Gradient Knowledge Distillation for Pre-trained Language Models

Lean Wang[1]**, Lei Li**[2]**, Xu Sun**[1,2]
[1]School of Electronics Engineering and Computer Science, Peking University
[2]MOE Key Lab of Computational Linguistics, School of Computer Science, Peking University
{lean, xusun}@pku.edu.cn lilei@stu.pku.edu.cn

Abstract

Knowledge distillation (KD) is an effective framework to transfer knowledge from a large-scale teacher to a compact yet well-performing student. Previous KD practices for pre-trained language models mainly transfer knowledge by aligning instance-wise outputs between the teacher and student, while neglecting an important knowledge source, i.e., the gradient of the teacher. The gradient characterizes how the teacher responds to changes in inputs, which we assume is beneficial for the student to better approximate the underlying mapping function of the teacher. Therefore, we propose *Gradient Knowledge Distillation* (GKD) to incorporate the gradient alignment objective into the distillation process. Experimental results show that GKD outperforms previous KD methods regarding student performance. Further analysis shows that incorporating gradient knowledge makes the student behave more consistently with the teacher, improving the interpretability greatly.[1]

1 Introduction

Knowledge distillation (KD) (Hinton et al., 2015; Romero et al., 2015) is a classic framework for model compression, which trains a compact student model by utilizing the learned knowledge in a large teacher PLM via teacher-student prediction alignments. Various alignment strategies have been proposed, like internal representation matching (Sun et al., 2019; Sanh et al., 2019; Jiao et al., 2020) and attention heatmap consistency (Wang et al., 2020), and obtain promising efficiency-performance trade-off. However, previous KD studies for PLMs mostly align the student to the teacher model in a point-to-point manner, neglecting an important knowledge source, i.e., the gradient of the teacher.

Viewing a model as a function mapping the input to the label space, the gradients thus can depict the curve of the function, capturing the model activation changes according to the input perturbation, which we assume can be beneficial for the student model. Besides, as gradients are closely related to interpretation, gradient alignment is likely to improve interpretation consistency.

Motivated by this, we explore incorporating gradient alignment in knowledge distillation. However, introducing gradient alignment to KD for PLMS is challenging due to the following two reasons. First, the inherent discreteness of natural language makes directly computing the deviations w.r.t. the input sentences intractable. To remedy this, we instead align the gradient w.r.t the input embedding, and initialize the embedding of the student with that of the teacher and freeze it during training. Second, our theoretical analysis shows that the commonly adopted Dropout regularization (Srivastava et al., 2014) will disturb the gradient alignment, so we deactivate dropout while conducting distillation. Further analysis and experiments can be found in Section 3 and Table 1. Experiments on large-scale datasets in the GLUE benchmark (Wang et al., 2019) demonstrate that gradient knowledge improves the student's performance and behavior consistency with the teacher model. Besides, our analysis of

[1] Our code is available at https://github.com/lancopku/GKD.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

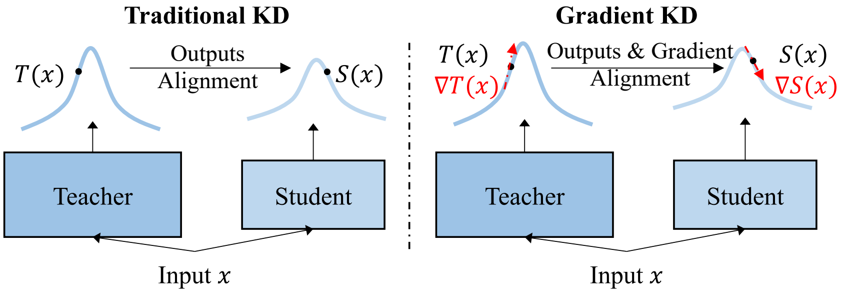

Figure 1: Comparison between traditional KD and our gradient KD. Gradient KD introduces an extra gradient alignment objective to better inform the student of the model behavior of the teacher.

word saliency (Ding et al., 2019) shows that the alignments on gradients can benefit the behavior consistency in word saliency level.

