Constructing Contrastive Samples via Summarization for Text Classification with Limited Annotations

Yangkai Du[1], Tengfei Ma[2], Lingfei Wu[3], Fangli Xu[4], Xuhong Zhang[1]
Bo Long[3], Shouling Ji[1]
[1]Zhejiang University; [2]IBM Research; [3]JD.COM; [4]Squirrel AI Learning
`{yangkaidu,zhangxuhong,sji}@zju.edu.cn`
`tengfei.ma1@ibm.com, {lingfei.wu,bo.long}@jd.com, lili@yixue.us`

Abstract

Contrastive Learning has emerged as a powerful representation learning method and facilitates various downstream tasks especially when supervised data is limited. How to construct efficient contrastive samples through data augmentation is key to its success. Unlike vision tasks, the data augmentation method for contrastive learning has not been investigated sufficiently in language tasks. In this paper, we propose a novel approach to construct contrastive samples for language tasks using text summarization. We use these samples for supervised contrastive learning to gain better text representations which greatly benefit text classification tasks with limited annotations. To further improve the method, we mix up samples from different classes and add an extra regularization, named Mixsum, in addition to the cross-entropy-loss. Experiments on real-world text classification datasets (Amazon-5, Yelp-5, AG News, and IMDb) demonstrate the effectiveness of the proposed contrastive learning framework with summarization-based data augmentation and Mixsum regularization.

1 Introduction

Learning a good representation has been an essential problem in the deep learning era. Especially, in the area of natural language processing, the language model pre-training techniques, such as BERT (Devlin et al., 2019), have been overwhelming in a wide range of tasks by learning contextualized representations. However, the success of these pre-trained models hinge largely on plenty of labeled data for fine-tuning. With limited labels on the target task, fine-tuning BERT has been shown unstable(Zhang et al., 2021). In practice, it is costly to gather labeled data for a new task, and lack of training data is still a big challenge in many real-world problems.

Recently, contrastive learning methods have become popular self-supervised learning tools and gained big progress in few-shot learning due to its better discriminative ability (Gidaris et al., 2019; Su et al., 2020). Various contrastive learning methods have been developed and lead to state-of-the-art performance in many computer vision tasks. They are also extended to the fully supervised setting by leveraging label information to make further improvement. In natural language processing, contrastive learning has not been fully investigated but it is attracting more and more attentions.

A contrastive learning method generally consists of two components: finding positive samples and negative samples for each anchor sample; and building up an effective objective function to discriminate them. In many contrastive learning frameworks, how to efficiently find the contrastive samples has been the key to their success. For example, in MoCo(He et al., 2020), the contrastive pairs are constructed by matching an encoded query with a dynamic dictionary; in SimCLR(Chen et al., 2020), the contrastive pairs are created by applying two different data augmentation operators, and it was shown that composition of data augmentation operations is crucial for learning good representations. In supervised contrastive learning, essentially the positive sample space has been augmented. Instead of only using the anchor sample and its own transformation, all samples in the same class can be further regarded as positive pairs.

In this paper, we focus on using contrastive learning to assist the text classification tasks with limited labels. Considering the specialty of the text classification task, we propose two novel strategies to further enhance the performance of supervised contrastive learning. We assume that a good summarization system can keep the most critical information of original texts and the generated summary tends to belong to the same category as the original text. Thus we utilize text summarization as a data augmentation method to create more positive and negative samples for supervised contrastive

learning. Furthermore, we propose Mixsum, an idea similar to the methodology of mix-up(Zhang et al., 2018), which combines texts from different categories and creates new summary samples to further augment the data for contrastive learning. We adapt the supervised contrastive loss to the Mixsum setting, and show that it brings great benefit for text classification when training data is extremely scarce.

Our main contributions are listed as below:

- We propose a new contrastive learning framework for text representation learning and mitigate the label deficiency problem for text classification.

- We employ text summarization, a new data augmentation method, to construct positive and negative sample pairs for contrastive learning.

- We improve the supervised contrastive learning method by mixing up the samples in different categories. Combining with the summarization based data augmentation method, our model shows superior performance on three real-world datasets.

2 Background and Related Works

2.1 Contrastive Learning

The main idea of contrastive learning is minimizing the vector distance between anchor examples and positive examples while maximizing the vector distance between anchor examples and negative examples.

Self-supervised contrastive Learning has been demonstrated effective on many computer vision tasks (He et al., 2020; Chen et al., 2020). In a self-supervised contrastive learning framework, anchor samples are the original data samples, positive samples are the augmented anchor sample, and negative samples are generally set to all other samples in the mini-batch.

$$L_{self} = \sum_{i=1}^{N} -log \frac{exp(f(x_i) \cdot f(x_{2i})/\tau)}{\sum_{k=1}^{2N} 1_{i \neq k} exp(f(x_i) \cdot f(x_k)/\tau)} \quad (1)$$

