.g., GLUE)
U-D KD	–	Teacher, module-level	Module compression	Encoder LMs (e.g., NLI)
U-A Mixup	–	Teacher, sample-level	Logit KD + mixup	Vision models
KPD (ours)	Teacher IG, layer selection	Student, token-level	Intermediate layers, key tokens, isometric	Generative LLMs

regimes: when the distillation data is scarce, KPD still offers stable, non-trivial gains over standard KD across different model families.

Overall, the above analyses show that KPD generalizes well across (i) different teacher-student architectures and capacity gaps (three LLM families, three compression ratios), (ii) distinct task domains (general instruction-following and specialized medical and legal scenarios), and (iii) low-resource distillation regimes (20% training data), consistently providing stable gains when plugged into a wide range of KD backbones.

B.7 Analysis of Computational Overhead

We have discussed the computational overhead of KPD in Figure 4b. In this section, we give a more detailed and clear description of the training inefficiency based on Figure 4b.

In practice, the additional cost is mostly one-time. Specifically, KPD adds three components on top of a standard white-box KD pipeline: (i) **Student-side uncertainty computation (Stage 1).** For each token, we compute the uncertainty as $u_t = 1 - q(y_t \mid y_{<t}, x)$. This uses the same student logits already needed for the distillation loss. The computation is a simple post-processing step and incurs negligible overhead (no extra forward or backward passes). (ii) **Teacher-side integrated gradients on a small probing set (Stage 2).** We construct a probing dataset $\mathcal{D}_{probe}$ consisting of key tokens with high student uncertainty and apply Integrated Gradients over FFN neurons to identify knowledge-bearing layers. As detailed in Appendix A.5 and A.6, this probing is run on a small set of examples (e.g., a limited number of Dolly instructions), with probing steps $s = 20$ and threshold $\theta = 0.2$. This is a one-time offline computation per teacher–student pair and does not occur during every optimization step of distillation. (iii) **Intermediate-layer loss on selected layers (Stage 3).** During distillation, we compute an additional hidden-layer KL loss $\mathcal{L}_{hidden}$ only for the top-k_2 teacher layers and their proportionally mapped student layers. This uses the same forward pass as the baseline white-box method. We simply read out intermediate hidden states and apply a small extra KL computation. Thus, the per-step overhead is modest and scales with k_2 (which we fix to 3 for all experiments).

Among the three stages above, the only component that contributes noticeable overhead is Stage 2, and thus we compare the extra probing time with the overall distillation time. Concretely, Figure 4b (right) shows that the total probing overhead accounts for only (i) **4.3%** of the distillation wall-clock time for **Llama2-7B** $\rightarrow$ **TinyLlama-1.1B**, (ii) **4.9%** of the distillation wall-clock time for **Llama3.1-8B** $\rightarrow$ **Llama3.2-1B**, and (iii) **5.3%** of the distillation wall-clock time for **Qwen2.5-7B** $\rightarrow$ **Qwen2.5-1.5B**. All methods (baselines and KPD-augmented variants) are trained with the same distillation setup, so these numbers directly reflect the extra computation introduced by KPD on top of an existing KD pipeline. In other words, the one-time probing stage adds only about **4-5%** additional wall-clock time relative to the total distillation cost, while yielding consistent performance gains across all three teacher-student pairs.

Overall, while KPD adds an extra probing stage, the overall training inefficiency is modest (around 5% extra wall-clock time), and the compute-performance trade-off is favorable given the consistent gains observed across teacher-student pairs and KD backbones.

B.8 Analysis of Different Attribution and Uncertainty-Guided Methods

This subsection clarifies the conceptual and empirical differences between our KPD and representative attribution- or uncertainty-guided KD methods, including **AD-KD** [83], **Uncertainty-Driven KD** [84], and **Uncertainty-Aware Mixup** [85].

Table 14: Comparison with attribution and uncertainty-guided methods in Rouge-L (Llama3.1-8B → Llama3.2-1B).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
(Llama3.1-8B → Llama3.2-1B						
AD-KD	23.65	14.12	27.45	26.89	14.97	21.42
Uncertainty-Driven KD	24.21	13.78	27.89	27.12	15.23	21.65
Uncertainty-Aware Mixup	23.92	14.35	28.03	26.64	14.68	21.52
KPD (KD)	**26.43**	**16.57**	**30.12**	**29.47**	**17.33**	**23.98**

B.8.1 Conceptual Analysis

Firstly, KPD differs from these approaches in the following aspects.

Different Attribution Strategies and Objectives. AD-KD computes teacher-side attributions using Integrated Gradients (IG) over the input embeddings and uses these token-level attribution scores to reweight the distillation loss and to make the student mimic the teacher's rationales at the output layer. Attribution here is only used to highlight which tokens are important for the teacher's final prediction. Uncertainty-Driven KD and Uncertainty-Aware Mixup do not explicitly use attribution. Compared to AD-KD, our KPD applies Integrated Gradients to teacher neurons and layers conditioned on the student's key tokens. Instead of attributing only to input tokens, KPD computes attribution scores for each neuron and layer and aggregates them to identify a small set of teacher layers that consistently contribute the most to correcting those uncertain tokens. These layers are then treated as knowledge-bearing layers and become the targets of intermediate hidden-state distillation. Compared to AD-KD's token-only attribution, KPD's layer-wise attribution turns interpretability into concrete layer selection, answering not only which tokens matter but also where in the LLM the relevant knowledge is stored. Compared to methods that ignore attribution altogether, KPD can localize the teacher's useful knowledge in depth, enabling more focused and interpretable distillation instead of uniformly using all or heuristic layers.

Different Uncertainty Estimation Strategies and Objectives. AD-KD does not explicitly model uncertainty. It relies solely on teacher attributions to reshape the token-level loss. Uncertainty-Driven KD introduces an uncertainty estimation module (UEM) attached to compressed blocks. This uncertainty is defined and used at the module level to decide how much information to retain in each compressed module when merging transformer layers. It is not used to pinpoint specific prediction errors or token-level knowledge gaps. Uncertainty-Aware Mixup estimates teacher predictive uncertainty at the sample level and uses it to select informative training examples and to schedule mixup in order to reduce teacher queries while preserving accuracy in image classification. Here, uncertainty drives sample selection and efficiency, not fine-grained supervision placement. Compared to Uncertainty-Driven KD and Uncertainty-Aware Mixup, our KPD uses the student's own token-level predictive distribution to estimate uncertainty. For each generated sequence, we compute uncertainty for each token (e.g., $u_t = 1 - q(y_t|y_{<t}, x)$) and select the top-$k$ most uncertain tokens as key tokens. These tokens explicitly mark where the student is currently least confident and thus where they lack knowledge in a generative setting. All subsequent teacher probing and distillation are conditioned on this student-side uncertainty. By basing uncertainty on the student and using it at the token level, KPD directly exposes the student's knowledge gaps instead of only the teacher's difficulty or module and single-sample hardness. This design turns uncertainty into a routing signal for distillation: it tells us exactly which tokens need extra supervision and later interacts with teacher-side attribution to determine where (which layers) to intervene, rather than simply reweighting losses or selecting hard samples.

Different Distillation Strategies (Last-Layer Distillation vs. Targeted Intermediate-Layer Distillation). AD-KD augments standard KD at the final layer and logits with an additional loss to match teacher rationales. Supervision is still primarily concentrated at the output head or fixed representation layers. Uncertainty-Driven KD uses uncertainty to guide module compression and feature reconstruction inside merged transformer blocks, but does not perform targeted, token- and layer-specific intermediate distillation driven by attribution and uncertainty together. Uncertainty-

Aware Mixup operates entirely at the logit level for vision models, using uncertainty-aware sample mixing without any explicit intermediate-layer distillation. KPD first uses student uncertainty to select key tokens and teacher attribution to select knowledge-bearing layers, then performs intermediate representation distillation at these probed teacher layers, with the probing-selected tokens serving as the layer-localization signal rather than restricting the loss to only those tokens. Furthermore, we design an isometric mapping strategy that aligns each probed teacher layer to a depth-proportional student layer, so that knowledge identified as early or late in the teacher is transferred to correspondingly early or late blocks in the student. Instead of more supervision everywhere at the last layer or uniformly across depth, KPD provides highly localized, interpretable supervision: it is restricted to (i) the tokens where the student is most uncertain and (ii) the teacher layers that attribution deems most relevant. The isometric mapping grounded in probed layers ensures that the transferred knowledge respects the hierarchical organization of the teacher, which is particularly important when distilling deep generative LLMs into shallower students.

Different Application Scopes (Discriminative Small Language Models vs. Generative Large Language Models (LLMs)) AD-KD is designed and evaluated on BERT-style discriminative small encoder-only language models for GLUE and related classification tasks, where inputs are short sequences, outputs are classification labels, and distillation focuses on encoder representations and classifier heads. Uncertainty-Driven KD similarly targets transformer encoders (e.g., BERT) and concentrates on compressing stacked encoder blocks while preserving performance on NLI and other sentence-level tasks. Uncertainty-Aware Mixup is proposed for vision CNNs [86] on CIFAR100 [87] and ImageNet [88], where uncertainty operates over images and batches, and distillation is framed in terms of image-level logits. Differently, KPD is specifically designed for generative large language models (LLMs) and instruction-following and knowledge-intensive generation. In this setting, (i) token-level uncertainty along long output sequences is central, and (ii) cross-layer attributions over transformer blocks capture where different types of knowledge are stored. Our two-sided probing (student uncertainty + teacher attribution) and isometric mapping are tailored to this autoregressive, multi-layer generative regime.

Overall, we summarize the differences between our method and existing attribution- and uncertainty-guided KD methods in Table 13. From Table 13, we observe that although KPD builds on standard tools such as uncertainty estimation and Integrated Gradients, it combines them in a distinct, student-centric distillation pipeline: (i) we use the student's token-level uncertainty to explicitly select key tokens that expose its knowledge gaps, (ii) we use teacher-side attribution to move beyond token weighting and select the small set of knowledge-bearing layers that matter most for those gaps, and (iii) we perform probed-layer–aware isometric hidden-state distillation only between these attribution-selected teacher layers and their depth-aligned student counterparts. This coordinated use of student-centric token selection, attribution-driven layer selection, and probing-informed isometric layer mapping has, to our knowledge, not been explored in prior attribution- or uncertainty-guided KD methods, and it constitutes the core novelty of our framework.

