A Multi-Level Optimization Framework for End-to-End Text Augmentation

Sai Ashish Somayajula
UC San Diego, USA
ssomayaj@ucsd.edu

Linfeng Song
Tencent AI Lab, USA
lfsong@tencent.com

Pengtao Xie*
UC San Diego, USA
p1xie@eng.ucsd.edu

Abstract

Text augmentation is an effective technique in alleviating overfitting in NLP tasks. In existing methods, text augmentation and downstream tasks are mostly performed separately. As a result, the augmented texts may not be optimal to train the downstream model. To address this problem, we propose a three-level optimization framework to perform text augmentation and the downstream task end-to-end. The augmentation model is trained in a way tailored to the downstream task. Our framework consists of three learning stages. A text summarization model is trained to perform data augmentation at the first stage. Each summarization example is associated with a weight to account for its domain difference with the text classification data. At the second stage, we use the model trained at the first stage to perform text augmentation and train a text classification model on the augmented texts. At the third stage, we evaluate the text classification model trained at the second stage and update weights of summarization examples by minimizing the validation loss. These three stages are performed end-to-end. We evaluate our method on several text classification datasets where the results demonstrate the effectiveness of our method. Code is available at https://github.com/Sai-Ashish/End-to-End-Text-Augmentation.

1 Introduction

Data augmentation (Sennrich et al., 2015; Fadaee et al., 2017; Wei and Zou, 2019) is an effective technique for mitigating the deficiency of training data and preventing overfitting. In natural language processing, many data augmentation methods have been proposed, such as back translation (Sennrich et al., 2015), synonym replacement (Wang and Yang, 2015), random insertion (Wei and Zou, 2019), and so on. In existing approaches, data augmentation and downstream tasks are performed separately: Augmented texts are created first, then they are used to train a downstream model. The downstream task does not influence the augmentation process. As a result, the augmented texts may not be optimal for training the downstream model.

In this paper, we aim to address this problem. We propose an end-to-end learning framework based on multi-level optimization (Feurer et al., 2015), which performs data augmentation and downstream tasks in a unified manner where not only augmented texts influence the training of the downstream model, but also the performance of downstream task affects how data augmentation is performed.

In our framework, we use a text summarization (Gambhir and Gupta, 2017) model to perform data augmentation. Given an original text t with class label c, we feed t into the summarization model to generate a summary s. We set the class label of s to be c. (s, c) is treated as an augmented text-label pair of (t, c). The motivation of using a summarization model for text augmentation is two-fold. First, the major semantics of an original text is preserved in its summary; therefore, it is sensible to assign the class label of the original text to its summary. Second, the summary excludes non-essential details in the original text; as a result, the semantic diversity between the summary and the original text is rich, which well serves the purpose of creating diverse augmentations. The summarization model is trained on a summarization dataset $\{(t_i, s_i)\}_{i=1}^{M}$ where t_i is an original text and s_i is the corresponding summary. For the downstream task, we assume it is text classification. We assume there is a text classification training set $\{(x_i^{(tr)}, y_i^{(tr)})\}_{i=1}^{N^{(tr)}}$ where $x_i^{(tr)}$ is an input text and $y_i^{(tr)}$ is the corresponding class label, and there is a text classification validation set $\{(x_i^{(val)}, y_i^{(val)})\}_{i=1}^{N^{(val)}}$.

*Corresponding author

Transactions of the Association for Computational Linguistics, vol. 10, pp. 343–358, 2022. https://doi.org/10.1162/tacl_a_00464
Action Editor: Dani Yogatama. Submission batch: 09/2021; Revision batch: 12/2021; Published 4/2022.

Our framework consists of three learning stages that are performed end-to-end. We train a text summarization model G on the summarization dataset at the first stage. Considering a domain difference between the summarization data and text classification data, we associate each summarization training pair with a weight $a \in [0, 1]$. A smaller a indicates a large domain difference between this summarization pair and the text classification data, and this pair should be down-weighted during the training of the summarization model. These weights are tentatively fixed at this stage and will be updated later. At the second stage, we use the trained summarization model to perform text augmentation for the classification dataset and train a text classification model on the augmented and original datasets. At the third stage, we validate the classification model trained at the second stage and update weights of summarization training examples by minimizing the validation loss. The three stages are performed end-to-end where they mutually influence each other. We evaluate our framework on several text classification datasets. Various experiments demonstrate the effectiveness of our method.

The major contributions of this work include:

- We propose a three-level optimization framework to perform text augmentation in an end-to-end manner. Our framework consists of three learning stages that mutually influence each other: 1) training text summarization model; 2) training text classification model; 3) updating weights of summarization data by minimizing the validation loss of the classification model.
- Experiments on various datasets demonstrate the effectiveness of our framework.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the method. Section 4 gives experimental results. Section 5 concludes the paper.

2 Related Work

2.1 Data Augmentation in NLP

As an effective way of mitigating the deficiency of training data, data augmentation has been broadly studied in NLP (Feng et al., 2021). Sennrich et al. (2015) proposed a back translation method for data augmentation, which improves the BLEU (Papineni et al., 2002a) scores in machine translation (MT). The back translation technique first converts the sentences to another language. It again translates it back to the original language to augment the original text.

Fadaee et al. (2017) propose a data augmentation method for low-frequency words. Specifically, the method generates new sentence pairs that contain rare words. Kafle et al. (2017) introduce two data augmentation methods for visual question answering. The first method uses semantic annotations to augment the questions. The second technique generates new questions from images using an LSTM network (Hochreiter and Schmidhuber, 1997). Wang and Yang (2015) propose an augmentation technique that replaces query words with their synonyms. Synonyms are retrieved based on cosine similarities calculated on word embeddings. Kolomiyets et al. (2011) propose to augment data by replacing the temporal expression words with their corresponding synonyms. They use the vocabulary from the Latent Words Language Model (LWLM) and the WordNet.

Şahin and Steedman (2019) propose two text augmentation techniques based on dependency trees. The first technique crops the sentences by discarding dependency links. The second technique rotates sentences by tree fragments that are pivoted at the root. Chen et al. (2020) propose augmenting texts by interpolating input texts in a hidden space. Wang et al. (2018) propose augmenting sentences by randomly replacing words in input and target sentences with words from the vocabulary. SeqMix (Guo et al., 2020) proposes to create augments by softly merging input/target sequences.

EDA (Wei and Zou, 2019) uses four operations to produce data augmentation: synonym replacement, random insertion, random swap, and random deletion. Kobayashi (2018) proposes to replace words stochastically with words predicted by a bi-directional language model. Andreas (2020) proposes a compositional data augmentation approach that constructs a synthetic training example by replacing text fragments in a real example with other fragments appearing in similar contexts. Kumar et al. (2021) apply pretrained Transformer models including GPT-2, BERT, and BART for conditional data augmentation, where the concatenation of class labels and input texts are fed into

these pretrained models to generate augmented texts. Kumar et al. (2021) propose a language model-based data augmentation method. This approach first finetunes a language model on limited training data, then feeds class labels into the finetuned model to generate augmented sentences. Min et al. (2020) explore several syntactically informative augmentation methods by applying syntactic transformations to original sentences and showed that subject/object inversion could increase robustness to inference heuristics.

