# A LITTLE HELP GOES A LONG WAY: EFFICIENT LLM TRAINING BY LEVERAGING SMALL LMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

A primary challenge in large language model (LLM) development is their onerous pre-training cost. Typically, such pre-training involves optimizing a self-supervised objective (such as next-token prediction) over a large corpus. This paper explores a promising paradigm to improve LLM pre-training efficiency *and* quality by suitably leveraging a *small* language model (SLM). In particular, this paradigm relies on an SLM to both (1) provide soft labels as additional training supervision, and (2) select a small subset of valuable ("informative" and "hard") training examples. Put together, this enables an effective transfer of the SLM's predictive distribution to the LLM, while prioritizing specific regions of the training data distribution. Empirically, this leads to reduced LLM training time compared to standard training, while improving the overall quality. Theoretically, we develop a statistical framework to systematically study the utility of SLMs in enabling efficient training of high-quality LLMs. In particular, our framework characterizes how the SLM's seemingly low-quality supervision can enhance the training of a much more capable LLM. Furthermore, it also highlights the need for an *adaptive* utilization of such supervision, by striking a balance between the bias and variance introduced by the SLM-provided soft labels. We corroborate our theoretical framework by improving the pre-training of an LLM with 2.8B parameters by utilizing a smaller LM with 1.5B parameters on the Pile dataset.

## 1 INTRODUCTION

Owing to the recent surge in their ever-growing capabilities, large language models (LLMs) (Chowdhery et al., 2022; Touvron et al., 2023; OpenAI, 2023; Anil et al., 2023; Jiang et al., 2023; Gemini-Team et al., 2023; Anthropic, 2024), have become the focal point of machine learning research. Several research efforts focus on either further enhancing LLM performance, or utilizing LLMs in novel applications ranging from conversational agents/assistants (Thoppilan et al., 2022) to novel material design (Rubungo et al., 2023). Highly capable general-purpose LLMs rely on two critical ingredients: choosing a model architecture with a very large number of parameters (Chowdhery et al., 2022; Smith et al., 2022), and *pre-training* this model on a corpus with a very large number of examples (AI@Meta, 2024; Computer, 2023). Due to the large size of model and corpus, the computational cost of pre-training can be highly onerous. Thus, sustainable advancement and widespread adoption of LLMs hinges on designing novel architectures and algorithms that can reduce the overall training (particularly, pre-training) compute cost and improve the data efficiency for LLM development.

This paper focuses on leveraging *small language models* (*SLMs*) for efficient LLM pre-training. Interestingly, a growing literature (see, e.g., Gupta et al., 2024; Chen et al., 2023; Yue et al., 2024) shows that, despite their limited model capacity, SLMs can acquire a good domain understanding of pre-training data distribution. Particularly, SLMs can perform well on a large portion of "*easy*" instances, while still providing valuable information towards identifying the remaining "*hard*" instances, e.g., via the confidence, margin, or similar measures based on their predictive distribution. This prompts us to explore the following question:

*Can we speed up pre-training of a high-quality* large *LM by transferring the predictive distribution resulting from pre-training of a lower-quality* small *LM?*

Note that suitable SLMs are often readily available during LLM development as previous-generation models trained on similar pre-training corpora or smaller models trained for initial exploration around architectural and algorithmic choices on the current pre-training corpora itself. Furthermore, if proven, the potential of SLMs to enhance LLM quality and efficiency, coupled with their relatively cheaper development cost, strongly justifies training such models even to solely aid LLM training.

*Knowledge distillation* (KD; Buciluǎ et al., 2006; Hinton et al., 2015) is a natural candidate to achieve our underlying objective by utilizing the SLM as *teacher* model to transfer its predictive distribution to *student* LLM during pre-training. However, it is unclear if KD can be helpful in realizing our goal as unlike a typical KD setup – wherein a larger or stronger teacher is used to train a smaller or weaker student – we are hoping to leverage a smaller and *weaker teacher* LM to improve the pre-training efficiency and quality of a larger and *stronger student* LM.

We begin by developing a statistical framework to study KD in the context of language modeling. We obtain novel risk bounds that delineate the desirable properties of the teacher LM-provided supervision that can enhance the student LM's performance, even when one employs a perceivably weaker SLM as the teacher. To the best of our knowledge, ours are the first such bounds for language modeling. Notably, our bounds subsume standard pre-training as a special case, and control LM generalization as we scale model capacity and the amount of pre-training data, with latter being measured in terms of either number of training sequences or number of training tokens. We believe that these bounds are of independent interest to broader community working on LLMs.

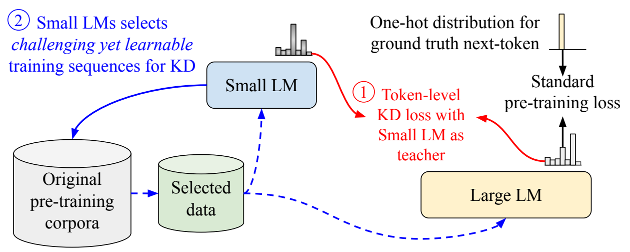


Figure 1: An overview of the proposed SALT. SALT utilizes an SLM in two ways to improve the pre-training of LLM: ① To perform KD with SLM as teacher in the early phase of LLM pre-training; and ② To obtain a valuable subset of pre-training corpora to be utilized during the KD.

Our statistical analysis lays the foundation for an adaptive pre-training method that leverages SLM via KD only in the so-called "easy" regions where SLM can approximate the ground truth next-token distribution well. Combining this with the tendency of neural network to learn easier examples first (Kalimeris et al., 2019; Refinetti et al., 2023), we propose <u>s</u>mall model <u>a</u>ided <u>l</u>arge model <u>t</u>raining (SALT) – a two-stage pre-training approach that employs KD from an SLM in the early phase of the LLM pre-training and resorts to standard self-supervision-based training for the rest of the pre-training. We then expand the SALT method by employing SLM to additionally perform *data selection* for the KD phase. Our selection procedure focuses on identifying *challenging* yet *learnable* sequences from the easy region of the data distribution to ensure an effective transfer of SLM's predictive distribution during KD (see Fig. 1 for an overview). Our empirical study, focusing on both few-shot and post-supervised fine-tuning (SFT) performance, validates the utility of SALT for improving both quality and training efficiency of LLMs compared to standard pre-training. Our key contributions are summarized as follows:

(i) We present a statistical framework for KD in the context of language modeling, which provides novel risk bounds describing how even a perceivably weaker teacher LM can improve the quality of a larger student LM (cf. Sec. 3).

(ii) Guided by our framework, we propose a two-stage pre-training method, namely SALT, that uses SLMs as teacher to perform KD during the first stage corresponding to early phase of LLM pre-training. We extend SALT by utilizing SLMs to perform data selection for the KD phase, facilitating an effective transfer of predictive distribution from SLM to LLM (cf. Sec. 4).

(iii) We showcase the utility of SALT (with and without data selection) by training a 2.8B parameter LM with the help of 1.5B parameter LM on the Pile dataset (Gao et al., 2020). The 2.8B LM trained with SALT outperforms a 2.8B LM trained via standard pre-training method on a wide range of popular few-shot benchmarks while utilizing less than 0.7X training step budget, resulting in $\sim 28\%$ wall-clock time reduction. Moreover, SALT models consistently demonstrate significant downstream performance gains after SFT on multiple domains (cf. Sec. 5).

## 2 Background

**Language modeling.** Given a large corpus, language modeling aims to train a model that can assign probabilities or likelihood to each sequence $\mathbf{x} \in \mathcal{V}^\star$, where $\mathcal{V}$ denotes the underlying vocabulary with $V = |\mathcal{V}|$ tokens. Assuming that the language model (LM) is parameterized by $\boldsymbol{\theta}$, it assigns the following probability to a $T$-token long input sequence $\mathbf{x} = [x_1, x_2, \ldots, x_T]$:

$$P_{\boldsymbol{\theta}}(\mathbf{x}) = P_{\boldsymbol{\theta}}(x_1)P_{\boldsymbol{\theta}}(x_2|x_1)\cdots P_{\boldsymbol{\theta}}(x_T|x_1,\ldots,x_{T-1}).$$

Transformers (Vaswani et al., 2017) are the most prominent architecture supporting LMs (OpenAI, 2023; Touvron et al., 2023; Gemini-Team et al., 2023), which we briefly discuss in Appendix A.1.

**Standard LM pre-training.** Typically, LM pre-training involves the *next-token prediction* task: given a training sequence $\mathbf{x} = [x_1, x_2, \ldots, x_T]$, for each $t \in [T]$, one maximizes the log-likelihood $\log P_{\boldsymbol{\theta}}(x_t|\mathbf{x}_{\leq t-1})$. This amounts to *minimizing* the *cross-entropy* loss between the per-token LM prediction distribution $P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})$ and the one-hot distribution[1] $\mathbb{1}_{x_t}(\cdot)$ defined by the *ground truth* next-token $x_t$. Thus, the overall loss associated with $\mathbf{x}$ becomes

$$\ell(\mathbf{x};\theta) = 1/T \cdot \sum\nolimits_{t\in[T]} -\log P_{\boldsymbol{\theta}}(x_t|\mathbf{x}_{\leq t-1}) = 1/T \cdot \sum\nolimits_{t\in[T]} \mathsf{CE}\big(\mathbb{1}_{x_t}(\cdot), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big), \quad (1)$$

where, $\mathsf{CE}(P_1, P_2) = -\sum_{v\in\mathcal{V}} P_1(v)\log P_2(v)$ is the cross-entropy between distributions $P_1$ and $P_2$.

**Knowledge distillation for LM.** Going beyond the ground truth next-token based loss in (1), one can utilize the per-token prediction distribution provided by another LM, say the one parameterized by $\boldsymbol{\zeta}$, as additional supervision. Formally, given the context $\mathbf{x}_{\leq t-1}$, one can train the LM parameterized by $\boldsymbol{\theta}$ via aligning its prediction distribution $P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})$ with $P_{\boldsymbol{\zeta}}(\cdot|\mathbf{x}_{\leq t-1})$. KL divergence is a common choice to promote such an alignment, which amounts to minimizing the following cross-entropy loss:

$$\ell(\mathbf{x};\boldsymbol{\zeta}\to\boldsymbol{\theta}) = 1/T \cdot \sum\nolimits_{t\in[T]} \mathsf{CE}\big(P_{\boldsymbol{\zeta}}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big). \quad (2)$$

Training based on the loss in (2) is referred to as the *token-level knowledge distillation* (KD; Kim & Rush, 2016) in the literature, with LMs parameterized by $\boldsymbol{\zeta}$ and $\boldsymbol{\theta}$ termed as the teacher and student LMs, respectively. See Appendix A.2 for a discussion on other related KD for LM variants. *Temperature scaling* of teacher is a common strategy (Zheng & Yang, 2024) where, given a temperature $\rho > 0$, one utilizes $P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}) = P_{\boldsymbol{\zeta}}(\cdot|\mathbf{x}_{\leq t-1})^{\rho}/\sum_{v'\in\mathcal{V}} P_{\boldsymbol{\zeta}}(v'|\mathbf{x}_{\leq t-1})^{\rho}$ during KD, resulting in the loss:

$$\ell(\mathbf{x};\boldsymbol{\zeta}^{\rho}\to\boldsymbol{\theta}) = 1/T \cdot \sum\nolimits_{t\in[T]} \mathsf{CE}\big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big). \quad (3)$$

In practice, one typically utilizes both ground truth next-token as well as teacher's next-token distribution and, for a *distillation loss weight* $\omega \in [0, 1]$, minimizes the following as the loss for $\mathbf{x}$:

$$\ell^{\omega}(\mathbf{x};\theta) \triangleq (1-\omega)\cdot\ell(\mathbf{x};\boldsymbol{\theta}) + \omega\cdot\ell(\mathbf{x};\boldsymbol{\zeta}^{\rho}\to\boldsymbol{\theta}). \quad (4)$$

Note that, for brevity, our notation $\ell^{\omega}(\mathbf{x};\theta)$ omits the dependence on $\zeta$ and $\rho$.

## 3 Theoretical analysis: When can KD help language modeling?

As alluded in the introduction, we hope to leverage KD with an SLM as the teacher to speed-up the pre-training of a high quality LLM. However, due to SLM's relatively limited capacity and inferior quality, it is not immediately clear that such a teacher can benefit the LLM. Motivated by this, we now develop a rigorous statistical framework for KD in the context of language modeling by building on the works of Menon et al. (2021); Dao et al. (2021a); Ren et al. (2022a). Novel risk bounds originating from this framework highlight how a teacher LM – even a perceivably weaker one – can benefit student LLM by striking the right balance in terms of a bias-variance trade-off.

Notably, our analysis allows one to control the generalization gap for the student LM in terms of both number of training sequences $N$ as well as number of total tokens $NT$, with the latter being highly non-trivial due to possibly arbitrary dependence within a training sequence. In this work, we crucially leverage certain natural stability conditions on the underlying distribution and function class to obtain such bounds in terms of $NT$. Next, we setup some necessary notation and then present our risk bounds as functions of $N$ and $NT$ in Sections 3.1 and 3.2, respectively. Sec. 3.3 utilizes our bounds to justify the utility of SLMs for improving LLM model quality via KD.

Let $\mathcal{D}$ be the data distribution that generates $N$ independent training sequences $\mathcal{S}_N = \{\mathbf{x}^{(i)}\}_{i\in[N]} \subset \mathcal{V}^T$, i.e., $\mathbf{x}^{(i)} \sim \mathcal{D},\ \forall\, i \in [N]$.[2] Given $\mathcal{S}_N$ and CE surrogate loss (cf. (1)), the *empirical surrogate risk*, i.e., standard training objective, and its population version for an LM parameterized by $\boldsymbol{\theta}$ are:

$$R_N(\boldsymbol{\theta}) = 1/N \cdot \sum\nolimits_{\mathbf{x}\in\mathcal{S}_N} \ell(\mathbf{x};\boldsymbol{\theta}) \quad \text{and} \quad R(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\big[\ell(\mathbf{x};\boldsymbol{\theta})\big]. \quad (5)$$

[1] For $v \in \mathcal{V}$, we define $\mathbb{1}_x(v) = 1$ if $v = x$ and $\mathbb{1}_x(v) = 0$ if $v \neq x$.

[2] Our analysis can be extended to *varying* length sequences at the cost of increased notational complexity.

On the other hand, the empirical surrogate risk for KD, i.e., the KD training objective, and its population version take the following form (note that we omit dependence on $\boldsymbol{\zeta}$ and $\rho$):

$$R_N^{\omega}(\boldsymbol{\theta}) = 1/N \cdot \sum\nolimits_{\mathbf{x}\in\mathcal{S}_N} \ell^{\omega}(\mathbf{x};\boldsymbol{\theta}) \quad \text{and} \quad R^{\omega}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\big[\ell^{\omega}(\mathbf{x};\boldsymbol{\theta})\big]. \tag{6}$$

### 3.1 Excess surrogate risk bound for LM in terms of number of sequences

Given a potentially *infinite* function class $\Theta$ for student LMs, let $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}^*$ be the minimizers of KD training objective (6) and the population risk in (5), respectively:

$$\hat{\boldsymbol{\theta}} := \arg\min\nolimits_{\boldsymbol{\theta}\in\Theta} R_N^{\omega}(\boldsymbol{\theta}) \quad \text{and} \quad \boldsymbol{\theta}^* = \arg\min\nolimits_{\boldsymbol{\theta}\in\Theta} R(\boldsymbol{\theta}). \tag{7}$$

We want to compare the test performance (population risk) of $\hat{\boldsymbol{\theta}}$ with that of $\boldsymbol{\theta}^*$ – the optimal LM in $\Theta$. Before stating our excess risk bound, we introduce an assumption that we rely on in our analysis.

**Assumption 3.1.** The per-token log-loss with at most $T$-token long sequences for the function class $\Theta$ is bounded by $M$, i.e., $\sup_{\boldsymbol{\theta}\in\Theta;\mathbf{x}\in\mathcal{V}^{\leq T-1}} \max_{v\in\mathcal{V}} |\log P_{\boldsymbol{\theta}}(v|\mathbf{x})| \leq M$.

*Remark* 3.2. Assuming loss to be bounded is a standard practice in the statistical learning literature which can be realized, e.g., by clipping the CE loss by a sufficiently large constant $M$. Recently, Lotfi et al. (2024a) realized such an assumption by adding a small amount of uniform noise to LM predictions in their study on generalization for LMs trained via standard pre-training *without* KD.

We are now ready to present our excess risk bound. Due to the page limit we provide an *informal* statement of our result below; see Appendix B.1 for the complete statement and its proof.

**Theorem 3.3** (Informal)**.** *Let $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}^*$ be as defined in (7). Define $f^{\boldsymbol{\theta}} : \mathcal{V}^T \to [0, M]$ by $f^{\boldsymbol{\theta}}(\mathbf{x}) = \ell^{\omega}(\mathbf{x};\boldsymbol{\theta}),\ \forall \mathbf{x}\in\mathcal{V}^T, \boldsymbol{\theta}\in\Theta$. Then, under Assumption 3.1, with probability at least $1-\delta$, we have*

$$R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*) \leq \frac{c_1}{\sqrt{N}} \cdot \left(\sqrt{V_N(f^{\hat{\boldsymbol{\theta}}})\log\left(2\mathcal{M}(N)/\delta\right)} + \sqrt{V_N(f^{\boldsymbol{\theta}^*})\log\left(4/\delta\right)}\right)$$
$$+ (4M\omega)/T \cdot \sum\nolimits_{t\in[T]} \mathbb{E}\left[\mathsf{D}_{\mathrm{TV}}\Big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\Big)\right] + c_2 M/(N-1)\cdot\log(\mathcal{M}(N)/\delta)$$

*where $\mathsf{D}_{\mathrm{TV}}$ is TV distance, $V_N(f^{\boldsymbol{\theta}}) = \frac{1}{N(N-1)}\sum_{1\leq i<j\leq N}\left(f^{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - f^{\boldsymbol{\theta}}(\mathbf{x}^{(j)})\right)^2$ is sample variance, $\mathcal{M}(N)$ depends on the growth function of $\{f^{\boldsymbol{\theta}} : \boldsymbol{\theta}\in\Theta\}$, and $c_1$ & $c_2$ are universal constants.*

### 3.2 Excess surrogate risk bound for LM in terms of number of tokens

For an LM $\boldsymbol{\theta}\in\Theta$ and a training sequence $\mathbf{x} = [x_1, x_2, \ldots, x_T] \in \mathcal{V}^T$, define

$$\xi_t(\mathbf{x};\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}\left[\ell^{\omega}(\mathbf{z};\boldsymbol{\theta})|\mathbf{z}_{\leq t-1} = \mathbf{x}_{\leq t-1}\right] - \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}\left[\ell^{\omega}(\mathbf{z};\boldsymbol{\theta})|\mathbf{z}_{\leq t} = \mathbf{x}_{\leq t}\right], \quad t\in[T]. \tag{8}$$

Note that $\xi_t(\mathbf{x};\boldsymbol{\theta})$ does not depend on $\mathbf{x}_{>t}$. For $t\in[T]$, $\xi_t(\mathbf{x};\boldsymbol{\theta})$ measures the expected KD loss deviation for the student when we condition on the context up to $(t-1)$-th vs. $t$-th token, respectively, and sample the remaining tokens from $\mathcal{D}$. In general, the deviation could be large as changing a *single* token in the context can significantly alter LM's distribution. However, a well-behaved LM should be robust to such perturbations. Motivated by this, we introduce the following assumption.

