Not Far Away, Not So Close: Sample Efficient Nearest Neighbour Data Augmentation via MiniMax

Ehsan Kamalloo[*†◊] **Mehdi Rezagholizadeh**[*§] **Peyman Passban**[†§] **Ali Ghodsi**[‡¶]
[◊]Department of Computing Science, University of Alberta
[§]Huawei Noah's Ark Lab
[‡]David R. Cheriton School of Computer Science, Univeristy of Waterloo
[¶]Department of Statistics and Actuarial Science, Univeristy of Waterloo
kamalloo@cs.ualberta.ca

Abstract

In Natural Language Processing (NLP), finding data augmentation techniques that can produce high-quality *human-interpretable* examples has always been challenging. Recently, leveraging kNN such that augmented examples are retrieved from large repositories of unlabelled sentences has made a step toward interpretable augmentation. Inspired by this paradigm, we introduce *MiniMax-kNN*, a sample efficient data augmentation strategy tailored for Knowledge Distillation (KD). We exploit a semi-supervised approach based on KD to train a model on augmented data. In contrast to existing kNN augmentation techniques that blindly incorporate all samples, our method dynamically selects a subset of augmented samples that maximizes KL-divergence between the teacher and student models. This step aims to extract the most efficient samples to ensure our augmented data covers regions in the input space with maximum loss value. We evaluated our technique on several text classification tasks and demonstrated that MiniMax-kNN consistently outperforms strong baselines. Our results show that MiniMax-kNN requires fewer augmented examples and less computation to achieve superior performance over the state-of-the-art kNN-based augmentation techniques.

1 Introduction

Knowledge distillation (KD) (Buciluǎ et al., 2006; Hinton et al., 2015) has been successful in improving the performance of various NLP tasks such as language modelling (Jiao et al., 2020; Sanh et al., 2019; Turc et al., 2019), machine translation (Tan et al., 2019; Wu et al., 2020), natural language understanding (Passban et al., 2020; Rashid et al., 2021), and multi-task learning (Clark et al., 2019). It aims to transfer the knowledge embedded in a model—called teacher—to another succedent model—called student, without compromising on accuracy (Furlanello et al., 2018).

Data plays a significant role in the success of KD. The importance of data becomes even more crucial when dealing with large teacher models (Lopez-Paz et al., 2015) or managing tasks with small amount of labelled data (Rashid et al., 2020; Nayak et al., 2019). The training objective of KD focuses on minimizing the discrepancy between representations of a teacher model and a student model. However, this might not be the case for regions which are not covered by training data in the input space. Data augmentation comes into play as a natural solution for such circumstances.

Most existing data augmentation techniques are not tailored for KD as the dynamics of teacher and student models are not considered in generating augmented data. Moreover, other model-based data augmentation techniques such as adversarial approaches do not generate interpretable samples for NLP tasks (Du et al., 2021). In this work, inspired by the success of retrieval-based augmentation techniques (Guu et al., 2020; Khandelwal et al., 2020; Du et al., 2021; Kassner and Schütze, 2020), we propose *MiniMax-kNN*, an interpretable data augmentation methodology. Our technique is interleaved with KD training to generate realistically-looking training points. For this purpose, we use a massive external respostiory of unlabelled sentences. In contrast to previous kNN augmentation techniques which naively extract and incorporate k samples, we propose a minimax approach to adapt kNN augmentation to KD and select our augmented samples more efficiently.

Experimental results show that our technique requires significantly fewer samples, reaches the state-of-the-art kNN augmentation technique (Du et al., 2021), and improves generalization to unseen

[*] Equal Contribution
[†] Work done while at Huawei Noah's Ark Lab

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3522–3533
August 1–6, 2021. ©2021 Association for Computational Linguistics

data.[1]

Our key contributions can be summarized as follows:

- We tailor kNN-based data augmentation for KD via MiniMax to select more impactful augmented samples for training.
- We significantly improve sample efficiency of kNN-based data augmentation.
- We conduct extensive experiments to evaluate our proposed method and manifest that we can maintain the test performance with training on only influential augmented examples.

2 Background

2.1 Data Augmentation in KD

KD (Hinton et al., 2015) is a training method that incorporates the knowledge of a teacher network in training a student network. The teacher can be trained on the same dataset as the student and often provides a suitable approximation of the underlying distribution of data. The training loss of the student using KD is formulated as in Eq. (1).

$$\begin{aligned}
&\mathcal{L} = (1-\lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{KD} \\
&\mathcal{L}_{CE} = CE\Big(y, \sigma(z_s(x)\Big) \\
&\mathcal{L}_{KD} = \mathcal{T}^2 KL\Big(\sigma(\frac{z_t(x)}{\mathcal{T}}), \sigma(\frac{z_s(x)}{\mathcal{T}})\Big)
\end{aligned} \quad (1)$$

where z_s and z_t refer to the logits of the student and teacher networks, $\sigma(.)$ is the softmax prediction, CE and KL refer to cross entropy and KL-divergence loss, respectively. λ is a hyper-parameter which controls the contribution of the KD loss with respect to the original cross entropy loss, and $\mathcal{T}$ is the temperature parameter which determines the smoothness of the output probability.

