MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification

Jiaao Chen
Georgia Tech
jchen896@gatech.edu

Zichao Yang
CMU
zichaoy@cs.cmu.edu

Diyi Yang
Georgia Tech
dyang888@gatech.edu

Abstract

This paper presents MixText, a semi-supervised learning method for text classification, which uses our newly designed data augmentation method called TMix. TMix creates a large amount of augmented training samples by interpolating text in *hidden space*. Moreover, we leverage recent advances in data augmentation to guess low-entropy labels for unlabeled data, hence making them as easy to use as labeled data. By mixing labeled, unlabeled and augmented data, MixText significantly outperformed current pre-trained and fined-tuned models and other state-of-the-art semi-supervised learning methods on several text classification benchmarks. The improvement is especially prominent when supervision is extremely limited. We have publicly released our code at https://github.com/GT-SALT/MixText.

1 Introduction

In the era of deep learning, research has achieved extremely good performance in most supervised learning settings (LeCun et al., 2015; Yang et al., 2016). However, when there is only limited labeled data, supervised deep learning models often suffer from over-fitting (Xie et al., 2019). This strong dependence on labeled data largely prevents neural network models from being applied to new settings or real-world situations due to the need of large amount of time, money, and expertise to obtain enough labeled data. As a result, semi-supervised learning has received much attention to utilize both labeled and unlabeled data for different learning tasks, as unlabeled data is always much easier and cheaper to collect (Chawla and Karakoulas, 2011).

This work takes a closer look at semi-supervised text classification, one of the most fundamental tasks in language technology communities. Prior research on semi-supervised text classification can be categorized into several classes: (1) utilizing variational auto encoders (VAEs) to reconstruct the sentences and predicting sentence labels with latent variables learned from reconstruction such as (Chen et al., 2018; Yang et al., 2017; Gururangan et al., 2019); (2) encouraging models to output confident predictions on unlabeled data for self-training like (Lee, 2013; Grandvalet and Bengio, 2004; Meng et al., 2018); (3) performing consistency training after adding adversarial noise (Miyato et al., 2019, 2017) or data augmentations (Xie et al., 2019); (4) large scale pretraining with unlabeld data, then finetuning with labeled data (Devlin et al., 2019). Despite the huge success of those models, most prior work utilized labeled and unlabeled data *separately* in a way that no supervision can transit from labeled to unlabeled data or from unlabeled to labeled data. As a result, most semi-

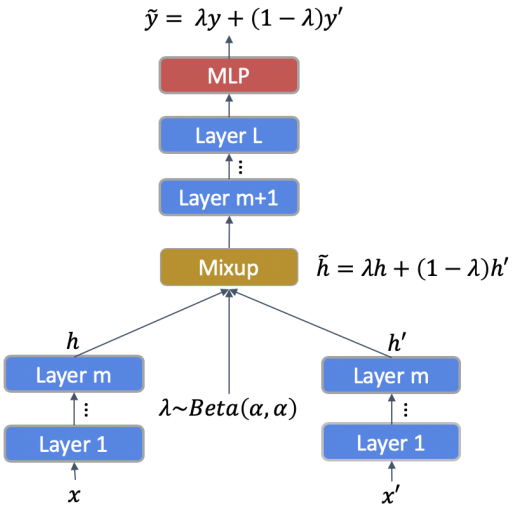

Figure 1: TMix takes in two text samples x and x' with labels y and y', mixes their hidden states h and h' at layer m with weight λ into $\tilde{h}$, and then continues forward passing to predict the mixed labels $\tilde{y}$.

supervised models can easily still overfit on the very limited labeled data, despite unlabeled data is abundant.

To overcome the limitations, in this work, we introduce a new data augmentation method, called **TMix** (Section 3), inspired by the recent success of Mixup (Gururangan et al., 2019; Berthelot et al., 2019) on image classifications. TMix, as shown in Figure 1, takes in two text instances, and interpolates them in their corresponding hidden space. Since the combination is continuous, TMix has the potential to create infinite mount of new augmented data samples, thus can drastically avoid overfitting. Based on TMix, we then introduce a new semi-supervised learning method for text classification called **MixText** (Section 4) to explicitly model the relationships between *labeled and unlabeled* samples, thus overcoming the limitations of previous semi-supervised models stated above. In a nutshell, MixText first guesses low-entropy labels for unlabeled data, then uses TMix to interpolate the label and unlabeled data. MixText can facilitate mining implicit relations between sentences by encouraging models to behave linearly in-between training examples, and utilize information from unlabeled sentences while learning on labeled sentences. In the meanwhile, MixText exploits several semi-supervised learning techniques to further utilize unlabeled data including self-target-prediction (Laine and Aila, 2016), entropy minimization (Grandvalet and Bengio, 2004), and consistency regularization (Berthelot et al., 2019; Xie et al., 2019) after back translations.

To demonstrate the effectiveness of our method, we conducted experiments (Section 5) on four benchmark text classification datasets and compared our method with previous state-of-the-art semi-supervised method, including those built upon models pre-trained with large amount of unlabeled data, in terms of accuracy on test sets. We further performed ablation studies to demonstrate each component's influence on models' final performance. Results show that our MixText method significantly outperforms baselines especially when the given labeled training data is extremely limited.

