Plug and Play Knowledge Distillation for kNN-LM with External Logits

Xuyang Jin[1*] **Tao Ge**[2†] **Furu Wei**[2]
[1] Tsinghua University
[2] Microsoft Research Asia
jinxy17@mails.tsinghua.edu.cn
{tage,fuwei}@microsoft.com

Abstract

Despite the promising evaluation results by knowledge distillation (KD) in natural language understanding (NLU) and sequence-to-sequence (seq2seq) tasks, KD for causal language modeling (LM) remains a challenge. In this paper, we present a novel perspective of knowledge distillation by proposing plug and play knowledge distillation (PP-KD) to improve a (student) kNN-LM that is the state-of-the-art in causal language modeling by leveraging external logits from either a powerful or a heterogeneous (teacher) LM. Unlike conventional logit-based KD where the teacher's knowledge is built-in during training, PP-KD is plug and play: it stores the teacher's knowledge (i.e., logits) externally and uses the teacher's logits of the retrieved k-nearest neighbors during kNN-LM inference at test time. In contrast to marginal perplexity improvement by logit-based KD in conventional neural (causal) LM, PP-KD achieves a significant improvement, enhancing the kNN-LMs in multiple language modeling datasets, showing a novel and promising perspective for causal LM distillation.

1 Introduction

The effectiveness of knowledge distillation (KD) has been extensively validated in Natural Language Processing (NLP) along with various distilled models (Sanh et al., 2019; Jiao et al., 2019; Wang et al., 2020) as well as emerging KD approaches (Xu et al., 2020; Pan et al., 2020). For causal language modeling, however, it is so rare to see a success of KD as it is in natural language understanding (NLU) and sequence-to-sequence (seq2seq) tasks; even the versatile logit-based KD (Hinton et al., 2015), which appears to work in almost any KD scenario with any model architecture with state-of-the-art results (Zhao et al., 2022), still does not show a substantial improvement in the metrics of causal language modeling itself (e.g., perplexity) although it may benefit downstream task fine-tuning for a causal LM (West et al., 2021).

With the motivation to advance the performance boundary, we study the k-nearest neighbor language model (kNN-LM) (Khandelwal et al., 2020) which is the state-of-art in causal language modeling, and propose plug and play knowledge distillation (PP-KD) to enhance its result, especially for the small-size model, by leveraging kNN logits from a teacher LM. Unlike conventional logit-based KD where the teacher's knowledge is built-in by training the student with an auxiliary loss to fit the teacher's logits, PP-KD stores the teacher's logits externally and uses them only at test time; thus it is plug and play.

As Figure 1 shows, PP-KD works during inference to enhance kNN results. Compared with the vanilla kNN-LM, it is required to additionally store the teacher's logits besides context representations and targets of training examples. After retrieving the k nearest neighbors (i.e., training contexts), we get both of their corresponding targets and logits from the datastore and aggregate them into the kNN prediction. As PP-KD is plug and play, we can easily enable/disable it by keeping/removing the effects of logits (in the red dashed boxes in Figure 1) on kNN prediction during inference; moreover, we can flexibly switch the teacher we want to employ simply by using its logits instead without retraining like conventional KD.

We study PP-KD with two kinds of teachers: one is a more powerful causal LM; the other is a heterogeneous masked LM (Devlin et al., 2018). Extensive experiments in Wiki-103 and BookCorpus demonstrate that PP-KD can significantly benefit causal language modeling, and that a stronger teacher or a teacher ensemble by a causal LM and a masked LM can further improve the perplexity.

The contribution of this paper is twofold:

[*] This work was initiated during the first author's internship at MSR Asia.
[†] Corresponding author

Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 463–469
November 20–23, 2022.

