Revisiting k-NN for Fine-tuning Pre-trained Language Models

Lei Li[1,2], **Jing Chen**[1,2], **Botzhong Tian**[1,2], **Ningyu Zhang**[1,2]*
[1]Zhejiang University & AZFT Joint Lab for Knowledge Engine, China
{leili21, chenjing_1984, tbozhong, zhangningyu}@zju.edu.cn

Abstract

Pre-trained Language Models (PLMs), as parametric-based *eager learners*, have become the de-facto choice for current paradigms of Natural Language Processing (NLP). In contrast, k-Nearest-Neighbor (k-NN) classifiers, as the *lazy learning* paradigm, tend to mitigate over-fitting and isolated noise. In this paper, we revisit k-NN classifiers for augmenting the PLMs-based classifiers. From the methodological level, we propose to adopt k-NN with textual representations of PLMs in two steps: (1) Utilize k-NN as prior knowledge to calibrate the training process. (2) Linearly interpolate the probability distribution predicted by k-NN with that of the PLMs' classifier. At the heart of our approach is the implementation of k-NN-calibrated training, which treats predicted results as indicators for easy versus hard examples during the training process. From the perspective of the diversity of application scenarios, we conduct extensive experiments on fine-tuning, prompt-tuning paradigms and zero-shot, few-shot and fully-supervised settings, respectively, across eight diverse end-tasks. We hope our exploration will encourage the community to revisit the power of classical methods for efficient NLP[1].

1 Introduction

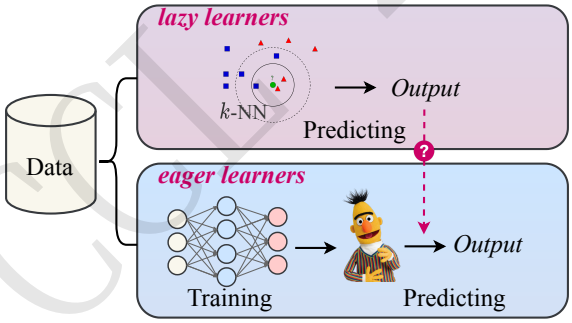

Figure 1: Revisiting how does a lazy learner (k-NN) help the eager learner (PLM).

Pre-trained Language Models (PLMs) (Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2020) have shown superior performance across a wide range of language-related downstream tasks (Kowsari et al., 2019; Nan et al., 2020). Afterward, the conventional paradigm *fine-tuning*, which extends extra task-specific classifiers on the top of PLMs, has been proposed to apply PLMs for downstream tasks. Recently, a new paradigm called prompt-tuning, which originated from GPT-3 (Brown et al., 2020), has been introduced and has shown better results for PLMs on few-shot and zero-shot tasks. Fine-tuning has proved to be effective on supervised tasks and is widely used as the standard method for natural language processing (NLP). Despite the effectiveness of adapting PLMs, parametric-based *eager learners* (Friedman, 2017), like PLMs with neural networks, require estimating the model parameters

* Corresponding Author.
[1]Code and datasets are available in https://github.com/zjunlp/Revisit-KNN.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

with an intensive learning stage. Besides, Training a large PLM model can require significant computing resources and energy, which have negative environmental consequences. As a result, there has been a growing interest in developing more efficient and sustainable methods for training and deploying PLMs.

A stark contrast to PLMs is the k-NN classifier: a simplest machine learning algorithm that does not have a training phase but simply predicts labels based on the nearest training examples instead. NLP researchers (Khandelwal et al., 2020; He et al., 2021) have found that k-NN enable excellent unconditional language modeling (Khandelwal et al., 2020; He et al., 2021) during test phrase. According the definition in (Friedman, 2017), k-NN is actually a *lazy learner* that can avoid over-fitting of parameters (Boiman et al., 2008) and effectively smooths out the impact of isolated noisy training data (Orhan, 2018). Though k-NN has the above advantages, previous works only leverage k-NN for testing, and there is no systematic examination of the full utilization of k-NN for PLMs.

