Calibrated Language Model Fine-Tuning for In- and Out-of-Distribution Data

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, Chao Zhang
Georgia Institute of Technology, Atlanta, USA
{lkkong,jianghm,yczhuang,jie.lyu,tourzhao,chaozhang}@gatech.edu

Abstract

Fine-tuned pre-trained language models can suffer from severe miscalibration for both in-distribution and out-of-distribution (OOD) data due to over-parameterization. To mitigate this issue, we propose a regularized fine-tuning method. Our method introduces two types of regularization for better calibration: (1) *On-manifold regularization*, which generates pseudo on-manifold samples through interpolation within the data manifold. Augmented training with these pseudo samples imposes a smoothness regularization to improve in-distribution calibration. (2) *Off-manifold regularization*, which encourages the model to output uniform distributions for pseudo off-manifold samples to address the over-confidence issue for OOD data. Our experiments demonstrate that the proposed method outperforms existing calibration methods for text classification in terms of expectation calibration error, misclassification detection, and OOD detection on six datasets. Our code can be found at https://github.com/Lingkai-Kong/Calibrated-BERT-Fine-Tuning.

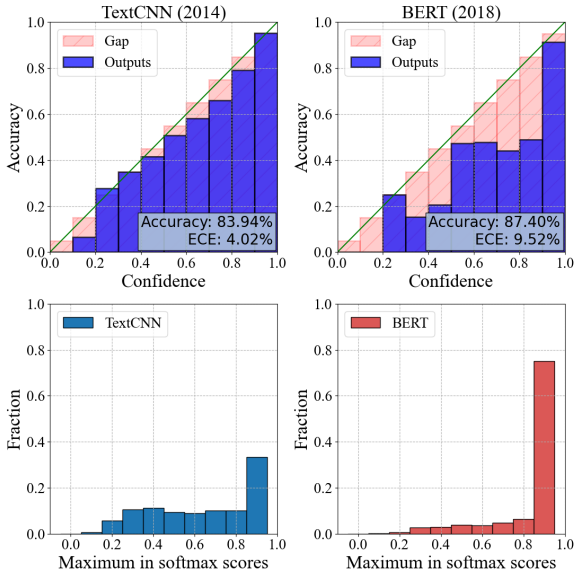

Figure 1: The reliability diagrams on in-distribution data (the first row) and the histograms of the model confidence on out-of-distribution (OOD) data (the second row) of CNN (Kim, 2014) and fine-tuned BERT-MLP classifier (Devlin et al., 2019). Though BERT improves classification accuracy, it makes over-confident predictions for both in-distribution and OOD data.

1 Introduction

Pre-trained language models have recently brought the natural language processing (NLP) community into the transfer learning era. The transfer learning framework consists of two stages, where we first pre-train a large-scale language model, (*e.g.*, BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020) and T5 (Raffel et al., 2019)) on a large text corpus and then fine-tune it on downstream tasks. Such a fine-tuning approach has achieved SOTA performance in many NLP benchmarks (Wang et al., 2018, 2019).

Many applications, however, require trustworthy predictions that need to be not only accurate but also well *calibrated*. In particular, a well-calibrated model should produce reliable confident estimates for both in-distribution and out-of-distribution (OOD) data: (1) For in-distribution data, a model should produce predictive probabilities close to the true likelihood for each class, *i.e.*, confidence $\approx$ true likelihood. (2) For OOD data, which do not belong to any class of the training data, the model output should produce high uncertainty to say 'I don't know', *i.e.*, confidence $\approx$ random guess, instead of producing absurdly wrong yet wildly confident predictions. Providing such calibrated output probabilities can help us to achieve better model robustness (Lee et al., 2018), model fairness (Chouldechova, 2017) and improve label efficiency via uncertainty driven learning (Gal et al., 2017; Siddhant and Lipton, 2018; Shen et al., 2018).

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1326–1340,
November 16–20, 2020. ©2020 Association for Computational Linguistics

Unfortunately, Guo et al. (2017) have shown that due to over-parameterization, deep convolutional neural networks are often miscalibrated. Our experimental investigation further corroborates that fine-tuned language models can suffer from miscalibration even more for NLP tasks. As shown in Figure 1, we present the calibration of a BERT-MLP model for a text classification task on the 20NG dataset. Specifically, we train a TextCNN (Kim, 2014) and a BERT-MLP using $20NG_{15}$ (the first 15 categories of 20NG) and then evaluate them on both in-distribution and OOD data. The first row plots their reliability diagrams (Niculescu-Mizil and Caruana, 2005) on the test set of $20NG_{15}$. Though BERT improves the classification accuracy from 83.9% to 87.4%, it also increases the expected calibration error (ECE, see more details in Section 2) from 4.0% to 9.5%. This indicates that BERT-MLP is much more miscalibrated for in-distribution data. The second row plots the histograms of the model confidence, *i.e.*, the maximum output probability, on the test set of $20NG_5$ (the unseen 5 categories of 20NG). While it is desirable to produce low probabilities for these unseen classes, BERT-MLP produces wrong yet over-confident predictions for such OOD data.

Such an aggravation of miscalibration is due to the even more significant over-parameterization of these language models. At the pre-training stage, they are trained on a huge amount of unlabeled data in an unsupervised manner, *e.g.*, T5 is pre-trained on 745 GB text. To capture rich semantic and syntactic information from such a large corpus, the language models are designed to have enormous capacity, *e.g.*, T5 has about 11 billion parameters. At the fine-tuning stage, however, only limited labeled data are available in the downstream tasks. With the extremely high capacity, these models can easily overfit training data likelihood and be over-confident in their predictions.

To fight against miscalibration, a natural option is to apply a calibration method such as temperature scaling (Guo et al., 2017) in a post-processing step. However, temperature scaling only learns a single parameter to rescale all the logits, which is not flexible and insufficient. Moreover, it cannot improve out-of-distribution calibration. A second option is to mitigate miscalibration during training using regularization. For example, Pereyra et al. (2017) propose an entropy regularizer to prevent over-confidence, but it can needlessly hurt legitimate high confident predictions. A third option is to use Bayesian neural networks (Blundell et al., 2015; Louizos and Welling, 2017), which treat model parameters as probability distributions to represent model uncertainty explicitly. However, these Bayesian approaches are often prohibitive, as the priors of the model parameters are difficult to specify, and exact inference is intractable, which can also lead to unreliable uncertainty estimates.

We propose a regularization approach to addressing miscalibration for fine-tuning pre-trained language models from a data augmentation perspective. We propose two new regularizers using pseudo samples both *on* and *off* the data manifold to mitigate data scarcity and prevent over-confident predictions. Specifically, our method imposes two types of regularization for better calibration during fine-tuning: (1) **On-manifold regularization**: We first generate *on-manifold samples* by interpolating the training data and their corresponding labels along the direction learned from hidden feature space; training over such augmented on-manifold data introduces a smoothness constraint within the data manifold to improve the model calibration for in-distribution data. (2) **Off-manifold regularization**: We generate *off-manifold* samples by adding relatively large perturbations along the directions that point outward the data manifold; we penalize the negative entropy of the output distribution for such off-manifold samples to address the over-confidence issue for OOD data.

We evaluate our proposed model calibration method on six text classification datasets. For in-distribution data, we measure ECE and the performance of misclassification detection. For out-of-distribution data, we measure the performance of OOD detection. Our experiments show that our method outperforms existing state-of-the-art methods in both settings, and meanwhile maintains competitive classification accuracy.