**Assumption 3.4.** Given the data distribution $\mathcal{D}$ and *finite* function class $\Theta$, the following holds for any $\mathbf{x}\in\mathrm{Support}(\mathcal{D})$, $\boldsymbol{\theta}\in\Theta$, and $t\in[T]$:

$$|\xi_t(\mathbf{x};\boldsymbol{\theta})| \leq C_t \leq C; \quad \text{and} \quad \mathbb{E}\left[\xi_t^2(\mathbf{x};\boldsymbol{\theta})|\mathbf{x}_{\leq t-1}\right] \leq V_t. \tag{9}$$

Under Assumption 3.4, we obtain the following result on student's generalization gap.

**Theorem 3.5.** *Let $\Theta$ be a finite function class. Under Assumption 3.4, with probability at least $1-\delta$, the following holds for the student LM $\hat{\boldsymbol{\theta}}\in\Theta$ obtained via KD:*

$$R(\hat{\boldsymbol{\theta}}) \leq R_N^{\omega}(\hat{\boldsymbol{\theta}}) + \sqrt{2\sum\nolimits_t V_t/N\cdot\log\left(|\Theta|/\delta\right)} + 2C/(3N)\cdot\log\left(|\Theta|/\delta\right) +$$
$$(4M\omega)/T\cdot\sum\nolimits_{t\in[T]}\mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}}\mathsf{D}_{\mathrm{TV}}\big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\big). \tag{10}$$

*Remark* 3.6 (Dependence of $C, \{V_t\}$ on $T$). Theorem 3.5 captures a fine-grained dependence on $C$ and $\{V_t\}$ from Assumption 3.4. For a robust LM, when $C$ is $\mathcal{O}(1/T)$ and $\{V_t\}$ are $\mathcal{O}(1/T^2)$ (note the scaling of $\ell^{\omega}$ by $T$), we get the tightest generalization gap decaying with $NT$. For a non-robust LM in the worst case, i.e., $C$ and $\{V_t\}$ scaling as $\mathcal{O}(1)$ and $\mathcal{O}(1/T)$, the generalization gap gracefully falls back to a bound akin to Theorem 3.3 that only decays with $N$.

**Algorithm 1** Small model aided large model training (SALT)

**Input:** Training data $\mathcal{S}_N = \{\mathbf{x}^{(i)}\}_{i\in[N]} \subset \mathcal{V}^T$, gradient-based optimization algorithm $\mathcal{A}$, SLM parameterized by $\boldsymbol{\zeta}$, distillation loss weight $\omega \in [0,1]$, teacher temperature $\rho > 0$, batch size $B$, training step budget $n$, learning rate schedule $\{\eta_j\}_{j\in[n]}$, and $n_{\mathrm{KD}} \leq n$.

**Output:** Pre-trained LLM parameterized by $\hat{\boldsymbol{\theta}} \in \Theta$.

1: Initialize $\boldsymbol{\theta}_0 \in \Theta$.
2: **for** $j = 1, 2, \ldots, n_{\mathrm{KD}}$ **do** // First stage of LLM pre-training via KD
3: Construct a new batch of $B$ training sequences $\mathcal{B}_j = \{\mathbf{x}^{(i)}\}_{i\in[B]} \subset \mathcal{S}_N$.
4: Update $\theta_{j+1}$ with step size $\eta_j$ via one step of $\mathcal{A}$ on $\mathcal{L}^{\mathrm{KD}}(\mathcal{B}_j) = \frac{1}{B}\sum_{\mathbf{x}\in\mathcal{B}_j} \ell^{\omega}(\mathbf{x};\boldsymbol{\theta}_j)$.
5: **end for**
6: **for** $j = n_{\mathrm{KD}}+1, n_{\mathrm{KD}}+2, \ldots, n$ **do** // Second stage: standard training
7: Construct a new batch of $B$ training sequences $\mathcal{B}_j = \{\mathbf{x}^{(i)}\}_{i\in[B]} \subset \mathcal{S}_N$.
8: Update $\theta_{j+1}$ with step size $\eta_j$ via one step of $\mathcal{A}$ on $\mathcal{L}^{\mathrm{Std}}(\mathcal{B}_j) = \frac{1}{B}\sum_{\mathbf{x}\in\mathcal{B}_j} \ell(\mathbf{x};\boldsymbol{\theta}_j)$.
9: **end for**
10: $\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}_n$

*Remark* 3.7. Recently, Lotfi et al. (2024b) obtained generalization bounds for LLMs trained *without* KD in terms of total number of tokens. However, even when we specialize Theorem 3.5 to standard pre-training by setting $\omega$ to 0, our result significantly differs from theirs both in terms of proof technique as well as its implications. Crucially, results in Lotfi et al. (2024b) only holds for the contexts seen during training whereas our bound enable controlling LM's generalization gap on novel contexts generated from the data distribution during the test time. Also, see Remark 3.6.

### 3.3 KD OUT-PERFORMING STANDARD PRE-TRAINING: A BIAS-VARIANCE TRADE-OFF

Empowered by our novel risk bounds, we now provide justification for why KD can outperform standard pre-training. Specifically, based on Theorem 3.5, we clarify when KD might lead to a tighter bound than standard pre-training. Similar conclusions follow from Theorem 3.3 by extending the arguments from Menon et al. (2021) to language modeling. As per Theorem 3.5, three key quantities control the generalization of the student: (1) $\sum_t V_t$ which relates to loss variance; (2) $C$ which relates to extreme loss values; and (3) divergence between the teacher-provided distribution and the ground truth distribution: $\textsc{Div}(\zeta,\omega) = \omega \cdot \sum_t \mathbb{E}\left[\mathsf{D}_{\mathrm{TV}}\left(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\right)\right]$. Under Assumption 3.1, only $\sum_t V_t$ and $\textsc{Div}(\zeta,\omega)$ are crucial in distinguishing KD and standard pre-training.

Since $\textsc{Div}(\zeta, 0) = 0$, one may surmise that standard pre-training (i.e, $\omega = 0$) leads to tighter bound. But as we detail in Appendix C due to the page limit, the variance term becomes smaller as we *increase* $\omega$. Thus, with a careful selection of $\omega$, the *variance reduction* via KD can offset the *bias* $\textsc{Div}(\zeta,\omega)$. In particular, if the teacher closely approximates the ground truth distribution so that the bias $\textsc{Div}(\zeta,\omega)$ is small even for large $\omega$, then the variance reduction via KD becomes prominent, resulting in significantly tighter generalization gap compared to standard pre-training.

**Performance gain via KD from an SLM as teacher.** While small teacher LMs – the main interest of this work – also lead to variance reduction, they are typically not powerful enough to model the true distribution over the entire data domain very well. Thus, any effect of variance reduction via KD with such a teacher would be washed away by the large bias $\textsc{Div}(\zeta,\omega)$. This highlights the need for an adaptive form of KD from SLMs. Even SLMs with their limited capacity can approximate the true distribution well on certain regions of the data domain, which we call the 'easy' regions. Thus, one can employ KD from SLMs on the easy regions to benefit from the variance reduction without incurring large bias and guarantee improved student LLM performance on these regions. For the remaining ('hard') regions, where the bias is large enough to overshadow the contributions of variance reduction, one should not perform KD from SLMs and utilize the standard pre-training loss.

## 4 SALT: SMALL MODEL AIDED LARGE MODEL TRAINING

We now operationalize the key takeaway from Sec. 3 by proposing SALT – a simple yet effective two-stage pre-training method. SALT relies on the inherent preference of a model to first focus on easier supervision before fitting more complex supervision during training (Kalimeris et al., 2019; Refinetti et al., 2023) to perform selective KD from a teacher SLM. We then expand SALT to perform *explicit selection* of training sequences where we want to conduct KD from an SLM on. In particular, we identify *challenging* sequences which are *learnable* as per teacher SLM; as a result, performing KD on those sequences result in further variance reduction (Katharopoulos & Fleuret, 2018).

**Two-stage LLM pre-training via SALT.** Inspired by our analysis, we propose a two-stage pre-training methods for LLMs in Algorithm 1. The algorithm employs KD with SLM as a teacher in the first stage which lasts till $n_{\mathrm{KD}}$ training steps, and transitions to standard pre-training *without* KD in the second stage. Note that we are interested in the selective transfer of predictive distribution from teacher SLM to student LLM in those regions where SLM performs well by capturing true distribution. By design, KD aims to align predictive distributions of the teacher and student. On the regions where SLM performs well, we expect it to exhibit reasonably confident predictive distribution that should align with ground truth next-token (Gupta et al., 2024), thereby constituting an easier supervision signal for the LLM. In contrast, on hard instances where SLM's predictive distribution is not confident enough or does not align well with the ground truth next-token, learning will be delayed to the later phase of the training (Kalimeris et al., 2019; Refinetti et al., 2023). Thus, SALT relies on the tendency of neural networks to focus on easier instances early during the training to perform desirable knowledge transfer from SLM in the first stage. Once the student LLM is sufficiently aligned with teacher SLM on easier regions, it starts utilizing its model capacity to further align with SLM on more complex regions where high divergence between SLM and ground truth distribution can become detrimental to the LLM's performance. Switching to standard pre-training based solely on the self-supervision from next-tokens in the second stage prevents this undesirable over-alignment. We empirically verify the above intuition behind SALT in Sec. 5.4.

**$\mathsf{SALT}_{\mathrm{DS}}$: SALT with data selection.** We now endow SALT approach (cf. Algorithm 1) with explicit selection of examples where we want to transfer teacher SLM's predictive distribution on, with SLM itself enabling the data selection. In particular, we want to select the most informative (or challenging) examples among the ones that SLM performs well on. Towards this, given the SLM $\boldsymbol{\zeta}$ and a positive integer $k$, we assign a score $S_{\boldsymbol{\zeta},k}(\mathbf{x})$ to training sequence $\mathbf{x}$, with a higher score indicating a higher likelihood to be included for training. More specifically, we compute the per-token cross-entropy loss of SLM on $\mathbf{x}$ and aggregate (typically using the median) into a sequence-level score. This encourages selecting more challenging examples. However, in the spirit of selecting examples that are still *learnable*, we remove all losses where the ground-truth token is not in top-$k$ outputs of the model before aggregating:

$$S_{\boldsymbol{\zeta},k}(\mathbf{x}) = \mathrm{median}\Big(\Big\{ -\mathbb{1}\{x_i \in \mathrm{argtop}_k(P_{\boldsymbol{\zeta}}(\cdot|\mathbf{x}_{<i}))\} \log P_{\boldsymbol{\zeta}}(x_i|\mathbf{x}_{<i}) : i \in [n]\Big\}\Big), \tag{11}$$

where $\mathrm{argtop}_k(P_{\boldsymbol{\zeta}}(\cdot|\mathbf{x}_{<i}))$ denotes the top-$k$ scoring tokens at position $i$ according to SLM. Given a scored pool of sequences, we select the top-$m$ scoring sequences where $m$ is sufficiently large to complete the first stage of Algorithm 1. The reader may note, that if the SLM has been trained with the same dataset that it is scoring, then the computed per-token loss (and resulting sequence score) may be heavily biased. To circumvent that, we use an "early checkpoint" of the model $\boldsymbol{\zeta}_{n_0}$, which has trained on a small number of examples from the overall training set $n_0 \ll N$. We then sample from the remainder of the training examples using score $S_{\boldsymbol{\zeta}_{n_0},k}(\cdot)$. Although the early model $\boldsymbol{\zeta}_{n_0}$ may be a lower quality model due to training with relatively little data, it is only the relative ordering of examples that is important when computing a score, rather than the absolute score.

# 5 Experiments

We now showcase the potential of leveraging SLMs for improving LLM pre-training, by realizing both better quality and improved training efficiency. We demonstrate the additive utility of two aspects of our proposal: (1) employing SLMs as teacher models during the early phase of LLM pre-training (SALT); and (2) further performing data selection via SLMs during the KD phase ($\mathsf{SALT}_{\mathrm{DS}}$).

Throughout our study, we compare with a natural baseline (denoted Baseline) where the large LM is pre-trained in a standalone manner with a self-supervised objective over a pre-training set. This enables us to fairly compare our proposed approach as we fix the model architecture and the underlying pre-training data in our evaluation. The key takeaways from this section are:

(1) SALT and $\mathsf{SALT}_{\mathrm{DS}}$ attain Baseline performance with less than 70% of training steps, and significantly outperform Baseline with the same number of training steps (cf. Sec. 5.2). Additionally, SALT can leverage improved quality small teacher LM to further improve the performance of the large student LM (cf. Appendix H)

(2) SALT and $\text{SALT}_{\text{DS}}$ pre-trained LMs consistently outperform BASELINE after SFT on arithmetic reasoning, summarization, and natural language inference (NLI) tasks (cf. Sec. 5.3).

(3) Step transition from KD phase to standard training phase in SALT (cf. Algorithm 1) constitutes a good design choice as it outperforms other natural alternatives (cf. Appendix H).

We also compare SALT with RKD (standing for *reverse KD*) where we perform KD with SLM as the teacher *throughout* pre-training. RKD results in sub-par performance even compared to BASELINE as the relatively poor quality of SLM limits LLM performance during the later part of training.

### 5.1 Experimental Setup

**Model architectures and pre-training data.** We work with standard decoder-only Transformer-based LMs. Our small model (SLM) has 1.5B parameters and our large model has 2.8B parameters. We use a SentencePiece tokenizer (Kudo & Richardson, 2018) from Du et al. (2022) with a vocabulary size of $256K$. We pre-train all LMs on the Pile dataset (Gao et al., 2020) for 545 billion tokens via UL2 objective (Tay et al., 2023) with a mixture of causal LM, prefix LM and span corruption tasks. We use a batch size of 2048 and input sequence length of 1280. Based on our hyperparameter search, we set the distillation weight $\omega = 0.667$ and teacher temperature $\rho = 0.25$. Further details on model architecture and pre-training are provided in Appendix E.

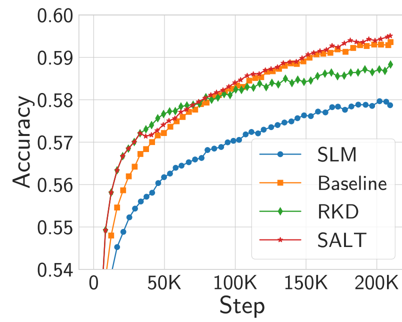


Figure 2: Fraction of correct next-token predictions for various LMs during training, on a subset of the Pile training set.

**Few-shot evaluation tasks.** Following the literature (see, e.g., Anil et al., 2023; Touvron et al., 2023),we perform few-shot evaluation of pre-trained LMs on a wide range of standard benchmarks. We focus on English benchmarks as the pre-training data is mostly English. We defer the full list of benchmarks to Appendix F which can be categorized into: (1) world knowledge, (2) reading comprehension, (3) commonsense reasoning, (4) natural language generation (NLG), and (5) SuperGLUE. We also consider LAMBADA (Paperno et al., 2016) and MBPP (Austin et al., 2021) which are Cloze and code generation tasks, respectively. We conduct 1-shot evaluation for all benchmarks, except for MBPP which is 3-shot. For each benchmark, we report the corresponding prevalent metric in the literature. See Appendix G for details.

**Fine-tuning tasks.** We focused on (arithmetic) reasoning, summarization, and NLI tasks where the few-shot performance of all pre-trained models (including baseline and our models) were poor. For arithmetic reasoning, we utilize GSM8K (Nie et al., 2020). For summarization, we employ XSum (Narayan et al., 2018) and CNN/DailyMail (Nallapati et al., 2016). For NLI tasks, we employ ANLI-R1, ANLI-R2, and ANLI-R3 (Nie et al., 2020).

### 5.2 Results: SALT enables efficient training of high quality LLMs

Interestingly, KD from the seemingly weaker SLM does improve LLM training in the beginning compared to BASELINE, as reflected in the next-token prediction accuracy over the training set in Fig. 2. However, continuing KD from the weaker teacher eventually become detrimental. As evident in Fig. 2 and Tab. 1, RKD significantly underperforms BASELINE on both training and validation set. In contrast, SALT leverages KD from SLM only during first $n_{\text{KD}}$ training steps (cf. Algorithm 1).

**Quality improvements via SALT.** Unlike RKD, SALT (with $n_{\text{KD}} = 36\text{K}$ steps) yields a pre-trained LLM that improves upon BASELINE on both training set (cf. Fig. 2) and held-out validation set (cf. Tab. 1). Tab. 2 presents domain-wise few-shot performance of BASELINE, RKD, SALT, and $\text{SALT}_{\text{DS}}$ (see Tab. 5 Appendix G for per-task performance ). Both SALT and $\text{SALT}_{\text{DS}}$ consistently outperform BASELINE (as well as RKD) at the end of training, i.e., @100% steps or 208K steps. In particular, $\text{SALT}_{\text{DS}}$ outperforms BASELINE in 6 out of 7 domains as well as the overall average; similarly, SALT improves upon BASELINE in 5 out of 7 domains as well as the overall average,establishing the utility of SALT approaches in successfully leveraging SLMs to boost the quality of LLMs.

**Training efficiency realized via SALT.** As per Tab. 2, SALT surpasses BASELINE at 146K steps on average few-shot performance, suggesting a savings of 30% training compute cost. While $n_{\text{KD}} = 36\text{K}$ out of those 146K steps involve KD which is typically computationally costlier than standard training,

Table 2: **Domain-wise few-shot performance of pre-trained LMs.** SALT and $\text{SALT}_{\text{DS}}$ already outperform BASELINE in terms of average few-shot performance at 70% of their training step budget, thereby improving both training efficiency and model quality. RKD (i.e., naively distilling from SLM throughout pre-training) performs much worse than BASELINE. The best and second-best results for each domain are **boldfaced** and <u>underlined</u>, respectively.

| Domain | # Tasks | SLM | BASELINE | RKD | SALT | | $\text{SALT}_{\text{DS}}$ | |
|---|---|---|---|---|---|---|---|---|
| | | | *@100% steps* | *@100% steps* | *@70% steps* | *@100% steps* | *@70% steps* | *@100% steps* |
| **World Knowledge** | 4 | 15.90 | <u>22.19</u> | 18.69 | 21.59 | **22.70** | 20.64 | 21.72 |
| **Reading Comprehension** | 4 | 46.30 | 53.00 | 51.00 | 53.55 | <u>54.55</u> | 54.35 | **54.93** |
| **Commonsense Reasoning** | 7 | 57.76 | 61.99 | 58.30 | 61.27 | 61.67 | <u>62.00</u> | **62.10** |
| **LAMBADA** | 1 | 26.90 | 36.20 | 31.10 | <u>50.70</u> | 48.30 | 48.00 | **53.00** |
| **SuperGLUE** | 8 | 61.59 | 65.53 | 62.91 | **66.30** | 65.28 | <u>65.99</u> | 65.58 |
| **NLG** | 3 | 3.13 | 4.60 | 3.40 | 4.63 | 4.73 | <u>4.80</u> | **4.83** |
| **MBPP** | 1 | 9.60 | 16.20 | 11.40 | 15.60 | <u>17.00</u> | 16.60 | **17.80** |
| **Average** | 28 | 42.56 | 47.32 | 44.39 | 47.86 | <u>47.94</u> | 47.89 | **48.26** |

as we argue next, the fact that our teacher is a SLM ensures that we still realize efficiency gains via SALT. In particular, the additional compute cost in KD is one forward pass of SLM per training step. As a rule of thumb, a forward pass constitutes $1/4$th cost of a training step (which comprises both forward and backward passes). In our setup, a forward pass of SLM is approximately half as expensive as that for the LLM. Thus, KD from SLM adds a factor of $1/8$ to the cost of the training step of the standard training. (Our implementation verifies this as we observe 12.0% slow down during KD compared to standard training.) Since KD lasts for $n_{\text{KD}} = 36$K out 146K training steps, the training cost required by SALT to surpass BASELINE is approximately equivalent to that of $(146 + 36/8)$K steps of the standard training. This translates to an efficiency gain of $\sim 28\%$ compared to the 208K steps taken by BASELINE.

