LABO: Towards Learning Optimal Label Regularization via Bi-level Optimization

Peng Lu[1,2], **Ahmad Rashid**[1,3], **Ivan Kobyzev**[1]
Mehdi Rezagholizadeh[1], **Philippe Langlais**[2]
[1]Huawei Noah's Ark Lab, Canada
[2] Department of Computer Science and Operations Research, Université de Montréal
[3] Department of Statistics and Actuarial Science, University of Waterloo
peng.lu@umontreal.ca

Abstract

Regularization techniques are crucial to improving the generalization performance and training efficiency of deep neural networks. Many deep learning algorithms rely on weight decay, dropout, batch/layer normalization to converge faster and generalize. Label Smoothing (LS) is another simple, versatile and efficient regularization which can be applied to various supervised classification tasks. Conventional LS, however, regardless of the training instance assumes that each non-target class is equally likely. In this work, we present a general framework for training with label regularization, which includes conventional LS but can also model instance-specific variants. Based on this formulation, we propose an efficient way of learning LAbel regularization by devising a Bi-level Optimization (LABO) problem. We derive a deterministic and interpretable solution of the inner loop as the optimal label smoothing without the need to store the parameters or the output of a trained model. Finally, we conduct extensive experiments and demonstrate our LABO consistently yields improvement over conventional label regularization on various fields, including seven machine translation and three image classification tasks across various neural network architectures while maintaining training efficiency.

1 Introduction

Deep neural networks (DNNs) form the backbone of current state-of-the-art algorithms in various fields including natural language processing (Vaswani et al., 2017), computer vision (He et al., 2016; Dosovitskiy et al., 2021) and speech recognition (Schneider et al., 2019; Chen et al., 2021). However, heavily overparameterized models may incur overfitting and suffer from poor generalizations (Goodfellow et al., 2016). To address the issue, many regularization techniques have been developed in the literature: weight decay which constrains the optimization space (Krogh and Hertz, 1991), batch or layer normalization which speeds up the training of feed-forward NNs (Ioffe and Szegedy, 2015; Ba et al., 2016), and dropout which implicitly approximates the effect of averaging the predictions of all sparse subnetworks networks (Srivastava et al., 2014). Label smoothing (LS) is another simple regularization technique; it is widely applied to many applications including image classification (Szegedy et al., 2016) and token-level sequence generation (Pereyra et al., 2017) for enhancing the generalization, without suffering additional computational costs. It encourages a model to treat each non-target class as equally likely for classification by using a uniform distribution to smooth one-hot labels (He et al., 2019a; Vaswani et al., 2017). Although combining the uniform distribution with the original one-hot label is beneficial for regularization, conventional LS does not take into account the true relations between different label categories. More specifically, for token-level generation, uniformly allocating the probability mass on non-target words disregards the semantic relationship between non-target words and the context. On the other hand, the probability of the target is predefined and unchanged. However, the distribution of natural language exhibits remarkable variations in the per-token perplexity (Holtzman et al., 2020), which encourages us to adapt corresponding target probabilities for different contexts.

One of the instance-dependent techniques of learning the relation between different target categories is Knowledge Distillation (KD) (Hinton et al., 2015; Buciluă et al., 2006), which is a popular technique of transfer learning utilizing knowledge from related tasks or teacher models (Caruana, 1997; Pan and Yang, 2010; Lu et al., 2019). It is widely applied for model compression and ensembling across applications ranging from computer vision (He et al., 2019b; Xu et al., 2020; Lu et al., 2021) to natural language processing (Jiao et al., 2020; Sanh et al., 2019). However, KD requires

Findings of the Association for Computational Linguistics: ACL 2023, pages 5759–5774
July 9-14, 2023 ©2023 Association for Computational Linguistics

training a separate model as a teacher for every new task. Besides, it either introduces an extra inference pass to get the teacher's prediction for each instance during the training or requires saving the teacher's prediction for all training samples to avoid the extra forward pass of teacher models. This greatly increases the time and space complexity in practice. Especially for token-level generation tasks, e.g. machine translation, to save all the output probabilities of teacher models costs $O(NLV)$ space, where N is the number of sequences, L is the averaged length of all sequences and V is the vocabulary size. Besides the empirical success of KD, it is unclear how student networks benefit from these smoothed labels. A series of investigations have looked at regularization and have demonstrated that the success of both KD and label smoothing is due to a similar regularization effect of smoothed targets (Yuan et al., 2020; Zhang and Sabuncu, 2020). Based on this finding and the low training overhead of LS, there is a significant interest, in the community, in algorithms that can enhance conventional LS. (Zhang and Sabuncu, 2020) demonstrates the importance of an instance specific LS regularization. They demonstrate better performance compared to LS, but use a trained model to infer prior knowledge on the label space and thereby sacrifice some of the efficiency of LS.

In this work, we first revisit the conventional LS and generalize it to an instance-dependent label regularization framework with a constraint on overconfidence. Within this framework, we demonstrate that both LS and KD can be interpreted as instances of a smoothing distribution with a confidence penalty. Finally, we propose to learn the optimal smoothing function along with the model training, by devising a bi-level optimization problem. We solve the inner problem by giving a deterministic and interpretable solution and applying gradient-based optimization to solve the outer optimization.

Our contributions can be summarized as follows:

- We explicitly formulate a unified label regularization framework, in which the regularization distribution can be efficiently learnt along with model training by considering it as a bi-level optimization problem.
- We derive a closed-form solution to solve the inner loop of the bi-level optimization, which not only improves efficiency but makes the algorithm interpretable.
- We conducted extensive experiments on Machine Translation, IWSLT'14 (DE-EN, EN-DE, EN-FR, FR-EN), WMT'14 (EN-DE, DE-EN), IWLST'17 ({DE,FR}-EN), image classification (CIFAR10, CIFAR100 and ImageNet) and show that our method outperforms label smoothing consistently while maintaining the training efficiency.

2 Background

We will provide a brief overview of existing label regularization methods.

Label Smoothing. LS is a regularization technique to improve the generalization performance of neural networks by preventing the model from predicting the training examples overconfidently. It smoothes a one-hot target label with the uniform distribution $U(\cdot) = \frac{1}{K}$, where K is the number of classes. As a result, LS training is equivalent to training with a smoothed label $\hat{P}_U$, where:

$$\hat{P}_U(j|x) = \begin{cases} 1 - \alpha + \alpha U(j), & j = k \\ \alpha U(j), & j \neq k \end{cases}, \quad (1)$$

and k is the ground-truth class. We note from Eq. 1 that a higher α can lead to a smoother label distribution and a lower α to a *peakier* label.