B.8.2 Empirical Analysis

To further substantiate the above conceptual differences, we also adapt AD-KD, Uncertainty-Driven KD, and Uncertainty-Aware Mixup to our generative LLM compression setting and evaluate them under the Llama3.1-8B $\rightarrow$ Llama3.2-1B configuration. Concretely, for AD-KD we follow its original design and add a rationale-matching loss on top of the distillation objective by computing teacher-side Integrated Gradients at the input tokens and encouraging the student to mimic these token-level attributions during autoregressive training. For Uncertainty-Driven KD, we introduce an uncertainty estimation module into the student's stacked blocks and use it, as in the original work, to guide which parts of the compressed blocks should be retained when merging transformer layers for the generative model. For Uncertainty-Aware Mixup, we estimate the teacher's predictive uncertainty at the sample level and perform uncertainty-aware mixup in the input and embedding space, applying KD on the mixed examples in the autoregressive training regime.

The results are summarized in Table 14. We observe that directly porting these traditional attribution and uncertainty-guided methods to generative LLM distillation leads to noticeably weaker results: their average Rouge-L scores lie in a narrow band between 21.42 and 21.65, whereas our KPD-based KD reaches 23.98, outperforming them by more than 2.3-2.5 on average. Moreover, KPD achieves higher Rouge-L on all five datasets (Dolly, SelfInst, S-NI, UnNI, and Vicuna) than any of the three

Table 15: Comparison with plug-in KD methods in Rouge-L (Llama3.1-8B → Llama3.2-1B).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
KD	25.72	15.19	29.06	28.19	16.19	22.87
KD + TED	25.87	15.45	29.31	28.41	16.61	23.13
KD + DSKD	26.12	15.64	29.46	28.50	16.34	23.21
KD + RLKD	25.94	15.62	29.14	28.64	16.02	23.07
KD + KPD	**26.43**	**16.57**	**30.12**	**29.47**	**17.33**	**23.98**
Distillm	28.33	19.23	37.60	35.65	18.24	27.81
Distillm + TED	28.47	19.24	37.72	35.73	18.46	27.92
Distillm + DSKD	28.58	19.38	37.78	35.87	18.61	28.04
Distillm + RLKD	28.51	19.31	37.68	35.69	18.53	27.94
Distillm + KPD	**28.79**	**19.58**	**37.96**	**36.02**	**18.86**	**28.24**

baselines. For instance, on SelfInst and Vicuna, KPD's scores are substantially higher than those of AD-KD, Uncertainty-Driven KD, and Uncertainty-Aware Mixup. These results indicate that directly adapting attribution- or uncertainty-based KD methods designed for encoder-only or vision models to the generative LLM setting is not sufficient, and that the student-centric token selection, attribution-driven layer selection, and probed-layer-aware intermediate distillation in KPD are crucial for obtaining strong performance in large-scale generative model compression.

B.9 Analysis of Different Plug-In-Style KD Methods

To further contextualize KPD among **plug-in-style methods** for generative LLM distillation, we additionally implement three recent and representative plug-in methods for LLM compression: (i) **TED** [89], which adaptively selects task-aware layers for distillation, (ii) **DSKD** [52], which distills knowledge in both token and feature spaces, (iii) **RLKD** [35], which encourages the consistency of the ranking of peak predictions between the teacher and student models. Under the same settings as Sec. A.5 (Llama3.1-8B → Llama3.2-1B), we consider two representative base distillation schemes: KD and Distillm, and then compare their plug-in variants: (i) **KD** vs. **KD+TED** vs. **KD+DSKD** vs. **KD+RLKD** vs. **KD+KPD**; (ii) **Distillm** vs. **Distillm+TED** vs. **Distillm+DSKD** vs. **Distillm+RLKD** vs. **Distillm+KPD**. For each configuration, we run 5 random seeds and report the average Rouge-L on five instruction-following benchmarks. The results are shown in Tables 15 and 16. From Tables 15 and 16, we make the following observations.

Firstly, plug-ins are generally beneficial, but KPD yields the largest gains. For both teacher–student pairs (Llama3.1-8B → Llama3.2-1B and Qwen2.5-7B → Qwen2.5-1.5B) and for both distillation backbones (KD and Distillm), equipping the baseline with a plug-in (TED, DSKD, RLKD, or KPD) consistently improves the *average* Rouge-L, indicating that adding a task-aware, representation-aware, or behavior-alignment component on top of a basic distillation framework is generally helpful in this generative LLM setting. When we look at the magnitude of the improvements, however, KPD stands out among all plug-ins. On the Llama KD backbone, the average Rouge-L increases from 22.87 (KD) to 23.13 (KD+TED), 23.21 (KD+DSKD), 23.07 (KD+RLKD), and **23.98 (KD+KPD)**, showing that KPD delivers a much larger gain than TED, DSKD, and RLKD under identical protocols. On the Llama Distillm backbone, the average rises from 27.81 to 27.92 (Distillm+TED), 28.04 (Distillm+DSKD), 27.94 (Distillm+RLKD), and **28.24 (Distillm+KPD)**, again with KPD providing the largest improvement. A similar pattern holds for the Qwen family: on the Qwen KD baseline, the average Rouge-L increases from 26.02 to 26.68 (KD+TED), 26.84 (KD+DSKD), 26.76 (KD+RLKD), and **27.67 (KD+KPD)**, and on Qwen Distillm it increases from 31.02 to 31.25 (Distillm+TED), 31.24 (Distillm+DSKD), 31.22 (Distillm+RLKD), and **31.61 (Distillm+KPD)**. Taken together, these comparisons across two model families and two backbones indicate that KPD is not merely an add-on, but a plug-in that consistently yields larger incremental gains than other representative plug-in methods of comparable scope.

Secondly, KPD's advantage is consistent across datasets and model families, not driven by a single benchmark. Another important observation is that KPD's improvements are not concentrated on one or two datasets, but rather spread across all five instruction-following benchmarks and both Llama and Qwen families. For the Llama KD baseline, KD+KPD achieves the best Rouge-L among all plug-in variants (KD+TED, KD+DSKD, KD+RLKD, KD+KPD) on every single dataset: for

Table 16: Comparison with plug-in KD methods in Rouge-L (Qwen2.5-7B → Qwen2.5-1.5B).

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg
KD	23.20	18.39	37.19	31.75	19.59	26.02
KD + TED	24.68	18.92	37.84	32.04	19.90	26.68
KD + DSKD	26.12	18.66	38.37	31.92	19.11	26.84
KD + RLKD	25.97	18.52	38.14	31.84	19.32	26.76
KD + KPD	**27.76**	**19.14**	**38.93**	**32.25**	**20.26**	**27.67**
Distillm	29.38	23.84	41.30	38.96	21.61	31.02
Distillm + TED	29.74	23.85	41.47	39.11	22.10	31.25
Distillm + DSKD	29.76	24.22	41.35	39.24	21.63	31.24
Distillm + RLKD	29.64	24.06	41.39	39.16	21.84	31.22
Distillm + KPD	**29.94**	**24.27**	**41.91**	**39.62**	**22.28**	**31.61**

Table 17: Ablations on uncertainty estimators and attribution methods (Rouge-L on Dolly).

	Setting 1	Setting 2	Setting 3	Setting 4	Setting 5	Setting 6	Setting 7
Uncertainty Attribution	Prob (default) IG (default)	Entropy IG	Margin IG	Random IG	Prob grad×act	Prob attention	Prob random
Llama3.1-8B → Llama3.2-1B	**26.43**	24.35	24.06	24.17	23.56	23.62	23.96
Qwen2.5-7B → Qwen2.5-1.5B	**27.76**	25.63	25.45	25.27	24.86	25.92	25.17

instance, it reaches 26.43 on Dolly and 16.57 on SelfInst, outperforming TED, DSKD, RLKD, and shows similar dominance on S-NI, UnNI, and Vicuna. The same pattern appears on the Llama Distillm backbone, where Distillm+KPD yields the highest Rouge-L on each dataset. Crucially, this behavior replicates on the Qwen teacher–student pair: both KD+KPD and Distillm+KPD consistently attain the top scores across Dolly, SelfInst, S-NI, UnNI, and Vicuna, outperforming TED, DSKD, RLKD variants in all cases. Since these datasets differ significantly in style, difficulty, and source, and the two model families differ in architecture and pretraining, the fact that **KPD dominates across all datasets and both families** suggests that it captures a broadly useful mechanism for directing distillation to the student's weak spots, rather than being tailored to a particular benchmark or architecture.

Thirdly, KPD scales well from weaker to stronger baselines across different model families. Finally, KD and Distillm represent relatively weak and strong distillation recipes, respectively, and the Llama and Qwen pairs further diversify the setting. The results in Tables 15 and 16 show that KPD brings clear gains in all these regimes. On the weaker KD backbones, KPD produces substantial increases in the average Rouge-L for both Llama (from 22.87 to 23.98) and Qwen (from 26.02 to 27.67), reflecting a sizeable improvement in the student's ability to follow instructions across diverse tasks even when starting from a simple KD procedure. On the stronger Distillm backbones, where there is limited room for improvement and the baselines are already competitive, KPD still increases the average Rouge-L from 27.81 to 28.24 on Llama and from 31.02 to 31.61 on Qwen, indicating that it can extract additional performance on top of a carefully designed distillation algorithm. This behavior supports our claim that KPD functions as a generic plug-in: it remains effective when stacked on both simple and strong distillation recipes and across different LLM families, rather than being tightly coupled to a particular backbone or architecture.

B.10 Analysis of Result Significance

All reported numbers in the paper, including Rouge-L scores, accuracies, and GPT-4 preference statistics, are averaged over **5 independent runs** with different random seeds, and are therefore not based on single-seed point estimates. Then, we examine the variability across these 5 runs, and find that it is consistently small relative to the reported gains. For Rouge-L, the deviations around the mean are within a narrow band (**±0.05** across all datasets and methods), and for accuracy metrics, the variation is within **±0.01**. For GPT-4 pairwise evaluations, when we recompute the win, tie, lose counts across seeds on the 100-example subsets, the fluctuations in the number of wins are within **3**

Table 18: GPT-4 pairwise evaluation with confidence intervals (Dolly, KD+KPD vs. KD).

Method A vs. B	#Win	#Tie	#Lose	Win rate (excluding ties, 95% CI)
KD+KPD vs. KD	57	13	30	65.5% [55.5%, 75.5%]

votes. In other words, the improvements reported in the experimental tables are substantially larger than the underlying run-to-run noise and are robust to the choice of random seed.

B.11 Analysis of Alternative Uncertainty Estimators and Attribution Methods

This subsection examines whether KPD is sensitive to the specific choices of (i) the student-side uncertainty estimator used to select key tokens and (ii) the teacher-side attribution method used to identify knowledge-bearing layers. Concretely, the uncertainty estimator is applied token-wise, and then the top-k tokens are selected as key tokens; the attribution method produces a layer-level importance score, and the top layers are selected as knowledge-bearing layers.

