Reducing and Exploiting Data Augmentation Noise through Meta Reweighting Contrastive Learning for Text Classification

Guanyi Mou[§]
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA, 01605
gmou@wpi.edu

Yichuan Li[§]
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA, 01605
yli29@wpi.edu

Kyumin Lee
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA, 01605
kmlee@wpi.edu

Abstract—Data augmentation has shown its effectiveness in resolving the data-hungry problem and improving model's generalization ability. However, the quality of augmented data can be varied, especially compared with the raw/original data. To boost deep learning models' performance given augmented data/samples in text classification tasks, we propose a novel framework, which leverages both meta learning and contrastive learning techniques as parts of our design for reweighting the augmented samples and refining their feature representations based on their quality. As part of the framework, we propose novel weight-dependent enqueue and dequeue algorithms to utilize augmented samples' weight/quality information effectively. Through experiments, we show that our framework can reasonably cooperate with existing deep learning models (e.g., RoBERTa-base and Text-CNN) and augmentation techniques (e.g., Wordnet and Easydata) for specific supervised learning tasks. Experiment results show that our framework achieves an average of 1.6%, up to 4.3% absolute improvement on Text-CNN encoders and an average of 1.4%, up to 4.4% absolute improvement on RoBERTa-base encoders on seven GLUE benchmark datasets compared with the best baseline. We present an in-depth analysis of our framework design, revealing the non-trivial contributions of our network components. Our code is publicly available for better reproducibility. [1]

Index Terms—text classification, data augmentation, meta learning, contrastive learning

I. INTRODUCTION

Data augmentation has demonstrated its effectiveness among lots of works, from image classification [1] to text classification [2]. It not only reduces the human efforts in labeling samples but also improves models' generalization ability [1]. It works by "synthesizing new data from existing training data" [3]. Specifically, it usually applies label-preserving transformations to the raw/original training data to create more samples and enrich the training dataset.

However, the quality and the contribution of augmented instances/samples[2] are not always guaranteed [4]. Augmented

samples can vary at a macro level in terms of their quality because augmentation methods have different ways to produce augmented samples. Some augmentation methods can be more effective than the others in either a certain task or generally various tasks [5]. Augmented samples can also vary instance-by-instance at a micro level – even the same augmentation method may not be effective on all raw instances. What is more, augmented samples generated by the same augmentation method and the same raw/original instance can also contribute differently because of their data quality difference. For example, in the binary sentiment classification [6], randomly deleting neutral words may not cause a significant impact on a review's sentiment, but deleting a negation word is likely to flip its label.

In search of finding proper ways to reduce the noise[3] from augmented instances, researchers also proposed solutions at macro and micro levels. At a macro level, researchers try to identify the effective augmentation methods and then apply them to the raw data. The identification of effective augmentation methods can be non-differentiable or differentiable. The non-differentiable approach treats different augmentation methods as a hyperparameter and selects the best one or groups them based on validation performance [7]. However, it may require domain knowledge to design or construct the candidate augmentation methods' pool [8]. In addition, these augmentation methods may also include additional hyperparameters to tune. It will impede the process of generating high-quality augmented samples. For example, CoDa [7] tried various combinations of stacking augmentations, which is in practice, trying out various sequential binary valued hyperparameters. On the other hand, the differentiable approach considers optimizing the augmentation method and downstream task model together through back-propagation on defined objective functions [8]–[10]. Although it does not require human prior knowledge to design augmentation methods, it constrains the potential augmentation methods to be differentiable. Thus, traditional heuristic augmentations with discrete transformation operations cannot be fitted in such a setting. Moreover, such

---

[§]Equal contribution.
[1]https://github.com/bigheiniu/BigData_MRCo
[2]We use terms of "instance" and "sample" interchangeably.

[3]We treat noise as the negative impact of low-quality augmented samples.

an approach would require additional backward propagation to update the parameters of augmentation methods. The whole process can be time-consuming and non-trivial to identify the suitable augmentation methods for different tasks. Thus the scope of augmentations is limited.

At the micro-level, researchers attempt to identify effective augmented samples. They consider the augmentation methods as black-box. Given the augmented samples, most of the works utilize heuristic rules to filter instances or training a reweighting module to reweight augmented instances' loss. In heuristic metric filtering methods, people introduce prior knowledge/domain expertise related to the tasks (e.g., readability). However, human expert knowledge can be suboptimal in many cases. In sample reweighting approaches [4], [5], [11], [12], they assign an importance score for each augmented sample and minimize the reweighted loss on these samples. One of the challenges in the reweighting approaches is how to optimize the reweighting module and downstream task model together, as directly minimizing vanilla reweighted loss makes the reweighted module collapse by setting the weights close to 0 to minimize the loss. Many reweighting approaches [5], [11], [12] consider utilizing the meta learning to convert the optimization into the bilevel optimization. Other works like Yi et al. [4] add regularization to encourage the weight towards uniform distribution to avoid the collapse. Ideally, the reweighting model will assign small weights to low-quality samples, thus neglecting the contribution of these samples.

In a nutshell, macro-level approaches of reducing the augmentation noise cannot be applied to all the augmented samples. In micro-level approaches, although they are accessible for all the augmentation methods, they will ignore the potential contribution of filtered out samples. We believe a better approach shall incorporate as many augmentation methods and as many augmented samples as possible (no seemingly obvious upper limit in capacity), with minimum constraints lifted (no obstacles for general methods) and less external knowledge required. These virtues are crucial in building a flexible and generalized framework that effectively incorporates any existing design in augmentations.

Inspired by such motivation, we thus propose a novel framework *Meta Reweighting Contrastive (*MRCO*)* model to *(1)* reduce the noise from augmented samples by reweighting the augmentation loss; *(2)* exploit the small weight augmented samples by contrasting the large and small weight augmented samples; and *(3)* be accessible to any off-the-shelf augmentation methods and text representation learning methods. As mentioned earlier, directly minimizing the reweighted loss will cause the reweight model collapse. It will take a shortcut by assigning the small weights for all the augmented samples. To overcome this issue, we consider the optimization of reweighted loss as the bilevel optimization, like recent meta reweighting works [5], [13], [14]. In the inner optimization loop, the main module, which is specific for the downstream task, is trained with the reweighted loss; while, in the outer optimization loop, the reweight module, also called meta reweighting module, assigns the proper weight for the augmented samples so the whole framework can achieve better performance on the held-out raw training dataset (meta dataset). To fully exploit the augmented samples, we apply contrastive learning [15] to refine augmented samples' representations. Specifically, we consider any low-weighted augmented samples within the same class as the negative samples (wider global restriction on negative samples), while the samples from the same originated samples as the positive samples (stricter local restriction on positive samples). This fine-grained contrastive learning allows the model to reinforce the difference between high-quality samples and low-quality samples. Overall, MRCO differs from prior works in resolving the restrictions from many perspectives while still reaching high-level performance boosts: *(1)* It differs from macro-level augmentation noise reduction because it is capable of embracing as many augmented samples as possible; and *(2)* It also differs from the micro-level approach in terms of including all the augmented samples into model training. In summary, the contributions of this work are four-fold:

- To the best of our knowledge, we are the first to propose a framework combining both meta reweighting and contrastive learning in one unified deep learning model. Our framework leveraged bilevel optimization for automatically learning the weight of each augmented sample based on its quality. Then, our contrastive learning component leveraged the quality information to narrow down the gap between original data (golden, best quality) and the high-quality augmented data while enlarging the distance between the original data and low-quality augmented data.

- To fully utilize the weight/quality information in the contrastive design, we proposed a novel weight-dependent dequeue-enqueue algorithm called *Lifetime Aware Smallest Weight (LASW)* algorithm. We elaborate the details of our design in Section III.

- Our framework design lifted the minimum requirements to its surrounding components and generalized well: (1) It expects no specific data distribution (no distribution function leveraged); (2) It treats augmentation methods as black boxes; (3) It embraces as many augmentation methods and augmented samples as possible; and (4) It requires minimum adaptation efforts as a plug-in to any off-the-shelf classification models (i.e., it is highly cooperative with existing designs/models).