We summarize our contribution as follows: (1) We propose a general calibration framework, which can be applied to pre-trained language model fine-tuning, as well as other deep neural network-based prediction problems. (2) The proposed method adopts on- and off-manifold regularization from a data augmentation perspective to improve calibration for both in-distribution and OOD data. (3) We conduct comprehensive experiments showing that our method outperforms existing calibration methods in terms of ECE, miscalssification detec-

tion and OOD detection on six text classification datasets.

2 Preliminaries

We describe model calibration for both in-distribution and out-of-distribution data.

Calibration for In-distribution Data: For in-distribution data, a well-calibrated model is expected to output prediction confidence comparable to its classification accuracy. For example, given 100 data points with their prediction confidence 0.6, we expect 60 of them to be correctly classified. More precisely, for a data point X, we denote by $Y(X)$ the ground truth label, $\hat{Y}(X)$ the label predicted by the model, and $\hat{P}(X)$ the output probability associated with the predicted label. The calibration error of the predictive model for a given confidence $p \in (0, 1)$ is defined as:

$$\mathcal{E}_p = |\mathbb{P}(\hat{Y}(X) = Y(X)|\hat{P}(X) = p) - p|. \quad (1)$$

As (1) involves population quantities, we usually adopt empirical approximations (Guo, 2017) to estimate the calibration error. Specifically, we partition all data points into M bins of equal size according to their prediction confidences. Let $\mathcal{B}_m$ denote the bin with prediction confidences bounded between ℓ_m and u_m. Then, for any $p \in [\ell_m, u_m)$, we define the empirical calibration error as:

$$\hat{\mathcal{E}}_p = \hat{\mathcal{E}}_m = \frac{1}{|\mathcal{B}_m|}\Big|\sum_{i\in\mathcal{B}_m}\big[\mathbf{1}(\hat{y}_i = y_i) - \hat{p}_i\big]\Big|, \quad (2)$$

where y_i, $\hat{y}_i$ and $\hat{p}_i$ are the true label, predicted label and confidence for sample i.

To evaluate the overall calibration error of the predictive model, we can futher take a weighted average of the calibration errors of all bins, which is also known as the expected calibration error (ECE) (Naeini et al., 2015) defined as:

$$\text{ECE} = \sum_{m=1}^{M}\frac{|\mathcal{B}_m|}{n}\hat{\mathcal{E}}_m, \quad (3)$$

where n is the sample size.

We remark that the goal of calibration is to minimize the calibration error without significantly sacrificing prediction accuracy. Otherwise, a random guess classifier can achieve zero calibration error.

Calibration for Out-of-distribution Data: In real applications, a model can encounter test data that significantly differ from the training data. For example, they come from other unseen classes, or they are potential outliers. A well-calibrated model is expected to produce an output with high uncertainty for such out-of-distribution (OOD) data, formally,

$$P(Y = j) = 1/K \quad \forall j = 1, ..., K,$$

where K is the number of classes of the training data. As such, we can detect OOD data by setting up an uncertainty threshold.

3 Calibrated Fine-Tuning via Manifold Smoothing

We consider N data points of the target task $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $\mathbf{x}_i$'s denote the input embedding of the sentence and y_i's are the associated one-hot labels. Let $f(\cdot)$ denote the feature extraction layers (*e.g.*, BERT); let $g(\cdot)$ denote the task-specific layer; and let θ denote all parameters of f and g. We propose to optimize the following objective at the fine-tuning stage:

$$\min_\theta \mathcal{F}(\theta) = \mathbb{E}_{\mathbf{x},y\sim S}\ell(g\circ f(\mathbf{x}), y) + \lambda_{\text{on}}\mathcal{R}_{\text{on}}(g\circ f) + \lambda_{\text{off}}\mathcal{R}_{\text{off}}(g\circ f), \quad (4)$$

where ℓ is the cross entropy loss, and $\lambda_{\text{on}}, \lambda_{\text{off}}$ are two hyper-parameters. The regularizers $\mathcal{R}_{\text{on}}$ and $\mathcal{R}_{\text{off}}$ are for on- and off-manifold calibration, respectively.

3.1 On-manifold Regularization

The on-manifold regularizer $\mathcal{R}_{\text{on}}$ exploits the interpolation of training data within the data manifold to improve the in-distribution calibration. Specifically, given two training samples $(\mathbf{x}, y)$ and $(\tilde{\mathbf{x}}, \tilde{y})$ and the feature extraction layers f, we generate an on-manifold pseudo sample $(\mathbf{x}', y')$ as follows:

$$\mathbf{x}'^* = \underset{\mathbf{x}'\in\mathbb{B}(\mathbf{x},\delta_{\text{on}})}{\arg\min}\ D_x(f(\mathbf{x}'), f(\tilde{\mathbf{x}})), \quad (5)$$

$$y' = (1 - \delta_y)y + \delta_y\tilde{y}, \quad (6)$$

where δ_{on} and δ_y are small interpolation parameters for data and label, and D_x is a proper distance for features extracted by f such as cosine distance, *i.e.*, $D_x(\mathbf{a}, \mathbf{b}) = \langle \mathbf{a}/\|\mathbf{a}\|_2, \mathbf{b}/\|\mathbf{b}\|_2\rangle$, and $\mathbb{B}(\mathbf{x}, \delta_{\text{on}})$ denotes an ℓ_∞ ball centered at $\mathbf{x}$ with a radius δ_{on}, *i.e.*,

$$\mathbb{B}(\mathbf{x}, \delta_{\text{on}}) = \{\mathbf{x}' \mid \|\mathbf{x}' - \mathbf{x}\|_\infty \le \delta_{\text{on}}\}.$$

As can be seen, $\mathbf{x}'^*$ is essentially interpolating between $\mathbf{x}$ and $\tilde{\mathbf{x}}$ on the data manifold, and $D_x(f(\cdot), f(\cdot))$ can be viewed as a metric over such a manifold. However, as $f(\cdot)$ is learnt from finite training data, it can recover the actual data manifold only up to a certain statistical error. Therefore,

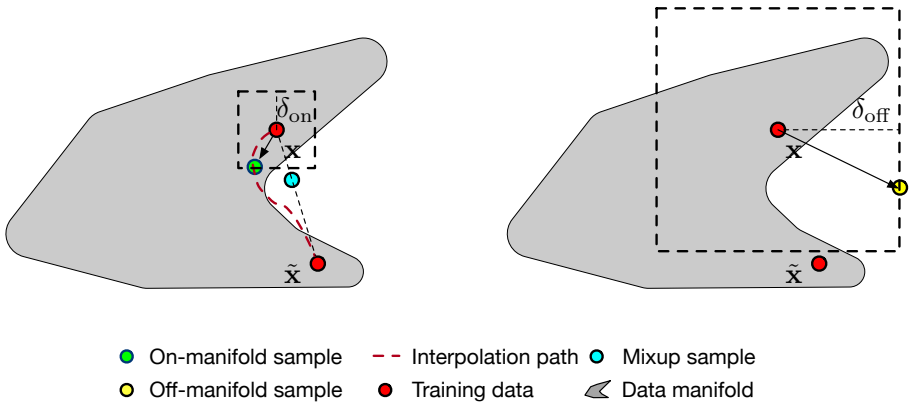

Figure 2: The on-manifold and off-manifold samples generated by our calibration procedure. Mixup adopts a coarse linear interpolation and the generated data point may deviate from the data manifold.

we constrain $\mathbf{x}'^*$ to stay in a small neighborhood of $\mathbf{x}$, which ensures $\mathbf{x}'^*$ to stay close to the actual data manifold.

This is different from existing interpolation methods such as Mixup (Zhang et al., 2018; Verma et al., 2019). These methods adopt coarse linear interpolations either in the input space or latent feature space, and the generated data may significantly deviate from the data manifold.