Equation 1 is the self-supervised contrastive learning objective for the popular SimCLR framework (Chen et al., 2020). For each mini-batch with N anchor samples, we can get another N positive samples by data augmentation, concatenate them to

form a new batch. Then for each anchor examples index, i in the range $\{1, 2, ..., N\}$, the index for the corresponding positive sample is $2i$, and all other $2N - 2$ samples in the batch are negative samples. $f(\cdot)$ is a representation model mapping the input samples to a normalized dense vector in R^d, and τ is the temperature parameter. Contrastive learning on NLP tasks also arises much research intensity recently. Fang et al. (2020) propose to learn sentence-level representations by fine-tuning BERT(Devlin et al., 2019) with back-translation based data augmentation and self-supervised contrastive learning objective function. Klein and Nabi (2020) propose to use contrastive learning for commonsense reasoning, and the proposed method alleviates the current limitation of supervised commonsense reasoning. Khosla et al. (2020) explore the general supervised contrastive learning loss and show the effectiveness of supervised contrastive learning. Gunel et al. (2020) introduced the supervised contrastive loss to the original cross-entropy loss for fine-tuning pre-trained transformers like Roberta(Liu et al., 2019) and BERT(Devlin et al., 2019), which is highly related to our work. Our approach is different from these previous works in that we utilize a new data augmentation, i.e. summarization, for supervised contrastive learning. Our Mixsum method is also never explored by those methods.

2.2 Beyond Empirical Risk Minimization

The general theme of supervised learning is minimizing the empirical risk of datasets by defining a loss function l, which describes the difference between the model prediction $f(x)$ and target label y. The expected risk of the datasets can be described in Equation 2.

$$R(f) = \int l(f(x), y) dP(x, y) \quad (2)$$

P(x,y) is the distribution of the dataset, which is unknown but can be approximated by empirical distribution. Then we can now approximate the expected risk by empirical risk in Equation 3.

$$R_e(f) = \frac{1}{n} \sum_{i=1}^{N} l(f(x_i), y_i) \quad (3)$$

Minimizing the empirical risk in Equation 3 is called Empirical Risk Minimization(ERM) (Vapnik, 1999). ERM will lead the model to memorize

the training samples and fail for data out of training samples. Motivated by the limitation of ERM, Zhang et al. (2018) propose a generic vicinal distribution, called mixup:

$$\begin{aligned}\tilde{x} &= \lambda x_i + (1-\lambda)x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda)y_j\end{aligned} \quad (4)$$

Zhang et al. (2018) use this new vicinal distribution described in Equation 4 to approximate the expected risk, and minimizing the empirical vicinal risk(Chapelle et al., 2001) in Equation 5.

$$R_v(f) = \frac{1}{n}\sum_{i=1}^{N} l(f(\tilde{x}_i), \tilde{y}_i) \quad (5)$$

The proposed vicinal distribution–mixup, can be viewed as a form of data augmentation that leads the model to behave in between the training samples and soften the labels. Experiments demonstrate that mixup can improve the robustness of the trained model and avoid undesirable oscillations when predicting unseen samples(Zhang et al., 2018).

Besides, Kim et.al (Kim et al., 2020) proposed MixCo, which create a vicinal distribution for self-supervised contrastive learning based on the idea of mixup(Zhang et al., 2018), they demonstrate the effectiveness of vicinal distribution minimization for self-supervised contrastive learning loss over image classification tasks. Inspired by mixup and MixCo, we propose a novel vicinal distribution, i.e. Mixsum, for supervised contrastive learning.

3 Methods

3.1 Problem Definition

The task we want to solve is text classification with limited annotations. In the text classification task, the input data is usually a sentence, a paragraph or a document. Assume we have a small number of training samples with labels D_{train} and a large amount of unlabeled data D_{test}. For each text sample $x \in D_{train}$, it has a label y which is from L classes. And we want to predict the labels of all samples in the test data.

3.2 Text Summarization

We propose to use text summarization as the data augmentation strategy for constructing positive and negative samples in supervised contrastive learning when the number of annotated training samples is limited. Intuitively, the summarization process can

filter out unnecessary and redundant information in the text and extract the most representative semantics. The summary owns the same label as its source text.

We use PreSumm (Liu and Lapata, 2019) for automatic text summarization. PreSumm utilizes BERT as a general framework for both extractive and abstractive summarization, both of them can achieve great summarization quality even without text-summary pairs for finetuning. For each input text x we can get its summary x' by feeding the input text x_i to PreSumm model 6,where i is the index in Minibatch.

$$x'_i = PreSumm(x_i) \quad (6)$$

We use the abstractive summarization model trained by (Liu and Lapata, 2019) without any text-summary pairs for fine-tuning. Compared to extractive summarization, which can only generate summaries by extracting key sentences from original paragraphs, abstractive summarization can generate information-rich, coherent and less-redundant summary compared to extractive summary and do not have the limitation that summary is only from the original text.

Assuming the generated summaries belong to the same class as their original source texts, we can add them to the training samples.

3.3 Supervised Contrastive Learning

Although fine-tuning pretrained model using cross-entropy is commonly used for text classification, and it achieves state-of-art results on many text classification tasks(Yang et al., 2019). However, this approach still can not achieve optimal performance in few-shot setting, where training data is limited. In order to alleviate this limitation, we propose to add a supervised contrastive learning objective (Gunel et al., 2020) and using text summaries as contrastive samples to train a more robust text classifier under the limited annotation setting.

