WHAT MAKES BETTER AUGMENTATION STRATEGIES? AUGMENT DIFFICULT BUT NOT TOO DIFFERENT

Anonymous authors
Paper under double-blind review

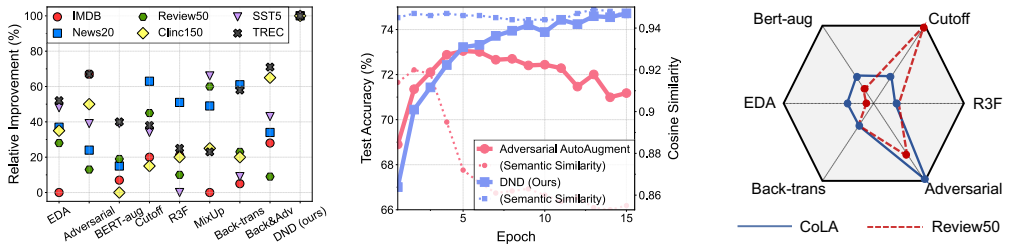
ABSTRACT

The practice of data augmentation has been extensively used to boost the performance of deep neural networks for various NLP tasks. It is more effective when only a limited number of labeled samples is available, *e.g.*, low-data or class-imbalanced regimes. Most current augmentation techniques rely on parameter tuning or inherent randomness; hence, their effectiveness largely varies on the tasks. To efficiently find the best augmentation strategy for each task, learning data augmentation policy is a promising solution, but the question of what makes a good augmentation in NLP tasks and how to design the reward function for learning a good policy remains under-explored. To answer this, we hypothesize that good data augmentation should construct more diverse and challenging samples for providing informative training signals, while avoiding the risk of losing the semantics of original samples. Therefore, we design a novel reward function for updating the augmentation policy to construct *difficult* but *not too different* samples (DND). Particularly, we jointly optimize a data augmentation policy while training the model, to construct the augmented samples with low confidence but a high semantic similarity with original ones. In addition, we introduce a sample re-weighting scheme to focus on difficult augmented samples after the original ones are learned confidently for more effective learning from the augmented ones. Our learning-based augmentation outperforms the recent state-of-the-art augmentation schemes on various text classification tasks and GLUE benchmark by successfully discovering the effective augmentations for each task. Remarkably, our method is more effective on the challenging low-data and class-imbalanced regimes, and the learned augmentation policy is well-transferable to the different tasks and models.

1 INTRODUCTION

Deep neural networks (DNNs) have shown near human- or superhuman-level performances on various NLP benchmark tasks. This success, however, crucially relies on the availability of large labeled datasets, which typically require a lot of human efforts to be constructed (Brown et al., 2020; Cheng et al., 2020). Although recent advance in language models (LMs) significantly boosts state-of-the-art performances using self-supervised pre-training with a massive unlabeled dataset (Devlin et al., 2018; Liu et al., 2019), the number of given labeled samples on each downstream task is still critical for the performance and stability of fine-tuned LMs (Sun et al., 2019; Zhang et al., 2020a). Data augmentation is one of the most effective ways to efficiently use the given labeled samples by enlarging the amount and diversity with label-preserving transformations, so it helps to improve the generalization of DNNs (Lim et al., 2019). Hence, the practice of data augmentation has been extensively used to boost the performance of DNNs and LMs for various NLP tasks (Jiang et al., 2020; Qu et al., 2021), and it is more effective when only a limited number of labeled samples is available, *e.g.*, low-data or class-imbalanced regimes (Xie et al., 2020; Kim et al., 2020).

Most current augmentation techniques rely on parameter tuning (Zhu et al., 2020) or inherent randomness (Wei & Zou, 2019); hence, their effectiveness largely varies between the tasks (see Figure 1(a)). To efficiently find the best augmentation strategy for each task, learning augmentation policy (*i.e.,* distribution of multiple augmentations) has been explored as a promising solution (Cubuk et al., 2019; Hataya et al., 2020). Here, a key for successful policy learning is on the design of reward function used to update it, which implicates what a good augmentation is; hence, various rewards are studied especially under vision tasks, *e.g.,* maximizing validation accuracy (Cubuk et al.,



(a) Task-wise dependency (b) Importance of reward (c) Task-wise adaptability

Figure 1: (a) Task-dependency of each augmentation method measured on 5-shot tasks in Section 4.2. (b) Test accuracy and semantic similarity between original and augmented samples under different reward functions to optimize the augmentation policy: Adversarial AutoAugment (Zhang et al., 2020b) and DND (Ours). (c) Distribution of augmentations learned via DND. Depend on a given task, the distribution of augmentations is automatically adapted.

2019) or training loss (Zhang et al., 2020b). However, we found that these existing reward functions would not be enough for learning effective augmentation policy in NLP tasks. This is due to the different nature of NLP tasks from discrete input space or vulnerability of the semantics for a slight modification (Wang et al., 2018; Garg & Ramakrishnan, 2020), For example, we found that the recent state-of-the-art augmentation learning approach used for image classification is not effective for NLP tasks as it suffers from losing the semantics of original sentence in augmentations (see Figure 1(b)). This motivates us to explore what criteria make a good augmentation, and how to design the reward function for policy to find them in NLP tasks.

Contribution. In this paper, we develop a simple yet effective augmentation scheme for NLP tasks, coined learning to augment **D**ifficult, but **N**ot too **D**ifferent (DND). We first design a novel reward function for updating augmentation policy under the following intuition: an effective augmented sample should be more *difficult* than the original sample for providing 'informative' training signals, while maintaining its semantic meaning *not too different* from the original for avoiding 'wrong' signals. Specifically, we compose the reward function using two complementary ingredients from training model: (1) model's training loss for the given downstream task (on augmented samples) and (2) semantic similarity (between augmented and original samples) measured by using the contextualized sentence embeddings from the training language model.

We further apply a sample-wise re-weighting scheme when updating the model and policy with the augmented samples, to learn them more effectively by incorporating the original sample's learning status. Specifically, we assign more weight on the augmented sample, which (1) its corresponding original sample is learned enough, *i.e.*, has high confidence, and (2) the confidence gap between original and augmented samples is large. With the proposed reward function and re-weighting scheme, the augmentation policy is simultaneously optimized during the model fine-tuning with a gradient-based optimization, by applying an efficient continuous relaxation to the non-differentiable components within the augmentation policy (*e.g.,* sampling the augmentations from the policy).

We demonstrate the effectiveness of the proposed augmentation policy learning scheme on various text classification datasets and GLUE benchmark (Wang et al., 2019) , where our method consistently improves over the recent state-of-the-art augmentation schemes by successfully discovering the effective augmentation methods for each task (see Figure 1(c)). For example, on the six different text classification datasets, DND exhibited 16.45% and 9.08% relative test error reduction on average, compared to the vanilla and the previous best augmentation method, respectively. We also found that DND is more effective on the challenging low-resource and class-imbalanced regimes, and the learned augmentation policy can be easily transferable to the different tasks and models. This implies the broad applicability of our augmentation scheme: for instance, it might substitute the cost of augmentation learning on new tasks by using the pre-trained augmentation policies from DND.