### 5.3 IMPROVED DOWNSTREAM PERFORMANCE REALIZED VIA SALT POST SFT

Tab. 2 already showcases that SALT can leverage SLMs to obtain large pre-trained LMs that out-perform widely adopted standard pre-training (BASELINE). That said, all the pre-trained models (including BASELINE) exhibit relatively poor few-shot performance on certain benchmarks, e.g., NLG or summarization task (in Tab. 2) and also MATH (Hendrycks et al., 2021) and ANLI (Nie et al., 2020) benchmarks. This raises the question of whether SALT is beneficial for downstream application performance.

Table 1: **Accuracy and log-perplexity on a held-out set of Pile.** We evaluate the models at an early and the final checkpoint. At the end of pre-training (208K steps), both SALT and $\text{SALT}_{\text{DS}}$ improve upon BASELINE in terms of both next-token prediction accuracy and log-perplexity. RKD is *identical* to SALT during its first stage, hence they have the same performance at $n_{\text{KD}} = 36$K steps.

| Model | Evaluation stage (steps) | Accuracy (↑) | Log (↓) perplexity |
|---|---|---|---|
| SLM | Final (208K) | 57.70 | 1.951 |
| BASELINE | Early (36K) | 56.68 | 2.011 |
| RKD | | 57.21 | 2.160 |
| SALT | | 57.21 | 2.160 |
| $\text{SALT}_{\text{DS}}$ | | 56.47 | 2.188 |
| BASELINE | Final (208K) | 58.99 | 1.868 |
| RKD | | 58.46 | 2.071 |
| SALT | | <u>59.10</u> | <u>1.863</u> |
| $\text{SALT}_{\text{DS}}$ | | **59.17** | **1.857** |

Supervised fine-tuning (SFT) is a standard approach to convert a pre-trained LM into a domain-specific proficient model. We employ SFT on a range of downstream tasks covering three domains, namely arithmetic reasoning, NLG or summarization, and NLI. For each downstream benchmark, we perform SFT on the pre-trained LMs obtained via BASELINE, SALT, and $\text{SALT}_{\text{DS}}$. During SFT, we train each pre-trained LM for 10K steps with Adafactor algorithm (Shazeer & Stern, 2018). We utilize cosine learning rate schedule (Loshchilov & Hutter, 2017) with a peak learning rate of 1e-4 and linear warm-up phase consisting of 200 steps. These hyperparameters were optimized for the BASELINE; and we did not conduct hyperparameter tuning for the pre-trained LMs resulting from SALT approaches. Finally, we employ greedy decoding during the evaluation of the fine-tuned LMs. Tab. 3 presents SFT performance on six benchmarks covering the aforementioned three downstream domains. SALT consistently outperforms BASELINE on all benchmarks and, in most cases, $\text{SALT}_{\text{DS}}$ performs the best. This showcases that SALT (*with* or *without* data selection) enables significant improvements for

Table 3: **Supervised fine-tuning (SFT) results.** Performance of various pre-trained checkpoints on downstream tasks after SFT. For each benchmark, pre-trained 2.8B models are fine-tuned on the corresponding train split and evaluated on the validation split (test split in case of GSM8K). *Acc*, *Rg1*, *Rg2*, and *RgL* represent the *Accuracy*, *Rouge-1*, *Rouge-2*, and *Rouge-Lsum* metrics, respectively.

| | **GSM8K** | **XSum** | | | **CNN/DailyMail** | | | **ANLI-R1** | **ANLI-R2** | **ANLI-R3** |
|---|---|---|---|---|---|---|---|---|---|---|
| | *Acc* | *Rg1* | *Rg2* | *RgL* | *Rg1* | *Rg2* | *RgL* | *Acc* | *Acc* | *Acc* |
| BASELINE | 31.84 | 43.39 | 21.09 | 35.91 | 42.84 | 20.43 | 40.38 | 63.70 | 56.90 | 57.83 |
| SALT | 34.87 | 43.45 | 21.21 | 36.04 | 43.19 | 20.65 | 40.74 | 67.00 | **57.80** | **59.67** |
| $\text{SALT}_{\text{DS}}$ | **35.25** | **43.77** | **21.44** | **36.24** | **43.41** | **20.87** | **40.95** | **67.30** | 57.70 | 59.58 |

a range of *difficult* downstream domains even when the corresponding pre-trained LMs exhibits only a small performance gains over BASELINE.

### 5.4 SLM ENABLES FAST LEARNING ON EASY EXAMPLES

Table 4: **Few-shot evaluation on different buckets of XLSum-EN.** Each number shows average Rouge-2 scores on the corresponding bucket. We use gray, green, and red to highlight the results similar to, better than, and worse than BASELINE performance, respectively.

| | Evaluation stage (steps) | Easy | Medium | Hard |
|---|---|---|---|---|
| SLM | Final (208K) | 8.04 | 0.43 | 0.00 |
| BASELINE | Early (36K) | 6.15 | 1.61 | 0.71 |
| RKD | Early (36K) | 6.76 | 1.40 | 0.58 |
| SALT | Early (36K) | 6.76 | 1.40 | 0.58 |
| BASELINE | Final (208K) | 8.80 | 2.52 | 0.97 |
| RKD | Final (208K) | 7.87 | 1.68 | 0.74 |
| SALT | Final (208K) | 9.68 | 2.67 | 0.99 |

Recall that SALT aims to realize quality and efficiency gains for LLM pre-training by quickly transferring the predictive distribution of an SLM to the LLM via KD, focusing on the 'easy' regions of the data distribution where the SLM performs well. Subsequently, SALT falls back on ground truth next-token-based supervision to refine LLM's performance on the 'hard' regions where the SLM fares poorly. Here, we set out to empirically demonstrate that this key intuition behind SALT is indeed borne out in practice. Focusing on various few-shot eval benchmarks, we partition instances in each benchmark into 'easy', 'medium', and 'hard' buckets based on the teacher SLM's performance (see Appendix J for details). We then evaluate BASELINE, SALT, and RKD pre-trained LLMs on these individual buckets after $n_{\text{KD}} = 36\text{K}$ training steps when the KD phase of SALT ends as well as at the end of the pre-training, i.e., after 208K steps. Tab. 4 presents these results on the XLSum-EN task (see Appendix J for results on other benchmarks), which validate: (1) KD from SLM quickly enables LLM to perform well on 'easy' instances; and (2) standard pre-training after KD phase ending at $n_{\text{KD}}$-th step helps LLM performance on 'hard' instances the most.

## 6 RELATED WORK

Here, we provide a brief account of related work, focusing on the prior work on assisting large model training via small models, data selection, and theoretical treatment of KD. Due to the page limit, we defer the discussion on various KD methods for language modeling that are not pertinent to the main objective of this work to Appendix A.

**Aiding large model training with small models.** Small models often help identify good hyper-parameters that can be utilized for large model training with minimal modifications (Yang et al., 2021). Progressive or stage-wise training methods (Gong et al., 2019; Gu et al., 2020; Reddi et al., 2023; Yao et al., 2023; Li et al., 2023) train a large model in stages where the parameters for the model at a given stage get initialized based on the parameters of a smaller model from the previous stage. Another related line of work simply informs the large model initialization based on a smaller model without resorting to progressive stage-wise training (Trockman & Kolter, 2023; Wang et al., 2023; Samragh et al., 2024). Most of these works crucially depend on the architectural overlaps between the small and large models, e.g., requiring them to share same depth, width, or more generally same model family. In contrast, one can distill from a smaller model to a larger model without such constraints. Furlanello et al. (2018) study self-distillation to iteratively train a model, with the final model from an iteration acting as a teacher for the next iteration. More closer to the setting studied in this work, *albeit* in image classification setting, Yuan et al. (2020); Xie et al. (2020) consider distillation from a weaker model. In the work that is closest to our proposal, Qin et al. (2022) distill an LM from a smaller LM. Moreover, they also distill during the early phase of larger student LM pre-training to avoid negative impact on large LM's final performance. We demonstrate the utility of such an approach with larger LMs as well as larger pre-training corpus, with a wider range of evaluation benchmarks. Furthermore, we provide a statistical framework to rigorously justifies the utility of a seemingly weaker teacher during LLM pre-training. Other recent efforts on using smaller LMs to boost larger LMs have primarily focused on fine-tuning (Yang et al., 2024; Mitchell et al., 2024) or alignment (Burns et al., 2023), while we focus on pre-training which is the most compute and data intensive phase of LLM development.

**Data Selection.** Ankner et al. (2024) select examples with reference model sequence-level log-perplexities in a specified range. They aggregate per-token log-perplexities by taking the mean, whereas, we take the median and also zero out noisy tokens. Mindermann et al. (2022) select examples and Lin et al. (2024) select tokens for training based on *excess* training loss over a reference model. While we perform *offline* data selection before training, these two works select data from training batches on the fly. Methods which encourage diversity through semantic embeddings (e.g., Abbas et al. (2023); Tirumala et al. (2024)) can yield a better final model when employed in $\mathsf{SALT}_{\mathrm{DS}}$. Please refer to Albalak et al. (2024) for a comprehensive survey of data selection techniques for LMs.

**Theoretical understanding of knowledge distillation.** Focusing on (deep) linear networks, Phuong & Lampert (2019) study the generalization bounds for KD and relate its success to various factors including data geometry and optimization bias. Menon et al. (2021); Ren et al. (2022b) show that KD leads to reduced variance of the training objective, thereby improving generalization, while also relating the effectiveness of KD to teacher's ability to approximate Bayes class probabilities. Cotter et al. (2021) argue that KD can improve generalization as long as teacher approximates a suitable transformation of Bayes class probabilities. Dao et al. (2021b) study KD from the perspective of semiparameteric inference to analyze the excess risk of distilled student. Focusing on self-distillation for kernel ridge regression, Mobahi et al. (2020) show that distillation can enhance regularization. Allen-Zhu & Li (2023) explain the utility of distillation via better feature learning. Xu et al. (2020) study the interplay between optimization and label smoothing via teacher-provided soft labels. Focusing on linearized neural networks, Harutyunyan et al. (2023) attribute the success of KD to the reduced supervision complexity. More recently, Safaryan et al. (2023) argue that KD can act as a form of partial variance reduction, thereby improving convergence. We would like to highlight that the existing literature *does not* provide generalization bounds for KD in a *sequence* learning setting such as language modeling, and we provide the first statistical treatment of KD for language modeling. Notably, similar to our work, Xu et al. (2020); Nagarajan et al. (2023) also explore the utility of KD only during an early-phase of student training, albeit not in a language modeling setting.

## 7 Conclusion

We conduct a systematic study of the utility of SLMs to improve both training efficiency and performance of LLM pre-training. Towards this, we introduce a novel statistical framework for KD in the context of language modeling, which guides the design of SALT – a simple KD-based approach that selectively transfers predictive distribution from an SLM to an LLM. We further enhance SALT and perform explicit data selection via SLMs to effectively transfer knowledge from SLMs to LLMs. SALT significantly reduces the pre-training time for LLMs while ensuring good overall quality as measured by the LLM's few-shot performance as well as downstream performance after fine-tuning.

While SALT can play a crucial role in efficiently sustaining the trend of developing LLMs with increasing capabilities, it also has the potential to help institutions train proficient lightweight LMs. Conventionally, one distills a large powerful model into a lightweight model with good quality. However, given that many LLMs are proprietary, such strong-to-weak distillation is not feasible at many institutions. Our proposed approach can leverage even *smaller* LMs to enhance the quality of a small LM with acceptable inference cost. Exploring if our proposed approach can introduce new capabilities in a general-purpose small LM with the help of one or multiple *smaller* LMs that are experts in their respective domains is an interesting avenue for future work. Interestingly, our data selection approach in $\mathsf{SALT}_{\mathrm{DS}}$ demonstrates that it is indeed possible to leverage data selection to improve KD for language modeling. Building on this initial investigation and further exploring and extending data selection approaches tailored to $\mathsf{SALT}_{\mathrm{DS}}$ is another interesting line of future work.

## REFERENCES

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. SemDeDup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arxiv:2303.09540*, 2023.

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

AI@Meta. Llama 3 Model Card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. A Survey on Data Selection for Language Models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=XfHWcNTSHp.

Zeyuan Allen-Zhu and Yuanzhi Li. Towards Understanding Ensemble, Knowledge Distillation and Self-Distillation in Deep Learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Uuf2q9TfXGA.

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.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L. Leavitt, and Mansheej Paul. Perplexed by perplexity: Perplexity-based data pruning with small reference models, 2024. URL https://arxiv.org/abs/2405.20541.

AI Anthropic. The Claude 3 Model Family: Opus, Sonnet, Haiku. *Claude-3 Model Card*, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program Synthesis with Large Language Models. *arXiv preprint arXiv:2108.07732*, 2021.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, 2013.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs. 2018. URL http://github.com/google/jax.

Cristian Bucilă, Rich Caruana, and Alexandru Niculescu-Mizil. Model Compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pp. 535–541, New York, NY, USA, 2006. ACM.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *arXiv preprint arXiv:2305.05176*, 2023.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling Language Modeling with Pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A Benchmark for Information-Seeking Question Answering in Typologically Diverse Languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv:1803.05457v1*, 2018.

Together Computer. RedPajama: an Open Dataset for Training Large Language Models. October 2023. URL https://github.com/togethercomputer/RedPajama-Data.

Andrew Cotter, Aditya Krishna Menon, Harikrishna Narasimhan, Ankit Singh Rawat, Sashank J Reddi, and Yichen Zhou. Distilling Double Descent. *arXiv preprint arXiv:2102.06849*, 2021.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pp. 177–190. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-33428-6.

Tri Dao, Govinda M Kamath, Vasilis Syrgkanis, and Lester Mackey. Knowledge distillation as semiparametric inference. In *International Conference on Learning Representations*, 2021a.

Tri Dao, Govinda M Kamath, Vasilis Syrgkanis, and Lester Mackey. Knowledge Distillation as Semiparametric Inference. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=m4UCf24r0Y.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. The CommitmentBank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2): 107–124, Jul. 2019. doi: 10.18148/sub/2019.v23i2.601. URL https://ojs.ub.uni-konstanz.de/sub/index.php/sub/article/view/601.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient Scaling of Language Models with Mixture-of-Experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5547–5569. PMLR, 17–23 Jul 2022.

Xiequan Fan, Ion Grama, and Quansheng Liu. Hoeffding's inequality for supermartingales. *Stochastic Processes and their Applications*, 122(10):3545–3559, 2012.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing Smaller Language Models towards Multi-Step Reasoning. *arXiv preprint arXiv:2301.12726*, 2023.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born-Again Neural Networks. In *International conference on machine learning*, pp. 1607–1616. PMLR, 2018.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, Thomas Kollar, Yair Carmon, Achal Dave, Reinhard Heckel, Niklas Muennighoff, and Ludwig Schmidt. Language models scale reliably with over-training and on downstream tasks, 2024.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gemini-Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*, 2023.

Linyuan Gong, Di He, Zhuohan Li, Tao Qin, Liwei Wang, and Tieyan Liu. Efficient Training of BERT by Progressively Stacking. In *International Conference on Machine Learning*, pp. 2337–2346. PMLR, 2019.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. SemEval-2012 task 7: Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pp. 394–398, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL https://aclanthology.org/S12-1052.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Xiaotao Gu, Liyuan Liu, Hongkun Yu, Jing Li, Chen Chen, and Jiawei Han. On the Transformer Growth for Progressive BERT Training. *arXiv:2010.12562*, 2020.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge Distillation of Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Neha Gupta, Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Aditya Krishna Menon, and Sanjiv Kumar. Language Model Cascades: Token-Level Uncertainty And Beyond. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=KgaBScZ4VI.

Hrayr Harutyunyan, Ankit Singh Rawat, Aditya Krishna Menon, Seungyeon Kim, and Sanjiv Kumar. Supervision complexity and its role in knowledge distillation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8jU7wy7N7mA.

Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 4693–4703, 2021.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. *NeurIPS*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NeurIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for Natural Language Understanding. *arXiv preprint arXiv:1909.10351*, 2019.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. Sgd on neural networks learns functions of increasing complexity. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/b432f34c5a997c8e7c806a895ecc5e25-Paper.pdf.

Angelos Katharopoulos and Francois Fleuret. Not all samples are created equal: Deep learning with importance sampling. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2525–2534. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/katharopoulos18a.html.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 252–262, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1023. URL https://aclanthology.org/N18-1023.

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

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

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen Mckeown. WikiLingua: A New Benchmark Dataset for Cross-Lingual Abstractive Summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4034–4048, 2020.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794. Association for Computational Linguistics, September 2017.

Gyeongbok Lee, Seung-won Hwang, and Hyunsouk Cho. SQuAD2-CR: Semi-supervised Annotation for Cause and Rationales for Unanswerability in SQuAD 2.0. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 5425–5432, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.667.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, 01 2019. doi: 10.18653/v1/P19-1612.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The Winograd Schema Challenge. In *13th International Conference on the Principles of Knowledge Representation and Reasoning, KR 2012*, Proceedings of the International Conference on Knowledge Representation and Reasoning, pp. 552–561. Institute of Electrical and Electronics Engineers Inc., 2012. ISBN 9781577355601.

Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Xuying Meng, Siqi Fan, Peng Han, Jing Li, Li Du, Bowen Qin, et al. Flm-101b: An open llm and how to train it with $100 k budget. *arXiv preprint arXiv:2309.03852*, 2023.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Rho-1: Not All Tokens Are What You Need. *arXiv preprint arXiv:2404.07965*, 2024.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

Sanae Lotfi, Marc Anton Finzi, Yilun Kuang, Tim G. J. Rudner, Micah Goldblum, and Andrew Gordon Wilson. Non-Vacuous Generalization Bounds for Large Language Models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32801–32818. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/lotfi24a.html.

Sanae Lotfi, Yilun Kuang, Brandon Amos, Micah Goldblum, Marc Finzi, and Andrew Gordon Wilson. Unlocking Tokens as Data Points for Generalization Bounds on Larger Language Models. *arXiv preprint arXiv:2407.18158*, 2024b.

Andreas Maurer and Massimiliano Pontil. Empirical Bernstein Bounds and Sample Variance Penalization. *arXiv preprint arXiv:0907.3740*, 2009.

Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A Statistical Perspective on Distillation. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7632–7642. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/menon21a.html.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *EMNLP*, 2018.