Confidence Penalty. Another important technique similar to label smoothing is the confidence penalty (CP) (Pereyra et al., 2017), in which a regularization term $\mathbb{H}(p_\theta(x))$ is introduced into the objective function to directly punish the overconfidence of predictions:

$$\mathbb{H}(q(x), p_\theta(x)) - \beta \mathbb{H}(p_\theta(x)) \quad (2)$$

where $q(x)$ is the one-hot label, $p_\theta(x)$ is the output distribution of models, $\mathbb{H}(q(x), p_\theta(x))$ is the cross-entropy loss between the labels and the student's output, $\mathbb{H}(p_\theta(x))$ is the entropy of the output and β controls the strength of the confidence penalty.

Knowledge Distillation. Given access to a trained teacher model, assume that we want to train a student. Denote by $P_T(x)$ and $p_\theta(x)$ the teacher's and student's predictions respectively. For a classification problem, the total KD loss is defined as:

$$(1 - \alpha)\mathbb{H}(q(x), p_\theta(x)) + \alpha \mathbb{KL}(P_T(x), p_\theta(x)), \quad (3)$$

where $\mathbb{KL}$ is the Kullback–Leibler (KL) divergence and α is the scaling parameter. Note that we assume a temperature of 1 and omit it without loss of generality. KD training is equivalent to training with a smoothed label $\hat{P}(x)$, where:

$$\hat{P}_T(j|x) = \begin{cases} 1 - \alpha + \alpha P_T(j|x), & j = k \\ \alpha P_T(j|x), & j \neq k \end{cases}, \quad (4)$$

where k is the ground-truth class.

Both LS and KD incorporate training a model with a smoothed label. KD tends to perform better as the teacher's predictions over non-target classes can capture the similarities between different classes (Müller et al., 2019; Shen et al., 2021). However, LS and CP techniques are more efficient for training, since they do not require training a separate teacher network for every new task.

3 Methodology

In this section, we interpret our optimal label regularization from the bi-level optimization perspective. First, we provide a close look at the conventional uniform label regularization and show the generalized framework bridges the objectives of conventional LS and KD methods. Then, we introduce a closed-form solution to our optimal instance-dependent label smoothing and describe an online implementation under this formulation.

3.1 Generalized Label Regularization: A Close Look

Suppose that we have a K-class classification task to be learned by a neural network, $S_\theta(\cdot)$. Given a training set of $\{x_i, y_i\}_{i=1}^N$ samples where x_i is a data sample and y_i is the ground-truth label, the model $S_\theta(\cdot)$ outputs the probability $p_\theta(j|x_i)$ of each class $j \in \{1, \dots, K\}$:

$$p_\theta(j|x_i) = \frac{\exp(z_{i,j})}{\sum_{j'=1}^K \exp(z_{i,j'})}, \quad (5)$$

where $z_{i,j} = [S_\theta(x_i)]_j$ is the logit for j-th class of input x_i.

A general label smoothing can be formally written as:

$$\hat{P}_{ls}(j|x_i) = \begin{cases} 1 - \alpha + \alpha \cdot P_{ls}(j|x_i), & j = k \\ \alpha \cdot P_{ls}(j|x_i), & j \neq k \end{cases}, \quad (6)$$

where $\hat{P}_{ls}$ is a smoothed label, P_{ls} is a smoothing distribution, k is the ground-truth class of x_i, and α is a hyperparameter that determines the amount of smoothing. If $\alpha = 0$, we obtain the original one-hot encoded label. For the original label smoothing method, the probability $P_{ls}(\cdot|x)$ is independent on the sample x and is taken to be the uniform distribution $P_{ls}(j) = U(j)$, However, the training can benefit from instance-dependent regularization. (Yuan et al., 2020; Zhang and Sabuncu, 2020). In this work, we consider the general form of LS $\hat{P}_{ls}(\cdot|x_i)$ which is instance-dependent and not necessarily uniform.

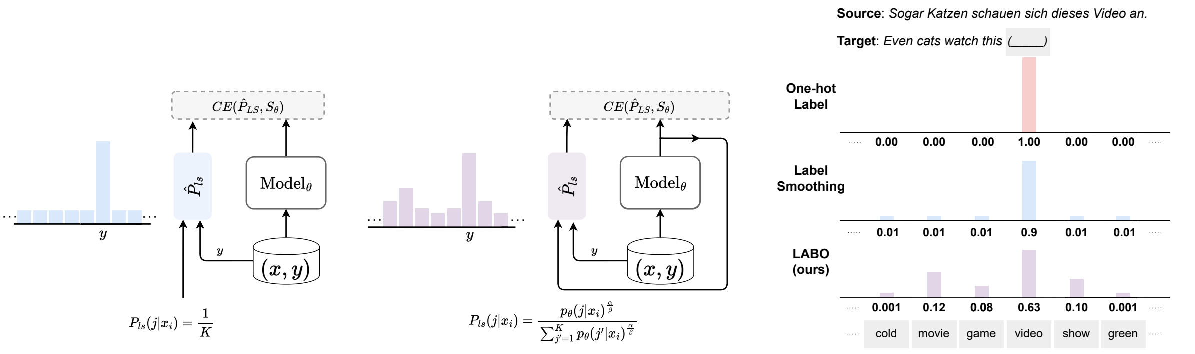
Let us consider the Cross Entropy (CE) loss with the smoothed labels:

$$\begin{aligned} \mathcal{L}_\theta(P_{ls}) &= \frac{1}{N}\sum_{i=1}^N \left[-\sum_{j=1}^K \hat{P}_{ls}(j|x_i) \log p_\theta(j|x_i) \right] \\ &= \frac{1}{N}\sum_{i=1}^N \mathbb{E}_{\hat{P}_{ls}} \left[-\log p_\theta(j|x_i) \right]. \quad (7) \end{aligned}$$

Note that computing the weighted sum of the negative log-likelihood of each probability of label can be viewed as taking expectation of the negative log-likelihood over label space under a certain distribution $\hat{P}_{ls}$. We modify this loss by adding a KL divergence term $\mathbb{KL}(P_{ls}(\cdot|x_i)\|U(\cdot))$ into Eq. 7 which encourages the sample-wise smoothing distribution $P_{ls}(\cdot|x_i)$ to be close to the uniform distribution to handle the over-confidence.