Note that our method not only interpolates $\mathbf{x}$ but also y. This can yield a soft label for $\mathbf{x}'^*$, when $\mathbf{x}$ and $\tilde{\mathbf{x}}$ belong to different classes. Such an interpolation is analogous to semi-supervised learning, where soft pseudo labels are generated for the unlabelled data. These soft-labelled data essentially induce a smoothing effect, and prevent the model from making overconfident predictions toward one single class.

We remark that our method is more adaptive than the label smoothing method (Müller et al., 2019). As each interpolated data point involves at most two classes, it is unnecessary to distribute probability mass to other classes in the soft label. In contrast, label smoothing is more rigid and enforces all classes to have equally nonzero probability mass in the soft label.

We then define the on-manifold regularizer as

$$\mathcal{R}_{\text{on}}(g \circ f) = \mathbb{E}_{(\mathbf{x}',y')\sim S_{\text{on}}} D_{\text{KL}}(y', g \circ f(\mathbf{x}')),$$

where S_{on} denotes the set of all pseudo labelled data generated by our interpolation method, and D_{KL} denotes the KL-divergence between two probability simplices.

3.2 Off-manifold Regularization

The off-manifold regularizer, $\mathcal{R}_2$, encourages the model to yield low confidence outputs for samples outside the data manifold, and thus mitigates the over-confidence issue for out-of-distribution (OOD) data. Specifically, given a training sample $(\mathbf{x}, y)$, we generate an off-manifold pseudo sample $\mathbf{x}''^*$ by:

$$\mathbf{x}''^* = \max_{\mathbf{x}''\in\mathbb{S}(\mathbf{x},\delta_{\text{off}})} \ell(g \circ f(\mathbf{x}''), y), \tag{7}$$

where $\mathbb{S}(\mathbf{x}, \delta_{\text{off}})$ denotes an ℓ_∞ sphere centered at $\mathbf{x}$ with a radius δ_{off}.

Since we expect $\mathbf{x}''^*$ to mimic OOD data, we first need to choose a relatively large δ_{off} such that the sphere $\mathbb{S}(\mathbf{x}, \delta_{\text{off}})$ can reach outside the data manifold. Then, we generate the pseudo off-manifold sample from the sphere along the adversarial direction. Existing literature (Stutz et al., 2019; Gilmer et al., 2018) has shown that such an adversarial direction points outward the data manifold.

Algorithm 1 Our Proposed Efficient Stochastic Optimization Algorithm for Solving (4). d is the dimension of features.

for # training iterations **do**
  Sample a mini-batch $B = \{\mathbf{x}_i, y_i\}$ from S.
  // Generate on-manifold samples:
  For each $\mathbf{x}_i \in B$, randomly select $\{\tilde{\mathbf{x}}_i, \tilde{y}_i\}$ from B, initialize $\mathbf{x}'_i \leftarrow \mathbf{x}_i + v_i$ with $v_i \sim \text{UNIF}[-\delta_{\text{on}}, \delta_{\text{on}}]^d$
  $\Delta'_i \leftarrow \text{sign}(\nabla_{\mathbf{x}'_i} D_x(f(\mathbf{x}'_i), f(\tilde{\mathbf{x}}_i)))$
  $\mathbf{x}'_i \leftarrow \Pi_{\|\mathbf{x}'_i - \mathbf{x}_i\|_\infty \le \delta_{\text{on}}}(\mathbf{x}'_i - \delta_{\text{on}}\Delta'_i)$
  $y' \leftarrow (1-\delta_y)y_i + \delta_y \tilde{y}_i$
  // Generate off-manifold samples:
  For each $\mathbf{x}_i \in B$, initialize $\mathbf{x}''_i \leftarrow \mathbf{x}_i + v'_i$ with $v'_i \sim \text{UNIF}[-\delta_{\text{off}}, \delta_{\text{off}}]^d$
  $\Delta''_i \leftarrow \text{sign}(\nabla_{\mathbf{x}''_i} \ell(g \circ f(\mathbf{x}''_i), y)$
  $\mathbf{x}''_i \leftarrow \Pi_{\|\mathbf{x}''_i - \mathbf{x}_i\|_\infty = \delta_{\text{off}}}(\mathbf{x}''_i + \delta_{\text{off}}\Delta''_i)$
  Update θ using ADAM
end for

By penalizing the prediction confidence for these off-manifold samples, we are able to encourage low prediction confidence for OOD data. Hence, we define the off-manifold regularizer as

$$\mathcal{R}_{\text{off}}(g \circ f) = \mathbb{E}_{\mathbf{x}'' \sim S_{\text{off}}} - \mathcal{H}(g \circ f(\mathbf{x}'')), \quad (8)$$

where S_{off} denotes the set of all generated off-manifold samples, and $\mathcal{H}(\cdot)$ denotes the entropy of the probability simplex.

3.3 Model Training

We can adopt stochastic gradient-type algorithms such as ADAM (Kingma and Ba, 2014) to optimize (4). At each iteration, we need to first solve two inner optimization problems in (5) and (7), and then plug $\mathbf{x}'$ and $\mathbf{x}''$ into (4) to compute the stochastic gradient. The two inner problems can be solved using the projected sign gradient update for multiple steps. In practice, we observe that one single update step with random initialization is already sufficient to efficiently optimize θ. Such a phenomenon has also been observed in existing literature on adversarial training (Wong et al., 2019). We summarize the overall training procedure in Algorithm 1.

4 Experiments

To evaluate calibration performance for in-distribution data, we measure the expected calibration error (ECE) and the misclassification detection score. For out-of-distribution data, we measure the OOD detection score.

We detect the misclassified and OOD samples by model confidence, which is the output probability associated with the predicted label $\hat{P}(X)$. Specifically, we setup a confidence threshold $\tau \in [0, 1]$, and take the samples with confidence below the threshold, *i.e.*, $\hat{P}(X) < \tau$, as the misclassified or OOD samples. We can compute the detection F_1 score for every τ: $F_1(\tau)$, and obtain a calibration curve ($F_1(\tau)$ vs. τ). Then, we set τ_{upper} as the upper bound of the confidence threshold, since a well calibrated model should provide probabilities that reflect the true likelihood and it is not reasonable to use a large τ to detect them. We use the empirical Normalized Bounded Area Under the Calibration Curve (NBAUCC) as the overall detection score:

$$\text{NBAUCC}_{\tau_{\text{upper}}} = \frac{1}{M} \sum_{i=1}^{M} F_1 \left(\frac{\tau_{\text{upper}}}{M} i \right),$$

where M is the number of sub-intervals for the numerical integration. We set $M = 50$ throughout the following experiments. Note that the traditional binary classification metrics, *e.g.*, AUROC and AUPR, cannot measure the true calibration because the model can still achieve high scores even though it has high confidences for the misclassified and OOD samples. We provide more explanations of the metrics in Appendix C. We report the performance when $\tau_{\text{upper}} = 0.5$ here and the results when $\tau_{\text{upper}} = 0.7$ and 1 in Appendix D.

4.1 Datasets

For each dataset, we construct an in-distribution training set, an in-distribution testing set, and an OOD testing set. Specifically, we use the following datasets:

20NG[1]. The 20 Newsgroups dataset (20NG) contains news articles with 20 categories. We use Stanford Sentiment Treebank (SST-2) (Socher et al., 2012) as the OOD data.

20NG$_{15}$. We take the first 15 categories of 20NG as the in-distribution data and the other 5 categories (20NG$_5$) as the OOD data.

WOS (Kowsari et al., 2017). Web of Science (WOS) dataset contains scientific articles with 134 categories. We use AGnews (Zhang et al., 2015) as the OOD data.