Sören Mindermann, Jan Brauner, Muhammed Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. Prioritized Training on Points that are Learnable, Worth Learning, and Not Yet Learnt. In *International Conference on Machine Learning (ICML)*, 2022.

Eric Mitchell, Rafael Rafailov, Archit Sharma, Chelsea Finn, and Christopher D Manning. An Emulator for Fine-tuning Large Language Models using Small Language Models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Eo7kv0sllr.

Hossein Mobahi, Mehrdad Farajtabar, and Peter Bartlett. Self-Distillation Amplifies Regularization in Hilbert Space. *Advances in Neural Information Processing Systems*, 33:3351–3361, 2020.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1098. URL https://aclanthology.org/N16-1098.

Vaishnavh Nagarajan, Aditya K Menon, Srinadh Bhojanapalli, Hossein Mobahi, and Sanjiv Kumar. On student-teacher deviations in distillation: does it pay to disobey? *Advances in Neural Information Processing Systems*, 36:5961–6000, 2023.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In Stefan Riezler and Yoav Goldberg (eds.), *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/K16-1028. URL https://aclanthology.org/K16-1028.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807. Association for Computational Linguistics, October-November 2018. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial NLI: A New Benchmark for Natural Language Understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.441. URL https://aclanthology.org/2020.acl-main.441.

OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction Tuning with GPT-4. *arXiv preprint arXiv:2304.03277*, 2023.

Mary Phuong and Christoph Lampert. Towards Understanding Knowledge Distillation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5142–5151. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/phuong19a.html.

Mohammad Taher Pilehvar and José Camacho-Collados. WiC: 10,000 Example Pairs for Evaluating Context-Sensitive Representations. *arXiv*, abs/1808.09121, 2018.

Bernardo Ávila Pires and Csaba Szepesvári. Multiclass Classification Calibration Functions. *arXiv preprint arXiv:1609.06385*, 2016.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. In Mirella Lapata, Phil Blunsom, and Alexander Koller (eds.), *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 157–163, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-2025.

Yujia Qin, Yankai Lin, Jing Yi, Jiajie Zhang, Xu Han, Zhengyan Zhang, Yusheng Su, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Knowledge Inheritance for Pre-trained Language Models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, July 2022.

Sashank J. Reddi, Sobhan Miryoosefi, Stefani Karp, Shankar Krishnan, Satyen Kale, Seungyeon Kim, and Sanjiv Kumar. Efficient Training of Language Models using Few-Shot Learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14553–14568. PMLR, 2023.

Maria Refinetti, Alessandro Ingrosso, and Sebastian Goldt. Neural networks trained with SGD learn distributions of increasing complexity. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28843–28863. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/refinetti23a.html.

Yi Ren, Shangmin Guo, and Danica J. Sutherland. Better supervisory signals by observing learning paths. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022a.

Yi Ren, Shangmin Guo, and Danica J. Sutherland. Better Supervisory Signals by Observing Learning Paths. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=IogodjAdbHj.

Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. Scaling Up Models and Data with `t5x` and `seqio`. *arXiv:2203.17189*, 2022. URL https://arxiv.org/abs/2203.17189.

S. M. Ross. *Stochastic Processes*. Wiley Series in Probability and Mathematical Statistics, 1983.

Andre Niyongabo Rubungo, Craig Arnold, Barry P Rand, and Adji Bousso Dieng. LLM-Prop: Predicting Physical And Electronic Properties Of Crystalline Solids From Their Text Descriptions. *arXiv preprint arXiv:2310.14029*, 2023.

Mher Safaryan, Alexandra Peste, and Dan Alistarh. Knowledge Distillation Performs Partial Variance Reduction. In *Advances in Neural Information Processing Systems*, volume 36, pp. 75229–75258. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ee1f0da706829d7f198eac0edaacc338-Paper-Conference.pdf.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *AAAI*, pp. 8732–8740. AAAI Press, 2020.

Mohammad Samragh, Iman Mirzadeh, Keivan Alizadeh Vahid, Fartash Faghri, Minsik Cho, Moin Nabi, Devang Naik, and Mehrdad Farajtabar. Scaling Smart: Accelerating Large Language Model Pre-training with Small Model Initialization, 2024.

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.

Noam Shazeer. Fast Transformer Decoding: One Write-Head is All You Need. *arXiv preprint arXiv:1911.02150*, 2019. URL http://arxiv.org/abs/1911.02150.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using Deep-Speed and Megatron to Train Megatron-Turing NLG 530B, A Large-Scale Generative Language Model. *arXiv preprint arXiv:2201.11990*, 2022.

Ingo Steinwart. How to Compare Different Loss Functions and Their Risks. *Constructive Approximation*, 26(2):225–287, 2007.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient Knowledge Distillation for BERT Model Compression. *arXiv preprint arXiv:1908.09355*, 2019.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. UL2: Unifying Language Learning Paradigms. *arXiv preprintarXiv:2205.05131*, 2023.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. LaMDA: Language Models for Dialog Applications. *arXiv preprint arXiv:2201.08239*, 2022.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. D4: Improving LLM pretraining via document de-duplication and diversification. *Advances in Neural Information Processing Systems*, 36, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023.

Asher Trockman and J Zico Kolter. Mimetic Initialization of Self-Attention Layers. In *International Conference on Machine Learning*, pp. 34456–34468. PMLR, 2023.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. *arXiv preprint arXiv:1908.08962*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *Advances in Neural Information Processing Systems*, 30, 2017.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. SuperGLUE: A Stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

Yite Wang, Jiahao Su, Hanlin Lu, Cong Xie, Tianyi Liu, Jianbo Yuan, Haibin Lin, Ruoyu Sun, and Hongxia Yang. LEMON: Lossless model expansion. *arXiv preprint arXiv:2310.07999*, 2023.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. f-Divergence Minimization for Sequence-Level Knowledge Distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10817–10834, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.605. URL https://aclanthology.org/2023.acl-long.605.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with Noisy Student improves ImageNet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10687–10698, 2020.

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.

Yi Xu, Yuanhong Xu, Qi Qian, Hao Li, and Rong Jin. Towards Understanding Label Smoothing. *arXiv preprint arXiv:2006.11653*, 2020.

Ge Yang, Edward Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer. *Advances in Neural Information Processing Systems*, 34:17084–17097, 2021.

Yu Yang, Siddhartha Mishra, Jeffrey N Chiang, and Baharan Mirzasoleiman. SmallToLarge (S2L): Scalable Data Selection for Fine-tuning Large Language Models by Summarizing Training Trajectories of Small Models. *arXiv preprint arXiv:2403.07384*, 2024.

Yiqun Yao, Zheng Zhang, Jing Li, and Yequan Wang. Masked structural growth for 2x faster language model pre-training. *arXiv preprint arXiv:2305.02869*, 2023.

Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting Knowledge Distillation via Label Smoothing Regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3903–3911, 2020.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large Language Model Cascades with Mixture of Thought Representations for Cost-Efficient Reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6okaSfANzh.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, July 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension. *CoRR*, abs/1810.12885, 2018.

Tong Zhang. Statistical Analysis of Some Multi-Category Large Margin Classification Methods. *Journal of Machine Learning Research*, 5(Oct):1225–1251, 2004.

Kaixiang Zheng and En-Hui Yang. Knowledge distillation based on transformed teacher matching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MJ3K7uDGGl.

# A Little Help Goes a Long Way: Efficient LLM Training by Leveraging Small LMs

## Appendix

## A Language modeling and knowledge distillation

### A.1 Language modeling via Transformer-based models

Here, we briefly discuss how Transformers are typically employed for language modeling in modern systems. Given a context $\mathbf{x}_{\leq t} = [x_1, \ldots, x_t] \in \mathcal{V}^t$, a Transformer-based LM first produces a sequence of $d$-dimensional token embeddings $E(\mathbf{x}_{\leq t}) = [\mathrm{e}_{x_1}^\top, \mathrm{e}_{x_2}^\top, \ldots, \mathrm{e}_{x_t}^\top] \in \mathbb{R}^{d\times t}$, where $\mathbf{e}_v \in \mathbb{R}^d$ denotes the token embedding for $v \in \mathcal{V}$. A Transformer network $f_{\boldsymbol{\psi}}$ then processes $E(\mathbf{x}_{\leq t})$ to produce a target embedding $f_{\boldsymbol{\psi}}\big(E(\mathbf{x}_{\leq t})\big) \in \mathbb{R}^d$, which is multiplied by $W \in \mathbb{R}^{V\times d}$, namely a classification layer, to obtain a logit vector $u_{\mathbf{x}_{\leq t}} := W f_{\boldsymbol{\psi}}\big(E(\mathbf{x}_{\leq t})\big) \in \mathbb{R}^V$. Accordingly, we have $\boldsymbol{\theta} = \{E, \boldsymbol{\psi}, W\}$ as the parameters of the LM. Applying softmax operation on the logit vector produces the probability that the LM assigns to each token in $\mathcal{V}$ as the possible continuation (also known as next token) to the context $\mathbf{x}_{\leq t}$:

$$P_{\boldsymbol{\theta}}(v|\mathbf{x}_{\leq t}) = \frac{\exp(u_{\mathbf{x}_{\leq t}}(v)/\tau)}{\sum_{v'\in\mathcal{V}} \exp(u_{\mathbf{x}_{\leq t}}(v')/\tau)}, \qquad \forall v \in \mathcal{V}. \tag{12}$$

Here, $\tau$ denotes the (inverse) temperature associated with the softmax operation. Unless stated otherwise, we assume that $\tau = 1$.

### A.2 Other common variants of knowledge distillation for LM

**Top-$k$ token-level KD.** Instead of aligning the teacher and student's full per-token prediction distributions, one could only match these distribution on $\mathcal{T} \subset \mathcal{V}$, e.g., $k \ll V$ elements of $\mathcal{V}$ that receive the highest scores from the teacher:

$$\ell(\mathbf{x}; \boldsymbol{\zeta} \to \boldsymbol{\theta}) = -\sum_{t=1}^{T} \Big( \sum_{v\in\mathcal{T}} P_{\boldsymbol{\zeta}}^{\mathcal{T}}(v|\mathbf{x}_{\leq t-1}) \cdot \log P_{\boldsymbol{\theta}}^{\mathcal{T}}(v|\mathbf{x}_{\leq t-1}) \Big), \tag{13}$$

where $P^{\mathcal{T}}$ denotes the restriction of $P$ (defined over $\mathcal{V}$) to $\mathcal{T}$:

$$P^{\mathcal{T}}(v) = \begin{cases} \frac{P(v)}{\sum_{v'\in\mathcal{T}} P(v')} & \text{if } v \in \mathcal{T}, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

**Sequence-level KD.** Unlike token-level KD, sequence-level KD aims to align teacher and student's distributions on all sequences up to sequence length $T$. In particular, the sequence-level KD loss takes the form:

$$\ell^{\mathrm{seq}}(\mathbf{x}; \boldsymbol{\zeta} \to \boldsymbol{\theta}) = -\sum_{\tilde{\mathbf{x}}\in\mathcal{V}^{\leq n}} P_{\boldsymbol{\zeta}}(\tilde{\mathbf{x}}) \cdot \log P_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}) \tag{15}$$

In practice, it's natural to focus on a subset of all candidate target sequences $\mathcal{U} \subset \mathcal{V}^{\leq T}$:

$$\ell^{\mathrm{seq}}(\mathbf{x}; \boldsymbol{\zeta} \to \boldsymbol{\theta}) = -\sum_{\tilde{\mathbf{x}}\in\mathcal{U}} P_{\boldsymbol{\zeta}}(\tilde{\mathbf{x}}) \cdot \log P_{\boldsymbol{\theta}}(\tilde{\mathbf{x}})$$

A common choice for $\mathcal{U}$ is the set of say $k$ most likely sequences under the teacher's distribution $P_{\boldsymbol{\zeta}}$.

### A.3 Recent literature on knowledge distillation (KD) for language modeling

A large body of literature focuses on utilizing KD (Buciluǎ et al., 2006; Hinton et al., 2015) as a core technique to improve LMs (Kim & Rush, 2016; Gou et al., 2021; Xu et al., 2024). For instance, Sanh et al. (2019); Turc et al. (2019); Wang et al. (2020); Sun et al. (2019); Jiao et al. (2019) relied on KD to compress BERT-style LMs during pre-training, fine-tuning, or both. More recently, KD has been primarily employed in the instruction-tuning or fine-tuning phase where a general purpose LM is adapted to a specific collection of tasks (Xu et al., 2024). Black-box KD methods for LM only assume access to training sequences sampled from a teacher LM (Taori et al., 2023; Fu et al., 2023; Peng et al., 2023). With access to token-level distributions from teacher LM, *token-level distillation* from teacher LM to student LM is possible (Kim & Rush, 2016). In contrast, *sequence-level distillation* involves sampling training sequences from the teacher LM, the student LM, or both before aligning teacher and student's predictive distribution on such sequences (Kim & Rush, 2016; Agarwal et al., 2024; Gu et al., 2024; Wen et al., 2023).

## B Deferred proofs from Section 3

### B.1 Proof of Theorem 3.3

Before stating the formal version of Theorem 3.3 and its proof, let us recall the necessary notation. Given a function class for student LMs $\Theta$, $\hat{\boldsymbol{\theta}}$ denotes the LM obtained by minimizing the training objective for KD in (6), i.e.,

$$\hat{\boldsymbol{\theta}} := \arg\min_{\boldsymbol{\theta}\in\Theta} R^{\omega}_{N}(\boldsymbol{\theta}). \tag{16}$$

Further, $\boldsymbol{\theta}^*$ represents the optimal or best performing LM in $\Theta$, i.e.,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Theta} R(\boldsymbol{\theta}). \tag{17}$$

Finally, recall our assumption regarding the bounded loss values.

**Assumption B.1.** Given a function class $\Theta$ for (student) LM, the per-token log-loss with at most $T$-token long sequences for the underlying function class $\Theta$ is bounded by $M$, i.e.,

$$\sup_{\boldsymbol{\theta}\in\Theta;\mathbf{x}\in\mathcal{V}^{\leq T-1}} \max_{v\in\mathcal{V}} |\log P_{\boldsymbol{\theta}}(v|\mathbf{x})| \leq M. \tag{18}$$

Towards establishing Theorem 3.3, we first state the following intermediate result.

**Proposition B.2.** *Let $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}^*$ be as defined in (16) and (17), respectively. Then, under Assumption B.1, the excess surrogate risk for $\hat{\boldsymbol{\theta}}$ satisfies the following.*

$$\begin{aligned} R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*) \leq \frac{4M\omega}{T} \cdot \sum_{t=1}^{T} \mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}} \mathsf{D}_{\rm TV}\Big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\Big) \\ + \Big(R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}_{N}(\hat{\boldsymbol{\theta}})\Big) + \left(R^{\omega}_{N}(\boldsymbol{\theta}^*) - R^{\omega}(\boldsymbol{\theta}^*)\right), \end{aligned} \tag{19}$$

*where $\mathsf{D}_{\rm TV}(\cdot,\cdot)$ denotes the total-variation distance between two probability distributions.*

*Proof of Proposition B.2.* For convenience, recall that

$$\begin{aligned} R^{\omega}(\boldsymbol{\theta}) &= \mathbb{E}_{\mathbf{x}}\Big[(1-\omega)\cdot\ell(\mathbf{x};\boldsymbol{\theta}) + \omega\cdot\ell(\mathbf{x};\boldsymbol{\zeta}^{\rho}\to\boldsymbol{\theta})\Big] \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\frac{1}{T}\sum_{t=1}^{T}\mathsf{CE}\big(P^{(x_t,\omega)}_{\zeta,\rho}, P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big)\right] \\ &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}}\Big[\mathsf{CE}\big(\underbrace{(1-\omega)\cdot\mathcal{D}(\cdot|\mathbf{x}_{\leq t-1}) + \omega\cdot P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1})}_{P^{(\mathcal{D},\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1})}, P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big)\Big] \\ &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}}\Big[\mathsf{CE}\big(P^{(\mathcal{D},\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big)\Big]. \end{aligned}$$

Note that we have

$$
\begin{aligned}
&R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*) \\
&\overset{(i)}{=} R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*) - \Big(R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*)\Big) + \Big(R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*)\Big) \\
&\overset{(ii)}{=} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}_{\mathbf{x}_{1:t}\sim\mathcal{D}}\Big[\sum_{v\in\mathcal{V}} \Big(P^{(\mathcal{D},\omega)}_{\zeta,\rho}(v|\mathbf{x}_{1:t}) - \mathcal{D}(v|\mathbf{x}_{1:t})\Big)\cdot\Big(\log P_{\hat{\boldsymbol{\theta}}}(v|\mathbf{x}_{1:t}) - \log P_{\boldsymbol{\theta}^*}(v|\mathbf{x}_{1:t})\Big)\Big] + \\
&\qquad R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*) \\
&\overset{(iii)}{\leq} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}_{\mathbf{x}_{1:t}\sim\mathcal{D}}\Big[\Big\|P^{(\mathcal{D},\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{1:t}) - \mathcal{D}(\cdot|\mathbf{x}_{1:t})\Big\|_1 \cdot \Big\|\log P_{\hat{\boldsymbol{\theta}}}(\cdot|\mathbf{x}_{1:t}) - \log P_{\boldsymbol{\theta}^*}(\cdot|\mathbf{x}_{1:t})\Big\|_\infty\Big] + \\
&\qquad R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*) \\
&\overset{(iv)}{\leq} \frac{4M\omega}{T}\cdot\sum_{t=0}^{T-1} \mathbb{E}_{\mathbf{x}_{1:t}\sim\mathcal{D}}\Big[\mathsf{D}_{\rm TV}\Big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{1:t}), \mathcal{D}(\cdot|\mathbf{x}_{1:t})\Big)\Big] + \underbrace{R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*)}_{\rm (I)},
\end{aligned}
\tag{20}
$$

where $(i)$ follows from adding and subtracting $R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*)$; $(ii)$ employs the definition of $R(\hat{\boldsymbol{\theta}})$, $R(\boldsymbol{\theta}^*)$, $R^{\omega}(\hat{\boldsymbol{\theta}})$, and $R^{\omega}(\boldsymbol{\theta}^*)$; $(iii)$ invokes Hölder's inequality; and $(iv)$ follows from the definition of total-variation distance $\mathsf{D}_{\rm TV}(\cdot,\cdot)$ and the fact that underlying per-token loss terms are bounded by $M$.