Alternative Uncertainty Estimators for key-Token Selection. Four token-level uncertainty estimators are considered: (i) **Prob (default)**: $u_t = 1 - p(y_t)$, for example, one minus the predicted probability of the ground-truth token at position t, and the top-k tokens with the largest u_t are selected; (ii) **Entropy**: $u_t = -\sum_v p(v) \log p(v)$, and the top-$k$ highest-entropy tokens are selected; (iii) **Margin**: $u_t = p_{(1)} - p_{(2)}$, the margin between the top-1 and top-2 probabilities ($p_{(1)}$ and $p_{(2)}$), and the top-k tokens with the *smallest* margins are selected; (iv) **Random**: the predictive distribution is ignored and k token positions are sampled uniformly at random.

Alternative attribution methods for layer selection. Four layer attribution variants are considered: (i) **IG (default)**: Integrated Gradients (IG) with respect to hidden activations, followed by aggregating neuron-level attributions within each layer to obtain a layer importance score; (ii) **grad$\times$act**: elementwise product of gradients and activations, followed by per-layer aggregation; (iii) **attention**: attention-based importance, computed by averaging attention weights to approximate how much a layer contributes to the key tokens; (iv) **Random**: a fixed number of teacher layers are sampled uniformly at random and treated as knowledge-bearing layers.

Configurations and results. Seven configurations are evaluated in Table 17: Settings 1–4 fix the attribution method to IG and vary uncertainty estimators (Prob, Entropy, Margin, Random); Settings 5–7 fix the uncertainty estimator to Prob and vary attribution methods (grad$\times$act, attention, random). The reported numbers are Rouge-L on Dolly under two teacher-student pairs.

Several consistent observations follow from Table 17. **Firstly, principled uncertainty estimators outperform random token selection.** When we fix the attribution method to IG (Settings 1-4), using any uncertainty-based scheme (prob, entropy, or margin) is clearly better than selecting key tokens at random. For the Llama teacher–student pair, all three principled estimators (Settings 1-3) yield higher Rouge-L than the random-token baseline (Setting 4), with our probabilistic measure (Setting 1) performing best. The same pattern holds for the Qwen pair: probabilistic, entropy-based, and margin-based uncertainty (Settings 1-3) all outperform random token selection (Setting 4). This confirms that where we place extra supervision (which tokens we choose) matters, and that leveraging meaningful uncertainty scores is more effective than choosing tokens arbitrarily.

Secondly, IG is the strongest attribution method, and other attribution variants are weaker. When we fix the uncertainty estimator to our probabilistic measure and vary the attribution method (Settings 1, 5-7), we observe that IG (Setting 1) consistently achieves the highest Rouge-L for both model families. Alternatives such as grad×act (Setting 5) and attention-based importance (Setting 6) also work better than random layer selection (Setting 7), but they are still clearly inferior to IG. This suggests that while various attribution methods can provide useful signals, our IG-based layer selection is the most effective way to identify knowledge-bearing layers in the teacher, and random layer choices are suboptimal even when token-level uncertainty is well estimated.

Thirdly, our original combination (probability-based uncertainty + IG) is the best overall. Across all seven configurations, the original design of KPD-using probability-based student uncertainty to select key tokens and IG to select knowledge-bearing layers (Setting 1)-achieves the highest

Please act as an impartial judge and compare the quality of response A and response B provided by two AI assistants to the user question displayed below. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of the response.

Just tell me which response do you think is better:
If A is significantly better than B, just answer me "A";
If B is significantly better than A, just answer me "B";
If A and B have similar quality (both good or both wrong), just answer me "Tied".

[Question]
{question or instruction}

[Response A]
{response A}

[Response B]
{response B}

Figure 11: Judging prompt for GPT-4.

Table 19: GPT-4 preferences for KD+KPD vs. KD under two judging prompts (Dolly).

Prompt	#Win	#Tie	#Lose	Win rate (excluding ties)
Original prompt	57	13	30	65.5%
Paraphrased prompt	58	12	30	65.9%

Rouge-L on Dolly for both the Llama and Qwen teacher-student pairs. Other combinations (e.g., entropy+IG or prob+attention) are consistently weaker. These findings support our design choices while also indicating that KPD is not overly sensitive to the exact probing variant: as long as probing is guided by meaningful uncertainty and attribution signals, we obtain clear gains over random baselines.

Overall, this additional study shows that the improvements brought by KPD stem from a principled combination of student-side uncertainty and teacher-side attribution, rather than from an arbitrary choice of probing heuristics.

B.12 Analysis of Significance and Robustness of GPT-4 Pairwise Evaluation

Beyond reporting GPT-4 pairwise preferences over 100 test examples per dataset, we further strengthen the reliability of this evaluation along three aspects: (i) an explicit significance analysis with confidence intervals, (ii) a prompt-robustness study with two different judging prompts, and (iii) a clarification of how we quantify the uncertainty of GPT-4 preferences.

Significance Analysis. Using the win, tie, lose counts for KD+KPD vs. KD on Dolly (100 test examples), we compute the win rate and its 95% confidence interval via a standard binomial proportion test (ties are excluded when computing the proportion). The results are summarized in Table 18. Here, KD+KPD wins 57 times and loses 30 times (with 13 ties), yielding a win rate of 65.5% over KD on the 87 non-tied cases. The corresponding 95% confidence interval, [55.5%, 75.5%], lies entirely above 50%, indicating that GPT-4's preference for the KPD-augmented student over the baseline KD student is statistically significant, rather than a random fluctuation on this 100-example subset.

Prompt Robustness. To mitigate potential bias introduced by a particular prompt formulation, we have already adopted an A/B-style GPT-4 evaluation protocol in all experiments reported in the paper, following the setting of DSKD [52]. For completeness, we provide additional details in this section. Specifically, for each test instance, we take the outputs from the baseline (e.g., KD) and the KPD-enhanced model (e.g., KD+KPD), and randomly assign them to "Response A" and "Response B". GPT-4 never sees which model produced which response. It only sees two anonymous candidates and is asked to decide which one is better. The main judging prompt we use is shown in Figure 11.

You are an objective evaluator. Two AI assistants, A and B, have answered the same user question shown below. Your job is to judge which answer is better overall, taking into account correctness, completeness, usefulness, clarity, and level of detail.

Please respond with only one token:
- If you think A is clearly better than B, answer "A".
- If you think B is clearly better than A, answer "B".
- If the two answers are of similar quality (both good or both poor), answer "Tied".

[Question]
{question or instruction}

[Response A]
{response A}

[Response B]
{response B}

Figure 12: Modified but the same judging prompt for GPT-4.

By randomizing which model's output is labeled as A/B and forcing GPT-4 to respond with only "A", "B", or "Tied", we largely avoid systematic bias arising from model identity or position in the prompt.

To further verify that our conclusions are not sensitive to the exact wording of the judging prompt, we additionally evaluate GPT-4 preferences on Dolly using a paraphrased prompt that expresses the same evaluation criteria but with different phrasing, as shown in Figure 12. We again randomize which model's output is treated as A/B for each example and compare KD+KPD vs. KD under both prompts. The results are summarized in Table 19. Under the original judging prompt, KD+KPD wins 57 times, loses 30 times, and ties 13 times, yielding a win rate of 65.5% over KD on the 87 non-tied cases. With the paraphrased prompt, KD+KPD wins 58 times, loses 30 times, and ties 12 times, resulting in a very similar win rate of 65.9%. The two prompts therefore lead to nearly identical preferences, and in both cases GPT-4 clearly favors KD+KPD over KD. This indicates that our GPT-4 conclusions are robust to reasonable prompt variations and are not an artifact of a particular wording choice.

Uncertainty Analysis. Regarding the uncertainty of GPT-4 judgments, there are two sources of variability: (i) sampling noise from using a finite test set, and (ii) randomness in the evaluation procedure (e.g., different random seeds for selecting test examples or shuffling A/B). We address these as follows. For **sampling noise**, we quantify uncertainty using the confidence intervals in Table 18, which tell us how much the win rate would vary if we repeated the evaluation on different samples under the same prompt. For **procedure randomness**, we repeat the GPT-4 evaluation five times with different random seeds. Across these runs, the fluctuation in the number of wins for KD+KPD vs. KD is small (within about 2 votes on a 100-example set), which is much smaller than the gap between wins and losses. This indicates that our GPT-4 preference results are stable across runs and not driven by a single sample. Moreover, we do not ask GPT-4 to report its own "confidence score" in the prompt, as such self-reported confidence is known to be unreliable and harder to calibrate. Instead, we rely on the statistically grounded measures above (win-loss gaps, confidence intervals, and multi-run variability) to characterize the uncertainty of GPT-4 preferences.

B.13 Visualization of Knowledge Probing in Teacher Models

We visualize the distribution of activated neurons across transformer layers for three teacher models: Llama3.1-8B, Qwen2.5-7B, and Llama2-7B. As shown in Figure 13, 14, and 15, all models exhibit a clear concentration of activated neurons in both the early and final layers. This suggests that the critical knowledge required by the student model is primarily encoded in the initial and terminal layers of the teacher. Such a pattern motivates us to focus more on these layers during distillation.

Interestingly, we observe a significant contrast between model families. For the Llama series (Llama2-7B and Llama3.1-8B), most activated neurons are located in the final layers, indicating that these models tend to store essential knowledge in deeper representations. In contrast, Qwen2.5-7B exhibits

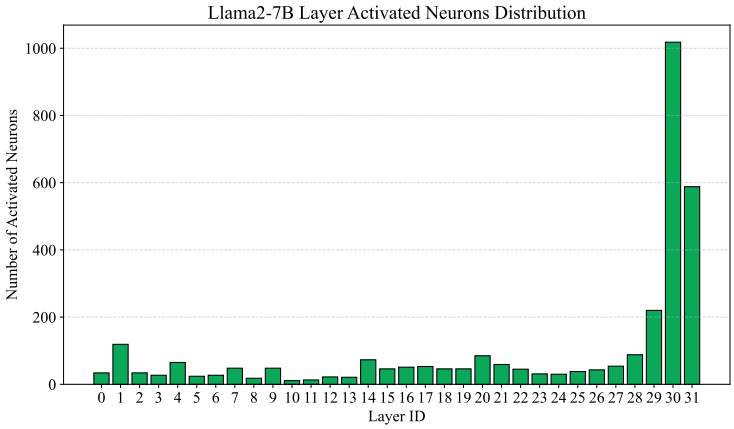


Figure 13: Llama2-7B layer activated neurons distribution.

