Tutoring Helps Students Learn Better: Improving Knowledge Distillation for BERT with Tutor Network

Junho Kim[1*], **Jun-Hyung Park**[2*], **Mingyu Lee**[1],
Wing-Lam Mok[1], **Joon-Young Choi**[1], **SangKeun Lee**[1,2]
[1]Department of Artificial Intelligence [2]Department of Computer Science and Engineering
Korea University, Seoul, Republic of Korea
{monocrat, irish07, decon9201, wlmokac, johnjames, yalphy}@korea.ac.kr

Abstract

Pre-trained language models have achieved remarkable successes in natural language processing tasks, coming at the cost of increasing model size. To address this issue, knowledge distillation (KD) has been widely applied to compress language models. However, typical KD approaches for language models have overlooked the difficulty of training examples, suffering from incorrect teacher prediction transfer and sub-efficient training. In this paper, we propose a novel KD framework, Tutor-KD, which improves the distillation effectiveness by controlling the difficulty of training examples during pre-training. We introduce a tutor network that generates samples that are easy for the teacher but difficult for the student, with training on a carefully designed policy gradient method. Experimental results show that Tutor-KD significantly and consistently outperforms the state-of-the-art KD methods with variously sized student models on the GLUE benchmark, demonstrating that the tutor can effectively generate training examples for the student[1].

1 Introduction

Pre-trained language models (PLMs) have achieved great success in extensive natural language processing (NLP) tasks by learning generalized language representations from large text corpora (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Clark et al., 2020). BERT (Devlin et al., 2019) and its variants (Liu et al., 2019; Yang et al., 2019; Clark et al., 2020; Joshi et al., 2020), have shown significant performance improvement on natural language understanding tasks through learning bidirectional contextualized representations. However, due to the high memory footprints and computational costs, these models are challenging to be used in resource-constrained situations, such as mobile devices.

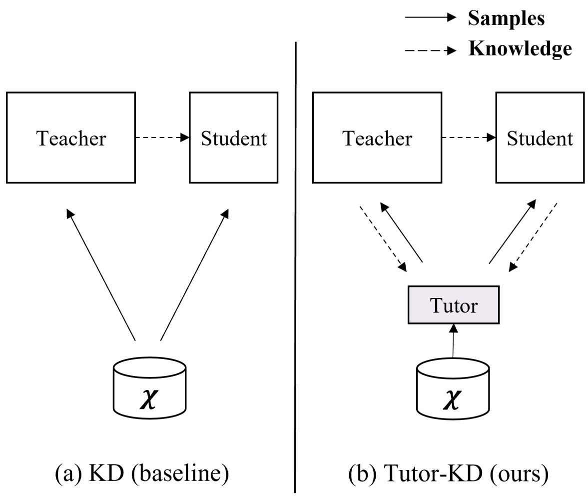


Figure 1: Comparison between (a) KD (baseline) and (b) Tutor-KD (ours) framework. Our framework generates samples for effective knowledge transfer from the teacher to the student.

KD (Hinton et al., 2015; Romero et al., 2015) is a promising approach for compressing pre-trained transformers, by transferring knowledge from a large source network (i.e., a teacher) to a small target network (i.e., a student). Previous studies have shown that the performance of the student can be significantly improved by learning informative learning signals, such as output probabilities (Sanh et al., 2019), intermediate feature representations, and attention probabilities (Jiao et al., 2020; Sun et al., 2020; Wang et al., 2020, 2021) from the pre-trained teacher.

Despite these compelling results, typical KD approaches used in language modeling have two key limitations arising from ignoring the difficulty of training examples. First, since the student mimics learning signals from the teacher regardless of whether it is right or wrong, incorrect teacher predictions from KD for overly difficult examples can be transferred to the student. Second, with the

*These authors contributed equally to this work.
[1]Our code is publicly available at https://github.com/JunhoKim94/TutorKD/

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 7371–7382
December 7-11, 2022 ©2022 Association for Computational Linguistics

ignorance of the difficulty of training examples, the performance of the language models on downstream tasks can be significantly affected (Clark et al., 2020), which may lead to sub-efficient KD.