To this end, we have conducted a comprehensive and in-depth empirical study of the k-NN classifier for natural language understanding (NLU). Our approach involves leveraging the predictive results of a k-NN classifier and augmenting conventional parametric PLM classifiers in two steps: (1) We explore the role of k-NN as prior knowledge for calibrating training by using k-NN results as an indicator of easy vs. hard examples in the training set; (2) During inference, we linearly interpolate probability distributions with the PLM's predicted distributions to make the final prediction; (3) We conduct extensive experiments with fine-tuning in fully-supervised, few-shot and zero-shot settings, aiming to reveal the different scenarios where k-NN is applicable. We hope this work can open up new avenues for improving NLU of PLMs via k-NN and inspire future research to reconsider the role of "old-school" methods.

2 Related Work

k-NN in the era of PLMs. The k-Nearest Neighbor (kNN) classifier is a classic non-parametric algorithm that predicts based on representation similarities. While kNN has lost some visibility compared to current deep learning approaches in recent years, it has not fallen off the radar completely. In fact, kNN has been used to enhance pre-trained language models (PLMs) in various tasks, such as unconditional language modeling (Khandelwal et al., 2020; He et al., 2021), machine translation (Khandelwal et al., 2021; Gu et al., 2018), and question answering (Kassner and Schütze, 2020). Most recently, (Alon et al., 2022; Meng et al., 2021) further respectively propose automaton-augmented and GNN-augmented retrieval to alleviate the computationally costly datastore search for language modeling. However, previous researchers (He et al., 2021; Khandelwal et al., 2021; Kassner and Schütze, 2020; Li et al., 2021; Meng et al., 2021; Alon et al., 2022; Zhang et al., 2022) mainly focus on generative tasks or adopt simple interpolation strategies to combine k-NN PLMs only at test time. (Shi et al., 2022) propose to leverage k-NN for zero-shot inference.

Revisiting k-NN for PLMs. Unlike them, we focus on empirically demonstrating that incorporating k-NN improves PLMs across a wide range of NLP tasks in fine-tuning and prompt-tuning paradigms on various settings, including the fully-supervised, few-shot and zero-shot settings. Note that our work is the first to comprehensively explore k-NN during both the training and inference process further for fruitful pairings: in addition to the approaches mentioned above, we propose to regard the distribution predicted by k-NN as the prior knowledge for calibrating training, so that the PLM will attend more to the examples misclassified by k-NN.

3 Methodology

The overall framework is presented in Figure 2. We regard the PLM as the feature extractor that transforms the input textual sequence x into an instance representation $\mathbf{x}$ with dimensions D. We revisit k-NN in §3.1 and then introduce our method to integrate k-NN with tuning paradigms in §3.2.

3.1 Nearest Neighbors Revisited

Given the training set of n labeled sentences $\{x_1, \ldots, x_n\}$ and a set of target labels $\{y_1, \ldots, y_n\}$, $y \in [1, C]$, the k-NN classifier can be illustrated in the next three parts:

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

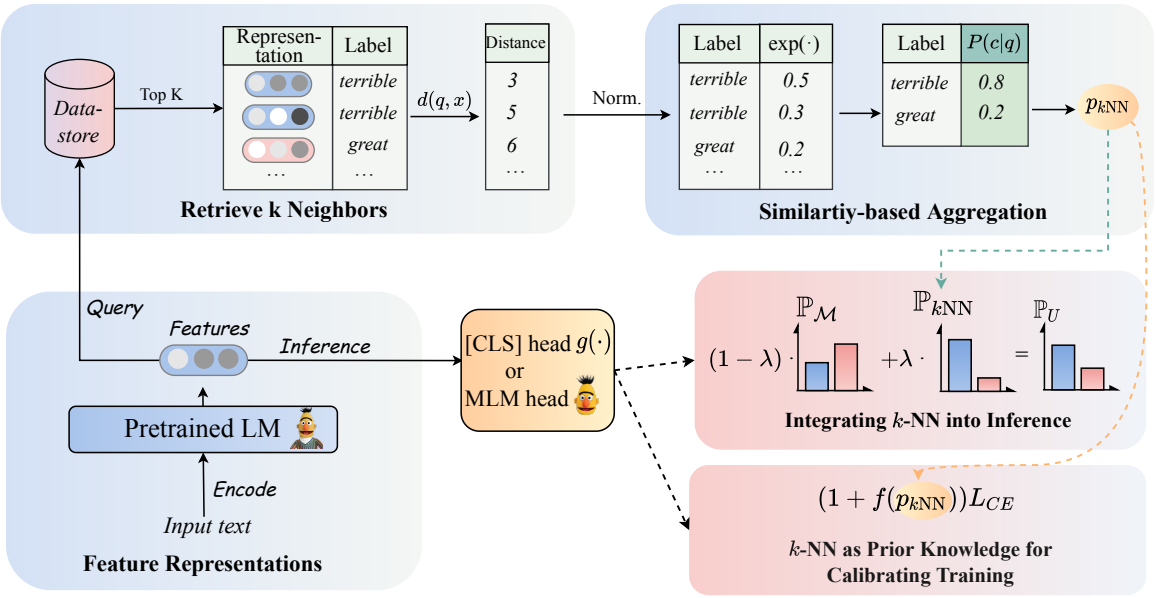

Figure 2: Overview of incorporating k-NN for PLMs

Feature Representations For k-NN, we firstly have to collect the corresponding set of features $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ from the training set. Concretely, we assign $\mathbf{x}$ with the embedding of the `[CLS]` token of the last layer of the PLM for the fine-tuning procedure. More specifically, we define the feature representations as follows:

$$\mathbf{x} = \mathbf{h}_{\texttt{[CLS]}}, \tag{1}$$

The feature representation $\mathbf{q}$ of a query example x_q also follows the above equation.

Retrieve k Neighbors Following the commonly practiced in k-NN (Friedman, 2017; Wang et al., 2019), we pre-process both $\mathbf{q}$ and features in the training set $\mathcal{D}$ with $l2$-normalization. We then compute the similarity between the query $\mathbf{q}$ and each example in $\mathcal{D}$ with Euclidean distance as : $d(\mathbf{q}, \mathbf{x}), \forall \mathbf{x} \in \mathcal{D}$, where $d(\cdot, \cdot)$ is the Euclidean distance calculation function. According to the similarity, we select the top-k representations from $\mathcal{D}$, which are the closest in the distance to $\mathbf{q}$ in the embedding space.

Similarity-based Aggregation Let $\mathcal{N}$ donate the set of retrieved top-k neighbors, and $\mathcal{N}_y$ be the subset of $\mathcal{N}$ where the whole examples have the same class y. Then the k-NN algorithm converts the top-k neighbors to $\mathbf{q}$ and the corresponding targets into a distribution over $\mathcal{C}$ labels. The probability distribution of $\mathbf{q}$ being predicted as c is:

$$p_{k\text{NN}}(c|\mathbf{q}) = \frac{\sum_{\mathbf{x} \in \mathcal{N}_y} \exp\ (-d(\mathbf{q}, \mathbf{x})/\tau)}{\sum_{y \in C} \sum_{\mathbf{x} \in \mathcal{N}_y} \exp\ (-d(\mathbf{q}, \mathbf{x})/\tau)}, \tag{2}$$

where τ is the hyper-parameter of temperature.

3.2 Comprehensive Exploiting of k-NN

In this section, we propose to comprehensively leverage the k-NN, the representative of *lazy learning*, to augment the PLM-based classifier.

Role of k-NN as Prior Knowledge for Calibrating Training. As k-NN can easily make predictions for each query instance encountered without any training, it is intuitive to regard its predictions as priors to guide the network in focusing on hard examples during the training process of language models. We distinguish between easy and hard examples based on the results of k-NN. Given the probability distribution $p_{k\text{NN}}$ of $\mathbf{q}$ being predicted as true label y, we propose to adjust the relative loss for the

correctly-classified or misclassified instances identified by k-NN, in order to reweight the cross-entropy loss $\mathcal{L}_{CE}$. Specifically, we define the calibrated training loss $\mathcal{L}_J$ as:

$$\mathcal{L}_U = (1 + f(p_{k\text{NN}}))\,\mathcal{L}_{CE}, \tag{3}$$

where $f(p_{k\text{NN}})$ donates the modulating factor [1] for calibration. We are inspired by Focal-loss (Lin et al., 2018) to employ the modulating factor, while our focus is on exploring the application of k-NN in the fine-tuning of PLMs.