$$\begin{aligned} \mathcal{R}_\theta(P_{ls}) = \frac{1}{N}\sum_{i=1}^N \Big[& \mathbb{E}_{\hat{P}_{ls}} \left[-\log p_\theta(j|x_i) \right] + \\ & \beta \mathbb{KL}(P_{ls}(\cdot|x_i)\|U(\cdot)) \Big], \end{aligned} \quad (8)$$

where β is a hyper-parameter. The instance-dependent smoothing $P_{ls}(\cdot|x_i)$ can be viewed as the prior knowledge over the label space for a sample x_i. This KL divergence term can be understood as a measure of the 'smoothness' or 'over-confidence' of each training label. Specifically for token-level generation tasks, the over-confidence of model results in the output of repetitive or most frequent but irrelevant text, which is detrimental to the generalization of the model (Chorowski and Jaitly, 2017; Holtzman et al., 2020; Meister et al., 2020a). We choose a Uniform distribution to constrain P_{ls} in the KL term because we would like P_{ls} to contain more information on non-target classes. It plays a role similar to the temperature in KD


$CE(\hat{P}_{LS}, S_\theta)$
$\hat{P}_{ls}$
Model$_\theta$
y
(x, y)
$P_{ls}(j|x_i) = \frac{1}{K}$
$CE(\hat{P}_{LS}, S_\theta)$
$\hat{P}_{ls}$
Model$_\theta$
y
(x, y)
$P_{ls}(j|x_i) = \frac{p_\theta(j|x_i)^{\frac{\alpha}{\beta}}}{\sum_{j'=1}^{K} p_\theta(j'|x_i)^{\frac{\alpha}{\beta}}}$
Source: Sogar Katzen schauen sich dieses Video an.
Target: Even cats watch this (____)
One-hot Label
0.00 0.00 0.00 1.00 0.00 0.00
Label Smoothing
0.01 0.01 0.01 0.9 0.01 0.01
LABO (ours)
0.001 0.12 0.08 0.63 0.10 0.001
cold movie game video show green

(a) Label Smoothing

(b) LABO (Ours)

(c) An example of different methods for a token prediction

Figure 1: Comparison of Label Smoothing and LABO. a) shows Label Smoothing. b) shows LABO. c) gives an illustration of a one-hot target, label smoothing and our LABO method given a translation pair sampled from the IWSLT'14 (DE-EN) dataset. It shows a part of the probability vector when predicting the next ground-truth word "*video*". Our method assigns larger probabilities to the context-relevant tokens compared to label smoothing.

and controls the sharpness of the distribution P_{ls}. On the one hand side, in case of KD, it regulates the amount of prior knowledge we inject in the smoothed label. On the other hand side, it reduces the overconfidence of the trained network by increasing the entropy of smoothed labels (a phenomenon studied by Zhang and Sabuncu (2020)).

Next, we show the post-hoc interpretability of this formulation. The following two Remarks discuss the relationship of this objective with conventional LS, confidence penalty and KD.

Remark 1. *When $P_{ls}(\cdot|x_i)$ is taken to be the uniform distribution $U(\cdot)$ for any x_i, the objective in Eq. 8 reduces back to the one in Eq. 7 since $\mathbb{KL}\big(U(\cdot)\|U(\cdot)\big) = 0$.*

Remark 2. *This framework can include KD with temperature $\tau = 1$ as a special case. Suppose $P_T(\cdot|x_i)$ to be the output probability of a teacher model $T(\cdot)$ for any x_i, the objective of KD can be rewritten as an expectation of negative log-likelihood w.r.t. a transformed distribution $\hat{P}_T$ plus a KL term between the teacher output and uniform distribution.*

$$\mathcal{R}_\theta(P_T) = \frac{1}{N}\sum_{i=1}^{N}\Big[\mathbb{E}_{\hat{P}_T}\left[-\log p_\theta(j|x_i)\right] + \alpha\mathbb{KL}(P_T(\cdot|x_i)\|U(\cdot))\Big] + \Theta(\log K), \tag{9}$$

where $\hat{P}_T(j|x_i) = (1-\alpha+\alpha\cdot P_T(k|x_i))$ for $j = k$ the ground truth and $\hat{P}_T(j|x_i) = \alpha\cdot P_T(j|x_i)$ for $j \neq k$. This objective function Eq. 8 bridges the objective of label smoothing and knowledge distillation, since it is easy to convert our objective to KD or LS by replacing the $P_{ls}(\cdot|x_i)$ in Eq. 8 with $P_T(\cdot|x_i)$ or $U(\cdot)$, respectively. There is an inherent relation among these methods. KD and LS, in particular, can be interpreted as instances of our method with a fixed smoothing distribution, which is consistent with recent work (Yuan et al., 2020; Zhang and Sabuncu, 2020).

3.2 Learning Label regularization via Bi-level Optimization

The choice of the smoothing distribution P_{ls} determines the label regularization method. As described above, in KD the smoothing distribution is an output of the pre-trained teacher model, and in LS it is a uniform distribution. Generally speaking, the optimal smoothing distribution is unknown, and ideally, we would like to learn the optimal P_{ls}. In this regard, we set up the following two-stage optimization problem:

$$\min_\theta \mathcal{R}(P^*_{ls}(\theta), \theta), \quad \text{subject to } P^*_{ls}(\theta) = \arg\min_{P_{ls}} \mathcal{R}_\theta(P_{ls}). \tag{10}$$

This optimization setting, also called bilevel optimization (Colson et al., 2007), is strongly NP-hard (Jeroslow, 1985) so getting an exact solution is difficult. To solve this problem in our case, we first prove that the inner loop is a convex optimization problem by computing the Hessian matrix $\mathbf{H}_i$ of $\mathcal{R}_\theta(P_{ls})$ w.r.t. $P_{ls}(\cdot|x_i)$ for each training instance x_i.

$$\mathbf{H}_i = \text{diag}(\frac{\beta}{P_{ls}(1)}, \frac{\beta}{P_{ls}(2)}, \cdots, \frac{\beta}{P_{ls}(K)}) \tag{11}$$

When β is greater than zero, the Hessian is positive definite. Therefore, for the inner optimization loop we can derive the closed-form solution by using a Lagrangian multiplier. The following Theorem 1 gives the explicit formulation of $\mathcal{R}_\theta(P^*_{ls})$. For the details of this derivation please refer to Appendix A.

Theorem 1. *The solution to the inner loop of optimization in Eq. 10 is given by:*

$$P^*_{ls}(j|x_i) = \frac{p_\theta(j|x_i)^{\frac{\alpha}{\beta}}}{\sum_{j'=1}^{K} p_\theta(j'|x_i)^{\frac{\alpha}{\beta}}}, \tag{12}$$

where $p_\theta(j|x_i)$ is the output probability of j-th class of model $S_\theta(\cdot)$, α is the smoothing coefficient defined in Eq. 6, and β is defined in Eq. 8.