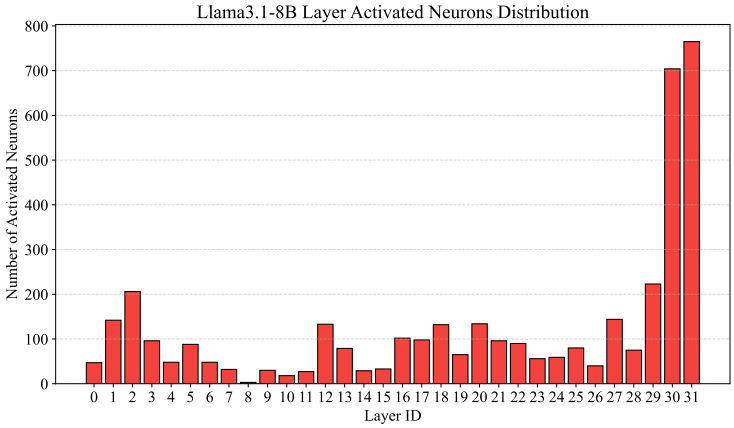


Figure 14: Llama3.1-8B layer activated neurons distribution.

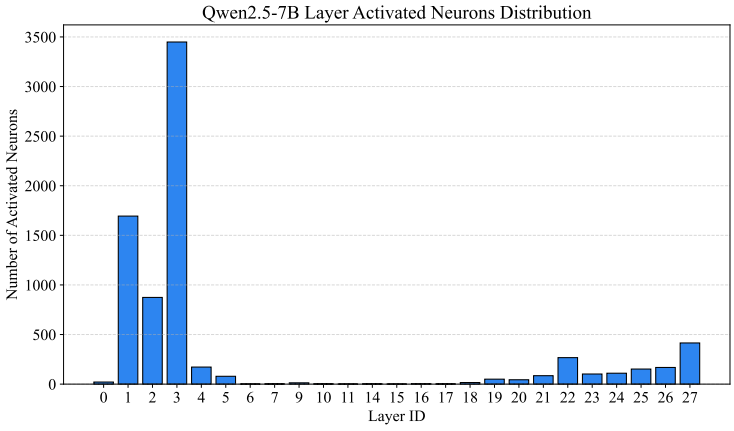


Figure 15: Qwen2.5-7B layer activated neurons distribution.

the opposite pattern, with a dominant concentration of activated neurons in the early layers, implying that this model family encodes more task-relevant knowledge at the bottom of the network.

These findings validate the effectiveness of our proposed **isometric mapping strategy**, which adaptively aligns student layers to teacher layers based on relative depth. Unlike heuristic selections, our **isometric mapping strategy** accommodates architectural differences, allowing student models

to absorb knowledge from both early and late teacher layers depending on where key information resides. This flexibility contributes to its superior and consistent performance across model families.

B.14 Case Study

We sample two sets of predictions, which are shown in Tables 20 and 21. Specifically, we compare Llama3.2-1B responses with and without KPD on two reasoning examples: the recipe instruction case and the guard change reasoning case in the UnNI dataset [73]. We observe that models distilled with KPD consistently provide more faithful and contextually grounded outputs compared to their non-KPD counterparts. In the guard change case (Table 20), KPD-enhanced models (such as KD, ImitKD, and MiniLLM) demonstrate a better grasp of temporal and spatial reasoning. For instance, the KD model with KPD correctly infers the contradiction between the real-world time of the event and the textual description, offering a nuanced explanation. Without KPD, the same model incorrectly relies on external factual knowledge (e.g., claiming the event occurs at 9 and 10 o'clock) without reconciling it with the passage. Similarly, in the recipe instruction case (Table 21), responses from KPD-distilled models (e.g., SeqKD, RKD, Distillm) are more aligned with the ground truth, correctly identifying all necessary steps and filtering out irrelevant ones. In contrast, several non-KPD models either introduce hallucinated steps or omit critical instructions, as seen in KD and GKD.

These results suggest that KPD plays a crucial role in guiding student models to better utilize and interpret contextual knowledge embedded in teacher signals. By probing and distilling intermediate knowledge representations, KPD enhances the capacity of the student model to reason faithfully and align outputs with the evidence presented in the input. In conclusion, across both cases, KPD yields more accurate and interpretable behavior, highlighting its effectiveness as a distillation strategy for instruction-aligned reasoning tasks.

Table 20: Case 1 - Llama3.2-1B responses with and without KPD.

Instruction: There are 3 pieces of information given in the text: two places and one time period. Based on this information, you need to answer which statement is true.		
Input: Text: Every evening from dusk till dawn, hundreds of people line up outside the gates of Buckingham Palace hoping to catch a glimpse of the changing of the guards.		
True Answer: The statement Ëvery evening from dusk till dawn, hundreds of people line up outside the gates of Buckingham Palaceïs true.		
Model	**w/ KPD**	**w/o KPD**
KD	The changing of the guards at Buckingham Palace occurs during the day, so this statement is false. However, the text mentions that "hundreds of people line up," which suggests the changing of the guards occurs at night, so this statement is true.	The statement that is true is: The changing of the guards happens twice a day, at 9 o'clock and 10 o'clock.
SeqKD	Statement 1: The changing of the guards happens every evening from dusk till dawn. Statement 2: There are 3 pieces of info in the text: one place and one time period. Statement 3: Since changing of the guards happens once every day, there are no fixed days.	The text gives two places and one time about tourists. Though time is not precise, the activity may take place at the palace from dusk till dawn.
RKD	It is true that hundreds of people line up every evening from dusk till dawn to catch a glimpse of the changing of the guards.	The time period is evening and the place is Buckingham Palace.
GKD	The "two places and one time period" represents a gap of 1 hour between 5:45 pm and 6:45 pm.	Both locations: Buckingham Palace and The Wall Change Period are not relevant to the provided information.
ImitKD	The correct statement is true. During the changing of the guards, hundreds of people line up at the palace gates, hoping to catch a glimpse. The gates are open between 5 pm and midnight.	None of the choices are true.
MiniLLM	Since it is stated that the activity happens "every evening from dusk till dawn," the answer must be true.	Out of the given information, statements A, B, and C can be true. Two places and time periods occurred throughout the week.
Distillm	The first time period is incorrect. The correct time period is "every evening from dusk till dawn."	The number of people lining up at the same time and place does not change.

Table 21: Case 2 - Llama3.2-1B responses with and without KPD.

Instruction: For this task, you will be given a list of instructions describing how to make something. Your job is to read the instructions and answer whether or not each instruction is necessary in order for the final product to turn out correctly. Instructions that are not needed can be omitted without affecting the outcome of the recipe.		
Input: Instructions: ['Preheat oven at 375 degrees', 'Bake for 20 minutes', 'Add salt'].		
True Answer: The following instructions are necessary in order for the recipe to turn out correctly: - Preheat oven at 375 degrees - Bake for 20 minutes - Add salt		
Method	**w/ KPD**	**w/o KPD**
KD	The necessary ingredients for this recipe are: 1. Baking powder 2. Olive oil or similar 3. Salt. With those ingredients, my final product turned out properly. Not-necessary ingredients: 1. Milk. While milk is something I would typically include if browning was an issue, it is not necessary here.	Preheat oven at 175 degrees; bake at 350 degrees; add salt.
SeqKD	Yes, oven preheating and baking for 20 minutes are necessary for this recipe to turn out correctly. The salt should also be added to the dish.	The oven needs to be preheated, the cake should bake for 20 minutes, and the salt should be added to enhance the flavor.
RKD	The instructions are necessary, as the product can only turn out correctly if the oven is preheated and baked for 20 minutes, and the salt is added.	Every one of these instructions is necessary in order for the product to turn out correctly.
GKD	1. Preheat oven at 375. 2. Bake for 20 minutes. 3. Add salt.	None of these instructions are necessary. The recipe can proceed even if some or all of the instructions are omitted.
ImitKD	Preheat oven at 375 degrees is unnecessary; it will not affect whether the recipe is successful or not. Bake for 20 minutes is also unnecessary. Add salt ensures baking time is right.	Option 1: True. Option 2: False. Option 3: True.
MiniLLM	20 minutes was left out from the instructions, so you could omit that from your recipe without affecting the outcome. Preheat oven at 350 degrees is necessary.	The first instruction (preheat oven. . .) is essential. You can add salt after the recipe is complete and it will not affect the outcome.
Distillm	The following instructions are necessary for the recipe to turn out correctly: Preheat oven at 375 degrees, bake for 20 minutes, add salt.	Yes, all three instructions are necessary in the recipe. Instructions that are not necessary can be omitted without affecting correct execution.

Table 22: Comparison between offline probing and online probing on Llama3.1-8B → Llama3.2-1B. The metric is average Rouge-L on domain-agnostic benchmarks.

Method	Offline Probing (ours)	Online Probing
KD	**23.98**	20.41
SeqKD	**25.38**	20.47
GKD	**25.26**	20.41
Distillm	**28.24**	23.14

B.15 Discussion on Target-Aware Uncertainty

We further clarify the role of the student-side uncertainty score used in KPD. In the main method, we define the token-level score as

$$u_t = 1 - q(y_t \mid y_{<t}, x), \tag{16}$$

where $q(y_t \mid y_{<t}, x)$ is the probability assigned by the student to the ground-truth token under teacher forcing. This score should not be interpreted as a generic distribution-level uncertainty measure in the same sense as predictive entropy. Instead, it is a target-aware difficulty score that measures how much the student underestimates the supervised target at a specific generation position.

This distinction is important because high entropy can arise from different causes. For example, some tokens may be inherently ambiguous because many continuations are plausible, while others may be difficult because the student fails to assign sufficient probability to the reference token. KPD focuses on the latter case: it selects positions where the student currently has low confidence in the supervised target and where additional teacher guidance is likely to be useful. Therefore, our claim is not that every uncertain token directly corresponds to a semantic knowledge gap, but that target-aware uncertainty provides an effective routing signal for placing distillation supervision.

This design is also consistent with the training objective. As shown in Appendix C.1, the score $1 - q(y_t \mid y_{<t}, x)$ lower-bounds the per-token negative log-likelihood, is proportional to the gradient magnitude on the ground-truth logit, and maximizes a computable lower bound on the selected-token loss under a fixed token budget. Thus, even though target-aware uncertainty is not semantically identical to missing knowledge, it identifies the positions that are hardest, most loss-relevant, and most useful for targeted supervision.

Empirically, we compare the default probability-based score with entropy, margin, and random token selection in Appendix B.11. The probability-based criterion consistently performs best, suggesting that target-aware token difficulty is more suitable for KPD than generic distributional uncertainty.