WOS$_{100}$. We use the first 100 classes of WOS as the in-distribution data and the other 34 classes (WOS$_{34}$) as the OOD data.

Yahoo (Chang et al., 2008). This dataset contains questions with 10 categories posted to 'Yahoo! Answers'. We randomly draw 2000 from $140,000$ samples for each category as the training set. We use Yelp (Zhang et al., 2015) as the OOD data.

Yahoo$_8$. We use the first 8 classes of Yahoo as the in-distribution data and the other 2 classes (Yahoo$_2$) as the OOD data.

The testing set of OOD detection consists of the in-distribution testing set and the OOD data. More dataset details can be found in Appendix A. We remark that 20NG$_{15}$, WOS$_{100}$, and Yahoo$_8$ are included to make OOD detection more challenging, as the OOD data and the training data come from similar data sources.

4.2 Baselines

We consider the following baselines:

• **BERT** (Devlin et al., 2019) is a pre-trained base BERT model stacked with one linear layer.

[1]We use the 20 Newsgroups dataset from: http://qwone.com/~jason/20Newsgroups/

• **Temperature Scaling (TS)** (Guo, 2017) is a post-processing calibration method that learns a single parameter to rescale the logits on the development set after the model is fine-tuned.
• **Monte Carlo Dropout (MCDP)** (Gal and Ghahramani, 2016) applies dropout at testing time for multiple times and then averages the outputs.
• **Label Smoothing (LS)** (Müller et al., 2019) smoothes the one-hot label by distributing a certain probability mass to other non ground-truth classes.
• **Entropy Regularized Loss (ERL)** (Pereyra et al., 2017) adds a entropy penalty term to prevent DNNs from being over-confident.
• **Virtual Adversarial Training (VAT)** (Miyato et al., 2018) introduces a smoothness-inducing adversarial regularizer to encourage the local Lipschitz continuity of DNNs.
• **Mixup** (Zhang et al., 2018; Thulasidasan et al., 2019) augments training data by linearly interpolating training samples in the input space.
• **Manifold-mixup (M-mixup)** (Verma et al., 2019) is an extension of Mixup, which interpolates training samples in the hidden feature space.

4.3 Implementation Details

We use ADAM (Kingma and Ba, 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as the optimizer. For our method, we simply set $\lambda_{\text{on}} = \lambda_{\text{off}} = 1, \delta_{\text{on}} = 10^{-4}, \delta_{\text{off}} = 10^{-3}$, and $\delta_y = 0.1$ for all the experiments. We also conduct an extensive hyper-parameter search for the baselines. See more details in Appendix B.

4.4 Main Results

Our main results are summarized as follows:
Expected Calibration Error: Table 1 reports the ECE and predictive accuracy of all the methods. Our method outperforms all the baselines on all the datasets in terms of ECE except for Yahoo, where only ERL is slightly better. Meanwhile, our method does not sacrifice the predictive accuracy.
Misclassification Detection: Table 2 compares the $\text{NBAUCC}_{0.5}$ on misclassification detection of different methods. As shown, our method outperforms all the baselines on all the six datasets.
Out-of-distribution Detection: Table 2 reports the $\text{NBAUCC}_{0.5}$ on OOD detection of different methods. Again, our method achieves the best performance on all the six datasets. The improvement is particularly remarkable on the 20NG dataset, where $\text{NBAUCC}_{0.5}$ increases from 47.00 to 63.92 compared with the strongest baseline. We also find

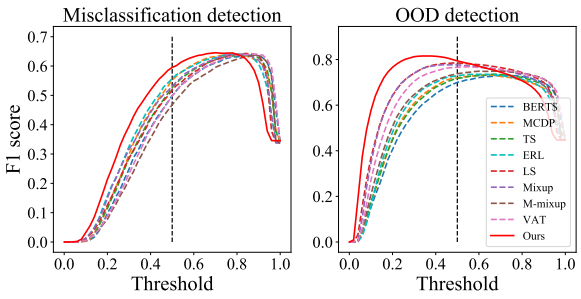

Figure 3: Calibration curves of OOD detection and misclassification detection on WOS. Our method can achieve high F_1 scores starting from a small threshold which indicates that it indeed provides low confidences for misclassified and OOD samples; the F_1 scores of the baselines peak at high thresholds which indicates that they are poorly calibrated.

that detecting the unseen classes from the original dataset is much more challenging than detecting OOD samples from a totally different dataset.
Significance Test: We perform the Wilcoxon signed rank test (Wilcoxon, 1992) for significance test. For each dataset, we conduct experiments using 5 different random seeds with significance level $\alpha = 0.5$. We find that our model outperforms other baselines on all the datasets significantly, with only exceptions of ERL in ECE on Yahoo and ERL in misclassification detection on 20NG.

4.5 Parameter Study

We investigate the effects of the interpolation parameters for on-manifold data, *i.e.*, δ_{on} and δ_y, and the perturbation size for off-manifold samples, *i.e.*, δ_{off}. The default values are $\delta_{\text{on}} = 10^{-4}, \delta_{\text{off}} = 10^{-3}$ and $\delta_y = 0.1$. Figure 4 shows the reuslts on 20NG_{15}, 20NG, WOS_{100}, and WOS datasets. Our results are summarized as follows:
• The performance of all metrics versus δ_{on} is stable within a large range from 10^{-5} to 10^{-2}. When δ_{on} is larger than 10^{-1}, the predictive accuracy begins to drop.
• The performance versus δ_{off} is more sensitive: (1) when δ_{off} is too small, ECE increases dramatically becasue the generated off-manifold samples are too close to the manifold and make the model under-confident. (2) when δ_{off} is too large, the off-manifold regularization is too weak and OOD detection performance drops.
• In general, δ_{on} should be small to let $\mathbf{x}'$ stay on the data manifold while δ_{off} should be large to let $\mathbf{x}''$ leave the data manifold. However, the regularization effect of $\mathcal{R}_{\text{on}}$ ($\mathcal{R}_{\text{off}}$) depends on both λ_{on}

Model	ECE						Accuracy					
	$20NG_{15}$	20NG	WOS_{100}	WOS	$Yahoo_8$	Yahoo	$20NG_{15}$	20NG	WOS_{100}	WOS	$Yahoo_8$	Yahoo
BERT	9.24	11.61	6.81	6.74	10.11	10.54	87.42	84.55	81.94	79.40	73.58	71.89
TS	4.42	8.17	3.63	4.43	5.18	4.24	87.42	84.55	81.94	79.40	73.58	71.89
MCDP	6.88	9.17	4.00	3.55	6.54	6.72	87.45	84.55	82.09	79.67	73.67	71.99
LS	4.35	6.15	4.35	4.67	4.89	3.61	87.54	85.02	81.95	79.47	73.66	71.54
ERL	7.16	6.10	3.74	3.35	3.42	**2.96**	87.67	84.83	81.96	79.48	73.63	72.01
VAT	9.07	11.28	7.27	6.76	10.96	7.92	87.61	85.20	81.65	79.71	73.71	72.08
Mixup	5.98	9.02	4.72	4.21	4.60	5.18	87.49	84.86	81.97	79.51	73.88	71.82
M-mixup	5.04	7.78	6.48	6.68	7.01	6.07	87.40	84.45	81.77	79.57	73.67	72.03
Ours	**3.69**	**4.43**	**3.24**	**3.04**	**3.03**	3.42	87.44	84.53	81.59	79.06	73.71	72.17

Table 1: ECE and accuracy (in percentage). We report the average performance of 5 random initializations.

