RW-KD: Sample-wise Loss Terms Re-Weighting for Knowledge Distillation

Peng Lu[1,3], **Abbas Ghaddar**[1], **Ahmad Rashid**[1], **Mehdi Rezagholizadeh**[1],
Ali Ghodsi[2], **Philippe Langlais**[3]
[1]Huawei Noah's Ark Lab
[2] Department of Statistics and Actuarial Science, University of Waterloo
[3] RALI/DIRO, Université de Montréal, Canada
{peng.lu1,abbas.ghaddar,ahmad.rashid,mehdi.rezagholizadeh}@huawei.com
ali.ghodsi@uwaterloo.ca, felipe@iro.umontreal.ca

Abstract

Knowledge Distillation (KD) is extensively used in Natural Language Processing to compress the pre-training and task-specific fine-tuning phases of large neural language models. A *student* model is trained to minimize a convex combination of the prediction loss over the labels and another over the *teacher* output. However, most existing works either fix the interpolating weight between the two losses apriori or vary the weight using heuristics. In this work, we propose a novel sample-wise loss weighting method, RW-KD. A meta-learner, simultaneously trained with the student, adaptively re-weights the two losses for each sample. We demonstrate, on 7 datasets of the GLUE benchmark, that RW-KD outperforms other loss re-weighting methods for KD.

1 Introduction

Knowledge Distillation (Ba and Caruana, 2014; Hinton et al., 2015) has proven highly effective for compressing a large-scale NLP model (Devlin et al., 2019; Radford et al., 2019), called *teacher* in KD terms, into a smaller one, the *student*. A key factor behind KD's success is the use of teacher output as soft labels for supervising the training of the student (Müller et al., 2019; Yuan et al., 2020). The latter model is trained by jointly minimizing the losses on both hard and soft labels. The contribution of each loss term is conventionally controlled by a balancing hyperparameter.

However, recent studies suggested that hard and soft label importance is sample-wise (Tang et al., 2020; Zhou et al., 2021), and only a subset of training samples are crucial for distillation (Li et al., 2018; Zhang et al., 2021). For instance, teacher outputs may be of poor quality for some samples (Ghaddar et al., 2021a,b), but highly informative for others (Cho and Kang, 2020). Also, researchers have found that adjusting loss weights during training greatly benefits performance of KD (Clark et al., 2019; Mukherjee and Awadallah, 2020; Jafari et al., 2021). However, the contribution of loss terms is heuristically decayed by an annealing factor, yet another hyperparameter.

We argue that using the same weights for all training samples, referred to in our work as single-weight, prevents exploiting the full advantage of KD, because each data sample might have different optimal weights for the loss terms. We propose a meta-learning approach to learn sample-wise weights of loss terms. We revisit *learning to weight* approaches (Ren et al., 2018; Shu et al., 2019), initially proposed for noisy sample down-weighting, and adapt it for loss terms weighting in KD.

Experimental results show that our KD loss weighting scheme consistently outperforms its counterparts on 7 tasks from the GLUE benchmark (Wang et al., 2019). A fine-grained analysis of the learned weights shows that, compared to the baselines, our meta-learner explores a greater range of KD weights to find the sample-wise optimal values.

2 Related Work

In recent years, Knowledge Distillation for BERT-like models (Devlin et al., 2019; Liu et al., 2019) has been extensively studied, leveraging intermediate layer matching (Ji et al., 2021; Wu et al., 2020; Passban et al., 2021), data augmentation (Fu et al., 2020; Jiao et al., 2020; Rashid et al., 2021; Kamalloo et al., 2021), adversarial training (Zaharia et al., 2021; Rashid et al., 2020, 2021), and lately loss terms re-weighting (Clark et al., 2019; Zhou et al., 2021; Jafari et al., 2021). In this work, we explore the latter direction with a meta learning approach (Li et al., 2019; Fan et al., 2020).