In this paper, we propose Tutor-KD, a novel KD framework for PLMs to address the aforementioned limitations. Our key idea is to introduce a tutor network that controls the difficulty of training examples during KD. Specifically, the tutor generates training examples, which are easy for the teacher but difficult for the student, by replacing the masked tokens with corrupted tokens via masked language modeling (MLM). To accurately identify the difficulty of the tutor-generated training samples, we propose a novel method to train the tutor based on a policy gradient with carefully designed rewards. The tutor network is therefore optimized to generate tokens with relatively lower teacher losses (i.e., more accurate predictions) and consequently prevent overly difficult samples from being generated. Simultaneously, the student trained with generated examples can be benefited to learn more effectively due to the increased difficulty.

We conduct extensive experiments on downstream NLP tasks using various sizes of student models. Our experimental results show that the proposed approach significantly improves the language model distillation performance. Specifically, the 6-layer model with 768 hidden dimensions distilled from BERT-base outperforms the teacher model on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019). Moreover, our framework shows notable effectiveness for extremely small-sized student models that are designed $7.5\times$ smaller than BERT-base. Finally, we demonstrate that our designed rewards can generate effective samples for both the teacher and the student. As a summary of our main contributions:

- We propose Tutor-KD, a novel KD framework for PLMs that improves the distillation effectiveness by considering the difficulty of training examples.
- We present a tutor network with the corresponding training scheme to generate training samples that are easy for the teacher but difficult for the student based on a policy gradient.
- We verify that Tutor-KD improves the effectiveness of KD on students with various sizes through extensive experiments.

2 Related Work

2.1 Pre-trained Language Model

Unsupervised pre-training of language models has achieved impressive results across various NLP tasks with generalized representation learning. In particular, BERT (Devlin et al., 2019) has obtained strong bidirectional contextual representations with MLM, which recovers the masked tokens in the input sequences. Recently, numerous studies have been conducted to improve MLM (Liu et al., 2019; Joshi et al., 2020; Zhang et al., 2019; Yang et al., 2019; Clark et al., 2020). RoBERTa (Liu et al., 2019) achieves strong performance by dynamically masking the input sequences during pre-training. ELECTRA (Clark et al., 2020) significantly improves the training efficiency and language model performance by introducing replaced token detection (RTD), a pre-training task to predict whether the input tokens are replaced by plausible alternatives from the generator network or not.

2.2 Knowledge Distillation

KD has been proven to be a promising approach to compress language models, transferring knowledge from a large teacher model to a small student model. Hinton et al. (2015) first propose KD, by using the soft target probability distribution from the teacher model. Romero et al. (2015) use information on the intermediate representations from the hidden layers of a teacher network. Hu et al. (2018) introduce the attention distillation from transformers.

In this work, we focus on task-agnostic knowledge distillation for PLMs, which can be adapted to downstream tasks through fine-tuning and be utilized to initialize task-specific distillation (Sun et al., 2019; Aguilar et al., 2020; Rashid et al., 2021; Haidar et al., 2022; Zhang et al., 2022). DistilBERT (Sanh et al., 2019) uses soft label distillation and cosine embedding losses from the teacher. TinyBERT (Jiao et al., 2020) and MobileBERT (Sun et al., 2020) transfer hidden representations and self-attention distributions. MiniLM (Wang et al., 2020) and MiniLMv2 (Wang et al., 2021) only use the self-attention distributions of the transformer layer to avoid restrictions on the number of student layers. While most previous works have been conducted for better distillation on PLMs by transferring informative signals, studies on generating better samples for distillation have not yet been well explored. Different from previous works, we focus on controlling the difficulty of training exam-