2.2 Bi-level Optimization

Many NLP applications (Feurer et al., 2015; Baydin et al., 2017; Finn et al., 2017; Liu et al., 2018; Shu et al., 2019; Zheng et al., 2019) are based on bi-level optimization (BLO), such as neural architecture search (Liu et al., 2018), data selection (Shu et al., 2019; Ren et al., 2020; Wang et al., 2020), meta learning (Finn et al., 2017), hyperparameter tuning (Feurer et al., 2015), label correction (Zheng et al., 2019), training data generation (Such et al., 2019), learning rate adaptation (Baydin et al., 2017), and so forth. In these BLO-based applications, model parameters are learned by minimizing a training loss in an inner optimization problem while meta parameters are learned by minimizing a validation loss in an outer optimization problem. In these applications, meta parameters are neural architectures, weights of data examples, hyperparameters, and so on.

3 Method

This section proposes a three-level optimization framework to perform end-to-end text augmentation.

3.1 Overview

We assume the target task is text classification. We train a BERT-based (Devlin et al., 2018) text classifier on a training set $\mathcal{D}_c^{(tr)} = \{(x_i^{(tr)}, y_i^{(tr)})\}_{i=1}^{N^{(tr)}}$ where $x_i^{(tr)}$ is an input text and $y_i^{(tr)}$ is the corresponding class label. Meanwhile, we have access to a classification validation set $\mathcal{D}_c^{(val)} = \{(x_i^{(val)}, y_i^{(val)})\}_{i=1}^{N^{(val)}}$. In many application scenarios, the training data is limited, which incurs a high risk of overfitting. To address this problem, we perform data augmentation of the training data to enlarge the number of training examples. We use a text summarization model to perform data augmentation. Given an original training pair $(x_i^{(tr)}, y_i^{(tr)})$, we feed the input text $x_i^{(tr)}$ into the text summarization model and get a summary s_i. Because s_i preserves the major semantics of $x_i^{(tr)}$, we can assign the class label $y_i^{(tr)}$ of $x_i^{(tr)}$ to s_i. In the end, we obtain an augmented training pair $(s_i, y_i^{(tr)})$. This process can be applied to every original training example and create corresponding augmented training examples.

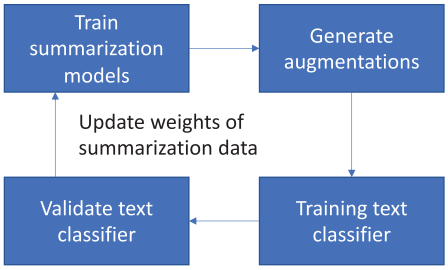

Figure 1: Overview of our framework.

To enable the text summarization model and the text classifier to influence and benefit from each other mutually, we develop a three-level optimization framework to train these two models end-to-end. Our framework consists of three learning stages performed in a unified manner. At the first stage, we train the text summarization model. At the second stage, we use the summarization model trained at the first stage to perform text augmentation and train the classifier on the augmented examples. At the third stage, we evaluate the classifier on a validation set and update weights of summarization training examples by minimizing the validation loss. Figure 1 shows an overview of our framework. Next, we describe the three stages in detail.

3.2 Stage I

At the first stage, we train the text summarization model. We use BART (Lewis et al., 2019) to perform summarization. BART is a pretrained Transformer (Vaswani et al., 2017) model consisting of an encoder and a decoder. The encoder takes a text as input, and the decoder generates a summary of the text. Let S denote the summarization model. The training data is $\mathcal{D}_s = \{(t_i, s_i)\}_{i=1}^{M}$ where t_i is an input text and s_i is the corresponding summary. Often, the summarization dataset $\mathcal{D}_s$ has a domain shift with the classification dataset $\mathcal{D}_c$. For example, in $\mathcal{D}_s$, if its domain difference

with the classification dataset is large, the summarization model trained by this example may not be suitable to perform data augmentation for $\mathcal{D}_c$. To address this problem, we associate each example in $\mathcal{D}_s$ with a weight $a \in [0, 1]$. If a is close to 0, it means that the domain difference between this example and $\mathcal{D}_s$ is large, and this example should be down-weighted during training the summarization model.

At this stage, we solve the following optimization problem:

$$S^*(A) = \min_S \sum_{i=1}^{M} a_i l(S, t_i, s_i) \tag{1}$$

where $A = \{a_i\}_{i=1}^{M}$ and $l(\cdot)$ is the teacher-forcing loss. The loss of (t_i, s_i) is weighted by the weight a_i of this example. If (t_i, s_i) has large domain difference with $\mathcal{D}_s$, a_i should be close to 0, then $a_i l(S, t_i, s_i)$ is made close to 0, which effectively excludes (t_i, s_i) from the training process. The optimally trained model S^* depends on A since S^* depends on the loss function, and the loss function depends on A. A is tentatively fixed at this stage and will be updated at a later stage. A cannot be updated at this stage. Otherwise, a trivial solution will be yielded where all values in A are 0.

3.3 Stage II

At the second stage, we use the summarization model $S^*(A)$ trained at the first stage to perform data augmentation of $\mathcal{D}_c^{(tr)} = \{(x_i^{(tr)}, y_i^{(tr)})\}_{i=1}^{N^{(tr)}}$. For each $x_i^{(tr)}$ in $\mathcal{D}_c^{(tr)}$, we feed it into $S^*(A)$ to generate a summary $g(x_i^{(tr)}, S^*(A))$. In the end, we obtain an augmented dataset $\mathcal{G}(\mathcal{D}_c^{(tr)}, S^*(A)) = \{(g(x_i^{(tr)}, S^*(A)), y_i^{(tr)})\}_{i=1}^{N^{(tr)}}$. We train a BERT-based text classifier C on the original data $\mathcal{D}_c^{(tr)}$ and augmented data $\mathcal{G}(\mathcal{D}_c^{(tr)}, S^*(A))$.

At this stage, we solve the following optimization problem:

$$\begin{aligned} &C^*(S^*(A)) = \\ &\min_C L(C, \mathcal{D}_c^{(tr)}) + \gamma L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S^*(A))) \end{aligned} \tag{2}$$

where $L(\cdot)$ denotes a cross-entropy classification loss and γ is a tradeoff parameter. The first loss term is defined on the original training dataset, and the second loss term is defined on the augmented training dataset. The optimally trained classifier C^* depends on $S^*(A)$ since C^* depends on the training loss, which depends on $S^*(A)$.