Although KD has been shown to be successful in model compression (Buciluă et al., 2006) and improving the performance of neural networks (Furlanello et al., 2018), the core prerequisites for effective KD are often overlooked. Lopez-Paz et al. (2015) give a good insight about these conditions using the VC-dimension analysis:

$$O(\frac{|\mathcal{F}_s|_c + |\mathcal{F}_t|_c}{n^\alpha}) + \varepsilon_t + \varepsilon_l \leq O(\frac{|\mathcal{F}_s|_c}{\sqrt{n}}) + \varepsilon_s \quad (2)$$

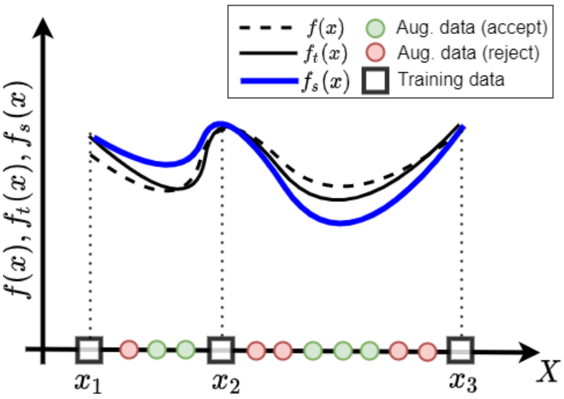

Figure 1: Data sparsity problem in KD; f, f_t, and f_s are representing the underlying function, teacher, and student outputs respectively. We show 10 augmented samples around x_2 with small circles on the X-axis. The green circles show the augmented samples which are selected by our MiniMax-kNN because these points correspond to maximum divergence regions of the teacher and student networks. The red circles are rejected augmented samples.

where $\mathcal{F}_s$ and $\mathcal{F}_t$ are the function classes corresponding to the teacher and student; $|.|_c$ is a function class capacity measure; $O(.)$ is the estimation error of training the learner; ε_s is the approximation error of the best estimator function belonging to the $\mathcal{F}_s$ class with respect to the underlying function; ε_t is a similar approximation error for the teacher with respect to the underlying function; ε_l is the approximation error of the best student function with respect to the teacher function; n is the number of training samples, and $\frac{1}{2} \leq \alpha \leq 1$ is a parameter related to the difficulty of the problem.

According to Eq. (2), it is clear that when the capacity of the teacher is large or when the number of the training samples is small, training with KD can be less beneficial. Figure 1 illustrates this problem through a synthetic example that KD loss forces the student to follow the teacher on training samples but there is no guarantee for such phenomenon to happen in regions in the input space that are not covered by training data. Therefore, the chance of a mismatch between two networks would be higher if training data is sparse or when there is a large gap between two networks.

Data augmentation can be considered as a remedy for this problem. To the best of our knowledge, most existing techniques are not sample efficient and blindly consider all generated samples in their training. As illustrated in Figure 1, different augmented samples might have different contribution to the final teacher/student loss. Moreover, these

[1] Source code is available at https://github.com/ehsk/Minimax-kNN

augmentation techniques are not tailored for KD. Our MiniMax-kNN solution addresses these two problems.

2.2 Nearest Neighbour Data Augmentation

The kNN augmentation strategy consists of two main stages: (a) a paraphrastic nearest neighbour retrieval engine, and (b) a training method using augmented samples.

Initially, training examples are queried over a large sentence repository using a general-purpose paraphrastic encoder. The aim of this stage is to find interpretable unannotated augmented samples that are semantically close to training data. For this purpose, we use one of the sentence repositories from SentAugment (Du et al., 2021), comprising 100M sentences collected from Common Crawl. We also employ the same paraphrastic sentence encoder, namely SASE, introduced in SentAugment. SASE is an XLM model (Lample and Conneau, 2019), fine-tuned on a number of well-known paraphrase datasets using a triplet loss to maximize the cosine similarity between representations of paraphrases. The similarity between a pair of sentence representations obtained from SASE can be adopted for unsupervised semantic similarity. Du et al. (2021) show that SASE achieves high correlation (0.73 on average) with human judgment on several STS benchmarks. Consequently, the kNN operation can be summarized as follows: Suppose a dataset $\{x_i, y_i\}_{i=1}^{N}$ where x_i and y_i denote an example and its corresponding label respectively. Given a large sentence repository $\mathcal{R}$ encoded using SASE, kNN is determined via top k sentences with respect to $\cos(\text{SASE}(x_i), \text{SASE}(s_j))$ where $s_j \in \mathcal{R}$.

Next, in step (b), a model is trained on the original data by minimizing $\mathcal{L}_{CE}$ from Eq. (1). The trained model learns task-specific knowledge that is further useful in finding relevant augmented examples. To this end, retrieved examples that are close to original examples within the teacher's space are retained to form augmented data. Augmented examples are subsequently incorporated into training via KD. A student model is then distilled from the teacher by leveraging teacher's soft labels on the combination of original data and augmented samples. In particular, for original examples $\{x_i, y_i\}$, $\mathcal{L}$ from Eq. (1) is minimized during training, whereas for the augmented examples, we only minimize $\mathcal{L}_{KD}$.

