Towards Precise Knowledge Distillation for Large Language Models via Knowledge Probing

Anonymous ACL submission

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 cumulative gradient calculation 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. Further ablations show that probing both the student and teacher yields more precise distillation. Our code is available at `https://anonymous.4open.science/r/KPD-CA0D`.

1 Introduction

Large language models (LLMs) (Brown et al., 2020; Achiam et al., 2023; Anil et al., 2023; Yang et al., 2024) have brought transformative advances to a wide range of natural language processing (NLP) tasks, including text generation (Liang et al., 2024; Liu et al., 2025), code completion (Eghbali and Pradel, 2024; Wang et al., 2024), and question answering (Zhuang et al., 2023; Monteiro et al., 2024). However, their ever-growing parameter scales lead to severe computational and deployment challenges, especially in latency-sensitive or resource-limited environments (Li and Duan, 2024; Jin et al., 2025). 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 (Li et al., 2024; Finkelstein and Freitag, 2024; He et al., 2025).

Current LLM distillation techniques can be grouped into two major paradigms: black-box and white-box distillation (Zhong et al., 2024; Xu et al., 2024a). The black-box paradigm (Hsieh et al., 2023; Ho et al., 2023; Zhang et al., 2023) 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 (Agarwal et al., 2024; Gu et al., 2024; Ko et al., 2024) directly aligns hidden representations between the teacher and student to enable finer-grained knowledge transfer.

Despite their advantages, the existing white-box paradigm still suffers from two key limitations: **(1) Lack of knowledge awareness.** Existing methods treat all tokens and layers uniformly, failing to identify what knowledge is most essential. As shown in Figure 1 (a), the student model assigns particularly low confidence to the token like *movies*, which suggests uncertainty and a knowledge gap. However, Figure 1 (b) shows that the final layers of the teacher model treat all tokens with similar confidence, making it difficult to isolate critical knowledge solely from surface-level outputs. **(2)**

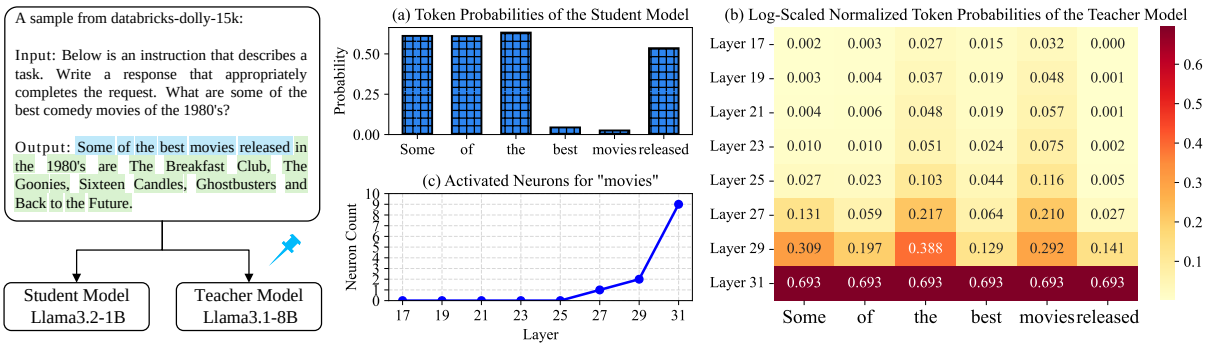

Figure 1: We use Llama 3.1-8B for the teacher model and Llama 3.2-1B for the student model to probe a sample from `databricks-dolly-15k`. (a) shows the prediction results of the student model for the first 6 tokens. (b) shows the output probability in each layer of the teacher model after layer normalization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

Inaccurate knowledge transfer. Heuristically selected layers may not contain the precise knowledge the student lacks. From Figure 1 (b)-(c), we observe that intermediate layers (e.g., 27, 29) of the teacher assign higher focus and neuron activations to the token *movies*, which the student struggles with. It implies that key knowledge resides in specific layers and can be revealed via neuron probing.

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 receives 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. Our contributions are summarized 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, which can be incorporated into diverse distillation approaches to 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.
- We demonstrate through extensive experiments and ablations that KPD improves various baseline distillation methods across multiple datasets and LLM families, achieving up to 4.56% improvement in Rouge-L.