2 Related Works

Data augmentation in NLP tasks. Recently, the interest in data augmentation has also increased on NLP tasks (Feng et al., 2021) and various data augmentation approaches have been proposed,

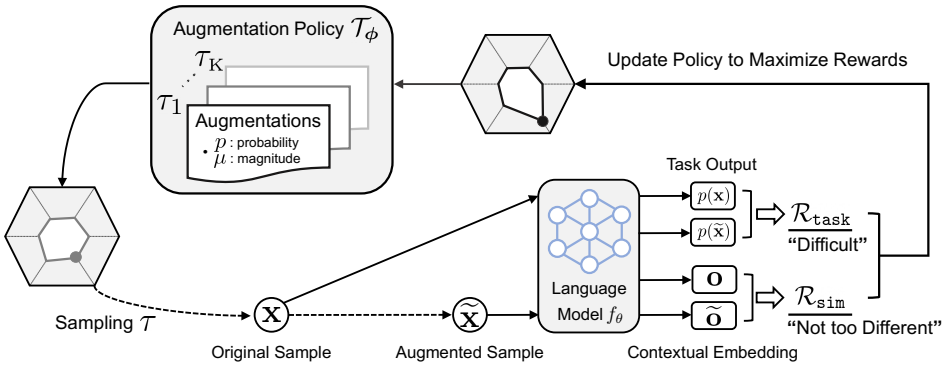


Figure 2: Illustration of learning to augment **D**ifficult, but **N**ot too **D**ifferent (DND).

such as word replacement with pre-defined rules (Kang et al., 2018; Wei & Zou, 2019) or pre-trained language models (Yi et al., 2021), back-translation (Xie et al., 2020), MixUp (Guo et al., 2019), and Cutoff (Shen et al., 2020). In a broader sense, adversarial training (Jiang et al., 2020; Zhu et al., 2020) can also be viewed as constructing augmented samples by adding perturbations to the word embedding. Stacking of these augmentations has been shown to provide further improvement (Qu et al., 2021), but it requires the exhaustive searching cost for exploring each configuration one by one. Hence, the analysis what makes a good data augmentation is an emerging research direction (Miyato et al., 2018; Gontijo-Lopes et al., 2020). For example, after the extensive study on the existing augmentations in the vision tasks, Gontijo, et al. (Gontijo-Lopes et al., 2020) identify that the successful augmentations have the following properties: higher training loss and smaller distribution shift from the original. But, any specific way to obtain such a desired augmentation is not suggested.

Learning-based data augmentation. Automatically finding the effective data augmentation from data has naturally emerged to overcome the exhausting search cost from deciding which augmentation would be applied and tuning its parameters. Learning augmentation policy (*i.e.,* distribution of multiple augmentations), optimized to find the effective augmentation given a huge search space, is the representative method for this. Here, a key for successful policy learning is a design of reward function used to update it; hence, various objectives for learning augmentation policy have been proposed, especially in the computer vision. Maximizing the validation accuracy is arguably one of the most natural and successful approaches (Cubuk et al., 2019; Hu et al., 2019). Also, matching the distributions of original and augmented samples is popularly used as a learning objective by considering data augmentation as the problem of filling the missing data points of training distribution (Tran et al., 2017; Hataya et al., 2020). Recently, the augmentation policy optimized to increase the training loss outperforms previous approaches (Zhang et al., 2020b; Wu et al., 2020).

On the other hand, finding the efficient optimization method for learning augmentation policy is also widely explored to overcome the non-differentiability within the augmentation policy (*e.g.,* sampling the augmentations from the policy). Reinforcement learning has been the representative method (Cubuk et al., 2019; Zhang et al., 2020b), but more efficient continuous relaxation has been recently proposed (Hataya et al., 2020; Li et al., 2020). However, such learning-based data augmentation schemes have been under-explored for NLP tasks.

3 DND: Augment Difficult, but Not too Different

3.1 Overview and Problem Description

Overview. In this section, we present our technique, coined learning to augment Difficult, but Not too Different (DND). Our main idea is optimizing the augmentation policy to construct a sample that (a) is more difficult than the original sample for improving the generalization of the classifier, yet (b) maintains the semantics of the original sample. In Section 3.2, we elaborate the reward and the training objective for our augmentation policy. In Section 3.3, we provide details of augmentation policy, and its continuous relaxation for maximizing our optimization over a non-differentiable training objective. We offer a full description of the proposed DND in Algorithm 1.

Problem description. We first describe the problem setup of our interest under a text classification scenario. Let $\mathcal{D}$ denote the given training dataset consisting of tuples $(\mathbf{x}, y) \in \mathcal{D}$ where $\mathbf{x} = [x_1, \ldots, x_L]$ and y are the sequence of input tokens and the target label, respectively. Our goal is to train a classifier $f_\theta(\mathbf{x})$, initialized with a pre-trained transformer-based language model (*e.g.,* BERT (Devlin et al., 2018)), to minimize the task-specific loss $\mathcal{L}_{\texttt{task}}(\mathbf{x}, y)$ such as a cross-entropy loss $\ell_{\texttt{xe}}(f_\theta(\mathbf{x}), y)$. For improving the model's generalization upon a (limited) training data $\mathcal{D}$, one can construct the augmented sample $\widetilde{\mathbf{x}} = \tau(\mathbf{x})$ by applying a label-preserving transformation τ. In contrast to existing works relying on parameter tuning or inherent randomness for finding effective τ, we consider optimizing the augmentation policy $\mathcal{T}_{\boldsymbol{\phi}}$ where $\tau \sim \mathcal{T}_{\boldsymbol{\phi}}$ to maximize a reward function $\mathcal{R}$.

3.2 Reward and Training Objective

To train our data augmentation policy $\mathcal{T}_\phi$, we train it to maximize the following reward function:

$$\mathcal{R}(\mathbf{x}, \widetilde{\mathbf{x}}, y) = \mathcal{R}_{\texttt{task}}(\widetilde{\mathbf{x}}, y) + \lambda_{\texttt{s}} \mathcal{R}_{\texttt{sim}}(\mathbf{x}, \widetilde{\mathbf{x}}),$$

where $\widetilde{\mathbf{x}} = \tau(\mathbf{x}),\ \tau \sim \mathcal{T}_\phi$. $\mathcal{R}_{\texttt{task}}$ and $\mathcal{R}_{\texttt{sim}}$ are the newly proposed reward functions to (1) provide 'informative' learning signal to training classifier and (2) keep the semantic meaning of augmented sample $\widetilde{\mathbf{x}}$ to be similar to the original $\mathbf{x}$, respectively.