Learning to weight approaches (Ren et al., 2018; Zhang et al., 2020) were mainly proposed to learn per-sample loss weights in order to discount

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 3145–3152
November 7–11, 2021. ©2021 Association for Computational Linguistics

noisy samples thanks to an auxiliary meta-learner which re-weights training samples of the main model. Such approaches often train a meta-learner on a clean validation set, or on small-loss training samples if no clean data is available. The meta-learner architecture varies from a simple multi-layer perceptron (MLP) as in Meta-Weight-Net (Shu et al., 2019) to LSTM-based encoder as in MentorNet (Jiang et al., 2018).

The work of Jin et al. (2021) on multi-modal model compression with KD is the most similar to ours. The authors train a MLP meta learner (Shu et al., 2019), on the validation set, which assigns sample-level weights for 3 loss terms that are calculated when text, image, and both modalities are given as input. In our work, we use a transformer-based meta learner to estimate the sample-wise optimal weights for KD with gradient similarity (see Section 3.2).

3 Methodology

Let $T(\cdot)$ be a fine-tuned fixed teacher, and $S_\theta(\cdot)$ the student model parameterized with θ. 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 respective label, vanilla KD (Hinton et al., 2015) consists of minimizing a weighted combination of two different losses:

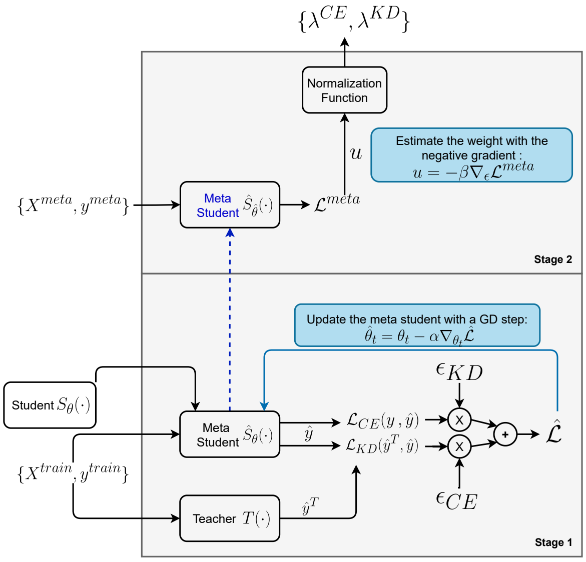
$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}[(1-\alpha)\cdot\mathcal{L}_{CE}(y_i, S_\theta(x_i)) \qquad (1)$$
$$+\alpha\cdot\mathcal{L}_{KD}(T(x_i), S_\theta(x_i))]$$

where $\mathcal{L}_{CE}$ is a cross-entropy (CE) loss on hard labels, and $\mathcal{L}_{KD}$ is the Kullback-Leibler divergence (Kullback, 1997) between teacher and student logits. $\alpha \in [0, 1]$ is a hyperparameter controlling the contribution of both losses. For simplicity, we refer to $\mathcal{L}_{CE}(y_i, S_\theta(x_i))$ as $\mathcal{L}_{CE}(x_i)$ and $\mathcal{L}_{KD}(T(x_i), S_\theta(x_i))$ as $\mathcal{L}_{KD}(x_i)$ hereafter.

Reweighting KD We propose a sample-wise reweighting method for KD to learn a balance between the CE and KD loss for every training sample. The new training loss is computed as follows:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}[\lambda_i^{CE}\cdot\mathcal{L}_{CE}(x_i) \qquad (2)$$
$$+\lambda_i^{KD}\cdot\mathcal{L}_{KD}(x_i)]$$

where $\lambda_i^{CE} + \lambda_i^{KD} = 1$.