2 Related Work

Knowledge Distillation. Knowledge distillation (Buciluǎ et al., 2006; Hinton et al., 2015; Hu et al., 2023) is widely used for compressing LLMs (Zhong et al., 2024; Xu et al., 2024a; Shing et al., 2024; Sun et al., 2025; Ko et al., 2025; Wang et al., 2025). In generative LMs, methods are categorized as black- and white-box (Xu et al., 2024a). Black-box approaches fine-tune students with teacher-generated CoTs (Hsieh et al.,

2023; Ho et al., 2023; Zhang et al., 2023). White-box distillation aligns teacher and student representations at the token (Sanh et al., 2019) or sequence (Kim and Rush, 2016; Wei et al., 2024) level. Subsequent work reduces exposure bias by mixing outputs (Lin et al., 2020), adds sequence-level f-divergence (Wen et al., 2023), and refines objectives via reverse KL (Gu et al., 2024), on-policy learning (Agarwal et al., 2024), or skewed KL off-policy updates (Ko et al., 2024). To handle vocabulary mismatches, DSKD (Zhang et al., 2024b) and MultiLevelOT (Cui et al., 2025) align hidden states across tokenizers. However, prior works lack guidance on what and where to distill, which limits the distillation performance.

Knowledge Probing. Knowledge probing analyzes how LLMs encode linguistic and factual information (Youssef et al., 2023; Ju et al., 2025). Traditional methods train classifiers on labeled data to test whether properties such as syntax or semantics are linearly decodable from hidden states (Conneau et al., 2018; Tenney et al., 2019; Hewitt and Liang, 2019). Recent work probes emergent capabilities truthful awareness (Azaria and Mitchell, 2023), in-context knowledge (Ju et al., 2024), and safety alignment (Xu et al., 2024b), showing that knowledge is sparsely distributed across layers and neurons (Belinkov, 2022; Zhao et al., 2024). Another line treats feedforward networks (FFNs) as structured memory to localize factual knowledge (Geva et al., 2021); integrated gradients identify knowledge neurons that strongly influence factual predictions (Dai et al., 2022). In parallel, intervention-based methods such as amnesic probing (Elazar et al., 2021) and LEACE (Belrose et al., 2023) suppress access to specific properties by removing key representational directions. Building on these insights, we adopt a probing-driven approach for precise distillation for LLMs.

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 of $|\mathcal{V}|$ tokens. The output sequence up to the t-th token is denoted as $y_{<t} = (y_1, y_2, ..., y_t)$, 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 can be denoted as $p(y_t|y_{<t}, x)$ as follows:

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

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 is defined as:

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

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 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:

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

More distillation losses are listed in Appendix A.

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 needs and where the teacher provides it.

3.3 Uncertainty-Based Probing for Student $\mathcal{S}$

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

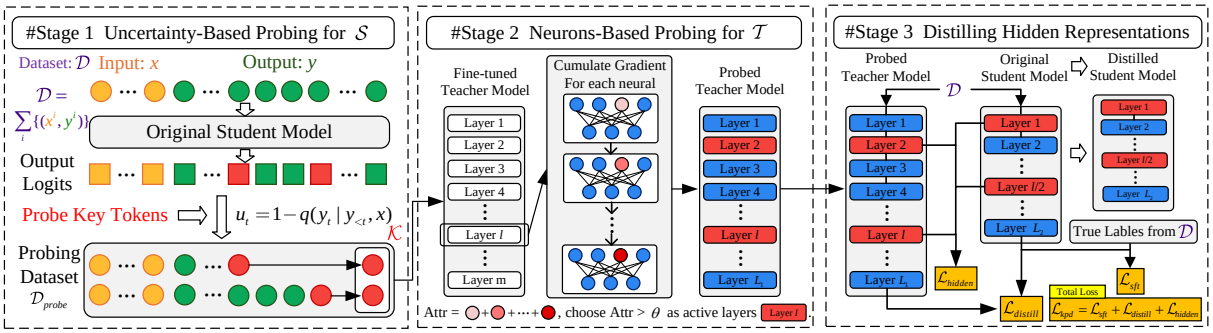

Figure 2: An overview of KPD framework.

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 for each generated token $y_t \in y$ as:

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

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. The construction details of $\mathcal{D}_{probe}$ are in Appendix B.5.

3.4 Neurons-Based Probing for Teacher $\mathcal{T}$

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 us to accurately 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 $y_t \in \mathcal{K}$. Inspired by the Knowledge Neurons (Dai et al., 2022), we evaluate the influence of individual FFN neurons on the prediction of y_t to locate key layers of stored knowledge. 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:

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

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 (Russell, 2020):

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

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.

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 $\mathcal{S}$. The algorithm process for the probing is shown in Appendix B.2, and the visualization of probing is shown in Appendix C.5.