2 Related Work

Knowledge distillation has been widely used in computer vision (CV) (Kong et al., 2019; Mullapudi et al., 2019), natural language process (NLP) (Sanh et al., 2019; Chen et al., 2020) and multimodal field (Dai et al., 2022; Li et al., 2021a). In the NLP field, knowledge distillation methods can be roughly classified into one-stage methods and two-stage ones. One-stage methods perform distillation at the fine-tuning stage. BiLSTM$_{\text{SOFT}}$ (Tang et al., 2019) performs knowledge distillation with the teacher's logits on an augmented dataset, BERT-PKD (Sun et al., 2019) performs knowledge distillation with the teacher's logits and hidden states, and PD (Turc et al., 2019) uses a small pre-trained masked language model as a student to enhance the effect of distillation. Two-stage methods perform distillation at both the pre-training stage and the fine-tuning stage. DistilBERT (Sanh et al., 2019) and MobileBERT (Sun et al., 2020) focus on the pre-training stage, aiming to get a task-agnostic model that can be fine-tuned or distilled on downstream tasks. TinyBERT (Jiao et al., 2020) first distills a task-agnostic model during pre-training and then performs task-specific distillation on an augmented dataset to further improve performance.

Besides, in the computer vision field, aligning gradients (Srinivas and Fleuret, 2018) or using gradients as a weighting factor (Zhu and Wang, 2021) have been explored before. But there is little exploration for PLMs. Wu et al. incorporates gradients via a momentum distillation method which is similar to the momentum mechanism (Qian, 1999), but as far as we know, direct gradient alignment has not been explored in the context of KD for PLMs.

3 Gradient Knowledge Distillation For PLMs

Backgrounds: KD for PLMs Formally, for a classification task, we denote $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$ as the training dataset with N instances, where $\mathbf{x}_i$ is the input sentence and $\mathbf{y}_i$ is the label. Without loss of generality, we take BERT as the representative of PLM model, which transforms the sentence $\mathbf{x}_i$ to a contextualized representation $\mathbf{h}_i = \text{BERT}(\mathbf{x}_i)$. A softmax layer with a learnable parameter $\mathbf{W}$ is appended for producing probability vector $\mathbf{p}_i = \text{softmax}(\mathbf{W}\mathbf{h}_i)$ over the label set. We denote $\mathbf{p}_i^T$ and $\mathbf{p}_i^S$ as the probability vector generated by the teacher T and the student S for the ith input, respectively. KD aims to transform the large teacher's knowledge into a smaller student model. Vanilla KD (Hinton et al., 2015) achieves it by utilizing the soft targets produced by the teacher:

$$\mathcal{L}_{\text{KD}} = (1-\alpha)\mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{Soft-CE}} \tag{1}$$

$$\mathcal{L}_{\text{Soft-CE}} = \tau^2 \sum_{i=1}^{N} D_{\text{KL}}\left(\mathbf{p}_{i,\tau}^{T} \| \mathbf{p}_{i,\tau}^{S}\right), \tag{2}$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy objective for classification tasks, $\mathbf{p}_{i,\tau} = \text{softmax}(\mathbf{W}\mathbf{h}_i/\tau)$, α is a hyper-parameter and τ is the temperature hyper-parameter. Further explorations introduce alignments on hidden states (Sun et al., 2019; Sanh et al., 2019) or attention heatmap (Wang et al., 2020), and explore dynamic learning schedule (Li et al., 2021b).

Improving KD with Gradient Alignments Different from previous KD studies which consider aligning the teacher and the student in a point-wise manner, we propose to align the change of the model around the inputs by introducing an extra objective on gradient consistency between the student and the teacher. In this way, the student can better understand how the output should change when the input changes, which is helpful for the student to behave similarly to the teacher. However, we find that the discreteness of the text and the dropout regularization hinder the alignment of the gradients. First, as the input sentence tokens are discrete, it is intractable to get the derivative w.r.t the original input sentence. To remedy this, we instead take the derivative w.r.t the input embeddings. Besides, to make sure that the input spaces of the student and the teacher are aligned for calculating the gradient, we initialize the student embedding with that of the teacher, which is a commonly adopted practice in previous KD studies (Sun et al., 2019; Jiao et al., 2020), and fix the student embedding layer during the training. Denote $p^S_{m,i}$ and $p^T_{m,i}$ as the maximum probability in the obtained class probability distribution from the student and the teacher for an input sentence $\mathbf{x}_i$, the gradient alignment is defined as a mean-square error objective between two normalized deviations:

$$\mathcal{L}_{\text{GKD}} = \sum_{i=1}^{N}\sum_{j=1}^{L_i} \| \frac{\frac{\partial p^S_{m,i}}{\partial \mathbf{E}^S_{i,j}}}{\|\frac{\partial p^S_{m,i}}{\partial \mathbf{E}^S_{i,j}}\|_2} - \frac{\frac{\partial p^T_{m,i}}{\partial \mathbf{E}^T_{i,j}}}{\|\frac{\partial p^T_{m,i}}{\partial \mathbf{E}^T_{i,j}}\|_2} \|_2^2 \,, \tag{3}$$

where L_i denotes the length of the ith tokenized sentence and $\mathbf{E}_{i,j}$ denotes the input embedding vector corresponding to the jth token of the ith sentence. As the gradient computation will take all the weights of the model into consideration, the objective encourages the weight updates towards making the student produce consistent gradients with the teacher model.

Second, we find the commonly adopted Dropout (Srivastava et al., 2014) will cause a biased gradient estimation. We provide a mathematical analysis in Theorem 1. Without loss of generality, in Theorem 1, we derive the gradient when inputs are applied with a dropout mask once.

Theorem 1 *Consider a function f whose input $\mathbf{x_0} \odot \boldsymbol{\xi}$ comes from a dropout layer, where each component of $\boldsymbol{\xi} \in \{0, \frac{1}{1-\delta}\}^d$ is drawn independently from a scaled Bernoulli($1-\delta$) random variable. Suppose f can be estimated by its second-order Taylor expansion around $\mathbf{x}_0$, then*

$$\mathbf{E}_{\boldsymbol{\xi}}[\nabla f(\mathbf{x}_0 \odot \boldsymbol{\xi})] = \nabla f(\mathbf{x}_0) + \frac{\delta}{1-\delta}\,\mathrm{diag}(\nabla^2 f(\mathbf{x}_0))\mathbf{x}_0 \,. \tag{4}$$

As the dropout in the teacher model is deactivated, we will align $\nabla T(\mathbf{x}_0)$ with $\nabla S(\mathbf{x}_0) + \frac{\delta}{1-\delta}\,\mathrm{diag}(\nabla^2 S(\mathbf{x}_0))\mathbf{x}_0$ when conducting distillation with dropout activated in the student model. Besides, this bias accumulates when the model goes deeper and leads to more deviated gradients for shallow layers, which is verified in Appendix B. To remedy this, we deactivate the dropout in the student model for a consistent gradient calculation.

With the deactivated dropout and the gradient alignment objective, our gradient knowledge distillation (GKD) is achieved by minimizing the combined objectives (α and β are hyper-parameters):

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{soft-CE}} + \beta\mathcal{L}_{\text{GKD}} \,, \tag{5}$$

Furthermore, we apply the gradient alignment on the `[CLS]` vector to enhance the effect, as `[CLS]` vectors are usually adopted as the sentence representation for sentence-level prediction:

$$\mathcal{L}_{\text{GKD-CLS}} = \sum_{i=1}^{N}\sum_{j=1}^{m} \| \frac{\frac{\partial p^S_{m,i}}{\partial \mathbf{h}^s_{i,S_j,\text{CLS}}}}{\|\frac{\partial p^S_{m,i}}{\partial \mathbf{h}^s_{i,S_j,\text{CLS}}}\|_2} - \frac{\frac{\partial p^T_{m,i}}{\partial \mathbf{h}^T_{i,T_j,\text{CLS}}}}{\|\frac{\partial p^T_{m,i}}{\partial \mathbf{h}^T_{i,T_j,\text{CLS}}}\|_2} \|_2^2 \,, \tag{6}$$

where $\mathbf{h}^t_{i,k,\text{CLS}}$ denotes to the hidden state of the `[CLS]` token for the ith example in the kth layer. The Skip layer mapping strategy is adopted for S_j and T_j.[2] As the `[CLS]` representations in the student and the teacher can be different, which may hinder gradient alignment, we incorporate an extra loss to align them:

$$\mathcal{L}_{\text{PKD}} = \sum_{i=1}^{N}\sum_{j=1}^{m} \| \frac{\mathbf{h}^S_{i,S_j,\text{CLS}}}{\|\mathbf{h}^S_{i,S_j,\text{CLS}}\|_2} - \frac{\mathbf{h}^T_{i,T_j,\text{CLS}}}{\|\mathbf{h}^T_{i,T_j,\text{CLS}}\|_2} \|_2^2 \,. \tag{7}$$

[2] For a 6-layer student, S_j takes value $\{1, 2, 3, 4, 5\}$, T_j takes value $\{2, 4, 6, 8, 10\}$ and $m = 5$.