$\{\lambda^{CE}, \lambda^{KD}\}$
Normalization Function
u
Estimate the weight with the negative gradient : $u = -\beta\nabla_\epsilon\mathcal{L}^{meta}$
$\{X^{meta}, y^{meta}\}$
Meta Student $\hat{S}_{\hat{\theta}}(\cdot)$
$\mathcal{L}^{meta}$
Stage 2
Update the meta student with a GD step: $\hat{\theta}_t = \theta_t - \alpha\nabla_{\theta_t}\hat{\mathcal{L}}$
ϵ_{KD}
Student $S_\theta(\cdot)$
Meta Student $\hat{S}_\theta(\cdot)$
$\hat{y}$
$\mathcal{L}_{CE}(y, \hat{y})$
$\mathcal{L}_{KD}(\hat{y}^T, \hat{y})$
$\hat{\mathcal{L}}$
$\{X^{train}, y^{train}\}$
Teacher $T(\cdot)$
$\hat{y}^T$
ϵ_{CE}
Stage 1

Figure 1: Meta-reweight Module. In Stage 1, the parameter θ of the meta student is updated to be a function of ϵ. In Stage 2, the optimal weights $\{\lambda^{CE}, \lambda^{KD}\}$ are estimated with the negative gradient of $\mathcal{L}^{meta}$ w.r.t ϵ.

Finding the optimal weights for each loss is intractable. Our solution is inspired by Koh and Liang (2017) and Ren et al. (2018). These works investigate which training samples are most responsible for the generalization performance. We follow this line of works and perturb different losses in KD training to identify which loss is more influential and informative.

3.1 Meta-reweight KD

We define our problem as a meta-learning one and use the validation set to define a meta-learning loss function. Our meta-reweight module is depicted in Figure 1.

Meta-objective. The optimal selection of $\lambda = \{\lambda_i^{CE}, \lambda_i^{KD}\}|_{i=1}^{N}$ is derived from its performance on the meta dataset of M samples[1]:

$$\lambda^* = \arg\min_{\lambda \geqslant 0} \frac{1}{M}\sum_{j=1}^{M}\mathcal{L}^{meta}(\theta^*(\lambda)) \qquad (3)$$

where $\mathcal{L}^{meta}$ is the loss computed on samples from the meta dataset. Since computing the optimal λ^* and θ^* need two nested optimization loops, we adopt an online strategy to estimate λ and update θ respectively.

[1] We consider the validation dataset as the meta dataset.

Meta-reweighting In order to derive the optimal weights on two different losses for each sample before updating the student, we use a meta model to compute the weight taking a gradient step on the meta loss. First, we initialize a meta-student $\hat{S}_\theta(\cdot)$ with the same parameters of the student model $S_\theta(\cdot)$ at the beginning of every iteration.

Next, we feed a mini-batch of n training samples $X^{(n)} = \{x_i, y_i\}|_{i=1}^n$ to the meta-student and compute the CE and KD losses, then perturb their weights by ϵ_i^{CE} and ϵ_i^{KD} respectively for each example and calculate the weighted loss:

$$\hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon) = \sum_{i=1}^{n} \epsilon_i^{CE} \mathcal{L}_{CE}(x_i) + \epsilon_i^{KD} \mathcal{L}_{KD}(x_i) \tag{4}$$

where $\epsilon = \{\epsilon_i^{CE}, \epsilon_i^{KD}\}|_{i=1}^n$ is the collection of all perturbations. We then take a gradient step update on the current parameter θ_t:

$$\hat{\theta}_t = \theta_t - \alpha \nabla_{\theta_t} \hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon) \tag{5}$$

where α is the step size of the gradient descent. Next, we feed a mini-batch of meta examples $X^{(m)} = \{x_j, y_j\}|_{j=1}^m$ to the meta-student $\hat{S}_{\hat{\theta}_t}(\cdot)$ and compute the meta loss $\mathcal{L}^{meta}(X^{(m)}; \hat{\theta}_t)$ as:

$$\frac{1}{m} \sum_{j=1}^{m} [\mathcal{L}_{CE}(x_j) + \mathcal{L}_{KD}(x_j)] \tag{6}$$