Data (OOD)	Misclassification Detection						OOD Detection					
	$20NG_{15}$	20NG	WOS_{100}	WOS	$Yahoo_8$	Yahoo	$20NG_{15}$ $20NG_5$	20NG SST-2	WOS_{100} WOS_{34}	WOS AGnews	$Yahoo_8$ $Yahoo_2$	Yahoo Yelp
BERT	2.30	2.86	16.53	20.52	7.47	8.43	2.66	21.65	23.12	49.84	8.35	13.88
TS	6.08	5.74	21.20	23.76	10.48	12.74	6.62	32.64	28.12	53.32	11.55	20.27
MCDP	4.37	5.28	20.44	24.16	10.12	10.75	3.99	25.10	27.28	53.52	9.98	15.93
LS	4.72	6.75	20.37	23.56	11.19	16.15	5.70	41.08	27.12	58.48	12.02	19.81
ERL	8.54	10.35	20.49	25.13	12.89	15.47	8.78	47.00	27.73	56.67	13.78	23.47
VAT	2.52	3.36	18.70	19.96	6.54	10.37	2.96	29.62	23.41	54.60	7.42	17.65
Mixup	4.99	4.51	20.65	24.80	10.75	11.29	5.86	31.84	26.77	58.02	11.62	19.84
M-mixup	2.16	3.16	16.94	19.39	9.09	11.79	2.36	26.08	24.08	51.39	10.08	22.41
Ours	**9.10**	**10.76**	**26.93**	**30.80**	**14.34**	**17.88**	**9.69**	**63.92**	**35.60**	**71.13**	**14.94**	**29.40**

Table 2: $\mathrm{NBAUCC}_{0.5}$ on misclassification detection and OOD detection. We report the average performance of 5 random initializations.

(λ_{off}) and δ_{on} (δ_{off}). Therefore, it is not necessary to let δ_{on} be smaller than δ_{off}. We can also tune λ_{on} and λ_{off} to achieve better performance.

• The performance versus δ_y is relatively stable except for the metric of ECE. When δ_y is larger than 0.2, ECE begins to increase.

4.6 Ablation Study

We investigate the effectiveness of the on-manifold regularizer $\mathcal{R}_{\text{on}}$ and the off-manifold regularizer $\mathcal{R}_{\text{off}}$ via ablation studies. Table 3 shows the results on the $20NG_{15}$ and 20NG datasets.

• As expected, removing either component in our method would result in a performance drop. It demonstrates that these two components complement each other. All the ablation models outperform the BERT baseline model, which demonstrates the effectiveness of each module.

• We observe that the optimal δ_{on} is different when using only $\mathcal{R}_{\text{on}}$. This indicates that the hyperparameters of $\mathcal{R}_{\text{on}}$ and $\mathcal{R}_{\text{off}}$ should be jointly tuned, due to the joint effect of both components.

• By removing $\mathcal{R}_{\text{off}}$, we observe a severe OOD performance degradation on the 20NG dataset (from 63.92 to 43.87). This indicates that $\mathcal{R}_{\text{off}}$ is vital to out-of-distribution calibration. Meanwhile, the performance degradation is less severe on $20NG_{15}$ (from 9.69 to 7.94). It is because $\mathcal{R}_{\text{on}}$ can also help detect the OOD samples from similar data sources. ($20NG_5$).

• By removing $\mathcal{R}_{\text{on}}$, the in-distribution calibration performance drops as expected.

5 Related Works and Discussion

Other Related Works: Lakshminarayanan et al. (2017) propose a model ensembling approach to improve model calibration. They first train multiple models with different initializations and then average their predictions. However, fine-tuning multiple language models requires extremely intensive computing resources.

Kumar et al. (2018) propose a differentiable surrogate for the expected calibration error, called

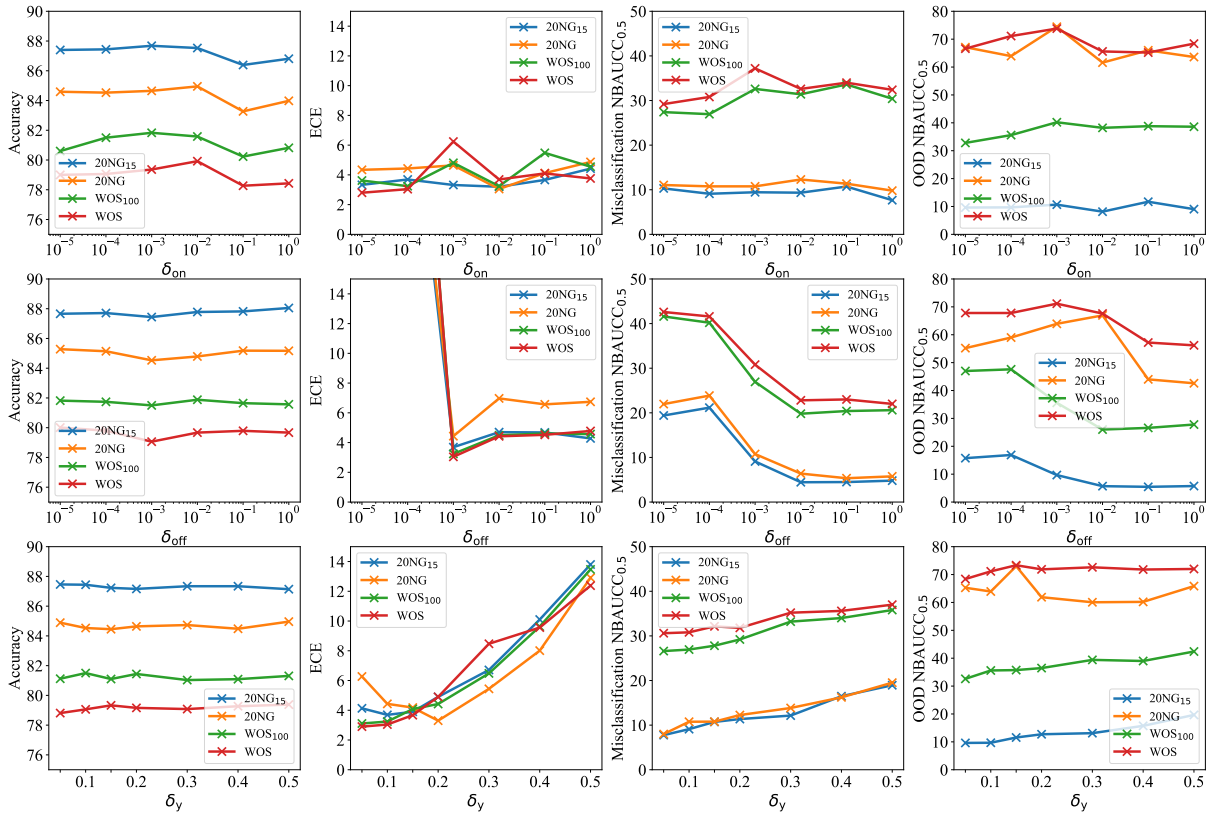

Figure 4: Parameter study of δ_{on}, δ_{off} and δ_y.