- We conducted extensive experiments on seven GLUE benchmark datasets. As a result of the advantages in the design mentioned above, our framework achieved an average of 1.6%, up to 4.3% absolute improvement on Text-CNN encoders, and an average of 1.4%, up to 4.4% absolute improvement on RoBERTa-base encoders on the benchmark datasets compared with the best baseline. A detailed analysis also provided more insights.

We organize the rest of this paper in the following sections. We formulate common terminologies in Section II. We then introduce our designs in Section III. Experiment results are
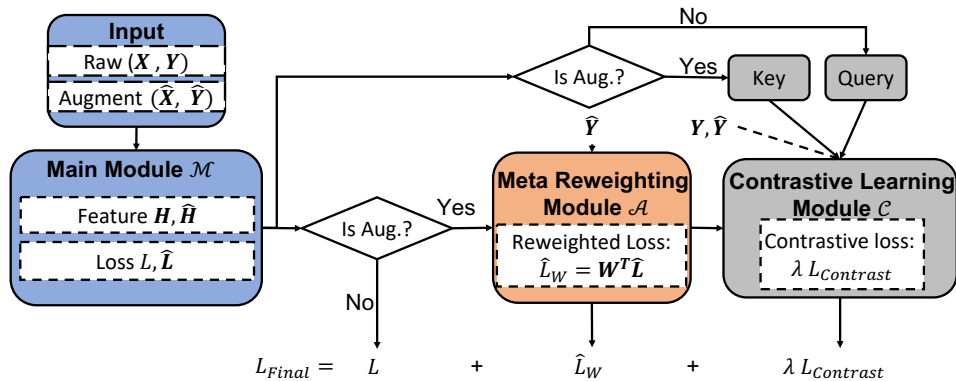
Fig. 1: A high-level view of the overall framework.

described in Section IV. Further analysis is presented in Section V. We describe related work in Section VI. Lastly, we conclude our research in Section VII.

II. NOTATION TERMINOLOGY

In this paper, we utilize the bold uppercase letters to denote the vectors or matrices (e.g., $\mathbf{Y}$), chancery fonts to represent the sets (e.g., $\mathcal{D}$) and calligraphic fonts to denote the module (e.g., $\mathcal{M}$). For other notations, we will illustrate them in the relevant sections. Formally, given the raw/original samples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}|}$ and augmented samples $\hat{\mathcal{D}} = \{(\hat{x}_j, \hat{y}_j)\}_{j=1}^{|\hat{\mathcal{D}}|}$, we train the classification model $\mathcal{M}$. The $x/\hat{x}$ is an instance of text and $y/\hat{y}$ is the class label. $(\mathbf{X}, \mathbf{Y})$ and $(\hat{\mathbf{X}}, \hat{\mathbf{Y}})$ are the mini-batch for optimization. For the raw dataset $\mathcal{D}$ and augmented dataset $\hat{\mathcal{D}}$, there are $|\mathcal{Y}|$ different classes.

III. FRAMEWORK

Given original/raw input text and their task-specific label $(\mathbf{X}, \mathbf{Y})$ in benchmark datasets, augmentation methods will generate augmented samples represented as $(\hat{\mathbf{X}}, \hat{\mathbf{Y}})$. While we generally believe the raw inputs are of golden quality, those augmented samples often contain noise. Our framework will consume both raw inputs and the augmented samples and automatically minimize the negative impact of noise within augmented samples. Thus, eventually, such a design will effectively boost the prediction/classification performance concerning the given task. A high-level view of our overall framework is shown in Fig. 1. It mainly contains 3 parts:

- *Main module:* it learns the text representation $\mathbf{h}_i$ of the input x_i and make the prediction $p(y_i)$ for x_i and the main module can be any off-the-shelf text encoder;
- *Meta reweighting module:* to reduce the noise brought by low-quality augmented samples, it assigns a weight for each augmented sample's loss, where the high-quality augmented samples will receive large weights while low-quality augmented samples will receive small weights.
- *Contrastive learning module:* it utilizes contrastive learning to enlarge the distance between high-quality augmented samples and low-quality augmented samples on weights. This helps the model fully exploit the low-weight

augmented samples, which the meta reweighting learning model will tend to ignore.

We elaborate on the last two modules in the following sections.

A. Meta Reweighting Module

Reweight-Plugin: Researchers often assumed that the outcome of data augmentations was class-invariant. However, this assumption does not always hold, and simply treating all the augmented samples with the same importance is suboptimal [16]. In addition, in some cases, although the augmented samples are consistent with the label, they introduce unrecognized noise as a side effect. The noise level within the augmented samples is not uniformly distributed, and it hurts the model's final performance.

Inspired by the previous work [14], [17], to properly exploit these augmented samples, we propose a new reweight module $\mathcal{A}$, assigning the importance weight $\mathbf{W}$ to these augmented samples. The reweight module will be only applied to augmented samples $\hat{\mathcal{D}}$ during training, while it will be discarded for the raw samples $\mathcal{D}$ or during evaluation. Concretely, given the augmented samples' hidden representations $\hat{\mathbf{H}}$ and labels for these samples $\hat{\mathbf{Y}}$, the reweight process is formed as:

$$
\begin{aligned}
\mathbf{W} &= \mathcal{A}(\hat{\mathbf{H}}, \hat{\mathbf{Y}}) \\
\mathcal{A} &\to \sigma\left(\text{MLP}([\hat{\mathbf{H}}; \mathcal{E}(\hat{\mathbf{Y}})])\right)
\end{aligned}
\tag{1}
$$

where σ is the *sigmoid* activation function, $[\cdot\,;\cdot]$ is vector concatenation, MLP is a multi-layer perceptron, and $\mathcal{E}$ is the embedding layer of the label, mapping each label into a dense vector $\hat{\mathbf{h}}_{label} \in \mathbb{R}^d$. In practice, we found that, for MLP, a shallow network structure ($\#layer < 3$) with dropouts is sufficient in reaching satisfying performance. The additional label embedding $\mathcal{E}$ allows us first learn the global representation of each class, and then allows MLP to capture the interaction between augmented samples' hidden representation $\hat{\mathbf{H}}$ and given class $\hat{\mathbf{Y}}$.

Objective Functions: The objective function of classification is the linear combination of raw samples' loss L and the
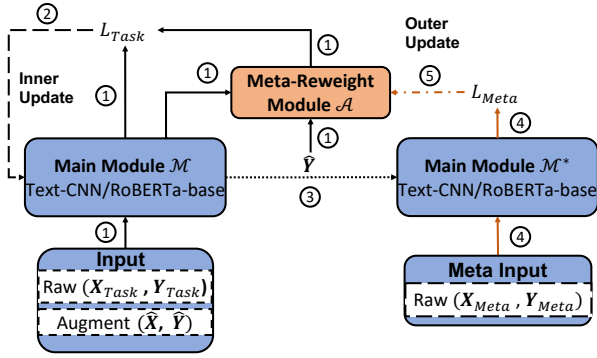
Fig. 2: Approximated bilevel optimization procedure for meta reweighting module $\mathcal{A}$: ① utilize raw instances $(\mathbf{X}_{Task}, \mathbf{Y}_{Task})$ and augmented instances $(\hat{\mathbf{X}}, \hat{\mathbf{Y}})$ to do the forward pass for L_{Task}; ② backward pass for L_{Task} on $\mathcal{M}$, retaining the backward propagation graph for $\mathcal{A}$; ③ update main module $\mathcal{M}$ to $\mathcal{M}^*$ through backward propagation; ④ utilize the raw meta input $(\mathbf{X}_{Meta}, \mathbf{Y}_{Meta})$ forward pass for L_{Meta}; ⑤ backward pass for L_{Meta} to update meta reweighting module $\mathcal{A}$.

reweighed augmentation loss $\hat{L}_W$:

$$
\begin{aligned}
L_{Task} &= L + \hat{L}_W \\
&= L + \mathbf{W}^T \hat{\mathbf{L}} \\
&= \mathbb{E}_{(x,y)\in\mathcal{D}} l(y, p(y)) + \mathbb{E}_{(\hat{x},\hat{y})\in\hat{\mathcal{D}}, w\in\mathbf{W}}\, w\, l(\hat{y}, p(\hat{y}))
\end{aligned}
\tag{2}
$$

where l is the cross-entropy loss and $\hat{\mathbf{L}}$ is the loss vector for all the augmented samples without reduction. Directly minimizing Eq. 2 will cause the model to collapse as the reweight plugin $\mathcal{A}$ will assign small (even zero) weights for all the augmented samples to reduce $\hat{L}_w$ without meaningful learning, and the main module will only learn from the raw samples since the gradients from the augmented samples are negligible.