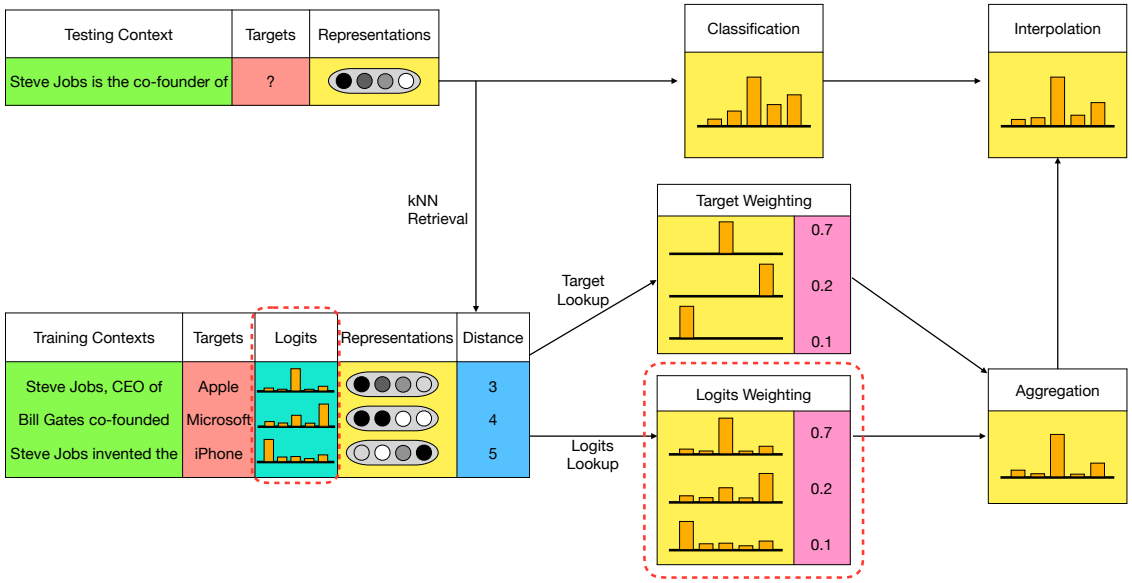

Figure 1: Overview of PP-KD for the kNN-LM. The red dashed boxes indicate the external plug and play logits in PP-KD for enhancing kNN results at test time, which can be flexibly enabled/disabled or replaced with more informative logits by a more powerful teacher. Please note that only logits are from the teacher model; the context representation for kNN search is still by the student model.

- We propose an effective knowledge distillation approach – PP-KD that can significantly improve causal language modeling, especially for small-size models.
- The proposed PP-KD demonstrates a novel perspective on knowledge distillation with many promising results analog to conventional "built-in" KD approaches.

2 Plug and Play Knowledge Distillation for kNN-LM

2.1 Basic Concept

Unlike the vanilla kNN-LM (Khandelwal et al., 2020) that builds its datastore $(\mathcal{K}, \mathcal{V})$ using the hard-label targets, PP-KD additionally builds a datastore $\mathcal{U}$ for corresponding logits from the teacher model.

Formally, PP-KD needs to build datastore $(\mathcal{K}, \mathcal{V}, \mathcal{U})$ that stores context vectors, hard targets and logits from the teacher respectively. $(\mathcal{K}, \mathcal{V})$ are used in the same way as in the vanilla kNN-LM:

$$P_{\text{hard-}k\text{NN}}(w^*|c^*) \propto \sum_{(\boldsymbol{c},w)\in\mathcal{N}} \mathbb{1}_{w=w^*} \exp\frac{-d(\boldsymbol{c}^*,\boldsymbol{c})}{T} \quad (1)$$

where $\mathcal{N} \subseteq (\mathcal{K}, \mathcal{V})$ is the set of retrieved nearest contexts $\boldsymbol{c}$ with hard targets w by querying with $\boldsymbol{c}^*$, $d(\boldsymbol{c}^*, \boldsymbol{c})$ is the distance[1] of $\boldsymbol{c}^*$ and $\boldsymbol{c}$, and T is the temperature in softmax.