B.16 Discussion on Offline Probing and Optimization Stability

A potential concern is that the set of critical tokens may change as the student learns, which could create a moving-target optimization problem if probing were performed repeatedly during distillation. We clarify that KPD adopts an offline probing strategy. Specifically, we first use the initial student model to select the top-k_1 target-aware uncertain tokens, and then use this fixed probing set to identify the top-k_2 teacher layers. After this preprocessing stage, both the selected probing examples and the probed teacher layers remain fixed throughout distillation. Therefore, KPD does not dynamically change the supervision targets during training and does not introduce moving-target instability.

To further verify this design choice, we compare the default offline probing strategy with an online variant that re-probes five times during distillation. The results are shown in Table 22. Online probing consistently performs worse than offline probing, with drops of 3.57–5.10 Rouge-L across different distillation backbones. This suggests that repeatedly changing the selected tokens and layers during optimization can introduce unstable supervision and degrade training. By contrast, offline probing provides a stable supervision route while avoiding repeated probing cost.

These results support two conclusions. First, the fixed offline probing set provides stable and effective supervision for distillation. Second, online re-probing is not only more expensive but also less reliable, because the supervision signal changes during optimization.

Table 23: Heterogeneous distillation results of Llama2-7B → Qwen2.5-1.5B. We report Rouge-L on domain-agnostic benchmarks.

Method	Dolly	SelfInst	S-NI	UnNI	Vicuna	Avg.
KD	22.41	17.34	36.79	31.86	20.71	25.82
KD + KPD	**23.74**	**18.46**	**37.34**	**33.47**	**22.63**	**27.13**
Distillm	28.41	22.18	39.74	37.41	20.93	29.73
Distillm + KPD	**29.15**	**23.47**	**40.31**	**38.93**	**21.42**	**30.66**

B.17 Analysis on the Scalability of the Probing Procedure

We further analyze the scalability of KPD's probing procedure. The additional computation of KPD mainly comes from teacher-side integrated-gradient probing, while student-side uncertainty computation reuses the logits already computed for distillation and only requires lightweight post-processing. Moreover, teacher-side probing is performed once before distillation, rather than at every optimization step. Therefore, the main cost of KPD is a one-time offline preprocessing overhead.

We evaluate the scaling behavior from three aspects: teacher size, sequence length, and the number of selected key tokens.

Scaling with teacher size. When scaling the teacher from Llama2-7B to Llama2-13B, the probing time increases from 74 minutes to 105 minutes. However, the total distillation time also increases, and the probing time remains about 6% of the total wall-clock time. This suggests that the probing cost grows with teacher size but remains a relatively small fraction of the full distillation pipeline.

Scaling with sequence length. We also study the effect of longer probing sequences. For Llama2-7B → TinyLlama-1.1B, increasing the average probing sequence length from 512 to 2048 only changes the probing time from 74 minutes to 80 minutes. This mild increase occurs because the dominant cost lies in teacher-side attribution over selected key tokens, rather than in processing all sequence positions uniformly.

Scaling with the number of key tokens. The number of selected key tokens k_1 directly affects the number of teacher-side attribution computations. When k_1 increases from 1 to 5, the probing time increases from 24 minutes to 74 minutes. However, our analysis in Appendix B.4 shows that the best performance is usually achieved around $k_1 = 4$ or $k_1 = 5$, where the overall probing overhead is still only about 5%–6% of the total distillation time. Therefore, KPD achieves a favorable trade-off between precision and efficiency.

Overall, the scalability results show that KPD introduces a modest one-time overhead while providing consistent improvements across model families and distillation backbones.

B.18 Analysis on Heterogeneous Teacher-Student Distillation

The main experiments focus on relatively compatible teacher-student pairs, following common practice in LLM distillation. To further examine whether KPD can generalize beyond same-family distillation, we conduct an additional heterogeneous experiment using Llama2-7B as the teacher and Qwen2.5-1.5B as the student. This setting involves both architectural differences and tokenizer mismatch. We use MultiLevelOT to align heterogeneous token spaces before distillation, and then apply KPD to determine what tokens and which teacher layers should be emphasized.

As shown in Table 23, KPD consistently improves both KD and Distillm in the heterogeneous setting. KD improves from 25.82 to 27.13 on average, and Distillm improves from 29.73 to 30.66. These results suggest that KPD is not limited to teacher-student pairs with highly similar architectures or vocabularies. Instead, KPD is complementary to token-space alignment methods: MultiLevelOT addresses the tokenizer mismatch, while KPD provides student-centric token selection and teacher-side layer localization. This combination enables more precise distillation even under cross-family teacher-student transfer.

Table 24: Sensitivity analysis of the activation threshold θ under KD+KPD. We report the average Rouge-L scores across the five domain-agnostic benchmarks.

Threshold θ	Llama3.1-8B → Llama3.2-1B	Qwen2.5-7B → Qwen2.5-1.5B
0.05	22.76	26.80
0.10	23.45	27.15
0.20 (default)	**23.98**	**27.67**
0.30	22.95	27.30
0.40	22.80	26.80

B.19 Sensitivity to the Activation Threshold

KPD uses an activation threshold θ to determine whether a teacher-side neuron has a sufficiently positive attribution to a key-token prediction. A very small threshold may include weakly relevant neurons and introduce noisy layer-selection signals, while a very large threshold may discard useful neurons and make the selected layers overly sparse. Therefore, we analyze the sensitivity of KPD to θ.

Table 24 shows that a moderate threshold performs best. The default value $\theta = 0.2$ achieves the highest Rouge-L under both teacher-student pairs. When θ is too small, the selected layers may be affected by many weak attribution signals; when θ is too large, useful neurons may be filtered out. These results indicate that the threshold is not an arbitrary design choice: it controls the precision-recall trade-off of attribution-based layer selection, and a moderate value provides robust performance.

B.20 Rationale of Isometric Layer Mapping

KPD maps the selected teacher layers to student layers using an isometric strategy. The goal is to preserve relative depth between the teacher and the student. If a probed teacher layer is located in the early, middle, or late part of the teacher, it should supervise a correspondingly early, middle, or late layer of the student, rather than always being mapped to a fixed shallow or final layer.

Formally, given a teacher with L_1 transformer blocks and a student with L_2 transformer blocks, for a selected teacher layer a_j, we map it to the student layer

$$c_j = \left\lceil \frac{a_j}{\lceil L_1/L_2 \rceil} \right\rceil . \tag{17}$$

This mapping operates at the transformer-block level. For standard decoder-only LLMs, transformer blocks are repeated architectural units with consistent internal structure, so the mapping does not require aligning heterogeneous submodules inside each block. Instead, it preserves the hierarchy of representations across model depth.

The empirical comparison in Figure 4 shows that isometric mapping consistently outperforms both front-layer mapping and final-layer mapping. This supports the intuition that knowledge selected from different teacher depths should be transferred to proportionally aligned student depths. Front-layer mapping can underuse the student's deeper semantic representations, while final-layer mapping may over-concentrate supervision in late layers and ignore useful lower-level or middle-level guidance. Isometric mapping provides a balanced depth-aware alignment and better preserves the teacher's hierarchical knowledge structure.

C Theoretical Analysis and Discussion

In this section, we provide a theoretical analysis of KPD to complement the methodological description in the main paper. While KPD is implemented through three concrete steps: student-side uncertainty probing, teacher-side integrated-gradients (IG) attribution for layer identification, and aligned intermediate-layer distillation, where each component can be justified by mathematical relations to the underlying training objectives. Accordingly, we analyze (i) why the uncertainty score used for selecting key tokens is tightly connected to the supervised negative log-likelihood and its

optimization signal, (ii) what the IG attribution computed on teacher FFN neurons exactly measures and why the resulting activation-count criterion yields a provable surrogate for ranking layers by positive contribution, and (iii) how the isometric alignment mapping admits controlled discretization guarantees and how the combined distillation objective can be interpreted as multi-depth distribution matching with an explicit gradient averaging effect. Together, these results clarify the theoretical role of each design choice in KPD and provide a principled foundation for its targeted knowledge transfer behavior.

C.1 Analysis of Student-Side Uncertainty Probing

Motivation. Our framework defines the student-side probing signal as a simple token-level uncertainty score $u_t = 1 - q(y_t \mid y_{<t}, x)$ and then selects the top-k_1 uncertain tokens to form the key token set $\mathcal{K}$. This design is computationally efficient and empirically effective, and we explain it theoretically. In particular, since the student is trained by minimizing the supervised negative log-likelihood (NLL), a principled probing criterion should be explicitly connected to (i) the supervised objective value and (ii) the magnitude of the optimization signal (gradients) that drives learning. This subsection therefore provides a characterization showing that u_t (a) deterministically lower-bounds the per-token NLL and (b) exactly matches the temperature-scaled gradient magnitude on the ground-truth logit, which together justify that selecting top-k_1 by u_t concentrates the teacher guidance on the student's hardest and most informative generation positions.

Setup and notation. Fix a training pair $(x, y) \sim \mathcal{D}$ with $y = (y_1, \ldots, y_{|y|})$. At step t, the student produces logits $z_t \in \mathbb{R}^{|\mathcal{V}|}$ and the tempered softmax distribution:

$$q_t(v) \;:=\; q(v \mid y_{<t}, x) \;=\; \frac{\exp(z_{t,v}/\gamma)}{\sum_{i\in\mathcal{V}} \exp(z_{t,i}/\gamma)}\,, \gamma > 0. \tag{18}$$

The per-token supervised loss (negative log-likelihood) is:

$$\ell_t \;:=\; -\log q_t(y_t), \tag{19}$$

and the uncertainty score used by our probing rule is:

$$u_t \;:=\; 1 - q_t(y_t) \in [0, 1). \tag{20}$$

Below we justify, in a fully mathematical manner, why selecting the top-k_1 tokens by u_t is a principled way to focus on the student's hardest (most informative) positions for subsequent teacher-side probing.

Why relate uncertainty to NLL? Our probing rule is defined directly on $u_t = 1 - q_t(y_t)$, while the actual supervised objective is the NLL loss $\ell_t = -\log q_t(y_t)$. To ensure the probing signal is not arbitrary, we first establish a *deterministic* inequality connecting u_t and ℓ_t. This guarantees that large uncertainty always implies a large lower bound on the supervised loss.