Meta-Optimization: To prevent the collapse problem of reweight module $\mathcal{A}$ during optimization, we follow recent work on sample loss reweighting [18], [19], formulating this problem as a bilevel optimization problem. The optimal reweight module $\mathcal{A}$ should achieve good performance on unseen raw samples. To evaluate the effectiveness of the reweight module, we hold out a subset $\mathcal{D}_{Meta} \subseteq \mathcal{D}$. After main module $\mathcal{M}$ and reweight module $\mathcal{A}$ trained on $\mathcal{D}_{Task} = \mathcal{D} - \mathcal{D}_{Meta}$ and relevant augmented samples $\hat{\mathcal{D}}_{Task}$, the main module $\mathcal{M}$ should achieve better performance on $\mathcal{D}_{Meta}$. Thus, the objective function of bilevel optimization is formulated as:

$$
\begin{aligned}
&\min_{\theta_{\mathcal{A}}} L_{Meta}\left(\theta^*_{\mathcal{M}}(\theta_{\mathcal{A}}), \theta_{\mathcal{A}}, \mathcal{D}_{Meta}\right) \\
&\text{s.t. } \theta^*_{\mathcal{M}}(\theta_{\mathcal{A}}) = \operatorname{argmin}_{\theta_{\mathcal{M}}} L_{Task}(\theta_{\mathcal{M}}, \theta_{\mathcal{A}}, \mathcal{D}_{Task}, \hat{\mathcal{D}_{Task}})
\end{aligned}
\tag{3}
$$

where $\theta_{[\cdot]}$ is the parameters of the relevant module. The outer loop update for reweight module $\theta_{\mathcal{A}}$ will require the optimal main module $\theta^*_{\mathcal{M}}$ at every iteration, which is computationally infeasible. To reduce the computation requirement, we approximate optimal $\theta^*_{\mathcal{M}}$ by one-step stochastic gradient descent (SGD):

$$
\theta^*_{\mathcal{M}}(\theta_{\mathcal{A}}) \approx \theta_{\mathcal{M}} - \alpha \nabla_{\theta_{\mathcal{M}}} L_{Task}(\theta_{\mathcal{M}}, \theta_{\mathcal{A}}, \mathcal{D}_{Task}, \hat{\mathcal{D}})
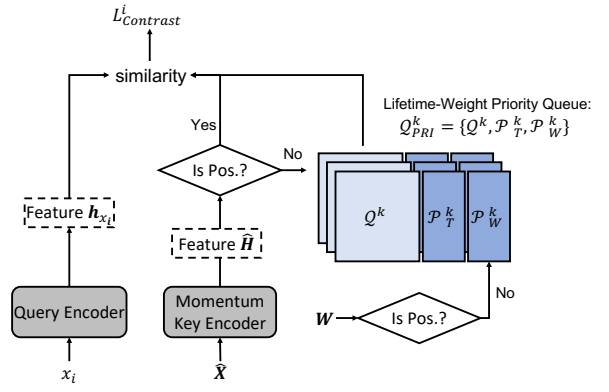\tag{4}
$$

Fig. 3: The structure of contrastive learning module.

where α is the meta learning rate. The approximated bilevel optimization procedure is illustrated in Fig. 2, the inner optimization (steps ① to ③) is task-dependent, optimizing the parameters $\theta_{\mathcal{M}}$ by one-step SGD. The outer optimization (steps ④ and ⑤) tries to update the meta reweighting module $\mathcal{A}$ parameters $\theta_{\mathcal{A}}$ based on the updated main module $\theta^*_{\mathcal{M}}$ so the inner learning algorithm can better fit outer objective L_{Meta}.

B. Contrastive Learning Module

Theoretically, the optimal meta reweighting module $\mathcal{A}$ will assign small weights for low-quality samples. Although this allows the main module $\mathcal{M}$ to ignore the influence of the noise brought by these low-quality samples (negligible gradients in the model's back-propagation), it also neglects the potential contribution of the noised samples. Inspired by the recent work in contrastive learning [15], [20], contrasting the distance among the query instance, negative instances, and positive instances to learn better feature representation, we believe the small weights can be emphasized by contrastive learning. Generally, we consider the small weighted augmented samples as negative instances while large weighted ones as positive instances. By contrasting the large weighted and small weighted augmented samples, these small weighted augmented samples can further contribute to better feature representation learning.

1) Prior Knowledge of Contrastive Learning: Given a query sample x_i, positive augmented samples $\hat{\mathbf{X}}^i_{Pos}$, and negative augmented samples $\hat{\mathbf{X}}_{Neg}$, the overall objective function of contrastive learning is:

$$
\begin{aligned}
L_{Contrast} &= -\mathbb{E}\left[\frac{1}{|\hat{\mathbf{X}}^i_{Pos}|} \sum_{\hat{x}^i_p \in \hat{\mathbf{X}}^i_{Pos}} \log P_{Contrast}\right] \\
P_{Contrast} &= \frac{\exp\left(\mathbf{h}^T_{x_i} \mathbf{h}_{\hat{x}^i_p}\right)}{\sum_{\hat{x}\in\hat{\mathbf{X}}_{Neg}\bigcup\{\hat{x}_p\}} \exp\left(\mathbf{h}^T_{x_i}\mathbf{h}_{\hat{x}}\right)}
\end{aligned}
\tag{5}
$$

Negative samples are not instance dependent, as $\hat{\mathbf{X}}_{Neg}$ are usually stored into a large queue [15] and can be shared across instances, while the positive samples $\hat{\mathbf{X}}^i_{Pos}$ are usually originated from query sample x_i. To achieve effective performance, the existing approaches [7], [15] often include a

---

Algorithm 1: LASW Dequeue-Enqueue Algorithm

Input : Weight-lifetime priority queues $\mathcal{Q}_{PRI} = \{\mathcal{Q}_{PRI}^k\}_{k=1}^{|\mathcal{Y}|}$, where $\mathcal{Q}_{PRI}^k = (\mathcal{Q}^k, \mathcal{P}_W^k, \mathcal{P}_T^k)$, maximum lifetime τ, augmented mini-batch $\{\hat{\mathbf{H}}, \hat{\mathbf{Y}}\}$ and weight $\mathbf{W}$.