Table 1: Results from the GLUE evaluation server with the best scores in bold. The metric for QQP is F1 score and others are accuracy. Avg. denotes the average score over SST-2, QQP, MNLI-m, MNLI-mm and QNLI.

Model	**SST-2**	**QQP**	**MNLI-m / mm**	**QNLI**	**Avg.**
BERT$_{\text{BASE}}$	93.7	71.5	84.8 / 83.8	91.3	85.0
Fine-tuning	91.0	69.6	81.3 / 79.5	87.3	81.7
Vanilla KD	92.0	71.0	82.2 / 81.2	88.9	83.1
BERT-PKD	91.8	71.0	82.4 / 81.8	89.1	83.2
GKD	92.0	71.5	**82.9** / 81.7	89.3	83.5
GKD-CLS	**93.0**	**71.6**	82.6 / **81.9**	**89.5**	**83.7**
w/ Dropout	91.8	71.0	82.3 / 81.6	89.1	83.2

Therefore, the objective of the enhanced version GKD-CLS is (α, β and γ are hyperparameters)

$$\mathcal{L} = (1-\alpha)L_{\text{CE}} + \alpha\mathcal{L}_{\text{soft-CE}} + \beta\mathcal{L}_{\text{PKD}} + \gamma(\mathcal{L}_{\text{GKD}} + \mathcal{L}_{\text{GKD-CLS}}). \tag{8}$$

4 Experiments

In this section, we compare our method with other methods from several aspects. We mainly focus on applying our distillation method in the fine-tuning stage due to the limited computational resources. Still, we do experiments to combine our method with Distilbert (Sanh et al., 2019), to demonstrate that our method can improve the performance of the models distilled in the pre-training stage (§ 4.3).

4.1 Experimental Settings

Datasets As fine-tuning the BERT model on small datasets like RTE (Bentivogli et al., 2009) and MRPC (Dolan and Brockett, 2005) can be quite unstable (Zhang et al., 2021; Mosbach et al., 2021), we select four large-scale sentiment classification and natural language inference datasets from the GLUE benchmark (Wang et al., 2019) for more stable evaluation, including Stanford Sentiment Treebank (SST-2) (Socher et al., 2013), Quora Question Pairs (QQP), Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2018) and Question-answering NLI (QNLI) (Rajpurkar et al., 2016).

Baselines We compare our method with direct fine-tuning and two task-specific KD methods. Vanilla KD (Hinton et al., 2015) aligns the logits of the student and the teacher. BERT-PKD (Sun et al., 2019) utilizes the hidden states of the `[CLS]` token as well as the logits. Besides, in the first part of Section 4.3, we combine our method with DistilBERT (Sanh et al., 2019), which performs distillation during pre-training.

Training Details We use a 12-layer BERT-base-uncased model as the teacher, and the students have the same architecture as the teacher but with 6 layers. We fine-tune the teacher on specific tasks to get the task-specific teacher. We conduct a hyper-parameter search for the baseline methods and our method in a way similar to Sun et al. (2019). Appendix C gives detailed experimental settings.

4.2 Main Results

Table 1 presents the test results obtained from the GLUE evaluation server. We find that (1) Methods with gradient alignment objectives achieve the best performance, verifying our assumption that gradient information benefits the student model. (2) Introducing gradient alignment on `[CLS]` (GKD-CLS) further boosts the performance, since `[CLS]` representations play a significant role in classification. (3) De-activating dropout is important. Activating Dropout in GKD-CLS leads to inferior results, which is consistent with our analysis (Section 3) that dropout will bias the gradient estimation, thus harming the performance.

Table 2: Results obtained by fine-tuning or distilling DistilBERT model on specific tasks. † denotes the results taken from Jiao et al. (2020). Detailed experimental settings can be found in Appendix C.2.