3.4 Stage III

At the third stage, we evaluate the classifier trained at the second stage on the classification validation set $\mathcal{D}_c^{(val)} = \{(x_i^{(val)}, y_i^{(val)})\}_{i=1}^{N^{(val)}}$ and update the weights A by minimizing the validation loss. At this stage, we solve the following optimization problem:

$$\min_A \quad L(C^*(S^*(A)), \mathcal{D}_c^{(val)}) \tag{3}$$

3.5 A Three-Level Optimization Framework

Putting all pieces together, we have the following three-level optimization framework.

$$\begin{aligned} \min_A \quad & L(C^*(S^*(A)), \mathcal{D}_c^{(val)}) \\ s.t. \quad & C^*(S^*(A)) = \min_C L(C, \mathcal{D}_c^{(tr)}) + \\ & \gamma L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S^*(A))) \\ & S^*(A) = \min_S \sum_{i=1}^{M} a_i l(S, t_i, s_i) \end{aligned} \tag{4}$$

There are three optimization problems in this framework, each corresponding to a learning stage. From bottom to top the optimization problems correspond to learning stages I, II, and III, respectively. The first two optimization problems are nested on the constraint of the third optimization problem. These three stages are conducted end-to-end in this unified framework. The solution $S^*(A)$ obtained in the first stage is used to perform text augmentation in the second stage. The classification model trained in the second stage is used to make predictions at the third stage. The importance weights A updated in the third stage change the training loss in the first stage and consequently changes the solution $S^*(A)$, which subsequently changes $C^*(S^*(A))$.

3.6 Optimization Algorithm

In this section, we develop a gradient-based optimization algorithm to solve the problem defined in Eq. (4). Drawing inspiration from Liu et al. (2018), we approximate $S^*(A)$ using one step gradient descent update of S:

$$S^*(A) \approx S' = S - \eta_s \nabla_S \sum_{i=1}^{M} a_i l(S, t_i, s_i) \tag{5}$$

We plug $S^*(A) \approx S'$ into the objective function at the second stage and get an approximate objective. We approximate $C^*(S^*(A))$ using one-step gradient descent update of C:

$$C^*(A) \approx C' = C - \eta_c \nabla_C (L(C, \mathcal{D}_c^{(tr)}) + \gamma L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S'))) \quad (6)$$

Finally, we plug $C^*(A) \approx C'$ into the validation loss and get an approximated objective. Then we update A by gradient descent:

$$A \leftarrow A - \eta_a \nabla_A L(C', \mathcal{D}_c^{(val)}) \quad (7)$$

where

$$\begin{aligned} &\nabla_A L(C', \mathcal{D}_c^{(val)})) = \tfrac{\partial S'}{\partial A} \tfrac{\partial C'}{\partial S'} \tfrac{\partial L(C', \mathcal{D}_c^{(val)})}{\partial C'} \\ &= \eta_s \eta_c \gamma \nabla^2_{A,S} \sum_{i=1}^{M} a_i l(S, t_i, s_i) \\ &\nabla^2_{S',C} L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S')) \nabla_{C'} L(C', \mathcal{D}_c^{(val)}) \end{aligned} \quad (8)$$

Eq. (8) involves an expensive matrix-vector product, whose computational complexity can be reduced by a finite difference approximation. Eq. (8) can be approximated as:

$$\begin{aligned} &\approx \tfrac{\eta_s \eta_c \gamma}{2\alpha} \{ [\nabla_{S'} L(C^+, \mathcal{G}(\mathcal{D}_c^{(tr)}, S')) - \\ &\nabla_{S'} L(C^-, \mathcal{G}(\mathcal{D}_c^{(tr)}, S'))] \nabla^2_{A,S} \sum_{i=1}^{M} a_i l(S, t_i, s_i) \} \end{aligned} \quad (9)$$

where

$$\alpha = \frac{0.01}{\left\| \nabla_{C'} L(C', \mathcal{D}_c^{(val)}) \right\|_2},$$

$$C^{\pm} = C \pm \alpha \nabla_{C'} L(C', \mathcal{D}_c^{(val)})$$

The matrix-vector multiplication in Eq. (9) can be further approximated by:

$$\frac{1}{\alpha_S^{\pm}} \{ \nabla_A \sum_{i=1}^{M} a_i l(S_{\pm}^{+}, t_i, s_i) - \nabla_A \sum_{i=1}^{M} a_i l(S_{\pm}^{-}, t_i, s_i) \} \quad (10)$$

where

$$\begin{aligned} \alpha_S^{\pm} &= \frac{0.01}{\left\| \nabla_{S'} L(C^{\pm}, \mathcal{G}(\mathcal{D}_c^{(tr)}, S')) \right\|_2}, \\ S_{\pm}^{+} &= S \pm \alpha_S^{+} \nabla_{T'} L(C^+, \mathcal{G}(\mathcal{D}_c^{(tr)}, S')) \\ S_{\pm}^{-} &= S \pm \alpha_S^{-} \nabla_{T'} L(C^-, \mathcal{G}(\mathcal{D}_c^{(tr)}, S')) \end{aligned}$$

Algorithm 1 Optimization algorithm

while not converged **do**
  Update weight parameters S using Eq. (5)
  Update weight parameters C using Eq. (6)
  Update meta parameters A using Eq. (7)
end while

Dataset	Train	Validation	Test
IMDB	6.7k	6.7k	25k
Yelp Review	10k	10k	38k
SST-2	10k	10k	872
Amazon Review	10k	10k	10k

Table 1: Split statistics of the classification datasets.

These update steps iterate until convergence. The overall algorithm is summarized in Algorithm 1.

4 Experiments

In this section, we report experimental results. The key takeaways include: 1) our proposed end-to-end augmentation method performs better than baselines that conduct augmentation and classification separately; 2) our framework is agnostic to the choices of classification models and can be applied to improve a variety of classifiers; 3) our framework is particularly effective when the number of training examples is small; 4) our framework is more effective when the length of input texts is large.

4.1 Dataset

For the text summarization data, we use the CNN-DailyMail dataset (See et al., 2017). It contains summaries of around 300k news articles. The full CNN-DailyMail training set is used for training the summarization model. We used four text classification datasets: 1) IMDB containing movie reviews (Maas et al., 2011), 2) Yelp Review: a binary sentiment classification dataset (Zhang et al., 2015), 3) SST-2: Stanford Sentiment Treebank (Socher et al., 2013), and 4) Amazon Review: a product review dataset from Amazon (McAuley and Leskovec, 2013). The split statistics of these datasets are summarized in Table 1.

4.2 Baseline

We compare our method with the following baseline methods.