After retrieving the kNN training contexts, we get their corresponding logits from $\mathcal{U}$:

$$P_{\text{logit-}k\text{NN}}(w^*|c^*) \propto \sum_{(\boldsymbol{c},\boldsymbol{u})\in\widetilde{\mathcal{N}}} \boldsymbol{u} \times \exp\frac{-d(\boldsymbol{c}^*,\boldsymbol{c})}{T} \quad (2)$$

where $\widetilde{\mathcal{N}} = \{(\boldsymbol{c}, \boldsymbol{u})|(\boldsymbol{c}, \cdot) \in \mathcal{N}\} \subseteq (\mathcal{K}, \mathcal{U})$, $\boldsymbol{u} \in \mathbb{R}^{|V|}$ is the teacher's prediction logits given context $\boldsymbol{c}$.

The final kNN prediction is linearly aggregated from $P_{\text{hard-}k\text{NN}}$ and $P_{\text{logit-}k\text{NN}}$:

$$P_{k\text{NN}}(\cdot) = \mu P_{\text{hard-}k\text{NN}}(\cdot) + (1-\mu) P_{\text{logit-}k\text{NN}}(\cdot) \quad (3)$$

$P_{k\text{NN}}$ will be then linearly interpolated with the backbone neural LM's prediction P_{LM}:

$$P(\cdot) = \lambda P_{k\text{NN}}(\cdot) + (1-\lambda) P_{LM}(\cdot) \quad (4)$$

After the datastore $(\mathcal{K}, \mathcal{V}, \mathcal{U})$ are all built offline in advance, we can perform PP-KD that is plug and play during inference: if we want to disable it, then we can just skip Eq (2) and set μ in Eq (3) to 1.0, which will degrade into the vanilla kNN-LM; if we want to switch the teacher, we can simply replace $\mathcal{U}$ storing the original teacher's logits with $\mathcal{U}'$ that stores the new teacher's logits.

2.2 Logits: Homogeneous *VS* Heterogeneous

The most straightforward way to generate logits is using a powerful homogeneous (i.e., causal) LM

[1] As the previous work, context is represented by the Transformer's last layer's FFN input states, and distances between contexts are the FAISS (Johnson et al., 2019) squared L^2 distances.

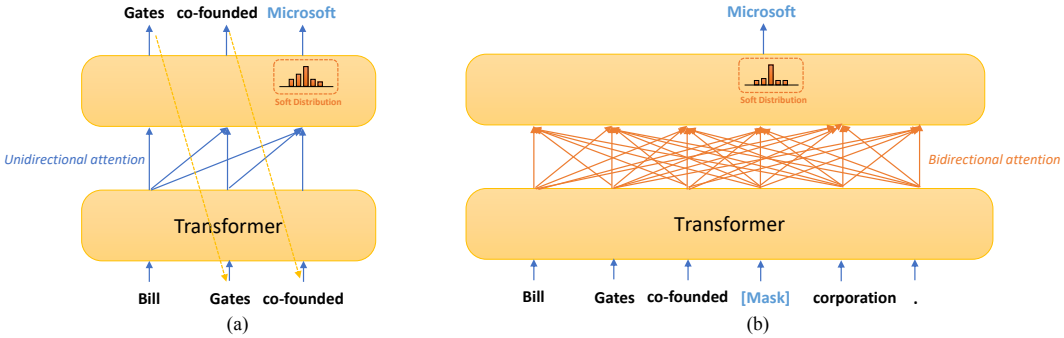

Figure 2: The comparison of logits by **(a)** a causal LM and **(b)** a masked LM.

to generate the probability distribution over the vocabulary for building the datastore $\mathcal{U}$, as shown in Figure 2(a).

In addition to homogeneous LMs that have similar perspectives for causal language modeling, we propose to use a heterogeneous LM – a masked LM – to generate logits from a different view. As Figure 2(b) shows, for generating the logits for the target "*Microsoft*" in the training example to build datastore $\mathcal{U}$, a masked LM will use both its leftward and rightward context.