Dataset		20NG$_{15}$				20NG			
Model	δ_{on}	Accuracy	ECE	OOD	Mis	Accuracy	ECE	OOD	Mis
BERT	-	87.42	9.24	2.66	2.30	84.55	11.61	21.65	2.86
w/ $\mathcal{R}_{\text{off}}$	-	86.48	6.51	6.22	6.09	83.90	7.98	55.40	7.12
w/ $\mathcal{R}_{\text{on}}$	10^{-2}	88.73	2.77	7.94	8.08	85.60	5.00	35.80	8.66
w/ $\mathcal{R}_{\text{on}}$	10^{-3}	88.29	3.52	7.39	6.83	85.69	4.43	38.00	9.01
w/ $\mathcal{R}_{\text{on}}$	10^{-4}	87.93	4.48	5.33	4.83	85.12	6.76	43.87	5.95
w/ $\mathcal{R}_{\text{on}}$	10^{-5}	87.61	4.69	3.83	4.73	85.39	6.35	35.70	5.30
w/ Both	10^{-4}	87.44	3.69	9.69	9.10	84.53	4.43	63.92	10.76

Table 3: Ablation study on the 20NG$_{15}$ and 20NG datasets. For OOD detection and misclassification detection, we report BAUCC$_{0.5}$. We set $\delta_y = 0.1$ and $\delta_{\text{off}} = 10^{-3}$.

maximum mean calibration error (MMCE), using kernel embedding. However, such a kernel embedding method is computationally expensive and not scalable to the large pre-trained language models.

Accelerating Optimization: To further improve the calibration performance of our method, we can leverage some recent minimax optimization techniques to better solve the two inner optimization problems in (5) and (7) without increasing the computational complexity. For example, Zhang et al. (2019) propose an efficient approximation algorithm based on Pontryagin's Maximal Principle to replace the multi-step projected gradient update for the inner optimization problem. Another option is the learning-to-learn framework (Jiang et al., 2018), where the inner problem is solved by a learnt optimizer. These techniques can help us obtain $\mathbf{x}'$ and $\mathbf{x}''$ more efficiently.

Connection to Robustness: The interpolated training samples can naturally promote the local Lipschitz continuity of our model. Such a local smoothness property has several advantages: (1) It makes the model more robust to the inherent noise in the data, *e.g.*, noisy labels; (2) it is particularly helpful to prevent overfitting and improve generalization, especially for low-resource tasks.

Extensions: Our method is quite general and can be applied to other deep neural network-based problems besides language model fine-tuning.

6 Conclusion

We have proposed a regularization method to mitigate miscalibration of fine-tuned language models from a data augmentation perspective. Our method imposes two new regularizers using generated on- and off- manifold samples to improve both in-distribution and out-of-distribution calibration. Extensive experiments on six datasets demonstrate that our method outperforms state-of-the-art calibration methods in terms of expected calibration error, misclassification detection and OOD detection.

Acknowledgement

This work was supported in part by the National Science Foundation award III-2008334, Amazon Faculty Award, and Google Faculty Award.

References

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622.

Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, page 830–835.

Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

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.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, Ian Goodfellow, and G Brain. 2018. The relationship between high-dimensional geometry and adversarial examples. *arXiv preprint arXiv:1801.02774*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330.

Hongyu Guo. 2017. A deep network with visual text composition behavior. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 372–377, Vancouver, Canada. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*.

Haoming Jiang, Zhehui Chen, Yuyang Shi, Bo Dai, and Tuo Zhao. 2018. Learning to defense by learning to attack. *arXiv preprint arXiv:1811.01213*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

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

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, , Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 364–371.

Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. 2018. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning

of language representations. In *International Conference on Learning Representations*.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177.

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*.

Christos Louizos and Max Welling. 2017. Multiplicative normalizing flows for variational Bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227.

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

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems*, pages 4696–4705.

Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, page 2901–2907.

Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *International Conference on Machine Learning*, page 625–632.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *International Conference on Learning Representations*.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium. Association for Computational Linguistics.

Richard Socher, Yoshua Bengio, and Christopher D. Manning. 2012. Deep learning for NLP (without magic). In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, page 5, Jeju Island, Korea. Association for Computational Linguistics.

David Stutz, Matthias Hein, and Bernt Schiele. 2019. Disentangling adversarial robustness and generalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6976–6987.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. 2019. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 13888–13899.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer.

Eric Wong, Leslie Rice, and J Zico Kolter. 2019. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems*, pages 227–238.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

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 Dataset Details

	#Train	#Dev	#Test	#Label
$20NG_{15}$	7010	1753	5833	15
$20NG_5$	-	-	1699	5
20NG	9051	2263	7532	20
SST-2	-	-	1822	2
WOS_{100}	16794	4191	13970	100
WOS_{34}	-	-	4824	34
WOS	22552	5639	18794	134
AGnews	-	-	7600	4
$Yahoo_8$	16000	4000	48000	8
$Yahoo_2$	-	-	12000	2
Yahoo	20000	5000	60000	10
Yelp	-	-	38000	2

Table 4: Dataset statistics and dataset split. '-' denotes that this part is not used. The original Yahoo dataset contains $140,000$ training samples for each class which is too large; we randomly draw $2,000$ and 500 samples for each class as our training and development set.

All the data are publicly available. We also offer the links to the data as follows:

1. 20NG: http://qwone.com/~jason/20Newsgroups/.
2. SST-2: https://nlp.stanford.edu/sentiment/index.html.
3. WOS: https://data.mendeley.com/datasets/9rw3vkcfy4/2.
4. AGnews: https://github.com/yumeng5/WeSTClass.
5. Yahoo: https://www.kaggle.com/soumikrakshit/yahoo-answers-dataset.
6. Yelp: https://github.com/yumeng5/WeSTClass.

B Experiment Details

We use ADAM (Kingma and Ba, 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as the optimizer in all the datasets. We use the learning rate of 5×10^{-5} and batch size 32 except 1×10^{-5} and 16 for $Yahoo_8$ and Yahoo. We set the maximum number of epochs to 5 in $Yahoo_8$ and Yahoo and 10 in the other datasets. We use the dropout rate of 0.1 as in (Devlin et al., 2019). The documents are tokenized using wordpieces and are chopped to spans no longer than 150 tokens on $20NG_{15}$ and 20NG and 256 on other datasets..

Hyper-parameters: For our method, we use $\lambda_{\text{on}} = \lambda_{\text{off}} = 1$, $\delta_{\text{on}} = 10^{-4}$, $\delta_{\text{off}} = 10^{-3}$ and $\delta_y = 0.1$ for all the datasets. We then conduct an extensive hyper-parameter search for the baselines: for label smoothing, we search the smoothing parameter from $\{0.05, 0.1\}$ as in (Müller et al., 2019); for ERL, the penalty weight is chosen from $\{0.05, 0.1, 0.25, 0.5, 1, 2.5, 5\}$; for VAT, we search the perturbation size in $\{10^{-3}, 10^{-4}, 10^{-5}\}$ as in (Jiang et al., 2020); for Mixup, we search the interpolation parameter from $\{0.1, 0.2, 0.3, 0.4\}$ as suggested in (Zhang et al., 2018; Thulasidasan et al., 2019); for Manifold-mixup, we search from $\{0.2, 0.4, 1, 2, 4\}$. We perform 10 stochastic forward passes for MCDP at test time. For hyper-parameter tuning, we run all the methods 5 times and then take the average. The hyper-parameters are selected to get the best ECE on the development set of each dataset. The interpolation of Mixup is performed on the input embeddings obtained from the first layer of the language model; the interpolation of Manifold-mixup is performed on the features obtained from the last layer of the language model.

C Metrics of Misclassification and Out-of-distribution detection

Existing works on out-of-distribution (OOD) detection and misclassification detection (Hendrycks and Gimpel, 2016) use traditional binary classification metrics, *e.g.*, AUPR and AUROC. As we discussed in Section 1 and 2, the output probability of a calibrated model should reflect the true likelihood. However, AUROC and AUPR cannot reflect true model calibration because the model can still achieve high scores even though it has high confidences for misclassified and OOD samples. We argue that it is more reasonable to use the Normalized Bounded Area Under the Calibration Curve (NBAUCC) defined as in Section 4.