As a result, we reduced the two-stage optimization problem in Eq. 10 to a regular single-stage minimization:

$$\min_\theta \frac{1}{N}\sum_{i=1}^{N} \mathcal{R}_\theta(P^*_{ls}(\cdot|x_i)), \tag{13}$$

where

$$\mathcal{R}_\theta(P^*_{ls}(\cdot|x_i)) = \sum_{j=1}^{K}\big[-\hat{P}^*_{ls}(j|x_i)\log p_\theta(j|x_i) \\ +\beta P^*_{ls}(j|x_i)\log(K\cdot P^*_{ls}(j|x_i))\big], \tag{14}$$

$P^*_{ls}(j|x_i)$ is given in Theorem 1, and $\hat{P}^*_{ls}(j|x_i)$ is defined in Eq. 6. Note that the solution P^*_{ls} is deterministic and interpretable. Moreover, the two remarks below demonstrate the relation of our LABO with LS and KD methods.

Remark 3. *When β is extremely large, P^*_{ls} will be close to the Uniform distribution, and our objective function will be equivalent to optimizing the CE loss with a Uniform LS regularizer.*

Remark 4. *There is an intrinsic connection between the P^*_{ls} distribution and generating softmax outputs with a temperature factor. Specifically, when $\beta = \alpha\cdot\tau$, we could have*

$$P^*_{ls}(j|x_i) = \frac{\exp(\frac{z_{i,j}}{\tau})}{\sum_{j'=1}^{K}\exp(\frac{z_{i,j'}}{\tau})}. \tag{15}$$

The smoothing distribution in this case becomes the temperature smoothed output of the model, which is similar as the smoothed targets used in KD methods[1].

To summarize, our method can be expressed as an alternating two-stage process. We generate optimal smoothed labels P^*_{ls} using Theorem 1 in the first stage. Then, in the second phase, we fix the P^*_{ls} to compute loss $\mathcal{R}_\theta(P^*_{ls})$ and update the model parameters.

3.3 Implementation Details

Two-stage training. Our solution provides the closed-form answer to the inner loop optimization (1st-stage) and for the outer loop (2nd-stage) the model $f(\theta)$ is updated by gradient-descent. LABO conducts a one-step update for the 2nd stage, namely, for each training step, we compute the optimal smoothing P^*_{ls} and update the model, which eliminates the need for additional memory or storage for the parameters or outputs of a prior, trained model. The training process is shown in Algorithm 1.

Adaptive α. The value of α determines the probability mass to smooth. To get rid of hyper-parameter searching, we provide an instance-specific α as a function of the entropy of the output.

$$\alpha = \frac{\mathbb{H}(U) - \rho\mathbb{H}(P_\theta)}{\mathbb{H}(U)}, \tag{16}$$

where $\rho \in [0.5, 1]$, in our experiments, we use $\rho = 0.5$. In the experiments, we fix the ratio of $\frac{\beta}{\alpha}$ as a hyper-parameter, so the value of β will change accordingly.

Hypergradient In the outer loop, the derivative of loss $\mathcal{R}(P^*_{ls}(\theta), \theta)$ w.r.t. θ consists of two components:

$$\nabla_{P^*_{ls}}\mathcal{R}\nabla_\theta P^*_{ls} + \nabla_\theta\mathcal{R} \tag{17}$$

because P^*_{ls} is the global solution of objective $\mathcal{R}$ in the inner loop, $\nabla_{P^*_{ls}}\mathcal{R}$ equals zero. Therefore, the hypergradient equals zero, we neglect this component in computation for efficiency.

4 Experiments

We evaluate the performance of our proposed LABO method on both Machine Translation and Computer Vision. For machine translation, we evaluate on the IWSLT'14 (Cettolo et al., 2014), IWLST'17 (Cettolo et al., 2017) and WMT'14 (Bojar et al., 2014) datasets using transformer-base models and for image classification on CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009) and ImageNet2012 (Deng et al., 2009) using ResNet-based

[1] The derivation can be found in Appendix B.

Table 1: BLEU (mean± std) with beam size 5 for method LS or LABO on Bilingual Translation Tasks.

	IWSLT'14		IWSLT'14		WMT'14	
Method	(DE-EN)	(EN-DE)	(EN-FR)	(FR-EN)	(EN-DE)	(DE-EN)
Transformer	33.9± 0.09	27.8± 0.11	40.0± 0.15	39.0± 0.14	27.1± 0.03	29.8± 0.10
w/ LS	34.5± 0.14	28.3± 0.16	40.5± 0.16	39.8± 0.05	27.7± 0.09	31.9± 0.09
w/ CP	34.2± 0.15	27.9± 0.07	40.4± 0.20	39.2± 0.13	27.4± 0.13	30.4± 0.08
w/ FL	33.0± 0.13	26.8± 0.16	39.2± 0.16	38.3± 0.05	26.6± 0.09	28.9± 0.09
w/ AFL	34.2± 0.14	27.9± 0.13	40.5± 0.09	39.5± 0.16	27.5±0.10	30.3±0.08
w/ LABO (ours)	**35.2**± 0.07	**28.8**± 0.15	**40.9**± 0.08	**40.3**± 0.05	**28.3**± 0.05	**32.3**± 0.06

Algorithm 1 LABO: Two-stage training

Input: Training set D_{train}, batch size n, number of steps T, learning rate η, $\hat{P}_{ls}$ warm-up steps T_w;

```
for $i \leftarrow 1$ to $T$ do
    Sample a mini-batch $S = \{(x_i, y_i)\}|_{i=1}^{n}$ from $D_{\text{train}}$;
    if $i < T_w$ then
        Compute the $\hat{P}_{ls}$ with Uniform Distribution for the mini-batch data;
    else
        Compute the $\hat{P}_{ls}$ according to $\hat{P}_{ls}^{*}$ solution for the mini-batch data;
    end if
    Update $\theta_{t+1} = \theta_t - \eta \nabla_\theta R_\theta(\hat{P}_{ls}^{*}, S)$;
end for
```

models of various sizes (parameters). All experiments were performed on one or more NVIDIA Tesla (V100) GPUs.