Please note it is valid to use the masked LM's logits in the datastore $\mathcal{U}$ because using masked LM's logits of training examples **DOES NOT** leak rightward context information of test examples during inference (see Appendix A for more details).

3 Experiments

3.1 Experimental Setting

Data Following Khandelwal et al. (2020), we use the well-known language modeling benchmark – WIKITEXT-103 (Merity et al., 2017) which is a subset of English Wikipedia, consisting of 28K selected Wikipedia articles. We follow the original train/validation/test split of WIKITEXT-103 which contains 103M, 250K and 250K tokens respectively, and use word-level perplexity as our evaluation metric.

Model We mainly test PP-KD on the most popular GPT-style (Radford et al., 2018) architecture which is a decoder-only Transformer with 3 different sizes shown in Appendix B. We tune the hyperparameters T, μ and λ in Eq (1-4) on the validation set.

Datastore, Indexing and kNN search We first create the $(\mathcal{K}, \mathcal{V})$ following Khandelwal et al. (2020), and their corresponding datastore $\mathcal{U}$ that stores the teacher's logits. We then build FAISS index using 1M randomly sampled keys (quantized to 64 bytes) to learn 4K cluster centroids. During inference, we look up 32 cluster centroids for the 1K nearest neighbors.

Baselines As there is little work studying KD for causal language modeling, as Li et al. (2021) notes, we mainly compare PP-KD with the kNN-LM trained from scratch as well as the conventional logit-based KD approach which is adopted by the most famous distilled causal LM – DistilGPT-2[2].

As we see causal language modeling as an end task in this paper, we use perplexity as the metric for evaluation. The details of model architecture, training, evaluation and datastores are shown in Appendix B.

3.2 Results

Table 1 shows the results of PP-KD for LMs of different sizes. In contrast to conventional KD that has little improvement in perplexity over the model trained from scratch as observed by Rajbhandari et al. (2022), PP-KD with a powerful teacher can significantly improve the perplexity, and a more powerful teacher tends to result in a larger improvement, which is a very rare success in KD for causal language modeling. Interestingly, even if we use the teacher with the same size as the student, we can still observe an improvement, which aligns well with previous work's observation regarding self distillation (Furlanello et al., 2018).

After confirming that PP-KD can effectively improve causal language modeling with a powerful homogeneous (i.e., causal LM) teacher, we study whether a heterogeneous (i.e., masked LM) teacher can be used for PP-KD, and show the results in Table 2. Surprisingly, the heterogeneous logits whose perspective is different from causal LM can also

[2]https://huggingface.co/distilgpt2

Size	Model	Perplexity From scratch (no teacher)	Perplexity (KD) Small	Mid	Large	Perplexity (PP-KD) Small	Mid	Large
Small (4L-384-6H)	LM	35.24	35.31	35.15	35.04	-	-	-
	kNN-LM	28.50	28.75	28.54	28.33	**27.47***	**26.32***	**25.74***
Mid (6L-768-8H)	LM	28.55	-	28.40	28.33	-	-	-
	kNN-LM	23.76	-	23.75	23.61	-	**23.25***	**22.72***
Large (12L-768-12H)	LM	25.25	-	-	25.35	-	-	-
	kNN-LM	21.76	-	-	21.79	-	-	**21.37***

Table 1: Results of PP-KD for models of various sizes with different causal LM teachers (we do not use a teacher that is smaller than the student for distillation). * denotes the result of PP-KD significantly ($p < 0.05$) outperforms the corresponding kNN-LM trained from scratch and via conventional logit-based KD. aL-b-cH denotes the model has a layers with dimension of b and c heads.

Model (large-size)	Perplexity
LM	25.25
kNN-LM	21.76
PP-KD (homogeneous logits)	21.37
PP-KD (heterogeneous logits)	21.02
PP-KD (mixed logits)	**20.83**

Table 2: Perplexity of the large-size kNN-LM distilled with logits by the large-size homogeneous (causal) and heterogeneous (masked) teachers. Mixed refer to averaging homogeneous and heterogeneous logits.