Rewarding difficult samples. To provide informative training signals, we encourage the augment policy to construct a harder view of the original sample; hard samples are known to facilitate training for many tasks, *e.g.*, object detection (Shrivastava et al., 2016), curriculum learning (Bengio et al., 2009) and data augmentation (Zhang et al., 2020b; Wu et al., 2020). To this end, we reward the policy for generating an augmented sample $\widetilde{\mathbf{x}} = \tau(\mathbf{x}),\ \tau \sim \mathcal{T}_\phi$ with high training loss, while down-weighing the case where the high training loss incurred from the original sample $\mathbf{x}$, not the augmentation policy $\mathcal{T}_\phi$. To this end, we update the augmentation policy $\mathcal{T}_\phi$ to maximize the following reward:

$$\mathcal{R}_{\texttt{task}}(\widetilde{\mathbf{x}}, y) = w(\widetilde{\mathbf{x}}) \mathcal{L}_{\texttt{task}}(\widetilde{\mathbf{x}}, y), \quad w(\widetilde{\mathbf{x}}) = p_y(\mathbf{x})^\alpha \big(\max\{p_y(\mathbf{x}) - p_y(\widetilde{\mathbf{x}}), 0\} \big)^\beta \tag{1}$$

where $p(\mathbf{x}) = \text{Softmax}\big(f_\theta(\mathbf{x})\big)$. Intuitively, the re-weighting factor $w(\widetilde{\mathbf{x}})$ emphasizes the samples of which (a) the original sample is confidently learned (*i.e.*, large $p_y(\mathbf{x})$) and (b) the gap of hardness between the original and the augmented sample is large (*i.e.,* large $p_y(\mathbf{x}) - p_y(\widetilde{\mathbf{x}})$).

Rewarding not too different samples. However, without proper regularization, the reward $\mathcal{R}_{\texttt{task}}$ may derive the augmentation policy to generate samples that are semantically meaningless, *e.g.*, out-of-distribution samples (Gong et al., 2020; Wei et al., 2020) or samples losing the characteristics of the target label y. See Figure 1(b) and 6(b) for a demonstration of such a behavior. Hence, for reducing such a potential risk, we further update the augmentation policy $\mathcal{T}_{\boldsymbol{\phi}}$ to keep the semantic meaning of augmented sample $\widetilde{\mathbf{x}}$ be similar to the original $\mathbf{x}$. To this end, we first extract the semantics of sentence $\mathbf{x}$ from its contextualized embedding $\mathbf{o} = [o_1, \ldots, o_L]$ where o_ℓ is ℓ-th output of the transformer model. Then, a linear classifier g_W is applied to classify whether the given embedding pair $(\mathbf{o}_1, \mathbf{o}_2)$ comes from the same sample ($y_{\texttt{pos}}$) or not ($y_{\texttt{neg}}$), respectively. With this classifier, we formulate the reward, $\mathcal{R}_{\texttt{sim}}$, for updating policy to correctly classify the given $(\mathbf{x}, \widetilde{\mathbf{x}})$ as positive, hence maintain the semantic of original $\mathbf{x}$ as follow:

$$\mathcal{R}_{\texttt{sim}}(\mathbf{x}, \widetilde{\mathbf{x}}) = -\ell_{\texttt{xe}}\big(g_W(\mathbf{m}, \widetilde{\mathbf{m}}), y_{\texttt{pos}}\big) \tag{2}$$

where $\ell_{\texttt{xe}}$ is the cross-entropy loss and

$$g_W(\mathbf{m}_1, \mathbf{m}_2) = W \cdot \text{Concat}(\mathbf{m}_1, \mathbf{m}_2, |\mathbf{m}_1 - \mathbf{m}_2|), \quad \mathbf{m} = \text{Mean-Pool}(\mathbf{o}) = \frac{1}{L} \sum\nolimits_{\ell=1}^{L} o_\ell \tag{3}$$

To train this linear classifier g_W, we add the following loss for training the classifier:

$$\mathcal{L}_{\texttt{sim}}(\mathbf{x}, \tilde{\mathbf{x}}) = \ell_{\texttt{xe}}\big(g_W(\mathbf{m}, \widetilde{\mathbf{m}}), y_{\texttt{pos}}\big) + \ell_{\texttt{xe}}\big(g_W(\mathbf{m}, \mathbf{n}), y_{\texttt{neg}}\big) \tag{4}$$

where $\mathbf{n}$ is the mean-pooled contextualized embedding of the other training sample except $\mathbf{x}$ itself.

Furthermore, since the quality of contextualized embedding $\mathbf{o}$ is critical for our algorithm, we additionally constrain each o_ℓ to preserve the semantics of input tokens x_ℓ. To this end, we additionally train our classifier with a word reconstruction loss $\mathcal{L}_{\texttt{recon}}$:

$$\mathcal{L}_{\texttt{recon}}(\mathbf{x}) = \sum\nolimits_{l=1}^{L} \ell_{\texttt{xe}}(o_l \mathbf{V}^\top, x_l) \tag{5}$$

where $\mathbf{V}$ is the word embeddings within the transformer model.

Algorithm 1 Learning to augment difficult, but dot too different (DND)

Input: Classifier from a pre-trained language model f_θ, linear classifier g_W, data augmentation policy $\mathcal{T}_{\boldsymbol{\phi}}$, training data $\mathcal{D}$, model learning rate $\eta_{\boldsymbol{\theta}}$, policy learning rate $\eta_{\boldsymbol{\phi}}$, update frequency T_p
for each iteration t **do**
  Draw a mini-batch $\mathcal{B} = \{(\mathbf{x}_i, y_i)_{i=1}^{B}\}$ from $\mathcal{D}$
  Construct augmented samples from policy $\tilde{\mathcal{B}} = \{(\tilde{\mathbf{x}}_i, y_i)_{i=1}^{B}\}$ where $\tilde{\mathbf{x}}_i = \tau(\mathbf{x}_i),\ \tau \sim \mathcal{T}_{\boldsymbol{\phi}}$
  Update model to minimize $\mathcal{L}_{\texttt{task}}(\mathbf{x}_i, y_i) + w(\widetilde{\mathbf{x}}_i)\mathcal{L}_{\texttt{task}}(\widetilde{\mathbf{x}}_i, y_i) + \mathcal{L}_{\texttt{sim}}(\mathbf{x}_i, \widetilde{\mathbf{x}}_i) + \lambda_{\texttt{r}}\mathcal{L}_{\texttt{recon}}(\mathbf{x}_i)$
  if t % T_p = 0 **then**
    Update policy to maximize $\mathcal{R}_{\texttt{task}}(\tilde{\mathbf{x}}_i, y_i) + \lambda_{\texttt{s}}\mathcal{R}_{\texttt{sim}}(\mathbf{x}_i, \tilde{\mathbf{x}}_i)$
  end if
end for

3.3 Continuous relaxation for training the augmentation policy

Here, we describe the details on how the augmentation policy is optimized to maximize the non-differentiable reward function defined in Section 3.2. We first note that $\mathcal{T}_{\boldsymbol{\phi}}$ consists of T consecutive operations $\mathcal{O}_1, \dots, \mathcal{O}_T$ where the operations are used to sample the augmentation from a pool of K augmentations $\tau_1, \dots, \tau_K$. Here, each operation $\mathcal{O}_t$ is parameterized as $\mathcal{O}_t(\boldsymbol{p}_t, \boldsymbol{\mu}_t)$, where $\boldsymbol{p} = [p_1, \dots, p_K],\ \sum_k p_k = 1$ is the probability to sample the augmentation from the pool, and $\boldsymbol{\mu} = [\mu_1, \dots, \mu_K]$ is the magnitudes used when each augmentation τ_k is sampled, *i.e.,* $\tau_k(\cdot; \mu_k) \sim \mathcal{O}(\boldsymbol{p}, \boldsymbol{\mu})$ with a probability p_k. Hence, as the non-differentiabiliy from operations breaks the gradient flow from the classifier (*e.g.,* sampling the augmentation), it is non-trivial how to optimize the parameters of augmentation policy $\boldsymbol{\phi} = \{\boldsymbol{p}_t, \boldsymbol{\mu}_t\}_{t=1}^{T}$.