4.1 Experiments on Machine Translation

We evaluate our method on six machine translation benchmarks including IWSLT'14 German to English (DE-EN), English to German (EN-DE), English to French (EN-FR), French to English (FR-EN), WMT'14 English to German (EN-DE) and German to English (De-EN) benchmark. We use the 6-layer encoder-decoder transformer as our backbone model for all experiments. We follow the hyper-parameter settings for the architecture and training reported in (Gehring et al., 2017; Vaswani et al., 2017). Specifically, we train the model with a maximum of 4,096 tokens per mini-batch for 150 or 50 epochs on IWSLT and WMT datasets, respectively. For optimization, we apply Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.98$ and weight decay 1e-4. For LABO, we also perform explore {1.15, 1.25} for the only hyper-parameter $\tau = \beta/\alpha$. We report on the BLEU-4 (Papineni et al., 2002) metric to compare between the different models[2].

Baselines. We compare our methods with three baselines that try to handle the over-confidence problem. LS uses the combination of a one-hot vector and a uniform distribution to construct the target distribution. CP (Pereyra et al., 2017) punishes over-confidence by regularizing the entropy of model predictions. FL (Lin et al., 2017) utilizes the focal loss to assign smaller weights to well-learned tokens for every training iteration. AFL (Raunak et al., 2020) is a generalize Focal loss which establish a the trade-off for penalizing low-confidence predictions.

Bilingual Translation. Tab. 1 shows the results of IWLST'14 DE-EN, EN-DE, EN-FR, FR-EN and WMT'14 EN-DE, DE-EN translation tasks. The backbone transformers showed close or better BLEU scores with the numbers reported in (Vaswani et al., 2017). All confidence penalizing techniques except FL can improve the performance of transformer models. The drop in performance of FL is consistent with the finding of long-tail phenomena in neural machine translation systems (Raunak et al., 2020). Our methods consistently outperform the baseline Transformer (w/ LS), which demonstrates its effectiveness. This is across different language pairs and dataset sizes.

Multilingual Translation. We also evaluate our LABO method on IWLST'17 ({DE, FR}-EN) dataset by using multilingual transformers. We learn a joint BPE code for all three languages and use sacrebleu for evaluating the test set. Tab.2 shows LABO achieves consistent improvement over the original label smoothing on the multilingual translation dataset.

[2] Our experiments were conducted with Fairseq toolkit (github.com/pytorch/fairseq).

Table 2: BLEU scores for method LS or LABO on multilingual Translation Tasks.

IWSLT'17	BLEU	
Model	DE-EN	FR-EN
Transformer	26.9	35.4
w/ LS	28.0	36.8
w/ LABO (ours)	**28.4**	**37.2**

4.2 Experiments on Image Classification

Setup for CIFAR experiments. We evaluated our method on different model architectures including MobileNetV2 (Sandler et al., 2018), and ResNet18 (He et al., 2016). We follow standard data augmentation schemes: random crop and horizontal flip to augment the original training images. We sampled 10$\%$ images from the training split as a validation set. The models are trained for 200 epochs with a batch size of 128. For optimization, we used stochastic gradient descent with a momentum of 0.9, and weight decay set to 5e-4. The learning rate starts at 0.1 and is then divided by 5 at epochs 60, 120 and 160. All experiments are repeated 5 times with random initialization. For the KD experiments, we used ResNeXt29 as the teacher. All teacher models are trained from scratch and picked based on their best accuracy. To explore the best hyper-parameters, we conduct a grid search over parameter pools. We explore {0.1, 0.2, 0.5, 0.9} for α and {5, 10, 20, 40} for KD temperature.

Results. Next, we conduct a series of experiments on two models to compare our approach with other methods without requiring any pre-trained models on two CIFAR datasets. Note that KD is reported as a reference. All experiments are repeated 5 times with random initialization. The baseline methods include CS-KD which doesn't require training a teacher and constrains the smoothed output between different samples of the same class to be close (Yun et al., 2020), TF-reg which regularizes the training with a manually designed teacher (Yuan et al., 2020) and Beta-LS which leverages a similar capacity model to learn an instance-specific prior over the smoothing distribution (Zhang and Sabuncu, 2020). KR-LS utilizes a class-wise target table which captures the relation of classes (Ding et al., 2021). We follow their best hyper-parameter settings. The time for all baselines is measured based on the original implementations. [3][4][5]

Tab. 3 shows the test accuracy and training time for 200 epochs of different methods on two models. It can be seen that our method consistently improves the classification performance on both lightweight and complex models, which indicates its general applicability. Besides, it shows the training time of our method is close to Base or LS methods, which show its stable efficiency over other baselines such as Beta-LS, which still requires a separate model to output a learned prior over the smoothing distribution. Our method achieves better performance than other strong baseline methods consistently. We have to mention that our computation for smoothing distribution is deterministic and excluded from the computation graph for the gradient calculation. Therefore, the implementation of our method requires less time and space during the training as we don't need to train a separate model.

5 Discussion

One of important problems of neural language models is the large discrepancy of predicted probabilities between tokens with low and high frequency, In other words, the Long-Tailed Phenomena in the neural language models (Zhao and Marcus, 2012; Demeter et al., 2020). In this section, we study the impact of our method on the prediction of tokens with different frequencies.

We first computed the averaged frequency of tokens in every source sentence $x = [x_i]|_{i=1}^{N}$ for the validation and test sets of IWSLT 2014 (De-En). This Frequency Score (FS) is defined in (Raunak et al., 2020):

$$F(x) = \frac{\sum_{i=1}^{N} f(x_i)}{N}, \tag{18}$$

where $f(x_i)$ is the frequency of the token x_i in the training corpus. Next, we divide each dataset into three parts of 2400 sentences in order of decreasing FS. Tab. 4 shows the results of LS and LABO on different splits. All models perform much better on split-most than split-medium and least. Our method first demonstrates consistent improvements on three splits for both validation and test datasets. Besides, our LABO provides at least the same magnitude of improvements on least and medium splits as the split-most.

[3] github.com/alinlab/cs-kd

[4] github.com/yuanli2333/Teacher-free-Knowledge-Distillation

[5] github.com/ZhiluZhang123/neurips_2020_distillation

Table 3: Comparison between different Smoothed labels methods. Averaged test accuracy and training time are reported. The training time is measured on a single NVIDIA V100 GPU.