Intergrating k-NN into Inference Let $\mathbb{P}_{\mathcal{M}}$ denote the class distribution predicted by the PLM, and $\mathbb{P}_{k\text{NN}}$ be the class distribution predicted by a k-NN classifier. Then, the $\mathbb{P}_{\mathcal{M}}$ is reformulates by interpolating the non-parametric k nearest neighbor distribution $P_{k\text{NN}}$ using parameter λ (Khandelwal et al., 2020) to calculate the final probability $\mathbb{P}_U$ of the label as:

$$\mathbb{P}_U = \lambda \mathbb{P}_{k\text{NN}} + (1 - \lambda)\mathbb{P}_{\mathcal{M}}, \tag{4}$$

where $\lambda \in [0, 1]$ is an adjustable hyper-parameter.

4 Experiments

Dataset	Type	# Class	Test Size
SST-5	sentiment	5	2,210
TREC	question cls	5	500
MNLI	NLI	3	9,815
QNLI	NLI	2	5,463
BoolQ	QA	2	3,245
CB	NLI	3	250
SemEval	relation extraction	19	2,717
TACREV	relation extraction	42	15,509

Table 1: Detailed dataset statistics.

4.1 Datasets

We choose a variety of NLP tasks to evaluate our proposed methods, including sentiment analysis task (SST-5 (Socher et al., 2013)), question classification task (TREC (Voorhees and Tice, 2000)), NLI tasks (MNLI (Williams et al., 2018) and QNLI (Rajpurkar et al., 2016)), sentence-pair classification task (BoolQ (Clark et al., 2019) and CB (De Marneffe et al., 2019)), and information extraction tasks (SemEval (Hendrickx et al., 2010) and TACREV (Alt et al., 2020)). We also list a detailed introduction of datasets in Table 1.

4.2 Experimental Settings

Compared Baseline Methods. We adopt RoBERTa$_{\text{large}}$ (Liu et al., 2019) as the underline PLM and conduct comprehensive experiments to integrate k-NN into PLMs. We choose the baseline approaches and the variant of our proposed method as follows: (1) **k-NN**: the method described in §3.1, which performs classification directly through nearest neighbor retrieval of instance features without relying on any pre-trained language models (PLMs). (2) **FT**: which denotes vanilla fine-tuning with PLMs. (3) **FT_Scratch**: which denotes vanilla PLMs in zero-shot setting. (4) **PT**: which denotes prompt-tuning with PLMs, similar to (Gao et al., 2021). (5) **UNION-INF**: a variant of our method, which simply linear interpolate k-NN and paradigms of PLMs during the test time. (6) **UNION-ALL**: the completeness of our approach, which involves applying k-NN as prior knowledge for calibrating training and also integrating k-NN into inference.

[1] We specify the $f(p_{k\text{NN}}) = (1 - p_{k\text{NN}})^{\gamma}$, and other factors are also alternative.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

Shot	Method	SST-5 Acc.	TREC F1.	MNLI Acc.	QNLI Acc.	BoolQ Acc.	CB F1.	SemEval F1.	TACREV F1.	AVG Score.
Full	k-**NN**	35.8	80.0	41.5	57.2	61.4	42.3	2.5	5.3	40.8
	FT	59.2	97.8	83.9	89.1	81.7	89.5	89.4	72.5	82.9
	UNION-INF	59.5	98.0	84.0	89.2	82.9	89.6	89.2	67.8	82.5
	UNION-ALL	60.9	98.2	84.2	90.8	83.4	90.5	89.6	73.1	83.8
16	k-**NN**	$25.6_{2.4}$	$46.1_{5.0}$	$33.7_{0.3}$	$51.6_{1.3}$	$50.4_{2.6}$	$40.8_{4.9}$	$0.5_{0.4}$	$0.9_{0.3}$	31.1
	FT	$43.3_{0.7}$	$86.6_{4.7}$	$44.4_{4.5}$	$55.3_{3.7}$	$56.0_{4.2}$	$68.3_{4.7}$	$64.1_{2.3}$	$25.6_{0.3}$	55.5
	UNION-INF	$43.0_{1.2}$	$86.7_{4.5}$	$44.5_{4.5}$	$55.4_{3.4}$	$55.4_{4.3}$	$65.6_{4.7}$	$65.1_{2.1}$	$30.5_{1.7}$	55.8
	UNION-ALL	$43.7_{0.5}$	$90.0_{3.9}$	$51.7_{1.8}$	$58.1_{2.7}$	$57.6_{2.7}$	$69.8_{4.5}$	$67.2_{3.3}$	$32.1_{3.1}$	58.9
0	**FT_Scratch**	23.8	22.6	31.6	49.5	37.8	21.5	8.2	0.1	24.4
	PT	36.7	38.2	50.9	50.8	62.2	39.7	10.9	1.1	36.3
	UNION-INF	51.6	82.4	67.5	67.4	62.9	56.9	11.8	3.2	50.5
	UNION-ALL	35.1	38.0	53.7	50.4	62.4	50.3	11.3	1.4	37.8