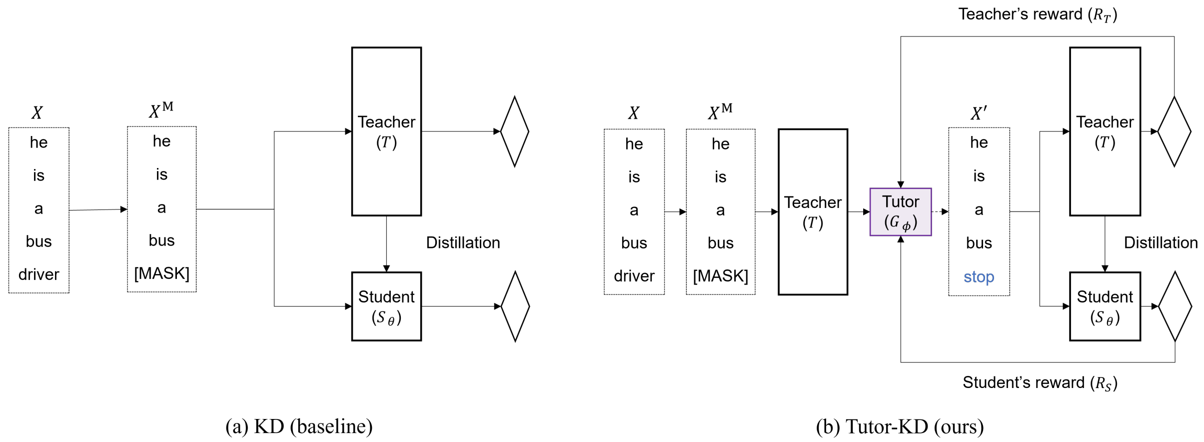


Figure 2: Overview of the (a) baseline KD and the proposed (b) Tutor-KD framework. In baseline KD, the student directly receives knowledge regarding the masked sample X^M from the teacher. In Tutor-KD, the tutor is trained to generate sample X' for the student by maximizing the two rewards (R_T, R_S), representing the degree of difficulty level for the teacher and the student when dealing with X', respectively.

ples to increase the effectiveness of KD.

2.3 Data Augmentation for KD

Prior studies have demonstrated that sampling more challenging examples for student models is conducive to more effective training on downstream tasks. For instance, MATE-KD (Rashid et al., 2021) suggests a min-max adversarial data augmentation approach for KD, where an extra generator model is trained to maximize the loss between teacher and student. ComKD (Li et al., 2021b) and CILDA (Haidar et al., 2022) incorporate progressive training and contrastive loss with an adversarial augmentation approach, respectively. Despite the advancements brought by previous works, they overpass the problem of inaccurate teacher prediction and only focus on generating adversarial samples for KD on specific target tasks. Yet, we empirically observe that transferring knowledge with overly difficult samples for the teacher can be harmful to student performance. Our study differs from the existing approaches in that we aim to distill knowledge by generating samples that are easy for the teacher but difficult for the student. Moreover, Tutor-KD aims at producing an efficient language model by transferring general linguistic knowledge for widespread applicability on various downstream NLP tasks.

3 Methodology

In this section, we introduce our proposed KD framework with its implementation details. Figure 2 illustrates the overall architecture of the proposed framework. The core idea is to define the respective difficulty level of training examples for the teacher and the student while maximizing the rewards for training the tutor.

3.1 Tutor Network

Inspired by previous works (Rashid et al., 2021; Clark et al., 2020), rather than directly utilizing a generation-based model, we adopt an MLM-based model as a generator to maintain training stability. Given an original sample $X = [x_1, x_2, ..., x_n]$, our goal is to generate pseudo training sample $X' = [x'_1, x'_2, ..., x'_n]$ for transferring the teacher's knowledge signals. We deploy a tutor network G with trainable parameter ϕ, which is trained to map masked input X^M to the pseudo training example X'. First, we randomly mask tokens at positions $\mathbf{m} = [m_1, ..., m_k]$ of X to obtain X^M. The tutor network then encodes X^M and predicts MLM output distribution[2] p_G at each masked position t. At each masked position t, we sample the replacements from p_G and generate X' as:

$$\begin{aligned} x'_t &\sim p_G(x_t|X^M) \text{ for } t \in \mathbf{m} \\ X' &= \text{replace}(X^M, \mathbf{m}, \mathbf{x}'), \end{aligned} \tag{1}$$

where replace is an operation that replaces the masked tokens with the sampled tokens at position $\mathbf{m}$. The tutor is trained to maximize the rewards

[2] Our tutor network is constructed as an MLM classifier. It generates X' based on the last hidden layer representations of the teacher model from X^M.

from the teacher R_T and the student R_S, by feeding the pseudo sample X' to both the teacher and the student model. The student is then trained during the minimization step.