Model	Cross-entropy
causal LM	4.58
masked LM	1.81

Table 3: Cross-entropy of the large-size causal and masked LM on training examples.

benefit PP-KD, and is even marginally better than the homogeneous teacher. The reason we suppose is that the heterogeneous logits are more informative (reflected by much lower cross entropy as shown in Table 3) owing to the bi-directional attention that can access the rightward context in the retrieved training example. Moreover, we mix the homogeneous and heterogeneous logits by simply averaging them, and observe that the mixed logits can even further improve the result. We suspect this is because the mixed logits play a similar role as teacher ensemble which can benefit results, as widely confirmed by previous KD literature.

We then verify PP-KD with mixed logits by a larger teacher (Baevski and Auli, 2018) on the kNN-LM with the famous DistilGPT-2 and GPT2-small (Radford et al., 2019) architecture on both Wiki-103 and BookCorpus[3] (Zhu et al., 2015). According to Table 4, PP-KD significantly outperforms the counterparts trained from scratch, or via conventional logit KD, with negligible latency overhead, demonstrating a rare success in knowledge distillation for causal language modeling.

[3]We split the corpus with the ratio of 90/5/5 for training/validation/test.

Model	Wiki-103 PPL	BookCorpus PPL
Teacher kNN-LM	16.1	11.7
DistilGPT-2 kNN-LM	23.2	17.9
DistilGPT-2 kNN-LM (KD)	23.1	17.3
DistilGPT-2 kNN-LM (PP-KD)	**21.9**	**16.1**
GPT2-small kNN-LM	21.8	15.9
GPT2-small kNN-LM (KD)	21.6	15.6
GPT2-small kNN-LM (PP-KD)	**20.9**	**14.9**

Table 4: A comparison between the kNN-LMs with PP-KD (mixed logits) and those trained from scratch and with conventional logit KD. The teacher's architecture adopted is Baevski and Auli (2018). The configuration details of the teacher, DistilGPT-2 and GPT2-small are presented in Appendix B. We follow Baevski and Auli (2018) to use adaptive input and softmax specially for Wiki-103 to handle the large vocabulary; while for BookCorpus, we use the same BPE and vocabulary as GPT2. PP-KD clearly outperforms the counterparts trained from scratch or via KD for both the DistilGPT-2 and GPT2-small kNN-LMs, while it introduces negligible latency overhead compared with time-consuming kNN retrieval.

4 Conclusion and Future Work

We present PP-KD – a novel perspective for leveraging more powerful (teacher) models to improve state-of-the-art kNN-LMs for causal language modeling. Compared with conventional "built-in" KD, PP-KD leverages the teacher's logits stored externally to enhance the prediction at test time and achieves a rare success in causal LM distillation.

As a preliminary and focused study, this work shows promising results of PP-KD in language

modeling (as an end task), while it still has much room for improvement (e.g., more efficient implementation, more effective and informative logits as well as more in-depth analyses for PP-KD) and great potential to benefit downstream tasks. We leave these for future work and look forward to building a connection between PP-KD and the emerging retrieval augmented modeling in a bigger picture.

References

Alexei Baevski and Michael Auli. 2018. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. In *International Conference on Machine Learning*, pages 1607–1616. PMLR.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models.

Tianda Li, Yassir El Mesbahi, Ivan Kobyzev, Ahmad Rashid, Atif Mahmud, Nithin Anchuri, Habib Hajimolahoseini, Yang Liu, and Mehdi Rezagholizadeh. 2021. A short study on compressing decoder-based language models. *arXiv preprint arXiv:2110.08460*.

Stephen Merity, Caiming Xiong, James Bradbury, and R. Socher. 2017. Pointer sentinel mixture models. *ArXiv*, abs/1609.07843.