Table 2: Results on eight NLP tasks across the fully-supervised, few-shot (16-shot) and zero-shot settings. For the 16-shot setting, we provide the mean and standard deviation across three different random seeds. Scores that are marked with an underline signify the best results among all methods.

Settings. We test the above methods in full-supervised, few-shot and zero-shot experiments, we assign different settings, respectively: (1) **Full-supervised setting:** We use full trainsets to train the PLMs and as neighbors to retrieve. (2) **Few-shot setting:** We follow LM-BFF (Gao et al., 2021) to conduct 16-shot experiment and test the average performance with a fixed set of seeds $\mathcal{S}_{\text{seed}}$, across three different sampled $\mathcal{D}_{\text{train}}$ for each task. In this setting, we use the few-shot training set as k-NN neighbors to retrieve. (3) **Zero-shot setting:** We directly evaluate the vanilla FT and UNION-INF on the test set **without training**. As for UNION-ALL, we take the prompt tuning (Gao et al., 2021) to tag the pseudo labels on **unlabeled** trainsets and apply untrained k-NN in the training and inference.

4.3 Hyper-parameter Settings

We report the hyper-parameters in Table 3. For the GLUE and SuperGLUE datasets, we follow LM-BFF[2] to construct templates and verbalizer for prompt-tuning. While for RE datastes, we follow KnowPrompt (Chen et al., 2021) to construct templates and verbalizer. We utilize Pytorch to conduct experiments with 1 Nvidia 3090 GPUs. We used the AdamW optimizer for all optimizations, with a linear warmup of the learning ratefollowed by a linear decay over the remainder of the training. The hyper-parameter settings used in our experiments are listed below.

Hyper-parameter	Value
maximum sequence length	{128, 256}
max training step	1000
evaluation step	100
learning rate	{1e-5, 2e-5, 5e-5}
batch size	8
gradient accumulation step	{2, 4, 8}
adam epsilon	1e-8
k	{16, 32, 128}
λ	{0.1 : .1 : 0.9}
τ	{0.01, 0.1, 1, 10}

Table 3: Hyper-parameter settings.

[2] https://github.com/princeton-nlp/LM-BFF

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

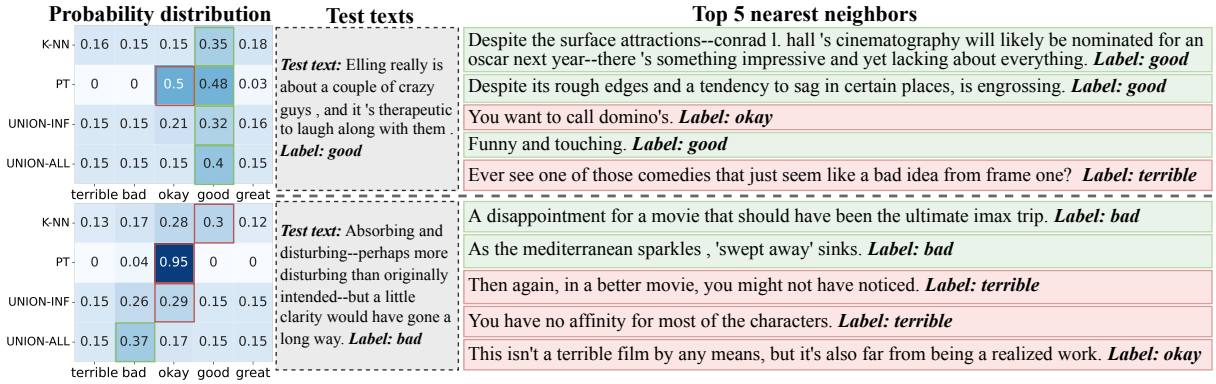

Figure 3: Case analysis to show how k-NN benefits the prediction of PLMs. We illustrate the test texts, the predicted probability distribution, and the top-5 nearest neighbors from the 16-shot training set of the SST-5 dataset.