To this end, we adopt a continuous relaxation for each operation following (Hataya et al., 2020; Li et al., 2020), hence makes it be differentiable with respect to two trainable parameters of policy: probability $\boldsymbol{p}$ and magnitude $\boldsymbol{\mu}$. Here, the key idea is using 1) Gumbel-Softmax (Jang et al., 2017) instead of Categorical distribution for sampling with $\boldsymbol{p}$, and 2) straight-through estimator for $\boldsymbol{\mu}$ (Bengio et al., 2013). Therefore, by using this continuous relaxation, the classifier and the augmentation policy would be alternately trained in a fully differentiable manner. More discussions about the used augmentation pool and details are in Section 4.1 and Appendix A.3.

4 Experiments

In this section, we evaluate our algorithm using two important NLP tasks: (1) text classification and (2) entailment tasks. We first describe the experimental setups in Section 4.1. In Section 4.2, we present empirical evaluations on DND and other baseline augmentations under both classification and entailment tasks. Finally, in Section 4.3, we provide additional analysis on our algorithm regarding (a) ablation of each component, (b) transferability of learned policy, (c) behavior of the learned augmentation policy, and (d) additional qualitative results.

4.1 Experimental Setups

Datasets and tasks. For the text classification task, we use the following benchmark datasets: (1) News20 (Lang, 1995), (2) Review50 (Chen & Liu, 2014), and (3) CLINC150 (Larson et al., 2019) for topic classification, (4) IMDB (Maas et al., 2011) and SST-5 (Socher et al., 2013) for sentiment classification, and (6) TREC (Li & Roth, 2002) for question type classification. For datasets without given validation data, we use 10% of its training samples for the validation. With these datasets, we construct the 5-shot and class-imbalanced variants to further verify the effectiveness of augmentation for more challenging scenarios. We also validate the proposed method on GLUE benchmark (Wang et al., 2019), which is a collection of diverse natural language understanding tasks: 1 regression task (STS-B) and seven different classification tasks (RTE, MRPC, CoLA, SST-2, QNLI, QQP, and MNLI). As different metrics are used for each task in GLUE benchmark, we specify the used metric under the name of each task in Table 3. For the evaluation, the given development set is used. Detailed descriptions of tasks and construction of 5-shot and class-imbalanced datasets are in Appendix A.1.

Table 1: Test accuracy of RoBERTa classifiers after fine-tuning with each data augmentation method on 6 different text classification datasets under (5-shot / full) setups. All the values and error bars are mean and standard deviation across 3 random seeds. The best and the second best results are indicated in bold and underline, respectively.

Method	IMDB	SST-5	TREC	News20	Review50	CLINC150
Vanilla	59.1$_{\pm 3.2}$ / 95.2$_{\pm 0.1}$	25.3$_{\pm 2.5}$ / 57.1$_{\pm 0.5}$	69.3$_{\pm 6.3}$ / 97.1$_{\pm 0.4}$	46.3$_{\pm 2.1}$ / 82.0$_{\pm 0.4}$	45.1$_{\pm 0.5}$ / 71.4$_{\pm 0.1}$	86.4$_{\pm 0.3}$ / 95.5$_{\pm 0.2}$
EDA	56.7$_{\pm 2.1}$ / 95.3$_{\pm 0.1}$	27.4$_{\pm 1.8}$ / 56.1$_{\pm 0.8}$	71.8$_{\pm 3.5}$ / 96.6$_{\pm 0.6}$	47.8$_{\pm 2.4}$ / 83.1$_{\pm 0.1}$	46.4$_{\pm 1.1}$ / 71.8$_{\pm 0.1}$	87.1$_{\pm 0.8}$ / 95.9$_{\pm 0.1}$
Adversarial	<u>66.1</u>$_{\pm 2.7}$ / 95.5$_{\pm 0.1}$	27.0$_{\pm 3.5}$ / 56.2$_{\pm 1.0}$	72.5$_{\pm 3.7}$ / 97.3$_{\pm 0.3}$	47.3$_{\pm 0.6}$ / <u>83.4</u>$_{\pm 0.2}$	45.7$_{\pm 0.6}$ / 73.0$_{\pm 0.2}$	87.4$_{\pm 0.3}$ / 96.0$_{\pm 0.2}$
BERT-aug	59.8$_{\pm 6.0}$ / 95.3$_{\pm 0.1}$	27.0$_{\pm 3.2}$ / 56.4$_{\pm 1.4}$	71.2$_{\pm 2.1}$ / 96.7$_{\pm 0.1}$	46.9$_{\pm 1.9}$ / 82.7$_{\pm 0.5}$	46.0$_{\pm 1.0}$ / 72.2$_{\pm 0.2}$	85.0$_{\pm 0.3}$ / 95.9$_{\pm 0.4}$
Cutoff	61.2$_{\pm 3.7}$ / 95.4$_{\pm 0.2}$	26.8$_{\pm 1.4}$ / 56.2$_{\pm 0.8}$	71.1$_{\pm 4.3}$ / <u>97.5</u>$_{\pm 0.5}$	<u>48.9</u>$_{\pm 2.5}$ / 83.4$_{\pm 0.4}$	47.2$_{\pm 2.0}$ / 72.7$_{\pm 0.2}$	86.7$_{\pm 0.7}$ / 96.0$_{\pm 0.1}$
R3F	61.3$_{\pm 2.4}$ / 95.5$_{\pm 0.1}$	25.3$_{\pm 3.5}$ / 57.1$_{\pm 1.1}$	70.5$_{\pm 4.3}$ / <u>97.5</u>$_{\pm 0.5}$	48.4$_{\pm 1.1}$ / 83.3$_{\pm 0.3}$	45.6$_{\pm 1.2}$ / 72.8$_{\pm 0.2}$	86.8$_{\pm 0.1}$ / 95.7$_{\pm 0.1}$
MixUp	52.9$_{\pm 3.3}$ / 95.4$_{\pm 0.2}$	<u>28.2</u>$_{\pm 2.2}$ / 57.0$_{\pm 1.2}$	70.4$_{\pm 4.2}$ / 97.1$_{\pm 0.3}$	48.3$_{\pm 0.9}$ / 82.6$_{\pm 0.9}$	<u>47.9</u>$_{\pm 1.1}$ / 72.4$_{\pm 0.1}$	86.9$_{\pm 0.4}$ / <u>96.1</u>$_{\pm 0.3}$
Back-trans	59.6$_{\pm 3.8}$ / 95.3$_{\pm 0.1}$	25.7$_{\pm 2.7}$ / <u>57.5</u>$_{\pm 0.3}$	72.1$_{\pm 3.7}$ / 97.0$_{\pm 0.2}$	48.8$_{\pm 2.1}$ / 82.6$_{\pm 0.6}$	46.2$_{\pm 0.6}$ / 72.0$_{\pm 0.2}$	86.8$_{\pm 0.4}$ / 95.9$_{\pm 0.3}$
Back&Adv	62.0$_{\pm 0.8}$ / <u>95.6</u>$_{\pm 0.1}$	27.2$_{\pm 2.0}$ / 56.9$_{\pm 0.4}$	<u>72.7</u>$_{\pm 4.2}$ / 96.7$_{\pm 0.3}$	47.7$_{\pm 2.2}$ / 83.2$_{\pm 0.5}$	45.5$_{\pm 1.0}$ / <u>73.1</u>$_{\pm 0.2}$	<u>87.7</u>$_{\pm 0.1}$ / 95.8$_{\pm 0.1}$
Ours	**69.6**$_{\pm 2.7}$ / **95.7**$_{\pm 0.1}$	**29.7**$_{\pm 2.0}$ / **58.3**$_{\pm 0.4}$	**74.1**$_{\pm 1.4}$ / **98.0**$_{\pm 0.2}$	**50.4**$_{\pm 1.4}$ / **85.2**$_{\pm 0.4}$	**49.8**$_{\pm 0.5}$ / **74.9**$_{\pm 0.2}$	**88.4**$_{\pm 0.4}$ / **96.6**$_{\pm 0.1}$