3.5 Intermediate Representation Distilling

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 of the teacher model $\mathcal{T}$, we define an isometric mapping to assign corresponding layers $\mathcal{Q}^{\mathcal{S}}$ in $\mathcal{S}$:

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

where $c_j = a_j / \lceil \frac{L_1}{L_2} \rceil$ 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 $\mathcal{T}$. Similarly, we denote the probability of predicting the t-th token in the c_j-th layer of $\mathcal{S}$ 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 $\mathcal{S}$. Finally, the hidden loss is calculated as follows:

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

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**) (Conover et al., 2023), and test on **Dolly**, Self-Instruct (**SelfInst**) (Wang et al., 2022), Vicuna Evaluation (**Vicuna**) (Chiang et al., 2023), Super Natural Instructions (**S-NI**) (Wang et al., 2022), and Unnatural Instructions (**UnNI**) (Honovich et al., 2023). For domain-specific datasets, we proceed as follows: (1) **Medical**: we distill on GenMedGPT (Li et al., 2023) and evaluate on MMLU (Hendrycks et al., 2021) medical tasks: clinical knowledge (**CK**), professional medicine (**PM**), and college medicine (**CM**). (2) **Legal**: we distill on Lawyer-Instruct (Alignment-Lab-AI, 2024) and USTerms (Chalkidis et al., 2023) and evaluate on MMLU law tasks: international law (**IL**) and professional law (**PL**). Details of the datasets are shown in Appendix B.1.

Models. For student models, we choose TinyLlama-1.1B (Zhang et al., 2024a), Llama3.2-1B (Grattafiori et al., 2024), and Qwen2.5-1.5B (Yang et al., 2024). For teacher models, we choose Llama2-7B (Touvron et al., 2023), Llama3.1-8B (Grattafiori et al., 2024), and Qwen2.5-7B (Yang et al., 2024). We conduct three distillation: 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.

Baselines. We apply our KPD framework to 9 distillation methods for LLMs: KD (Hinton et al., 2015), RKD (Gu et al., 2024), SeqKD (Kim and Rush, 2016), ImitKD (Lin et al., 2020), GKD (Agarwal et al., 2024), MiniLLM (Gu et al., 2024), Distillm (Ko et al., 2024), Distillm-2 (Ko et al., 2025), and ABKD (Wang et al., 2025). Details of baselines are in Appendix B.3.

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. More details are shown in Appendix B.4.

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

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

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

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

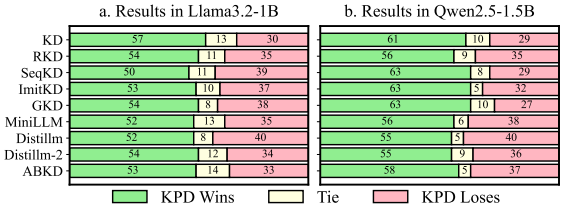

Figure 3: Main results of Llama3.2-1B and Qwen2.5-1.5B with GPT-4 scores on domain-agnostic datasets.

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 C.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 various baselines.

Domain-Specific Results. To test KPD on domain-specific settings, we perform distillation

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

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

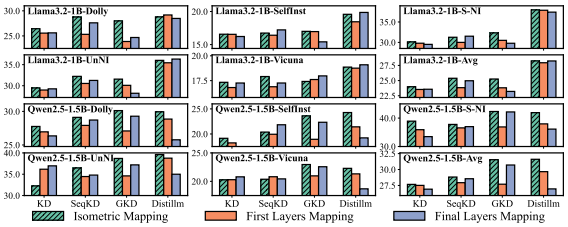

Figure 4: Results of mapping strategies: (1) Isometric Mapping aligns student layers proportionally with selected teacher layers; (2) Front-Layer Mapping uses only the first several student layers; (3) Final-Layer Mapping uses only the last several student layers.

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 C.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 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 C.2. **w/o student-probe** replaces uncertainty tokens with random ones, and **w/o teacher-probe** replaces layer selection with 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.

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 4. 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 all layers.

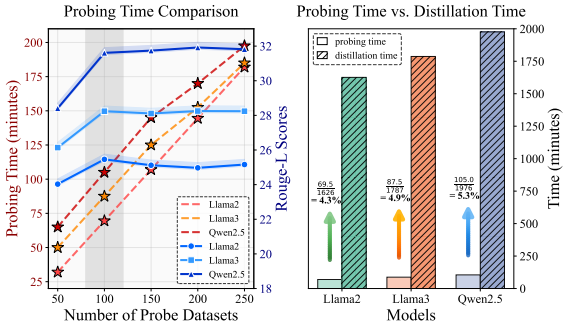

Figure 5: Analysis of the probing datasets scale and the comparison of probing time and average training time.