	CIFAR100 (Acc. & Time)				CIFAR 10 (Acc. & Time)			
	MobileNetv2		ResNet18		MobileNetv2		ResNet18	
Base	67.98	1.0×	76.92	1.0×	90.55	1.0×	94.82	1.0×
KD	70.99 (↑3.01)	3.7×	77.78 (↑0.86)	3.8×	91.52 (↑0.97)	3.7×	**95.28** (↑0.46)	3.9×
LS	68.69 (↑0.71)	1.0×	77.67 (↑0.75)	1.0×	90.82 (↑0.27)	1.0×	95.02 (↑0.20)	1.0×
CS-KD	70.36 (↑2.38)	1.1×	77.95 (↑1.03)	1.3×	91.17 (↑0.62)	1.1×	94.90 (↑0.08)	1.3×
TF-reg	70.08 (↑2.10)	1.0×	77.91 (↑0.99)	1.2×	90.97 (↑0.42)	1.1×	95.05 (↑0.23)	1.2×
Beta-LS	70.45 (↑2.47)	1.5×	77.83 (↑0.91)	1.4×	90.89 (↑0.34)	1.5×	94.87 (↑0.05)	1.6×
KR-LS	70.12 (↑2.14)	1.0×	77.82 (↑0.90)	1.0×	90.67 (↑0.12)	1.0×	94.76 (↓0.06)	1.0×
LABO	**71.05** (↑3.07)	1.0×	**78.10** (↑1.18)	1.0×	**91.53** (↑0.98)	1.0×	95.21 (↑0.39)	1.0×

5.1 Analysis on predictions with Low-frequency tokens

Table 4: Performance of LS and LABO on splits with different averaged token frequencies.

IWSLT'14 (DE-EN)	Validation		
Split	Most	Medium	Least
LS	40.3	35.2	33.1
LABO	40.7	35.8	33.8
Δ	+0.4	+0.6	+0.7
IWSLT'14 (DE-EN)	Test		
Split	Most	Medium	Least
LS	38.4	33.5	32.1
LABO	39.0	34.3	32.7
Δ	+0.6	+0.8	+0.6

Next, We investigate the probability distribution of the selected prediction in each step of beam search, namely, the the probability of top hypothesis finally chosen during decoding. Figure 2 shows the histogram for LS and LABO on three splits. The probabilities of our LABO method concentrated on around 0.4 while the corresponding probabilities of LS concentrated on 0.9. Our method reduces the discrepancy between predicted probabilities of different tokens, which facilitates the inference process during beam search by avoiding creating extremely large probabilities.

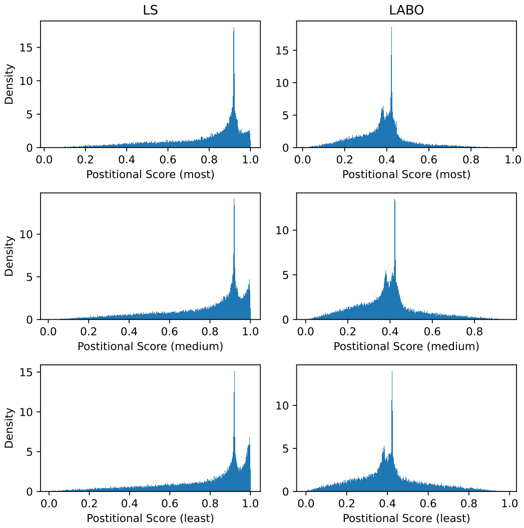


Figure 2: The distribution of probabilities of the selected predictions in each step of beam search for LS (left) and LABO (right) on splits with most, medium and least averaged token frequencies. All experiments are using beam size 5.

6 Related Work

There is a great body of works inspired by the KD technique. Some of them focus on boosting performance with explicitly regularized smoothed targets. Zhang and Sabuncu (2020) interpret student-teacher training as an amortized maximum aposteriori estimation and derive an equivalence between self-distillation and instance-dependent label smoothing. This analysis helped them to devise a regularization scheme, Beta Smoothing. However, they still use an extra model to infer a prior distribution on their smoothing technique during the training. Yun et al. (2020) introduce an additional regularization to penalize the predictive output between different samples of the same class. There are several works discussing the empirical impact of KD and giving different practical suggestions. Kobyzev et al. (2023) conducted extensive experiments to explore the effect of label regularization to the generalization ability of compact pre-trained language models.

Other works propose to learn class-wise label smoothing or progressive refining of smoothed labels. Ding et al. (2021) propose to capture the relation of classes by introducing decoupled labels table which increases space complexity by $O(K \times K)$. The concurrent work (Kim et al., 2021) utilizes the model trained at i-th epoch as the teacher to regularize the training at $(i + 1)$-th epoch along with annealing techniques. However, contrary to our work, it either requires a separate model as teacher, or stores the labels generated at the i-th epoch for training the next epoch, whereby increasing space complexity by $O(N \times K)$. Meister et al. (2020b) investigate the relationship between generalized entropy regularization and label smoothing and provide empirical results on two text generation tasks. Chen et al. (2022) propose a masked Label Smoothing to remove the conflict by reallocating the smoothed probabilities based on the difference among languages. Lu et al. (2022) propose to learn confidence for NMT by jointly training a ConNet model to estimate the confidence of the prediction. We develop LABO motivated by a principled starting point: to generalize the smoothing distribution to a general form with neither time nor space complexity increase during the training and inference.

7 Conclusion

Our aim in this work is to fill the accuracy gap between Label Smoothing and Knowledge Distillation techniques while maintaining the training efficiency of LS regularization. We proposed learning an instance-dependent label smoothing regularization simultaneously with training our model on the target. We began by generalizing the classical LS method and introduced our objective function by substituting the uniform distribution with a general, instance-dependent, discrete distribution. Within this formulation, we explained the relationship between the LS and KD. Then, using a bi-level optimization approach, we obtained an approximation for the optimal smoothing function. We conducted extensive experiments to compare our model with conventional LS, KD and various state-of-the-art label regularization methods on popular MT and V benchmarks and showed the effectiveness and efficiency of our technique. Finally, we analyze the impact of our methods on the prediction of neural machine translation systems under different averaged token frequency settings and show our methods can greatly reduce the discrepancy between predicted probabilities of different tokens.

In practice, apart from general regularization techniques like dropout and weight decay, many advanced techniques are designed for specific tasks like FixNorm (Nguyen and Chiang, 2018), CutMix (Yun et al., 2019) and SwitchOut (Wang et al., 2018). We leave the future work to combine these methods with our technique. Moreover, we plan to explore the practical applications of our method for large-scale model training. One specific application is to improve the pre-training of large language models and vision transformers.

Acknowledgments

We thank Mindspore,[6] which is a new deep learning computing framework, for partial support of this work.

8 Limitations

In the current work, we adapt one-step gradient descent training for the outer loop based on our bi-level optimization framework. Since this outer loop optimization doesn't have a closed-form solution, determining how many steps to perform for the outer loop for better outer optimization is still important to explore.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Jan Niehues, Sebastian Stüker, Katsuhito Sudoh, Koichiro Yoshino, and Christian Federmann. 2017. Overview of the IWSLT 2017 evaluation campaign.