Output: Updated $\mathcal{Q}_{PRI}$.

   for k in $\mathcal{Y}$ **do**

      Choose $\hat{\mathbf{H}}^k$ and $\mathbf{W}^k$, where $\hat{\mathbf{Y}} == k$;   {Select enqueue candidates}

      $\mathcal{P}_T^k = \mathcal{P}_T^k - 1$;   {Update instance's lifetime}

      Dequeue N instances from $\mathcal{Q}^k$, where $\mathcal{P}_T^k <= 0$ and discard them;   **{Lifetime aware dequeue-enque}**

      Enqueue $Top - N$ smallest weight instances $\hat{\mathbf{H}}_N^k$ from $\hat{\mathbf{H}}^k$ by $\mathbf{W}^k$;

      for $\hat{\mathbf{h}}^k$ in $\hat{\mathbf{H}}_{\neg N}^k$ **do**

         if $len(\mathcal{Q}^k[\mathcal{P}_W^k > \mathbf{W}_{\neg N}^k]) > 0$ **then**

            Dequeue the largest weighted instance from $\mathcal{Q}^k[\mathcal{P}_W^k > \mathbf{W}_{\neg N}^k]$ and discard it;   **{Small weight dequeue-enque}**

            Enqueue $\hat{\mathbf{h}}^k$;

         end if

         Set $\mathcal{P}_T^k = \tau$ and $\mathcal{P}_W^k = \mathbf{W}_{\neg N}^k$ for newly enqueued samples from $\hat{\mathbf{H}}_{\neg N}^k$;

      end for

   end for

---

large queue $\mathcal{Q}$ to store the historical hidden representation of negative augmented samples $\hat{\mathbf{H}}_{Neg}$, and a Siamese network design to encode the query sample x_i and augmented/key samples $\hat{\mathbf{X}} \supseteq \hat{\mathbf{X}}_{Pos}^i + \hat{\mathbf{X}}_{Neg}$. Concretely, let $\theta_{\mathcal{M}}$ and $\hat{\theta}_{\mathcal{M}}$ be the parameters of query encoder (i.e., the main module) and key encoder, respectively. We update the key encoder's parameters with momentum rule:

$$\hat{\theta}_{\mathcal{M}} = \gamma \hat{\theta}_{\mathcal{M}} + (1 - \gamma)\theta_{\mathcal{M}} \tag{6}$$

where γ is the ratio controlling the update step size.

2) Weight dependent Contrastive Learning:

Difference from prior works: Existing works take augmented samples originated from different raw samples [15] or different classes [21] as negative samples. Our design, however, consider small weighted augmented samples within the same class as negative samples. The purpose of requiring "within the same class" is two-fold. First, it is infeasible to compare the weight across classes. For example, in sentiment classifications, a small weighted positive augmented sample may have less contribution on meta-dataset $\mathcal{D}_{Meta}$. However, it is not necessarily close to the negative class. Second, the class difference may dominate the weight difference [21], and the contrastive learning module will only capture the coarse class difference however ignore the fine-grained weight difference. What is more, existing works are limited by their simple queue structure and FIFO dequeue-enqueue operations. We, however, designed lifetime-weight priority queues for different classes as discussed below.

Our design: We show our new contrastive learning in Fig. 3. Concretely, given a query sample x_i in class k: (1) we propose the small weighted augmented samples also in class k as the negative augmented samples $\hat{\mathbf{X}}_{Neg}$. They will be stored in the negative queue $\mathcal{Q}_{PRI}^k$; and (2) following [7], [15], we consider the large weighted augmented samples originated from x_i as positive augmented samples $\hat{\mathbf{X}}_{Pos}^i$. It is worth noting that x_i, $\hat{\mathbf{X}}_{Pos}^i$ and $\hat{\mathbf{X}}_{Neg}$ are from the same class k.

Our new approach allows us to capture the implicit knowledge from the noisy samples, and provides a fine-grained way to learn better feature representation inside each class. We designed lifetime-weight priority queues for $|\mathcal{Y}|$ different classes $\mathcal{Q}_{PRI} = \{\mathcal{Q}_{PRI}^k\}_{k=1}^{|\mathcal{Y}|}$. Specifically, for class k, the priority queue is $\mathcal{Q}_{PRI}^k = (\mathcal{Q}^y, \mathcal{P}_T^k, \mathcal{P}_W^k)$, where $\mathcal{Q}^y$ will store the negative samples, $\mathcal{P}_W^k$ is the lifetime priority index and $\mathcal{P}_W^k$ is the weight priority index. In addition, we also design a Lifetime Aware Small Weight (LASW) dequeue-enqueue algorithm to properly keep a queue of up-to-date small weight negative instances.

LASW dequeue-enqueue Algorithm. The LASW algorithm is presented in Algorithm 1. It contains two pairs of dequeue-enqueue operations. For each class k, it will firstly do the *Life time aware dequeue-enqueue* in step 4 and 5. It will dequeue instances from $\mathcal{Q}_{PRI}^k$ when instances' lifetime $\mathcal{P}_T^k$ less than 0. After the dequeued process, it will enqueue the same amount of samples from the mini-batch in weight ascending order. This can help us keep the instances inside the queue up-to-date and with small weight. The second dequeue-enqueue operation is named as *small weight dequeue-enqueue*, from step 6 to 12. It will compare the not enqueued instances $\hat{\mathbf{H}}_{\neg N}^k$ with all the instances inside $\mathcal{Q}^k$ on weight $\mathcal{W}_{\neg N}^k$, dequeue the largest weighted instances from $\mathcal{Q}^k[\mathcal{P}_W^k > \mathbf{W}_{\neg N}^k]$ and enqueue these instances from the mini-batch. After every enqueue operation, the module will reset the lifetime priority to τ and set the weight priority from $\mathbf{W}_k$ for these enqueued instances.

Positive sample selection. Prior works generally consider all relevant augmented samples as valid positive samples of the raw sample. Such treatment can bring up problems with low-quality augmented samples, where the noise within these augmented samples can overwhelm the contributions or even cause label-flipping problems. We set up an extra selection ratio $\rho \in [0, 1]$ as a hyperparameter, which will choose the top ρ augmented samples from the mini-batch as positive samples concerning their calculated weights.

C. Overall Objective Function

To learn a better feature representation for the downstream tasks, we consider integrating the classification loss and con-

TABLE I: Important hyperparameters used in our contrastive learning module.

Name	Notation	Brief Description
Positive Ratio	ρ	The proportion of high-weight augmented samples treated as positive samples.
Queue Size	N_{Neg}	The queue size of negative (low-weight) samples for each class.
Survive-Time-slot	τ	The maximum time for an augmented sample to live in a queue.
Importance Score	λ	The weight of contrastive loss $L_{Contrast}$ in the overall loss function Eq. 7.

TABLE II: Dataset Statistics.

Task	RTE	MRPC	CoLA	SST-2	QNLI	QQP	MNLI-m
Train	2.5k	3.7k	8.6k	67.4k	105k	364k	393k
Aug.	13.7k	22.0k	68.4k	533k	627k	2.18m	2.34m
Aug. Ratio	5.50	6.00	8.00	7.91	5.98	5.97	6.00
Dev	278	408	1.1k	873	5.5k	40k	9.8k

trastive loss and optimize them together. We can define the final objective function as:

$$L_{Final} = L_{Task} + \lambda\, L_{Contrast} \quad (7)$$

where λ is the hyper-parameter to control the importance of contrastive learning in the tasks. If $\lambda = 0$, this means the model only reduces the noise from augmented samples by reweighting these augmented samples. It should be noticed that doing the backpropagation on L_{Final} will only update the main module $\mathcal{M}$ to avoid reweight module $\mathcal{A}$ collapse. We update $\mathcal{A}$ by Eq. 3 instead of Eq. 2.

Important Hyperparameters. To further help readers better understand our novel design in the contrastive learning module, we summarize our newly introduced/invented hyperparameters in Table I with their names, notations, and brief descriptions. We will also conduct a deeper analysis of these hyperparameters in Section V to reveal their non-trivial contributions as well as the impact of varying their values in terms of overall performance.

IV. EXPERIMENT

With all the above descriptions and explanations of our network design, we can now verify the effectiveness of our framework MRCO. We conduct experiments on tasks from General Language Understanding Evaluation (GLUE) benchmark datasets [22]. We compare MRCO with the state-of-the-art approaches on learning from noised augmented data.

A. Datasets and Setup

Datasets. The natural language understanding benchmark dataset GLUE [22] contains 9 tasks/sub-datasets, ranging from single-sentence tasks to similarity and paraphrase tasks and inference tasks. Following CoDa [7]'s setting, we only focus on 7 out of the 9 tasks: CoLA, SST-2, MRPC, QQP, MNLI-m, QNLI, and RTE. Note that WNLI was not considered in CoDa, and STS-B is a regression task that is not the scope of this paper. We train the models on the given training set and report the results on the given development set. Evaluation metrics are predefined/given by each task. A brief statistics description of these tasks is shown in Table II. The amount of augmented data depends on the length of each raw sample and the task properties. For example, a raw instance with a longer text string will be easier to augment and thus have more augmentation variations than shorter raw instances.

