DIRECT ALIGNMENT OF DRAFT MODEL FOR SPECULATIVE DECODING WITH CHAT-FINE-TUNED LLMS

Raghavv Goel,* Mukul Gagrani, Wonseok Jeon, Junyoung Park, Mingu Lee, Christopher Lott
Qualcomm AI Research

ABSTRACT

Text generation with Large Language Models (LLMs) is known to be memory bound due to the combination of their auto-regressive nature, huge parameter counts, and limited memory bandwidths, often resulting in low token rates. Speculative decoding has been proposed as a solution for LLM inference acceleration. However, since draft models are often unavailable in the modern open-source LLM families, e.g., for Llama 2 7B, training a high-quality draft model is required to enable inference acceleration via speculative decoding. In this paper, we propose a simple draft model training framework for direct alignment to chat-capable target models. With the proposed framework, we train Llama 2 Chat Drafter 115M, a draft model for Llama 2 Chat 7B or larger, with only 1.64% of the original size. Our training framework only consists of pretraining, distillation dataset generation, and finetuning with knowledge distillation, with no additional alignment procedure. For the finetuning step, we use instruction-response pairs generated by target model for distillation in plausible data distribution, and propose a new Total Variation Distance++ (TVD++) loss that incorporates variance reduction techniques inspired from the policy gradient method in reinforcement learning. Our empirical results show that Llama 2 Chat Drafter 115M with speculative decoding achieves up to 2.3 block efficiency and $2.4\times$ speed-up relative to autoregressive decoding on various tasks with no further task-specific fine-tuning.

1 INTRODUCTION

Large language models (LLMs) have become universal and versatile tools in our daily life thanks to their impressive results and superior performance in a wide range of domains (Achiam et al., 2023; Anil et al., 2023; Roziere et al., 2023). There has been increasing demands for running edge LLMs directly on user devices for numerous reasons such as privacy, security, cost reduction, reliability, and personalization. However, LLMs naturally generates tokens in an auto-regressive manner which is an inherently slow process due to memory bandwidth bottleneck. Speculative decoding (SD) has been proposed to mitigate the inference speed bottleneck and accelerate LLM inference by introducing a smaller draft model to predict the output of the large target model(Leviathan et al., 2023; Chen et al., 2023). In each iteration of SD, the draft model generates a sequence of tokens and the target model accepts a sub-sequence of the draft tokens using a modified rejection sampling criteria. It has been shown that SD can provide up to $2\text{-}3\times$ speedup in LLM inference without any loss in text generation quality.

Despite the promising performance of SD, high-quality small language models that can serve as draft models are often unavailable, since the smallest members of modern open-source LLM families, such as Llama 2 (Touvron et al., 2023), Falcon (Penedo et al., 2023), and Baichuan (Baichuan, 2023) are at the scale of several billions of parameters. Models with this scale are either considered as target models on edge devices or too large as draft models to get meaningful inference speed improvement with SD. Enabling SD with these large models on small devices require a much smaller draft model which is well aligned with the target model. When it comes to training a draft model for a given target model, a practical challenge is that the original dataset used for training the target model may not be provided that prohibits following the same training procedure of the target model as a natural way of mimicking the target model behavior. Recent works for draft model training

---

*Correspondence to: {raghgoel, mingul, clott}@qualcomm.qti.com

has taken inspiration from knowledge distillation (Hinton et al., 2015) where student models were trained using token-level distillation (Kim & Rush, 2016) which was extended to distribution-level distillation (Zhou et al., 2023; Agarwal et al., 2023; Wen et al., 2023; Lin et al., 2020) when the entire teacher model distributions are accessible. These methods use either KL-divergence (forward, backward or Jenson-Shannon) or total variation distance for training the student model, focusing only on improving student model generation instead of improving SD performance (Agarwal et al., 2023; Wen et al., 2023).