Since the parameter $\hat{\theta}_t$ of the meta-student becomes a function of ϵ as $\nabla_{\theta_t} \hat{\mathcal{L}}$ is a function of ϵ, we can directly compute the gradient of meta loss w.r.t ϵ via the chain rule, which is implemented in practice by automatic differentiation of deep learning frameworks such as Pytorch (Paszke et al., 2019). Here we take the negative gradients as the estimation of weights:

$$u_i^{CE} = -\beta \frac{\partial}{\partial \epsilon_i^{CE}} \mathcal{L}^{meta}(X^{(m)}; \hat{\theta}_t) \tag{7}$$

$$u_i^{KD} = -\beta \frac{\partial}{\partial \epsilon_i^{KD}} \mathcal{L}^{meta}(X^{(m)}; \hat{\theta}_t) \tag{8}$$

where β is a scaling factor. We then normalize the weights $\{u_i^{CE}, u_i^{KD}\}$ for each training sample x_i to make them positive and ensure they sum to 1, leading to:

$$\lambda_i^{CE} = \frac{\max(u_i^{CE}, \delta)}{\max(u_i^{CE}, \delta) + \max(u_i^{KD}, \delta)} \tag{9}$$

$$\lambda_i^{KD} = \frac{\max(u_i^{KD}, \delta)}{\max(u_i^{CE}, \delta) + \max(u_i^{KD}, \delta)} \tag{10}$$

where δ = 1e-8 is a hyperparameter for helping training stability. In the end, we compute the final loss with locally optimal weights for the two losses for each sample in the training mini-batch and update our student model $S_\theta(\cdot)$.

The weight is estimated by computing gradients of meta loss w.r.t the *perturbation* on different losses and these gradients can indicate the sensitivity of the meta loss when we perturb each loss used for training. By using these gradients as the weight of different losses, we can adjust the impact of different losses towards better performance on the predefined meta-dataset. The detailed pseudo-code is presented in Algorithm 1.

Algorithm 1: Knowledge Distillation with Meta-reweighting

```
input  : $D_{train}$, $D_{meta}$, $S_\theta(\cdot)$, $T(\cdot)$
$S_\theta(\cdot)$ initialization;
for $i \leftarrow 1$ to $N\_epoch$ do
    for $t \leftarrow 1$ to $T$ do
        // Meta-reweighting
        $\hat{S}_{\theta_t}(\cdot) \leftarrow S_{\theta_t}(\cdot)$;
        $\{X_f, y_f\} \leftarrow$ MiniBatch($D_{train}, n$);
        $\hat{y}_f,\ \hat{y}_f^{\mathrm{T}} \leftarrow \hat{S}_{\theta_t}(X_f),\ T(X_f)$;
        $\{\epsilon_i^{CE}, \epsilon_i^{KD}\}|_{i=1}^n \leftarrow 0$;
        $\hat{\mathcal{L}}_i \leftarrow \epsilon_i^{\mathrm{CE}} \mathcal{L}_{CE}(y_{f,i}, \hat{y}_{f,i}) + \epsilon_i^{\mathrm{KD}} \mathcal{L}_{KD}(\hat{y}_{f,i}^{\mathrm{T}}, \hat{y}_{f,i})$;
        $\hat{\theta}_t \leftarrow \theta_t - \alpha \nabla_{\theta_t} \sum_{i=1}^{n} \hat{\mathcal{L}}_i$;
        $\{X_g, y_g\} \leftarrow$ MiniBatch($D_{meta}, m$) ;
        $\hat{y}_g,\ \hat{y}_g^{\mathrm{T}} \leftarrow \hat{S}_{\hat{\theta}_t}(X_g),\ T(X_g)$;
        $\mathcal{L}_i^{meta} \leftarrow \mathcal{L}_{CE}(y_{g,i}, \hat{y}_{g,i}) + \mathcal{L}_{KD}(\hat{y}_{g,i}^{\mathrm{T}}, \hat{y}_{g,i})$;
        $\nabla\epsilon \leftarrow -\beta \cdot \nabla_\epsilon \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_i^{meta}$;
        $\lambda_i^{CE} \leftarrow \frac{\max(\nabla\epsilon_i^{CE}, \delta)}{\max(\nabla\epsilon_i^{CE}, \delta) + \max(\nabla\epsilon_i^{KD}, \delta)}$;
        $\lambda_i^{KD} \leftarrow \frac{\max(\nabla\epsilon_i^{KD}, \delta)}{\max(\nabla\epsilon_i^{CE}, \delta) + \max(\nabla\epsilon_i^{KD}, \delta)}$;
        // Knowledge-Distillation
        $\mathcal{L} \leftarrow \lambda_i^{CE} \mathcal{L}_{CE}(y_{f,i}, \hat{y}_{f,i}) + \lambda_i^{KD} \mathcal{L}_{KD}(\hat{y}_{f,i}^{\mathrm{T}}, \hat{y}_{f,i})$;
        $\theta_{t+1} \leftarrow \theta_t - \nabla_{\theta_t} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}$;
```