3.2 Maximization Step

Our tutor network is trained to generate pseudo samples by maximizing the following loss function, which is calculated as the weighted sum of the two rewards:

$$\begin{aligned} \max_{\phi} L_P(\phi) &= \lambda R_T(T(X')) \\ &+ (1-\lambda) R_S(T(X'), S(X')), \end{aligned} \tag{2}$$

where R_T and R_S represent the rewards for the teacher and the student, respectively. S represents the student model with trainable parameters θ, T represents the teacher model, and λ represents the weight of the rewards.

Teacher's Reward. To generate samples that are easy for the teacher to discriminate, we introduce a reward R_T regarding on the teacher prediction for the generated sample X'. Following the principle of margin sampling (Settles, 2009; Scheffer et al., 2001), we suppose that the greater the difference between the probability of the original and the replaced tokens in the teacher's MLM prediction, the easier the teacher model to distinguish the replaced token. For example, the teacher can easily distinguish that *stop* is wrong in sentence X' if *driver* has a much larger probability than *stop* (Figure 2(b)). Therefore, we design our teacher's reward as the probability difference between the original and the replaced tokens. First, we calculate the original p_t^o and replaced p_t^r token probabilities from the MLM prediction at position t in the teacher model as:

$$p_t^o = \frac{\exp(z_t^o)}{\sum_i \exp(z_t^i)}, \quad p_t^r = \frac{\exp(z_t^r)}{\sum_i \exp(z_t^i)}, \tag{3}$$

where z_t^i represents the i-th logit value in the MLM logits vector at position t. Then we define the reward as follows:

$$\begin{aligned} R_T(T(X')) &= \sum_{t \in \mathbf{m}} r_T(x_t', X') \\ r_T(x_t', X') &= p_t^o - p_t^r, \end{aligned} \tag{4}$$

where r_T represents the teacher's reward at position t. To penalize the incorrect knowledge predicted by the teacher model, we allow a negative reward value as well.

Student's Reward. To generate samples that are difficult for the student to distinguish, we introduce the reward R_S, which represents the degree of difficulty of the generated sample X' for the student. We use the distillation loss between teacher and student in the position of the masked token. R_S is calculated as the loss between the original and predicted token logits as:

$$\begin{aligned} R_S(T(X'), S(X')) &= \sum_{i \in \mathbf{m}} r_S(x_t', X') \\ r_S(x_t', X') &= |a_t^T - a_t^S|, \end{aligned} \tag{5}$$

where a_t^T and a_t^S refer to the modified logit values of the teacher and the student model at position t, respectively (Section 3.3). We use L1 loss to match the reward scale with the teacher model.

Training Objective. Due to the discrete sampling in the generation step, it is impossible to back-propagate through sampling from $p_G(x_t|X^M)$. In this work, we adopt policy gradient reinforcement learning (Williams, 1992) to maximize our rewards. We assume that the rewards of the teacher and the student model depend only on x_t and the non-replaced tokens, following the assumption in the adversarial learning of ELECTRA (Clark et al., 2020). We rewrite the rewards r_T and r_S in Equations (4) and (5) as $r_T(x_t', X)$ and $r_S(x_t', X)$, respectively. Given these conditions, we use the REINFORCE gradient, and the loss is as follows:

$$\begin{aligned} \max_{\phi} \widetilde{L_P} = \mathop{\mathbb{E}}_{X,\mathbf{m}} \sum_{t \in \mathbf{m}} \mathop{\mathbb{E}}_{x_t' \sim p_G} [\log p_G(x_t'|X^M) \\ \times \{\lambda r_T(x_t', X) + (1-\lambda) r_S(x_t', X)\}]. \end{aligned} \tag{6}$$

For training efficiency, we approximate the expectations with a single sample and train ϕ with gradient ascent.