The main idea of supervised contrastive learning is minimizing the intra-class representation distance while maximizing the inter-class representation distance. It would be easier for the classifier to learn a good decision boundary by applying supervised contrastive learning. This process can be achieved by minimizing Equation 7.

For each batch with N input texts and N labels, we first apply summarization to get the augmented N text summaries; then, we get $2N$ samples in a
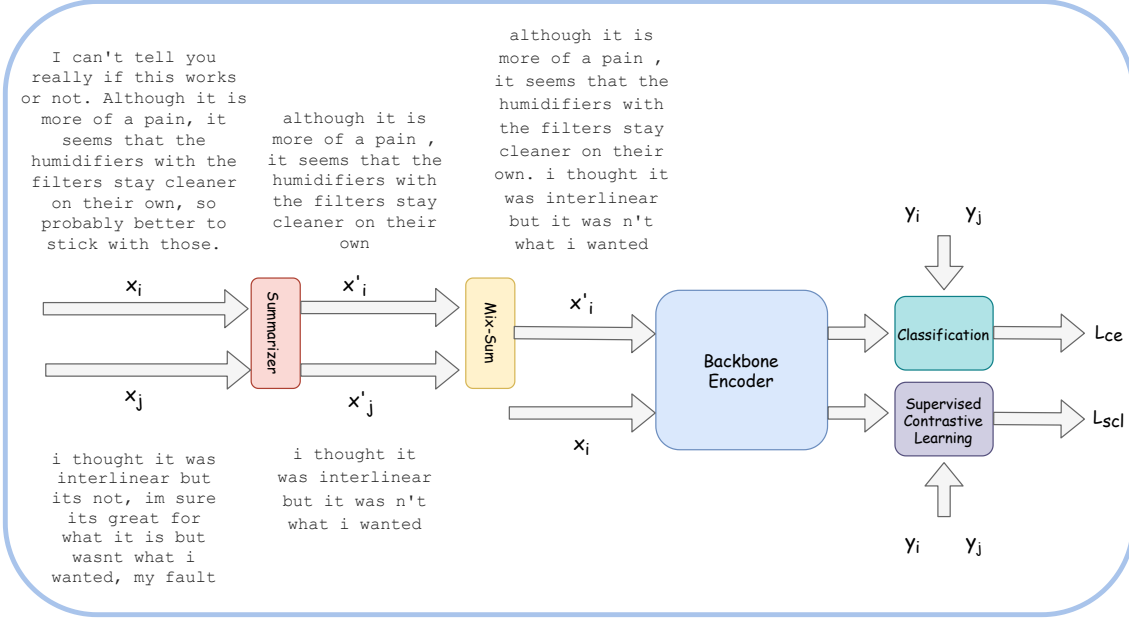
Figure 1: Illustration of using summaries as contrastive samples for text classification. x_i is the original text, x_i' is the summary of x_i, y_i is the target label for x_i. Randomly select another sample x_j, concatenate the summary of x_j-x_j' with x_i', and use it as the contrastive sample of x_i

batch. For each anchor sample x_i, we want to minimize the vector distance between x_i and positive samples x_j, whose labels y_i and y_j belong to the same class.

$$L_{sup}(X,Y) = -\frac{1}{2N}\sum_{i=1}^{2N}\frac{1}{N_{y_i}-1}\sum_{j=1}^{2N}$$

$$1_{i\neq j}1_{y_i=y_j}\frac{exp(g(x_i)\cdot g(x_j)/\tau)}{\sum_{k=1}^{2N}1_{k\neq i}exp(g(x_i)\cdot g(x_k)/\tau)} \tag{7}$$

Where N is the mini-batch size, and $2N$ is the size of the augmented batch after applying summarization. N_{y_i} is number of samples which have same labels as y_i. Labels for the summary is the same as the original text. X and Y are the batches of augmented training samples and target labels. $g(\cdot)$ is l_2 normalized representation of input text in $\mathbf{R}^n$, where n is the dimension of text feature used for supervised contrastive learning. The similarity measure of $g(\cdot)$ is cosine similarity with temperature parameter τ. The cosine similarity of $g(x_i)$ and $g(x_j)$ should be maximized when x_i and x_j come from the same class; otherwise it should be minimized.

Since contrastive learning can gain better performance when an MLP head is used (He et al., 2020), we also apply an MLP head upon the base text encoder $\Phi(\cdot)$. The text encoder $\Phi(\cdot)$ can be any pretrained text encoder which maps a text to a dense vector in $\mathbf{R}^d$, eg. BERT(Devlin et al., 2019),

XLNet(Yang et al., 2019), Roberta(Liu et al., 2019), LSTMs and CNNs(Zhang et al., 2015). d is the feature dimension of the text encoder. The entire text encoding process is expressed in Equation 8 and 9.

$$G(x) = MLP(\Phi(x)) \tag{8}$$

$$g(x) = \frac{G(x)}{||G(x)||} \tag{9}$$

Combining the cross-entropy loss in Equation 11 with a trade-off parameter λ, we can get the final loss function in Equation 10. λ is a hyperparameter to control the relative importance of cross-entropy loss and supervised contrastive loss.