Table 5 shows an illustrative example. As can be seen, h_1 is better calibrated than h_2, since h_1 can detect OOD samples under a wide range of threshold ($0.15 < \tau < 0.9$) while h_2 requires an absurdly large threshold ($0.85 < \tau < 0.9$). However, if we use the traditional AUPR and AUROC metrics, we will conclude that h_1 is as well calibrated as h_2 since $\text{AUPR}^{h_1} = \text{AUPR}^{h_2} = 0.417$ and $\text{AUROC}^{h_1} = \text{AUROC}^{h_2} = 1$. On the

Model	Confidence				Optimal τ	AUPR	AUROC	NBAUCC_1	$\mathrm{NBAUCC}_{0.5}$
	$x_{\mathrm{in},1}$	$x_{\mathrm{in},2}$	$x_{\mathrm{out},1}$	$x_{\mathrm{out},2}$					
h_1 (Miscalibrated)	0.9	0.95	0.8	0.85	$(0.85, 0.9)$	0.417	1	0.145	0
h_2 (Well-calibraterd)	0.9	0.95	0.1	0.15	$(0.15, 0.9)$	0.417	1	0.845	0.773

Table 5: NBAUCC vs. AUROC/AUPR

other hand, if we use NBAUCC, we will have $\mathrm{NBAUCC}_1^{h_1} = 0.845 > \mathrm{NBAUCC}_1^{h_1} = 0.145$, or $\mathrm{NBAUCC}_{0.5}^{h_1} = 0.773 > \mathrm{NBAUCC}_{0.5}^{h_1} = 0$ which can reflect the true calibration of the two classifiers.

We remark that it is more appropriate to use $\mathrm{NBAUCC}_{0.5}$ than NBAUCC_1 since a calibrated model should provide low confidences for the misclassified and OOD samples and it is unreasonable to use a large threshold to detect them.

D Additional Results

Table 6 and 7 report the NBAUCCs of all the methods on misclassification and OOD detection when $\tau_{\mathrm{upper}} = 0.7$ and $\tau_{\mathrm{upper}} = 1$. Table 8 and 9 report the ablation study results of all the methods when $\tau_{\mathrm{upper}} = 0.7$ and $\tau_{\mathrm{upper}} = 1$. Figure 5 and 6 report the parameter study results of all the methods when $\tau_{\mathrm{upper}} = 0.7$ and $\tau_{\mathrm{upper}} = 1$.

Data (OOD)	Misclassification Detection						OOD Detection					
	$20NG_{15}$	20NG	WOS_{100}	WOS	$Yahoo_8$	Yahoo	$20NG_{15}$ $20NG_5$	20NG SST-2	WOS_{100} WOS_{34}	WOS AGnews	$Yahoo_8$ $Yahoo_2$	Yahoo Yelp
BERT	17.86	18.48	35.84	39.08	28.83	29.67	13.52	42.86	40.04	59.42	26.63	38.30
TS	23.74	23.58	38.34	40.76	31.10	32.63	19.74	50.00	42.96	60.70	28.30	42.07
MCDP	23.58	24.58	38.54	41.20	31.43	32.57	16.82	44.96	42.74	60.72	27.47	39.83
LS	21.22	23.24	37.22	40.12	30.93	34.30	18.76	55.24	42.54	63.62	27.87	40.77
ERL	24.04	25.68	37.87	41.17	32.27	33.90	22.10	54.20	42.67	62.10	28.73	43.37
VAT	17.80	17.50	35.90	38.80	27.87	31.13	13.00	49.00	40.30	62.50	25.80	40.63
Mixup	21.42	21.86	37.72	40.92	30.97	32.97	16.70	50.94	42.13	62.98	28.00	44.57
M-mixup	17.86	19.24	36.48	38.33	29.67	31.50	14.06	44.56	41.51	61.30	27.43	44.20
Ours	**26.50**	**28.10**	**40.93**	**43.70**	**33.07**	**35.13**	**23.20**	**66.36**	**46.73**	**68.10**	**29.70**	**46.43**

Table 6: NBAUCC_1 on misclassification detection and OOD detection. We report the average performance of 5 random initializations.

Data (OOD)	Misclassification Detection						OOD Detection					
	$20NG_{15}$	20NG	WOS_{100}	WOS	$Yahoo_8$	Yahoo	$20NG_{15}$ $20NG_5$	20NG SST-2	WOS_{100} WOS_{34}	WOS AGnews	$Yahoo_8$ $Yahoo_2$	Yahoo Yelp
BERT	8.26	8.70	26.95	31.18	18.52	19.46	7.05	33.24	32.97	57.45	18.86	27.68
TS	14.60	13.72	31.73	33.89	22.32	24.61	12.91	43.55	37.84	59.86	22.17	34.03
MCDP	13.14	14.21	31.05	34.74	21.41	22.62	9.85	36.96	36.97	60.06	19.99	29.45
LS	12.45	14.24	30.92	33.51	22.94	27.52	11.63	49.60	36.04	65.28	22.38	33.00
ERL	17.92	20.04	30.83	35.26	25.07	27.34	15.43	55.69	36.69	61.93	24.07	36.74
VAT	8.44	9.66	29.39	30.57	17.23	21.74	7.26	41.35	32.56	60.81	17.64	31.17
Mixup	13.33	11.87	31.71	35.24	22.62	22.80	11.50	43.60	37.09	65.51	22.19	33.66
M-mixup	8.67	9.89	27.33	29.61	20.33	23.05	7.18	37.10	33.57	58.13	20.66	36.42
Ours	**18.35**	**20.18**	**36.63**	**40.01**	**25.94**	**29.15**	**16.55**	**68.72**	**43.40**	**72.62**	**25.03**	**41.11**

Table 7: $\mathrm{NBAUCC}_{0.7}$ on misclassification detection and OOD detection. We report the average performance of 5 random initializations.

Dataset		$20NG_{15}$				20NG			
Model	δ_{on}	Accuracy	ECE	OOD	Mis	Accuracy	ECE	OOD	Mis
BERT	-	87.42	9.24	13.52	17.86	84.55	11.61	42.86	18.48
w/ $\mathcal{R}_{\text{off}}$	-	86.48	6.51	18.10	24.53	83.90	7.98	63.73	25.40
w/ $\mathcal{R}_{\text{on}}$	10^{-2}	88.73	2.77	22.83	27.40	85.60	5.00	51.53	27.40
w/ $\mathcal{R}_{\text{on}}$	10^{-3}	88.29	3.52	21.03	24.13	85.69	4.43	53.87	26.30
w/ $\mathcal{R}_{\text{on}}$	10^{-4}	87.93	4.48	17.43	21.63	85.12	6.76	57.47	21.93
w/ $\mathcal{R}_{\text{on}}$	10^{-5}	87.61	4.69	15.73	21.43	85.39	6.35	52.07	21.63
w/ Both	10^{-4}	87.44	3.69	23.20	26.50	84.53	4.43	66.36	28.10

Table 8: Ablation study on the $20NG_{15}$ and 20NG datasets. For OOD detection and misclassification detection, we report NBAUCC_1. We set $\delta_y = 0.1$ and $\delta_{\text{off}} = 10^{-3}$.