In this work, we propose a framework for draft model training for speculative decoding. Our training pipeline consists of three phases : 1) pre-training; training a randomly initialized draft model on a large open source text corpus to gain the general language modeling capabilities. 2) generation of a distillation dataset by having the target model generate responses to various instructions. Since the goal of the draft model is to mimic the target model we first generate the plausible data distribution using open source instruction fine-tuning datasets. 3) Fine-tuning on the distillation dataset for aligning draft and target model behaviors with target model in the loop. In addition, we propose a novel distillation loss, Total Variation Distance++ (TVD++) loss, by making connections between the acceptance of drafts in SD and policy gradient method in reinforcement learning.

With this framework, we train a draft model for Llama 2-Chat-7B model (Touvron et al., 2023) of size 115M which is only 1.64% of the size of the target model, Llama 2-Chat-Drafter-115M. We evaluate our draft model on various tasks like open-ended generation on Databricks-dolly-15k dataset (Conover et al., 2023) and text summarization on XSum (Narayan et al., 2018), CNN/DailyMail datasets (Nallapati et al., 2016). We show that our trained model can obtain up to 2.4× speed-up with speculative decoding over auto-regressive inference demonstrating the effectiveness of our training pipeline. Additionally, our proposed loss outperforms commonly used distillation losses: KLD and TVD.

2 METHOD

Our draft model training pipeline involves three steps with the goal of aligning the output of draft model with that of the target language model in the context of speculative decoding process, as opposed to having the draft model as a standalone language model.

2.1 DRAFT MODEL PRETRAINING

A draft model is required to have the same tokenizer as the corresponding target model with the same or a subset of vocabulary. Some language model families sharing the same tokenizer have wide ranges of model sizes where we can pick draft models at a desired sizes, e.g., Pythia (?). However, this is not the case more often, e.g., Llama 2 (Touvron et al., 2023), where the smallest is still too large. In such cases, we train a draft language model from scratch with the regular next token prediction loss in order to have a good language model to be aligned with the target model more effectively. It is possible to incorporate initialization techniques such as Weight Subcloning (?) to expedite this step. While it may be trivial, in our empirical observations, pretrained draft models show significantly better alignment to target model, compared to those without pretraining.

2.2 DISTILLATION DATASET GENERATION

In language modeling, there are virtually infinite number of possible input data since the same input token can produce different outputs according to the conditions made by the specific contexts, i.e., different prompts and past generations, in the causal attention mechanism. To mimic the target behavior in realistic contexts, it is crucial to have data samples that are plausible in the target model generation. While one convenient way is follow the training procedure of the target model including the same instruction and alignment datasets, availability of such datasets and reward models are not always guaranteed. To mitigate such challenges and simplify the training procedure, we generate a *distillation dataset* by using seed instructions from publicly available datasets and letting the target model generate a diverse set of responses in various configuration, e.g., temperature, top-p, and system prompts. Note that unlike Zhou et al. (2023); Agarwal et al. (2023), we only use target model for generating responses as opposed to using the pre-trained draft model. This step can be seen as a data-level distillation.

2.3 Draft model finetuning via knowledge distillation

We then finetune the pretrained draft model using dataset generated as in 2.2. Knowledge distillation comes in two different flavors (a) black-box distillation (Kim & Rush, 2016) where the target model is not accessible and only the token generated by target model can be used for distillation, and (b) white-box distillation, where we have access to the entire target output distribution. White-box optimization provides much stronger learning signals given the loss is optimized over distribution as opposed to a single token label, which corresponds to our case.

The white-box distillation can be thought as a distribution matching problem, where the target model ($\mathcal{M}_q$) output distribution ($q(y|x)$) for a given input text (x) is the desired distribution for the draft model ($\mathcal{M}_p(\theta)$) output distribution ($p_\theta(y|x)$). The training objective can be defined using a distance metric between distributions such as Kullback–Leibler Divergence (KLD) and its variants: backward KLD, Jenson-Shannon Divergence (Agarwal et al., 2023), or Total Variation Distance (TVD). Theoretical analysis based on SD (*Corollary 3.6* in (Leviathan et al., 2023)) has shown that minimizing TVD is equivalent to maximizing acceptance-rate, the true objective of improving SD performance. In this paper, we further build on TVD using techniques from reinforcement learning. We first show that the gradient step taken when optimizing TVD loss is equivalent to the policy-gradient step for reward maximization in reinforcement learning. Based on this, we can bring tools from reinforcement learning such as variance reduction techniques for improving knowledge distillation loss in LLMs (Korbak et al., 2022).