$$L(X,Y) = \lambda\cdot L_{ce}(X,Y) + (1-\lambda)\cdot L_{sup}(X,Y) \tag{10}$$

$$L_{ce}(X,Y) = -\frac{1}{2N}\sum_{i=1}^{2N}$$

$$\mathbf{y_i}log(\mathbf{p}(x_i)) + (1-\mathbf{y_i})log(1-\mathbf{p}(x_i)) \tag{11}$$

$$\mathbf{p}(x_i) = \text{Softmax}(\mathbf{W}\cdot\Phi(x_i) + \mathbf{b}) \tag{12}$$

where $\mathbf{y_i}$ is the label of training sample x_i in one-hot representation. $\mathbf{p}(x_i)$ is the predicted probability distribution generated by the text classification model. $\Phi(\cdot)$ is the backbone text encoder, which is exactly the same as the text encoder used in the

supervised contrastive learning stage and the model weights also shared in supervised contrastive learning stage. $\mathbf{W}$ is a fully connected classification projection matrix in $\mathbf{R}^{C \times d}$, which map the text feature in $\mathbf{R}^d$ to score vector of output classes in $\mathbf{R}^C$. $\mathbf{b}$ is the bias of the classification head in $\mathbf{R}^C$. C is the number of different classes across the training samples.

3.4 Mixsum

We propose another novel method, i.e. Mixsum, by combining the idea of mix-up (Zhang et al., 2018) and using summarization to construct contrastive samples–to achieve better text classification performance under the limited annotation setting. Basically, the main idea is that summaries of concatenated texts from different classes contain the feature of both classes, then the newly generated summary can serve as the regularization for cross-entropy loss and supervised contrastive learning objective, which can lead the model to behave in between the training samples and soften the labels.

Similar to mixup (Zhang et al., 2018), which use a convex combination of the input image to create the vicinal distribution, we propose to combine the summaries of texts from two different classes and use the conjunct summary as the augmentation.

There are also other methods for mixing the texts from two different classes, such as linear interpolation of sentence-level features(Guo et al., 2019; Sun et al., 2020) and word-level features(Guo et al., 2019). Those methods are also applicable under our setting. In the summarization context, concatenating two documents with the same weight is the simplest and most intuitive way to keep our model neat and practical. Consequently, we choose this method for mixing up the texts and the λ for mixing the vicinal label in Equation 4 is also fixed at 0.5.

$$\hat{x}_i' = x_i' | x_j' \tag{13}$$

$$\hat{y}_i = 0.5 \cdot y_i + 0.5 \cdot y_j \tag{14}$$

Where x_i' is the summary of the original text x_i in a batch, then randomly pick another summary x_j' in the batch and conjunct them together to form a mix-up summary $\hat{x}_i'$. This process can be visualized in Figure 1. The new generated label $\hat{y}_i$ follows the mix-up method introduced in (Zhang et al., 2018).

Same as the contrastive samples augmentation strategy mentioned in Section 3.3, we concatenate

the original N input texts with the mix-up summaries to form a new Minibatch with $2N$ samples. Then we can formulate the new cross-entropy loss and supervised contrastive loss under Mixsum setting in Equation 15 and 19.

$$L_{ce}^{mix}(X, Y) = -\frac{1}{2N} \cdot$$
$$(\sum_{i=1}^{N} \boldsymbol{y_i} log(\boldsymbol{p}(x_i)) + (1 - \boldsymbol{y_i}) log(1 - \boldsymbol{p}(x_i))$$
$$\sum_{i=1}^{N} \hat{\boldsymbol{y_i}} log(\boldsymbol{p}(\hat{x_i'})) + (1 - \hat{\boldsymbol{y_i}}) log(1 - \boldsymbol{p}(\hat{x_i'})))$$
$$\tag{15}$$

The first N samples in the Minibatch X are original texts, and the loss of those N samples remains the same as the cross-entropy loss. The later N samples in the Minibatch are mix-up summary.

Taking the Equation 14 to Equation 15, we can further get the compact form for the cross entropy loss under Mixsum setting in Equation 16 and 17.

$$L_{ce}^{mix}(X, Y) = -0.5 \cdot \frac{1}{2N}$$
$$(\sum_{i=1}^{N} \boldsymbol{y_i} log(\boldsymbol{p}(x_i)) + (1 - \boldsymbol{y_i}) log(1 - \boldsymbol{p}(x_i)) +$$
$$\sum_{i=1}^{N} \boldsymbol{y_i} log(\boldsymbol{p}(\hat{x_i'})) + (1 - \boldsymbol{y_i}) log(1 - \boldsymbol{p}(\hat{x_i'})) +$$
$$\sum_{i=1}^{N} \boldsymbol{y_i} log(\boldsymbol{p}(x_i)) + (1 - \boldsymbol{y_i}) log(1 - \boldsymbol{p}(x_i)) +$$
$$\sum_{i=1}^{N} \boldsymbol{y_j} log(\boldsymbol{p}(\hat{x_i'})) + (1 - \boldsymbol{y_j}) log(1 - \boldsymbol{p}(\hat{x_i'})))$$
$$\tag{16}$$

$$L_{ce}^{mix}(X, Y) = 0.5 \cdot L_{ce}(X, Y) + 0.5 \cdot L_{ce}(X, Y_m) \tag{17}$$

$$Y_m = \{y_i\}^N | \{y_j\}^N \tag{18}$$

we can derive a similar compact form for supervised contrastive loss under Mixsum setting in Equation 19. The derivation is inspired by the cross entropy loss under Mixsum setting.