2 Related Work

2.1 Pre-training and Fine-tuning Framework

The pre-training and fine-tuning framework has achieved huge success on NLP applications in recent years, and has been applied to a variety of NLP tasks (Radford et al., 2018; Chen et al., 2019; Akbik et al., 2019). Howard and Ruder (2018) proposed to pre-train a language model on a large general-domain corpus and fine-tune it on the target task using some novel techniques like discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing. In this manner, such pre-trained models show excellent performance even with small amounts of labeled data. Pre-training methods are often designed with different objectives such as language modeling (Peters et al., 2018; Howard and Ruder, 2018; Yang et al., 2019b) and masked language modeling (Devlin et al., 2019; Lample and Conneau, 2019). Their performances are also improved with training larger models on more data (Yang et al., 2019b; Liu et al., 2019).

2.2 Semi-Supervised Learning on Text Data

Semi-supervised learning has received much attention in the NLP community (Gururangan et al., 2019; Clark et al., 2018; Yang et al., 2015), as unlabeled data is often plentiful compared to labeled data. For instance, Gururangan et al. (2019); Chen et al. (2018); Yang et al. (2017) leveraged variational auto encoders (VAEs) in a form of sequence-to-sequence modeling on text classification and sequential labeling. Miyato et al. (2017) utilized adversarial and virtual adversarial training to the text domain by applying perturbations to the word embeddings. Yang et al. (2019a) took advantage of hierarchy structures to utilize supervision from higher level labels to lower level labels. Xie et al. (2019) exploited consistency regularization on unlabeled data after back translations and tf-idf word replacements. Clark et al. (2018) proposed cross-veiw training for unlabeled data, where they used an auxiliary prediction modules that see restricted views of the input (e.g., only part of a sentence) and match the predictions of the full model seeing the whole input.

2.3 Interpolation-based Regularizers

Interpolation-based regularizers (e.g., Mixup) have been recently proposed for supervised learning (Zhang et al., 2017; Verma et al., 2019a) and semi-supervised learning (Berthelot et al., 2019; Verma et al., 2019b) for image-format data by overlaying two input images and combining image labels as virtual training data and have achieved state-of-the-art performances across a variety of tasks like image classification and network architectures. Different variants of mixing methods have also been

designed such as performing interpolations in the input space (Zhang et al., 2017), combining interpolations and cutoff (Yun et al., 2019), and doing interpolations in the hidden space representations (Verma et al., 2019a,c). However, such interpolation techniques have not been explored in the NLP field because most input space in text is discrete, i.e., one-hot vectors instead of continues RGB values in images, and text is generally more complex in structures.

2.4 Data Augmentations for Text

When labeled data is limited, data augmentation has been a useful technique to increase the amount of training data. For instance, in computer vision, images are shifted, zoomed in/out, rotated, flipped, distorted, or shaded with a hue (Perez and Wang, 2017) for training data augmentation. But it is relatively challenging to augment text data because of its complex syntactic and semantic structures. Recently, Wei and Zou (2019) utilized synonym replacement, random insertion, random swap and random deletion for text data augmentation. Similarly, Kumar et al. (2019) proposed a new paraphrasing formulation in terms of monotone submodular function maximization to obtain highly diverse paraphrases, and Xie et al. (2019) and Chen et al. (2020) applied back translations (Sennrich et al., 2015) and word replacement to generate paraphrases on unlabeled data for consistency training. Other work which also investigates noise and its incorporation into semi-supervised named entity classification (Lakshmi Narayan et al., 2019; Nagesh and Surdeanu, 2018).

3 TMix

In this section, we extend Mixup–a data augmentation method originally proposed by (Zhang et al., 2017) for images–to text modeling. The main idea of Mixup is very simple: given two labeled data points $(\mathbf{x}_i, \mathbf{y}_i)$ and $(\mathbf{x}_j, \mathbf{y}_j)$, where $\mathbf{x}$ can be an image and $\mathbf{y}$ is the one-hot representation of the label, the algorithm creates virtual training samples by linear interpolations:

$$\tilde{\mathbf{x}} = \text{mix}(\mathbf{x}_i, \mathbf{x}_j) = \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j, \quad (1)$$
$$\tilde{\mathbf{y}} = \text{mix}(\mathbf{y}_i, \mathbf{y}_j) = \lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j, \quad (2)$$

where $\lambda \in [0, 1]$. The new virtual training samples are used to train a neural network model. Mixup can be interpreted in different ways. On one hand, Mixup can be viewed a data augmentation approach which creates new data samples based on the original training set. On the other hand, it enforces a regularization on the model to behave linearly among the training data. Mixup was demonstrated to work well on continuous image data (Zhang et al., 2017). However, extending it to text seems challenging since it is infeasible to compute the interpolation of discrete tokens.