Model	CoLA	SST-2	MRPC	RTE	QNLI	QQP	MNLI-m/mm	Avg.
BERT-base	51.6	92.9	87.8	65.5	89.9	71.3	83.5/82.1	78.1
w/o KD	49.1	91.4	86.0	57.6	87.9	67.7	80.1/79.6	74.9
Vanilla-KD	49.4	91.2	86.3	58.4	88.2	68.6	80.4/79.5	75.2
ANL-KD	49.1	91.2	86.6	59.1	88.1	68.4	80.9/79.7	75.4
WLS-KD	50.0	91.8	87.0	59.6	88.3	68.9	81.8/80.2	75.9
RW-KD (our)	**50.5**	**92.5**	**87.2**	**60.5**	**88.5**	**69.5**	**82.1/80.8**	**76.5**

Table 1: Performance of the teacher and students with different loss re-weighting methods on GLUE test sets.

3.2 Weight Estimation via Gradient Similarity

Next, we show the relation between the weight estimation and the gradient similarity. To save space, we omit u_i^{KD}. The weight on the CE loss of i-th example is the similarity between the gradient of the i-th example on CE loss and the average gradient of mini-batch of the meta data computed for the meta loss at time step t. The computation of Eq 7 by backpropagation can be rewritten as [2]:

$$u_i^{CE} = \alpha\beta \cdot \langle \mathbf{J}_1, \mathbf{J}_2 \rangle \tag{11}$$

where $\mathbf{J}_1$ is the Jacobian vector of $\mathcal{L}^{meta}$ w.r.t $\hat{\theta}$ which indicates the direction of decrease in loss on a mini-batch of meta data, and $\mathbf{J}_2$ is the Jacobian vector of $\mathcal{L}_{CE}$ of i-th sample w.r.t θ which indicates the direction of decrease of the CE loss of i-th sample. Larger weights mean that moving along the $\mathbf{J}_2$ direction is likely to not only reduce the training loss, but also reduce the meta loss.

4 Experiments

4.1 Dataset and Evaluation

We run experiments on 7 tasks from the GLUE benchmark (Wang et al., 2019): 2 single-sentence (CoLA and SST-2) and 5 sentence-pair (MRPC, RTE, QQP, QNLI, QQP, and MNLI) classification tasks. Following prior works, we report Matthews correlation on CoLA, F1 score on MRPC and QQP, and accuracy for the other tasks on their corresponding test sets.

4.2 Baselines

We compare RW-KD to 4 losses re-weighting methods:

- **w/o KD** In this setting, the KL loss weight (α) is always set to zero.
- **Vanilla-KD** Here, we select the best performing α value for each task.
- **ANL-KD** In Annealing KD (Clark et al., 2019), α is gradually decreased from 1 to 0.

Finally, we consider the recent **WLS-KD** (Zhou et al., 2021) dynamic re-weighting method, where α is calculated as follow:

$$\alpha = 1 - \exp\left(-\frac{L^s_{\text{ce}}}{L^t_{\text{ce}}}\right) \tag{12}$$

where L^s_{ce} and L^t_{ce} are loss values on the hard label for the student and teacher respectively.