$$L_{sup}^{mix}(X, Y) \approx 0.5 \cdot L_{sup}(X, Y) + 0.5 \cdot L_{sup}(X, Y_m) \tag{19}$$

The constraints $1_{y_i = y_j}$ in Equation 7 can be written as $y_i \cdot y_j$, where y_i and y_j are the one hot label

vectors. Then in the Mixsum setting, each mixed label y_i^{mix} is obtained by $0.5 \cdot y_i + 0.5 \cdot y_i^m$, where $y_i \in Y$ and $y_i^m \in Y_m$. Thus, by expanding the LHS of Equation 19, we can replace the constraints $1_{y_i^{mix}=y_j^{mix}}$ with $y_i^{mix} \cdot y_j^{mix}$, which is

$$(0.5 \cdot y_i + 0.5 \cdot y_i^m) \cdot (0.5 \cdot y_j + 0.5 \cdot y_j^m) \quad (20)$$

Expanding Equation 20, we can get

$$0.25(y_i \cdot y_j + y_i \cdot y_j^m + y_i^m \cdot y_j + y_i^m \cdot y_j^m) \quad (21)$$

But Equation 21 is too complex for computation and also not neat, so we decided to do an approximation–using $y_i \cdot y_j + y_i^m \cdot y_j^m$ to approximate $y_i \cdot y_j^m + y_i^m \cdot y_j$. Then we can get

$$
\begin{aligned}
y_i^{mix} \cdot y_j^{mix} &\approx 0.5(y_i \cdot y_j + y_i^m \cdot y_j^m) \\
&\approx 0.5(1_{y_i=y_j} + 1_{y_i^m=y_j^m})
\end{aligned}
\quad (22)
$$

Benefit of doing this approximation is that it can reduce the complexity and make final form neat, and we commit that this approximation inevitably will lose some information.

Minimizing Equation 19 is sufficient to achieve the goal–pull the representation of Mixsum sample "in between" the representation of class y_j and y_i.

Finally, combining the cross-entropy loss and supervised contrastive loss under the Mixsum setting, we can get the final objective in Equation 23.

$$L^{mix}(X,Y) = \lambda L_{ce}^{mix}(X,Y) + (1-\lambda)L_{sup}^{mix}(X,Y) \quad (23)$$

4 Experiments

4.1 Datasets

We use Amazon-5, Yelp-5, AG News and IMDb text classification datasets for benchmarking, and the dataset splits are obtained from Zhang et al. (2015).

In order to demonstrate the effectiveness of the proposed methods under the limited annotation setting, we randomly sample ten subsets using ten different random seeds from each of Amazon-5, Yelp-5, AG-News and IMDb for each experiment, each subset contains 80 training samples and 1000 test samples. The statistics of sampled datasets is shown in Table 1.

4.2 Experimental Setting

For all the experiments, we test the proposed methods using several pretrained transformer models as backbone text-feature encoders including

Dataset	Train set	Test set	#Class
Amazon (S)	80	1000	5
Yelp (S)	80	1000	5
AG-News (S)	80	1000	4
IMDb (S)	80	1000	2

Table 1: Dataset statistics. (S) denotes the dataset sampled with small number of train samples.

Roberta-base model(Liu et al., 2019), and Bert-base model(Devlin et al., 2019). As for the pooling strategy of the backbone encoder, we simply use the feature of [CLS] token as the sentence feature, which is commonly used as the text feature for text classification. Adam optimizer (Kingma and Ba, 2015) is used for optimization. The maximum learning rate is set to $1e-5$, and the learning rate is decayed linearly with warm-up steps. The batch size is set to 8. We set the trade-off parameter λ to 0.9 for experiment involving L_{sup}, since 0.9 is the optimal trade-off parameter between supervised contrastive loss and cross-entropy loss when using Back-Translation for augmentation according to Gunel et al. (2020).

The summarization method we used for creating contrastive samples is PreSumm (Liu and Lapata, 2019), which is available on github[1], and we also use the Text-Rank algorithm for replacement when junk outputs are generated by PreSumm. It's inevitable for abstractive summarization methods like PreSumm to generate some junk outputs when certain input texts are given, and only a few junk outputs will be generated. Text-Rank is an extractive summarization method, which generates summaries by extracting existing sentences in the texts.

All of our code and datasets are available on the github repository[2].

4.3 Baselines

In order to testify the effectiveness of creating contrastive samples using summarization, we compare the proposed data augmentation strategy with Back-Translation(Edunov et al., 2018). Back-Translation is a common data augmentation strategy for contrastive learning in NLP(Fang et al., 2020). We first translate the training samples in English to Chinese and then translate back the Chinese texts to English using Google Translate.

[1] https://github.com/nlpyang/PreSumm
[2] https://github.com/ChesterDu/Contrastive_summary

We also conduct an ablation experiment under a setting that does not use summarization as contrastive samples. Under this setting, we simply remove the augmented samples in the data batch and only use original samples in the batch. The objective function under this setting only consists of cross-entropy loss and supervised contrastive loss of original samples.