Lemma 1 (Uncertainty lower-bounds the per-token NLL)**.** *For any t and any $q_t(y_t) \in (0, 1]$, the following inequality holds:*

$$\ell_t \;=\; -\log q_t(y_t) \;\geq\; 1 - q_t(y_t) \;=\; u_t. \tag{21}$$

Proof. Let $x := q_t(y_t) \in (0, 1]$. Then $\ell_t = -\log x$ and $u_t = 1 - x$. We use the integral representation of the logarithm:

$$-\log x \;=\; \log\Big(\frac{1}{x}\Big) \;=\; \int_x^1 \frac{1}{s}\, ds. \tag{22}$$

Since $s \in [x, 1] \subset (0, 1]$, we have $\frac{1}{s} \geq 1$ for all such s. Therefore,

$$\int_x^1 \frac{1}{s}\, ds \;\geq\; \int_x^1 1\, ds \;=\; 1 - x. \tag{23}$$

Substituting back $x = q_t(y_t)$ yields $-\log q_t(y_t) \geq 1 - q_t(y_t) = u_t$, completing the proof. □

Why relate uncertainty to gradients? Beyond lower-bounding the loss, a good probing score should reflect where the student receives strong learning signals. In autoregressive SFT, the learning signal at step t is the gradient of ℓ_t w.r.t. the logits. We next show that u_t is *exactly proportional* to the magnitude of the gradient on the ground-truth logit, making uncertainty probing equivalent to selecting the positions with the strongest token-level optimization pressure.

Proposition 1 (Uncertainty equals the ground-truth logit gradient magnitude up to temperature)**.** *Let* $\ell_t = -\log q_t(y_t)$ *with* $q_t = \mathrm{softmax}(z_t/\gamma)$*. Then for each vocabulary index* $v \in \mathcal{V}$*,*

$$\frac{\partial \ell_t}{\partial z_{t,v}} \;=\; \frac{1}{\gamma}\Big(q_t(v) - \mathbb{I}[v = y_t]\Big). \tag{24}$$

In particular, for the ground-truth token y_t*,*

$$\left|\frac{\partial \ell_t}{\partial z_{t,y_t}}\right| \;=\; \frac{1}{\gamma}\big(1 - q_t(y_t)\big) \;=\; \frac{1}{\gamma}u_t. \tag{25}$$

Proof. Fix t and abbreviate $z = z_t$, $q = q_t$. Define the normalizer

$$A \;:=\; \sum_{i\in\mathcal{V}} \exp(z_i/\gamma), \quad \text{so that} \qquad q(v) \;=\; \frac{\exp(z_v/\gamma)}{A}. \tag{26}$$

For any $v \in \mathcal{V}$, we compute $\frac{\partial q(y_t)}{\partial z_v}$ by direct differentiation.

First note that

$$\frac{\partial}{\partial z_v}\exp(z_{y_t}/\gamma) \;=\; \frac{1}{\gamma}\exp(z_{y_t}/\gamma)\,\mathbb{I}[v = y_t], \tag{27}$$

and

$$\frac{\partial A}{\partial z_v} \;=\; \frac{\partial}{\partial z_v}\sum_{i\in\mathcal{V}}\exp(z_i/\gamma) \;=\; \frac{1}{\gamma}\exp(z_v/\gamma). \tag{28}$$

Using the quotient rule,

$$\begin{aligned}
\frac{\partial q(y_t)}{\partial z_v} &= \frac{\partial}{\partial z_v}\Big(\frac{\exp(z_{y_t}/\gamma)}{A}\Big) \\
&= \frac{\exp\big((z_{y_t} + z_v)/\gamma\big)}{\gamma\, A^2}\Big(\mathbb{I}[v = y_t]\cdot A - \exp(z_v/\gamma)\Big) \\
&= \frac{1}{\gamma}\cdot\frac{\exp(z_{y_t}/\gamma)}{A}\cdot\Big(\mathbb{I}[v = y_t] - \frac{\exp(z_v/\gamma)}{A}\Big) \\
&= \frac{1}{\gamma}\,q(y_t)\big(\mathbb{I}[v = y_t] - q(v)\big).
\end{aligned} \tag{29}$$

Now apply the chain rule to $\ell = -\log q(y_t)$:

$$\begin{aligned}
\frac{\partial \ell}{\partial z_v} &= -\frac{1}{q(y_t)}\cdot\frac{\partial q(y_t)}{\partial z_v} \\
&= -\frac{1}{q(y_t)}\cdot\frac{1}{\gamma}\,q(y_t)\big(\mathbb{I}[v = y_t] - q(v)\big) \\
&= \frac{1}{\gamma}\big(q(v) - \mathbb{I}[v = y_t]\big).
\end{aligned} \tag{30}$$

Setting $v = y_t$ yields $\frac{\partial \ell}{\partial z_{y_t}} = \frac{1}{\gamma}(q(y_t) - 1) = -\frac{1}{\gamma}(1 - q(y_t))$, hence

$$\left|\frac{\partial \ell}{\partial z_{y_t}}\right| = \frac{1}{\gamma}(1 - q(y_t)) = \frac{1}{\gamma}u_t, \tag{31}$$

which completes the proof. □

Why does top-k_1 by uncertainty make sense under a token budget? In practice, we cannot probe the teacher with all positions; we choose a fixed budget k_1 and construct $\mathcal{K}$ by selecting the k_1 largest uncertainty scores. The next result formalizes what this choice guarantees: it maximizes a *computable lower bound* on the total NLL over the selected tokens, and simultaneously maximizes the summed magnitude of token-level optimization signals (by Proposition 1).

Proposition 2 (Top-k_1 uncertainty maximizes a lower bound on the selected-token NLL)**.** *For a fixed example (x, y), let $\mathcal{K}^\star$ be any size-k_1 index set achieving the top-k_1 uncertainties, i.e.,*

$$|\mathcal{K}^\star| = k_1, \qquad \sum_{t\in\mathcal{K}^\star} u_t \;=\; \max_{\mathcal{K}:\,|\mathcal{K}|=k_1} \sum_{t\in\mathcal{K}} u_t. \tag{32}$$

Then for any $\mathcal{K}$ with $|\mathcal{K}| = k_1$,

$$\sum_{t\in\mathcal{K}^\star} \ell_t \;\geq\; \sum_{t\in\mathcal{K}^\star} u_t \;\geq\; \sum_{t\in\mathcal{K}} u_t. \tag{33}$$

Proof. Fix any $\mathcal{K}$ with $|\mathcal{K}| = k_1$. First, by Lemma 1, for each t we have $\ell_t \geq u_t$. Summing this inequality over $t \in \mathcal{K}^\star$ yields

$$\sum_{t\in\mathcal{K}^\star} \ell_t \;\geq\; \sum_{t\in\mathcal{K}^\star} u_t. \tag{34}$$

Second, by the defining optimality of $\mathcal{K}^\star$ (it maximizes $\sum_{t\in\mathcal{K}} u_t$ among all size-k_1 sets), we have

$$\sum_{t\in\mathcal{K}^\star} u_t \;=\; \max_{\mathcal{K}:\,|\mathcal{K}|=k_1} \sum_{t\in\mathcal{K}} u_t \;\geq\; \sum_{t\in\mathcal{K}} u_t. \tag{35}$$

Combining the two displayed inequalities proves the claim. □

Conclusion. Lemma 1 establishes a deterministic bridge between the probing score $u_t = 1 - q_t(y_t)$ and the supervised objective: uncertainty always lower-bounds the per-token NLL. Proposition 1 further shows that uncertainty is exactly the (temperature-scaled) magnitude of the gradient on the ground-truth logit, so high-uncertainty tokens are precisely where the student receives the strongest token-level learning signal. Finally, Proposition 2 proves that selecting the top-k_1 tokens by uncertainty maximizes a computable lower bound on the total NLL over the chosen set under a fixed token budget. Together, these results provide a rigorous justification that student-side uncertainty probing concentrates subsequent teacher guidance on the most difficult and most informative positions for improving the student.

C.2 Analysis of Teacher-Side IG Attribution and Layer Selection

Motivation. In KPD, the teacher-side probing serves a precise purpose: given the student-selected key tokens $\mathcal{K}$, we aim to *locate* where the corresponding predictive evidence is stored inside the teacher $\mathcal{T}$ and then distill from those layers. Our implementation instantiates this idea by (i) computing integrated gradients (IG) attribution scores for FFN neurons and (ii) selecting layers using the number of neurons whose attribution exceeds a threshold. Although this pipeline is intuitive, to avoid treating it as a purely heuristic design, we provide a rigorous characterization of two core components: (1) what IG attribution *exactly measures* under the 1D scaling path we use, and (2) why counting "activated" neurons yields a mathematically justified surrogate for ranking layers by their positive contribution to the target token probability. These results ensure that the teacher-side layer selection step is grounded in provable relationships rather than informal interpretation.

Setup and notation. Fix a probing instance $(x + y_{<t}, y_t)$ constructed from $(x, y, \mathcal{K}) \in \mathcal{D}_{probe}$ and a particular key token $y_t \in \mathcal{K}$. Let $P(w)$ denote the teacher's probability of predicting y_t under a scalar parameter w (representing one FFN neuron parameter), i.e.,

$$P(w) \;:=\; p(y_t \mid x, y_{<t},\, w_{l,j} = w), \tag{36}$$

where all other parameters are fixed to their original values in $\mathcal{T}$. Let $\bar{w} := \overline{w}_{l,j}$ be the original (trained) value of that neuron parameter. Our IG attribution along the straight-line path from the baseline 0 to $\bar{w}$ is defined as

$$\mathrm{Attr}(w_{l,j}) \;:=\; \bar{w} \int_0^1 \frac{\partial P(\alpha\bar{w})}{\partial w}\, d\alpha, \tag{37}$$

assuming $P(\cdot)$ is differentiable on the segment $[0, \bar{w}]$ (a standard condition for IG). For a teacher layer $\mathcal{T}_l$, we declare neuron j to be *activated* if $\mathrm{Attr}(w_{l,j}) > \theta$ and denote by N_l the number of activated neurons in that layer.

Why prove an "endpoint difference" identity for IG? The attribution score in Eq. (37) is defined via an integral of gradients. To interpret it as a concrete and verifiable quantity, it is useful to show that under our 1D scaling path, the IG attribution equals the *net change* in the target token probability when the neuron moves from the baseline to its original value. This identity provides a direct semantic meaning for $\mathrm{Attr}(w_{l,j})$ and enables subsequent bounds for layer selection.