Baselines. We compare our method with various baseline augmentations in NLP tasks. We first consider a naïve fine-tuning without data augmentation, denoted by (*a*) Vanilla, followed by various augmentation schemes; (*b*) EDA (Wei & Zou, 2019): synonym replacement, random insertion, random swap, and random deletion are randomly applied; (*c*) Adversarial (Jiang et al., 2020; Zhu et al., 2020): adversarial example is constructed by adding adversarial perturbation on the word embedding; (*d*) BERT-aug (Yi et al., 2021): input sentence is randomly masked, then filled with pre-trained BERT to construct new sample; (*e*) Cutoff (Shen et al., 2020): continuous span of word embedding is masked out similar to Dropout (Srivastava et al., 2014); (*f*) R3F (Aghajanyan et al., 2021): random noise are added to word embedding, then consistency loss is applied between original and perturbed sample; (*g*) MixUp (Guo et al., 2019): two independent samples are mixed on the word embedding and label space; (*h*) Back-trans (Xie et al., 2020): two translation models perform source to target and target to source translations; (*i*) Back&Adv (Qu et al., 2021): composition of back-translation and adversarial perturbation, that was identified as the best way for using multiple augmentations (Qu et al., 2021). Details on the implementations are presented in Appendix A.2.

Training details. All the experiments are conducted by fine-tuning RoBERTa-base (Liu et al., 2019) using Adam optimizer (Kingma & Ba, 2015) with a fixed learning rate 1e-5 and the default hyperparameters of Adam. For the text classification tasks, the model is fine-tuned using the specified augmentation method with batch size 8 for 15 epochs. For GLUE benchmark task, we commonly use batch size 16, except RTE task with batch size 8 following (Aghajanyan et al., 2021). Under each augmentation, the model is fine-tuned for 20 epochs on small tasks (RTE, MRPC, STS-B, CoLA) or 10 epochs on larger tasks (SST-2, QNLI, QQP, MNLI).

In the case of DND, we optimize the augmentation policy using Adam optimizer. We choose hyperparameters in our method from a fixed set of candidates; the update frequency $T_p \in \{1, 5, 10\}$, $\lambda_{\mathtt{s}} \in \{0.1, 0.5, 1.0\}$, and $(\alpha, \beta) \in \{(0, 0), (0.5, 0.5)\}$ based on the validation set. Intuitively, the smaller datasets are beneficial for smaller T_p and $\lambda_{\mathtt{s}}$. Other hyperparameters are fixed as $\lambda_{\mathtt{r}} = 0.05$ and policy's learning rate γ_ϕ as 1e-3. We use a simple two-layer MLPs for W. As the data augmentation pool, we include all the baseline augmentation except MixUp, as it is unnatural to keep the semantic of original with this, due to its sample mixing. For the augmentations without specific magnitudes, we only update the probability to be sampled. With this augmentation pool, our augmentation policy is composed of $T = 2$ consecutive operations. Also, to construct more diverse augmentations, we simultaneously train four augmentation policies by randomly selecting one of them for each mini-batch. More details can be found in Appendix A.3.

4.2 Experimental Results on Text Classification and GLUE Benchmark

To verify the effectiveness of DND, we compare DND with various data augmentation schemes in the scenario of fine-tuning RoBERTa-base model for each downstream task. Table 1 summarizes the experimental results on six text classification tasks, along with the challenging low-resource scenarios (*i.e.,* 5-shot). On the original full datasets, DND consistently outperforms the baseline augmentations for all datasets. To be specific, DND exhibits 16.45% relative test error reduction compared to the vanilla method in the average. Furthermore, compared to the previous best augmentation method for each dataset, DND exhibits 9.08% relative test error reduction in the average. In the case of challenging 5-shot datasets, our method again outperforms the strong baselines with a larger margin.

Table 2: Comparison of classification performance (bACC/GM) with RoBERTa classifiers on 3 different text classification datasets under 2 different class-imbalance ratio $\gamma_{\texttt{imb}}$. Except the gray-colored row, the classifiers are fine-tuned with over-sampled datasets augmented by each data augmentation method. All the values and error bars are mean and standard deviation across 3 random seeds. The best and the second best results are indicated in bold and underline, respectively.