Lemma 1. *The gradient of total variation distance* $\mathrm{TVD}(p_\theta, q)$ *between the draft model* p_θ *and the target model* q *w.r.t. the draft model parameter* θ *is equal to* $\nabla_\theta \mathrm{TVD}(p_\theta, q) = \mathbb{E}_{X \sim p_\theta}[\nabla_\theta \log p_\theta(X)(-r(X))]$ *for* $r(x) := \mathbb{I}\{q(x) > p_\theta(x)\}$, *where* $\mathbb{I}\{q(x) > p_\theta(x)\}$ *is an indicator function, which is equal to* 1 *if* $q(x) > p_\theta(x)$, *or* 0, *otherwise.*

Proof. See the proof in Appendix A.1. □

Using Lemma 1, we can use variance reduction techniques from reinforcement learning (Schulman et al., 2015) to formulate a new loss. Specifically, for our purpose, we use the advantage normalization which normalizes the reward and propose *Total Variation Distance++ (TVD++)*. The normalization leads to negative rewards as opposed to 0 reward which we believe could lead to better learning signals for draft model fine-tuning. The stochastic gradient of TVD++ for a sample of n tokens is given as:

$$\nabla_\theta \mathrm{TVD}^{++}(p_\theta, q) = \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \log p_\theta(x_i) \left(\frac{r(x_i) - \mu}{\sigma} \right), \tag{1}$$

where $\mu = \frac{1}{n} \sum_{i=1}^{n} r(x_i)$ and $\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (r(x_i) - \mu)^2$ are the mean and variance for the n samples (tokens from vocabulary) for r in **Lemma 1**. Note that we use the entire distribution of target, and the mean, variance are computed over the input sequences and the entire vocabulary.

3 Experiments

Draft Model Configuration. We design a draft model by using smaller number of layers and hidden intermediate dimensions than those of Llama 2 7B Chat, our target model, based on the same network architecture (See Appendix A.2 for detailed configurations.). The size of our draft model is **115M**, which is only **1.64%** than the size of the target model **7B** to achieve negligible inference overheads even on edge device, e.g., mobile phones or laptops.

Training Datasets. Since Llama 2 pretraining dataset is not available, our draft model is pretrained on a 600B-token English language dataset that is curated from publicly available datasets. using the next token prediction objective.

For generating the *distillation dataset*, we take seed instructions from **OIG-small-chip2**(Nguyen et al.) and **OpenAssistant** (Köpf et al., 2023) to collect the target model responses sampled with various temperatures $\{0, 0.3, 0.7, 1.0\}$ and top-$p = 0.95$ for sample diversity (temperature=0 corresponds to greedy decoding).