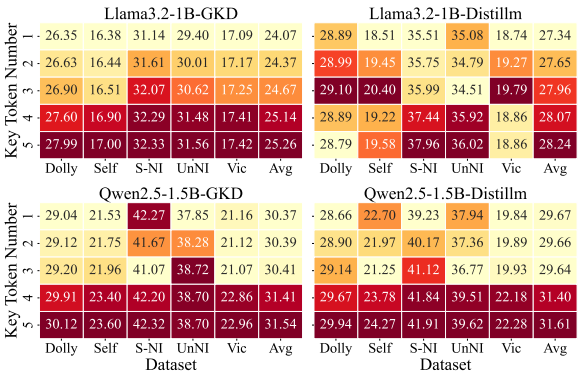

Figure 6: Results of probing token number k_1 from 1 to 5. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

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 5 (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

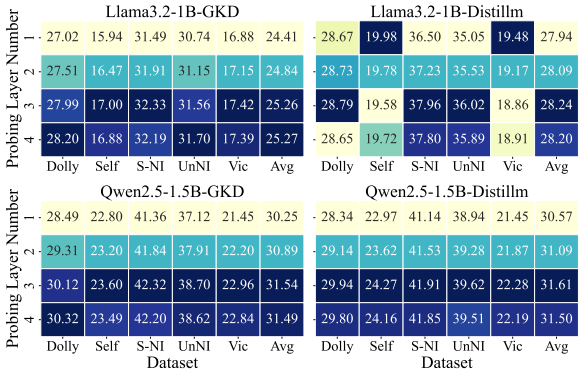

Figure 7: Results of probing layer number k_2 from 1 to 4. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

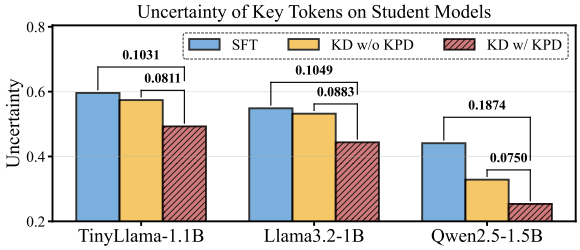

Figure 8: Results of the average uncertainty of key tokens in probing datasets across different student models.

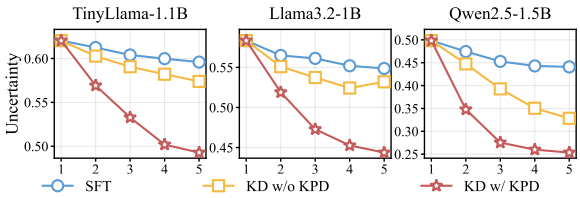

Figure 9: Results of the average uncertainty of key tokens in probing datasets during training epochs.

whole distillation process. Figure 5 (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.

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 6 and Appendix C.3. 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. 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. In practice, setting k_1 to 4-5 achieves a good trade-off between effectiveness and stability.

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 7 and Appendix C.4. 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. Thus, a moderate number of teacher layers ($k_2 = 3$) offers a good balance between knowledge diversity and transfer quality.

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 8, KPD consistently lowers uncertainty across students. For example, KPD decreases 0.0750-0.0883 and 0.1031-0.1874 compared to SFT and vanilla KD, respectively. As shown in Figure 9, 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. These reductions indicate that KPD concentrates supervision on the tokens the student initially finds most uncertain, yielding more precise knowledge distillation for LLMs than vanilla KD.

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.

Limitations

While our method KPD achieves precise knowledge transfer across the settings we study, two aspects remain outside our current scope. First, we do not systemically evaluate heterogeneous teacher-student distillation pairs (e.g., models of different series). In such cases, direct KL-based alignment of outputs or intermediate states is non-trivial due to vocabulary and representation mismatches. We view this as an engineering and interface challenge rather than a conceptual limitation and plan to extend KPD with tokenizer bridging and representation adapters (e.g., learned projections or transport-style matching) and to design architecture-aware probing for robust layer selection across models. Second, due to computational constraints, we do not include experiments with larger LLMs (e.g., > 50B). Given KPD's plug-and-play nature and modest wall-clock overhead, we expect the approach to transfer, but a thorough validation at that scale is left for future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

Alignment-Lab-AI. 2024. Lawyer-instruct.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, and 1 others. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

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

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

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

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

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *SIGKDD*, pages 535–541.

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

Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*.

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

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

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

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

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

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

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

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Hongbo Li and Lingjie Duan. 2024. Theory of mixture-of-experts for mobile edge computing. *arXiv preprint arXiv:2412.15690*.