Method	SST-2		News20		Review50	
	$\gamma_{\texttt{imb}} = 100$	$\gamma_{\texttt{imb}} = 20$	$\gamma_{\texttt{imb}} = 100$	$\gamma_{\texttt{imb}} = 20$	$\gamma_{\texttt{imb}} = 100$	$\gamma_{\texttt{imb}} = 20$
Vanilla	66.1$_{\pm 4.9}$ / 59.7$_{\pm 8.9}$	87.8$_{\pm 0.4}$ / 87.6$_{\pm 0.4}$	65.9$_{\pm 0.6}$ / 59.7$_{\pm 2.6}$	72.5$_{\pm 0.7}$ / 71.0$_{\pm 0.9}$	58.2$_{\pm 0.2}$ / 45.0$_{\pm 1.0}$	64.2$_{\pm 0.5}$ / 57.9$_{\pm 1.0}$
Re-sample	71.6$_{\pm 4.3}$ / 65.8$_{\pm 6.5}$	87.9$_{\pm 0.2}$ / 87.6$_{\pm 0.2}$	66.7$_{\pm 0.7}$ / 61.6$_{\pm 1.3}$	73.0$_{\pm 0.4}$ / 71.4$_{\pm 0.3}$	59.6$_{\pm 0.9}$ / 49.3$_{\pm 4.2}$	65.0$_{\pm 0.4}$ / 60.5$_{\pm 0.9}$
EDA	71.9$_{\pm 8.7}$ / 64.3$_{\pm 9.9}$	88.6$_{\pm 0.3}$ / 88.4$_{\pm 0.3}$	66.3$_{\pm 0.6}$ / 59.9$_{\pm 0.5}$	74.0$_{\pm 0.6}$ / 72.4$_{\pm 0.8}$	60.3$_{\pm 0.4}$ / 49.1$_{\pm 1.1}$	65.2$_{\pm 0.4}$ / 60.7$_{\pm 1.4}$
Adversarial	78.2$_{\pm 3.7}$ / 75.3$_{\pm 5.0}$	90.1$_{\pm 0.2}$ / 89.9$_{\pm 0.2}$	66.8$_{\pm 1.1}$ / 60.4$_{\pm 3.7}$	75.0$_{\pm 0.7}$ / 73.9$_{\pm 1.0}$	61.8$_{\pm 0.4}$ / 52.0$_{\pm 1.4}$	67.1$_{\pm 0.2}$ / 63.4$_{\pm 0.9}$
BERT-aug	66.8$_{\pm 4.7}$ / 57.9$_{\pm 7.9}$	89.8$_{\pm 0.2}$ / 89.7$_{\pm 0.4}$	66.5$_{\pm 0.3}$ / 59.8$_{\pm 0.2}$	73.3$_{\pm 0.9}$ / 71.6$_{\pm 1.5}$	60.2$_{\pm 0.6}$ / 50.3$_{\pm 1.9}$	65.3$_{\pm 0.5}$ / 60.8$_{\pm 0.9}$
Cutoff	77.2$_{\pm 3.8}$ / 74.2$_{\pm 5.5}$	89.4$_{\pm 0.3}$ / 89.3$_{\pm 0.3}$	67.0$_{\pm 0.6}$ / 61.4$_{\pm 1.0}$	73.7$_{\pm 1.1}$ / 71.9$_{\pm 1.0}$	60.8$_{\pm 0.4}$ / 50.2$_{\pm 0.7}$	65.6$_{\pm 0.6}$ / 61.1$_{\pm 1.4}$
R3F	72.5$_{\pm 1.3}$ / 67.9$_{\pm 2.4}$	89.1$_{\pm 1.6}$ / 88.9$_{\pm 1.8}$	66.5$_{\pm 0.8}$ / 60.9$_{\pm 2.1}$	73.9$_{\pm 0.4}$ / 72.4$_{\pm 0.4}$	59.6$_{\pm 0.6}$ / 47.2$_{\pm 2.7}$	65.2$_{\pm 0.5}$ / 60.5$_{\pm 1.1}$
MixUp	68.7$_{\pm 5.6}$ / 60.9$_{\pm 9.4}$	88.9$_{\pm 0.0}$ / 88.7$_{\pm 0.0}$	67.3$_{\pm 0.5}$ / 61.9$_{\pm 1.3}$	73.7$_{\pm 0.2}$ / 72.6$_{\pm 0.2}$	61.1$_{\pm 0.5}$ / 49.4$_{\pm 1.4}$	65.9$_{\pm 0.2}$ / 62.0$_{\pm 0.9}$
Back-trans	71.3$_{\pm 4.1}$ / 65.5$_{\pm 6.7}$	89.4$_{\pm 0.5}$ / 89.3$_{\pm 0.6}$	66.7$_{\pm 0.7}$ / 61.9$_{\pm 1.0}$	73.5$_{\pm 1.3}$ / 71.0$_{\pm 2.7}$	60.7$_{\pm 0.8}$ / 50.4$_{\pm 3.2}$	65.7$_{\pm 0.3}$ / 61.9$_{\pm 0.9}$
Back&Adv	77.8$_{\pm 4.5}$ / 75.3$_{\pm 6.6}$	89.3$_{\pm 0.2}$ / 89.1$_{\pm 0.3}$	67.0$_{\pm 0.5}$ / 60.7$_{\pm 0.6}$	74.0$_{\pm 0.3}$ / 71.7$_{\pm 1.5}$	61.7$_{\pm 0.3}$ / 52.5$_{\pm 1.6}$	66.7$_{\pm 0.2}$ / 63.0$_{\pm 0.5}$
Ours	**84.3**$_{\pm 2.8}$ / **83.5**$_{\pm 3.5}$	**91.1**$_{\pm 0.5}$ / **91.0**$_{\pm 0.6}$	**69.8**$_{\pm 1.0}$ / **66.4**$_{\pm 1.0}$	**76.8**$_{\pm 0.5}$ / **76.0**$_{\pm 0.7}$	**63.9**$_{\pm 0.9}$ / **54.0**$_{\pm 1.7}$	**68.4**$_{\pm 0.2}$ / **64.0**$_{\pm 0.8}$

Table 3: Evaluation results of vanilla (RoBERTa) classifiers with the GLUE development set after fine-tuning with each data augmentation method on the given task. All the values are mean across 3 random seeds and the standard deviation is presented in Appendix C. The best and the second best results are indicated in bold and underline, respectively. Gray-colored scores are those lower than the corresponding vanilla scores.

Method (Metrics)	RTE (Acc)	MRPC (Acc)	STS-B (P/S)	CoLA (Mcc)	SST-2 (Acc)	QNLI (Acc)	QQP (Acc/F1)	MNLI-m/mm (Acc)
Vanilla	79.4	89.62	90.72/90.42	62.2	94.46	92.64	90.87/88.17	87.34/86.85
EDA	77.8	90.03	90.14/89.92	55.8	94.90	92.50	91.41/88.50	87.46/87.31
Adversarial	80.5	91.00	**91.08/90.68**	63.5	94.88	93.18	91.50/88.58	87.91/87.84
BERT-aug	79.6	90.11	89.47/89.35	61.2	94.47	92.95	91.07/88.27	87.73/87.25
Cutoff	78.0	91.01	90.53/90.27	64.0	94.42	92.84	91.13/88.27	87.22/87.22
R3F	79.7	90.69	90.77/90.56	64.6	94.95	93.06	91.30/88.54	87.97/87.66
MixUp	79.8	89.95	90.46/90.16	64.3	94.50	92.57	90.80/87.84	87.50/87.02
Back-trans	79.9	90.20	90.54/90.33	58.8	94.53	92.64	90.82/88.05	87.50/87.06
Back&Adv	80.9	90.69	90.61/90.33	59.5	94.57	92.71	90.90/88.21	87.73/87.53
Ours	**81.5**	**91.18**	90.78/90.49	**65.0**	**95.30**	**93.48**	**91.85/89.00**	**88.21/88.12**

These results show that our method successfully provides 'informative' learning signals to the training classifier regardless of the number of given labeled datasets.