Setup. We utilize the Text-CNN [23] and RoBERTa-base [24] as our backbone/encoder main module for text feature extraction and classification. However, MRCO can easily be applied to other text encoders. We use Adam as our optimizer. For GLUE benchmark datasets, we utilize 5 representative data augmentation methods to augment the raw data, namely:

- *Wordnet* [25]: a method using Wordnet as a dictionary of synonyms to replace candidate words.
- *Easydata* [16]: a method using the combination of word synonym replacement, random word insertion, random word swap, and random word deletion.
- *Checklist* [26]: a method using dictionary-based named entity substitution.
- *Embedding* [27]: a method using word embeddings to search similar words for candidate word substitutions.
- *Charswap* [28]: a method with character-level noise injection methods (insertion, swap, deletion, and replacement).

One can easily extend our framework to other augmentation methods as well. As we stated earlier, one of the advantages of our framework is that, given any main module encoder and any augmented samples, MRCO shall be able to enhance the model performance further.

B. Baseline Methods

Since our framework is augmentation agnostic, we compare the baseline methods to understand which method best exploit the augmented samples. We include two branches of baseline methods: contrastive learning and vanilla approaches.

Contrastive Learning Approaches: CERT [29] used MoCo [15] and the augmented data for contrastive learning in the model pretraining phase, and then finetuned the adjusted pretrained model on the downstream task. It utilized a queue to store large amounts of negative samples and momentum update to learn a consistent feature representation of the en-queued negative samples. It considered the augmented samples from the same raw samples as positive samples and augmentation from other ones as negative ones. **SCL** [30], a recent work, leveraged supervised contrastive learning in the finetuning phase. It considers the augmented samples from

TABLE III: Experiment results on GLUE benchmark dataset in format avg_{std}. Best results: **BOLD**. Second-best: <u>UNDERLINED</u>. Overall result is the row average of all tasks. All results rounded to precision of 0.001.

Methods	RTE(ACC)	MRPC(ACC)	CoLA(MCC)	SST-2(ACC)	QNLI(ACC)	QQP(ACC)	MNLI-m(ACC)	Overall
Text-CNN	$.515_{.010}$	$.702_{.006}$	$.187_{.013}$	<u>$.837_{.003}$</u>	$.604_{.005}$	<u>$.791_{.001}$</u>	$.542_{.001}$	.598
+ Aug.	$.524_{.002}$	$.698_{.005}$	$.194_{.011}$	$.830_{.002}$	$.600_{.001}$	$.794_{.002}$	$.528_{.001}$	.594
+ Aug. & Filter	<u>$.527_{.011}$</u>	<u>$.724_{.000}$</u>	<u>$.202_{.004}$</u>	<u>$.855_{.001}$</u>	<u>$.615_{.001}$</u>	<u>$.796_{.001}$</u>	<u>$.551_{.001}$</u>	<u>.610</u>
MRCo	**$.534_{.001}$**	**$.735_{.002}$**	**$.213_{.001}$**	**$.866_{.001}$**	**$.632_{.001}$**	**$.839_{.002}$**	**$.565_{.001}$**	**.626**
RoBERTa-base	$.713_{.023}$	$.882_{.005}$	$.566_{.013}$	$.944_{.003}$	$.924_{.007}$	$.912_{.001}$	$.875_{.001}$	.830
+ Aug.	$.742_{.001}$	$.880_{.004}$	$.563_{.001}$	$.953_{.002}$	$.921_{.001}$	$.915_{.001}$	<u>$.876_{.002}$</u>	.836
+ Aug. & Filter	<u>$.744_{.002}$</u>	<u>$.883_{.004}$</u>	$.583_{.007}$	$.956_{.001}$	$.929_{.001}$	$.915_{.001}$	$.875_{.002}$	<u>.845</u>
CERT (origin)	$.722_{--}$	$--$	<u>$.629_{--}$</u>	$.936_{--}$	$.923_{--}$	$.914_{--}$	$.863_{--}$	$--$
SCL (origin)	$--$	$--$	$--$	$.949_{.006}$	$.925_{.004}$	$--$	$.853_{.005}$	$--$
MRCo	**$.760_{.004}$**	**$.902_{.009}$**	**$.673_{.002}$**	**$.960_{.001}$**	**$.930_{.002}$**	$.894_{.001}$	**$.893_{.001}$**	**.859**

different classes as the negative samples, while the positive samples are obtained from the inner-class augmented samples.

Vanilla Approaches: This part includes the more straightforward cases of the plain main module (Text-CNN and RoBERTa-base) and their performance with augmentations with (+ *Aug & Filter*) or without (+ *Aug*) manual empirical filter mechanisms. More specifically, we used Flesch Reading Ease [31] readability score as the filter metric. We set the upper limit and lower limit of the score as hyper-parameters. Augmented samples with scores between the two limits will be kept while the framework will omit others. We find the best-performing combination with a thorough grid search.

C. Main Results

We show the detailed experiment results in Table III, where we used the abbreviations "ACC" standing for accuracy and "MCC" standing for Matthews Correlation Coefficient. We run each experiment 5 times with different random seeds and report the average and the standard deviation in the format of avg_{std}. For each different network backbone/encoder (*main module*), we mark the best performing average result in bold and the second-best underlined. All results are rounded to a precision of 0.001. Rows noted with *(origin)* are results directly obtained from the referenced source papers, where some results were unavailable/not provided by the authors. In this case, we marked the blank space as "$--$" (i.e., the authors of CERT and SCL papers did not test their models over some GLUE tasks). GLUE tasks are arranged in the same order as Table II, where their training data volumes grow from left to right. We also calculate the macro average across tasks in the last column except for CERT and SCL because it is not fair to calculate the overall average/performance based on only certain task results. Through these experiment results, we found several interesting observations:

Overall result: MRCo showed superior performance against all the baselines and variations in almost all tasks. Most of the improvements between the best and second-best results are nontrivial concerning the averages and standard deviations. The most significant improvement is MRCo with RoBERTa-base under CoLA (MCC), where we observed a 4.4% absolute performance boost on average compared with the best baseline. The performance boost is especially larger when the size of the original training set is relatively small (tasks on the left of the table). On the RoBERTa-base encoder, the performance enhancement ranges from 1.6% to 4.4% on the top 3 smallest tasks, while for the remaining 4 tasks, the improvement ranges from 0 to 1.7%. This meets our intuition as deep learning models generally perform better with more provided data.

MRCo across different backbones: MRCo is effective on both backbone main module architectures, showing an improvement across tasks. MRCo made an absolute performance boost of 2.8% on Text-CNN backbones and 3.9% on RoBERTa-base backbones, indicating that our model is highly adaptive to a wide range of deep learning designs.

Plain modules vs. + *Aug*.: It seems there are apparent differences in the capability of embracing augmented data. In general, RoBERTa-base methods are better at incorporating the augmented samples, thus show overall performance improvement on the given metrics, while Text-CNNs struggle on several tasks. In particular, on SST-2, QQP, and QNLI, Text-CNN performed better without augmentations. These observations might point to the differing capability of models in tolerating noise within the data while still extracting useful information. Once again, as we observed before, the augmentations are less effective on tasks with larger training sets.

+ *Aug*. vs. + *Aug. & Filter*: Generally speaking, augmentations without filters is actually a special case for applying filters where the limits are set to $(-\inf, \inf)$. Moreover, the original baseline without any augmentation can also be deemed a special case where the filtering rules are so strict that none of the augmented samples meet the requirement, returning an empty set on augmented samples. We found filters were helpful under most tasks. It is worth noting that on certain tasks, the + *Aug*. was not helpful, whereas + *Aug. & Filter* managed to reverse the trend and boosted the performance (e.g., MRPC on both encoders and CoLA on RoBERTa-base).