4.3 Implementation

All models use a 12-layer BERT-base-uncased model (Devlin et al., 2019) as teacher, and the pretrained 6-layer distillBERT (Sanh et al., 2019) as initialization for the students. We perform hyperparameter tuning, and select best performing models using early stopping on dev sets.

4.4 Results

Table 1 shows the performances of the teacher, baselines, and our method on the GLUE test sets. First, we notice that ANL-KD fails to perform as we expected (only 0.2% gain on top of Vanilla-KD), although we extensively tested different α decay schedules.

It is worth mentioning that this approach was successful in multi-task KD when the teacher and the student are of same size. Second, we observe that RW-KD outperforms single-weight weighting schemes (Vanilla and ANL), and sample-wise WLS-KD method by 1.3%,1.1% and 0.6% respectively on all tasks. We plot the weights learned by the meta-learner to better understand why RW-KD performs better. Figure 2 shows the distribution of

[2] Derivation can be found in Appendix A.

training sample weights on 4 GLUE tasks for WLS-KD and RW-KD. Similar figures are observed on the remaining 3 tasks.

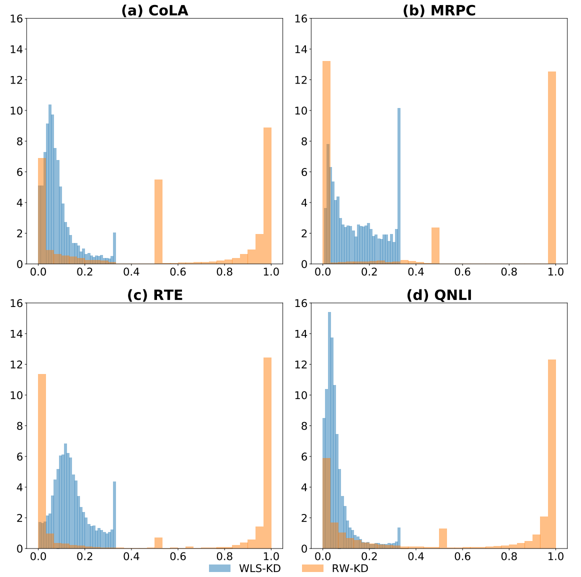


Figure 2: KL loss weight distributions of WLS-KD (blue) and RW-KD (orange) on 4 GLUE tasks. x-axis indicates the weight values, and y-axis shows percentage of samples.

On one hand, we observe that the majority of WLS weights are concentrated below 0.3 and that the best α values were around 0.5 for Vanilla KD. On the other hand, we observe that our meta-learner mostly produces weights with either very high or very low values, and less frequently weights around 0.5 (e.g. CoLA and RTE). Interestingly, this suggests that for many samples, either one of the hard or soft label loss is informative for the student. Consequently, a sample-wise loss weighting method seems a key component of KD.

5 Conclusion

In this paper, we show the importance of sample-wise loss term weighting in Knowledge Distillation and propose RW-KD a method which does this and leads to better distillation performance on 7 GLUE tasks. Future work involves combining RW-KD with state of the art KD methods that use extra loss terms such as intermediate layer similarity (Sanh et al., 2019; Jiao et al., 2020), attention matching (Sun et al., 2020; Wang et al., 2021), and adversarial (Rashid et al., 2021) losses. We expect that these methods can take full advantage of RW-KD, since they use single-weight loss terms weights. In addition to KD training, we will investigate applying our reweighting method to Multi-task Learning (MTL) scenarios (Caruana, 1997; Lu et al., 2019; Stickland and Murray, 2019), where learning to balance losses from different tasks is critical to benefit all tasks involved.

Acknowledgments

We thank Mindspore,[3] which is a new deep learning computing framework, for partial support of this work.

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2654–2662.

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

Ikhyun Cho and U Kang. 2020. Pea-kd: Parameter-efficient and accurate knowledge distillation. *CoRR*, abs/2009.14822.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. 2019. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5931–5937. Association for Computational Linguistics.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Yang Fan, Yingce Xia, Lijun Wu, Shufang Xie, Weiqing Liu, Jiang Bian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. 2020. Learning to teach with deep interactions. *CoRR*, abs/2007.04649.