4.4 Main Results

k-NN features result in performance gains. We compare the specific results with baseline models and provide comprehensive insights of k-NN on different paradigms and different settings. The results as shown in Table 1. Leverage k-NN features results in performance gains in both few-shot and fully-supervised settings. In the zero-shot setting, PT-based methods outperform FT-based and k-NN features further enhance the performance of PT-based methods, which demonstrates that it is flexible and general to integrate k-NN for PLMs.

Calibrating training vs. Incorporating into inference. It is necessary to study the different application scenarios of incorporating k-NN during the training and testing phases. From Table 2, we observe the following: (1) Leveraging k-NN during the test phrase is especially helpful for the zero-shot setting. While **UNION-ALL** performs worse due to the noise brought from the pseudo-labels on unsupervised data. (2) **UNION-INF** is not doing as well in the fully-supervised and few-shot setting. In contrast, **UNION-ALL** outperforms **UNION-INF** in these settings, especially in the few-shot setting. These findings reveal to us the applicable scenarios of incorporating k-NN and inspire further studies to utilize k-NN classifier more practically for efficient NLP.

4.5 Analysis

Q1: How does the lazy learner benefit eager learner? To further understand how does the *lazy learner* (k-NN) benefit the *eager learner* (PLM), we manually check cases in which k-NN, PT, UNION-INF and UNION-ALL produce different results. As shown in the example of the upper row of Figure 3, k-NN and UNION-ALL predict correctly when PT fails. This result is because UNION-ALL produces a more confident probability for the correct class via calibrating the attention on the easy vs. hard examples identified by the k-NN classifier. Note that the bottom row shows that UNION-ALL predicts correctly even when k-NN predicts wrongly, possibly due to the robustness of k-NN calibration.

Q2: Does the similarity metric matter? In the above experiments, we mainly utilize negative $L2$ distance to measure the similarity between the query $\mathbf{q}$ and the instance representation of the data store. It is intuitive to estimate the impact of different similarity metrics, such as cosine similarity. Thus, we present the performance of UNION-ALL using both metrics with the same hyperparameters as below.

Similarity Metric	$L2$	cos
16-shot SST-5 (%)	**43.7**	42.8
16-shot TREC (%)	**90.0**	89.4
16-shot QNLI (%)	**58.1**	57.2

We can find that UNION-ALL with cosine distance achieves nearly the same performance as those trained with $L2$, revealing that our UNION-ALL is robust to the similarity metric.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

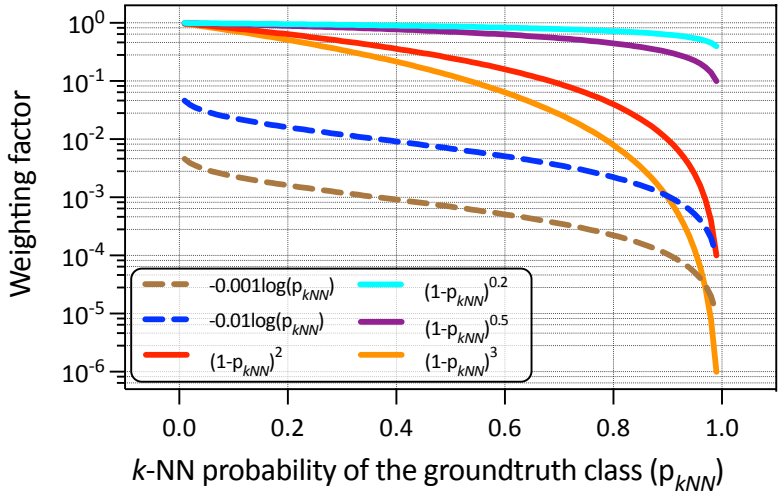

Figure 4: Comparison between the modulating factors NLL and Focal.