3 Related Work

3.1 KD in Tandem with Data Augmentation

Adaptive data augmentation can strengthen the capacity of the teacher in transferring knowledge to the student during distillation (Fu et al., 2020). Numerous studies (Chen et al., 2020b; Xie et al., 2020b) have applied KD for self-training in image classification tasks. In NLP, however, generating semantically plausible examples that can be easily inspected by humans is more challenging. In TinyBERT (Jiao et al., 2020), a contextual augmentation method is used along with KD, but such augmentation does not take the advantages of teacher or student's knowledge. A recent paradigm that heavily relies on data augmentation is zero-shot KD (Nayak et al., 2019; Rashid et al., 2020). In contrast, we explore the interpretability of augmentation in KD, which distinguishes our approach from the literature.

3.2 Data Augmentation in NLP

Word-level methods (Zhang et al., 2015; Xie et al., 2017; Wei and Zou, 2019) are heuristic based and do not necessarily yield natural sentences. More recently, contextual augmentations (Kobayashi, 2018; Yi et al., 2021) that substitute words for other words, is shown effective in text classification. However, these approaches do not produce diverse syntactic forms. Similarly, inspired by denoising auto-encoders, augmented examples can be sampled from the reconstruction distribution of corrupted sentences via Masked Language Modelling (Ng et al., 2020). Back-translation (Sennrich et al., 2016) is also another strategy to obtain augmented data (Yu et al., 2018; Xie et al., 2020a; Chen et al., 2020a; Qu et al., 2021).

Another line of work that mainly targets model robustness is to create new data or counterfactual examples via human-in-the-loop perturbations (Kaushik et al., 2020; Khashabi et al., 2020; Jin et al., 2020). Nonetheless, these strategies are task-specific and not scalable to generate data at massive scale. Besides, our method diverges from these studies in that we intend to build a semi-supervised system with minimal human intervention.

Several models (Miyato et al., 2017; Zhu et al., 2020; Jiang et al., 2020; Cheng et al., 2020; Qu et al., 2021) leveraged adversarial training for data augmentation. These methods manipulate the input embedding space to construct synthetic examples. Neighbourhoods around training instances in

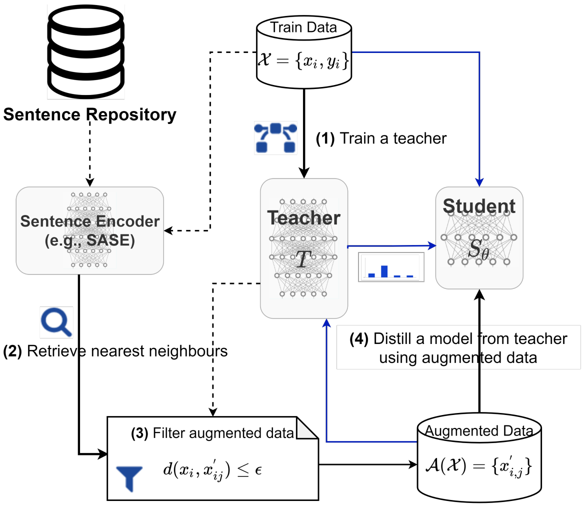
Sentence Repository
Train Data
$\mathcal{X} = \{x_i, y_i\}$
(1) Train a teacher
Sentence Encoder (e.g., SASE)
Teacher
T
Student
S_θ
(2) Retrieve nearest neighbours
(4) Distill a model from teacher using augmented data
(3) Filter augmented data
$d(x_i, x'_{ij}) \le \epsilon$
Augmented Data
$\mathcal{A}(\mathcal{X}) = \{x'_{i,j}\}$

Figure 2: A schematic view of MiniMax-kNN

the embedding space cannot be translated back to text and thus are not interpretable. Although we advocate for interpretable data augmentation, we do not compete with these techniques and in fact, gradient-based augmentation is complementary to our method.

Finally, kNN, a non-parametric search algorithm that probes an external data source to find nearest neighbours is employed in several NLP tasks such as language modelling (Khandelwal et al., 2020), machine translation (Khandelwal et al., 2021), cloze question answering (Kassner and Schütze, 2020), and open-domain question answering (Lewis et al., 2020). kNN offers access to explicit memory that can retrieve factual knowledge from a data store. kNN is highly interpretable as knowledge is stored in raw text, an easy format for humans to understand. Recently, SentAugment (Du et al., 2021) introduced a semi-supervised strategy with unlabelled sentences. It retrieves augmented samples from a universal data store using kNN. Our proposed strategy is in line with SentAugment at heart, but different in leveraging the augmented examples during training. We focus on sample efficiency and show that we can reduce the size of the augmented data—e.g., by 60% in sentiment classification as reported in Section 5.6—while reaching a competitive performance.

4 MiniMax-kNN Data Augmentation for KD

Inspired by Volpi et al. (2018) and Madry et al. (2018), we apply minimax framework to tailor a sample efficient kNN data augmentation for KD. Minimizing the maximum expected risk is used in adversarial training (Volpi et al., 2018) and it is shown to have guaranteed good performance on distributions (P) within a particular distance (ρ) from a source distribution (P_0):

$$\min_{\theta} \sup_{\mathcal{D}(P,P_0)\le\rho} \mathbf{E}[l(x', y'; \theta)] \tag{3}$$

where $\mathcal{D}$ is a notion of distance between distributions, l refers to the loss function, θ represents the parameters of the estimator model, and in our framework, (x', y') are augmented data samples.