Draft fine-tuning. To compare the effectiveness of our *TVD++*, we use different loss functions to fine-tune the draft output distribution to align with target output distribution: (i) **KLD** (ii) **TVD** and
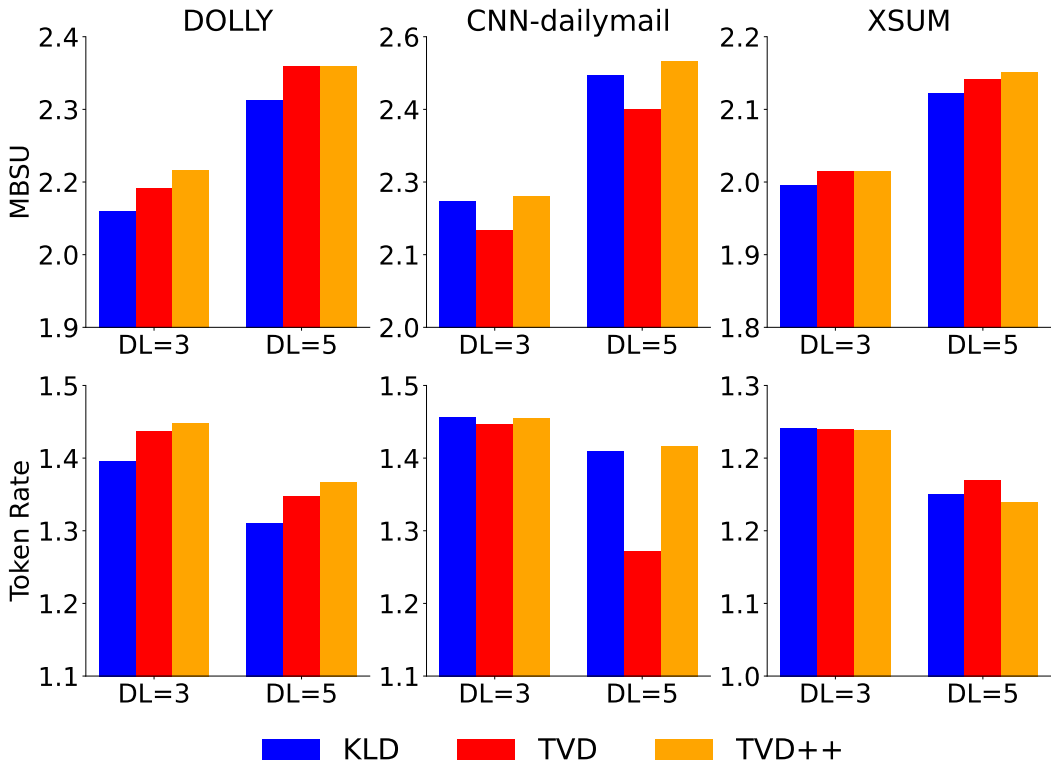
Figure 1: Draft models evaluated on Memory-Bound Speed-Up (MBSU) and token-rate (realtive to auto-regressive generation) for multiple tasks (Dolly, CNN-dailymail, XSum) with draft lengths (3, 5) and training losses (KLD, TVD, TVD++);

(iii) **TVD++**. In this step, we mix the distillation dataset with the pretraining dataset at 9:1 ratio in each training batch for regularization effect.

Evaluation metrics and tasks. We evaluate our draft model by measuring the (1) **block efficiency** (τ): average number of tokens generated per block (or target model run), for a block of size γ and input x, the maximum value of $\tau(x)$ can be $\gamma + 1$, (2) memory-bound speed-up (**MBSU**): hypothetical speed-up achieved by SD for a given block efficiency $\tau(x)$ and a relative latency c defined as ratio between number of parameters of draft to target model, i.e., $\text{MBSU}(x) := \frac{c\tau(x)}{c\gamma+1}$, and (3)**Token Rate**: number of tokens generated per second, in our case we measure the ratio of the SPD token-rate to auto-regressive token-rate, a ratio > 1 implies higher number of tokens were generated per second for SPD. We measure these metrics on a variety of tasks using different block lengths γ in $\{3, 5\}$. The evaluation tasks include: (a) open-ended text generation (Conover et al., 2023, databricks-**Dolly**-15k), (b) extreme summarization (Narayan et al., 2018, **XSum**), and (c) news summarization (Nallapati et al., 2016, **CNN-dailymail**). For open-ended text generation we random-sample with **temperature**= 0.6 and **top-**p= 0.9 for both draft model and target model, while we greedy-sample for the rest.