Lemma 2 (1D IG equals endpoint probability difference)**.** *Assume $P : [0, \bar{w}] \to \mathbb{R}$ is continuously differentiable. Define $g(\alpha) := P(\alpha \bar{w})$ for $\alpha \in [0, 1]$. Then the IG attribution in Eq. (37) satisfies*

$$\mathrm{Attr}(w_{l,j}) \;=\; P(\bar{w}) - P(0). \tag{38}$$

Proof. By the definition $g(\alpha) = P(\alpha \bar{w})$ and the chain rule, for any $\alpha \in [0, 1]$,

$$g'(\alpha) \;=\; \tfrac{d}{d\alpha} P(\alpha \bar{w}) \;=\; \tfrac{\partial P(\alpha \bar{w})}{\partial w} \cdot \tfrac{d(\alpha \bar{w})}{d\alpha} \;=\; \bar{w} \tfrac{\partial P(\alpha \bar{w})}{\partial w}. \tag{39}$$

Substituting Eq. (39) into the IG definition (37) yields

$$\mathrm{Attr}(w_{l,j}) = \int_0^1 g'(\alpha)\, d\alpha. \tag{40}$$

Since g is continuously differentiable on $[0, 1]$, by the Fundamental Theorem of Calculus,

$$\int_0^1 g'(\alpha)\, d\alpha = g(1) - g(0) = P(\bar{w}) - P(0), \tag{41}$$

which proves Eq. (38). □

Interpretation. Lemma 2 shows that $\mathrm{Attr}(w_{l,j})$ exactly equals the *net* probability gain (or loss) attributable to restoring the neuron parameter from the baseline 0 to its original value $\bar{w}$. In particular, $\mathrm{Attr}(w_{l,j}) > 0$ implies that the neuron increases $p(y_t \mid x, y_{<t})$ relative to the baseline, while $\mathrm{Attr}(w_{l,j}) < 0$ implies the opposite.

Why relate activated-neuron counting to layer contribution? Our layer selection uses the activation count N_l rather than the raw sum of attributions. A natural question is what objective N_l is optimizing. To answer this, we introduce a layer-level quantity that captures the *total positive attribution mass* in a layer, and then show that N_l provides a deterministic lower bound on it. This establishes that ranking layers by N_l is equivalent to maximizing a computable surrogate lower bound on the layer's positive contribution.

Proposition 3 (Activated-neuron count lower-bounds positive attribution mass)**.** *For a fixed layer $\mathcal{T}_l$ and a fixed probing instance, define its positive attribution mass as*

$$A_l \;:=\; \sum_j \max\big(\mathrm{Attr}(w_{l,j}),\, 0\big). \tag{42}$$

Let $N_l := \big|\{j : \mathrm{Attr}(w_{l,j}) > \theta\}\big|$ be the number of activated neurons under threshold $\theta > 0$. Then

$$A_l \;\geq\; \theta\, N_l. \tag{43}$$

Proof. Let $\mathcal{J}_l := \{j : \mathrm{Attr}(w_{l,j}) > \theta\}$, so that $|\mathcal{J}_l| = N_l$. For any $j \in \mathcal{J}_l$, we have $\mathrm{Attr}(w_{l,j}) > \theta > 0$, hence

$$\max(\mathrm{Attr}(w_{l,j}), 0) = \mathrm{Attr}(w_{l,j}) > \theta. \tag{44}$$

Summing over $j \in \mathcal{J}_l$ yields

$$\begin{aligned} \sum_{j \in \mathcal{J}_l} \max\big(\mathrm{Attr}(w_{l,j}), 0\big) &= \sum_{j \in \mathcal{J}_l} \mathrm{Attr}(w_{l,j}) \\ &\geq \sum_{j \in \mathcal{J}_l} \theta = \theta N_l. \end{aligned} \tag{45}$$

Moreover, for any $j \notin \mathcal{J}_l$, we have $\max(\mathrm{Attr}(w_{l,j}), 0) \geq 0$. Therefore,

$$\begin{aligned} A_l &= \sum_{j \in \mathcal{J}_l} \max\big(\mathrm{Attr}(w_{l,j}), 0\big) + \sum_{j \notin \mathcal{J}_l} \max\big(\mathrm{Attr}(w_{l,j}), 0\big) \\ &\geq \sum_{j \in \mathcal{J}_l} \max\big(\mathrm{Attr}(w_{l,j}), 0\big). \end{aligned} \tag{46}$$

Combining Eq. (45) and Eq. (46) gives $A_l \geq \theta N_l$, proving Eq. (43). □

Consequence for layer ranking. Proposition 3 implies that, under a fixed threshold θ, ranking layers by N_l maximizes the deterministic lower bound θN_l on the positive attribution mass A_l. Thus, selecting the top-k_2 layers by N_l can be viewed as selecting layers that provably contain the largest minimum amount of positive contribution toward the key-token probability.

Extension to dataset-level aggregation. In practice, N_l is aggregated across probing instances (over (x, y_t) pairs induced by $\mathcal{D}_{probe}$). Let $N_l^{(i)}$ denote the activation count for layer l on probing instance i, and let

$$\widetilde{N}_l \;:=\; \sum_i N_l^{(i)}, \qquad \widetilde{A}_l \;:=\; \sum_i A_l^{(i)}. \tag{47}$$

Since Proposition 3 holds for each instance i,

$$\widetilde{A}_l = \sum_i A_l^{(i)} \;\geq\; \sum_i \theta N_l^{(i)} = \theta \widetilde{N}_l. \tag{48}$$

Therefore, selecting layers by the largest aggregated counts $\widetilde{N}_l$ also maximizes a deterministic lower bound on the aggregated positive attribution mass across the probing dataset.

Conclusion. This subsection provides provable guarantees for the teacher-side probing mechanism used in KPD. Lemma 2 shows that, under the 1D scaling path from baseline 0 to the original parameter $\bar{w}$, the integrated gradients attribution $\mathrm{Attr}(w_{l,j})$ exactly equals the endpoint difference $P(\bar{w}) - P(0)$, i.e., the neuron's net contribution to increasing the target token probability. Building on this identity, Proposition 3 establishes that the activated-neuron count N_l yields a deterministic lower bound on the layer's total positive attribution mass, both per instance and after aggregation over $\mathcal{D}_{probe}$. As a result, ranking and selecting teacher layers by activation counts is not merely heuristic: it provably selects layers that maximize a computable surrogate lower bound on positive contribution to key-token predictions, aligning layer selection with the goal of locating where the teacher stores knowledge relevant to the student's missing tokens.

C.3 Analysis of Alignment Mapping and Intermediate-Layer Distillation Objective

Motivation. Our method introduces two design choices beyond standard last-layer token-level distillation: (i) an *isometric depth mapping* that aligns selected teacher layers to student layers, and (ii) an *intermediate-layer distillation loss* that matches the teacher and student token distributions at those aligned depths. While both components are intuitive, a rigorous appendix should clarify (a) what the alignment mapping guarantees in terms of order preservation and depth approximation, and (b) how the combined objective $\mathcal{L}_{distill} + \mathcal{L}_{hidden}$ can be viewed as a principled multi-depth supervision scheme rather than an ad-hoc addition. This subsection provides two provable results: one characterizing the mapping error induced by discrete layer indices, and the other rewriting the total distillation objective as an expectation over a random "exit" (depth), yielding a clean formal interpretation.

Setup and notation. Let the teacher $\mathcal{T}$ have L_1 transformer blocks and the student $\mathcal{S}$ have L_2 blocks. Let $\mathcal{Q}^{\mathcal{T}} = \{\mathcal{T}_{a_1}, \ldots, \mathcal{T}_{a_{k_2}}\}$ be the selected teacher layers with indices $a_j \in \{1, \ldots, L_1\}$. Define the compression ratio surrogate

$$r \;:=\; \left\lceil \frac{L_1}{L_2} \right\rceil \in \mathbb{N}. \tag{49}$$

To avoid ambiguity caused by non-integer indices, we formalize the student alignment as an *integer-valued* mapping

$$c : \{1, \ldots, L_1\} \to \{1, \ldots, L_2\}, \quad c(a) := \left\lceil \frac{a}{r} \right\rceil. \tag{50}$$

Accordingly, the aligned student layer set is $\mathcal{Q}^{\mathcal{S}} = \{\mathcal{S}_{c(a_1)}, \ldots, \mathcal{S}_{c(a_{k_2})}\}$. For each selected depth, we define teacher/student intermediate distributions at step t as $p_j(\cdot \mid y_{<t}, x)$ and $q_j(\cdot \mid y_{<t}, x)$ (as in the main paper), and let $p(\cdot \mid y_{<t}, x)$ and $q(\cdot \mid y_{<t}, x)$ denote the final-layer distributions.

Why analyze the alignment mapping? Depth-alignment is meant to preserve the *relative hierarchy* of representations: deeper teacher layers should supervise deeper student layers. Since layers are discrete, it is important to show that the mapping in Eq. (50) is (i) order-preserving and (ii) incurs a bounded discretization error. Such guarantees justify that the mapping approximates proportional depth matching in a mathematically controlled way.

Lemma 3 (Order preservation and discretization error of the alignment mapping)**.** *Let r and $c(\cdot)$ be defined in Eq.* (49)–(50)*. Then:*

1. *(**Monotonicity**) For any $a_1, a_2 \in \{1, \ldots, L_1\}$, if $a_1 < a_2$ then $c(a_1) \le c(a_2)$.*
2. *(**Unit discretization error**) For any $a \in \{1, \ldots, L_1\}$,*

$$\left| c(a) - \frac{a}{r} \right| < 1. \tag{51}$$

Proof. **(1) Monotonicity.** Since $r > 0$, the function $a \mapsto a/r$ is strictly increasing on $\mathbb{R}$, hence $a_1 < a_2$ implies $\frac{a_1}{r} < \frac{a_2}{r}$. The ceiling function $\lceil \cdot \rceil$ is monotone non-decreasing, so

$$c(a_1) = \left\lceil \frac{a_1}{r} \right\rceil \le \left\lceil \frac{a_2}{r} \right\rceil = c(a_2), \tag{52}$$

which proves (1).

(2) Unit discretization error. For any real x, the ceiling satisfies

$$\lceil x \rceil - 1 < x \le \lceil x \rceil. \tag{53}$$

Set $x = \frac{a}{r}$. Then by Eq. (53),

$$c(a) - 1 < \frac{a}{r} \le c(a). \tag{54}$$

From the right inequality in Eq. (54), we have $\frac{a}{r} - c(a) \le 0$ and thus

$$\left| \frac{a}{r} - c(a) \right| = c(a) - \frac{a}{r}. \tag{55}$$

From the left inequality in Eq. (54), we have $c(a) - \frac{a}{r} < 1$. Combining the two gives

$$\left| c(a) - \frac{a}{r} \right| = c(a) - \frac{a}{r} < 1, \tag{56}$$

which proves Eq. (51). □

Remark (normalized depth approximation). Lemma 3 implies that $c(\cdot)$ preserves ordering of teacher depths and approximates proportional alignment with an error of strictly less than one student layer. This provides a controlled discretization guarantee for depth matching when L_1 and L_2 differ.