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

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

Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, and 1 others. 2024. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*.

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

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

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Bertrand Russell. 2020. *Principles of mathematics*. Routledge.

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

Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. 2024. Taid: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. In *NeurIPS*.

Lin Sun, Guangxiang Zhao, Xiaoqi Jian, Yuhan Wu, Weihong Lin, Yongfu Zhu, Linglin Zhang, Jinzhu Wu, Junfeng Ran, Sai-er Hu, and 1 others. 2025. Tinyr1-32b-preview: Boosting accuracy with branch-merge distillation. *arXiv preprint arXiv:2503.04872*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, and

1 others. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

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

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, and 1 others. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

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

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

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

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

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

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

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

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

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

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

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

Appendix Contents

A Distillation Losses for LLMs **13**
- A.1 Token-level Distillation . . . 13
- A.2 Sequence-Level Distillation . . . 13

B Methodological and Experimental Details **13**
- B.1 Details of Datasets . . . 13
- B.2 Details of Knowledge Probing Algorithm . . . 14
- B.3 Details of Baselines . . . 14
- B.4 Details of Experimental Parameters . . . 15
- B.5 Details of the Probing Dataset . . . 16

C Additional Experimental Results **16**
- C.1 Main Results of TinyLlama-1.1B . . . 16
- C.2 Ablation Results of Qwen2.5-1.5B . . . 17
- C.3 Analysis Results of Number of Probing Tokens k_1 . . . 17
- C.4 Analysis Results of Number of Probing Layers k_2 . . . 17
- C.5 Visualization of Knowledge Probing in Teacher Models . . . 17
- C.6 Case Study . . . 19

A 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.

A.1 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 4, where $\mathrm{KL}(\cdot||\cdot)$ and $\mathrm{JSD}(\cdot||\cdot)$ between two probability distributions $p(\cdot)$ and $q(\cdot)$ can be defined as follows:

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

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

A.2 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 probability value for the t-th token $y_t \in y$ of the teacher model and the student model, respectively. Then, the sequence-level distillation loss can be defined as:

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

Compared with Equation 15 and the token-level forward KL divergence loss in Table 4, we observe that the sequence-level distillation only takes the single probability value of tokens in output y in each generative step, while the token-level distillation takes the whole probability distribution of the vocabulary in each generative step.

B Methodological and Experimental Details

B.1 Details of Datasets

Domain-Agnostic Datasets. For domain-agnostic datasets, we conduct experiments on five datasets: Databricks-Dolly-15k (**Dolly**) (Conover et al., 2023), Self-Instruct (**SelfInst**) (Wang et al., 2022), Vicuna Evaluation (**Vicuna**) (Chiang et al., 2023), Super Natural Instructions (**S-NI**) (Wang et al., 2022), and Unnatural Instructions (**UnNI**) (Honovich et al., 2023). 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** (Conover et al., 2023): 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 (Ouyang et al., 2022).

- **Self-Instruct** (Wang et al., 2022): 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 applications, along with 50K examples collected from public resources for benchmarking purposes.

- **Vicuna** (Chiang et al., 2023): The whole Vicuna dataset contains 70K samples of user-shared conversations. In our experiments, following the settings of (Gu et al., 2024), we choose 80 challenging questions to evaluate the model performance.

- **Super-Natural Instructions** (Wang et al., 2022): 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** (Honovich et al., 2023): 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.

Distillation Methods	Formula
Forward KL Divergence (Hinton et al., 2015)	$\mathcal{L}_{\text{FKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\lvert y\rvert} \text{KL}(p(\cdot\lvert y_{<t}, x)\lVert q(\cdot\lvert y_{<t}, x))\right]$
Reverse KL Divergence (Gu et al., 2024)	$\mathcal{L}_{\text{RKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\lvert y\rvert} \text{KL}(q(\cdot\lvert y_{<t}, x)\lVert p(\cdot\lvert y_{<t}, x))\right]$
Jensen-Shannon Divergence (Agarwal et al., 2024)	$\mathcal{L}_{\text{JS}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\lvert y\rvert} \text{JSD}(p(\cdot\lvert y_{<t}, x)\lVert q(\cdot\lvert y_{<t}, x))\right]$
Skew (Reverse) KL Divergence (Ko et al., 2024)	$\mathcal{L}_{\text{SKL}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{\lvert y\rvert} \text{KL}(p(\cdot\lvert y_{<t}, x)\lVert \alpha p(\cdot\lvert y_{<t}, x) + (1-\alpha) q(\cdot\lvert y_{<t}, x))\right]$

Table 4: Token-level distillation losses for generative LMs.