To this end, we propose a novel method to overcome this challenge — *interpolation in textual hidden space*. Given a sentence, we often use a multi-layer model like BERT (Devlin et al., 2019) to encode the sentences to get the semantic representations, based on which final predictions are made. Some prior work (Bowman et al., 2016) has shown that decoding from an interpolation of two hidden vectors generates a new sentence with mixed meaning of two original sentences. Motivated by this, we propose to apply interpolations within hidden space as a data augment method for text. For an encoder with L layers, we choose to mixup the hidden representation at the m-th layer, $m \in [0, L]$.

As demonstrated in Figure 1, we first compute the hidden representations of two text samples separately in the bottom layers. Then we mix up the hidden representations at layer m, and feed the interpolated hidden representations to the upper layers. Mathematically, denote the l-th layer in the encoder network as $g_l(.; \boldsymbol{\theta})$, hence the hidden representation of the l-th layer can be computed as $\mathbf{h}_l = g_l(\mathbf{h}_{l-1}; \boldsymbol{\theta})$. For two text samples $\mathbf{x}_i$ and $\mathbf{x}_j$, define the 0-th layer as the embedding layer, i.e., $\mathbf{h}_0^i = \mathbf{W}_{\text{E}}\mathbf{x}_i, \mathbf{h}_0^j = \mathbf{W}_{\text{E}}\mathbf{x}_j$, then the hidden representations of the two samples from the lower layers are:

$$\mathbf{h}_l^i = g_l(\mathbf{h}_{l-1}^i; \boldsymbol{\theta}), l \in [1, m],$$
$$\mathbf{h}_l^j = g_l(\mathbf{h}_{l-1}^j; \boldsymbol{\theta}), l \in [1, m].$$

The mixup at the m-th layer and continuing forward passing to upper layers are defined as:

$$\tilde{\mathbf{h}}_m = \lambda \mathbf{h}_m^i + (1-\lambda)\mathbf{h}_m^j,$$
$$\tilde{\mathbf{h}}_l = g_l(\tilde{\mathbf{h}}_{l-1}; \boldsymbol{\theta}), l \in [m+1, L].$$

We call the above method **TMix** and define the new mixup operation as the whole process to get $\tilde{\mathbf{h}}_L$:

$$\text{TMix}(\mathbf{x}_i, \mathbf{x}_j; g(.; \boldsymbol{\theta}), \lambda, m) = \tilde{\mathbf{h}}_L.$$

By using an encoder model $g(.; \boldsymbol{\theta})$, TMix interpolates textual semantic hidden representations

as a type of data augmentation. In contrast with Mixup defined in the data space in Equation 1, TMix depends on an encoder function, hence defines a much broader scope for computing interpolations. For ease of notation, we drop the explicit dependence on $g(.;\boldsymbol{\theta})$, λ and m in notations and denote it simply as $\text{TMix}(\mathbf{x}_i, \mathbf{x}_j)$ in the following sections.

In our experiments, we sample the mix parameter λ from a Beta distribution for every batch to perform the interpolation :

$$\lambda \sim \text{Beta}(\alpha, \alpha),$$
$$\lambda = \max(\lambda, 1-\lambda),$$

in which α is the hyper-parameter to control the distribution of λ. In TMix, we mix the labels in the same way as Equation 2 and then use the pairs $(\tilde{\mathbf{h}}_L, \tilde{y})$ as inputs for downstream applications.

Instead of performing mixup at random input layers like Verma et al. (2019a), choosing which layer of the hidden representations to mixup is an interesting question to investigate. In our experiments, we use 12-layer BERT-base (Devlin et al., 2019) as our encoder model. Recent work (Jawahar et al., 2019) has studied what BERT learned at different layers. Specifically, the authors found {3,4,5,6,7,9,12} layers have the most representation power in BERT and each layer captures different types of information ranging from surface, syntactic to semantic level representation of text. For instance, the 9-th layer has predictive power in semantic tasks like checking random swapping of coordinated clausal conjuncts, while the 3-rd layer performs best in surface tasks like predicting sentence length.

Building on those findings, we choose the layers that contain both syntactic and semantic information as our mixing layers, namely $\mathbf{M} = \{7, 9, 12\}$. For every batch, we *randomly sample* m, the layer to mixup representations, from the set $\mathbf{M}$ computing the interpolation. We also performed ablation study in Section 5.5 to show how TMix's performance changes with different choice of mix layer sets.

Text classification Note that TMix provides a general approach to augment text data, hence can be applied to any downstream tasks. In this paper, we focus on text classification and leave other applications as potential future work. In text classification, we minimize the KL-divergence between the mixed labels and the probability from the classifier as the supervision loss:

$$L_{\text{TMix}} = \text{KL}(\text{mix}(\mathbf{y}_i, \mathbf{y}_j) || p(\text{TMix}(\mathbf{x}_i, \mathbf{x}_j); \boldsymbol{\phi})$$

where $p(.;\boldsymbol{\phi})$ is a classifier on top of the encoder model. In our experiments, we implement the classifier as a two-layer MLP, which takes the mixed representation $\text{TMix}(\mathbf{x}_i, \mathbf{x}_j)$ as input and returns a probability vector. We jointly optimize over the encoder parameters $\boldsymbol{\theta}$ and the classifier parameters $\boldsymbol{\phi}$ to train the whole model.