Let us define the set of kNN augmented samples corresponding to the training sample $x_i \in \mathcal{X}$ from the training set, $\mathcal{X}$, to be $\mathcal{A}(x_i) = \{x'_{i1}, x'_{i2}, ..., x'_{ik}\}$. In the maximization phase, we define the loss $l(x', y'; \theta) = \mathrm{KL}\big(T(x'), S(x'; \theta)\big)$ between the softmax output of the teacher $T(x')$ and that of the student network $S(x'; \theta)$ with trainable parameters θ, with respect to the given augmented samples. Note that the augmented samples are unlabelled in the maximization phase. Then, we sort the augmented samples based on their loss value and form our MiniMax-kNN augmentation set $\bar{\mathcal{A}}(x_i)$ by selecting the top n out of the k samples in $\mathcal{A}(x_i)$. n is a hyper-parameter in our method that determines the sample efficiency of MiniMax-kNN. In order to enforce $\mathcal{D}(P, P_0) \le \rho$ on the distance between the two distributions in Eq. (3), in our kNN search, we set a maximum radial semantic distance ϵ between the sentence representation of accepted augmented samples in $\bar{\mathcal{A}}(x_i)$ and the sentence representation of their corresponding input x_i based on the angular distance metric:

$$d(x_i, x'_{ij}) = \frac{1}{\pi}\cos^{-1}\frac{< h^t_{cls}(x_i).h^t_{cls}(x'_{ij}) >}{\|h^t_{cls}(x_i)\|\|h^t_{cls}(x'_{ij})\|} \le \epsilon \tag{4}$$

where h^t_{cls} refers to the teacher's last layer hidden representation of the [CLS] token, and $<\cdot>$ denotes the dot product of two vectors. The discussion on how to adjust ϵ is given in Section 5.3.

In summary, our technique equips kNN augmentation with minimax to improve its sample efficiency. In contrast to adversarial data augmentation methods, our approach uses the minimax loss for selecting augmented samples. The overall structure of our augmentation strategy is visualized in Figure 2. We essentially follow three steps in each iteration during training:

(1) We construct teacher logits and student logits for augmented samples to measure KL-divergence between the two models.

(2) Out of all kNN samples, n samples with highest KL-divergence will be selected.

(3) KD loss is minimized for training data and selected augmented samples.

Our experiments reveal that this modification to KD underscores sample efficiency while retaining the test performance.

4.1 FLOPs Analysis of MiniMax-kNN

Minimax computations in MiniMax-kNN incur additional overhead costs during training, but how much precisely do minimax operations curtail the runtime performance? To answer this question, we analyze the logical compute complexity of our algorithm in terms of floating point operations (FLOPs) because it can be measured regardless of hardware considerations (Clark et al., 2020).

To this end, we compare FLOPs corresponding to each augmented example from MiniMax-kNN with vanilla kNN within an epoch. Suppose a forward pass and a backward pass for one batch takes F and B FLOPs, respectively. The number of matrix operations between the forward pass and the backward pass is not considerably different and hence, $F \approx B$ (Clark et al., 2020). For simplicity, we assume batch size is 1. Considering k_1 is the number of retrieved NNs, vanilla kNN requires $k_1F + k_1B$ additional FLOPs per epoch.

On the other hand, MiniMax-kNN selects n neighbours from k_2 retrieved nearest neighbours—i.e., $n < k_2$. The algorithm first takes the logits of all k_2 neighbours to compute KL-divergence vectors, which needs k_2F FLOPs, similar to vanilla kNN. The extra operations of MiniMax-kNN occur in the maximization step in which top n neighbours are determined with respect to their KL-divergence values. This operation can be carried out by sorting the KL-divergence vector, which costs S FLOPs. Note that $S \ll F$ because obtaining an output from a deep neural network model is far more costly than a sorting operation. The backward pass is then computed only for the n selected neighbours. Accordingly, the overall FLOPs for MiniMax-kNN is $k_2F + S + nB$. The difference between the FLOPs is:

$$\begin{aligned}\Delta_{\text{FLOPs}} &= \text{FLOPs}_{\text{vanilla-}k\text{NN}} - \text{FLOPs}_{\text{MiniMax-}k\text{NN}} \\ &= (k_1 - k_2)F + (k_1 - n)B + S\end{aligned}$$

Given that B can be approximated with F (as mentioned earlier) and $S \ll F$:

$$\Delta_{\text{FLOPs}} = (2k_1 - k_2 - n)F$$

Thus, as long as $k_2 + n < 2k_1$, MiniMax-kNN is more efficient than vanilla kNN. In experiments, we illustrate that how MiniMax-kNN surpasses vanilla-kNN while satisfying the FLOPs condition.

Dataset	#class.	#train	#dev	#test	avg. #tokens
SST-2	2	67.3K	872	1.8K	12.4
SST-5	5	8.5K	1.1K	2.2K	22.6
TREC	6	5K	500	500	11.4
CR	2	2.5K	640	642	21.4
IMP	2	3.9K	2.2K	2.6K	50.0

Table 1: Downstream tasks used for evaluation

5 Experiments

5.1 Datasets

We evaluate MiniMax-kNN on five datasets: SST-2 and SST-5 (Socher et al., 2013) for sentiment analysis, TREC (Li and Roth, 2002) for question type classification, CR (Hu and Liu, 2004) for product review classification, and Impremium's hate-speech detection dataset (IMP)[2]. Information related to all datasets is summarized in Table 1.