Domain-Agnostic Datasets. For the **Medical** track, we distill on **GenMedGPT** (Li et al., 2023) (115K), and evaluate on **MMLU** (Hendrycks et al., 2021) 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** (Alignment-Lab-AI, 2024) (9.2K instruction response pairs) and **USTerms** (Chalkidis et al., 2023) (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).

B.2 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 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.

Algorithm 1: Knowledge Probing

Input : Dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^{\lvert\mathcal{D}\rvert}$; student $\mathcal{S}$; teacher $\mathcal{T}$; thresholds k_1, k_2, θ
Output : Probing set $\mathcal{D}_{\text{probe}}$; activated teacher layers $\mathcal{Q}^{\mathcal{T}}$

1 # Step 1: Uncertainty-based probing of the student
2 **for each** $(x^i, y^i) \in \mathcal{D}$ **do**
3 **for each** $y_t^i \in y^i$ **do**
4 $u_t^i \leftarrow 1 - q(y_t^i \mid y_{<t}^i, x^i)$
5 $\mathcal{K}^i \leftarrow \text{TopK}_{k_1}(\{u_t^i\}_t)$
6 $\mathcal{D}_{\text{probe}} \leftarrow \{(x^i, y^i, \mathcal{K}^i)\}_{i=1}^{\lvert\mathcal{D}\rvert}$
7 # Step 2: Neuron-based probing of the teacher
8 **for each** $(x^i, y^i, \mathcal{K}^i) \in \mathcal{D}_{probe}$ **do**
9 **for each** $y_t \in \mathcal{K}^i$ **do**
10 **for each** $w_{l,j} \in \text{FFN}(L_l)$ **do**
11 compute $\text{Attr}(w_{l,j})$ via integrated gradients
12 **if** $\text{Attr}(w_{l,j}) > \theta$ **then**
13 mark $w_{l,j}$ as activated
14 $N_l \leftarrow \sum_j \mathbb{I}[\text{Attr}(w_{l,j}) > \theta]$
15 $\mathcal{Q}^{\mathcal{T}} \leftarrow \text{TopK}_{k_2}(\{N_l\}_{l=1}^{L_1})$

B.3 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** (Hinton et al., 2015), **RKD** (Gu et al., 2024), and **SeqKD** (Kim and Rush, 2016). (b) Six LLM-designed methods: **ImitKD** (Lin et al., 2020), **GKD** (Agarwal et al., 2024), **MiniLLM** (Gu et al., 2024), **Distillm** (Ko et al., 2024), **Distillm-2** (Ko et al., 2025), and **ABKD** (Wang et al., 2025). The details are listed as follows:

- **KD** (Hinton et al., 2015): 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** (Wang et al., 2022): 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** (Kim and Rush, 2016): Sequence-level knowledge distillation (SeqKD) trains the student model to imitate full output se-

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	–	–	5
Probing Steps s	–	–	20
Activation Threshold θ	–	–	0.2

Table 5: Training and distillation hyperparameters for the teacher model, student model, and our KPD framework.

quences generated by the teacher, rather than matching token-level distributions, enabling simpler decoding strategies like greedy search while maintaining strong performance.

- **ImitKD** (Lin et al., 2020): 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** (Agarwal et al., 2024): GKD addresses the distribution mismatch in sequence-level distillation by training the student on its own generated outputs, using teacher feedback as supervision. It also supports flexible loss designs and naturally integrates with reinforcement learning fine-tuning.
- **MiniLLM** (Gu et al., 2024): 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** (Ko et al., 2024): 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** (Ko et al., 2025): 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** (Wang et al., 2025): 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.

B.4 Details of Experimental Parameters

We implement our code with Pytorch and apply our framework for all baselines with 8*A100 resources. Table 5 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 5. 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

```
{
    "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 10: An instruction-response pair.

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 by (Dai et al., 2022; Ju et al., 2024), we set the 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 study the impact of their values in Analysis Experiments.

B.5 Details of the Probing Dataset

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 10. We then construct a probing dataset that queries the student model with truncated prompts to isolate each uncertain token, as shown in Figure 11. 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 cost of probing with the full `databricks-dolly-15k` dataset, we construct


```
{

  "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 11: An example of probing datasets.

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.

C Additional Experimental Results

C.1 Main Results of TinyLlama-1.1B

The main results of TinyLlama-1.1B with Rouge-L scores on domain-agnostic datasets are shown in Table 6. The main results of TinyLlama-1.1B with accuracy scores on domain-specific datasets are shown in Table 7.