[6] www.mindspore.cn/

In *Proceedings of the 14th International Conference on Spoken Language Translation*, pages 2–14, Tokyo, Japan. International Workshop on Spoken Language Translation.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*.

Liang Chen, Runxin Xu, and Baobao Chang. 2022. Focus on the target's vocabulary: Masked label smoothing for machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 665–671. Association for Computational Linguistics.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, and Furu Wei. 2021. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *CoRR*, abs/2110.13900.

Jan Chorowski and Navdeep Jaitly. 2017. Towards better decoding and language model integration in sequence to sequence models. In *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pages 523–527. ISCA.

Benoît Colson, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256.

David Demeter, Gregory Kimmel, and Doug Downey. 2020. Stolen probability: A structural weakness of neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2191–2197. Association for Computational Linguistics.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society.

Qianggang Ding, Sifan Wu, Tao Dai, Hao Sun, Jiadong Guo, Zhang-Hua Fu, and Shutao Xia. 2021. Knowledge refinery: Learning from decoupled label. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 7228–7235. AAAI Press.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. 2016. *Deep Learning*. Adaptive computation and machine learning. MIT Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. 2019a. Bag of tricks for image classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 558–567. Computer Vision Foundation / IEEE.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. 2019b. Bag of tricks for image classification with convolutional neural networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

R. G. Jeroslow. 1985. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32(2):146–164.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4163–4174.

Kyungyul Kim, ByeongMoon Ji, Doyoung Yoon, and Sangheum Hwang. 2021. Self-knowledge distillation with progressive refinement of targets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6567–6576.

Ivan Kobyzev, Aref Jafari, Mehdi Rezagholizadeh, Tianda Li, Alan Do-omri, Peng Lu, Pascal Poupart, and Ali Ghodsi. 2023. Do we need label regularization to fine-tune pre-trained language models? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 166–177, Dubrovnik, Croatia. Association for Computational Linguistics.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Anders Krogh and John A. Hertz. 1991. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, pages 950–957. Morgan Kaufmann.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007. IEEE Computer Society.

Peng Lu, Ting Bai, and Philippe Langlais. 2019. SC-LSTM: learning task-specific representations in multi-task learning for sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2396–2406. Association for Computational Linguistics.

Peng Lu, Abbas Ghaddar, Ahmad Rashid, Mehdi Rezagholizadeh, Ali Ghodsi, and Philippe Langlais. 2021. RW-KD: sample-wise loss terms re-weighting for knowledge distillation. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 3145–3152. Association for Computational Linguistics.

Yu Lu, Jiali Zeng, Jiajun Zhang, Shuangzhi Wu, and Mu Li. 2022. Learning confidence for transformer-based neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2353–2364. Association for Computational Linguistics.

Clara Meister, Elizabeth Salesky, and Ryan Cotterell. 2020a. Generalized entropy regularization or: There's nothing special about label smoothing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6870–6886. Association for Computational Linguistics.

Clara Meister, Elizabeth Salesky, and Ryan Cotterell. 2020b. Generalized entropy regularization or: There's nothing special about label smoothing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6870–6886. Association for Computational Linguistics.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4696–4705.

Toan Nguyen and David Chiang. 2018. Improving lexical choice in neural machine translation. 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 334–343, New Orleans, Louisiana. Association for Computational Linguistics.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Vikas Raunak, Siddharth Dalmia, Vivek Gupta, and Florian Metze. 2020. On long-tailed phenomena in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3088–3095. Association for Computational Linguistics.

Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 3465–3469. ISCA.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhiqiang Shen, Zechun Liu, Dejia Xu, Zitian Chen, Kwang-Ting Cheng, and Marios Savvides. 2021. Is label smoothing truly incompatible with knowledge distillation: An empirical study. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 856–861. Association for Computational Linguistics.

Kunran Xu, Lai Rui, Yishi Li, and Lin Gu. 2020. Feature normalized knowledge distillation for image classification. In *Computer Vision – ECCV 2020*, pages 664–680. Springer International Publishing.

Li Yuan, Francis E. H. Tay, Guilin Li, Tao Wang, and Jiashi Feng. 2020. Revisiting knowledge distillation via label smoothing regularization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 3902–3910. Computer Vision Foundation / IEEE.

Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6022–6031. IEEE.

Sukmin Yun, Jongjin Park, Kimin Lee, and Jinwoo Shin. 2020. Regularizing class-wise predictions via self-knowledge distillation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13873–13882. IEEE.

Zhilu Zhang and Mert R. Sabuncu. 2020. Self-distillation as instance-specific label smoothing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qiuye Zhao and Mitch Marcus. 2012. Long-tail distributions and unsupervised learning of morphology. In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*, pages 3121–3136. Indian Institute of Technology Bombay.

A Derivation of the proof for Theorem 1

Proof. First, observe that the function $\mathcal{R}_\theta$ defined in Eq. 8 is a convex combination of N nonnegative functions $\mathcal{R}_i = \mathbb{E}_{\hat{P}_{ls}}[-\log p_\theta(j|x_i)] + \beta\mathbb{KL}(P_{ls}(\cdot|x_i)\|U(\cdot))$, for $i = 1, \dots, N$. We will show that each of $\mathcal{R}_i$ is a convex function of the components of simplex $P_{ls}(\cdot|x_i)$ by computing the Hessian matrix of $\mathcal{R}_i$ with respect to $P_{ls}(\cdot|x_i)$:

$$\begin{aligned}\mathbf{H}(R_i(P_{ls}(1), \cdots, P_{ls}(K)) &= \begin{bmatrix} \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(1)^2}, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(1)\partial P_{ls}(2)}, & \cdots, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(1)\partial P_{ls}(K)} \\ \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(2)\partial P_{ls}(1)}, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(2)^2}, & \cdots, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(2)\partial P_{ls}(K)} \\ \vdots, & \vdots, & \ddots, & \vdots \\ \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(K)\partial P_{ls}(1)}, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(K)\partial P_{ls}(2)}, & \cdots, & \frac{\partial^2\mathcal{R}_i}{\partial P_{ls}(K)^2} \end{bmatrix} \qquad (19) \\ &= \mathrm{diag}(\frac{\beta}{P_{ls}(1)}, \frac{\beta}{P_{ls}(2)}, \cdots, \frac{\beta}{P_{ls}(K)})\end{aligned}$$

When β is greater than zero, the Hessian is positive definite. Therefore, each R_i is a convex function of the components of $P_{ls}(\cdot|x_i)$. As a result, $\mathcal{R}_\theta$ is a convex function of the collection of components of every simplex $P_{ls}(\cdot|x_i)$.