4 Semi-supervised MixText

In this section, we demonstrate how to utilize the TMix to help semi-supervised learning. Given a limited labeled text set $\mathbf{X}_l = \{\mathbf{x}_1^l, ..., \mathbf{x}_n^l\}$, with their labels $\mathbf{Y}_l = \{\mathbf{y}_1^l, ..., \mathbf{y}_n^l\}$ and a large unlabeled set $\mathbf{X}_u = \{\mathbf{x}_1^u, ..., \mathbf{x}_m^u\}$, where n and m are the number of data points in each set. $\mathbf{y}_i^l \in \{0,1\}^C$ is a one-hot vector and C is the number of classes. Our goal is to learn a classifier that efficiently utilizes both labeled data and unlabeled data.

We propose a new text semi-supervised learning framework called **MixText** [1]. The core idea behind our framework is to leverage TMix both on labeled and unlabeled data for semi-supervised learning. To fulfill this goal, we come up a label guessing method to generate labels for the unlabeled data in the training process. With the guessed labels, we can treat the unlabeled data as additional labeled data and perform TMix for training. Moreover, we combine TMix with additional data augmentation techniques to generate large amount of augmented data, which is a key component that makes our algorithm work well in setting with extremely limited supervision. Finally, we introduce an entropy minimization loss that encourages the model to assign sharp probabilities on unlabeled data samples, which further helps to boost performance when the number of classes C is large. The overall architecture is shown in Figure 2. We will explain each component in detail.

4.1 Data Augmentation

Back translations (Edunov et al., 2018) is a common data augmentation technique and can generate diverse paraphrases while preserving the semantics of the original sentences. We utilize back translations to paraphrase the unlabeled data. For each

[1] Note that MixText is a semi-supervised learning framework while TMix is a data augmentation approach.

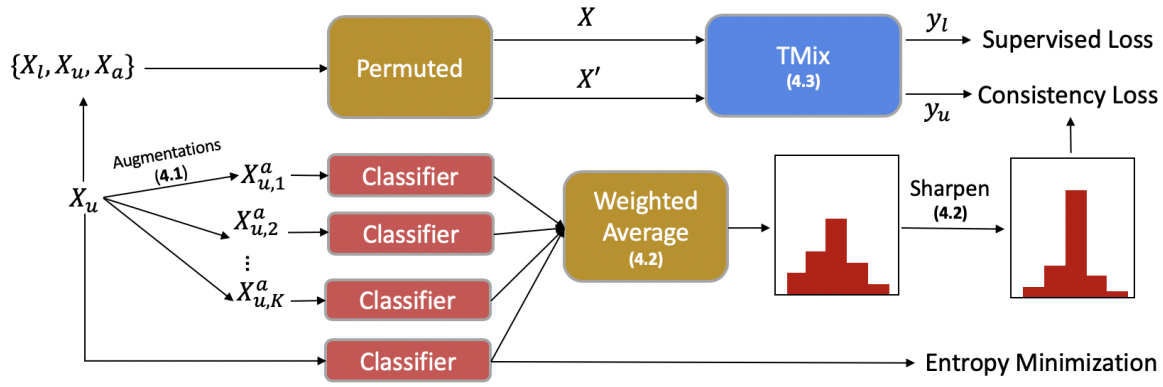

Figure 2: Overall Architecture of MixText. MixText takes in labeled data and unlabeled data, conducts augmentations and predicts labels for unlabeled data, performs TMix over labeled and unlabeled data, and computes supervised loss, consistency loss and entropy minimization term.

$\mathbf{x}^u_i$ in the unlabeled text set $\mathbf{X}_u$, we generate K augmentations $\mathbf{x}^a_{i,k} = \text{augment}_k(\mathbf{x}^u_i), k \in [1, K]$ by back translations with different intermediate languages. For example, we can translate original sentences from English to German and then translate them back to get the paraphrases. In the augmented text generation, we employ random sampling with a tunable temperature instead of beam search to ensure the diversity. The augmentations are then used for generating labels for the unlabeled data, which we describe below.

4.2 Label Guessing

For an unlabeled data sample $\mathbf{x}^u_i$ and its K augmentations $\mathbf{x}^a_{i,k}$, we generate the label for them using weighted average of the predicted results from the current model:

$$\mathbf{y}^u_i = \frac{1}{w_{ori} + \sum_k w_k}(w_{ori} p(\mathbf{x}^u_i) + \sum_{k=1}^{K} w_k p(\mathbf{x}^a_{i,k}))).$$

Note that $\mathbf{y}^u_i$ is a probability vector. We expect the model to predict consistent labels for different augmentations. Hence, to enforce the constraint, we use the weighted average of all predictions, rather than the prediction of any single data sample, as the generated label. Moreover, by explicitly introducing the weight w_{ori} and w_k, we can control the contributions of different quality of augmentations to the generated labels. Our label guessing method improves over (Tarvainen and Valpola, 2017) which utilizes teacher and student models to predict labels for unlabeled data, and UDA (Xie et al., 2019) that just uses $p(\mathbf{x}^u_i)$ as generated labels.