- No-Aug: No data augmentation is applied.
- EDA (Wei and Zou, 2019): Augmented sentences are created by randomly applying the following operations: synonym replacement, random insertion, random swap, and random deletion.
- GECA (Andreas, 2020): A compositional data augmentation approach that constructs a synthetic training example by replacing text fragments in a real example with other fragments appearing in similar contexts.
- LAMBADA (Kumar et al., 2021): A language model based data augmentation method. This approach first finetunes a language model on real training data, then feeds class labels into the finetuned model to generate augmented sentences.
- Sum-Sep: Summarization-based augmentation and text classification are performed separately. We first train a summarization model, use it to perform data augmentation, then train the classification model on the augmented training data.
- Multi-task learning (MTL): In this baseline, the summarization model and classification model are trained by minimizing a single objective function, which is the weighted sum of the summarization and classification losses. The corresponding formulation is:

$$\begin{aligned}&\min_A \ L(C^*(A), \mathcal{D}_c^{(val)})\\ &s.t.\ C^*(A) = \min_{C,S} L(C, \mathcal{D}_c^{(tr)}) +\\ &\gamma L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S)) + \lambda \min_S \sum_{i=1}^{M} a_i l(S, t_i, s_i)\end{aligned}$$

4.3 Hyperparameter Settings

For the text classification model, we use the one in EDA (Wei and Zou, 2019). It contains an input layer, a bi-directional hidden layer with 64 LSTM (Hochreiter and Schmidhuber, 1997) units, a dropout layer with a probability of 0.5, another bi-directional layer with 32 LSTM units, another dropout layer with a probability of 0.5, ReLU activation, a dense layer with 20 hidden units, and a softmax output layer. We set the maximum text length to 150. The loss function is cross-entropy loss. Model parameters are optimized using the Adam (Kingma and Ba, 2014) optimizer, with an epsilon of 10^{-8}. In Adam, β_1 and β_2 are set to 0.9 and 0.999, respectively. The learning rate is a constant 10^{-3}. The batch size used is 8. For the importance weights A of summarization data examples, we optimize them using an Adam optimizer, with a weight decay of 10^{-3}. Epsilon, β_1, and β_2 are set to 10^{-8}, 0.5, and 0.999, respectively. The learning rate is 3×10^{-4}. The tradeoff parameter γ is set to 1 for all experiments, unless otherwise stated.

For the summarization model, we use the distilled BART model (Shleifer and Rush, 2020). It has six encoder layers and six decoder layers. We set the maximum text length for the article to 1024 and the summary to 75. We use an SGD optimizer with a momentum of 0.9 and a learning rate of 10^{-3}. We use a cosine annealing learning rate scheduler with the minimum learning rate set to 5×10^{-4}. We randomly sample a subset of CNN/DailyMail to train the summarization model. We balance the CNN/DailyMail data and the classification data by making the number of sampled CNN/DailyMail examples roughly the same as the classification data examples.

Accuracy is used as the evaluation metric. Each experiment runs five times with random initializations. We report the mean and standard deviation of the results. The experiments are conducted on 1080Ti GPUs.

4.4 Main Results

Following Wei and Zou (2019), for each classification dataset, we randomly sample x percentage of training data for model training. Tables 2, 3, 4, and 5 show the average accuracies on the Yelp Review, IMDB, Amazon Review, and SST-2 datasets, under different percentages.

From these tables, we make the following observations. First, our method works better than Sum-Sep. The reason is that our method performs the summarization-based augmentation and text classification in an end-to-end framework while Sum-Sep performs these two tasks separately. In our end-to-end framework, the summarization and classification models mutually influence each other. The performance of the classification model guides the training of the summarization model. If the classification accuracy is not high, it indicates that the augmented data generated by the summarization model is not useful. When this

Perc.	No-Aug	EDA	GECA	LAMBADA	Sum-Sep	MTL	Ours
5%	63.45±4.21	68.45±2.49	69.72±1.72	69.30±1.38	72.53±0.44	72.94±0.52	**74.58**±0.37
10%	68.15±3.77	74.46±1.82	74.91±2.07	75.03±1.45	76.73±0.57	76.35±0.51	**78.62**±0.73
20%	72.43±3.16	75.07±4.22	78.29±0.95	76.25±2.57	81.29±0.09	80.47±0.25	**81.66**±0.13
50%	81.78±1.76	81.57±2.44	83.61±0.93	82.18±1.82	84.90±0.19	84.29±0.22	**85.55**±0.15
75%	84.52±1.55	82.70±2.30	85.02±0.94	83.71±0.71	86.18±0.25	86.07±0.28	**87.10**±0.31
100%	85.42±1.52	84.66±1.36	86.45±0.26	85.08±0.49	87.46±0.10	87.15±0.13	**87.79**±0.07

Table 2: Classification accuracy (%) on Yelp Review. Perc. denotes percentage.

Perc.	No-Aug	EDA	GECA	LAMBADA	Sum-Sep	MTL	Ours
5%	52.86±3.22	61.42±1.75	61.58±0.94	61.03±1.46	63.74±0.27	63.27±0.41	**64.79**±0.32
10%	56.76±3.38	64.06±1.92	63.27±2.76	64.95±1.83	68.42±0.11	67.92±0.14	**68.75**±0.08
20%	59.54±1.68	67.18±3.20	65.36±0.83	67.61±1.94	69.82±0.61	68.47±0.69	**72.04**±0.52
50%	66.90±1.98	71.67±1.33	72.89±0.58	71.52±1.57	72.86±0.31	72.40±0.19	**73.98**±0.25
75%	72.25±1.05	74.23±0.72	73.69±0.55	74.02±0.73	74.78±0.25	74.63±0.41	**75.78**±0.37
100%	73.96±0.85	75.75±0.27	75.38±0.44	76.45±0.03	76.70±0.08	76.11±0.09	**77.00**±0.05

Table 3: Classification accuracy (%) on IMDB.

Perc.	No-Aug	EDA	GECA	LAMBADA	Sum-Sep	MTL	Ours
5%	63.46±1.51	62.59±2.35	64.82±0.93	63.21±1.07	65.44±0.31	64.81±0.69	**67.53**±0.42
10%	66.03±1.44	66.20±2.91	68.49±0.82	66.15±1.35	69.76±0.21	69.33±0.26	**70.84**±0.11
20%	68.04±2.26	68.36±1.05	69.04±1.94	70.96±2.45	72.64±0.58	72.51±0.99	**75.28**±0.73
50%	76.03±0.75	75.39±1.69	77.02±0.50	76.31±0.37	77.07±0.88	77.05±0.19	**78.80**±0.62
75%	77.40±1.09	76.30±1.82	78.52±0.52	77.02±0.81	78.78±0.52	78.93±0.37	**80.46**±0.62
100%	78.36±1.02	78.13±1.38	80.16±0.32	78.94±0.69	81.43±0.12	81.22±0.33	**82.01**±0.25

Table 4: Classification accuracy (%) on Amazon Review.