MRCo vs. + *Aug & Filter*: Although + *Aug. & Filter* boosted the performance in most tasks compared with the original backbones (i.e., Text-CNN and RoBERTa-base with or without augmentations), our model's reweighting strategy performed even better than it. The largest improvements over + *Aug. & Filter* were 4.7% on Text-CNN and 4.4% on RoBERTa-base. The overall improvements are also strong, varying from 1.6% on Text-CNN to 1.4% on RoBERTa-base. One may easily observe that our MRCo dominated + *Aug. & Filter* across most tasks with only two exceptions: with RoBERTa-base as

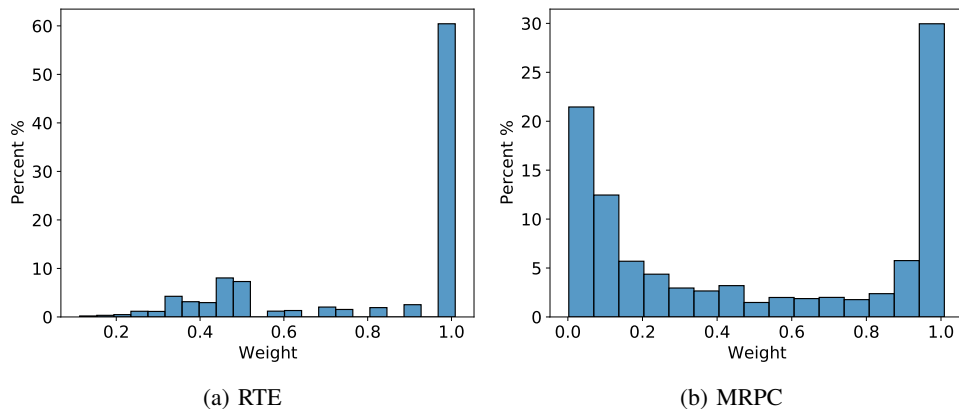|	
(a) RTE	(b) MRPC

Fig. 4: Augmented samples weight distribution from meta reweighting module with basic encoder RoBERTa-base.

the backbone, MRCO reached the same level of performance as + *Aug. & Filter* on QNLI, as there is few statistical difference between the two models; and on QQP, MRCO slightly under-performed against + *Aug. & Filter*. Such results reflected a similar conclusion as the prior works [15], [29], [30], where researchers found meta learning and contrastive learning tended to have more significant improvements on smaller datasets compared with larger datasets.

MRCO VS. existing approaches: We tried to apply the given code provided by CERT [29], but we were not able to achieve satisfying results/the same results that the authors reported. Therefore, we only reported the original results published in CERT(origin). It is worth noting that the original paper leveraged a slightly different encoder BERT-base and a more advanced augmenting method (i.e., back-translation). Overall, the performance improvement of our framework over CERT is still apparent. A similar conclusion can be applied to SCL (origin), where MRCO also achieved better performance across tasks. We also tried to implement and run our version of other baselines such as MMEL [4] and CoDa [7] since they did not release their source code. Unfortunately, the models poorly performed compared with other baselines (e.g., CERT(origin) and SCL(origin)). Therefore, we do not report their results.

V. ANALYSIS

In this section, we conduct extra experiments to shed more light on insights into our framework's success. We show a detailed analysis of each component's contribution in our framework and the training dynamics of MRCO.

A. Meta Reweighting Module Analysis

To understand whether the meta reweighting module can effectively capture the information within noisy data, we visualize the output weights of augmented samples generated from RTE and MRPC datasets. In Fig. 4, we find that augmented samples of these two datasets have different weight distribution patterns. In RTE task, most of the augmented samples receive weights close to 1, while in MRPC task, many augmented samples receive weights less than 0.5. As we already showed in Table III and stated in Section IV-C, in

RTE task, the model with all the augmentations +*Aug* achieved better performance than the model without augmentations. Such trend does not apply on MRPC, where augmentations +*Aug* were not helpful. This indicates that the quality of MRPC's augmented data is not as good as RTE's one. Despite such a disadvantage, MRCO still improved its performance in both tasks by differentiating noisy samples via assigning them with lower weights. The weight distribution analysis confirmed the effectiveness of our meta reweighting module.

B. Contrastive Learning Analysis

1) Hyperparameter Analysis: To better understand which parameters play an essential role in our contrastive learning module, we conduct several hyperparameter analysis on the number of negative samples N_{Neg}, the importance of contrastive learning loss λ, survive-time-slot τ and positive ratio ρ. Fig. 6 shows how the evaluation result changed when we vary these hyperparameters under RoBRETa-base encoder. In the following paragraphs, we describe their impacts on experiment results by referring to column-wise subfigures in Fig. 6.

Impact of queue size N_{Neg}: Existing works in contrastive learning [7], [15], [29] have shown that a large queue size guarantees better performance of a downstream task. Usually, the larger the queue size is, the better the model's performance is. Prior works thus adopt the largest possible queue size based on the training set size. However, our experiment results showed that the optimal queue size might not always be the largest possible number concerning the specific task. In our model, the best performance was reached under $N_{Neg} = 4,096$, rather than the larger 8,192 under both tasks. Such interesting differences might orient from our novel design of contrastive learning processes and algorithm. Another benefit of a smaller queue size is that our model has fewer computation costs.

Impact of survive-time-slot τ: We observe that our model achieves the best performance under $\tau = 200$ in RTE task, while it achieves the best performance under $\tau = 300$ in MRPC task. Such difference might be because the size of MRPC dataset is larger than RTE, so the small weight instances in MPRC would be considered proper negative
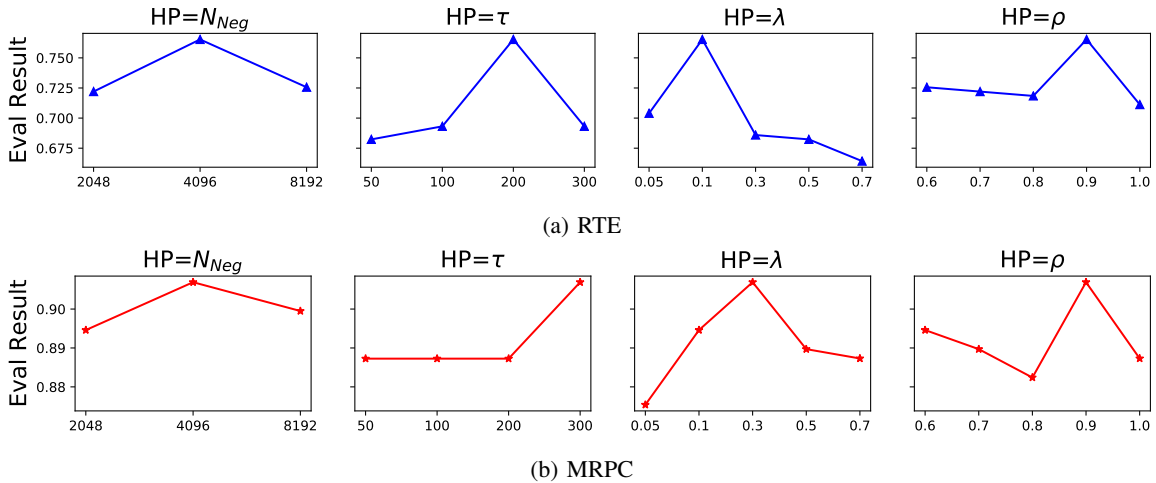
Fig. 6: Hyperparameter (HP) analysis of Contrastive Learning on RoBERTa-base encoder.

samples for a much longer time compared with augmented samples in RTE.

Impact of important score λ**:** We observe that properly combining the contrastive learning objective function and downstream task objective function yields the best performance, while the optimal values are more task-specific rather than globally general.

Impact of positive ratio ρ**:** We noticed that when $\rho = 0.9$, MRCO achieved the best performance on most tasks. However, when $\rho = 1.0$ (i.e., considered all the same origination augmented instance as positive samples), MRCO did not show the best performance. It indicates the importance of a higher standard for constructing positive samples (i.e., only keeping high-quality augmented samples as positive samples). It also reveals that the importance of using the hyper-parameter ρ.

TABLE IV: Ablation study of contrastive learning w/ RoBERTa-base encoder.