4.4 Results

All the experiment results reported are the average results of repeating experiments with ten different random seeds. The experiment settings for producing all the results are introduced in Section 4.3 and 4.2.

4.4.1 Comparison to Baseline

Methods	Bert	Roberta
Amazon(S)		
BT	31.6	28.7
Sum	33.4	30.0
Mixsum	**34.1**	**35.2**
Yelp(S)		
BT	36.4	35.7
Sum	38.2	39.0
Mixsum	**38.9**	**42.0**
AG-News(S)		
BT	81.9	74.5
Sum	82.3	76.2
Mixsum	**83.7**	**76.5**
IMDb(S)		
BT	74.5	85.6
Sum	75.1	87.3
Mixsum	**76.6**	**87.7**

Table 2: Comparison to Back-Translation baseline. BT denotes the setting that using Back-Translation to create contrastive samples. Sum denotes the setting that using summarization to create contrastive samples proposed by us. Mixsum denotes the setting that using Mixsum for supervised contrastive learning.

We have two findings from the experiment results in Table 2. First, the proposed contrastive samples generation technique, i.e. summarization, outperforms the Back-Translation method(Edunov et al., 2018) under limited annotation setting on all four datasets. Second, the proposed Mixsum method can further improve the performance of

using summarization for contrastive samples generation(Sum).

4.4.2 Ablation Study

In order to demonstrate the effectiveness of the proposed two methods, we conduct ablation experiments on Amazon(S), Yelp(S), AG-News(S) and IMDb(S) to see the classification accuracy gain of each methods. The results are shown in Table 3, 4, 5 and 6. L_{ce} represents the setting that only use cross entropy loss and without any data augmentation. $L_{ce} + L_{sup}(N)$ represents the setting that do not use summarization as contrastive samples, and only use original samples for supervised contrastive learning. Under this setting, we can simply remove the augmented samples in the data batch and only use original samples in the minibatch. $L_{ce} + L_{sup}(Sum)$ represents the setting that uses summarization to create contrastive samples, which is introduced in Section 3.3. $L_{ce} + L_{sup}(Sum + BT)$ represents the setting that combine summarization and Back-Translation together for contrastive samples generation. $L_{ce}^{mix} + L_{sup}^{mix}$ is the setting that uses Mixsum introduced in Section 3.4 for supervised contrastive learning.

Methods	Bert	Roberta
L_{ce}	30.5	29.1
$L_{ce} + L_{sup}(N)$	31.5	28.0
$L_{ce} + L_{sup}(Sum)$	32.5	30.0
$L_{ce} + L_{sup}(Sum + BT)$	29.1	25.3
$L_{ce}^{mix} + L_{sup}^{mix}$	**34.1**	**35.2**

Table 3: Ablation Results on Amazon(S),.

Methods	Bert	Roberta
L_{ce}	34.1	35.9
$L_{ce} + L_{sup}(N)$	34.9	36.7
$L_{ce} + L_{sup}(Sum)$	38.2	39.0
$L_{ce} + L_{sup}(Sum + BT)$	34.6	37.1
$L_{ce}^{mix} + L_{sup}^{mix}$	**38.9**	**42.0**

Table 4: Ablation Results on Yelp(S).

We have four findings from the Ablation Results.

- The proposed summarization method can significantly increase the performance, and the average performance gain is 2.61% across all datasets and models compared to L_{ce} setting.

Methods	Bert	Roberta
L_{ce}	79.9	74.2
$L_{ce} + L_{sup}(N)$	80.1	70.7
$L_{ce} + L_{sup}(Sum)$	82.3	76.2
$L_{ce} + L_{sup}(Sum + BT)$	81.5	74.3
$L_{ce}^{mix} + L_{sup}^{mix}$	**83.7**	**76.5**

Table 5: Ablation Results on AG-News(S).

Methods	Bert	Roberta
L_{ce}	72.2	83.8
$L_{ce} + L_{sup}(N)$	72.9	85.6
$L_{ce} + L_{sup}(Sum)$	75.1	87.3
$L_{ce} + L_{sup}(Sum + BT)$	71.4	86.2
$L_{ce}^{mix} + L_{sup}^{mix}$	**76.6**	**87.7**

Table 6: Ablation Results on IMDb(S).

- The proposed Mixsum method can further improve the performance of the classifier. The average performance gain compared to L_{ce} setting is 4.38%, and the average performance gain compared to the summarization method is 1.7%.

- Supervised contrastive learning without any augmented contrastive samples may or may not increase the classifier performance, the average performance gain is 0.0875% across all datasets and models. Sometimes it would even decrease the performance of classifier.

- Combining Sum and BT samples together can not outperforms the setting that only use one of them.

4.4.3 Sensitive analysis

In order to investigate how the number of training examples impacts the performance of the proposed methods, we report the test accuracy on datasets with 800 and 6500 training examples. The trade-off parameter λ is set to 0.99. We only conduct the experiment using the Roberta-base model for convenience since we think that the results obtained from Roberta are representative enough according to Ablation Results. The results is shown in Table 7 and 8.