Haojie Pan, Chengyu Wang, Minghui Qiu, Yichang Zhang, Yaliang Li, and Jun Huang. 2020. Meta-kd: A meta knowledge distillation framework for language model compression across domains. *arXiv preprint arXiv:2012.01266*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. *arXiv preprint arXiv:2201.05596*.

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

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *arXiv preprint arXiv:2110.07178*.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. *arXiv preprint arXiv:2002.02925*.

Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. 2022. Decoupled knowledge distillation. *arXiv preprint arXiv:2203.08679*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*.

A Details of Logits

As mentioned in Section 2.2, logits generated by a masked LM do not leak information. We use Figure 2 as the example to demonstrate how logits by a masked LM are generated and used:

> *Bill Gates co-founded* ***[Microsoft]*** *Corporation.*

We build datastore ($\mathcal{K}$, $\mathcal{V}$, $\mathcal{U}$) as in Table 5. At test time, assume that we find a retrieved nearest neighbor is the 4th entry in Table 5, meaning that the representation of the context at test time $f(c_{test})$ is very similar to $f($*"Bill Gates co-founded"*$)$, we can use its corresponding logits generated by either

Context	$\mathcal{K}$	$\mathcal{V}$	$\mathcal{U}_{left}$	$\mathcal{U}_{bi}$
[BOS]	*f("[BOS]")*	*Bill*	$\boldsymbol{u_{left}}$*(Bill)*	$\boldsymbol{u_{bi}}$*(Bill)*
Bill	*f("Bill")*	*Gates*	$\boldsymbol{u_{left}}$*(Gates)*	$\boldsymbol{u_{bi}}$*(Gates)*
Bill Gates	*f("Bill Gates")*	*co-founded*	$\boldsymbol{u_{left}}$*(co-founded)*	$\boldsymbol{u_{bi}}$*(co-founded)*
Bill Gates co-founded	*f("Bill Gates co-founded")*	*Microsoft*	$\boldsymbol{u_{left}}$*(Microsoft)*	$\boldsymbol{u_{bi}}$*(Microsoft)*
Bill Gates co-founded Microsoft	*f("Bill Gates co-founded Microsoft")*	*Corporation*	$\boldsymbol{u_{left}}$*(Corporation)*	$\boldsymbol{u_{bi}}$*(Corporation)*
...	...	...	...	...

Table 5: The datastore built for the example in Figure 2 where $f(c)$ is the representation of context c computed by the backbone LM, $\boldsymbol{u_{left}}$ and $\boldsymbol{u_{bi}}$ are the logits generated for the token to be predicted by the causal LM conditioned on the leftward context and the masked LM conditioned on both the leftward and rightward context, respectively. For example, in the 4th row, $\boldsymbol{u_{left}}(\textit{Microsoft}) = P_{\text{left-to-right}}(w|\textit{Bill Gates co-founded}) \in \mathbb{R}^{|V|}$ and $\boldsymbol{u_{bi}}(\textit{Microsoft}) = P_{\text{masked}}(w|\textit{Bill Gates co-founded [MASK] Corporation.}) \in \mathbb{R}^{|V|}$.

Size	#Layer	d_{model}	d_{ffn}	h	$(\mathcal{K}, \mathcal{V})$	$\mathcal{U}$
small	4	384	1024	6	149GB	148GB
mid	6	768	1536	8	297GB	296GB
large	12	768	3072	12	297GB	296GB
DistilGPT-2	6	768	3072	12	297GB	296GB
GPT2-small	12	768	3072	12	297GB	296GB
Baevski and Auli (2018)	16	1024	4096	16	445GB	444GB

Table 6: The detailed model architecture configuration of the GPT-style language models trained on WIKI-103 in our experiments. d_{model} and d_{ffn} are the dimensions of input/output and feed-forward inner layers respectively; h denotes the number of attention heads. All the models use a shared input/output vocabulary and embeding. For masked LMs, we use the same model size configuration (e.g., the number of layers and dimensionality). The last two columns report the disk size of datastores where the size of $(\mathcal{K}, \mathcal{V})$ depends on the backbone LM's hidden size while the size of $\mathcal{U}$ depends on the hidden size of the LM for logit generation. Please note that "large" and "GPT2-small" are actually identical.

a causal LM $\boldsymbol{u_{left}}(\textit{Microsoft}) \in \mathbb{R}^{|V|}$ or a masked LM $\boldsymbol{u_{bi}}(\textit{Microsoft}) \in \mathbb{R}^{|V|}$ for PP-KD.