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**

Table 6: Main results of TinyLlama-1.1B with Rouge-L scores on domain-agnostic 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**

Table 7: Main results of TinyLlama-1.1B with accuracy scores on domain-specific datasets.

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

C.2 Ablation Results of Qwen2.5-1.5B

The ablation results of Qwen2.5-1.5B are shown in Table 8.

C.3 Analysis Results of Number of Probing Tokens k_1

The complete analysis of the number of probing tokens k_1 is shown in Figure 12 in KD, SeqKD, GKD, and Distillm.

C.4 Analysis Results of Number of Probing Layers k_2

The complete analysis of the number of probing layers k_2 is shown in Figure 13 in KD, SeqKD, GKD, and Distillm.

C.5 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 14, 15, and 16, 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 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.

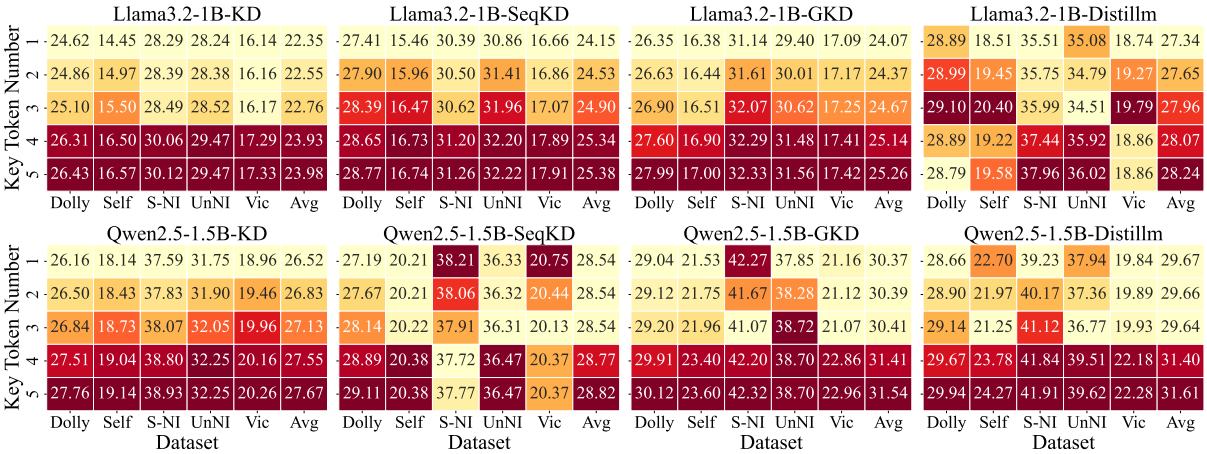

Figure 12: Complete results of probing token number k_1 from 1 to 5. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

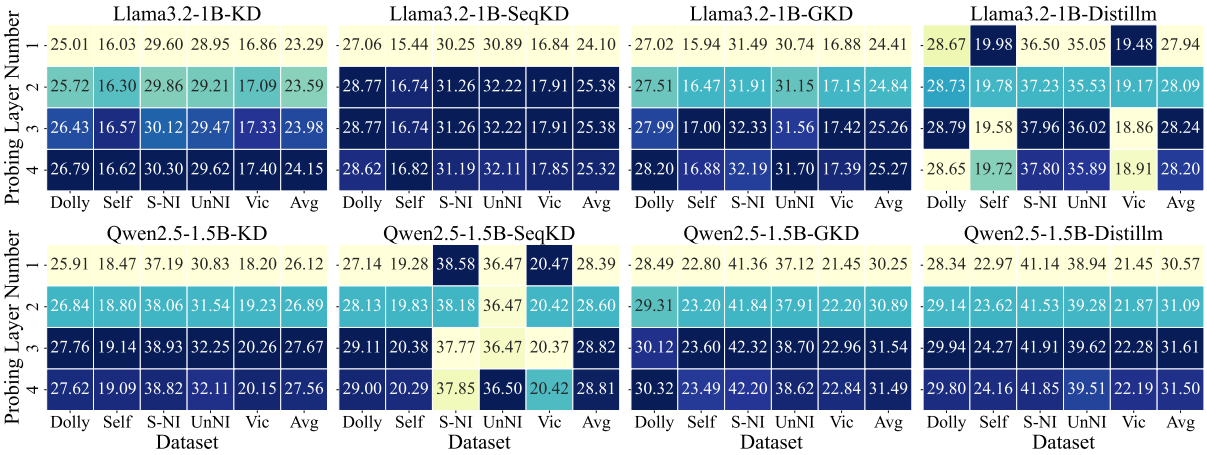

Figure 13: Results of probing layer number k_2 from 1 to 4. The darker each column, the better the performance. Self and Vic denote SelfInst and Vicuna, respectively.