To avoid the weighted average being too uniform, we utilize a sharpening function over predicted labels. Given a temperature hyper-parameter T:

$$\text{Sharpen}(\mathbf{y}^u_i, T) = \frac{(\mathbf{y}^u_i)^{\frac{1}{T}}}{||(\mathbf{y}^u_i)^{\frac{1}{T}}||_1},$$

where $||.||_1$ is l_1-norm of the vector. When $T \to 0$, the generated label becomes a one-hot vector.

4.3 TMix on Labeled and Unlabeled Data

After getting the labels for unlabeled data, we merge the labeled text $\mathbf{X}_l$, unlabeled text $\mathbf{X}_u$ and unlabeled augmentation text $\mathbf{X}_a = \{x^a_{i,k}\}$ together to form a super set $\mathbf{X} = \mathbf{X}_l \cup \mathbf{X}_u \cup \mathbf{X}_a$. The corresponding labels are $\mathbf{Y} = \mathbf{Y}_l \cup \mathbf{Y}_u \cup \mathbf{Y}_a$, where $\mathbf{Y}^a = \{\mathbf{y}^a_{i,k}\}$ and we define $\mathbf{y}^a_{i,k} = \mathbf{y}^u_i$, i.e., the all augmented samples share the same generated label as the original unlabeled sample.

In training, we randomly sample two data points $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$, then we compute $\text{TMix}(\mathbf{x}, \mathbf{x}')$, $\text{mix}(\mathbf{y}, \mathbf{y}')$ and use the KL-divergence as the loss:

$$L_{\text{TMix}} = \mathbb{E}_{\mathbf{x},\mathbf{x}' \in \mathbf{X}} \text{KL}(\text{mix}(\mathbf{y}, \mathbf{y}') || p(\text{TMix}(\mathbf{x}, \mathbf{x}'))).$$

Since $\mathbf{x}, \mathbf{x}'$ are randomly sampled from $\mathbf{X}$, we interpolate text from many different categories: mixup among among labeled data, mixup of labeled and unlabeled data and mixup of unlabeled data. Based on the categories of the samples, the loss can be divided into two types:

Supervised loss When $\mathbf{x} \in \mathbf{X}_l$, the majority information we are actually using is from the labeled data, hence training the model with supervised loss.

Consistency loss When the samples are from unlabeled or augmentation set, i.e., $\mathbf{x} \in \mathbf{X}^u \cup \mathbf{X}^a$, most information coming from unlabeled data, the KL-divergence is a type of consistency loss, constraining augmented samples to have the same labels with the original data sample.

4.4 Entropy Minimization

To encourage the model to produce confident labels on unlabeled data, we propose to minimize the entropy of prediction probability on unlabeled data as a self-training loss:

$$L_{\text{margin}} = \mathbb{E}_{\mathbf{x} \in \mathbf{X}_u} \max(0, \gamma - ||\mathbf{y}^u||_2^2),$$

where γ is the margin hyper-parameter. We minimize the entropy of the probability vector if it is larger than γ.

Combining the two losses, we get the overall objective function of MixText:

$$L_{\text{MixText}} = L_{\text{TMix}} + \gamma_m L_{\text{margin}}.$$

5 Experiments

5.1 Dataset and Pre-processing

We performed experiment with four English text classification benchmark datasets: AG News (Zhang et al., 2015), BPpedia (Mendes et al., 2012), Yahoo! Answers (Chang et al., 2008) and IMDB (Maas et al., 2011). We used the original test set as our test set and randomly sampled from the training set to form the training unlabeled set and development set. The dataset statistics and split information are presented in Table 1.

For unlabeled data, we selected German and Russian as intermediate languages for back translations using FairSeq[2], and the random sampling temperature was 0.9. Here is an example, for a news from AG News dataset: *"Oil prices rallied to a record high above $55 a barrel on Friday on rising fears of a winter fuel supply crunch and robust economic growth in China, the world's number two user"*, the augment texts through German and Russian are: *"Oil prices surged to a record high above $55 a barrel on Friday on growing fears of a winter slump and robust economic growth in world No.2 China"* and *"Oil prices soared to record highs above $55 per barrel on Friday amid growing fears over a winter reduction in U.S. oil inventories and robust economic growth in China, the world's second-biggest oil consumer"*.

[2] https://github.com/pytorch/fairseq

5.2 Baselines

To test the effectiveness of our method, we compared it with several recent models:

- **VAMPIRE** (Gururangan et al., 2019): VAriational Methods for Pretraining In Resource-limited Environments(VAMPIRE) pretrained a unigram document model as a variational autoencoder on in-domain, unlabeled data and used its internal states as features in a downstream classifier.
- **BERT** (Devlin et al., 2019): We used the pre-trained BERT-based-uncased model[3] and fine-tuned it for the classification. In details, we used average pooling over the output of BERT encoder and the same two-layer MLP as used in MixText to predict the labels.
- **UDA** (Xie et al., 2019): Since we do not have access to TPU and need to use smaller amount of unlabeled data, we implemented Unsupervised Data Augmentation(UDA) using pytorch by ourselves. Specifically, we used the same BERT-based-uncased model, unlabeled augment data and batch size as our MixText, used original unlabeled data to predict the labels with the same softmax sharpen temperature as our MixText and computed consistency loss between augmented unlabeled data.