Results. Figure 1 shows MBSU for draft models fine-tuned using different losses, with $\gamma = 3$ or 5. Our proposed TVD++ loss either outperforms the other two losses or performs on par with the best on all the tasks, showing its effectiveness for improving draft model fine-tuning. Furthermore, Figure 2 shows the improvement in block-efficiency with $\gamma = 3$ across different training checkpoints during fine-tuning stage. For open-ended text generation task (Dolly), we observe block efficiency increases by around $\sim 21\%$ with more fine-tuning over the base draft model, while block efficiency improvement is also observed for news summarization task (CNN-dailymail) from 2.29 for base draft to 2.4 for fine-tuned draft showing the efficacy of fine-tuning.
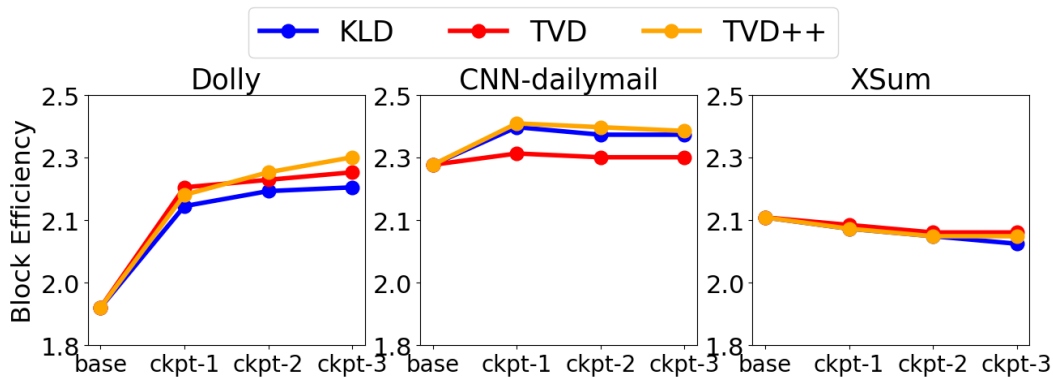
Figure 2: Draft models evaluated on block efficiency ($\gamma = 3$) over different checkpoints (ckpt) across fine-tuning stage, showing improvement over the base draft model, for multiple tasks (Dolly, CNN-DM, XSum) and training losses (KLD, TVD, TVD++)

4 CONCLUSION

In this work, we proposed a framework to train a draft model with direct alignment to a finetuned target LLM for speculative decoding. Our training pipeline consists of pre-training, generating a distillation dataset from the target LLM and fine-tuning the draft model on the distillation dataset with target LLM in the training loop. In addition, we further proposed a new loss building on TVD inspired from policy gradient in RL which outperforms both KLD and TVD in fine-tuning.

REFERENCES

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

Rishabh Agarwal, Nino Vieillard, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. Gkd: Generalized knowledge distillation for auto-regressive sequence models. *arXiv preprint arXiv:2306.13649*, 2023.

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.

Baichuan. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023. URL https://arxiv.org/abs/2309.10305.

Ond rej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pp. 272–307, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W18-6401.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

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

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations–democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*, 2023.

Tomasz Korbak, Hady Elsahar, Germán Kruszewski, and Marc Dymetman. On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting. *Advances in Neural Information Processing Systems*, 35:16203–16220, 2022.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. *arXiv preprint arXiv:2009.07253*, 2020.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

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. *arXiv preprint arXiv:1808.08745*, 2018.

Huu Nguyen, Sameer Suri, Tsui Ken, Shahules786, Together.xyz team, and Schuhmann Christoph. OIG-small-chip2. `https://huggingface.co/datasets/0-hero/OIG-small-chip2`.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023. URL `https://arxiv.org/abs/2306.01116`.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. f-divergence minimization for sequence-level knowledge distillation. *arXiv preprint arXiv:2307.15190*, 2023.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. Distillspec: Improving speculative decoding via knowledge distillation. *arXiv preprint arXiv:2310.08461*, 2023.