3.3 Minimization step

In the minimization step, the student network is trained to minimize the gap between the teacher and student predictions. In addition, to prevent the tutor from generating implausible tokens, we also distill the teacher's MLM knowledge into the tutor network. We minimize the following loss function:

$$\min_{\theta,\phi} L_{total} = L_S(\theta) + L_{tutor}(\phi), \tag{7}$$

where L_{tutor} denotes the KL divergence loss between the pre-trained MLM logits of the teacher and tutor model, L_S denotes the distillation loss of the student, which is calculated as the sum of the modified logit loss L_{logit} and the internal representation distillation loss L_{inter} as follows:

$$L_S(\theta) = \lambda_1 L_{logit}(\theta) + \lambda_2 L_{inter}(\theta). \quad (8)$$

Here, we use 25 and 0.5 for λ_1 and λ_2, respectively.

Logit Modification. Previous work (Jiao et al., 2020) finds that conducting MLM logit distillation with internal representation distillation does not bring improvements to the downstream tasks. For this reason, we design a modified logit to improve the effectiveness of distillation. Since class probability refers to the plausibility of words in the context based on the teacher model's prediction, we modify the replaced token's probability as the ratio to the original token probability. The modified probability of the teacher at position t is defined as:

$$a_t^T = \begin{cases} p_t^r/p_t^o & , \ p_t^r < p_t^o \\ 1 & , \ p_t^r \geq p_t^o, \end{cases} \quad (9)$$

By normalizing all tokens with a higher probability than the original tokens, our design on the modified probability ensures that there is no value higher than the ground truth label 1. Note that the probability of the teacher's MLM is taken from the masked sample[3]. The student model predicts the modified probabilities at position t for the replaced tokens as:

$$a_t^S = \sigma(p_S(x_t'|X')), \quad (10)$$

where σ denotes the sigmoid function. The final distillation loss for the modified probability is calculated as:

$$L_{logit} = \mathrm{CE}(\mathbf{a^T}/\tau, \mathbf{a^S}/\tau). \quad (11)$$

where $\mathbf{a^T}$ and $\mathbf{a^S}$ are the modified probability vectors calculated by the teacher and the student respectively, and CE denotes the cross-entropy loss.

Internal Representations. To transfer more fine-grained knowledge (Romero et al., 2015), we distill knowledge from the intermediate layer following previous works (Jiao et al., 2020; Sun et al., 2020). We consider two types of distillation strategies: L_{hidden} based on the intermediate hidden representations, and L_{att} based on the attention information. The objectives are given as:

$$L_{hidden} = \mathrm{MSE}(H^S W, H^T), \quad (12)$$

$$L_{att} = \sum_i^h \mathrm{MSE}(A_i^S, A_i^T), \quad (13)$$

where H^T and H^S refer to the hidden states of the teacher and the student network, respectively. W denotes a trainable linear transformation that transforms the hidden states of the student network into the same space as the teacher network's hidden states. A_i^T and A_i^S are the attention distributions corresponding to the i-th self-attention heads of the teacher and the student, respectively. The scalar value h represents the number of attention heads. The internal representation distillation loss is calculated as the sum of the above two types of losses as follows:

$$L_{inter} = L_{hidden} + L_{att}. \quad (14)$$

4 Experiment

In this section, we evaluate the effectiveness of our proposed distillation framework on the GLUE benchmark using different model settings.

4.1 Knowledge Distillation Setup

We use the uncased version of the BERT-base model provided by HuggingFace (Wolf et al., 2019) as a teacher model. BERT-base (Devlin et al., 2019) is a 12-layer transformer with a hidden size of 768 and 12 attention heads, containing 109M parameters. We use English Wikipedia and BookCorpus (Zhu et al., 2015) as the KD corpora, and follow the preprocessing and WordPiece tokenization of BERT. The vocabulary size is 30,522 and we set the maximum sequence length to 128. We use Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We also train a 6-layer model with a hidden size of 768 used in most previous works (Jiao et al., 2020; Wang et al., 2020, 2021), as a student model. We adopt a feed-forward filter size of 3072 and 12 attention heads for the student model.