5.2 Experimental Setup

We adopt the publicly available pre-trained RoBERTa$_{\text{Large}}$ (Liu et al., 2019) and DistilRoBERTa (Sanh et al., 2019)—using the Huggingface Transformers library (Wolf et al., 2020) and the Pytorch Lightning library[3]—for evaluating our approach. For KD, RoBERTa$_{\text{Large}}$ is selected as the teacher. For training, we adhere to findings in Mosbach et al. (2021) and Zhang et al. (2021) to circumvent the fine-tuning instability problem by training for longer iterations—i.e., 100 epochs—with early stopping and use Adam optimizer with bias correction. The model is evaluated on the development data at the end of each epoch and the best performing model is chosen for testing. Our learning rate schedule follows a linear decay scheduler with a warm-up on $\{10\%, 20\%\}$ of the total number of training steps. The learning rate is tuned for each task separately out of {1e-5, 2e-5, 3e-5}, and the batch size is chosen from [16, 128] depending on the dataset size. For KD hyperparameters, we use grid search to choose the best λ and $\mathcal{T}$ from {0.3, 0.4, 0.5, 0.6} and {5, 10, 12, 20}. We also schedule augmentation to start after a certain number of epochs in the training. On SST-5, SST-2, and

[2] https://www.kaggle.com/c/detecting-insults-in-social-commentary/

[3] https://github.com/PyTorchLightning/pytorch-lightning

Model	SST-5	SST-2	TREC	CR	IMP
RoBERTa$_{\text{Large}}$ (Teacher)	57.6	96.2	98.0	94.1	90.0
DistilRoBERTa	52.9	93.5	96.0	**92.1**	86.8
DistilRoBERTa + KD	53.2	93.6	96.6	**92.1**	87.7
DistilRoBERTa + vanilla-8NN	55.2	94.7	97.0	91.3	88.4
Aug. Size (#forward / #backward pass)	8x / 8x	8x / 8x	8x / 8x	8x / 8x	8x / 8x
DistilRoBERTa + MiniMax-8NN*	**55.4**	**95.2**	**97.6**	91.6	**88.6**
Aug. Size (#forward / #backward pass)	5x / 4x	7x / 4x	8x / 4x	8x / 2x	8x / 1x

Table 2: Test accuracy (↑) on the downstream tasks (*denotes our approach and **bold** numbers indicate the best result—excluding the teacher—for each task).

Model	SST-5	SST-2	TREC	CR	IMP
vanilla-8NN	55.2	94.7	97.0	91.3	88.4
$n = 1$	55.4	94.4	96.4	91.4	**88.6**
$n = 2$	54.6	95.0	96.4	91.6	88.5
$n = 4$	55.4	**95.2**	**97.6**	91.6	87.4
$n = 6$	**55.6**	94.4	96.6	**91.8**	87.9

Table 3: Test accuracy (↑) of DistilRoBERTa on the downstream tasks varying the number of selected NNs (n) in MiniMax-8NN (**bold** numbers indicate the best result for each task).

TREC, augmentation takes effect at epochs 8, 6, and 6, respectively, whereas on IMP, and CR, augmentation starts at the beginning of training. All experiments were conducted on two Nvidia Tesla V100 GPUs.

Few-shot learning setup We follow Du et al. (2021) to setup the environment for few-shot learning experiments. In particular, we sample 2 training subsets with replacement from the original training set for each task. Each subset is balanced and consists of 20 examples per label. The development set is reduced to 200 examples for all tasks except CR in which we keep all of the original set. The label distribution is retained in the reduced development data. Evaluation is conducted on the actual test dataset. To obtain reliable results, we repeat training with 10 different seeds on each sampled dataset and report the average across all runs—i.e., 20 runs per task. Few-shot experiments were run on a single Nvidia Tesla V100 GPU.

5.3 MiniMax-kNN Results

First, we investigate the impact of kNN data augmentation at test time and compare MiniMax-kNN with vanilla kNN data augmentation. To this end, we train a RoBERTa$_{\text{Large}}$ as teacher on the original data. Then, we distill a small size student based on DistilRoBERTa from the teacher using the augmented data and the original data.

In Table 2, we report the performance of MiniMax-kNN as well as the vanilla-kNN on the downstream tasks. In this experiment, the number of nearest neighbours (k) is set to 8 and for MiniMax-kNN, we empirically select the minimum number of augmented examples (n) out of 8-NNs such that MiniMax-kNN exceeds vanilla-kNN. We observe that using KD alone leads to a marginal improvement on all tasks. Adding more data results in further improvements but comes at the expense of substantially longer training time. On the other hand, MiniMax-kNN reduces the cost of training as it learns through less than half of the NNs and yet, consistently outperforms vanilla-kNN.

Varying the number of selected examples (n) in MiniMax-kNN We explore the number of selected augmentations by varying $n \in \{1, 2, 4, 6\}$ for 8-NNs on the downstream tasks. Results are reported in Table 3. Interestingly, picking n as small as either 1 or 2 results in superior performance of MiniMax-kNN, compared to vanilla-kNN, on all tasks. In TREC, and SST-2, the sweet spot is $n = 4$. In SST-5, and CR, MiniMax-kNN performs better as n grows. On the contrary, in IMP, accuracy declines by increasing n.