Next, we focus on the term (I) in (20):

$$
\begin{aligned}
R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}(\boldsymbol{\theta}^*) &\overset{(i)}{=} R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\hat{\boldsymbol{\theta}}) + R^{\omega}_N(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\boldsymbol{\theta}^*) + R^{\omega}_N(\boldsymbol{\theta}^*) - R^{\omega}(\boldsymbol{\theta}^*) \\
&= \Big(R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\hat{\boldsymbol{\theta}})\Big) + \Big(R^{\omega}_N(\boldsymbol{\theta}^*) - R^{\omega}(\boldsymbol{\theta}^*)\Big) + R^{\omega}_N(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\boldsymbol{\theta}^*) \\
&\overset{(ii)}{\leq} \Big(R^{\omega}(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\hat{\boldsymbol{\theta}})\Big) + \Big(R^{\omega}_N(\boldsymbol{\theta}^*) - R^{\omega}(\boldsymbol{\theta}^*)\Big)
\end{aligned}
\tag{21}
$$

where $(i)$ follows by adding and subtracting $R^{\omega}_N(\hat{\boldsymbol{\theta}})$ and $R^{\omega}_N(\boldsymbol{\theta}^*)$; and $(ii)$ holds as $\hat{\boldsymbol{\theta}}$ is the minimizer of $R^{\omega}_N(\cdot)$ in $\Theta$ which implies that $R^{\omega}_N(\hat{\boldsymbol{\theta}}) - R^{\omega}_N(\boldsymbol{\theta}^*) \leq 0$. Now, the statement in Proposition B.2 follows by combining (20) and (21). □

Note that the bound on excess surrogate risk in Proposition B.2 decomposes into three terms:

- First term captures the *divergence* between the ground truth per-token distribution and the teacher-induced per-token distribution leveraged during KD; and
- The last two terms corresponds to the deviation between empirical and population surrogate risks for the empirical risk minimizer $\hat{\boldsymbol{\theta}}$ and population risk minimzer $\boldsymbol{\theta}^*$ within the function class $\Theta$. Note that since, $\hat{\boldsymbol{\theta}}$ is a a random variable in itself (which depends on the training sample $\mathcal{S}_N$), one typically needs to bound the deviation uniformly over all functions $\boldsymbol{\theta}\in\Theta$. As we will see next, one can bound these deviations in terms of the properties of both model class $\Theta$ as well as the teacher-induced per-token distributions.

In order to make the excess surrogate risk bound in Proposition B.2 explicit, we need to bound the third term via a computable quantity. We apply sample variance-based bounds from Maurer & Pontil (2009) to get the following result.

**Theorem B.3** (Formal version of Theorem 3.3)**.** *Suppose Assumption B.1 holds. Let $\mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}$ be a function class that maps elements in $\mathcal{V}^T$ to $[0, M]$ as defined below:*

$$
\mathcal{F}^{\omega} := \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega} \triangleq \left\{\mathbf{x}\mapsto\frac{1}{T}\sum_{t=1}^{T}\mathsf{CE}\big(P^{(\mathcal{D},\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big),\ \forall\mathbf{x}\in\mathcal{V}^T, \boldsymbol{\theta}\in\Theta\right\}. \tag{22}
$$

*For $\epsilon > 0$, let $\mathcal{N}_\infty(\epsilon, \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}, N)$ denote the growth function for the function class $\mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}$, i.e.,*

$$
\mathcal{N}_\infty(\epsilon, \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}, N) \triangleq \sup_{\mathbf{X}=(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(N)})\in\mathcal{V}^{T\times N}} \mathcal{N}(\epsilon, \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}(\mathbf{X}), \|\cdot\|_\infty), \tag{23}
$$

*where* $\mathcal{N}(\epsilon, \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}(\mathbf{X}), \|\cdot\|_\infty)$ *denotes the smallest* $\epsilon$*-cover of the set*

$$\mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}(\mathbf{X}) = \left\{\left(f(\mathbf{x}^{(1)}), f(\mathbf{x}^{(2)}), \ldots, f(\mathbf{x}^{(N)})\right) : f \in \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}\right\} \subseteq \mathbb{R}^N$$

*with respect to* $\|\cdot\|_\infty$ *norm. Then, with probability at least* $1-\delta$*, for all* $\boldsymbol{\theta} \in \Theta$*, we have*

$$\begin{aligned} R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*) &\leq \frac{4M\omega}{T} \cdot \sum_{t=1}^{T} \mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}} \mathsf{D}_{\mathrm{TV}}\left(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\right) \\ &+ \sqrt{\frac{18V_N(f^{\hat{\boldsymbol{\theta}}}, \mathcal{S}_N)\log\left(\frac{2\mathcal{M}(N)}{\delta}\right)}{N}} + \frac{15M\log\left(\frac{2\mathcal{M}(N)}{\delta}\right)}{N-1} \\ &+ \sqrt{\frac{2V_N(f^{\boldsymbol{\theta}^*}, \mathcal{S}_N)\log\left(\frac{4}{\delta}\right)}{N}} + \frac{7M\log\left(\frac{4}{\delta}\right)}{3(N-1)}, \end{aligned} \tag{24}$$

*where* $\mathcal{M}(N) \triangleq 10 \cdot \mathcal{N}_\infty(1/N, \mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}, 2N)$*;* $f^{\boldsymbol{\theta}}$ *denotes the function in* $\mathcal{F}^{\boldsymbol{\zeta},\rho,\omega}$ *that corresponds to* $\boldsymbol{\theta}$*, as per (22); and* $V_N(f^{\boldsymbol{\theta}}, \mathcal{S}_N)$ *denotes the sample variance*

$$V_N(f^{\boldsymbol{\theta}}, \mathcal{S}_N) = \frac{1}{N(N-1)} \sum_{1\leq i<j\leq N} \left(f^{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - f^{\boldsymbol{\theta}}(\mathbf{x}^{(j)})\right)^2. \tag{25}$$

*Proof of Theorem B.3.* As discussed earlier, in light of Proposition B.2, we only need to bound two terms $R^\omega(\hat{\boldsymbol{\theta}}) - R^\omega_N(\hat{\boldsymbol{\theta}})$ and $R^\omega_N(\boldsymbol{\theta}^*) - R^\omega(\boldsymbol{\theta}^*)$ to obtain the desired result. Now utilizing Theorem 6 and Theorem 4 (with $\delta$ replaced with $\delta/2$) in Maurer & Pontil (2009) to bound the two terms, respectively, completes the proof of Theorem B.3. □

### B.2 Token-level generalization bound

Before providing a proof of Theorem 3.5, we first introduce some intermediate results that are needed to prove the theorem. Recall that our training sample $\mathcal{S}_N = \{\mathbf{x}^{(i)} = [x_1^{(i)}, \ldots, x_T^{(i)}]\}_{i\in[N]}$ comprises $N$ independent sequences such that $\mathbf{x}^{(i)} \sim \mathcal{D}, \forall i \in [N]$. With $\ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta})$ representing the KD loss on $i$-th sequence, we define the random variables

$$\begin{aligned} Z_0^{(i)} &= \mathbb{E}[\ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta})], \\ Z_t^{(i)} &= \mathbb{E}\left[\ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta}) \mid \mathbf{x}^{(i)}_{\leq t}\right], \text{ for } 1 \leq t \leq T, \end{aligned} \tag{26}$$

where $Z_T^{(i)} = \mathbb{E}\left[\ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta}) \mid \mathbf{x}^{(i)}\right] = \ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta})$. Note that $\{Z_t^{(i)}\}_{0\leq t\leq T}$ is a *Doob martingale sequence* with respect to the natural filtration $\{\mathcal{F}_t^{(i)}\}_{0\leq t\leq T}$ of the random variables $\{x_1^{(i)}, \ldots, x_t^{(i)}\}$ (Ross, 1983, pg 297). Accordingly, we define a *martingale difference sequence* $\{\xi_t^{(i)}, \mathcal{F}_t^{(i)}\}_{t\in[T]}$ such that for $t \in [T]$,

$$\xi_t^{(i)} := \xi_t(\mathbf{x}^{(i)}; \boldsymbol{\theta}) = Z_{t-1}^{(i)} - Z_t^{(i)} = \mathbb{E}\left[\ell^\omega(\mathbf{x}; \boldsymbol{\theta})|\mathbf{x}^{(i)}_{\leq t-1}\right] - \mathbb{E}\left[\ell^\omega(\mathbf{x}; \boldsymbol{\theta})|\mathbf{x}^{(i)}_{\leq t}\right]. \tag{27}$$

As per Assumption 3.4, the following holds for each $t \in [T]$:

$$|\xi_t^{(i)}(\mathbf{x}; \boldsymbol{\theta})| \leq C_t \leq C, \tag{28}$$

$$\mathbb{E}\left[\left(\xi_t^{(i)}\right)^2 |\mathbf{x}_{\leq t-1}\right] \leq V_t. \tag{29}$$

We are ready to state the first intermediate result which bounds the moment generating function for the following random variable associated with the KD loss on the $i$-th training sequence:

$$Z_0^{(i)} - Z_T^{(i)} = \mathbb{E}\left[\ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta})\right] - \ell^\omega(\mathbf{x}^{(i)}; \boldsymbol{\theta}).$$

**Lemma B.4.** *Under Assumption 3.4, the following holds for each $i \in [N]$:*

$$\mathbb{E}\left[e^{\lambda\cdot(Z_0^{(i)}-Z_T^{(i)})/C}\right] \le \exp\left(T\cdot f\left(\lambda, \frac{1}{T}\sum_{t=1}^{T}\frac{V_t}{C^2}\right)\right), \tag{30}$$

*where, for $\lambda \ge 0$ and $s \ge 0$,*

$$f(\lambda, s) \triangleq \log\left(\frac{1}{1+s}\cdot\exp(-\lambda s) + \frac{s}{1+s}\cdot\exp(\lambda)\right). \tag{31}$$

*Proof.* Note that

$$\begin{aligned}
\mathbb{E}\left[e^{\lambda\cdot(Z_0^{(i)}-Z_T^{(i)})/C}\right] &= \mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T}\xi_t^{(i)}/C}\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T}\xi_t^{(i)}/C}|\mathbf{x}_{\le T-1}^{(i)}\right]\right] \\
&\overset{(i)}{=} \mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T-1}\xi_t^{(i)}/C}\cdot\mathbb{E}\left[e^{\lambda\cdot\xi_T^{(i)}/C}|\mathbf{x}_{\le T-1}^{(i)}\right]\right] \\
&\overset{(ii)}{\le} \mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T-1}\xi_t^{(i)}/C}\cdot e^{f\left(\lambda,\ \frac{1}{C^2}\cdot\mathbb{E}\left[\left(\xi_T^{(i)}\right)^2|\mathbf{x}_{\le T-1}^{(i)}\right]\right)}\right] \\
&\overset{(iii)}{\le} \mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T-1}\xi_t^{(i)}/C}\cdot e^{f(\lambda,\ \frac{V_T}{C^2})}\right] \\
&= \mathbb{E}\left[e^{\lambda\cdot\sum_{t=1}^{T-1}\xi_t^{(i)}/C}\right]\cdot e^{f(\lambda,\ \frac{V_T}{C^2})}
\end{aligned} \tag{32}$$

where $(i)$ follows as $e^{\lambda\cdot\sum_{t=1}^{T-1}\xi_t^{(i)}}$ is $\mathcal{F}_{T-1}^{(i)}$-measurable; $(ii)$ follows from (Fan et al., 2012, Lemma 3.1); and $(iii)$ follows from Assumption 3.4 and the fact that, for $\lambda > 0$ and $s \ge 0$, $f(\lambda, s)$ is an increasing function in its second argument (Fan et al., 2012, Lemma 3.2). By following the similar steps in (32) for $\xi_{i,T-1}, \xi_{i,T-2}, \ldots, \xi_{i,1}$, we obtain that

$$\mathbb{E}\left[e^{\lambda\cdot(Z_0^{(i)}-Z_T^{(i)})/C}\right] \le e^{\sum_{t=1}^{T}f(\lambda,\frac{V_t}{C^2})}. \tag{33}$$

According to (Fan et al., 2012, Lemma 3.2) that, for $\lambda \ge 0$ and $s \ge 0$, $f(\lambda, s)$ is a concave function in its second argument. Thus, it follows from Jensen's inquality that

$$\frac{1}{T}\sum_{t=1}^{T}f\left(\lambda, \frac{V_t}{C^2}\right) \le f\left(\lambda, \frac{1}{T}\sum_{t=1}^{T}\frac{V_t}{C^2}\right). \tag{34}$$

By combining (33) and (34), we have

$$\mathbb{E}\left[e^{\lambda\cdot(Z_0^{(i)}-Z_T^{(i)})/C}\right] \le e^{T\cdot f\left(\lambda, \frac{1}{T}\sum_{t=1}^{T}\frac{V_t}{C^2}\right)}, \tag{35}$$

which completes the proof. ◻

Now we can leverage Lemma B.4 to obtain the following concentration inequality for the KD training objective.

**Lemma B.5.** *Let $\boldsymbol{\zeta}$ and $\boldsymbol{\theta} \in \Theta$ denote the teacher and student LM, respectively. Then, for $\epsilon > 0$, the following holds under Assumption 3.4.*

$$\mathbb{P}\left(\sum_{i=1}^{N}\left(\mathbb{E}\left[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right] - \ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right)/C \ge N\epsilon\right) \le \exp\left(-\frac{N\epsilon^2}{2(\sum_t\frac{V_t}{C^2}+\frac{1}{3}\epsilon)}\right). \tag{36}$$

*Proof.* Recall that, as per our notation, we have

$$\mathbb{E}\left[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right] - \ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta}) = Z_0^{(i)} - Z_T^{(i)}.$$

Thus,

$$\mathbb{P}\left(\sum_{i=1}^{N}\left(\mathbb{E}\big[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\big]-\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right)/C \geq N\epsilon\right)=\mathbb{P}\left(\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C\geq N\epsilon\right). \quad (37)$$

It follows from Markov's inequality that, for $\lambda \geq 0$,

$$\begin{aligned}\mathbb{P}\left(\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C\geq N\epsilon\right) &= \mathbb{P}\left(e^{\lambda\cdot\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C}\geq e^{N\lambda\epsilon}\right)\\ &\leq \frac{\mathbb{E}\left[e^{\lambda\cdot\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C}\right]}{e^{N\lambda\epsilon}}\\ &\overset{(i)}{=} \frac{\prod_{i\in[N]}\mathbb{E}\left[e^{\lambda\cdot\big(Z_0^{(i)}-Z_T^{(i)}\big)/C}\right]}{e^{N\lambda\epsilon}},\end{aligned} \quad (38)$$

where $(i)$ follows as $\{Z_0^{(i)}-Z_T^{(i)}\}_{i\in[N]}$ are independent random variables. By combining (38) with Lemma B.4, we obtain that

$$\mathbb{P}\left(\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C\geq N\epsilon\right)\leq e^{-N\cdot\left(\lambda\epsilon-T\cdot f(\lambda,\frac{1}{T}\sum_{t=1}^{T}\frac{V_t}{C^2})\right)}. \quad (39)$$

Since (39) holds for each $\lambda\geq 0$, we have

$$\mathbb{P}\left(\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C\geq N\epsilon\right)\leq \inf_{\lambda\geq 0} e^{-N\cdot\left(\lambda\epsilon-T\cdot f(\lambda,\frac{1}{T}\sum_{t=1}^{T}\frac{V_t}{C^2})\right)} \quad (40)$$

Now as argued in the Proof of Remark 2.1 in Fan et al. (2012), for $0\leq\lambda<3$, $s\geq 0$, we have

$$f(\lambda,s)\leq(e^{\lambda}-1-\lambda)s\leq\frac{\lambda^2 s}{2(1-\frac{1}{3}\lambda)}. \quad (41)$$

Thus, it follows from (40) that

$$\begin{aligned}\mathbb{P}\left(\sum_{i=1}^{N}\big(Z_0^{(i)}-Z_T^{(i)}\big)/C\geq N\epsilon\right) &\leq \inf_{0\leq\lambda<3}\exp\left(-N\cdot\left(\lambda\epsilon-\frac{\lambda^2}{2(1-\frac{1}{3}\lambda)}\cdot\sum_t\frac{V_t}{C^2}\right)\right)\\ &\leq \exp\left(-\frac{N\epsilon^2}{2(\sum_t\frac{V_t}{C^2}+\frac{1}{3}\epsilon)}\right).\end{aligned} \quad (42)$$

This completes the proof. □

Equipped with Lemma B.5, we are now ready to prove Theorem 3.5 below.

*Proof of Theorem 3.5.* Note that

$$R(\hat{\boldsymbol{\theta}})-R_N^{\omega}(\hat{\boldsymbol{\theta}})=\underbrace{R(\hat{\boldsymbol{\theta}})-R^{\omega}(\hat{\boldsymbol{\theta}})}_{\text{(I)}}+\underbrace{R^{\omega}(\hat{\boldsymbol{\theta}})-R_N^{\omega}(\hat{\boldsymbol{\theta}})}_{\text{(II)}}. \quad (43)$$

Following the similar analysis used in the proof of Theorem 3.3, we can bound the term (I) to obtain the following.

$$R(\hat{\boldsymbol{\theta}})-R^{\omega}(\hat{\boldsymbol{\theta}})\leq\frac{4M\omega}{T}\cdot\sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}}\mathsf{D}_{\mathrm{TV}}\Big(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}),\mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\Big). \quad (44)$$

Next, we focus on bounding the term (II). As per notation, for any $\boldsymbol{\theta}\in\Theta$, we have

$$R^{\omega}(\boldsymbol{\theta})-R_N^{\omega}(\boldsymbol{\theta})=\frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}\big[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\big]-\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\Big). \quad (45)$$

Thus, for a fixed $\boldsymbol{\theta} \in \Theta$, we have

$$\begin{aligned}
\mathbb{P}\left(R(\boldsymbol{\theta}) - R_N^{\omega}(\boldsymbol{\theta}) \geq \gamma\right) &= \mathbb{P}\left(\frac{1}{N}\sum_{i=1}^{N}\left(\mathbb{E}\big[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\big] - \ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right) \geq \gamma\right) \\
&= \mathbb{P}\left(\sum_{i=1}^{N}\left(\mathbb{E}\big[\ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\big] - \ell^{\omega}(\mathbf{x}^{(i)};\boldsymbol{\theta})\right)/C \geq N\cdot\frac{\gamma}{C}\right) \\
&\overset{(i)}{\leq} \exp\left(-\frac{N\gamma^2}{2(\sum_t V_t + \frac{1}{3}C\gamma)}\right), \qquad (46)
\end{aligned}$$

where $(i)$ follows from (36) with $\epsilon = \frac{\gamma}{C}$. With some algebra, one can see that the right hand side of (46) is bounded by $\delta/|\Theta|$ when

$$\gamma \geq \frac{2C}{3N}\cdot\log\left(|\Theta|/\delta\right) + \sqrt{\frac{2}{N}\cdot\sum_t V_t\cdot\log\left(|\Theta|/\delta\right)}. \qquad (47)$$

(To see this, set the right hand side of (46) to $\delta/|\Theta|$ to get a quadratic of the form $\gamma^2 = a\gamma + b$ with $a, b \geq 0$ and note that its non-negative root is $\leq a + \sqrt{b}$. All $\gamma \geq a + \sqrt{b}$ will make the right hand side of (46) $\leq \delta/|\Theta|$.)