Methods on RoBERTa-base	RTE	MRPC
+ Aug. & Filter	$.744_{.002}$	$.883_{.004}$
MRCO	$.760_{.004}$	$.902_{.009}$
MRCO *w/o* Contrastive Learning	$.722_{.001}$	$.899_{.001}$
MRCO *w/o* LASW	$.747_{.025}$	$.889_{.006}$

2) Ablation Study of the Contrastive Learning Module: In the hyper-parameter analysis of λ, we learned that the median value of λ achieved the best performance. To understand the impact of contrastive learning in MRCO, we assigned $\lambda = 0$, indicating MRCO without contrastive learning. In Table IV, we can observe that contrastive learning played an important role in the performance of MRCO. In particular, there was a 3.8% accuracy drop without using contrastive learning compared with our original MRCO. In addition, by comparing the performance between "+ *Aug. & Filter*" and "MRCO *w/o* Contrastive Learning", we observe that in the RTE task, only utilizing meta reweighting module is not sufficient to beat the manually designed filtering method. In addition, to understand the effectiveness of LASW dequeue-enqueue algorithm, we

also considered replacing it with a FIFO algorithm (named as MRCO *w/o* LASW). Such replacement downgraded our framework's performance by 1.3% on both RTE and MRPC tasks. These observations revealed clear evidence of the non-trivial contributions of our contrastive learning design.

C. Training Dynamics

To understand how quickly our model got converged (i.e., reaching the best result), we visualized the development set accuracy with RoBERTa-base on MRPC and RTE in Fig. 8. Compared with the + *Aug*, which used the same amount of augmented data, our approach achieved slightly faster convergence (three epochs in MRCO vs. four epochs in + *Aug*). Different from the existing contrastive learning framework [29], MRCO did not require an additional training stage to learn a better feature representation at the beginning. MRCO also converged faster than the other two baselines.

VI. RELATED WORK

In this section, we summarize related work in 3 topics: (1) text augmentation; (2) meta learning and instance reweighting; and (3) contrastive learning.

A. Text Augmentation

Researchers proposed various text augmentation methods. There are dictionary-based word substitutions [25], [26], [32], where words are replaced with their synonyms. Other works search for similar words in certain context-free word embedding space [27], [33], [34]. Alternatively, works leveraging contextualized embeddings in masked language models can be applied for text augmentations [35], [36]. Generative text augmentations also gained research interests [3], [37]–[39]. A large number of noise injection operations (swapping/insertion/deletion/cutoff) on words/subwords/chars can also be applied for augmentations to improve a model's robustness or for creating adversarial examples [28], [40]–[45]. Recent works also revealed *back-translation* [7], [46], [47] and *mixup* [48], [49] which were helpful in certain NLP tasks.
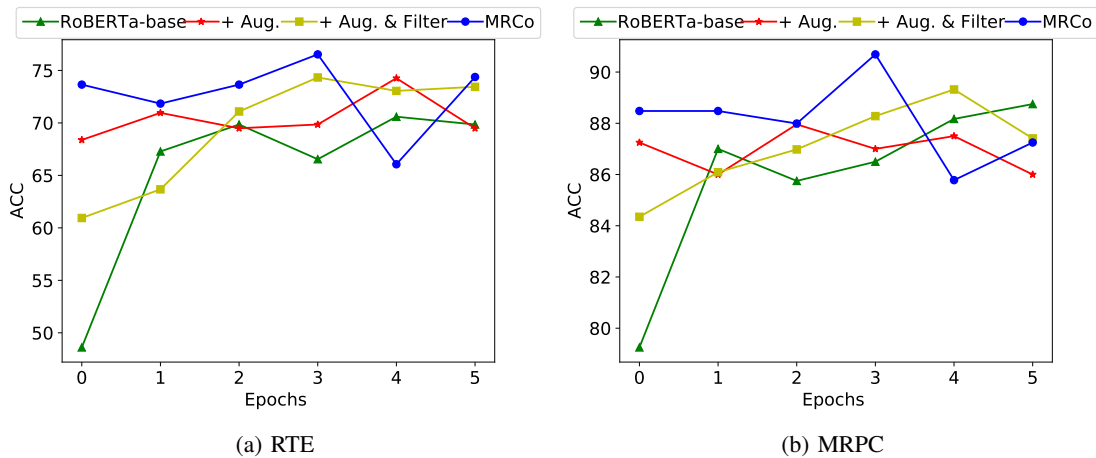
(a) RTE (b) MRPC

Fig. 8: Development set accuracy with basic encoder RoBERTa-base.

Combinations of these methods were also presented in several works [16], [50], [51]. Lastly, researchers developed practical tools of the aforementioned augmenting methods [52], [53]. In this paper, we utilized five representative augmentation methods as mentioned before, namely: *Wordnet* [25], *Easydata* [16], *Checklist* [26], *Embedding* [27] and *Charswap* [28].

B. Meta Learning and Instance reweighting

Meta Learning [54] is also known as "learning to learn" [55]. In this work, we mainly focus on leveraging meta learning techniques for reweighting augmented samples to reduce the noise within these augmented samples and preventing the model from collapse through bilevel optimization.

Recent work associated with data augmentations and meta learning [5], [8], [9], [17], [56]–[58] mainly focused on computer vision domain, in which the instance space of images is continuous. However, text is different from images in terms of the discrete nature of text strings. Alternatively, researchers leveraged reinforcement learning architectures where rewards' gradients could be calculated and back-propagated through the networks. Some networks also required the gradients to be back-propagated to the augmentation network. This approach is not applied to many heuristic text-based methods. Unlike the prior works, we proposed a framework that treats augmentation techniques as black boxes, and does not require the design of any reinforcement learning rewards.

Yi [4] proposed a reweight module to assign large weights to augmented samples with large loss values so that the model could pay more attention to harder-to-learn examples. Shu [5] also utilized meta learning to learn a reweight module based on the loss of augmented samples. However, Shu's reweight module assigns large weights to the samples with small loss as they view large loss as an indication of the potential noise in the samples. In this paper, we consider the quality of samples at different levels: the feature representations, and the associated label, which contain richer information than the singular-valued loss.

C. Contrastive Learning

The basic idea of contrastive learning is to encourage the feature representations of "similar" instances to be close while enforcing "different" instances to be apart. In unsupervised scenarios, *similar instances* can be images and their augmentation variations [7], [15]. While in supervised scenarios, such requirements are sometimes relaxed, where *similar instances* can be defined as ones with the same labels [21], [30].

We applied contrastive learning differently, as we defined similar augmented samples as those with similar weights generated by the meta reweighting module while large weighted augmented samples and small weighted augmented samples were naturally dissimilar. Those small weighted augmented samples were further ordered in a contrastive learning queue, with a novel proposed dequeue-enqueue algorithm rather than ordinary FIFO queue. Thus, our contrastive learning component was well adapted to the meta reweighting module.

VII. Conclusion

In this paper, we proposed a *Meta Reweighting Contrastive learning framework (*MRCo*)* to reduce and exploit the noise from data augmentation. Our model achieved performance improvements across various augmentation strategies and backbone encoder modules in the experiments. We further conducted extensive analysis of the model components to show their non-trivial contributions and give more insights in terms of the impact of varying values of different hyper-parameters and the actual training dynamics of our framework. In the future, we plan to extend our design to broader domains such as computer vision. We are sharing our implementations for better reproducibility.

Acknowledgment

REFERENCES

[1] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017.

[2] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," 2021.

[3] A. Anaby-Tavor, B. Carmeli, E. Goldbraich, A. Kantor, G. Kour, S. Shlomov, N. Tepper, and N. Zwerdling, "Do not have enough data? deep learning to the rescue!" in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7383–7390.