We observe that when the number of training samples increases, Mixsum still can achieve better performance in those three datasets compared to ablation methods. However, compared to results

Methods	Amazon(M)	Yelp(M)	AG(M)
L_{ce}	57.4	57.8	87.7
$L_{ce} + L_{sup}(N)$	57.6	57.4	87.4
$L_{ce} + L_{sup}$	56.7	58.1	87.4
$L_{ce}^{mix} + L_{sup}^{mix}$	**58.1**	**58.2**	**88.8**

Table 7: Test Accuracy on datasets with 800 training examples. (M) denotes the dataset sampled with 800 train samples.

Methods	Amazon(L)	Yelp(L)	AG(L)
L_{ce}	84.8	**61.0**	95.9
$L_{ce} + L_{sup}(N)$	84.6	59.8	95.9
$L_{ce} + L_{sup}(Sum)$	**84.8**	60.4	95.7
$L_{ce}^{mix} + L_{sup}^{mix}$	84.0	60.6	**96.3**

Table 8: Test Accuracy on datasets with 6500 training examples (L) denotes the dataset sampled with 6500 train samples.

when the number of training samples is only 80 in Section 4.4.2, we find that performance improvement of the proposed two methods is much smaller. When the number of training samples increases to 6500, the performance of the proposed methods even lower than the ablation setting. Combining results from Section 4.4.2, it's reasonable to infer that the proposed methods are beneficial under the limited annotation scenario, but they may not necessary when the number of training samples get larger.

In order to investigate how summarization methods will impact the performance of the proposed methods, we replace the original abstractive summarization method–PreSumm(Liu and Lapata, 2019) with extractive summarization method–TextRank. TextRank algorithm will rank the relative importance of the sentences in a text and select the most important sentence as the text summary. We report the test accuracy of using TextRank for text summarization in Table 9.

Methods	Amazon(S)	Yelp(S)	AG(S)
L_{ce}	29.1	35.9	74.2
$L_{ce} + L_{sup}(N)$	28.0	36.7	70.7
$L_{ce} + L_{sup}(Sum)$	26.7	38.5	75.7
$L_{ce}^{mix} + L_{sup}^{mix}$	**29.6**	**41.2**	**76.2**

Table 9: Text Accuracy on datasets by using extractive summarization

With this alternative summarization system, the performance of the proposed mix-sum regularization methods is not as good as using PreSumm. We think that it is the limitation of the extractive summarization that leads to the performance drop because extractive summarization can only create summaries from original texts and will bring more information loss compared to abstractive summarization. Besides, the performance of the proposed Mixsum regularization still outperforms other ablation models, which proved the generalization ability of the proposed Mixsum method over different summarization methods.

Furthermore, we also investigated effect of using different texts mixing methods. Sun et al. (2020) propose to mix the texts by linearly interpolating sentence-level features of texts. The sentence-level features are encoded by a pre-trained transformer model, like BERT and Roberta. We replace our texts mixing methods with the linear interpolation of sentence-level feature as introduced by Sun et al. (2020), and keep all other settings same as Mixsum introduced in Section 3 and 4. The results are shown in Table 10. All the experiment are repeated with 10 different random seeds.

Methods	Amazon(S)	Yelp(S)	AG(S)
Sum	30.0	39.0	76.2
Mixsum(Ours)	35.2	42.0	76.5
Mixsum(LISF)	32.5	41.1	77.3

Table 10: Comparison of using linear interpolation of sentence-level feature(LISF) as texts mixing methods with concatenation of summary texts(Ours).

We observe that replacing our texts mixing methods with LISF still can achieve similar results and outperforms the Sum setting. Thus, we believe that other different sentence mixing methods can also be adopted in Mixsum framework.

5 Conclusion

We proposed a novel data augmentation technique for constructing contrastive samples in supervised contrastive learning–summarization. Besides, we also proposed a Mixsum method based on using summarization to construct the contrastive samples. We demonstrate the effectiveness of the proposed two new techniques on text classification task under the limited annotation setting. The experiment results on four datasets show that Mixsum and using

summarization as contrastive samples can improve the performance of text classification under the limited annotations setting. Besides, We show that the proposed Mixsum methods can be generalized to different summarization methods and text mixing methods.

Our work also opens up several possibilities for future work, since using summarization to construct contrastive samples has shown the effectiveness in supervised contrastive learning. We may investigate whether using summarization as data augmentation can improve unsupervised text classification (Wu et al., 2018), and the robustness and performance of other NLP applications like question answering, commonsense reasoning and semantic code retrieval(Ling et al., 2021).

References

Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. 2001. Vicinal risk minimization. In *Advances in Neural Information Processing Systems*, volume 13, pages 416–422. MIT Press.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. CERT: Contrastive Self-supervised Learning for Language Understanding. *arXiv:2005.12766 [cs, stat]*. ArXiv: 2005.12766.

Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. 2019. Boosting few-shot visual learning with self-supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8059–8068.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. *CoRR*, abs/2011.01403.

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

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Sungnyun Kim, Gihun Lee, Sangmin Bae, and Se-Young Yun. 2020. MixCo: Mix-up Contrastive Learning for Visual Representation. *arXiv:2010.06300 [cs]*. ArXiv: 2010.06300.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tassilo Klein and Moin Nabi. 2020. Contrastive self-supervised learning for commonsense reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7517–7523, Online. Association for Computational Linguistics.