Now, by taking union bound, with probability at least $1 - \delta$, for all $\boldsymbol{\theta} \in \Theta$, we have the following.

$$R^{\omega}(\boldsymbol{\theta}) - R_N^{\omega}(\boldsymbol{\theta}) \leq \frac{2C}{3N}\cdot\log\left(|\Theta|/\delta\right) + \sqrt{\frac{2}{N}\cdot\sum_t V_t\cdot\log\left(|\Theta|/\delta\right)}. \qquad (48)$$

Since the minimizer of the KD training objective $\hat{\boldsymbol{\theta}}$ is in $\Theta$, with probablity at least $1 - \delta$, we have

$$(\mathrm{II}) = R^{\omega}(\hat{\boldsymbol{\theta}}) - R_N^{\omega}(\hat{\boldsymbol{\theta}}) \leq \frac{2C}{3N}\cdot\log\left(|\Theta|/\delta\right) + \sqrt{\frac{2}{N}\cdot\sum_t V_t\cdot\log\left(|\Theta|/\delta\right)}. \qquad (49)$$

Now, the statement of Theorem 3.5 follows by combining (43), (44), and (49). □

## C KD can improve generalization via variance reduction

Here, we leverage our novel generalization bounds to provide a theoretical justification for why KD can result in better generalization behavior compared to standard pre-training. In particular, we will focus on our bound in Theorem 3.5.[3] Note that, besides $|\Theta|$, $N$, and $T$ which are independent of the underlying training approach, there are three key quantities that dictate the generalization gap: (1) $\sum_t V_t$ which is related to the loss variance; (2) $C$ which is related to the extreme values that loss can take; and (3) the divergence between the teacher-provided next-token predictive distribution and the ground truth next-token distribution:

$$\textsc{Div}(\zeta,\omega) := \omega\cdot\sum_{t=1}^{T}\mathbb{E}\left[\mathsf{D}_{\mathrm{TV}}\left(P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}), \mathcal{D}(\cdot|\mathbf{x}_{\leq t-1})\right)\right].$$

Note that, under Assumption 3.1, both KD and standard pre-training loss terms are bounded by $M$, allowing us to provide the same $C$ (as a function of $M$ and $T$) for *both* KD and standard pre-training. Thus, we focus on the remaining two terms which relate to $\sum_t V_t$ and $\textsc{Div}(\zeta,\omega)$.

Note that standard pre-training, i.e., training without KD, corresponds to $\omega = 0$, which leads to $\textsc{Div}(\zeta,\omega = 0) = 0$. In contrast, with $\omega > 0$, KD would incur a non-zero value for $\textsc{Div}(\zeta,\omega)$. On the other hand, as we will argue next, KD can lead to smaller value of the variance term $\sum_t V_t$. Thus, as long as the underlying teacher LM approximates the true next-token distribution well enough, it can lead to improved (student) performance or equivalently smaller generalization gap by striking a balance between the divergence (or bias) $\textsc{Div}(\zeta,\omega)$ and variance $\sum_t V_t$; as a result, realizing a form of *bias vs. variance* trade-off for LM pre-training.

[3] One could draw similar conclusion from Theorem 3.3 by extending the arguments from Menon et al. (2021) to the language modeling setting.

The variance reduction in the case of KD is the cleanest to observe by focusing on the last summand in $\sum_t V_t$, i.e., $V_T$. Towards this, recall from Assumption 3.4 that, for each $\boldsymbol{\theta} \in \Theta$, $V_T$ bounds the second-order moment of $\xi_T(\mathbf{x}; \boldsymbol{\theta})$. Define the short-hand

$$P^{(x_t,\omega)}_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}) := (1-\omega)\cdot \mathbb{1}_{x_t}(\cdot) + \omega \cdot P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq t-1}) \tag{50}$$

and write

$$\begin{aligned}
&\xi_T(\mathbf{x};\boldsymbol{\theta}) = \mathbb{E}\left[\ell^{\omega}(\mathbf{x};\boldsymbol{\theta})|\mathbf{x}_{\leq T-1}\right] - \mathbb{E}\left[\ell^{\omega}(\mathbf{x};\boldsymbol{\theta})|\mathbf{x}_{\leq T}\right] \\
&= \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\mathsf{CE}\big(P^{(x_t,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big)|\mathbf{x}_{\leq T-1}\right] - \\
&\quad \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\mathsf{CE}\big(P^{(x_t,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big)|\mathbf{x}_{\leq T}\right] \\
&\overset{(i)}{=} \frac{1}{T}\sum_{t=1}^{T-1}\mathsf{CE}\big(P^{(x_t,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big) + \mathbb{E}\left[\frac{1}{T}\mathsf{CE}\big(P^{(x_T,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq T-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq T-1})\big)|\mathbf{x}_{\leq T-1}\right] \\
&\quad - \frac{1}{T}\sum_{t=1}^{T}\mathsf{CE}\big(P^{(x_t,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq t-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big) \\
&\overset{(ii)}{=} \mathbb{E}\left[\frac{1}{T}\mathsf{CE}\big(P^{(x_T,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq T-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq T-1})\big)|\mathbf{x}_{\leq T-1}\right] - \frac{1}{T}\mathsf{CE}\big(P^{(x_T,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq T-1}), P_{\boldsymbol{\theta}}(\cdot|\mathbf{x}_{\leq t-1})\big) \\
&= (1-\omega)\cdot\left(\mathbb{E}\left[-\frac{1}{T}\cdot\log P_{\boldsymbol{\theta}}(x_T|\mathbf{x}_{\leq t-1})|\mathbf{x}_{\leq T-1}\right] + \frac{1}{T}\cdot\log P_{\boldsymbol{\theta}}(x_T|\mathbf{x}_{\leq T-1})\right) 
\end{aligned} \tag{51}$$

where $(i)$ follows we can remove expectation for those terms that are functions of those random variables that we condition on; and $(ii)$ follows by removing the terms that cancel each other; and the last line follows as we have

$$P^{(x_T,\omega)}_{\zeta,\rho}(\cdot|\mathbf{x}_{\leq T-1}) = (1-\omega)\cdot\mathbb{1}_{x_T}(\cdot) + \omega\cdot P_{\boldsymbol{\zeta},\rho}(\cdot|\mathbf{x}_{\leq T-1}).$$

It follows from (51) that, for any $\boldsymbol{\theta}\in\Theta$, we have

$$\mathbb{E}\left[\xi_T^2(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\zeta})\right] = (1-\omega)\cdot\mathrm{Var}\left[\frac{1}{T}\cdot\log P_{\boldsymbol{\theta}}(x_T|\mathbf{x}_{\leq t-1}) \,\middle|\, \mathbf{x}_{\leq T-1}\right], \tag{52}$$

where $\mathrm{Var}\left[\cdot|\cdot\right]$ denotes conditional variance. Note that (52) shows that $V_T$ decreases with $\omega$ in $[0,1]$. This highlights that KD, i.e., $\omega > 0$, would realize a smaller variance than standard pre-training, i.e., $\omega = 0$. Thus, to realize improved generalization via KD, one needs to select the distillation weight $\omega$ so that the *variance reduction* via KD offsets the divergence term $\textsc{Div}(\zeta,\omega)$. In particular, when the teacher LM approximates the ground truth next-token distribution very well, i.e., $\textsc{Div}(\zeta,\omega)$ term is small even for a relatively large value of $\omega$, the variance reduction via KD becomes prominent, ensuring significant improvement over standard pre-training in terms of generalization performance.

## D Bounding excess risk for KD

Different from the surrogate (empirical or population) risks utilized in the main text (cf. Section 3),which utilize the cross-entropy loss as a surrogate loss, one could directly work with the risk defined with respect to a particular evaluation metric (and the corresponding loss) that one cares about. Since our training focuses on correct next-token prediction, we can focus on the accuracy of the next-token prediction under greedy-decoding as one such metric. This amounts to the following (population) risk with respect to $0/1$-*loss*.

$$\begin{aligned}
R_{0/1}(\boldsymbol{\theta}) &:= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\Big[\sum_{t=1}^{T} 1\{\arg\max_{v} P_{\boldsymbol{\theta}}(v|\mathbf{x}_{\leq t-1}) \neq x_t\Big] \\
&= \sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_{\leq t-1}\sim\mathcal{D}}\Big[\sum_{v\in\mathcal{V}}\mathcal{D}(v|\mathbf{x}_{\leq t-1})\cdot 1\{\arg\max_{v'} P_{\boldsymbol{\theta}}(v'|\mathbf{x}_{\leq t-1})\neq v\}\Big].
\end{aligned} \tag{53}$$

A large body of literature (see, e.g., Bartlett et al., 2006; Zhang, 2004; Steinwart, 2007; Pires & Szepesvári, 2016, and references therein) has studied *calibration functions* that enable converting bounds on *excess surrogate risk* to control the *excess risk*. Applying the calibration functions for the cross-entropy loss (Pires & Szepesvári, 2016), we obtain the following bound on the excess risk for next-token prediction:

$$R_{0/1}(\hat{\boldsymbol{\theta}}) - R_{0/1}(\boldsymbol{\theta}^*) \leq \mathrm{g}^{-1}\left(R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}^*)\right), \tag{54}$$

where $\mathrm{g}^{-1}(\cdot)$ denotes the inverse of the function $\mathrm{g} : \epsilon \mapsto \frac{1}{2}\big((1-\epsilon)\log(1-\epsilon) + (1+\epsilon)\log(1+\epsilon)\big)$.

## E Experimental Setup Details

**Model architectures and pre-training data.** We work with standard decoder-only Transformer-based LMs. Our small model (SLM) has 1.5B parameters. It comprises a 44 layer Transformer network with model dimension 1024, MLP hidden dimension 8192, and 4 attention heads. For the larger LM, we employ 2.8B parameter models consisting of 92 layer Transformer networks with model dimension 1024, MLP hidden dimension 8192, and 4 attention heads based on multi-query attention (Shazeer, 2019). We use a SentencePiece tokenizer (Kudo & Richardson, 2018) from Du et al. (2022) with a vocabulary size of $256K$. We employ weight tying (Press & Wolf, 2017), i.e., the same vocabulary embedding parameters are used for input token embedding and output embedding layers.

We pre-train all LMs on the Pile dataset (Gao et al., 2020) by minimizing the UL2 objective (Tay et al., 2023) with a mixture of four tasks: (1) *causal LM* task; (2) *prefix LM* task with mean prefix length of $1/4$th the sequence length, (3) *span corruption task* with $r = 15\%$ of the tokens corrupted and mean corrupted span length $\mu = 3$; and (4) *span corruption task* with $r = 50\%$ of the tokens corrupted and mean corrupted span length $\mu = 32$. The four tasks are mixed at a ratio of 6:2:1:1. We pre-train LMs for approximately 545 billion tokens, with a batch size of 2048 and input sequence length of 1280. This translates to a little over two epochs on the Pile data. As for the optimization method, we utilize Adafactor algorithm (Shazeer & Stern, 2018). We use a cosine learning rate decay schedule with a peak learning rate of 0.001, 4000 warmup steps and final learning rate of 0.0001. Training is done on 1024 TPU-v5e chips with JAX (Bradbury et al., 2018) and SeqIO (Roberts et al., 2022).

## F Few-shot evaluation tasks

We performed a comprehensive few-shot evaluation of pre-trained LMs on 28 benchmarks. Below, we list these by organizing them according to the corresponding domain.

**World Knowledge**: NQ-Open (Lee et al., 2019), TriviaQA (Joshi et al., 2017), TyDiQA-NoContext (English)(Clark et al., 2020), Web Questions (Berant et al., 2013).

**Reading Comprehension:** RACE-M, RACE-H (Lai et al., 2017), SQuADv2 (Lee et al., 2020), TyDiQA-GoldP (English)(Clark et al., 2020).

**Commonsense Reasoning:** ARC (Easy) and ARC (Challenge) (Clark et al., 2018), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PiQA (Bisk et al., 2020), StoryCloze (Mostafazadeh et al., 2016), Winogrande (Sakaguchi et al., 2020).

**SuperGLUE (Wang et al., 2019):** BoolQ (Clark et al., 2019), CB (de Marneffe et al., 2019), COPA (Gordon et al., 2012), RTE (Dagan et al., 2006), WiC (Pilehvar & Camacho-Collados, 2018), WSC (Levesque et al., 2012), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018).

**Natural Language Generation (NLG):** English portions of the three benchmarks – XLSum (Hasan et al., 2021), XSum (Narayan et al., 2018) and WikiLingua (Ladhak et al., 2020).

**Open-ended Cloze task:** LAMBADA Paperno et al. (2016).

**Code generation:** Mostly Basic Python Problems (MBPP) (Austin et al., 2021).

## G Additional Few-shot evaluation results

Table 5 is an expansion of Table 2 in the main text. All evaluations are 1-shot, except for MBPP which is 3-shot. In the metric column, *EM*, *Acc*, and *Rg2* are abbreviations for *Exact Match*, *Accuracy*, and *Rouge2*, respectively. For MBPP, the metric is the fraction of success ignoring challenge problems. As mentioned in Section 5.1, for each benchmark, we typically report the corresponding prevalent metric in the literature. For TyDiQA benchmarks, we report the F1 score as opposed to *EM* as it is the primary metric in Clark et al. (2020). For MultiRC in SuperGLUE, we report F1 metric as per Du et al. (2022).

Table 5: **Comprehensive few-shot performance of pre-trained LMs.** SLM serves as the teacher LM for SALT & $\text{SALT}_{\text{DS}}$ during the KD phase of their pre-training and for RKD throughout its pre-training. BASELINE employs standard pre-training *without* KD from SLM. SALT and $\text{SALT}_{\text{DS}}$ already outperform BASELINEin terms of average few-shot performance at 70% of their training step budget, thereby improving both training efficiency and model quality. RKD, i.e., naively preforming KD from the small model through the pre-training, performs much worse than BASELINE. The best and second-best results for each domain are **boldfaced** and underlined, respectively.

| Domain | Dataset | Metric | SLM | BASELINE | RKD | SALT | | $\text{SALT}_{\text{DS}}$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | *@100% steps* | *@100% steps* | *@70% steps* | *@100% steps* | *@70% steps* | *@100% steps* |
| World Knowledge | NaturalQuestions-Open | *EM* | 5.90 | 8.70 | 6.70 | 9.40 | **10.10** | 8.40 | 9.00 |
| | TriviaQA | *EM* | 30.09 | 43.15 | 34.87 | 39.87 | **43.71** | 39.37 | 41.27 |
| | TyDiQA-NoContext | *F1* | 22.20 | **28.20** | 26.10 | 27.90 | 27.10 | 25.90 | 27.20 |
| | WebQuestions | *EM* | 5.40 | 8.70 | 7.10 | 9.20 | **9.90** | 8.90 | 9.40 |
| | **Domain average** | | 15.90 | 22.19 | 18.69 | 21.59 | **22.70** | 20.64 | 21.72 |
| Reading Comprehension | RACE-M | *Acc* | 52.60 | 57.00 | 54.00 | 58.60 | **58.90** | 57.90 | 58.40 |
| | RACE-H | *Acc* | 37.50 | **42.30** | 39.70 | 42.20 | **42.30** | 42.10 | **42.30** |
| | SQuADv2 | *EM* | 43.30 | 54.80 | 50.90 | 54.60 | 55.90 | 57.60 | **57.90** |
| | TyDiQA-GoldP | *F1* | 51.80 | 57.90 | 59.40 | 58.80 | **61.10** | 59.80 | **61.10** |
| | **Domain average** | | 46.30 | 53.00 | 51.00 | 53.55 | 54.55 | 54.35 | **54.93** |
| Commonsense Reasoning | ARC-E | *Acc* | 64.60 | 68.40 | 66.00 | 67.60 | 67.60 | **69.40** | 69.00 |
| | ARC-C | *Acc* | 32.40 | 37.10 | 33.70 | 38.00 | **38.40** | 38.10 | 37.30 |
| | HellaSwag | *Acc* | 56.00 | 62.80 | 56.20 | 62.00 | 63.30 | 63.10 | **63.80** |
| | OpenBookQA | *Acc* | 48.00 | **50.00** | 45.80 | 47.20 | 48.20 | 47.60 | 48.20 |
| | PiQA | *Acc* | 72.00 | **75.40** | 72.60 | 73.20 | 73.70 | 74.10 | 73.90 |
| | StoryCloze | *Acc* | 73.10 | **77.20** | 73.70 | 76.90 | 76.80 | 77.00 | 77.10 |
| | WinoGrande | *Acc* | 58.20 | 63.00 | 60.10 | 64.00 | 63.70 | 64.70 | **65.40** |
| | **Domain average** | | 57.76 | 61.99 | 58.30 | 61.27 | 61.67 | 62.00 | **62.10** |
| | LAMBADA | *Acc* | 26.90 | 36.20 | 31.10 | 50.70 | 48.30 | 48.00 | **53.00** |
| SuperGLUE | BoolQ | *Acc* | 63.40 | 64.30 | 62.50 | 64.10 | 62.30 | **65.50** | 64.30 |
| | CB | *Acc* | 37.50 | 58.90 | 50.00 | **60.70** | 53.60 | 55.40 | 53.60 |
| | COPA | *Acc* | 77.00 | 79.00 | 71.00 | 76.00 | 77.00 | **81.00** | 77.00 |
| | MultiRC | *F1* | 53.80 | 54.20 | 53.50 | 57.50 | **58.60** | 50.70 | 53.00 |
| | RTE | *Acc* | 55.20 | 55.60 | **59.90** | 57.80 | 58.50 | 54.20 | 58.50 |
| | ReCoRD | *Acc* | 84.80 | 87.10 | 85.20 | 86.60 | 86.90 | 87.20 | **87.30** |
| | WiC | *Acc* | 48.40 | 47.20 | 47.20 | 49.80 | 48.10 | 50.00 | **50.90** |
| | WSC | *Acc* | 72.60 | 77.90 | 74.00 | 77.90 | 77.20 | **83.90** | 80.00 |
| | **Domain average** | | 61.59 | 65.53 | 62.91 | **66.30** | 65.28 | 65.99 | 65.58 |
| NLG | G*EM*-XLSum | *Rg2* | 2.80 | 4.10 | 3.40 | 4.40 | 4.40 | **4.60** | **4.60** |
| | G*EM*-XSum | *Rg2* | 2.80 | 5.10 | 3.20 | 5.00 | 5.10 | **5.40** | **5.40** |
| | WikiLingua | *Rg2* | 3.80 | 4.60 | 3.60 | 4.50 | **4.70** | 4.40 | 4.50 |
| | **Domain average** | | 3.13 | 4.60 | 3.40 | 4.63 | 4.73 | 4.80 | **4.83** |
| | MBPP | *Acc* | 9.60 | 16.20 | 11.40 | 15.60 | 17.00 | 16.60 | **17.80** |
| | **Average (28 tasks)** | | 42.56 | 47.32 | 44.39 | 47.86 | 47.94 | 47.89 | **48.26** |

## H Ablation study of various design choices in SALT

In this section, we explore how various design choices pertaining SALT affect its final performance.

**Distillation from a better quality small model.** So far we assumed that SLM is also pre-trained for the same number of tokens as the LLM. Since training for SLM is relatively cheaper, one could consider a scenario where one invests more compute resources in improving the small model if it can eventually be beneficial in improving the LLM quality via SALT. Towards this, we employ a small LM that is trained $\sim 2.5$ times longer – 498K steps vs. 208K steps in Section 5.2.[4] As evident in Table 6, SALT is indeed able to utilize the better small model as a teacher in the KD phase to further improve the LLM quality, as measured by the average few-shot performance.