Varying the number of nearest neighbours (k) In order to investigate the optimal number of NNs, we assess the effect of k on the downstream tasks. The Results are reported in Table 4. We observe that more data sometimes makes the training noisy and as a result, performance deteriorates—e.g., $k = 2$ in SST-5 and IMP. Nonetheless, when the augmentation size is sufficiently large, test results improve—i.e., $k = 8$ in all datasets except CR. Apart from three cases—i.e., $k = 2, 4$ in SST-2, and $k = 4$ in CR—MiniMax-kNN is superior to vanilla-kNN by incorporating roughly 50% fewer

Task	KD	$k=1$	$k=2$		$k=4$		$k=8$	
		vanilla	**vanilla**	**MiniMax**	**vanilla**	**MiniMax**	**vanilla**	**MiniMax**
SST-5	53.2	53.9	52.0	52.5	54.7	55.0	<u>55.2</u>	**55.4**
SST-2	93.6	93.7	<u>94.7</u>	94.2	94.6	93.8	<u>94.7</u>	**95.2**
TREC	96.6	96.2	96.4	96.6	96.8	96.8	<u>97.0</u>	**97.4**
CR	92.1	91.9	92.1	<u>92.2</u>	**92.4**	91.6	91.3	91.9
IMP	87.7	87.2	87.1	87.6	86.0	87.8	<u>88.4</u>	**88.6**

Table 4: Test accuracy (↑) of DistilRoBERTa on the downstream tasks varying the number of nearest neighbours (k). **KD** refers to knowledge distillation with no data augmentation. For MiniMax, n is equal to half of k neighbours for $k = 2, 4$ and when $k = 8$, n is selected as in Table 2 (**bold** and <u>underline</u> indicate best and second best results per task).

Model	SST-5		SST-2		TREC		CR		IMP	
vanilla-8NN	162.1		484.5		78.4		43.7		99.4	
MiniMax-8NN*	158.8	2% ↓	634.4	31% ↑	101.1	29% ↑	30.8	30% ↓	38.6	61% ↓

Table 5: Training time (in seconds) for one epoch (↓), averaged across epochs during training, on the downstream tasks along with the percent of reduction compared to vanilla-8NN. MiniMax-8NN and vanilla-8NN refer to the models we used for Table 2 (*denotes our approach).

examples.

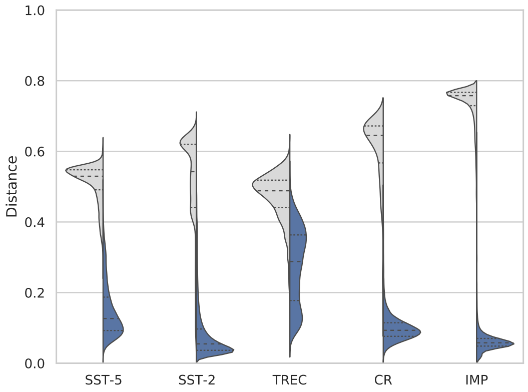

Figure 3: Distance distribution of augmented examples for each dataset (Left: *mismatched labels* / Right: *matched labels*)

Adjusting the maximum radial distance (ϵ) in MiniMax-kNN We plot the distance distribution of augmented data for two cases: (a) when the teacher predicts the same label as the original examples for augmented ones (*matched labels*) (b) when the predicted label for augmented examples do not match that of original examples (*mismatched labels*). Figure 3 illustrates a clear distinction between these two groups. Considering these insights, we find an empirical heuristic to set ϵ. When the overlap between groups is infinitesimal, we tune ϵ in the vicinity of the maximum distance of *matched labels*. The rationale here is to avoid altering the skewness of the original label distribution. Throughout our experiments, ϵ is set to 0.22, and 0.4 for SST-5, and SST-2, respectively. However, we find $\epsilon = \infty$ works best on CR, IMP, and TREC.

5.4 Runtime Efficiency

In §4.1, we showed that MiniMax-kNN is computationally more efficient than vanilla-kNN when $k_2 + n < 2k_1$. Given the number of nearest neighbours is identical ($k_1 = k_2$) in our experiments, any choice of n makes MiniMax-kNN more efficient than vanilla-kNN in theory. However, in our implementation of MiniMax, we feed selected examples again to the student, thereby triggering a redundant forward pass[4]. Although this change reduces the efficiency of MiniMax-kNN in practice, it significantly simplifies the implementation. Thus, the above condition evolves to $k_2 + 2n < 2k_1$ in our experiments. Nonetheless, in Table 2, this new efficiency constraint still holds on all tasks except SST-2, and TREC. To calculate the exact amount of speed-up, we measure the average training time corresponding to one epoch for each task. The results are outlined in Table 5. On IMP, MiniMax-kNN saves more than 60% of training time and on CR, MiniMax-kNN brings almost 30% speed-up. Also, MiniMax-kNN is slightly faster than vanilla-kNN on SST-5. However, MiniMax-kNN trains around 30% slower on SST-2 and TREC.

5.5 Ablation Study

We analyze each component of our augmentation strategy to understand how they impact the overall

[4] All augmented examples are initially fed to the student within a PyTorch `no_grad` block. Since we want to backpropagate through only selected examples, they should be fed again to the student.