Dataset		$20NG_{15}$				20NG			
Model	δ_{on}	Accuracy	ECE	OOD	Mis	Accuracy	ECE	OOD	Mis
BERT	-	87.42	9.24	7.05	8.26	84.55	11.61	33.24	8.70
w/ $\mathcal{R}_{off}$	-	86.48	6.51	11.75	14.79	83.90	7.98	62.67	15.42
w/ $\mathcal{R}_{on}$	10^{-2}	88.73	2.77	15.27	18.35	85.60	5.00	46.67	18.39
w/ $\mathcal{R}_{on}$	10^{-3}	88.29	3.52	13.86	15.66	85.69	4.43	50.07	18.17
w/ $\mathcal{R}_{on}$	10^{-4}	87.93	4.48	10.61	12.59	85.12	6.76	53.64	13.18
w/ $\mathcal{R}_{on}$	10^{-5}	87.61	4.69	8.71	12.25	85.39	6.35	46.24	12.20
w/ Both	10^{-4}	87.44	3.69	16.55	18.35	84.53	4.43	68.72	20.18

Table 9: Ablation study on the $20NG_{15}$ and 20NG datasets. For OOD detection and misclassification detection, we report $NBAUCC_{0.7}$. We set $\delta_y = 0.1$ and $\delta_{off} = 10^{-3}$.

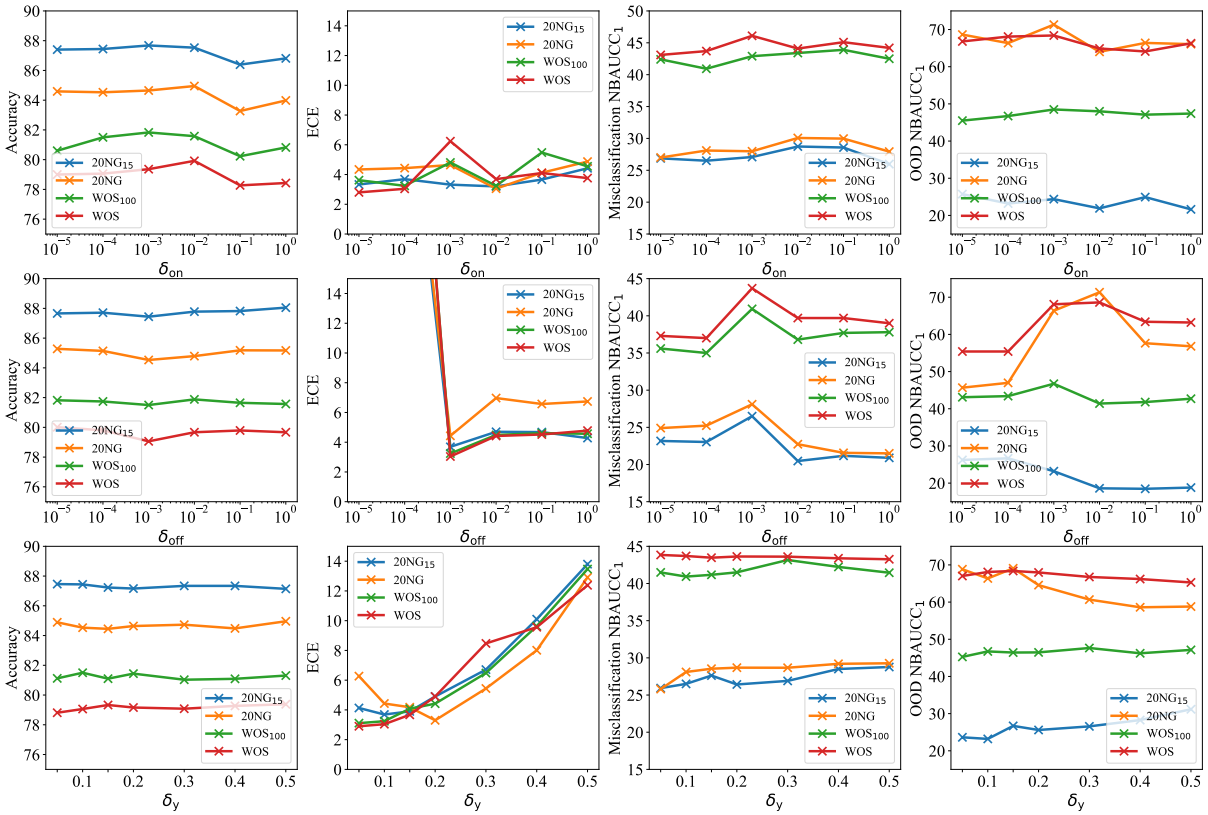

Figure 5: Parameter study of δ_{on}, δ_{off} and δ_y. We use $NBAUCC_1$ for OOD and misclassification detection.

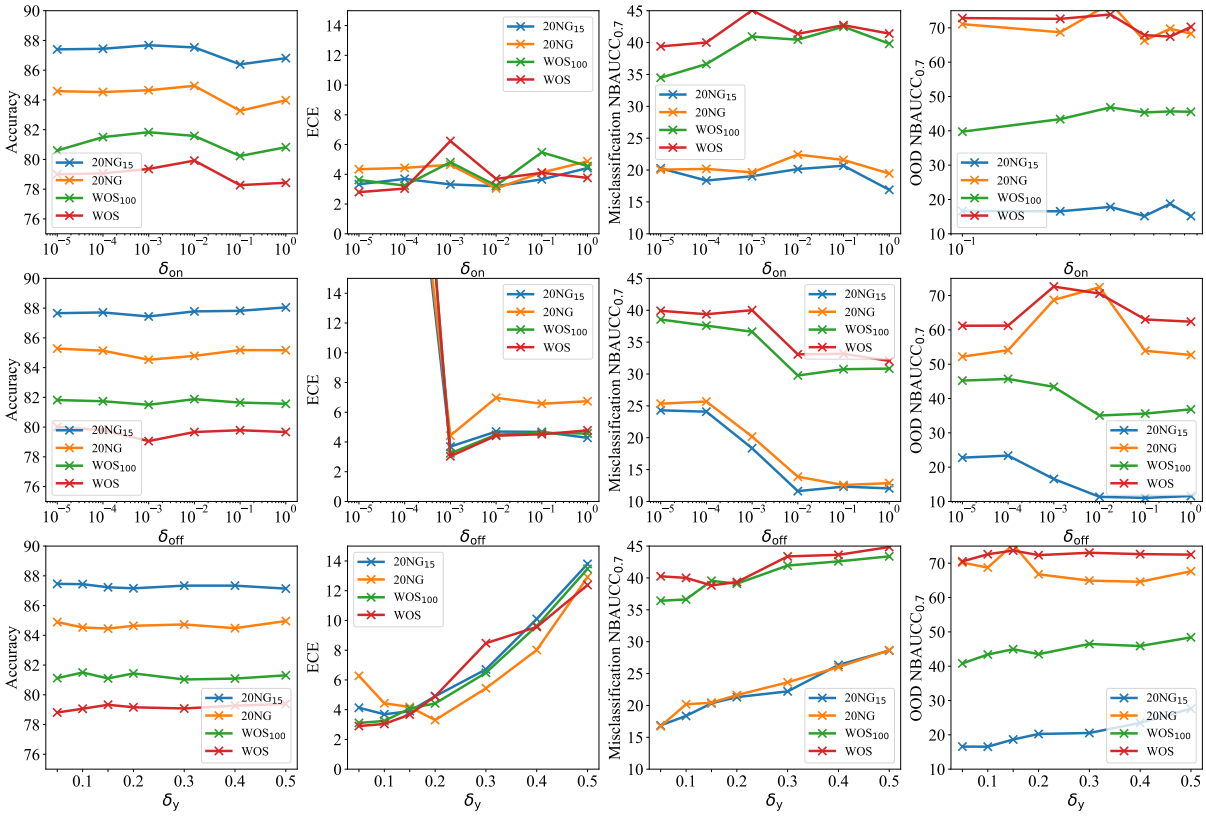

Figure 6: Parameter study of δ_{on}, δ_{off} and δ_y. We use $\text{NBAUCC}_{0.7}$ for OOD and misclassification detection.