Perc.	No-Aug	EDA	GECA	LAMBADA	Sum-Sep	MTL	Ours
5%	57.22±2.07	63.42±1.24	58.44±0.57	59.31±0.99	59.05±0.91	60.51±0.32	**63.76**±0.85
10%	62.04±1.44	66.86±0.73	64.92±0.49	65.79±0.75	65.15±0.37	64.86±0.66	**68.23**±0.51
20%	64.45±1.99	68.00±0.49	67.48±0.55	66.12±0.87	67.09±1.27	67.84±0.49	**69.61**±0.21
50%	71.90±1.51	74.77±0.39	72.40±0.55	73.35±0.64	72.25±0.27	74.02±0.44	**75.23**±0.26
75%	73.74±0.59	75.80±0.81	75.03±0.33	74.29±0.28	75.34±0.07	74.25±0.10	**76.00**±0.14
100%	77.75±1.49	77.64±0.30	77.10±0.42	75.41±0.85	75.80±0.21	75.39±0.43	**77.92**±0.05

Table 5: Classification accuracy (%) on SST-2, γ is set to 0.5 for 5% and 20%.

occurs, the summarization model will adjust its network weights and data weights to generate useful augmentations in the next round of learning. In contrast, in Sum-Sep, such a feedback loop (from classification to summarization) does not exist. Therefore, its performance is inferior.

Second, our method works better than MTL. In MTL, the summarization model and classification model are trained simultaneously by minimizing a single objective function, which is the weighted sum of the summarization loss and classification loss. This incurs a competition between these two tasks: seeking for more decrease of the loss of one task leads to less decrease of the loss of the other task. Our method avoids task competition by performing these two tasks sequentially and minimizing two different objective functions. The summarization model is trained by minimizing the summarization loss. Then the classification model is trained by minimizing the classification loss. In this way, there is no competition between these two tasks. Though performed in two stages, these two tasks can still mutually influence each other in our end-to-end framework.

Third, our method performs better than No-Aug. This shows that the augmented data generated by our method has good utility for model training.

Fourth, the improvement of our method over baselines is more prominent when the number of training examples is smaller (i.e., when the

Perc.	Ours	EDA	No-Aug
5%	**48.36**±3.72	40.16±1.94	44.40±5.26
20%	**64.24**±2.14	51.48±2.44	58.00±4.10
100%	**72.36**±1.29	57.96±1.21	69.52±2.43

Table 6: Classification accuracy (%) on TREC.

percentage is small). Under the 5% percentage, we observe an improvement of 11.13%, 11.9%, 4.07%, and 6.59% on Yelp, IMDB, Amazon, and SST-2 datasets, respectively, over No-Aug. When the training dataset size is smaller, the necessity of data augmentation is more significant. As the percentage increases, the impact of data augmentation decreases. For the 100% percentage, we observe an improvement of 2.37%, 3.04%, and 3.65% on Yelp, IMDB, and Amazon, respectively, over No-Aug.

Fifth, our method works better than EDA, GECA, and LAMBADA. Again, the reason is that our method performs data augmentation and classification end-to-end while these baselines perform them separately. Compared to EDA, under the 5% percentage, we observe an accuracy gain of 6.13%, 3.37%, and 4.94% on Yelp, IMDB, and Amazon, respectively. EDA augments each input sentence to produce 8-16 new sentences. Our method achieves a better accuracy (on Yelp, IMDB, and Amazon) or a similar accuracy (on SST-2) than EDA with just one augmentation per input sentence.

EDA uses simple and heuristic rules to generate augmented sentences which may be noisy and lack semantic meaningfulness. These noisy augmentations may render the classification model trained on them to perform worse. To verify this, we performed experiments on the TREC (Li and Roth, 2002; Hovy et al., 2001) dataset (which is split into a train/validation/test set with 3000, 2000, and 500 examples respectively). Table 6 shows the results. As can be seen, with EDA as augmentation, the classification performance becomes much worse (compared with No-Aug). In contrast, our method trains a summarization model to generate semantically meaningful augmentations and perform much better than EDA.

Sixth, our method is more effective on long texts. For example, our framework outperforms EDA under all percentages on datasets where the input texts are relatively long, including Yelp, IMDB, and Amazon (the average number of words per input text in Yelp, IMDB, and Amazon are 133, 234, and 75, respectively). On SST-2, where the texts are relatively short, our method performs marginally better than EDA under certain percentages. In CNN-DailyMail, which is used for training the summarization model, the input texts are also long. As a result, the summarization model trained on CNN-DailyMail can generate good-quality summaries for Yelp, IMDB, and Amazon. The generated summaries preserve the major information in the input texts, discard redundant information, and align well with the original sentences semantically. Table 7 shows some examples. In this case, it is meaningful for the summary to inherit the class label of the original sentence, and the resulting (augmentation, label) pairs are useful for training the classification model. In contrast, in SST-2, the average number of words per input text in training data is only 10. When a short sentence is fed into the summarization model, the model tends to perform open text generation by treating the input short sentence as a prompt. The generated text may deviate from the original sentence in semantics. Table 8 shows some examples. In this case, assigning the class label of the original sentence to the summary introduces labeling noises. Therefore, the improvement of our method over baselines may not be substantial. We performed statistical significance tests using double-sided t-tests. In

Original sentence: This review is for the Kindle edition of what is supposedly the Christopher Gill translation of Plato's Symposium. However, it turns out if you download it that it is the same as the Benjamin Jowett translation which is available for free, whereas here you have to pay upwards of $8 for it. I also checked Penguin Classics web site and they do not indicate that a eBook version of this book is available. So be careful when purchasing the Kindle edition. I had to return my purchase for this Kindle book. **Augmentation**: Critically reviewed the Christopher Gill translation of Plato's Symposium. An online version of the book is currently available for free. However, if you download it it is the same as the Benjamin Jowett translation. In contrast, you pay upwards of $8.
Original sentence: My issue is not with anything Tom Robbins wrote, but with narrator Barret Whitener: a white male who opts to read all the dialogue from Japanese characters in a bad, generic ''Asian'' accent. It's unbearable to listen to. In addition, Whitener can't even pronounce the simplest of Japanese words, like ''sake.'' How many more years must transpire before an Asian person can have a voice without being mocked? **Augmentation**: The book is written by a white male who reads it in a bad, generic ''Asian'' accent. He can't even pronounce the simplest of Japanese words, like ''sake'' Movie is currently being played by Playmakers' weekly series .series

Table 7: Two augments generated by our method for Amazon Review.

Original sentence: An object lesson in period filmmaking. **Augmentation**: An object lesson in period filmmaking is a time-saving technique. The film was shot by the crew of the late 1960s and early 1980s. It's been produced since then, with some scenes being filmed on location . A few years later, it has become an important part of this type of cinema.
Original sentence: Mean-spirited lashing. **Augmentation**: Critically ill-tempered lashing. ''I feel like I'm going to get out of my own way,'' says one person. An angry person lashed out at him for being unprovoked and hateful. demonstrated anger over the use of alcohol.