**Varying transition point.** A key design choice for SALT is the selection of the transition point $n_{\mathrm{KD}}$ from KD phase (first stage) to standard training (second stage). Table 7 shows few-shot performance of SALT as we vary the transition point. Note that SALT ensures quality gains for LLM with a wide range of values for $n_{\mathrm{KD}}$ while demonstrating an inverted U-shape for LLM quality. We see consistent performance improvement from $n_{\mathrm{KD}} = 0$ (equivalent to BASELINE) to $n_{\mathrm{KD}} = 60$K which eventually degrades at $n_{\mathrm{KD}} = 208$K (equivalent to RKD). Given the training overhead of KD phase (see discussion in Section 5.2), smaller value of $n_{\mathrm{KD}}$ helps ensure training efficiency gains via SALT. Thus, we worked with $n_{\mathrm{KD}} = 36$K in Section 5.2 as $n_{\mathrm{KD}} = 60$K only provides marginal quality gains if one takes into account the increased training cost due to longer KD phase.

**Different transition strategies.** In our study thus far, we have worked with *Step* transition between the two training stages in SALT where we abruptly stop performing KD after $n_{\mathrm{KD}}$ training steps. Looking at Figure 2, this causes an abrupt change in the model behavior during training, as observed in the next-token prediction accuracy curve for the training set (see similar behavior for log-perplexity in Figure 3). This raises a question if a smoother transition between the two stages can improve the training stability and thereby ensure higher final LLM quality. While there is a large space of potential choices of such smooth transition strategies, here we explore two natural candidates: (1) *Linear decay* where we linearly decrease the distillation loss weight to 0 between $n_{\mathrm{KD},1} = 32$K and $n_{\mathrm{KD}} = 36$K steps; and (2) *Linear ratio decay* where we linearly decrease the ratio of distillation loss weight and standard loss weight $\frac{\omega}{1-\omega}$ to 0 between $n_{\mathrm{KD},1} = 32$K and $n_{\mathrm{KD},2} = 36$K training steps. As recorded in Table 8, the step transition constitutes a reasonable design choice for SALT as it outperforms both the considered alternatives in terms of average few-shot performance of the resulting pre-trained LLM.

[4]This approach aligns with the recent studies (Touvron et al., 2023; Gadre et al., 2024) that train small LMs well beyond the their optimal compute budget as predicted by neural scaling laws (Hoffmann et al., 2022).

Table 6: **Effect of improved SLM(comprehensive few-shot evaluation).** SALT with a better teacher – a SLM trained for 498K steps as opposed to 208K steps – yields LLM with better average few-shot performance. For each benchmark, the best and second best results are **boldfaced** and <u>underlined</u>, respectively.

| Domain | Dataset | Metric | SLM trained for 208K steps | SLM trained for 498K steps | SALT w/ KD from SLM trained for 208K steps | SALT w/ KD from SLM trained for 498K steps |
|---|---|---|---|---|---|---|
| World Knowledge | NaturalQuestions-Open | *EM* | 5.90 | 6.30 | **10.10** | <u>9.00</u> |
| | TriviaQA | *EM* | 30.09 | 31.74 | **43.71** | <u>41.61</u> |
| | TyDiQA-NoContext | *F1* | 22.20 | 23.80 | **27.10** | <u>26.20</u> |
| | WebQuestions | *EM* | 5.40 | 7.60 | **9.90** | <u>9.10</u> |
| | **Domain average** | | 15.90 | 17.36 | **22.70** | <u>21.48</u> |
| Reading Comprehension | RACE-M | *Acc* | 52.60 | 54.40 | **58.90** | <u>57.00</u> |
| | RACE-H | *Acc* | 37.50 | 39.40 | **42.30** | <u>42.00</u> |
| | SQuADv2 | *EM* | 43.30 | 49.00 | <u>55.90</u> | **57.90** |
| | TyDiQA-GoldP | *F1* | 51.80 | 55.90 | **61.10** | <u>56.80</u> |
| | **Domain average** | | 46.30 | 49.67 | **54.55** | <u>53.43</u> |
| Commonsense Reasoning | ARC-E | *Acc* | 64.60 | 65.50 | <u>67.60</u> | **69.30** |
| | ARC-C | *Acc* | 32.40 | 34.30 | <u>38.40</u> | **39.10** |
| | HellaSwag | *Acc* | 56.00 | 57.80 | **63.30** | <u>63.20</u> |
| | OpenBookQA | *Acc* | 48.00 | 46.40 | <u>48.20</u> | **49.00** |
| | PiQA | *Acc* | 72.00 | 72.90 | <u>73.70</u> | **74.60** |
| | StoryCloze | *Acc* | 73.10 | 75.00 | <u>76.80</u> | **76.90** |
| | WinoGrande | *Acc* | 58.20 | 59.40 | <u>63.70</u> | **63.80** |
| | **Domain average** | | 57.76 | 58.76 | <u>61.67</u> | **62.27** |
| | LAMBADA | *Acc* | 26.90 | 37.80 | **48.30** | <u>47.80</u> |
| SuperGLUE | BoolQ | *Acc* | <u>63.40</u> | 61.40 | 62.30 | **65.80** |
| | CB | *Acc* | 37.50 | 42.90 | <u>53.60</u> | **73.20** |
| | COPA | *Acc* | 77.00 | <u>78.00</u> | 77.00 | **79.00** |
| | MultiRC | *F1* | <u>53.80</u> | 48.40 | **58.60** | 53.20 |
| | RTE | *Acc* | 55.20 | 52.30 | <u>58.50</u> | **61.70** |
| | ReCoRD | *Acc* | 84.80 | 85.50 | <u>86.90</u> | **87.10** |
| | WIC | *Acc* | <u>48.40</u> | 47.30 | 48.10 | **49.20** |
| | WSC | *Acc* | 72.60 | 72.30 | <u>77.20</u> | **79.30** |
| | **Domain average** | | 61.59 | 61.01 | <u>65.28</u> | **68.56** |
| NLG | *GEM*-XLSum | *Rg2* | 2.80 | 3.50 | **4.40** | <u>4.30</u> |
| | *GEM*-XSum | *Rg2* | 2.80 | 3.10 | <u>5.10</u> | **5.60** |
| | WikiLingua | *Rg2* | 3.80 | 3.80 | **4.70** | <u>4.40</u> |
| | **Domain average** | | 3.13 | 3.47 | <u>4.73</u> | **4.77** |
| | MBPP | *Acc* | 9.60 | 12.80 | <u>17.00</u> | **17.40** |
| | **Average (28 tasks)** | | 42.56 | 43.88 | <u>47.94</u> | **48.70** |

Table 7: **Effect of varying transitions step (comprehensive few-shot evaluation).** The performance improvement via SALT over BASELINE is stable in a wide range of $n_{\mathrm{KD}}$ (20k to 60k steps). Eventually, with much larger $n_{\mathrm{KD}}$, SALT performance degrades significantly (208k steps). For each benchmark, the best and second best results are **boldfaced** and underlined, respectively.

| Domain | Dataset | Metric | SLM | BASELINE | SALT w/ $n_{\mathrm{KD}} = 20\mathrm{K}$ | SALT w/ $n_{\mathrm{KD}} = 36\mathrm{K}$ | SALT w/ $n_{\mathrm{KD}} = 60\mathrm{K}$ | SALT w/ $n_{\mathrm{KD}} = 208\mathrm{K}$ (RKD) |
|---|---|---|---|---|---|---|---|---|
| **World Knowledge** | NaturalQuestions-Open | *EM* | 5.90 | 8.70 | 8.90 | **10.10** | 9.30 | 6.70 |
| | TriviaQA | *EM* | 30.09 | 43.15 | 41.52 | **43.71** | 42.84 | 34.87 |
| | TyDiQA-NoContext | *F1* | 22.20 | **28.20** | 26.40 | 27.10 | 26.60 | 26.10 |
| | WebQuestions | *EM* | 5.40 | 8.70 | 8.20 | **9.90** | 8.60 | 7.10 |
| | **Domain average** | | 15.90 | 22.19 | 21.26 | **22.70** | 21.83 | 18.69 |
| **Reading Comprehension** | RACE-M | *Acc* | 52.60 | 57.00 | 58.70 | **58.90** | 58.60 | 54.00 |
| | RACE-H | *Acc* | 37.50 | **42.30** | 41.00 | **42.30** | 42.10 | 39.70 |
| | SQuADv2 | *EM* | 43.30 | 54.80 | 55.30 | **55.90** | 55.50 | 50.90 |
| | TyDiQA-GoldP | *F1* | 51.80 | 57.90 | 56.50 | **61.10** | 59.30 | 59.40 |
| | **Domain average** | | 46.30 | 53.00 | 52.88 | **54.55** | 53.88 | 51.00 |
| **Commonsense Reasoning** | ARC-E | *Acc* | 64.60 | **68.40** | 67.80 | 67.60 | **68.40** | 66.00 |
| | ARC-C | *Acc* | 32.40 | 37.10 | 38.10 | 38.40 | **38.70** | 33.70 |
| | HellaSwag | *Acc* | 56.00 | 62.80 | 62.80 | **63.30** | 62.90 | 56.20 |
| | OpenBookQA | *Acc* | 48.00 | **50.00** | 48.00 | 48.20 | 48.20 | 45.80 |
| | PiQA | *Acc* | 72.00 | **75.40** | **75.40** | 73.70 | 74.40 | 72.60 |
| | StoryCloze | *Acc* | 73.10 | **77.20** | 76.90 | 76.80 | 76.50 | 73.70 |
| | WinoGrande | *Acc* | 58.20 | 63.00 | 63.40 | **63.70** | 62.00 | 60.10 |
| | **Domain average** | | 57.76 | **61.99** | 61.77 | 61.67 | 61.59 | 58.30 |
| | LAMBADA | *Acc* | 26.90 | 36.20 | 44.70 | 48.30 | **53.30** | 31.10 |
| **SuperGLUE** | BoolQ | *Acc* | 63.40 | **64.30** | 63.90 | 62.30 | 63.80 | 62.50 |
| | CB | *Acc* | 37.50 | 58.90 | **60.70** | 53.60 | 55.40 | 50.00 |
| | COPA | *Acc* | 77.00 | **79.00** | 76.00 | 77.00 | 77.00 | 71.00 |
| | MultiRC | *F1* | 53.80 | 54.20 | 53.80 | **58.60** | 55.20 | 53.50 |
| | RTE | *Acc* | 55.20 | 55.60 | 52.30 | 58.50 | **59.90** | **59.90** |
| | ReCoRD | *Acc* | 84.80 | **87.10** | 86.90 | 86.90 | 86.70 | 85.20 |
| | WIC | *Acc* | 48.40 | 47.20 | **51.30** | 48.10 | 50.00 | 47.20 |
| | WSC | *Acc* | 72.60 | **77.90** | 77.50 | 77.20 | **77.90** | 74.00 |
| | **Domain average** | | 61.59 | 65.53 | 65.30 | 65.28 | **65.74** | 62.91 |
| **NLG** | *GEM*-XLSum | *Rg2* | 2.80 | 4.10 | 4.50 | 4.40 | **4.70** | 3.40 |
| | *GEM*-XSum | *Rg2* | 2.80 | 5.10 | **5.80** | 5.10 | 4.80 | 3.20 |
| | WikiLingua | *Rg2* | 3.80 | 4.60 | 4.30 | **4.70** | 4.60 | 3.60 |
| | **Domain average** | | 3.13 | 4.60 | **4.87** | 4.73 | 4.70 | 3.40 |
| | MBPP | *Acc* | 9.60 | 16.20 | 16.60 | **17.00** | 16.40 | 11.40 |
| | **Average (28 tasks)** | | 42.56 | 47.32 | 47.40 | 47.94 | **47.99** | 44.39 |

Table 8: **Effect of different transition strategies (comprehensive few-shot evaluation.** The Step transition used in this work (cf. Algorithm 1) performs well compared to two natural alternative strategies. For each benchmark, the best and second best results are **boldfaced** and underlined, respectively.

| Domain | Dataset | Metric | SLM | BASELINE | SALT w/ Step | SALT w/ Linear decay | SALT w/ Linear ratio decay |
|---|---|---|---|---|---|---|---|
| World Knowledge | NaturalQuestions-Open | *EM* | 5.90 | 8.70 | **10.10** | 8.20 | 8.10 |
| | TriviaQA | *EM* | 30.09 | 43.15 | **43.71** | 43.46 | 43.51 |
| | TyDiQA-NoContext | *F1* | 22.20 | 28.20 | 27.10 | **28.40** | 27.20 |
| | WebQuestions | *EM* | 5.40 | 8.70 | **9.90** | 8.20 | 8.40 |
| | **Domain average** | | 15.90 | 22.19 | **22.70** | 22.07 | 21.80 |
| Reading Comprehension | RACE-M | *Acc* | 52.60 | 57.00 | **58.90** | 57.90 | 57.40 |
| | RACE-H | *Acc* | 37.50 | 42.30 | 42.30 | 42.10 | **43.50** |
| | SQuADv2 | *EM* | 43.30 | 54.80 | 55.90 | 56.40 | **57.10** |
| | TyDiQA-GoldP | *F1* | 51.80 | 57.90 | **61.10** | 58.30 | 57.80 |
| | **Domain average** | | 46.30 | 53.00 | **54.55** | 53.68 | 53.95 |
| Commonsense Reasoning | ARC-E | *Acc* | 64.60 | 68.40 | 67.60 | 68.60 | **68.70** |
| | ARC-C | *Acc* | 32.40 | 37.10 | 38.40 | 38.60 | **39.80** |
| | HellaSwag | *Acc* | 56.00 | 62.80 | 63.30 | 63.30 | **63.50** |
| | OpenBookQA | *Acc* | 48.00 | **50.00** | 48.20 | 48.00 | 47.40 |
| | PiQA | *Acc* | 72.00 | **75.40** | 73.70 | 74.60 | 73.90 |
| | StoryCloze | *Acc* | 73.10 | **77.20** | 76.80 | 76.60 | 76.50 |
| | WinoGrande | *Acc* | 58.20 | 63.00 | **63.70** | 62.70 | 63.10 |
| | **Domain average** | | 57.76 | **61.99** | 61.67 | 61.77 | 61.84 |
| | LAMBADA | *Acc* | 26.90 | 36.20 | **48.30** | 40.50 | 42.60 |
| SuperGLUE | BoolQ | *Acc* | 63.40 | 64.30 | 62.30 | **67.90** | 66.50 |
| | CB | *Acc* | 37.50 | **58.90** | 53.60 | 44.60 | 46.40 |
| | COPA | *Acc* | 77.00 | 79.00 | 77.00 | 79.00 | **81.00** |
| | MultiRC | *F1* | 53.80 | 54.20 | 58.60 | 53.90 | **61.60** |
| | RTE | *Acc* | 55.20 | 55.60 | **58.50** | 56.30 | 55.20 |
| | ReCoRD | *Acc* | 84.80 | 87.10 | 86.90 | 87.00 | **87.30** |
| | WIC | *Acc* | 48.40 | 47.20 | 48.10 | 46.60 | **50.50** |
| | WSC | *Acc* | 72.60 | 77.90 | 77.20 | 78.60 | **78.90** |
| | **Domain average** | | 61.59 | 65.53 | 65.28 | 64.24 | **65.92** |
| NLG | *GEM*-XLSum | *Rg2* | 2.80 | 4.10 | 4.40 | **4.70** | 4.50 |
| | *GEM*-XSum | *Rg2* | 2.80 | **5.10** | **5.10** | 4.60 | **5.10** |
| | WikiLingua | *Rg2* | 3.80 | 4.60 | 4.70 | **4.80** | 4.60 |
| | **Domain average** | | 3.13 | 4.60 | **4.73** | 4.70 | **4.73** |
| | MBPP | *Acc* | 9.60 | 16.20 | **17.00** | 15.20 | **17.00** |
| | **Average (28 tasks)** | | 42.56 | 47.32 | **47.94** | 47.11 | 47.75 |

# I LOG PERPLEXITY OF THE MODELS

Figure 3 shows the log perplexity of the SALT and RKD pre-trained models along with BASELINE and SLM. The log perplexity for RKD stays at a higher level than even SLM. Recall that RKD optimizes a sum of two losses – KD loss with weight $\omega = 0.667$ and the standard one-hot training loss with weight $1 - \omega$. As the training log perplexity plotted in Figure 3 is the same as the standard hot training loss, the methods which directly optimize for that alone (BASELINE, SLM and in the second stage, SALT) have lower log perplexity on training set than RKD which optimizes additionally for distillation loss.

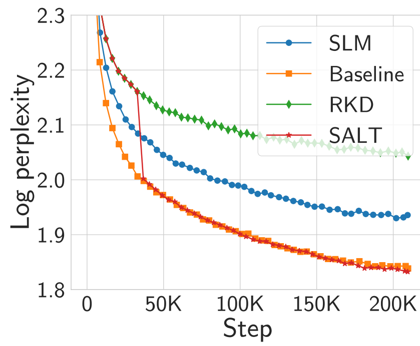


Figure 3: Log perplexity for different models during their pre-training, as measured on a subset of the Pile training set.

## J Additional results: Learning easy vs. hard instance via SALT

**Creation of different hardness buckets.** For each evaluation benchmark, we first assign a relative rank to each test instance/example in the benchmark, representing its degree of difficulty. A test example with the lowest rank (easiest) is the one on which the small teacher LM achieves the largest task evaluation score, e.g., Rouge-2 metric for the XLSum task. Similarly, subsequent test examples are assigned ranks in descending order of the task evaluation score achieved by the small teacher LM. If two examples have the same evaluation score, the one with higher confidence score from the teacher (on its generated output) is deemed to have a lower rank. Each test example is assigned to one of the three buckets: 'easy', 'medium', or 'hard', according to whether its difficulty rank is in the first, second, or third tertile, respectively.

In Tables 9, 10 and 11, we report the results for SQuAD-v2, TriviaQA and LAMBADA respectively, sliced by difficulty level.

Table 9: **Few-shot evaluation on different buckets of SQuAD-v2.** Each number shows average Exact Match scores on the corresponding bucket. We use gray, green, and red to highlight the results similar to, better than, and worse than BASELINE performance, respectively.

| | Evaluation stage (steps) | Easy | Medium | Hard |
|---|---|---|---|---|
| SLM | Final (208K) | 1.00 | 0.30 | 0.00 |
| BASELINE | Early (36K) | 0.86 | 0.41 | 0.23 |
| RKD | | 0.86 | 0.37 | 0.17 |
| SALT | | 0.86 | 0.37 | 0.17 |
| BASELINE | Final (208K) | 0.89 | 0.47 | 0.28 |
| RKD | | 0.91 | 0.42 | 0.20 |
| SALT | | 0.89 | 0.50 | 0.29 |

Table 10: **Few-shot evaluation on different buckets of TriviaQA.** Each number shows average Exact Match scores on the corresponding bucket. We use gray, green, and red to highlight the results similar to, better than, and worse than BASELINE performance, respectively.

| | Evaluation stage (steps) | Easy | Medium | Hard |
|---|---|---|---|---|
| SLM | Final (208K) | 0.90 | 0.00 | 0.00 |
| BASELINE | Early (36K) | 0.63 | 0.11 | 0.08 |
| RKD | | 0.67 | 0.10 | 0.06 |
| SALT | | 0.67 | 0.10 | 0.06 |
| BASELINE | Final (208K) | 0.80 | 0.28 | 0.22 |
| RKD | | 0.79 | 0.14 | 0.11 |
| SALT | | 0.81 | 0.27 | 0.23 |

Table 11: **Few-shot evaluation on different buckets of LAMBADA.** Each number shows average Accuracy on the corresponding bucket. We use gray, green, and red to highlight the results similar to, better than, and worse than BASELINE performance, respectively.

| | Evaluation stage (steps) | Easy | Medium | Hard |
|---|---|---|---|---|
| SLM | Final (208K) | 0.87 | 0.00 | 0.00 |
| BASELINE | Early (36K) | 0.47 | 0.12 | 0.12 |
| RKD | | 0.56 | 0.11 | 0.12 |
| SALT | | 0.56 | 0.11 | 0.12 |
| BASELINE | Final (208K) | 0.70 | 0.29 | 0.28 |
| RKD | | 0.65 | 0.17 | 0.17 |
| SALT | | 0.78 | 0.38 | 0.36 |

# K New results for 8.6B parameter model pre-training

In order to further validate the utility of the proposed SALT framework, we utilize it train even larger LM. In particular, we train a 8.6B parameter LM on the Pile dataset with the help of a 2.8B parameter small LM via SALT. In addition, we also explore $\text{SALT}_{\text{DS}}$ in this setting where, as per the discussion in Section 4, we utilize an early checkpoint (corresponding to $n_0 = 26K$ steps) of the 2.8B parameter model for data selection with $k = 10$ in (11).