5.3 Model Settings

We used BERT-based-uncased tokenizer to tokenize the text, bert-based-uncased model as our text encoder, and used average pooling over the output of the encoder, a two-layer MLP with a 128 hidden size and $tanh$ as its activation function to predict the labels. The max sentence length is set as 256. We remained the first 256 tokens for sentences that exceed the limit. The learning rate is 1e-5 for BERT encoder, 1e-3 for MLP. For α in the beta distribution, generally, when labeled data is fewer than 100 per class, α is set as 2 or 16, as larger α is more likely to generate λ around 0.5, thus creating "newer" data as data augmentations; when labeled data is more than 200 per class, α is set to 0.2 or 0.4, as smaller α is more likely to generate λ around 0.1, thus creating "similar" data as adding noise regularization.

For **TMix**, we only utilize the labeled dataset as the settings in Bert baseline, and set the batch size

[3] https://pypi.org/project/pytorch-transformers/

Dataset	Label Type	Classes	Unlabeled	Dev	Test
AG News	News Topic	4	5000	2000	1900
DBpedia	Wikipeida Topic	14	5000	2000	5000
Yahoo! Answer	QA Topic	10	5000	5000	6000
IMDB	Review Sentiment	2	5000	2000	12500

Table 1: Dataset statistics and dataset split. The number of unlabeled data, dev data and test data in the table means the number of data per class.

Datset	Model	10	200	2500	Dataset	Model	10	200	2500
AG News	VAMPIRE	-	83.9	86.2	DBpedia	VAMPIRE	-	-	-
	BERT	69.5	87.5	90.8		BERT	95.2	98.5	99.0
	TMix*	74.1	88.1	91.0		TMix*	96.8	98.7	99.0
	UDA	84.4	88.3	91.2		UDA	97.8	98.8	99.1
	MixText*	**88.4**	**89.2**	**91.5**		MixText*	**98.5**	**98.9**	**99.2**
Yahoo!	VAMPIRE	-	59.9	70.2	IMDB	VAMPIRE	-	82.2	85.8
	BERT	56.2	69.3	73.2		BERT	67.5	86.9	89.8
	TMix*	58.6	69.8	73.5		TMix*	69.3	87.4	90.3
	UDA	63.2	70.2	73.6		UDA	78.2	89.1	90.8
	MixText*	**67.6**	**71.3**	**74.1**		MixText*	**78.7**	**89.4**	**91.3**

Table 2: Performance (test accuracy(%)) comparison with baselines. The results are averaged after three runs to show the significance (Dror et al., 2018), each run takes around 5 hours. Models are trained with 10, 200, 2500 labeled data per class. VAMPIRE, Bert, and TMix do not use unlabeled data during training while UDA and MixText utilize unlabeled data. * means our models.

as 8. In **MixText**, we utilize both labeled data and unlabeled data for training using the same settings as in UDA. We set $K = 2$, i.e., for each unlabeled data we perform two augmentations, specifically German and Russian. The batch size is 4 for labeled data and 8 for unlabeled data. 0.5 is used as a starting point to tune temperature T. In our experiments, we set 0.3 for AG News, 0.5 for DBpedia and Yahoo! Answer, and 1 for IMDB.

5.4 Results

We evaluated our baselines and proposed methods using accuracy with 5000 unlabeled data and with different amount of labeled data per class ranging from 10 to 10000 (5000 for IMDB).

5.4.1 Varying the Number of Labeled Data

The results on different text classification datasets are shown in Table 2 and Figure 3. All transformer based models (BERT, TMix, UDA and MixText) showed better performance compared to VAMPIRE since larger models were adopted. TMix outperformed BERT, especially when labeled data was limited like 10 per class. For instance, model accuracy improved from 69.5% to 74.1% on AG News with 10 labeled data, demonstrating the effectiveness of TMix. When unlabeled data was introduced in UDA, it outperformed TMix such as from 58.6% to 63.2% on Yahoo! with 10 labeled data, because more data was used and consistency regularization loss was added. Our proposed MixText consistently demonstrated the best performances when compared to different baseline models across four datasets, as MixText not only incorporated unlabeled data and utilized implicit relations between both labeled data and unlabeled data via TMix, but also had better label guessing on unlabeled data through weighted average among augmented and original sentences.

5.4.2 Varying the Number of Unlabeled Data

We also conducted experiments to test our model performances with 10 labeled data and different amount of unlabeled data (from 0 to 10000) on AG News and Yahoo! Answer, shown in Figure 4. With more unlabeled data, the accuracy became much higher on both AG News and Yahoo! Answer, which further validated the effectiveness of the usage of unlabeled data.