Model	SST-2	QQP	MNLI-m/mm	QNLI	Avg.
$\text{BERT}_{\text{BASE}}$	93.7	71.5	84.8 / 83.8	91.3	85.0
DistilBERT†	92.5	70.1	82.6 / 81.3	88.9	83.1
+ Vanilla KD	92.8	70.4	83.1 / 81.8	90.0	83.6
+ BERT-PKD	92.9	**71.3**	83.6 / **82.8**	90.1	84.1
+ GKD-CLS	**93.7**	**71.3**	**83.7 / 82.8**	**90.2**	**84.3**

Table 3: Behavior consistency on SST-2 test set. Our GKD methods achieve the best performance on all loyalty metrics, especially on saliency loyalty.

Model	PL	LL	SL
$\text{BERT}_{\text{BASE}}$	100.0	100.0	100.0
Vanilla KD	93.7	93.7	31.2
BERT-PKD	93.6	93.7	31.6
GKD	93.9	94.3	49.7
GKD-CLS	**94.9**	**94.9**	**53.5**

4.3 Analysis

GKD Benefits DistilBERT Recently, many knowledge distillation methods (Sanh et al., 2019; Sun et al., 2020; Jiao et al., 2020) have explored knowledge distillation during the pre-training stage to build a task-agnostic model or improve task-specific performance. In these works, knowledge distillation can be performed in two stages—the pre-training stage and fine-tuning stage. Since we lack enough computational resources to perform knowledge distillation at the pre-training stage, we instead utilize the 6-layer DisilBERT model (Sanh et al., 2019), which is already distilled at the pre-training stage, to perform further GKD distillation at the fine-tuning stage. Here, as the input embeddings of DistilBERT and BERT-base-uncased do not match, we only align the gradients w.r.t. the [CLS] tokens in the gradient alignment objective of GKD-CLS and do not fix the student input embeddings as the teacher input embedding. The experimental results are shown in Table 2, and the detailed experimental settings can be found in Section C.2 in Appendix. The results demonstrate that our GKD method can be integrated into the two-stage distillation procedure and can bring more benefits than other methods.

GKD Improves Output and Interpretation Consistency Traditional metrics like accuracy cannot reflect how alike the teacher and the student behave, which is of great significance in deployments (Xu et al., 2021). To measure the output consistency, we adopt **Label Loyalty (LL)** and **Probability Loyalty (PL)** following Xu et al. (2021). LL measures the faithfulness between the student predictions and the teacher predictions. PL measures the distance between the student predictions and the teacher predictions at the distribution level.

Besides, we are interested in whether GKD benefits the consistency of model interpretability, since model interpretation plays an important role in fields like medicine and finance, and a student loyal to the teacher in interpretation is useful. For this purpose, we propose **Saliency Loyalty (SL)**. Specifically, we calculate the teacher's and the student's word saliency distribution using the Grad method (Ding et al., 2019), and the SL is defined as the Pearson correlation coefficient between them.

We evaluate different methods using these metrics on SST-2 test set. Results are shown in Table 3.[3] We observe that incorporating gradient into KD improves the behavior consistency regarding all the metrics, especially in SL, i.e. interpretation consistency. We also provide a case study of the saliency scores in Figure 2, which shows that the saliency scores can vary when all the methods predict the same, and that our methods show the best interpretation consistency, further verifying our motivation.

[3] Results on MNLI can be found in Table 5 in Appendix.

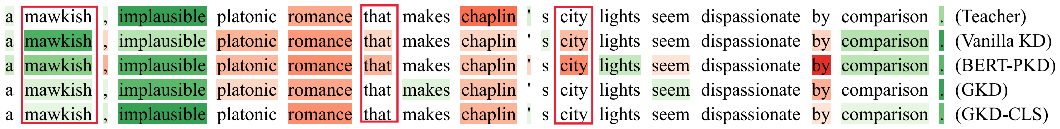

Figure 2: Saliency score visualization on SST-2 test set. Words with positive scores are marked green while negative ones red. While all methods predict the sentence as negative, the word saliency distributions vary. Our method achieves the best consistency with the teacher.

5 Conclusion

In this paper, we propose gradient knowledge distillation (GKD) by matching the unbiased gradient of the student to that of the teacher. As the gradient contains higher-order information, the alignment helps the student act more similarly to the teacher. Experiments show that GKD outperforms baseline methods regarding distillation performance and behavior consistency. For future work, we intend to integrate our method with different distillation frameworks to further examine its effectiveness.

References

Luisa Bentivogli, Ido Kalman Dagan, Dang Hoa, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC Workshop*.