Table 8: Two augmentation generated by our method for SST-2.

most cases, the p-values of our method against baselines are less than $1e-3$. This shows that our method is significantly better than baselines in general.

4.5 Ablation Studies

To evaluate the effectiveness of our proposed method, we compare with the following ablation settings.

- Aug-Only. We train the text classification model on augmented texts only, without using the original classification data, which amounts to solving the following problem:

$$\begin{aligned} &\min_A \ L(C^*(S^*(A)), \mathcal{D}_c^{(val)}) \\ &s.t.\ C^*(S^*(A)) = \min_C L(C, \mathcal{G}(\mathcal{D}_c^{(tr)}, S^*(A))) \\ &S^*(A) = \min_S \sum_{i=1}^{M} a_i l(S, t_i, s_i) \end{aligned} \tag{11}$$

- Ablation on classification models. In this study, we investigate whether our framework is effective for other classifiers, including CNN and RoBERTa (Liu et al., 2019). Following Wei and Zou (2019), the CNN model consists of input layer, 1D convolutional layer with 128 filters (kernel size is 5), global 1D max-pooling layer, ReLU activation function, dense layer (with 20 hidden units), softmax output layer. For RoBERTa, we use a batch size of 16 and a learning rate of 2×10^{-6} to train the model. The maximum classification text length is set to 128. γ is set to 1 for all datasets. The rest of the hyperparameters remain the same.
- Ablation on γ. In this study, we investigate how the test accuracy is affected by the hyperparameter γ.

Dataset	Ours	Aug-Only
IMDB	**68.75**±0.08	66.69±0.37
Yelp Review	**78.62**±0.73	77.94±0.91
Amazon Review	**70.84**±0.11	70.14±0.44
SST-2	**68.23**±0.51	66.17±0.42

Table 9: Comparison with Aug-Only, on 10% of the classification datasets.

Dataset	Ours	EDA	No-Aug
IMDB	**69.84**±0.15	69.33±0.08	65.57±0.52
Yelp	**78.95**±0.69	77.25±0.56	76.85±0.90
Amazon	**73.46**±0.22	72.86±0.48	70.62±0.79
SST-2	**67.32**±0.18	66.74±0.05	62.38±0.44

Table 10: Comparison of our method with EDA and No-Aug, with CNN as the text classifier, trained on 10% of the classification datasets.

Dataset	Ours	EDA	No-Aug
IMDB	**78.38**±0.19	77.26±3.31	77.33±0.62
Yelp	**88.25**±0.28	85.30±0.44	87.13±0.14
Amazon	**83.85**±0.34	81.36±0.23	82.54±0.24
SST-2	**76.34**±0.34	74.38±2.18	75.76±1.09

Table 11: Comparison of our method with EDA and No-Aug, with CNN as the text classifier, trained on 100% of the classification datasets.

Table 9 compares our method with Aug-Only. As can be seen, our full method outperforms Aug-Only. In Aug-Only, only augmented texts train the classification model without leveraging the original human-written texts. Because the augmented data is noisier than human-provided data, using augmented data only may lead to noisy classification models.

Tables 10 and 11 compare our method with EDA and No-Aug, with CNN as the text classifier, trained on 10% and 100% of the classification datasets, respectively. Tables 12 and 13 compare our method with EDA and No-Aug, with RoBERTa as the text classifier, trained on 10% and 100% of the classification datasets, respectively. As can be seen, with CNN and RoBERTa as classifiers, our method still outperforms EDA and No-Aug. This shows that our method is agnostic to text classifiers and can be leveraged to improve different text classifiers.

Figure 2 shows how test accuracy changes with γ, where the model is trained on 10% of IMDB. As can be seen, when γ increases from 0 to 0.5,

Dataset	Ours	EDA	No-Aug
IMDB	**84.40**±0.37	84.35±0.35	83.89±0.64
Yelp	**91.12**±0.29	90.38±0.39	90.23±0.39
Amazon	**90.56**±0.37	89.90±0.41	89.60±0.36
SST-2	**87.73**±0.86	87.10±0.83	87.57±0.74

Table 12: Comparison of our method with EDA and No-Aug, with RoBERTa as the text classifier, trained on 10% of the classification datasets.

Dataset	Ours	EDA	No-Aug
IMDB	**89.06**±0.07	88.59±0.10	88.50±0.14
Yelp	**93.60**±0.11	93.12±0.05	92.97±0.37
Amazon	**92.71**±0.04	92.33±0.06	92.20±0.14
SST-2	**91.28**±0.06	91.09±0.03	91.05±0.07

Table 13: Comparison of our method with EDA and No-Aug, with RoBERTa as the text classifier, trained on 100% of the classification datasets.

the classification accuracy increases. This is because a larger γ renders the classification model to be trained more by the augmented data. The augmented data provides additional training resources, which can mitigate the lack of original data. However, as γ continues to increase, the accuracy decreases. This is because an excessively large γ renders too much emphasis on augmented data and less attention paid to original data. Original data is less noisy than augmented data and therefore is more valuable than augmented data for training high-quality models. Similar results can be observed in Figure 3, where the model is trained on 10% of Amazon, SST-2, and Yelp.

We also perform an ablation study which replaces the summarization data with paraphrase data. The model S remains the same, which is still distill-BART. The paraphrase data contains 3,900 sentence pairs from Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005). These pairs are labeled as paraphrases by human. To ensure a fair comparison, we randomly sample 3,900 data examples from CNN-DailyMail, denoted by CNNDM-3900. Table 16 shows the results. As can be seen, CNNDM-3900 yields better performance than MRPC. This shows that using the summarization model for sentence augmentation is more effective than using the paraphrase model. The possible reason is that a

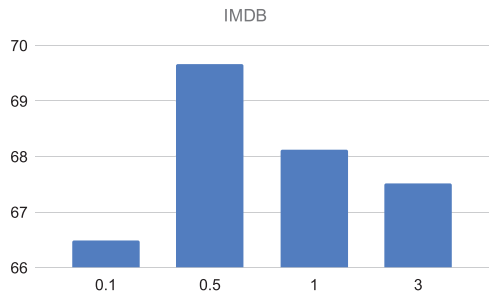

Figure 2: How test accuracy changes with γ, where the model is trained on 10% of IMDB.

summarization model discards less important information of the input text while a paraphrasing model preserves most information of the input text; as a result, augmentations generated by the summarization model are less noisy and have a larger semantic diversity to the input texts.

4.6 Analysis of Learned Weights of Summarization Data

Table 14 shows some randomly sampled summarization data examples whose weights learned by our framework are close to zero when the classification dataset is SST-2. Due to the space limit, we only show the summaries. As can be seen, these data are primarily about healthcare, energy, law, and politics, which have a large domain discrepancy with SST-2, which is about movie reviews. This shows that our framework is able to successfully identify out-of-domain summarization data and exclude them from training the summarization model.