Next, we evaluate DND on class-imbalanced datasets; more practical setups suffer from the limited training data. Here, we consider one more natural baseline, 'Re-sampling', that equally samples each class in a mini-batch. Then, it is combined with each augmentation by applying augmentation instead of using duplicated samples for re-sampling (for details, see Appendix A.1). For the evaluation, we report two popular metrics for class-imbalance: the arithmetic mean (bACC) and geometric mean (GM) over class-wise accuracy. In Table 2, one can observe that the gain from data augmentation is more significant compared to the case of balanced datasets, as the lack of diversity in minority classes would be addressed by it. However, our learning-based augmentation scheme further enlarges this gain by constructing the augmented samples specialized for improving minority classes, based on its task-wise adaptability. Specifically, DND exhibits 20.33% and 11.01% relative test error reductions in the average compared to Re-sampling and the previous best augmentation method, respectively.

Furthermore, we evaluate DND on the GLUE benchmark, composed of 8 different tasks. Table 3 summarizes the corresponding results, and one can observe that DND consistently improves the RoBERTa-base model for all tasks; furthermore, DND outperforms the strong baselines for most cases. We remark that the existing augmentations are sometimes even harmful, as they can't consider the characteristics of each task. For example, the goal of CoLA task is classifying whether the given sentence is grammatically valid or not. Hence, this task is much sensitive to linguistic change, and it results in the degradation with the word-level augmentation methods. In contrast, our method automatically reflects such inherent characteristics of each task.

Table 4: Ablation study on different criteria for updating policy. Test accuracy of vanilla (RoBERTa) classifiers after fine-tuning across various augmentations and learning objectives on News20 and Review50 are compared. All the values and error bars are mean and standard deviation across 3 random seeds. The best and the second best results are indicated in bold and underline, respectively.

Dataset	Vanilla	Fixed	Random	Difficult	Not Different	DND (no w)	DND
News20	81.98$\pm$0.44	83.37$\pm$0.23	84.47$\pm$0.22	83.96$\pm$0.12	<u>84.65</u>$\pm$0.13	85.02$\pm$0.20	**85.19**$\pm$0.35
Review50	71.44$\pm$0.07	73.05$\pm$0.18	72.97$\pm$0.11	<u>73.52</u>$\pm$0.06	72.64$\pm$0.02	74.13$\pm$0.16	**74.87**$\pm$0.17

Table 5: Transferability of DND: test accuracy of the specified language model after fine-tuning across various augmentations. The augmentation policy learned with RoBERTa on News20 dataset is used as a base policy being transferred. All the values and error bars are mean and standard deviation across 3 random seeds. The best is indicated in bold.

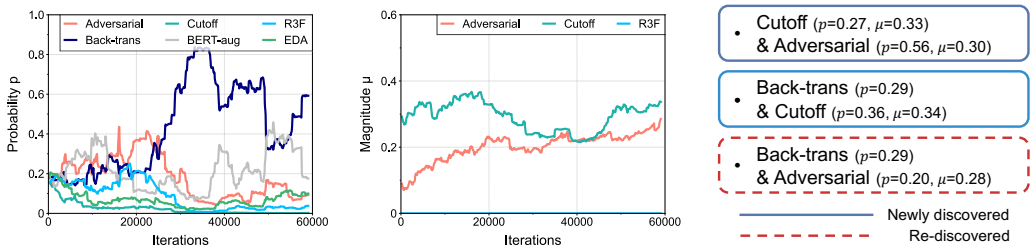
Model transfer	Dataset transfer	Vanilla	Fixed	Random	Transfer
RoBERTa	News20 $\rightarrow$ Review50	71.44$\pm$0.07	73.05$\pm$0.18	72.97$\pm$0.11	**73.93**$\pm$0.01
RoBERTa	News20 $\rightarrow$ Clinc150	95.48$\pm$0.21	96.50$\pm$0.07	96.31$\pm$0.04	**96.63**$\pm$0.07
RoBERTa $\rightarrow$ BERT	News20	84.11$\pm$0.58	84.65$\pm$0.12	86.01$\pm$0.06	**86.32**$\pm$0.02
RoBERTa $\rightarrow$ ALBERT	News20	79.70$\pm$0.23	80.03$\pm$0.10	79.37$\pm$0.31	**80.48**$\pm$0.08

4.3 Additional Analyses

Ablation study. To verify the effect of each component in our augmentation policy learning scheme, we compare the following augmentation methods: (1) *Vanilla*: no data augmentation; (2) *Fixed*: fixed augmentation is applied throughout the fine-tuning. We report the best performance among the baselines in Table 1; (3) *Random*: augmentation is randomly selected among the baseline augmentations at each training step; (4) *Difficult*: policy is only updated to increase the training loss of the augmented sample; (5) *Not Different*: policy is only updated to increase the similarity between augmented and original samples; (6) *DND (no w)*: policy is updated to increase both the training loss and semantic similarity for constructing difficult without re-weighting scheme in 1. From Table 4, one can verify that the effectiveness of *Difficult* is quite limited due to it easily diverges to construct too different examples as shown in Figure 1(b) and 6(b). Also, the augmented samples from *Not Different* do not much improve the performance as the learning signals from these samples would be duplicated with the original samples. However, when we properly combine these rewards, it successfully provides 'informative' learning signals to the classifier (*DND (no w)*) and can be further improved by the proposed re-weighting component (*DND*).

Transferability across datasets and architectures. Here, we provide additional experiments to verify the transferability of the learned policy with our method. To this end, we first save the status of the augmentation policy for each update during the training on the source data and then directly apply it to other tasks or models as dynamic augmentations. From Table 5, one can verify that the transferred augmentation outperforms the baseline augmentations and is even competitive with the performance of the jointly updated policy. Also, the learned policy could be transferred to the different models, such as BERT (Devlin et al., 2018) and ALBERT(Lan et al., 2019). These results imply that the proposed augmentation scheme would be greatly beneficial for successfully applying the data augmentation to the new task or model without the exhaustive search and update.

More analysis on learned policy. In Figure 3, we further present more analysis on the learned policy. Specifically, we focus on the parameters in the first operation $\mathcal{O}_1$. More detailed results can be found in Appendix B. We first note that the probability p is dynamically changed at an early stage, which implies the exploration step is naturally occurred. After enough iterations, most of the augmentations are converged, and few candidates are only actively updated. In contrast, each augmentation's magnitude μ is progressively updated and almost converged at the last training iterations. This stability is actually the result of our reward function for similarity, as the policy quickly diverges without this (see Appendix B). Interestingly, from successful guidance of the proposed reward, we observe that our augmentation policy could re-discover the previously known best composition of augmentations (which was found by exhaustive grid search). Also, it can newly discover the task-specific augmentations as shown in Figure 3(c). It reveals the reason for success from DND, along with the efficiency of this method.