A   APPENDIX

A.1   PROOF OF LEMMA 1

Lemma 1. *The gradient of total variation distance* $\mathrm{TVD}(p_\theta, q)$ *between the draft model* p_θ *and the target model* q *w.r.t. the draft model parameter* θ *is equal to*

$$\nabla_\theta \mathrm{TVD}(p_\theta, q) = \mathbb{E}_{X \sim p_\theta} \left[\nabla_\theta \log p_\theta(X)(-r(X)) \right]$$

for $r(x) := \mathbb{I}\{q(x) > p_\theta(x)\}$, *where* $\mathbb{I}\{q(x) > p_\theta(x)\}$ *is an indicator function, which is equal to* 1 *if* $q(x) > p_\theta(x)$, *or* 0, *otherwise.*

Proof. Let the draft and target model output distributions be defined as $p_\theta(x)$ and $q(x)$, respectively, where θ is the trainable draft model parameter and $x \in \{1, \ldots, |\mathcal{V}|\}$, where $\mathcal{V}$ is a set of tokens, i.e., vocabulary. We ignore the conditioning on input prompt here for ease of understanding. From Leviathan et al. (2023), TVD can be represented as

$$\mathrm{TVD}(p_\theta, q) = \frac{1}{2} \sum_x |q(x) - p_\theta(x)| = 1 - \sum_x \min(p_\theta(x), q(x)). \tag{2}$$

Then, one can derive the gradient of RHS in equation 2 w.r.t. θ as

$$\nabla_\theta \mathrm{TVD}(p_\theta, q) = \nabla_\theta \big(1 - \sum_x \min(p_\theta(x), q(x))\big)$$

$$= -\sum_x \mathbb{I}\{q(x) > p_\theta(x)\} \nabla_\theta p_\theta(x)$$

$$= \sum_x p_\theta(x)(-\mathbb{I}\{q(x) > p_\theta(x)\}) \nabla_\theta \log p_\theta(x) \tag{3}$$

$$= \mathbb{E}_{x \sim p_\theta(x)} \left[\nabla_\theta \log p_\theta(x)(-\mathbb{I}\{q(x) > p_\theta(x)\}) \right], \tag{4}$$

where equation 3 follows from using the log-derivative trick. Note that is equivalent to the policy gradient where a policy and a reward are $p_\theta(x)$ and $r(x) := \mathbb{I}\{q(x) > p_\theta(x)\}$, respectively (Williams, 1992). □

A.2   MODEL CONFIGURATIONS

The following configurations are used for our target and draft models:

Table 1: Draft and target model configurations

	Llama 2-Chat (7B, target)	Llama 2-Chat-Drafter (115M, draft)
Layers	32	4
Attention heads	32	8
Intermediate dim	11,008	2,816
Hidden dim	2,048	1,024
Activation	SiLU	SiLU

A.3   TRAINING HYPER-PARAMETERS

For draft model pretraining, we used deepspeed stage1 with a batch-size=496 on 32 A100 GPUs. Additionally, during pre-training a large batch-size can be used as compared to distillation which requires the target model weight and output consuming a lot of memory. The optimizer used is *AdamW* with *WarmUpDecayLR* learning-rate scheduler, maximum learning rate was 0.0001 while minimum was $1e-6$, with 5000 warm-up steps

For draft model fine-tuning, we used deepspeed stage 3 with batch-size=40 on 8 A-100 GPUs with maximum learning-rate=0.0003 with same optimizer and scheduler with 2000 warm-up steps.

A.4 Data processing

We preprocess the text data with the tokenizer of the target language model while appending End-Of-Sentence (EOS) token at the end of each input sequence. Furthermore, as a postporcessing step, all the sequences are concatenated into chunks of 2048 length, to maximize training throughput without adding pad tokens.

A.5 Performance Degradation at Out-Of-Distribution (OOD) Task

We also evaluated the block efficiency of multiple draft models on translation task, WMT18-DeEn (Bojar et al., 2018), from German to English. We found that all fine-tuned draft models were outperformed by the base model for this particular task since those models were *not* directly fine-tuned on the translation task, making the task OOD. We expect this issue will be resolved by adding in-distribution samples for training.
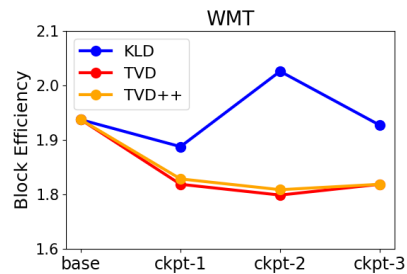

Figure 3: Block efficiency for WMT18-DeEn results with multiple draft models are described.