Yen-Chun Chen, Zhe Gan, Yu Cheng, Jingzhou Liu, and Jingjing Liu. 2020. Distilling knowledge learned in BERT for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7893–7905, Online. Association for Computational Linguistics.

Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. 2022. Enabling multimodal generation on clip via vision-language knowledge distillation.

Shuoyang Ding, Hainan Xu, and Philipp Koehn. 2019. Saliency-driven word alignment interpretation for neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 1–12, Florence, Italy. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

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

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*, pages 4163–4174, Online. Association for Computational Linguistics.

Hanyang Kong, Jian Zhao, Xiaoguang Tu, Junliang Xing, Shengmei Shen, and Jiashi Feng. 2019. Cross-resolution face recognition via prior-aided face hallucination and residual knowledge distillation.

Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021a. Align before fuse: Vision and language representation learning with momentum distillation.

Lei Li, Yankai Lin, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2021b. Dynamic knowledge distillation for pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ravi Teja Mullapudi, Steven Chen, Keyi Zhang, Deva Ramanan, and Kayvon Fatahalian. 2019. Online model distillation for efficient video inference. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 3572–3581. IEEE.

Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural networks : the official journal of the International Neural Network Society*, 12 1:145–151.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeurIPS Workshop on Energy Efficient Machine Learning and Cognitive Computing*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Suraj Srinivas and François Fleuret. 2018. Knowledge transfer with jacobian matching. 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 4730–4738. PMLR.

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.*, 15:1929–1958.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

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*, pages 2158–2170, Online. Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy J. Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *ArXiv*, abs/1903.12136.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *ArXiv preprint*, abs/1908.08962.

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, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Yang Yu, Tao Qi, Yongfeng Huang, and Qi Liu. 2021. NewsBERT: Distilling pre-trained language model for intelligent news application. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3285–3295, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Ke Xu, Julian McAuley, and Furu Wei. 2021. Beyond preserved accuracy: Evaluating loyalty and robustness of BERT compression. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10653–10659, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yichen Zhu and Yi Wang. 2021. Student customized knowledge distillation: Bridging the gap between student and teacher. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5057–5066.

A Proof of Theorem 1

As f can be estimated by its second-order Taylor expansion around $\mathbf{x}_0$, we can rewrite f as

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^T \nabla^2 f(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0) \;, \tag{9}$$

where $\nabla f\left(\mathbf{x}_0\right)$ is a vector and $\nabla^2 f(\mathbf{x}_0)$ is a matrix.

Then, the expectation of gradient is

$$\begin{aligned}
&\mathbf{E}_{\boldsymbol{\xi}}[\nabla f(\mathbf{x}_0 \odot \boldsymbol{\xi})] \\
=&\nabla f(\mathbf{x}_0) - \nabla^2 f(\mathbf{x}_0)\mathbf{x}_0 + \mathbf{E}_{\boldsymbol{\xi}}[\boldsymbol{\xi} \odot \nabla^2 f(\mathbf{x}_0)(\mathbf{x}_0 \odot \boldsymbol{\xi})] \\
=&\nabla f(\mathbf{x}_0) - \nabla^2 f(\mathbf{x}_0)\mathbf{x}_0 + \nabla^2 f(\mathbf{x}_0)\mathbf{x}_0 + \frac{\delta}{1-\delta}\,\mathrm{diag}(\nabla^2 f(\mathbf{x}_0))\mathbf{x}_0 \\
=&\nabla f(\mathbf{x}_0) + \frac{\delta}{1-\delta}\,\mathrm{diag}(\nabla^2 f(\mathbf{x}_0))\mathbf{x}_0 \;.
\end{aligned} \tag{10}$$

Here, to calculate the expectation, we use the fact that each component of $\boldsymbol{\xi} \in \{0, \frac{1}{1-\delta}\}^d$ is drawn independently from a scaled Bernoulli$(1-\delta)$ random variable.

B Dropout Biases Gradient

To further confirm Theorem 1 by experiment, we sample 1000 items from MNLI train dataset to calculate the cosine similarity between the gradient without dropout and the expectation of the gradient with dropout. The model we use is trained by GKD-CLS method. Here, for each item from MNLI, we calculate the gradient $\mathbf{v}$ 100 times (with dropout activated) and use the average $\frac{\sum_{i=1}^{100}\mathbf{v}_i}{100}$ to estimate the expectation $\mathbf{E}_{\xi}\mathbf{v}$. The result is shown in Table 4.