For simplicity, we derive the global optimum solution for each $\mathcal{R}_i$ with a Lagrangian multiplier:

$$L_i(P_{ls}, \lambda_L) = \sum_{j'=1}^{K}\left[-\hat{P}_{ls}(j')\log p(j') + \beta \cdot P_{ls}(j')\log\frac{P_{ls}(j')}{1/K}\right] + \lambda_L(\sum_{j'=1}^{K} P_{ls}(j') - 1), \qquad (20)$$

where we omit the dependency on x_i to simplify the notation. We set the corresponding gradients equal to 0 to obtain the global optimum for $j = 1, \dots, K$.

$$\frac{\partial L_i}{\partial P_{ls}(j)} = -\alpha\log p(j) + \beta\cdot\log P_{ls}(j) + \beta + \beta\log K + \lambda_L = 0 \qquad (21)$$

$$\begin{aligned}P^*_{ls}(j) &= \exp(\frac{\alpha}{\beta}\log p(j))\cdot\exp(\frac{-\beta - \beta\log K - \lambda_L}{\beta}) \\ &= \exp(\frac{\alpha}{\beta}\log p(j))\cdot C_{ls} \qquad (22)\end{aligned}$$

Since $\sum_{j'=1}^{K} P_{ls}(j') = 1$, we have

$$\sum_{j'=1}^{K} P_{ls}(j') = \sum_{j'=1}^{K}\exp(\frac{\alpha}{\beta}\log p(j'))\cdot\exp(\frac{-\beta - \beta\log K - \lambda_L}{\beta}) = 1 \qquad (23)$$

$$C_{ls} = \exp(\frac{-\beta - \beta\log K - \lambda_L}{\beta}) = \frac{1}{\sum_{j'=1}^{K}\exp(\frac{\alpha}{\beta}\log p(j'))} \qquad (24)$$

So the optimal $P^*_{ls}(j)$ is given by the formula:

$$P^*_{ls}(j) = \frac{\exp(\frac{\alpha}{\beta}\log p(j))}{\sum_{j'=1}^{K}\exp(\frac{\alpha}{\beta}\log p(j'))} = \frac{p(j)^{\alpha/\beta}}{\sum_{j'=1}^{K} p(j')^{\alpha/\beta}}. \qquad (25)$$

□

B Derivation from optimal smoothing to softmax output with temperature

When $\tau = \frac{\beta}{\alpha}$, we could have

$$\begin{aligned}P^*_{ls}(c|x_i) &= \frac{p_\theta(c|x_i)^{\frac{1}{\tau}}}{\sum_j p_\theta(j|x_i)^{\frac{1}{\tau}}} = \frac{(\frac{e^{z_{i,c}}}{\sum_m e^{z_{i,m}}})^{\frac{1}{\tau}}}{\sum_j(\frac{e^{z_{i,j}}}{\sum_m e^{z_{i,m}}})^{\frac{1}{\tau}}} \\ &= \frac{(e^{z_{i,c}})^{\frac{1}{\tau}}}{\sum_j(e^{z_{i,j}})^{\frac{1}{\tau}}} = \frac{e^{\frac{z_{i,c}}{\tau}}}{\sum_j e^{\frac{z_{i,j}}{\tau}}} \qquad (26)\end{aligned}$$

C Additional Results on ImageNet

Setup for ImageNet experiments. We evaluated our method on two model architectures including ResNet50 and ResNet152 with standard data augmentation schemes including random resize cropping and random horizontal flip. The models are trained for 90 epochs with a batch size of 256. We use SGD for optimization with a momentum of 0.9, and weight decay set to 1e-4. The learning rate starts at 0.1 and is then divided by 10 at epochs 30, 60 and 80.

Table 5: Comparison between different Smoothed labels methods. Validation accuracy and training time are reported. The training time is measured on 4 NVIDIA V100 GPUs.

	ImageNet (Acc. & Time)			
	ResNet50		ResNet152	
Base	75.81	1.0×	77.92	1.0×
LS	76.17 (↑0.36)	1.0×	78.33 (↑0.41)	1.0×
TF-reg	76.21 (↑0.40)	1.1×	78.12 (↑0.20)	1.1×
Beta-LS	76.13 (↑0.32)	1.5×	78.56 (↑0.64)	1.6×
KR-LS	76.32 (↑0.51)	1.3×	78.48 (↑0.56)	1.3×
LABO	**76.55** (↑0.74)	1.1×	**78.62** (↑0.70)	1.1×

Results. Tab. 5 shows the accuracy and training time for one epoch of different methods on two models. First, our method can consistently improve the classification performance, which indicates its robustness on the large-scale dataset. Again, our method achieves better performance with compared with other smoothing functions with a moderate training time increase over Base. Our computation overhead for smoothing distribution is introduced by computing Eq. 1 which is deterministic and excluded from the computation graph for the gradient calculation, hence, our method is more efficient than other latest advanced LS techniques.

D Data Statistics

Dataset	Train	Validation	Test
IWSLT'14 (DE-EN)	160,239	7,283	6,750
IWSLT'14 (FR-EN)	168,151	7,643	4,493
WMT'14 (EN-DE)	3,900,502	39,414	3,003
IWSLT'17 (DE-EN)	209,522	7,887	5,670
IWSLT'17 (FR-EN)	236,652	8,277	7,275
CIFAR10	45,000	5000	10,000
CIFAR100	45,000	5,000	10,000
ImageNet	1,281,167	50,000	100,000

E Experimental Details for MT

We conduct experiments by using the same hyperparameters for fair comparisons. Before training, we first apply BPE (Sennrich et al., 2016) to tokenize the corpus for each language pair. During the training, we set the label smoothing parameter to 0.1. We follow previous work to use Adam optimizer with betas to be (0.9,0.98) and the learning rate is 7e-4 for WMT and 5e-4 for the rest of tasks. During warming-up steps, the initial learning rate is 1e-7 and there are 1000 warm-up steps. For the warm-up steps of our smoothing, we use 10000 for WMT and 5000 for other tasks. Dropout rate is set to 0.3 and weight decay is set to 0.0001 for all experiments. We pick the checkpoint with the best performance on the validation set before inferring on the test set with beam size 5.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
section 7

□ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B □ Did you use or create scientific artifacts?

Not applicable. Left blank.

□ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
section 4

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
section 4

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
section 4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
section 4

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.