Table 15 shows some randomly sampled summarization data examples whose weights learned by our framework are close to one when the classification dataset is SST-2. Due to the space limit, we only show the summaries. As can be seen, these summarization data are primarily about movies, recreation, and media, which are close to SST-2 in terms of domain similarity.

When the classification dataset is IMDB, the weights of examples in Table 14 are also close to zero, and the weights of examples in Table 15 are also close to one. This is because IMDB and examples in Table 15 are about movies while examples in Table 14 are not.

Given the learned weights, we split the summarization data into two subsets: ONE, which contains examples whose weights are > 0.5, and

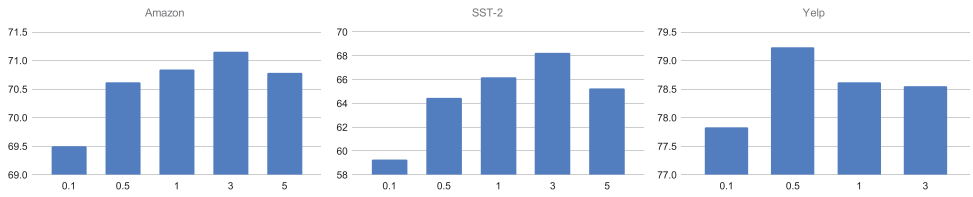

Figure 3: How test accuracy changes with γ, where the model is trained on 10% of Amazon, SST-2, and Yelp.

Half of states will expand Medicaid under Obamacare; half refuse or are on the fence. Low-income citizens and their advocates say Medicaid expansion necessary. States like Texas, Florida say a Medicaid expansion is costly and will fail. Politics at play and most states will eventually expand the program, political experts say.
The country plans to remove all subsidies in five years. The price of gasoline goes up four-folds. Iran has trillions of dollars in natural resources but still struggles, experts say. The changes aim to dampen domestic demand for fuel and oil, and bolster overall revenues.
Attorneys for three kidnapped women have ‘‘confidence and faith’’ in prosecution. Indictment lists use of chains, tape, vacuum cord against women. Castro also faces 139 counts of rape, 177 counts of kidnapping, one aggravated murder charge. A prosecutor’s committee will later consider whether seeking death penalty is appropriate.
North Korea’s announcement of a planned satellite launch has provoked alarm. Other countries say it is a way of testing missile technology. The Japanese defense minister orders the preparation of missile defenses.

Table 14: Some summarization data examples whose weights learned by our framework are close to 0. The classification dataset is SST-2.

Hollywood often misrepresents careers and the workplace in movies. Work is portrayed as easy or non-existent, and any outfit is appropriate. This can have a negative effect on younger generations deciding on careers.
Hollywood bringing back box-office juggernauts Iron Man, Batman and Spider-Man. 2012 may be a peak year for superheroes on film, with much more to come. Writer-director: DC Comics characters often face more challenges than Marvel. ’‘‘The genre has been popular for decades and is here to stay,’’ Boxofficeguru.com editor says.
Sam Smith wins best new artist, best pop vocal album. Beyonce now has 20 Grammys, passing Aretha Franklin.
Ted Turner turns 75 years old this month. He founded CNN, the first 24-hour cable news network, in 1980. In 1990, Turner hired Wolf Blitzer, host of CNN’s ‘‘The Situation Room’’. Blitzer reflects on what he learned from his former boss.

Table 15: Some summarization data examples whose weights learned by our framework are close to 1. The classification dataset is SST-2.

ZERO, which contains examples whose weights are ≤ 0.5. For each subset, we use it to train a summarization model (no reweighting of data during training), use the summarization model to generate augmentations, and train the classification model on augmented data (together with real data). Table 17 shows the results. As can be seen, augmentations generated by the summarization model trained on ONE improve classification performance in most cases. In contrast, augmentations generated by the summarization model trained on ZERO are not helpful.

Data	No-Aug	Summarize	Paraphrase
Yelp	68.15±3.77	**75.06**±0.62	74.33±0.47
IMDB	56.76±3.38	**64.94**±0.15	64.13±0.07
Amazon	66.03±1.44	**67.36**±0.08	66.39±0.14
SST-2	62.04±1.44	**67.44**±0.27	66.72±0.11

Table 16: Classification accuracy (%) on 10% of different datasets, when using summarization data and paraphrase data to train the augmentation model.

Data	No-Aug	ONE	ZERO
Yelp	68.15±3.77	**77.27**±0.96	69.92±2.09
IMDB	56.76±3.38	**66.94**±0.23	57.85±0.41
Amazon	66.03±1.44	**69.24**±0.15	67.74±0.22
SST-2	62.04±1.44	**67.50**±0.34	63.69±0.46

Table 17: Classification accuracy (%) on 10% of different datasets, in the study of how summarization data weights affect downstream classification performance.

Using In-domain Data to Train a Summarization Model. We conduct an experiment that uses in-domain data to train the summarization model. Specifically, we replace the CNN-Dailymail dataset with a movie summarization (MovSum) dataset crawled from IMDB. The MovSum dataset contains 135K (movie synopsis, movie summary) pairs where the summary (shorter) is treated as a summarization of the synopsis (longer). MovSum is in the same domain as SST-2 since they are both about movies. We randomly sample 135K examples from CNN-Dailymail (denoted by CNNDM-135K) and train the summarization

Summarization Data	Classification Accuracy	Percentage of ones
MovSum	**67.72**±0.36	84.5
CNNDM-135K	67.15±0.21	39.3

Table 18: Results of our proposed method on SST-2 (10%) under different summarization datasets.

model on this subset for a fair comparison. Classification is conducted on SST-2 (using 10% training examples).

Table 18 shows the results. As can be seen, using MovSum to train the summarization model leads to better classification performance on SST-2. This is because, compared with CNNDM-135K, MovSum has a higher domain similarity with SST-2. A summarization model trained using in-domain data can generate in-domain augmentations, which are more suitable to train the classification model. In addition, we measure the percentage of ones in the learned weights of summarization data examples. Under MovSum, the percentage is larger. This is because more data examples in MovSum are in the same domain as SST-2, compared with CNNDM-135K.

4.7 Analysis of Generated Augmentations

We measure the diversity of generated augmentation. Two types of diversity measures are used: I) how different the generated augmentation is from the input text; and II) how different sentences in a generated augmentation are. We measure type-I diversity in the following way: Given an input text t and an augmentation s generated from t, calculate the BLEU (Papineni et al., 2002b) score between s and t. We measure type-II diversity in the following way: for each pair of sentences in a generated augmentation, calculate their BLEU score, then take the average over all pairs. In the BLEU score, the number of grams is set to 4. Table 19 shows the results. As can be seen, the BLEU scores are low for both types of diversity, which indicates that the augmentations generated by our method are diverse.