Table 5: Results of loyalty on MNLI test dataset.

Model	**PL** (m / mm)	**LL** (m / mm)	**SL** (m / mm)
BERT$_{\text{BASE}}$	100.0 / 100.0	100.0 / 100.0	100.0 / 100.0
Vanilla KD	88.2 / 87.4	87.9 / 87.0	15.8 / 15.9
BERT-PKD	88.6 / 87.8	88.7 / 87.7	17.1 / 16.8
GKD	**89.7 / 89.1**	**89.5 / 88.9**	**35.6 / 35.5**
GKD-CLS	89.5 / 88.9	89.4 / 88.7	28.2 / 28.0

Table 4: Cosine similarity between the gradient without dropout and the expectation of the gradient with dropout.

$\mathbf{v}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{E}}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{h}_{1,\text{CLS}}}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{h}_{2,\text{CLS}}}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{h}_{3,\text{CLS}}}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{h}_{4,\text{CLS}}}$	$\frac{\partial p^s_{m,i}}{\partial \mathbf{h}_{5,\text{CLS}}}$
$\frac{\mathbf{E}_\xi[\mathbf{v}]\cdot\mathbf{v}}{\|\mathbf{E}_\xi[\mathbf{v}]\|\cdot\|\mathbf{v}\|}$	0.84	0.89	0.90	0.91	0.91	0.91

C Experimental Settings

C.1 Settings for the Main Results Shown in Table 1

For hyper-parameter search, we use a strategy similar to the one proposed by Sun et al. (2019). That is, we first search hyper-parameters α, τ for Vanilla KD method, and then fix α, τ to search the additional hyper-parameters in other methods. Besides, we initialize Vanilla KD (Hinton et al., 2015), BERT-PKD (Sun et al., 2019) and our method with the embedding layer and first 6 hidden layers of the teacher model.

For all the tasks, we fix the training batch size as 32, the training epoch number as 4, and the learning rate as 5e-5.

For Vanilla KD, we perform grid search for α in $\{0.2, 0.5, 0.7\}$ and τ in $\{5, 10, 20\}$ and get the best α, τ on validation set. This α, τ will be used for BERT-PKD, GKD and GKD-CLS. Then, for BERT-PKD, we search β in $\{10, 100, 500, 1000\}$; for our GKD method, we search β in $\{0.05, 0.1, 0.2, 0.4\}$; for our GKD-CLS method, we fix β to be 500 and search γ in $\{0.02, 0.05, 0.1, 0.2\}$.

We run all experiments on RTX 2080 Ti GPUs. One signal run of our method takes about 1.5h / 7.5h / 7.5h / 2h for SST-2 / QQP / MNLI / QNLI on a single RTX 2080 Ti GPU and the baseline methods take roughly half the time for a single run.

We choose the models that show the highest accuracy on validation datasets to generate the predictions on test datasets, which are then submitted to the official GLUE evaluation server.

C.2 Settings for the Analysis Results Shown in Table 2

The experimental settings for Table 2 are similar to Section C.1, except that we use the DistilBERT model as the student rather than initializing the student from the teacher. Besides, we only align the gradients w.r.t. the `[CLS]` tokens in the gradient alignment objective of GKD-CLS since the embeddings of DistilBERT and BERT-base-uncased do not match, and we set β as 1000 in GKD-CLS to encourage the alignment of `[CLS]` tokens.

D Loyalty Results on MNLI

We also evaluate the label loyalty, probability loyalty and saliency loyalty on the MNLI test dataset. The results are shown in Table 5.

E Discussion of Distillation Cost

Since our method needs to align the gradient, we need to conduct backpropagation twice in order to calculate the gradient of the gradient alignment objective w.r.t. the parameters, which is then used for model parameter update. So, our method requires roughly twice the computational cost as normal KD in the training procedure. For example, on the largest MNLI dataset, our method needs about 7.5h to finish training on a single RTX 2080 Ti GPU while vanilla KD takes about 3.7h. However, the training overhead is negligible as we only need to train the model once for deployments and is totally acceptable considering the better distillation performance. And, the inference speed of our method is the same as normal KD, since both methods use the student of the same architecture for inference.