Therefore, it is clear that the logits by the masked LM will not leak the rightward context of the test example during inference.

B Details of Experiments

Table 6 shows the detailed model architecture information as well as the disk space cost for building datastore. Note that in practice, we save the final layer hidden representation for storing the logits, which can be simply mapped into probability distribution over the vocabulary by a linear transformation with softmax activation at a negligible time cost compared with kNN search, instead of directly saving the final probability distribution whose space cost is huge. As shown in Table 6, the datastore $\mathcal{U}$'s space cost is on par with $(\mathcal{K}, \mathcal{V})$, meaning that the PP-KD only needs twice as much space as the original kNN-LM. Moreover, as we do not perform search operations over $\mathcal{U}$ (remember that we get logits from $\mathcal{U}$ by using indices that are obtained by kNN search – see the example in Table 5 in Appendix A), we do not even have to load the whole datastore $\mathcal{U}$ into memory[4]. Therefore, given that hard disks are cheap and easy to scale, the additional space cost for $\mathcal{U}$ will not a problem in practice.

Configurations	Values
Train	
Number of epochs	100
Devices	8 Nvidia V100 GPU
Max tokens per GPU	3,072
Optimizer	Nesterov Accelerated Gradient momentum = 0.99
Learning rate	1e-5, 5e-5, 1e-4, 3e-4
Learning rate scheduler	cosine
Warmup	16,000
Evaluation	
Maximum Context Length	512 tokens

Table 7: Detailed configuration for training and evaluation.

Table 7 elaborates the hyperparamters for training and evaluating models in Table 7. For the hyperparamters T, μ and λ, we tune them on the validation set. Specifically, for the vanilla kNN-LM, the best configurations are: $T = 10, \lambda = 0.25$; for the PP-KD, the best configurations are: $T = 10, \mu =$

[4]For example, we can split the datastore $\mathcal{U}$ into many small file pieces offline in advance. During inference, we only load the small pieces that cover the indices.

Context	**Target**	P_{LM}	$P_{\text{hard-}k\text{NN}}$	$P_{\text{logit-}k\text{NN}}$	**#Hit(hard)**
Homarus gammarus, know as the European lobster, is a	spieces	0.176	0.025	0.137	33
This may occur several times a year for young lobsters, but decreases to once every	1	0.002	0.003	0.125	6
The two species can be distinguished by	a	0.046	0.031	0.121	84
Served as Officer Commanding North - Western Area in 1946, and as	Director	0.017	0.013	0.114	33
Air Vice Marshal Frank Headlam, CB, CBE (15	July	0.048	0.036	0.093	78
He took over as Air Officer Commanding (AOC) OPCOM from Air Vice Marshal	Val	0.000	0.002	0.199	8

Table 8: The cases that logits help improve perplexity. $P_{\text{hard-}k\text{NN}}$ and $P_{\text{logit-}k\text{NN}}$ refer to Eq (1) and Eq (2) respectively. **#Hit(hard)** denotes the number of neighbors whose targets are correct among the the retrieved k ($k = 1024$) nearest neighbors.

$0.4, \lambda = 0.5$.

Finally, we present more concrete examples in Table 8 where hard-label kNN-LM cannot perform well but the PP-KD works well. For these examples, $P_{\text{hard-}k\text{NN}}$ for the correct target is either almost equivalent or even lower than the backbone LM's probability P_{LM} because very few retrieved neighbors' targets are the correct one. However, the PP-KD addresses this problem by fully utilizing the logits information, substantially promoting the correct targets.