Model	**SST-5**	**SST-2**	**TREC**	**CR**	**IMP**
SIZE	100	40	120	40	40
SentAugment (Du et al., 2021)	44.4 ± 1.0	86.7 ± 2.3	92.1 ± 2.4	89.7 ± 2.0	81.9 ± 1.4
AUG. SIZE	1000	1000	1000	1000	1000
RoBERTa$_{\text{Large}}$ (Teacher)	43.9 ± 2.5	81.1 ± 2.5	89.9 ± 3.5	83.7 ± 2.7	75.5 ± 5.5
RoBERTa$_{\text{Large}}$ + KD	44.8 ± 2.5	82.3 ± 4.4	91.9 ± 2.1	83.9 ± 5.3	81.4 ± 1.9
RoBERTa$_{\text{Large}}$ + vanilla-10NN	45.5 ± 2.3	85.5 ± 2.9	91.6 ± 1.9	88.0 ± 1.5	81.5 ± 3.3
RoBERTa$_{\text{Large}}$ + MiniMax-10NN ($n = 6$)*	46.8 ± 1.4	86.5 ± 1.8	91.8 ± 1.0	88.3 ± 1.5	81.7 ± 2.3
AUG. SIZE (#forward / #backward pass)	1030 / 700	380 / 280	835 / 720	400 / 280	360 / 280

Table 6: Few-shot learning results of MiniMax-kNN on the downstream tasks (*denotes our approach). Compared to SentAugment, our proposed approach achieves competitive performance, but with the use of fewer augmented examples.

#	**Model**	**SST-5**	Δ
1	DistilRoBERTa	52.9	-
2	+ KD	53.2	+0.3
3	+ KD + random-8	54.4	+1.2
4	+ KD + random-8 + reranked	54.2	-0.2
5	+ KD + random-8 + reranked + ϵ	52.3	-1.9
6	+ KD + 8NN	54.7	+1.5
7	+ KD + 8NN + reranked (=vanilla)	55.2	+0.5
8	+ vanilla-8NN + ϵ	55.0	-0.2
9	+ MiniMax-8NN + ϵ	55.4	+0.4

Table 7: Ablation study of MiniMax-kNN on SST-5. Δ denotes the performance difference with respect to the previous row, but for the first row in each section, it indicates the difference with the accuracy of KD—i.e., row 2.

effectiveness of MiniMax-kNN. To this end, three components of our strategy are targeted for an ablation study. First, the effect of nearest neighbours is measured by replacing them with random examples from the sentence repository. Then, to determine whether reranking neighbours by teacher is helpful, we preserve the order of nearest neighbours returned by the SASE. Finally, we relax the maximum radial distance to include all nearest neighbours.

In Table 7, we report the results on SST-5. Surprisingly, random augmentation (row 3) scores only 0.3% lower than kNN augmentation (row 6). Reranking nearest neighbours by the teacher further boosts the results by 0.5% (row 7). The presence of maximum radial distance is not helpful for vanilla-kNN as it leads to 0.4% drop in the accuracy (row 8). Finally, our selection mechanism in MiniMax-kNN (row 9) leads to a 0.2% improvement compared to vanilla-kNN (row 7).

5.6 Few-shot experiments

Our data augmentation strategy can be applied to few-shot learning scenarios where a minuscule number of labelled data is available. Therefore, we simulate a few-shot learning setting as described in Section 5.2. In addition to vanilla-kNN and no augmentation baselines, we compare our results with SentAugment (Du et al., 2021), the state-of-the-art method in kNN data augmentation. In SentAugment, experiments are conducted on 5 randomly sampled small datasets and top 3 results of 10 different runs are averaged across sampled datasets, which means average over 15 runs in total. To be comparable to SentAugment, we average across all 10 runs for 2 sampled datasets, average over 20 runs in total, to report our results. In SentAugment, augmented few-shot datasets contain 1000 examples including the original data. For MiniMax-kNN, we use 10-NNs in this experiment with a maximum radial distance.

Table 6 shows the few-shot learning results. The performance of our baselines follows a similar trend in the full-size data experiments. In particular, KD without augmentation slightly improves the test accuracy; Vanilla-kNN brings almost 1.9% improvement on average, and MiniMax-kNN consistently surpasses vanilla-kNN by 0.7% on average. Compared to SentAugment, MiniMax-kNN reaches a competitive performance. The key advantage of MiniMax-kNN lies in sample efficiency. Specifically, MiniMax-kNN falls short by only 0.3% on SST-2, and IMP with using less than 40% of the SentAugment augmented data on average. On CR, MiniMax-kNN lags behind by 1.4%, but again on roughly 40% of the SentAugment data size. Moreover, SentAugment outperforms our approach by 0.3% on TREC, while the size of augmentation is reduced by almost 20%. Lastly, MiniMax-kNN outperforms SentAugment in SST-5 by 2.4% with almost same amount of data.