As described in Appendix E, the 2.8B parameter model consists of a narrow and deep 92 layer Transformer network with model dimension 1024, MLP hidden dimension 8192, and 4 attention heads based on multi-query attention (Shazeer, 2019). As for the 8.6B parameter LM, it is based on a shallow and wide 32 layer Transformer network with model dimension 4096, MLP hidden dimension 16384, and 32 attention heads based on multi-query attention. Similar to the rest of the experiments in this work, we use a SentencePiece tokenizer (Kudo & Richardson, 2018) from Du et al. (2022) with a vocabulary size of $256K$. For the rest of the hyperparameters, we follow the setup described in Appendix E.

Next, focusing on the tasks listed in Appendix F, we present the few-shot performance for the 8.6B LMs trained via SALT and $\text{SALT}_{\text{DS}}$ and contrast that with the performance of the natural baseline – an 8.6B LM trained via the standard pre-training approach. Subsequently, Section K.2 explores the post-SFT performance for 8.6B parameter LMs on the same tasks considered in Section 5.3.

## K.1 Few-shot evaluations for 8.6B models pre-trained via 2.8B SLM teacher

Table 12: **Domain-wise few-shot performance of pre-trained 8.6B parameter LMs.** SALT and $\text{SALT}_{\text{DS}}$ utilize a 2.8B parameter SLM during their pre-training. Note that SALT and $\text{SALT}_{\text{DS}}$ already outperform BASELINE in terms of average few-shot performance at 70% of their training step budget, thereby improving both training efficiency and model quality. RKD (i.e., naively distilling from SLM throughout pre-training) performs much worse than BASELINE. The best and second-best results for each domain are **boldfaced** and underlined, respectively.

| Domain | # Tasks | SLM | BASELINE | SALT | | $\text{SALT}_{\text{DS}}$ | |
|---|---|---|---|---|---|---|---|
| | | | *@100% steps* | *@70% steps* | *@100% steps* | *@70% steps* | *@100% steps* |
| **World Knowledge** | 4 | 22.19 | 26.91 | 27.66 | **28.97** | 28.04 | <u>28.47</u> |
| **Reading Comprehension** | 4 | 53.00 | 56.40 | 56.83 | <u>57.42</u> | 56.10 | **57.48** |
| **Commonsense Reasoning** | 7 | 61.99 | 66.01 | 66.89 | <u>67.09</u> | 66.61 | **67.24** |
| **LAMBADA** | 1 | 36.20 | 58.70 | **65.50** | <u>64.80</u> | 54.30 | 55.00 |
| **SuperGLUE** | 8 | 65.53 | 69.69 | 69.19 | 70.38 | <u>71.06</u> | **71.26** |
| **NLG** | 3 | 4.60 | 5.40 | **5.97** | **5.97** | 5.23 | 5.30 |
| **MBPP** | 1 | 16.20 | 20.80 | 19.80 | 22.00 | <u>22.80</u> | **23.20** |
| **Average** | 28 | 47.32 | 51.73 | 52.24 | **52.96** | 52.29 | <u>52.81</u> |

Table 13: **Comprehensive few-shot performance of pre-trained 8.6B parameter LMs.** SLM is a 2.8B parameter model that serves as the teacher LM for SALT & $\text{SALT}_{\text{DS}}$ during the KD phase of their pre-training and for RKD throughout its pre-training. BASELINE employs standard pre-training *without* KD from SLM. SALT and $\text{SALT}_{\text{DS}}$ already outperform BASELINEin terms of average few-shot performance at 70% of their training step budget, thereby improving both training efficiency and model quality. The best and second-best results for each domain are **boldfaced** and underlined, respectively.

| Domain | Dataset | Metric | SLM | BASELINE | SALT | | $\text{SALT}_{\text{DS}}$ | |
|---|---|---|---|---|---|---|---|---|
| | | | | *@100% steps* | *@70% steps* | *@100% steps* | *@70% steps* | *@100% steps* |
| World Knowledge | NaturalQuestions-Open | *EM* | 8.70 | 10.50 | 11.80 | 11.50 | 12.00 | **12.30** |
| | TriviaQA | *EM* | 43.15 | 54.86 | 55.16 | 57.07 | 56.85 | **58.99** |
| | TyDiQA-NoContext | *F1* | 28.20 | 30.40 | 29.70 | **32.30** | 32.00 | 30.60 |
| | WebQuestions | *EM* | 8.70 | 11.90 | 14.00 | **15.00** | 11.30 | 12.00 |
| | **Domain average** | | 22.19 | 26.91 | 27.66 | **28.97** | 28.04 | 28.47 |
| Reading Comprehension | RACE-M | *Acc* | 57.00 | 60.70 | 61.70 | **62.40** | 59.60 | 60.70 |
| | RACE-H | *Acc* | 42.30 | 45.40 | 44.90 | **45.70** | 43.30 | 44.20 |
| | SQuADv2 | *EM* | 54.80 | 61.20 | 56.50 | 57.00 | 56.90 | **61.50** |
| | TyDiQA-GoldP | *F1* | 57.90 | 58.30 | 64.20 | **64.60** | **64.60** | 63.50 |
| | **Domain average** | | 53.00 | 56.40 | 56.83 | 57.42 | 56.10 | **57.48** |
| Commonsense Reasoning | ARC-E | *Acc* | 68.40 | 73.30 | 74.10 | **74.60** | 73.00 | 74.00 |
| | ARC-C | *Acc* | 37.10 | 42.70 | 45.00 | **46.20** | 43.90 | 44.50 |
| | HellaSwag | *Acc* | 62.80 | 70.40 | 70.00 | 70.80 | 70.70 | **71.60** |
| | OpenBookQA | *Acc* | 50.00 | 51.20 | **53.40** | 53.20 | 53.00 | 52.80 |
| | PiQA | *Acc* | 75.40 | 77.30 | 76.60 | 76.40 | 76.80 | **77.70** |
| | StoryCloze | *Acc* | 77.20 | 80.00 | 80.20 | 80.00 | 80.20 | **80.30** |
| | WinoGrande | *Acc* | 63.00 | 67.20 | 68.90 | 68.40 | 68.70 | **69.80** |
| | **Domain average** | | 61.99 | 66.01 | 66.89 | 67.09 | 66.61 | **67.24** |
| | LAMBADA | *Acc* | 36.20 | 58.70 | **65.50** | 64.80 | 54.30 | 55.00 |
| SuperGLUE | BoolQ | *Acc* | 64.30 | 70.70 | 74.20 | 74.70 | **76.80** | 76.00 |
| | CB | *Acc* | 58.90 | 60.70 | 53.60 | 58.90 | **64.30** | **64.30** |
| | COPA | *Acc* | 79.00 | **87.00** | **87.00** | 84.00 | 85.00 | 86.00 |
| | MultiRC | *F1* | 54.20 | 55.90 | 52.60 | 55.80 | 59.90 | **61.70** |
| | RTE | *Acc* | 55.60 | 61.40 | 64.60 | **67.10** | 62.50 | 62.50 |
| | ReCoRD | *Acc* | 87.10 | 89.20 | 89.40 | 89.30 | **89.50** | 89.20 |
| | WIC | *Acc* | 47.20 | **50.50** | 50.00 | 50.00 | 47.30 | 47.20 |
| | WSC | *Acc* | 77.90 | 82.10 | 82.10 | **83.20** | **83.20** | **83.20** |
| | **Domain average** | | 65.53 | 69.69 | 69.19 | 70.38 | 71.06 | **71.26** |
| NLG | GEM-XLSum | *Rg2* | 4.10 | 4.90 | 5.30 | **5.40** | 4.80 | 4.70 |
| | GEM-XSum | *Rg2* | 5.10 | 6.10 | **7.20** | 7.00 | 6.00 | 6.20 |
| | WikiLingua | *Rg2* | 4.60 | 5.20 | 5.40 | **5.50** | 4.90 | 5.00 |
| | **Domain average** | | 4.60 | 5.40 | **5.97** | **5.97** | 5.23 | 5.30 |
| | MBPP | *Acc* | 16.20 | 20.80 | 19.80 | 22.00 | 22.80 | **23.20** |
| | **Average (28 tasks)** | | 47.32 | 51.73 | 52.24 | **52.96** | 52.29 | 52.81 |

### K.2 Post SFT results for 8.6B models pre-trained via 2.8B SLM teacher

Table 14: **Supervised fine-tuning (SFT) results.** Performance of various pre-trained checkpoints on downstream tasks after SFT. For each benchmark, pre-trained 8.6B models are fine-tuned on the corresponding train split and evaluated on the validation split (test split in case of GSM8K). *Acc*, *Rg1*, *Rg2*, and *RgL* represent the *Accuracy*, *Rouge-1*, *Rouge-2*, and *Rouge-Lsum* metrics, respectively.

| | GSM8K | XSum | | | CNN/DailyMail | | | ANLI-R1 | ANLI-R2 | ANLI-R3 |
|---|---|---|---|---|---|---|---|---|---|---|
| | *Acc* | *Rg1* | *Rg2* | *RgL* | *Rg1* | *Rg2* | *RgL* | *Acc* | *Acc* | *Acc* |
| Baseline | 41.85 | 45.10 | 22.68 | 37.36 | 43.73 | 21.19 | 41.29 | 68.80 | 58.90 | 60.58 |
| SALT | **42.84** | 45.37 | 23.04 | 37.69 | 43.69 | 21.16 | 41.22 | **70.20** | 59.30 | **63.25** |
| $\text{SALT}_{\text{DS}}$ | 42.23 | **45.81** | **23.34** | **38.14** | **43.80** | **21.28** | **41.35** | 69.30 | **59.50** | 62.17 |

## L Distribution of $\xi_t$

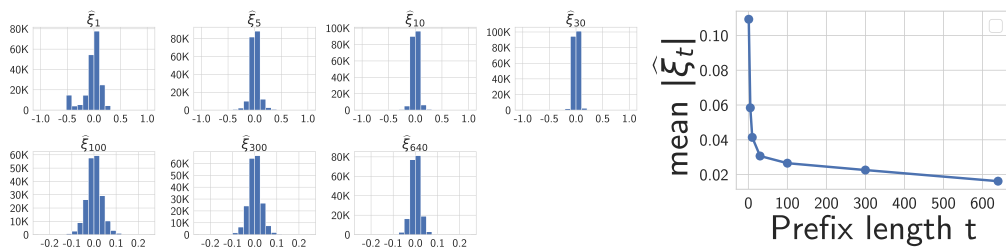


Figure 4: Histograms of $\widehat{\xi}_t$, for prefix lengths $t \in [1, 5, 10, 30, 100, 300, 640]$ for Baseline **2.8B parameter** model, estimated with $n_{\text{com}} = 64$ completions for each expectation in (55). The sequence length is 1280. The distribution gets concentrated around 0 as $t$ increases. In the bottom histograms, we reduce the x-axis range to about one-fourth that of the top histograms, to focus on the trend within the bottom row. In the rightmost plot, the mean of $|\widehat{\xi}_t|$ over validation set decreases rapidly with $t$.

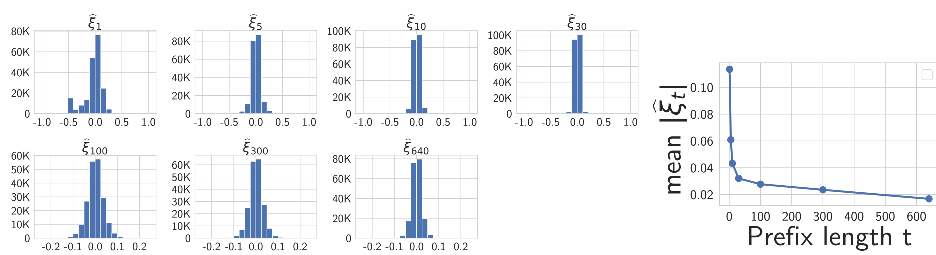


Figure 5: Histograms of $\widehat{\xi}_t$, $t \in [1, 5, 10, 30, 100, 300, 640]$ for Baseline **1.5B parameter** model, estimated with $n_{\text{com}} = 64$ completions for each expectation in (55). The sequence length is 1280. The distribution gets concentrated around 0 as $t$ increases. In the bottom histograms, we reduce the x-axis range to about one-fourth that of the top histograms, to focus on the trend within the bottom row. In the rightmost plot, the mean of $|\widehat{\xi}_t|$ over validation set decreases rapidly with $t$.

In this section, we attempt to understand the distribution of $\xi_t(\mathbf{x}; \boldsymbol{\theta})$ as $\mathbf{x} \sim \mathcal{D}$ for a learned model parameterized by $\boldsymbol{\theta}$. Recall from (8) that

$$\xi_t(\mathbf{x}; \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}\left[\ell^{\omega}(\mathbf{z}; \boldsymbol{\theta}) | \mathbf{z}_{\leq t-1} = \mathbf{x}_{\leq t-1}\right] - \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}\left[\ell^{\omega}(\mathbf{z}; \boldsymbol{\theta}) | \mathbf{z}_{\leq t} = \mathbf{x}_{\leq t}\right], \quad t \in [T]. \tag{55}$$

In order to estimate $\xi_t(\mathbf{x}; \boldsymbol{\theta})$, we intend to use a plugin estimator for the two expectations in this equation. To estimate, say, the second expectation above with a Monte-Carlo average, we need to be able to sample *completions* of $\mathbf{x}_{\leq t}$ so that $\mathbf{x}$ follows the distribution $\mathcal{D}$. The best access to data distribution $\mathcal{D}$ is via the training data; however, it is generally not possible to sample *multiple*

| $T$ | $t = 1$ | $t = 5$ | $t = 10$ | $t = 30$ | $t = 100$ | $t = 300$ | $t = 640$ |
|---|---|---|---|---|---|---|---|
| 64 | 0.338 | 0.133 | 0.106 | 0.087 | – | – | – |
| 128 | 0.261 | 0.099 | 0.074 | 0.057 | 0.042 | – | – |
| 256 | 0.188 | 0.082 | 0.060 | 0.044 | 0.033 | – | – |
| 512 | 0.134 | 0.071 | 0.053 | 0.039 | 0.031 | 0.019 | – |
| 1280 | 0.109 | 0.058 | 0.041 | 0.031 | 0.027 | 0.023 | 0.016 |

Table 15: Mean $|\widehat{\xi}_t|$ for **2.8B parameter** model decreases as we increase the sequence length $T$. Conversely, for a fixed sequence length $T$, mean $|\widehat{\xi}_t|$ decreases with prefix length $t$. "–" indicates that the entry is not meaningful because the prefix length $t$ is more than the sequence length.

| $T$ | $t = 1$ | $t = 5$ | $t = 10$ | $t = 30$ | $t = 100$ | $t = 300$ | $t = 640$ |
|---|---|---|---|---|---|---|---|
| 64 | 0.345 | 0.135 | 0.108 | 0.087 | – | – | – |
| 128 | 0.267 | 0.102 | 0.076 | 0.058 | 0.042 | – | – |
| 256 | 0.194 | 0.085 | 0.062 | 0.045 | 0.033 | – | – |
| 512 | 0.138 | 0.074 | 0.055 | 0.040 | 0.032 | 0.019 | – |
| 1280 | 0.113 | 0.061 | 0.043 | 0.032 | 0.028 | 0.024 | 0.017 |

Table 16: Mean $|\widehat{\xi}_t|$ for **1.5B parameter** model decreases as we increase the sequence length $T$. Conversely, for a fixed sequence length $T$, mean $|\widehat{\xi}_t|$ decreases with prefix length $t$. "–" indicates that the entry is not meaningful because the prefix length $t$ is more than the sequence length.

completions starting with the *same* prefix $\mathbf{x}_{\leq t}$ from the training data. Due to this difficulty, we sample completions from an *oracle* language model, as an approximation to the true data distribution. We use the BASELINE 8.6B model described in Appendix K as our oracle.

For a sequence $\mathbf{x}$ and prefix length $t \in [T]$, we employ a plugin estimate

$$\widehat{\xi}_t(\mathbf{x}; \boldsymbol{\theta}) := \frac{1}{n_{\text{com}}} \sum_{i=1}^{n_{\text{com}}} \ell^{\omega}([\mathbf{x}_{1:t-1}, \mathbf{y}^i(\mathbf{x}_{1:t-1})]; \boldsymbol{\theta}) - \frac{1}{n_{\text{com}}} \sum_{i=1}^{n_{\text{com}}} \ell^{\omega}([\mathbf{x}_{1:t}, \mathbf{y}^i(\mathbf{x}_{1:t})]; \boldsymbol{\theta}) \tag{56}$$

where $\mathbf{y}^i(\mathbf{x}_{1:s}), s \in [T]$ is a completion of $\mathbf{x}_{1:s}$, generated by the oracle, with a length of $|\mathbf{y}^i(\mathbf{x}_{1:s})| = (T - s)$ so that the concatenation $[\mathbf{x}_{1:s}, \mathbf{y}^i(\mathbf{x}_{1:s})]$ has length $T$. $n_{\text{com}}$ denotes the number of completions sampled from the oracle for estimating the expectations.

We compute $\widehat{\xi}_t(\mathbf{x}; \boldsymbol{\theta})$ for two models: BASELINE 1.5B and BASELINE 2.8B LMs. As for $\mathbf{x}$, we employ sequences in the validation set (held out from training any model, including the oracle). The number of completions is $n_{\text{com}} = 64$. The validation set size is $\sim 200K$.

In Figure 4, we observe that for 2.8B LM the estimates of $\widehat{\xi}_t$ increasingly concentrate around 0 as $t$ increases. Further, the mean $|\widehat{\xi}_t|$ decreases quickly with $t$. For the first few tokens of the sequence, it is hard to predict the next token, because the context is not sufficient to make a good prediction. These observations suggest that the magnitude of $\xi_t$ decreases with $t$. Intuitively, the upper bounds $C_t$ and $V_t$ defined in Assumption 3.4 should also decrease with $t$. (Similar results hold for 1.5B LM in Figure 5.)

Table 15 shows that for the 2.8B LM, the mean $|\widehat{\xi}_t|$ decreases as the sequence length $T$ increases from 64 to 1280. For modern LMs configured to train on much longer sequences, the table indicates that the mean $|\xi_t|$ are likely to be small. Table 16 shows similar behavior for 1.5B LM.