5.4.3 Loss on Development Set

To explore whether our methods can avoid overfitting when given limited labeled data, we plotted

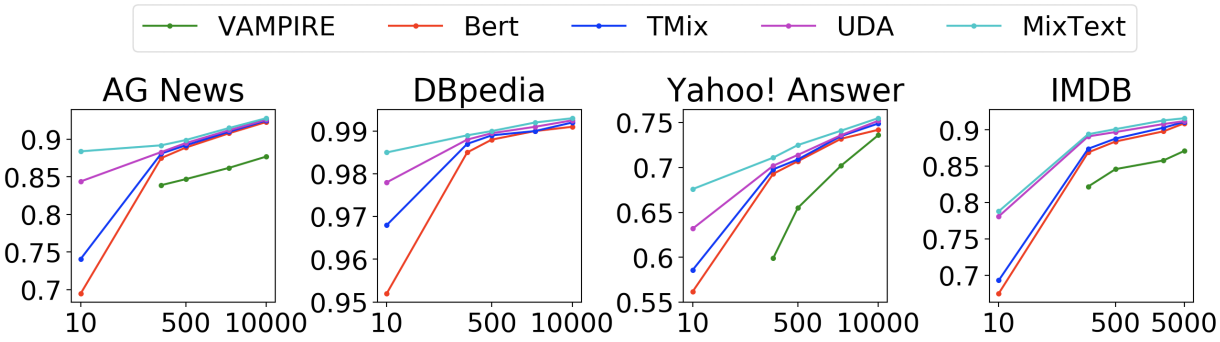

Figure 3: Performance (test accuracy (%)) on AG News, DBpedia, Yahoo! Answer and IMDB with 5000 unlabeled data and varying number of labeled data per class for each model.

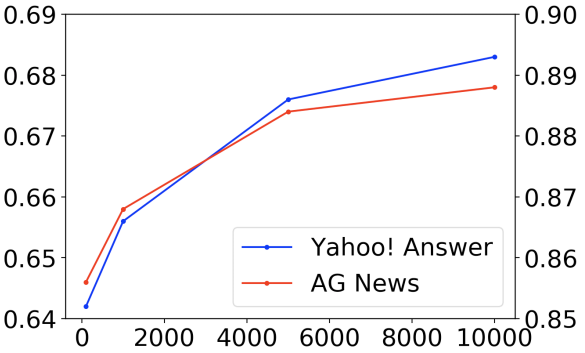

Figure 4: Performance (test accuracy (%)) on AG News (y axis on the right) and Yahoo! Answer (y axis on the left) with 10 labeled data and varying number of unlabeled data per class for MixText.

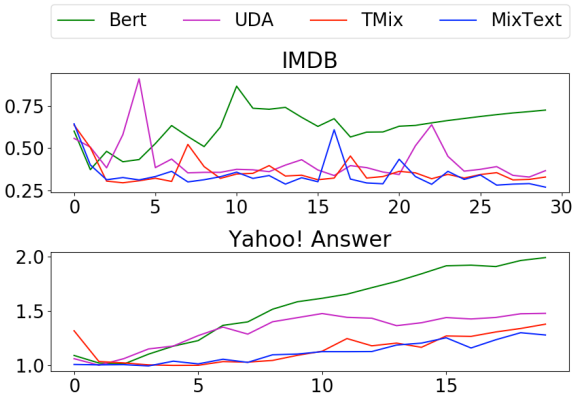

Figure 5: Loss on development set on IMDB and Yahoo! Answer in each epoch while training with 200 labeled data and 5000 unlabeled data per class.

the losses on development set during the training on IMDB and Yahoo! Answer with 200 labeled data per class in Figure 5. We found that the loss on development sets tends to increase a lot in around 10 epochs for Bert, indicating that the model overfitted on training set. Although UDA can alleviate the overfitting problems with consistency regularization, TMix and MixText showed more stable trends and lower loss consistently. The loss curve for TMix also indicated that it can help solving overfitting problems even without extra data.

5.5 Ablation Studies

We performed ablation studies to show the effectiveness of each component in MixText.

5.5.1 Different Mix Layer Set in TMix

We explored different mixup layer set M for TMix and the results are shown in Table 3. Based on (Jawahar et al., 2019), the {3,4,5,6,7,9,12} are the most informative layers in BERT based model and each of them captures different types of information (e.g., surface, syntactic, or semantic). We chose to mixup using different subsets of those layers to see which subsets gave the optimal performance. When no mixup is performed, our model accuracy was 69.5%. If we just mixup at the input and lower layers ({0, 1, 2}), there seemed no performance increase. When doing mixup using different layer sets (e.g., {3,4}, or {6,7,9}), we found large differences in terms of model performances: {3,4} that mainly contains surface information like sentence length does not help text classification a lot, thus showing weaker performance. The 6th layer captures depth of the syntactic tree which also does not help much in classifications. Our model achieved the best performance at {7, 9, 12}; this layer subset contains most of syntactic and semantic information such as the sequence of top level constituents in the syntax tree, the object number in main clause, sensitivity to word order, and the sensitivity to random replacement of a noun/verb.

Mixup Layers Set	Accuracy(%)
$\emptyset$	69.5
{0,1,2}	69.3
{3,4}	70.4
{6,7,9}	71.9
{7,9,12}	**74.1**
{6,7,9,12}	72.2
{3,4,6,7,9,12}	71.6

Table 3: Performance (test accuracy (%)) on AG News with 10 labeled data per class with different mixup layers set for TMix. $\emptyset$ means no mixup.