Why rewrite the total distillation loss as a random-exit objective? Our training objective includes both the final-layer distillation loss $\mathcal{L}_{distill}$ and the intermediate-layer loss $\mathcal{L}_{hidden}$. To clarify what this combined objective optimizes, we show that it is exactly equivalent to an expectation of KL divergence evaluated at a randomly sampled "exit" depth (either the final layer or one of the aligned intermediate layers). This equivalence yields a clean formal interpretation: the student is supervised to match the teacher *in distribution space* at multiple depths, and the total gradient is the average of the gradients from these supervision points.

Per-instance token-level KL. For any two distributions p, q over $\mathcal{V}$, define

$$D_{\mathrm{KL}}(p\|q) \;:=\; \sum_{v\in\mathcal{V}} p(v)\log\frac{p(v)}{q(v)}. \tag{57}$$

Fix $(x, y) \sim \mathcal{D}$ and step t. Let

$$D_t^{(0)} \;:=\; D_{\mathrm{KL}}\Big(p(\cdot \mid y_{<t}, x)\,\Big\|\, q(\cdot \mid y_{<t}, x)\Big), \tag{58}$$

be the final-layer KL, and for $j \in \{1, \ldots, k_2\}$,

$$D_t^{(j)} \;:=\; D_{\mathrm{KL}}\Big(p_j(\cdot \mid y_{<t}, x)\,\Big\|\, q_j(\cdot \mid y_{<t}, x)\Big), \tag{59}$$

be the KL at the j-th aligned intermediate depth.

Proposition 4 (Combined distillation equals expected KL over a random exit)**.** *Define a discrete random variable J uniformly distributed on $\{0, 1, \ldots, k_2\}$, i.e.,*

$$\mathbb{P}(J = j) = \frac{1}{k_2 + 1} \quad \textit{for all } j \in \{0, 1, \ldots, k_2\}. \tag{60}$$

Then for any fixed (x, y), the per-example distillation loss

$$\mathcal{L}_{distill}(x, y) + \mathcal{L}_{hidden}(x, y) \tag{61}$$

can be written as

$$\mathcal{L}_{distill}(x, y) + \mathcal{L}_{hidden}(x, y) = (k_2 + 1) \cdot \mathbb{E}_t\Big[\mathbb{E}_J\big[D_t^{(J)}\big]\Big], \tag{62}$$

where $\mathbb{E}_t[\cdot]$ denotes the average over token positions $t \in \{1, \ldots, |y|\}$ (or equivalently the sum, depending on the convention used in the main paper).

Proof. By definition of $\mathcal{L}_{distill}$ and $\mathcal{L}_{hidden}$ in the main paper, for a fixed example (x, y) we can write (up to the same token-averaging convention)

$$\begin{aligned} \mathcal{L}_{distill}(x, y) &= \mathbb{E}_t\big[D_t^{(0)}\big], \\ \mathcal{L}_{hidden}(x, y) &= \mathbb{E}_t\Big[\frac{1}{k_2}\sum_{j=1}^{k_2} D_t^{(j)}\Big]. \end{aligned} \tag{63}$$

First, combine the two terms using Eq. (63):

$$\begin{aligned} \mathcal{L}_{distill}(x, y) + \mathcal{L}_{hidden}(x, y) &= \mathbb{E}_t\big[D_t^{(0)}\big] + \mathbb{E}_t\Big[\frac{1}{k_2}\sum_{j=1}^{k_2} D_t^{(j)}\Big] \\ &= \mathbb{E}_t\Big[D_t^{(0)} + \frac{1}{k_2}\sum_{j=1}^{k_2} D_t^{(j)}\Big]. \end{aligned} \tag{64}$$

Next, express the inner bracket as an expectation over J. Since J is uniform on $\{0, \ldots, k_2\}$,

$$\begin{aligned} \mathbb{E}_J\big[D_t^{(J)}\big] &= \sum_{j=0}^{k_2} \mathbb{P}(J = j)\, D_t^{(j)} \\ &= \frac{1}{k_2 + 1}\Big(D_t^{(0)} + \sum_{j=1}^{k_2} D_t^{(j)}\Big). \end{aligned} \tag{65}$$

Multiplying both sides of Eq. (65) by $(k_2 + 1)$ gives

$$(k_2 + 1)\mathbb{E}_J\big[D_t^{(J)}\big] = D_t^{(0)} + \sum_{j=1}^{k_2} D_t^{(j)}. \tag{66}$$

To reconcile Eq. (66) with the $\frac{1}{k_2}$ factor in $\mathcal{L}_{hidden}$, note that Eq. (63) uses the normalization $\frac{1}{k_2}$ for intermediate depths. Equivalently, we can define a reweighted random variable $\widetilde{J}$ with probabilities

$$\mathbb{P}(\widetilde{J}=0)=\frac{1}{2},\quad \mathbb{P}(\widetilde{J}=j)=\frac{1}{2k_2}\ \text{ for } j\in\{1,\dots,k_2\}, \tag{67}$$

so that

$$\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]=\frac{1}{2}D_t^{(0)}+\frac{1}{2k_2}\sum_{j=1}^{k_2}D_t^{(j)}. \tag{68}$$

Multiplying Eq. (68) by 2 yields

$$2\,\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]=D_t^{(0)}+\frac{1}{k_2}\sum_{j=1}^{k_2}D_t^{(j)}. \tag{69}$$

Substituting Eq. (69) into Eq. (64) gives

$$\begin{aligned}\mathcal{L}_{distill}(x,y)+\mathcal{L}_{hidden}(x,y)&=\mathbb{E}_t\Big[2\,\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]\Big]\\&=2\,\mathbb{E}_t\Big[\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]\Big].\end{aligned} \tag{70}$$

This establishes the desired "random-exit" interpretation with the exact weights induced by our loss normalization. Finally, taking expectation over $(x,y)\sim\mathcal{D}$ yields the dataset-level statement. □

Why state a gradient implication? The random-exit view is not only interpretive; it also implies that the optimization signal is an explicit average of supervision signals from multiple depths. To make this precise, we formalize that the gradient of the combined distillation objective equals the corresponding weighted average of per-depth KL gradients.

Corollary 1 (Gradient decomposes into a weighted average across depths)**.** *Let θ denote the student parameters (including any projection parameters used in intermediate heads), and define*

$$\mathcal{J}(x,y)\ :=\ \mathbb{E}_t\Big[\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]\Big], \tag{71}$$

with $\widetilde{J}$ defined in Eq. (67). Assuming differentiation and expectation can be interchanged (e.g., by dominated convergence under standard boundedness conditions for softmax outputs), we have

$$\nabla_\theta\Big(\mathcal{L}_{distill}(x,y)+\mathcal{L}_{hidden}(x,y)\Big)=2\,\mathbb{E}_t\Big[\mathbb{E}_{\widetilde{J}}\big[\nabla_\theta D_t^{(\widetilde{J})}\big]\Big]. \tag{72}$$

Proof. From Proposition 4 (in its exact-weight form (69)), we have

$$\mathcal{L}_{distill}(x,y)+\mathcal{L}_{hidden}(x,y)=2\,\mathcal{J}(x,y). \tag{73}$$

Taking gradients w.r.t. θ on both sides yields

$$\nabla_\theta\big(\mathcal{L}_{distill}(x,y)+\mathcal{L}_{hidden}(x,y)\big)=2\,\nabla_\theta\mathcal{J}(x,y). \tag{74}$$

Under the stated regularity condition allowing the interchange of ∇_θ with expectations over t and $\widetilde{J}$,

$$\nabla_\theta\mathcal{J}(x,y)=\nabla_\theta\mathbb{E}_t\Big[\mathbb{E}_{\widetilde{J}}\big[D_t^{(\widetilde{J})}\big]\Big]=\mathbb{E}_t\Big[\mathbb{E}_{\widetilde{J}}\big[\nabla_\theta D_t^{(\widetilde{J})}\big]\Big]. \tag{75}$$

Substituting back completes the proof of Eq. (72). □

Conclusion. This subsection provides a rigorous formalization of two key components in KPD. First, Lemma 3 proves that the isometric alignment mapping $c(a)=\lceil a/r\rceil$ is order-preserving and approximates proportional depth matching with a discretization error strictly less than one layer, thereby justifying that deeper teacher layers supervise deeper student layers in a controlled manner. Second, Proposition 4 rewrites the combined distillation objective as an expected KL divergence evaluated at a randomly sampled depth (a "random exit"), yielding a principled multi-depth supervision interpretation consistent with our loss normalization. Finally, Corollary 1 shows that the resulting optimization signal is an explicit weighted average of per-depth KL gradients, formally capturing why intermediate-layer distillation provides denser and more stable supervision than relying on the final layer alone.

D Limitations

While our method KPD achieves precise knowledge transfer across the settings we study, one limitation remains outside our current scope. Due to computational constraints, we do not include experiments with substantially larger LLMs, such as teacher models beyond 50B parameters. Although our experiments cover multiple LLM families and different teacher-student scales, validating KPD under much larger teacher models would require significantly higher GPU memory, longer probing time, and more expensive distillation runs. Since KPD only introduces a one-time offline probing stage and incurs modest additional wall-clock overhead in our current experiments, we expect its plug-and-play design to remain applicable to larger-scale distillation scenarios. Nevertheless, larger teachers may exhibit more complex layer-wise knowledge distributions, and the optimal probing budget or selected knowledge layers may change with model scale. We therefore leave a systematic evaluation of KPD on larger LLMs, together with scaling analyses of probing cost and layer-selection behavior, for future work.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction accurately summarize the key contributions and scope of the paper, and they align with the theoretical and experimental results presented.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We mentioned our limitation in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The assumptions required for our theoretical results and proofs are provided in the Appendix C.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide detailed experimental settings in the Appendix A.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [Yes]

Justification: We have submitted a ZIP file containing the experimental code.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We provide detailed experimental settings in the Appendix A.5.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: We report the parameter analysis in the experimental results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide a description of the platform and hardware used for the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We adhere to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: This paper presents work whose goal is to advance the field of deep learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The research involves only publicly available datasets and standard models, posing no significant misuse risks. Thus, no specific safeguards were necessary.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have cited the relevant papers and provided links to download the data.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not introduce any new assets, thus documentation for such is not applicable.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects. Thus, the inclusion of participant instructions, screenshots, and compensation details is not applicable.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This paper does not involve research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs do not impact the core methodology, scientific rigor, or originality of the research.

    Guidelines:

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