5.7 Qualitative Analysis

We study the quality of augmented examples retrieved from the sentence repository. Table 8 presents four examples from SST-5, CR, and TREC along with the corresponding top 3-NNs. The top

(i) **SST-5:** this is a stunning film, a one-of-a-kind tour de force.	**very positive**
Here is masterful film-making in action. (5)	*very positive*
It's an expertly-crafted spectacle-event movie. (1)	*very positive*
This is a unique cinematographic experience. (6)	*very positive*
(ii) **CR:** one also exhibited extremely slow speed when going to the menu.	**negative**
No menu appears to make it very quick and easy to use. (15)	*negative*
Switching between options in the main menu is relatively slow. (13)	*negative*
the only niggle i have found is that the menus are a bit slow at times. (8)	*negative*
(iii) **SST-5:** final verdict: you've seen it all before.	**very negative**
Below is the final result. (15)	*neutral*
The final verdict: Go ahead and buy (4)	*positive*
Nut in the end, the final result always pays out. (7)	*positive*
(iv) **TREC:** What causes the body to shiver in cold temperatures?	**DESC**
How is it possible that a higher minimum wage could actually lead to more inequality within a country? (11)	*DESC*
How did the minimum wage increase come about? (13)	*DESC*
How is the new minimum wage hike impacting them? (7)	*DESC*

Table 8: Examples, derived from the augmented CR, SST-5, and TREC, after teacher reranking (the numbers in the bracket indicate the initial rank by SASE). For the nearest neighbours, the teacher's *predictions* are also provided, although soft labels will be used during training. Row (iii) shows an example of label mismatch and row (iv) highlights a mediocre paraphrase retrieval despite matching labels.

two rows show clear-cut examples that the nearest neighbours are in fact paraphrased forms of original samples. Also, the teacher predicts the same label as the original examples for these augmented examples. We observe task-specific knowledge that the teacher has learned from original data helps to rank retrieved sentences—e.g., in the second row, reranking pushes the neighbours at ranks 15 and 13 to the top 3.

However, the augmented data is not always perfect. To identify the limitations of kNN augmentation, we manually inspect 20 samples, randomly drawn from SST-5 and TREC. We find that inaccurate paraphrase retrieval undercuts the quality of augmented examples shown in the bottom rows of Table 8. A side effect of this weakness is the domain mismatch, denoting that augmentation can introduce out-of-domain data. For instance, the input data in TREC is expected to be in interrogative mood, but the retrieval may return declarative sentences. A potential solution to this problem could be utilizing a different repository, entirely comprised of questions in this case, similar to that of Perez et al. (2020). However, curating such repository for specialized domains can be challenging. Moreover, the improvements we observe in the experiments show that this issue is not prevalent in our selected tasks.

6 Conclusion

In this paper, we presented a sample efficient semi-supervised data augmentation technique, namely MiniMax-kNN. The augmentation procedure is framed as finding nearest neighbours from a massive repository of unannotated sentences. The crucial aspect of kNN augmentation is interpretability as augmented examples are written in natural language. We adopt KD to learn from unlabelled data. The key ingredient of our approach is to find the most impactful examples that maximize the KL-divergence between the teacher and the student models. We show that MiniMax-kNN can reduce the augmented data size by 50% while improving upon vanilla augmentation.

Acknowledgments

We thank MindSpore[5]—a new deep learning framework—for partially supporting this work.

References

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 535–541, New York, NY, USA. Association for Computing Machinery.

Jiaao Chen, Yuwei Wu, and Diyi Yang. 2020a. Semi-supervised models via data augmentation for classifying interactive affective responses. In *AffCon@ AAAI*.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. 2020b. Big self-supervised models are strong semi-supervised learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 22243–22255. Curran Associates, Inc.

[5] https://www.mindspore.cn

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. AdvAug: Robust adversarial augmentation for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5961–5970, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. 2019. BAM! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5931–5937, Florence, Italy. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov, and Alexis Conneau. 2021. Self-training improves pre-training for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5408–5418, Online. Association for Computational Linguistics.

Jie Fu, Xue Geng, Zhijian Duan, Bohan Zhuang, Xingdi Yuan, Adam Trischler, Jie Lin, Chris Pal, and Hao Dong. 2020. Role-wise data augmentation for knowledge distillation. arXiv:2004.08861.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 1607–1616. PMLR.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 3929–3938.

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

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8018–8025.

Nora Kassner and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3424–3430, Online. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. 2020. More bang for your buck: Natural perturbation for robust question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 163–170, Online. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32, pages 7059–7069.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

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. arXiv:1907.11692.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. 2015. Unifying distillation and privileged information. arXiv:1511.03643.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. 2019. Zero-shot knowledge distillation in deep networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 4743–4751. PMLR.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1268–1283, Online. Association for Computational Linguistics.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2020. ALP-KD: Attention-based layer projection for knowledge distillation. arXiv:2012.14022.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8864–8880, Online. Association for Computational Linguistics.

Yanru Qu, Dinghan Shen, Yelong Shen, Sandra Sajeev, Jiawei Han, and Weizhu Chen. 2021. CoDA: Contrast-enhanced and diversity-promoting data augmentation for natural language understanding. In *International Conference on Learning Representations*.

Ahmad Rashid, Vasileios Lioutas, Abbas Ghaddar, and Mehdi Rezagholizadeh. 2020. Towards zero-shot knowledge distillation for natural language processing. arXiv:2012.15495.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. MATE-KD: Masked Adversarial TExt, a companion to knowledge distillation. arXiv:2105.05912.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

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*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. arXiv:1908.08962.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. 2018. Generalizing to unseen domains via adversarial data augmentation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizadeh, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1016–1021, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020a. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268. Curran Associates, Inc.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020b. Self-training with noisy student improves ImageNet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10687–10698.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. In *International Conference on Learning Representations*.

Mingyang Yi, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Zhi-Ming Ma. 2021. Reweighting augmented samples by minimizing the maximal expected loss. In *International Conference on Learning Representations*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. In *International Conference on Learning Representations*.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. FreeLB: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.