Jie Fu, Xue Geng, Zhijian Duan, Bohan Zhuang, Xingdi Yuan, Adam Trischler, Jie Lin, Chris Pal, and Hao Dong. 2020. Role-wise data augmentation for knowledge distillation. *CoRR*, abs/2004.08861.

Abbas Ghaddar, Philippe Langlais, Ahmad Rashid, and Mehdi Rezagholizadeh. 2021a. Context-aware adversarial training for name regularity bias in named entity recognition. *Trans. Assoc. Comput. Linguistics*, 9:586–604.

[3] https://www.mindspore.cn/

Abbas Ghaddar, Philippe Langlais, Mehdi Rezagholizadeh, and Ahmad Rashid. 2021b. End-to-end self-debiasing framework for robust NLU training. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1923–1929. Association for Computational Linguistics.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 2493–2504. Association for Computational Linguistics.

Mingi Ji, Byeongho Heo, and Sungrae Park. 2021. Show, attend and distill: Knowledge distillation via attention-based feature matching. 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 7945–7952. AAAI Press.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2309–2318. PMLR.

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 *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4163–4174. Association for Computational Linguistics.

Woojeong Jin, Maziar Sanjabi, Shaoliang Nie, Liang Tan, Xiang Ren, and Hamed Firooz. 2021. Modality-specific distillation. *CoRR*, abs/2101.01881.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via minimax. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3522–3533. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. 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 1885–1894. PMLR.

Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S. Kankanhalli. 2019. Learning to learn from noisy labeled data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5051–5059. Computer Vision Foundation / IEEE.

Minghan Li, Tanli Zuo, Ruicheng Li, Martha White, and Weishi Zheng. 2018. Accelerating large scale knowledge distillation via dynamic importance sampling. *CoRR*, abs/1812.00914.

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

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.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Xtremedistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2221–2234. 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.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: attention-based layer projection for knowledge distillation. 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 13657–13665. AAAI Press.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

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 8024–8035.

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

Ahmad Rashid, Vasileios Lioutas, Abbas Ghaddar, and Mehdi Rezagholizadeh. 2020. Towards zero-shot knowledge distillation for natural language processing. *CoRR*, abs/2012.15495.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. MATE-KD: masked adversarial text, a companion to knowledge distillation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1062–1071. Association for Computational Linguistics.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4331–4340. PMLR.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1917–1928.

Asa Cooper Stickland and Iain Murray. 2019. BERT and pals: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2158–2170. Association for Computational Linguistics.

Jiaxi Tang, Rakesh Shivanna, Zhe Zhao, Dong Lin, Anima Singh, Ed H. Chi, and Sagar Jain. 2020. Understanding and improving knowledge distillation. *CoRR*, abs/2002.03532.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2140–2151. Association for Computational Linguistics.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizadeh, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1016–1021. Association for Computational Linguistics.

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.

George-Eduard Zaharia, Andrei-Marius Avram, Dumitru-Clementin Cercel, and Traian Rebedea. 2021. Dialect identification through adversarial learning and knowledge distillation on romanian BERT. In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects, VarDial@EACL 2021, Kiyv, Ukraine, April 20, 2021*, pages 113–119. Association for Computational Linguistics.

Haoran Zhang, Zhenzhen Hu, Wei Qin, Mingliang Xu, and Meng Wang. 2021. Adversarial co-distillation learning for image recognition. *Pattern Recognit.*, 111:107659.

Zizhao Zhang, Han Zhang, Sercan Ömer Arik, Honglak Lee, and Tomas Pfister. 2020. Distilling effective supervision from severe label noise. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9291–9300. Computer Vision Foundation / IEEE.