We also check whether generated augmentations are abstractive. We randomly sample 300 generated augmentations and ask 5 undergraduates to manually annotate whether they are abstractive (with score 1) or extractive (score 0). Table 20 shows the results. The average score is close to 1, indicating that the augmentations generated by our method are abstractive.

Dataset	Type-I	Type-II
Yelp	0.15	0.07
IMDB	0.22	0.06
Amazon	0.11	0.04
SST-2	0.13	0.06

Table 19: Diversity of generated augmentations.

Dataset	Score
Yelp	0.83
IMDB	0.79
Amazon	0.88
SST-2	0.92

Table 20: Abstractiveness of generated augmentations.

Model	Rouge-2 F1	Rouge-L F1
GPT-2 (Radford et al., 2019)	8.27	26.58
Sum-Sep	16.06	34.07
MTL-SST2	11.06	28.44
Ours-SST2	12.77	30.27
MTL-IMDB	10.79	29.20
Ours-IMDB	13.06	30.72
MTL-Yelp	12.15	29.43
Ours-Yelp	12.97	30.62
MTL-Amazon	11.73	29.16
Ours-Amazon	13.13	30.71

Table 21: Evaluation of summarization models.

4.8 Performance of Summarization Models

We report the performance of the summarization models in Table 21. For our method and MTL, the reported scores are averages for different percentages of classification training data, where the classification model is LSTM. From this table, we make two observations. First, our method outperforms GPT2 (Radford et al., 2019), a competitive model used for text summarization. This shows that our method can generate meaningful summaries. Second, our method performs worse than Sum-Sep. The reason is that in our method, the summarization model is trained for the sake of generating good augmentations for the classification model while Sum-Sep is trained solely to maximize the summarization performance. Our

goal is not to generate best summaries, but rather generating best augmentations for classification using the summarization model.

5 Conclusions and Discussion

In this paper, we propose a three-level optimization framework to perform text augmentation and classification end-to-end. Our framework consists of three learning stages performed end-to-end: 1) training a text summarization model; 2) training a text classification model; and 3) updating weights of summarization examples by minimizing the validation loss of the classification model. Each learning stage corresponds to one level of the optimization problem in the framework. The three levels of optimization problems are nested and solved in a unified way. Our framework enables the augmentation process to be influenced by the performance of the text classification task so that the augmented texts are specifically suitable for training the classification model. Experiments on various datasets demonstrate the effectiveness of our method.

Our framework can be extended to other downstream tasks beyond classification, including but are not limited to text-to-image generation, visual question answering, dialog generation, etc. To extend our framework to a downstream task T, we need to change the loss functions in Eq. (2) and Eq. (3) to the loss of task T. For example, to apply our framework for text-to-image generation, given each text-image (t, i) pair, we perform augmentation of the input text t using the summarization model trained in Eq. (1), to get an augmented text $\hat{t}$. $(\hat{t}, i)$ would be treated as an augmented data pair. Then we define GAN-based losses (Goodfellow et al., 2014) on each $(\hat{t}, i)$ and each (t, i) to train a text-to-image generation model. We plan to study such extensions in future work.

Our current framework has the following limitations. First, it incurs additional computation and memory costs due to the usage of a summarization model. Second, currently, our method uses a text summarization model to generate augmentations. Publicly available summarization datasets are limited in size, limiting the augmentation model's quality. We plan to address these limitations in future works. To reduce memory and computation costs, we will perform parameter sharing where the encoder weights of the summarization model and those of the classification model are tied together. We plan to leverage data-rich text generation tasks for augmentation to address the second limitation, such as using machine translation models to perform back translation.

Acknowledgments

This work is partially supported by a gift fund from CCF and Tencent. We also thank the editor and the anonymous reviewers for their valuable suggestions.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. https://doi.org/10.18653/v1/2020.acl-main.676

Atilim Gunes Baydin, Robert Cornish, David Martínez-Rubio, Mark Schmidt, and Frank D. Wood. 2017. Online learning rate adaptation with hypergradient descent. *CoRR*, abs/1703.04782.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*. https://doi.org/10.18653/v1/2020.acl-main.194

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440*. https://doi.org/10.18653/v1/P17-2090

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data

augmentation approaches for NLP. *arXiv preprint arXiv:2105.03075*. https://doi.org/10.18653/v1/2021.findings-acl.84

Matthias Feurer, Jost Springenberg, and Frank Hutter. 2015. Initializing bayesian hyperparameter optimization via meta-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR.org.

Mahak Gambhir and Vishal Gupta. 2017. Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review*, 47(1):1–66. https://doi.org/10.1007/s10462-016-9475-9

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27.

Demi Guo, Yoon Kim, and Alexander M. Rush. 2020. Sequence-level mixed sample data augmentation. *arXiv preprint arXiv:2011.09039*. https://doi.org/10.18653/v1/2020.emnlp-main.447

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*. https://doi.org/10.3115/1072133.1072221

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. 2017. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 198–202. https://doi.org/10.18653/v1/W17-3529

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*. https://doi.org/10.18653/v1/N18-2072

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 271–276. ACL; East Stroudsburg, PA.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2021. Data augmentation using pre-trained transformer models.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*. https://doi.org/10.18653/v1/2020.acl-main.703

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*. https://doi.org/10.3115/1072228.1072378

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

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

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland,

Oregon, USA. Association for Computational Linguistics.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 165–172. https://doi.org/10.1145/2507157.2507163

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002a. BLEU: A method for automatic evaluation of machine translation. In *ACL*. https://doi.org/10.3115/1073083.1073135

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002b. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics. https://doi.org/10.3115/1073083.1073135

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

Zhongzheng Ren, Raymond Yeh, and Alexander Schwing. 2020. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21786–21797. Curran Associates, Inc.

Gözde Gül Şahin and Mark Steedman. 2019. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*. https://doi.org/10.18653/v1/D18-1545

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Association for Computational Linguistics, Vancouver, Canada.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*. https://doi.org/10.18653/v1/P16-1009

Sam Shleifer and Alexander M. Rush. 2020. Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, pages 1919–1930.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. 2019. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. *CoRR*, abs/1912.07768.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

William Yang Wang and Diyi Yang. 2015. 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*, pages 2557–2563. https://doi.org/10.18653/v1/D15-1306

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: An efficient data augmentation algorithm for neural machine translation. *arXiv preprint arXiv:1808.07512*.

https://doi.org/10.18653/v1/D18-1100

Yulin Wang, Jiayi Guo, Shiji Song, and Gao Huang. 2020. Meta-semi: A meta-learning approach for semi-supervised learning. *CoRR*, abs/2007.02394.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*. https://doi.org/10.18653/v1/D19-1670

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *arXiv:1509.01626 [cs]*.

Guoqing Zheng, Ahmed Hassan Awadallah, and Susan T. Dumais. 2019. Meta label correction for learning with weak supervision. *CoRR*, abs/1911.03809.