Xiang Ling, Lingfei Wu, Saizhuo Wang, Gaoning Pan, Tengfei Ma, Fangli Xu, Alex X Liu, Chunming Wu, and Shouling Ji. 2021. Deep graph matching and searching for semantic code retrieval. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–21.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

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

Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. 2020. When does self-supervision improve few-shot learning? In *European Conference on Computer Vision*, pages 645–666. Springer.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip Yu, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for NLP tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440, Barcelona, Spain (Online). International Committee on Computational Linguistics.

V. N. Vapnik. 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999.

Lingfei Wu, Ian EH Yen, Kun Xu, Fangli Xu, Avinash Balakrishnan, Pin-Yu Chen, Pradeep Ravikumar, and Michael J Witbrock. 2018. Word mover's embedding: From word2vec to document embedding. In *2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

A Pseudo-code

Pseudo-code of using summaries for supervised contrastive learning, mentioned in Section 3.3.

Algorithm 1 Pseudo-code of Sum

Require: Initialize the backbon encoder f, classification head $proj$ and mlp head mlp.

   Trade off parameter λ.

   for sampled minibatch $\{x_k\}_{k=1}^N, \{y_k\}_{k=1}^N$ **do**

      for $k \in \{1, ...N\}$ **do**

         $\hat{x}_{2k} = Summ(x_k)$ // use summarization for augmentation

         $z_{2k} = f(\hat{x}_{2k-1})$ // get backbone representation

         $s_{2k} = proj(z_{2k-1})$ // project the representation to prediction score

         $g_{2k} = mlp(z_{2k-1})$ // Apply MLP head to get feature representation of summary

         $g_{2k} = Norm(g_{2k})$ // Normalize the feature vector

         $\hat{x}_k = x_k$ // original texts

         $z_k = f(\hat{x}_k)$ // get backbone representation

         $s_k = proj(z_k)$ // project the representation to prediction score

         $g_k = mlp(z_k)$ // Apply MLP head to get feature representation of summary

         $g_k = Norm(g_k)$ // Normalize the feature vector

         $\hat{y}_{2k} = y_k$ //the label of summary is same as original text

         $\hat{y}_k = y_k$

      end for

      $l_{ce} = CrossEnropy(\{s_k\}_{k=1}^{2N}, \{\hat{y}_k\}_{k=1}^{2N})$ // cross entropy loss of augmented batch

      $l_{sup} = SupConLoss(\{g_k\}_{k=1}^{2N}, \{\hat{y}_k\}_{k=1}^{2N}))$ // contrastive loss of augmented batch

      $L = \lambda l_{ce} + (1-\lambda) l_{sup}$ //compute total loss

      Compute $\nabla_{\theta_f} L, \nabla_{\theta_{proj}} L, \nabla_{\theta_{mlp}} L$

      Update $f, proj, mlp$ to optimize L

   end for

Pseudo-code of using Mixsum for supervised contrastive learning, mentioned in Section 3.4.

Algorithm 2 Pseudo-code of Mixsum

---

for sampled minibatch $\{x_k\}_{k=1}^N, \{y_k\}_{k=1}^N$ **do**
   $perm\_index = shuffle(\{1...N\})$
   for $k \in \{1, ...N\}$ **do**
      $j = perm\_index[k]$ // permutation index

      $\hat{x}_{2k} = Summ(x_k) + Summ(x_j)$ // use mix-sum for augmentation
      $z_{2k} = f(\hat{x}_{2k})$ // get backbone representation
      $s_{2k} = proj(z_{2k})$ // project the representation to prediction score
      $g_{2k} = mlp(z_{2k})$ //Apply MLP head to get feature vector
      $g_{2k} = Norm(g_{2k})$ //Normalize the feature vector

      $\hat{x}_k = x_k$ //original texts
      $z_k = f(\hat{x}_k)$ // get backbone representation
      $s_k = proj(z_k)$ // project the representation to prediction score
      $g_k = mlp(z_k)$ //Apply MLP head to get feature vector
      $g_k = Norm(g_{2k})$ //Normalize the feature vector

      $\hat{y}_k = y_k$ // mix the label
      $\hat{y}_{2k} = y_k$
      $\tilde{y}_k = y_k$
      $\tilde{y}_{2k} = y_j$
   end for
   $l_{ce} = CrossEnropy(\{s_k\}_{k=1}^{2N}, \{\hat{y}_k\}_{k=1}^{2N})/2$
   $l_{ce}+ = CrossEnropy(\{s_k\}_{k=1}^{2N}, \{\tilde{y}_k\}_{k=1}^{2N})/2$
   $l_{scl} = SupConLoss(\{g_k\}_{k=1}^{2N}, \{\hat{y}_k\}_{k=1}^{2N}))/2$
   $l_{scl}+ = SupConLoss(\{g_k\}_{k=1}^{2N}, \{\tilde{y}_k\}_{k=1}^{2N}))/2$
   $L = \lambda l_{ce} + (1 - \lambda)l_{scl}$

   Compute $\nabla_{\theta_f} L, \nabla_{\theta_{proj}} L, \nabla_{\theta_{mlp}} L$
   Update $f, proj, mlp$ to optimize L

end for