Helong Zhou, Liangchen Song, Jiajie Chen, Ye Zhou, Guoli Wang, Junsong Yuan, and Qian Zhang. 2021. Rethinking soft labels for knowledge distillation: A bias-variance tradeoff perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

A Derivation of the gradients

We show how to compute the gradient of meta loss w.r.t ϵ_i^{CE} at time step t:

$$\begin{aligned} u_i^{CE} =& -\beta \frac{\partial}{\partial \epsilon_i^{CE}} \mathcal{L}^{meta}(X^{(m)}; \hat{\theta}_t) \\ =& -\beta \sum_{j=1}^{K} \frac{\partial \hat{\theta}_{t,j}}{\epsilon_i^{CE}} \frac{\partial \mathcal{L}^{meta}}{\partial \hat{\theta}_{t,j}} \end{aligned}$$

where K is the number of parameters of the student. Since $\hat{\theta}_t$ is a function of ϵ:

$$\hat{\theta}_t = \theta_t - \alpha \nabla_{\theta_t} \hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon)$$

We continue to expand the middle part $\frac{\partial \hat{\theta}_{t,j}}{\partial \epsilon_i^{CE}}$:

$$\begin{aligned} \frac{\partial \hat{\theta}_{t,j}}{\partial \epsilon_i^{CE}} =& \frac{\partial(\theta_{t,j} - \alpha \frac{\partial \hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon)}{\partial \theta_{t,j}})}{\partial \epsilon_i^{CE}} \\ =& -\alpha \frac{\partial(\frac{\partial \hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon)}{\partial \theta_{t,j}})}{\partial \epsilon_i^{CE}} \end{aligned}$$

and we have

$$\hat{\mathcal{L}}(X^{(n)}; \theta_t, \epsilon) = \sum_{i=1}^{n} \epsilon_i^{CE} \mathcal{L}_{CE}(x_i) + \epsilon_i^{KD} \mathcal{L}_{KD}(x_i)$$

Then we can continue to expand:

$$\begin{aligned} =& -\alpha \cdot \frac{\partial(\frac{\partial \sum_{i=1}^{n} \epsilon_i^{CE} \cdot \mathcal{L}_{CE}(x_i) + \epsilon_i^{KD} \cdot \mathcal{L}_{KD}(x_i)}{\partial \theta_{t,j}})}{\partial \epsilon_i^{CE}} \\ =& -\alpha \cdot \frac{\partial(\frac{\partial \epsilon_1^{CE} \cdot \mathcal{L}_{CE}(x_1)}{\partial \theta_{t,j}} + \cdots + \frac{\partial \epsilon_n^{CE} \cdot \mathcal{L}_{CE}(x_n)}{\partial \theta_{t,j}})}{\partial \epsilon_i^{CE}} \\ =& -\alpha \cdot \frac{\partial \mathcal{L}_{CE}(x_i)}{\partial \theta_{t,j}} \end{aligned}$$

Therefore, the local optimal weight u_i^{CE} represents the similarity between the two Jacobian vectors .

$$\begin{aligned} u_i^{CE} =& \alpha\beta \sum_{j=1}^{K} \frac{\partial \mathcal{L}^{meta}}{\partial \hat{\theta}_{t,j}} \frac{\partial \mathcal{L}_{CE}(x_i)}{\partial \theta_{t,j}} \\ =& \alpha\beta \cdot \langle \mathbf{J}_1, \mathbf{J}_2 \rangle \end{aligned}$$

where $\mathbf{J}_1 = [\frac{\partial \mathcal{L}^{meta}}{\partial \hat{\theta}_{t,1}}, \cdots, \frac{\partial \mathcal{L}^{meta}}{\partial \hat{\theta}_{t,K}}]^T$ is the Jacobian vector of $\mathcal{L}^{meta}$ w.r.t $\hat{\theta}$ on a mini-batch of meta data, $\mathbf{J}_2 = [\frac{\partial \mathcal{L}_{CE}(x_i)}{\partial \theta_{t,1}}, \cdots, \frac{\partial \mathcal{L}_{CE}(x_i)}{\partial \theta_{t,K}}]^T$ is the Jacobian vector of $\mathcal{L}_{CE}$ w.r.t θ of the i-th training sample.