Model	Accuracy(%)
MixText	**67.6**
- weighted average	67.1
- TMix	63.5
- unlabeled data	58.6
- all	56.2

Table 4: Performance (test accuracy (%)) on Yahoo! Answer with 10 labeled data and 5000 unlabeled data per class after removing different parts of MixText.

5.5.2 Remove Different Parts from MixText

We also measured the performance of MixText by stripping each component each time and displayed the results in Table 4. We observed the performance drops after removing each part, suggesting that all components in MixText contribute to the final performance. The model performance decreased most significantly after removing unlabeled data which is as expected. Comparing to weighted average prediction for unlabeled data, the decrease from removing TMix was larger, indicating that TMix has the largest impact other than unlabeled data, which also proved the effectiveness of our proposed Text Mixup, an interpolation-based regularization and augmentation technique.

6 Conclusion

To alleviate the dependencies of supervised models on labeled data, this work presented a simple but effective semi-supervised learning method, MixText, for text classification, in which we also introduced TMix, an interpolation-based augmentation and regularization technique. Through experiments on four benchmark text classification datasets, we demonstrated the effectiveness of our proposed TMix technique and the Mixup model, which have better testing accuracy and more stable loss trend, compared with current pre-training and fine-tuning models and other state-of-the-art semi-supervised learning methods. For future direction, we plan to explore the effectiveness of MixText in other NLP tasks such as sequential labeling tasks and other real-world scenarios with limited labeled data.

Acknowledgement

We would like to thank the anonymous reviewers for their helpful comments, and Chao Zhang for his early feedback. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research. DY is supported in part by a grant from Google.

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. 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 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. *CoRR*, abs/1905.02249.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21.

Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, pages 830–835. AAAI Press.

Nitesh V. Chawla and Grigoris I. Karakoulas. 2011. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *CoRR*, abs/1109.2047.

Jiaao Chen, Jianshu Chen, and Zhou Yu. 2019. Incorporating structured commonsense knowledge in story completion. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6244–6251.

Jiaao Chen, Yuwei Wu, and Diyi Yang. 2020. Semi-supervised Models via Data Augmentation for Classifying Interactive Affective Responses. In *Workshop On Affective Content Analysis, The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018. Variational sequential labelers for semi-supervised learning. In *Proc. of EMNLP*.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. 2018. Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*.

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.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *CoRR*, abs/1808.09381.

Yves Grandvalet and Yoshua Bengio. 2004. Semi-supervised learning by entropy minimization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 529–536, Cambridge, MA, USA. MIT Press.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A. Smith. 2019. Variational pretraining for semi-supervised text classification. *CoRR*, abs/1906.02242.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha Talukdar. 2019. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. 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 3609–3619, Minneapolis, Minnesota. Association for Computational Linguistics.

Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *CoRR*, abs/1610.02242.

Pooja Lakshmi Narayan, Ajay Nagesh, and Mihai Surdeanu. 2019. Exploration of noise strategies in semi-supervised named entity classification. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 186–191, Minneapolis, Minnesota. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature Cell Biology*, 521(7553):436–444.

Dong-Hyun Lee. 2013. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*.

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.

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 - Volume 1*, HLT '11, pages 142–150, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pablo N. Mendes, Max Jakob, and Christian Bizer. 2012. Dbpedia for nlp: A multilingual cross-domain knowledge base. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 983–992, New York, NY, USA. ACM.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *International Conference on Learning Representations*.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2019. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):1979–1993.

Ajay Nagesh and Mihai Surdeanu. 2018. An exploration of three lightly-supervised representation learning approaches for named entity classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2312–2324.

Luis Perez and Jason Wang. 2017. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *CoRR*, abs/1511.06709.

Antti Tarvainen and Harri Valpola. 2017. Weight-averaged consistency targets improve semi-supervised deep learning results. *CoRR*, abs/1703.01780.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019a. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6438–6447, Long Beach, California, USA. PMLR.

Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. 2019b. Interpolation consistency training for semi-supervised learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3635–3641. International Joint Conferences on Artificial Intelligence Organization.

Vikas Verma, Meng Qu, Alex Lamb, Yoshua Bengio, Juho Kannala, and Jian Tang. 2019c. Graphmix: Regularized training of graph neural networks for semi-supervised learning. *ArXiv*, abs/1909.11715.

Jason W. Wei and Kai Zou. 2019. EDA: easy data augmentation techniques for boosting performance on text classification tasks. *CoRR*, abs/1901.11196.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Diyi Yang, Jiaao Chen, Zichao Yang, Dan Jurafsky, and Eduard Hovy. 2019a. Lets make your request more persuasive: Modeling persuasive strategies via semi-supervised neural nets on crowdfunding platforms. 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 3620–3630.

Diyi Yang, Miaomiao Wen, and Carolyn Rose. 2015. Weakly supervised role identification in teamwork interactions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1671–1680.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. *CoRR*, abs/1702.08139.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. *CoRR*, abs/1905.04899.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.