[4] M. Yi, L. Hou, L. Shang, X. Jiang, Q. Liu, and Z.-M. Ma, "Reweighting augmented samples by minimizing the maximal expected loss," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=9G5MIc-goqB

[5] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 1917–1928. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/e58cc5ca94270acaceed13bc82dfedf7-Abstract.html

[6] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[7] Y. Qu, D. Shen, Y. Shen, S. Sajeev, J. Han, and W. Chen, "Coda: Contrast-enhanced and diversity-promoting data augmentation for natural language understanding," *CoRR*, vol. abs/2010.08670, 2020. [Online]. Available: https://arxiv.org/abs/2010.08670

[8] Z. Tang, Y. Gao, L. Karlinsky, P. Sattigeri, R. Feris, and D. Metaxas, "OnlineAugment: Online data augmentation with less domain knowledge," in *Computer Vision – ECCV 2020*. Springer International Publishing, 2020, pp. 313–329. [Online]. Available: https://doi.org/10.1007/978-3-030-58571-6_19

[9] Z. Hu, B. Tan, R. Salakhutdinov, T. Mitchell, and E. P. Xing, "Learning data manipulation for augmentation and weighting," 2019.

[10] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data," 2019.

[11] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," 2019.

[12] F. Zhou, J. Li, C. Xie, F. Chen, L. Hong, R. Sun, and Z. Li, "Metaaugment: Sample-aware data augmentation policy learning," 2020.

[13] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta label correction for learning with weak supervision," 2019.

[14] K. Shu, G. Zheng, Y. Li, S. Mukherjee, A. H. Awadallah, S. Ruston, and H. Liu, "Leveraging multi-source weak social supervision for early detection of fake news," 2020.

[15] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[16] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," in *EMNLP-IJCNLP*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6382–6388. [Online]. Available: https://www.aclweb.org/anthology/D19-1670

[17] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta label correction for noisy label learning," in *AAAI 2021*, February 2021. [Online]. Available: https://www.microsoft.com/en-us/research/publication/meta-label-correction-for-noisy-label-learning/

[18] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," 2019.

[19] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," 2020.

[20] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," 2021.

[21] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 18 661–

18 673. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf

[22] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. [Online]. Available: https://www.aclweb.org/anthology/W18-5446

[23] Y. Kim, "Convolutional neural networks for sentence classification," 2014.

[24] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[25] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[26] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of nlp models with checklist," in *ACL*, 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.442

[27] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples," in *EMNLP*, 2018. [Online]. Available: https://doi.org/10.18653/v1/d18-1316

[28] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger: Generating adversarial text against real-world applications," in *NDSS*, 2019. [Online]. Available: https://doi.org/10.14722/ndss.2019.23138

[29] H. Fang, S. Wang, M. Zhou, J. Ding, and P. Xie, "Cert: Contrastive self-supervised learning for language understanding," *arXiv preprint arXiv:2005.12766*, 2020.

[30] B. Gunel, J. Du, A. Conneau, and V. Stoyanov, "Supervised contrastive learning for pre-trained language model fine-tuning," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=cu7IUiOhujH

[31] R. Flesch, "How to write plain english: Let's start with the formula," *University of Canterbury*, 1979.

[32] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, "Word-level textual adversarial attacking as combinatorial optimization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020, pp. 6066–6080. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.540

[33] W. Y. Wang and D. Yang, "That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 2557–2563.

[34] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling BERT for natural language understanding," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020, pp. 4163–4174. [Online]. Available: https://doi.org/10.18653/v1/2020.findings-emnlp.372

[35] S. Garg and G. Ramakrishnan, "BAE: bert-based adversarial examples for text classification," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020, pp. 6174–6181. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-main.498

[36] X. Wu, S. Lv, L. Zang, J. Han, and S. Hu, "Conditional bert contextual augmentation," in *International Conference on Computational Science*. Springer, 2019, pp. 84–95.

[37] T. Wullach, A. Adler, and E. M. Minkov, "Towards hate speech detection at large via deep generative modeling," *IEEE Internet Computing*, pp. 1–1, 2020. [Online]. Available: https://doi.org/10.1109/mic.2020.3033161

[38] R. Cao and R. K.-W. Lee, "Hategan: Adversarial generative-based data augmentation for hate speech detection," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 6327–6338. [Online]. Available: https://doi.org/10.18653/v1/2020.coling-main.557

[39] V. Kumar, A. Choudhary, and E. Cho, "Data augmentation using pre-trained transformer models," *arXiv preprint arXiv:2003.02245*, 2020.

[40] Z. Xie, S. I. Wang, J. Li, D. Lévy, A. Nie, D. Jurafsky, and A. Y. Ng, "Data noising as smoothing in neural network

language models," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.* OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=H1VyHY9gg

[41] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual,* H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html

[42] F. M. Luque, "Atalaya at TASS 2019: Data augmentation and robust embeddings for sentiment analysis," in *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019,* ser. CEUR Workshop Proceedings, M. Á. G. Cumbreras, J. Gonzalo, E. M. Cámara, R. Martínez-Unanue, P. Rosso, J. Carrillo-de-Albornoz, S. Montalvo, L. Chiruzzo, S. Collovini, Y. Gutiérrez, S. M. J. Zafra, M. Krallinger, M. Montes-y-Gómez, R. Ortega-Bueno, and A. Rosá, Eds., vol. 2421. CEUR-WS.org, 2019, pp. 561–570. [Online]. Available: http://ceur-ws.org/Vol-2421/TASS_paper_1.pdf

[43] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in *2018 IEEE Security and Privacy Workshops (SPW).* IEEE, 2018, pp. 50–56. [Online]. Available: https://doi.org/10.1109/spw.2018.00016

[44] D. Pruthi, B. Dhingra, and Z. C. Lipton, "Combating adversarial misspellings with robust word recognition," in *ACL,* 2019. [Online]. Available: https://doi.org/10.18653/v1/p19-1561

[45] D. Shen, M. Zheng, Y. Shen, Y. Qu, and W. Chen, "A simple but tough-to-beat data augmentation approach for natural language understanding and generation," *arXiv preprint arXiv:2009.13818,* 2020.

[46] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers.* The Association for Computer Linguistics, 2016. [Online]. Available: https://doi.org/10.18653/v1/p16-1009

[47] S. Edunov, M. Ott, M. Auli, and D. Grangier, "Understanding back-translation at scale," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018,* E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Association for Computational Linguistics, 2018, pp. 489–500. [Online]. Available: https://doi.org/10.18653/v1/d18-1045

[48] H. Guo, Y. Mao, and R. Zhang, "Augmenting data with mixup for sentence classification: An empirical study," *arXiv preprint arXiv:1905.08941,* 2019.

[49] J. Chen, Z. Yang, and D. Yang, "Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020,* D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020, pp. 2147–2157. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.194

[50] C. Coulombe, "Text data augmentation made simple by leveraging NLP cloud apis," *CoRR,* vol. abs/1812.04718, 2018. [Online]. Available: http://arxiv.org/abs/1812.04718

[51] G. Rizos, K. Hemker, and B. Schuller, "Augment to prevent: short-text data augmentation in deep learning for hate-speech classification," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management,* 2019, pp. 991–1000. [Online]. Available: https://doi.org/10.1145/3357384.3358040

[52] E. Ma, "Nlp augmentation," https://github.com/makcedward/nlpaug, 2019.

[53] J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp," in *EMNLP,* 2020. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-demos.16

[54] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial intelligence review,* vol. 18, no. 2, pp. 77–95, 2002.

[55] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science,* vol. 350, no. 6266, pp. 1332–1338, 2015.

[56] J. Rajendran, A. Irpan, and E. Jang, "Meta-learning requires meta-augmentation," in *Advances in Neural Information Processing Systems,* H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 5705–5715. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/3e5190eeb51ebe6c5bbc54ee8950c548-Paper.pdf

[57] H. Pham, X. Wang, Y. Yang, and G. Neubig, "Meta back-translation," *CoRR,* vol. abs/2102.07847, 2021. [Online]. Available: https://arxiv.org/abs/2102.07847

[58] R. Ni, M. Goldblum, A. Sharaf, K. Kong, and T. Goldstein, "Data augmentation for meta-learning," *CoRR,* vol. abs/2010.07092, 2020. [Online]. Available: https://arxiv.org/abs/2010.07092