Q3: How dose the modulating factor $f(p_{k\mathbf{NN}})$ works? Since we adopt focal loss (Focal) as the modulating factor for main experiments, we further explore other functions as modulating factors, such as negative log-likelihood (NLL). As shown in Figure 4, we visualize two modulating factors with different settings of α and γ, where α donates a scalar that represent the proportion of the term of NLL, and γ is the exponential coefficient for Focal. We can find that NLL and Focal produce large weights for the misclassified examples, demonstrating the diversity of modulating factor selection.

5 Limitations

We only explore leveraging the training data for k-NN search, while various external domain data are also suitable for k-nearest neighbor retrieval. Moreover, incorporating k-NN also faces the following limitations: (1) the requirement of a large memory for retrieval; (2) hyper-parameters (such as λ and α) used for retrieval have an impact on the performance of model training; (3) if the number of nearest neighbors k is too large, it will also affect the efficiency.

6 Conclusion and Future Work

In this paper, we propose a novel method to enhance PLM-based classifiers using k-NN. Specifically, we introduce a calibration process and linear interpolation of inference phrases to effectively integrate k-NN into the training pipeline. To evaluate the effectiveness of our approach, we conduct a comprehensive and in-depth analysis of the role of k-NN in various NLU tasks and tuning paradigms. Our results demonstrate that the integration of k-NN is flexible and can significantly enhance the performance of large models. Future work should explore the combination of k-NN and LLMs such as (1) Inject external knowledge into the LLMs with k-NN. Specifically, k-NN can be used to retrieve relevant knowledge from an external database during the reasoning process, which can help correct errors and reduce the prevalence of gibberish output and factual errors that are common in LLMs. (2) Retrieve contextual information to enhance LLMs. k-NN algorithms can automatically retrieve relevant information based on the input sentence, such as instructions or other relevant context. (3) Augment the training data for LLMs. k-NN is a powerful tool for identifying similar instances in a large dataset, which can help overcome the limitations of data scarcity and improve the performance LLMs.

References

Uri Alon, Frank F. Xu, Junxian He, Sudipta Sengupta, Dan Roth, and Graham Neubig. 2022. Neuro-symbolic language modeling with automaton-augmented retrieval.

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. TACRED revisited: A thorough evaluation of the TACRED relation extraction task. In *Proceedings of ACL 2020*.

Oren Boiman, Eli Shechtman, and Michal Irani. 2008. In defense of nearest-neighbor based image classification. pages 1–8. IEEE.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of NeurIPS 2020*.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. *CoRR*, abs/2104.07650.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jerome H Friedman. 2017. *The elements of statistical learning: Data mining, inference, and prediction*. springer open.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of ACL*.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5133–5140. AAAI Press.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. In *Proc. of EMNLP*.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of SemEval*, pages 33–38.

Nora Kassner and Hinrich Schütze. 2020. Bert-knn: Adding a knn search component to pretrained language models for better QA. In *Findings of EMNLP*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150.

Proceedings of the 22nd China National Conference on Computational Linguistics, pages 889-897, Harbin, China, August 3 - 5, 2023.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. KNN-BERT: fine-tuning pre-trained models with KNN classifier. *CoRR*, abs/2110.02523.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2018. Focal loss for dense object detection.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yuxian Meng, Shi Zong, Xiaoya Li, Xiaofei Sun, Tianwei Zhang, Fei Wu, and Jiwei Li. 2021. GNN-LM: language modeling based on global contexts via GNN. *CoRR*, abs/2110.08743.

Guoshun Nan, Zhijiang Guo, Ivan Sekulić, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of ACL*.

Emin Orhan. 2018. A simple cache model for image recognition. 31:10107–10116.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Weijia Shi, Julian Michael, Suchin Gururangan, and Luke Zettlemoyer. 2022. Nearest neighbor zero-shot inference. *CoRR*, abs/2205.13792.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*.

Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. 2019. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Ningyu Zhang, Xin Xie, Xiang Chen, Shumin Deng, Chuanqi Tan, Fei Huang, Xu Cheng, and Huajun Chen. 2022. Reasoning through memorization: Nearest neighbor knowledge graph embeddings. *CoRR*, abs/2201.05575.