(a) Dynamics of probability p (b) Dynamics of magnitude μ (c) Discovered augmentations

Figure 3: Analysis about the learned augmentation policy on Review50. (a, b) Dynamics of probability and magnitude during fine-tuning, respectively. (c) Top-3 representative augmentations, discovered by our augmentation policy. Best viewed in color.

(a) *Difficulty* and *Similarity*

Original (p: 0.98, s: 1.00)	Worked for about 5 minutes. Turned it off, wouldn't turn back on without **shaking** it. Cheesy **plastic**. Actually really **bright** when it did work.
Difficult & *Different* (p: 0.01, s: 0.73)	Ran for about 5 minutes. Turned it off, wouldn't turn back on without shaking it. Cheesy plastic. Actually really **cold** after it did run.
Easy & *not Different* (p: 0.96, s: 0.98)	Live for just 5 minutes. Turned it off, wouldn't turn back out without shaking it. Old plastic. Actually really bright when it did work.
Difficult & *not Different* (p: 0.53, s: 0.97)	Worked for about 5 months. Turned it off, wouldn't come back on without shaking it. Cheesy plastic. But looking bright when it did work.

(b) Example of augmented samples (Label: 'Lamp')

Figure 4: Qualitative results of difficult, but not too different samples; (a) Average *difficulty* (difficulty ↑ = confidence ↓) and *similarity* of the augmented samples under each augmentation method on News20. (b) Real examples on Review50. Different criteria are used for the selection, and two most salient words of original sentence are remarked in **bold**. Critically altered words are colored in **red**. Here, p and s represent the confidence and similarity, respectively. Best viewed in color.

Difficult, but not different samples. To further verify whether the augmented samples with the desired property are constructed from the learned policy, we construct the augmented samples using each augmentation method and compare the average difficulty and similarity in Figure 4(a). We observe that the augmented samples from our augmentation policy actually have a desired property: they show significantly low confidence, *i.e.*, more difficult, while it moderately preserves the semantic of original samples. Furthermore, we present the augmented samples from BERT-aug by varying the difficulty and similarity in Figure 4(b). Here, one can verify that the *difficulty* is solely not enough as too many changes can alter its semantic. Also, the *similarity* is solely not enough as it tends to construct too easy samples which can't provide 'informative' signals to the training model. However, by considering two aspects jointly, reasonable and effective augmented samples can be constructed. More qualitative results are in Appendix D.

5 Conclusion

We propose a simple and effective augmentation scheme for NLP tasks. Our main contribution is a design of a novel reward function for optimizing augmentation policy to (a) construct the difficult augmented samples for providing 'informative' signals, while (b) preserve their semantic meaning not too different from the original for avoiding 'wrong' signals. Then, the augmentation policy is optimized to maximize both (a) the model's training loss and (b) the semantic similarity not to find too different samples, with a continuous relaxation of discrete operations for efficient training. We further introduce a sample-wise re-weighting to the augmented samples, for learning them more effectively. Our augmentation method is much effective in challenging low-data and class-imbalanced regimes. Also, the learned augmentation policy is transferable across tasks and models. As the augmentation policy learning for NLP tasks is under-explored in the literature, we expect our work to contribute to exploring this direction of research.

ETHICS STATEMENT

Data augmentation is generally effective for improving the performance on the target task; hence it now becomes a de-facto standard for training DNNs, in various input domains (Cubuk et al., 2019; Park et al., 2019; You et al., 2020). However, it also has a potential risk to amplify the undesirable property of the model at the same time, *e.g.,* gender bias (Bordia & Bowman, 2019), since the enlarged diversity by data augmentation is proportional to the size of used samples for augmentation again. Since the practice of data augmentation is usually based on pre-defined fixed augmentations, it is hard to remove such inherency of undesirable properties explicitly.

In this respect, the learning-based augmentation scheme would be greatly beneficial, as it can easily alleviate this risk by setting the proper objective functions with respect to the desired property (Ziegler et al., 2019). For example, one could construct the 'fair' data augmentation scheme by jointly adding the reward for fairness and task accuracy, as we have explored in this work. Hence, we believe our proposed learning-based augmentation scheme would be a step towards improving a such fairness aspect in data augmentation, especially for NLP task.

REPRODUCIBILITY STATEMENT

We describe the implementation details of the method in Section 4.1 and Appendix A.3. Also, we provide the details of the datasets and baselines in Appendix A.1 and A.2, respectively. We also provide our code in the supplementary material. All the used packages are along with the code. In our experiments, we use a single GPU (NVIDIA TITAN Xp) and 8 CPU cores (Intel Xeon E5-2630 v4).

REFERENCES

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. Better fine-tuning by reducing representational collapse. In *International Conference on Learning Representations (ICLR)*, 2021.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 41–48, 2009.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Shikha Bordia and Samuel R Bowman. Identifying and reducing gender bias in word-level language models. *arXiv preprint arXiv:1904.03035*, 2019.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Zhiyuan Chen and Bing Liu. Mining topics in documents: standing on the shoulders of big data. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1116–1125, 2014.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. Advaug: Robust adversarial augmentation for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 113–123, 2019.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

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

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Siddhant Garg and Goutham Ramakrishnan. Bae: Bert-based adversarial examples for text classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Chengyue Gong, Dilin Wang, Meng Li, Vikas Chandra, and Qiang Liu. Keepaugment: A simple information-preserving data augmentation approach. *arXiv preprint arXiv:2011.11778*, 2020.

Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*, 2020.

Hongyu Guo, Yongyi Mao, and Richong Zhang. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*, 2019.

Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *International Conference on Computer Vision (ICCV)*, pp. 1–16. Springer, 2020.

Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom Mitchell, and Eric P Xing. Learning data manipulation for augmentation and weighting. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. AdvEntuRe: Adversarial training for textual entailment with knowledge-guided examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2418–2428, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1225. URL https://www.aclweb.org/anthology/P18-1225.

Jaehyung Kim, Jongheon Jeong, and Jinwoo Shin. M2m: Imbalanced classification via major-to-minor translation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Ken Lang. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pp. 331–339. Elsevier, 1995.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*, 2019.

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

Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M Robertson, and Yongxin Yang. Differentiable automatic data augmentation. In *International Conference on Computer Vision (ICCV)*, pp. 580–595. Springer, 2020.

Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

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

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

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

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*, 2020.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 761–769, 2016.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1631–1642, 2013.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pp. 194–206. Springer, 2019.

Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. A bayesian data augmentation approach for learning deep models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*, 2019.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Longhui Wei, An Xiao, Lingxi Xie, Xin Chen, Xiaopeng Zhang, and Qi Tian. Circumventing outliers of autoaugment with knowledge distillation. *arXiv preprint arXiv:2003.11342*, 2(8), 2020.

Sen Wu, Hongyang Zhang, Gregory Valiant, and Christopher Ré. On the generalization effects of linear transformations in data augmentation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 10410–10420. PMLR, 2020.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*, 2020a.

Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *International Conference on Learning Representations (ICLR)*, 2020b.

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

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.