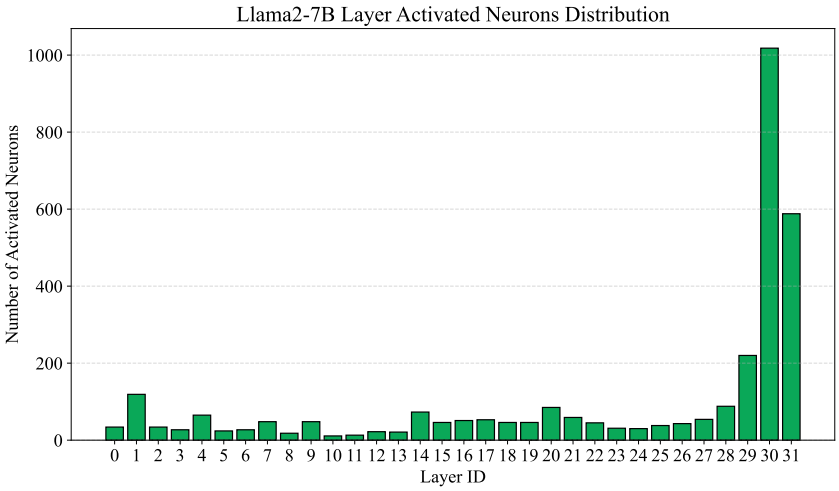

Figure 14: Llama2-7B layer activated neurons distribution.

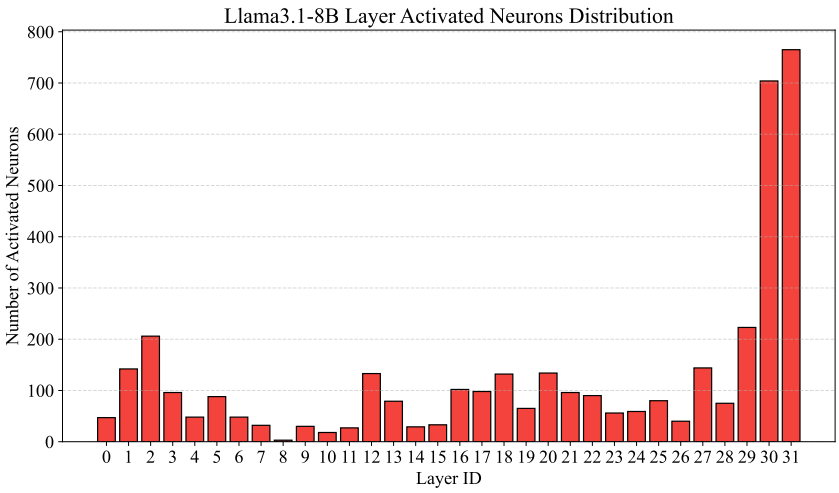

Figure 15: Llama3.1-8B layer activated neurons distribution.

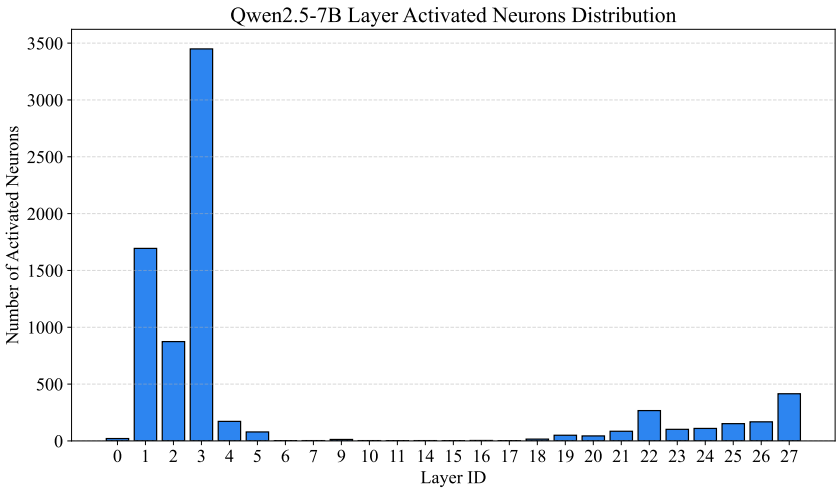

Figure 16: Qwen2.5-7B layer activated neurons distribution.

C.6 Case Study

We sample two set of predictions, which are shown in Tables 9 and 10. 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 (Honovich et al., 2023). 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 9), 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 10), 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 9: 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 10: 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.