Extremely Small Models. We also design and train on three extremely small student models, which are 7.5× or more smaller than the BERT-base. The hidden size of the model with 5.7M parameters is 264, with a word embedding size of 132 and a feed-forward filter size of 1056, while the number of layers is reduced to 2. For the models with 9M and 14M parameters, the hidden size, middle layer size, and word embedding size of the

[3] We observe that the teacher usually makes better predictions in X^M than X'. Thus, our design adopts the teacher's MLM probabilities from X^M

Model	#Params	MNLI	QNLI	QQP	SST	CoLA	STS	MRPC	RTE	Avg.
BERT-base	109M	83.9	90.6	91.2	92.8	59.9	86.3	88.7	65.4	82.2
BERT-small	66M	81.8	89.2	90.6	91.3	52.5	84.5	87.3	66.7	80.5
DistilBERT	66M	82.1	89.4	90.5	90.7	43.6	84.8	87.8	59.9	78.6
TinyBERT	66M	82.4	90.0	90.4	91.7	45.9	85.1	87.8	64.9	79.7
MiniLM †	66M	83.2	90.1	90.7	91.8	54.1	85.5	88.3	66.2	81.2
MiniLM v2	66M	**83.6**	**90.4**	90.8	92.0	52.1	85.2	88.5	67.3	81.2
Tutor-KD (ours)	66M	**83.6**	**90.4**	**91.0**	**92.2**	**61.0**	**86.4**	**88.7**	**68.9**	**82.7**

Table 1: Comparison among various 6-layer models distilled from BERT-base on the GLUE benchmark. The weight of BERT-small is from (Turc et al., 2019). MiniLM † denotes that the results are evaluated from our re-implemented model. For the results of other models, we fine-tune the latest version of their publicly available models for a fair comparison.

models remain unchanged, with only the number of layers changed to 6 and 12, respectively. We use 12 attention heads for all extremely small student models.

Hyper-parameters. For distillation, we train all our student models using a batch size of 128 and a peak learning rate of 5e-4 for 1M steps. We use a linear learning rate warmup for the first 10% of the total steps followed by a linear learning rate decay. The dropout rate and L2 normalization weight are 0.1 and 0.01, respectively.

Hardware Details. We train all our student models using a single RTX 3090 GPU. We use mixed-precision training (Micikevicius et al., 2018) to expedite the training procedure. All the experiments are performed using the PyTorch framework.

4.2 Evaluation Setup

Following previous studies on language model pre-training (Devlin et al., 2019) and distillation (Sanh et al., 2019; Jiao et al., 2020; Sun et al., 2020; Wang et al., 2020), we evaluate our models on the GLUE benchmark (Wang et al., 2019). The GLUE benchmark comprises eight sentence-level classification tasks. Specifically, there are two single sentence tasks: CoLA (Warstadt et al., 2019) and SST (Socher et al., 2013), three sentence similarity tasks: MRPC (Dolan and Brockett, 2005), STS-B (Cer et al., 2017) and QQP (Chen et al., 2018), and three natural language inference tasks: MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Bentivogli et al., 2009). For evaluation metrics, we report Matthew's correlation for CoLA, Spearman's correlation for STS-B, and accuracy for the remaining tasks.

We use a batch size of 32, a maximum sequence length of 128, and fine-tune for 5 epochs by choosing the best learning rate from {2e-5, 3e-5, 4e-5, 5e-5} on the development set. For challenging tasks such as CoLA, MRPC, and RTE, we use 10 epochs instead. We add a linear classifier on top of the [CLS] token to predict label probabilities. We report the average results of 4 random fine-tuning.

4.3 Main Results

For a fair comparison, we mainly compare our model with several task-agnostic KD baselines (Sanh et al., 2019; Jiao et al., 2020; Wang et al., 2020, 2021) without data augmentation. Following previous works, we distill BERT-base into a 6-layer student model with a hidden size of 768.

Table 1 presents the results for the development set of the GLUE benchmark. Our model achieves state-of-the-art performance for the student model with 768 hidden sizes. Specifically, our model shows a 1.5 points better performance than MiniLM v2 on the GLUE average. In addition, our student model obtains 82.7 points on the GLUE average, which is higher than the performance of the BERT-base.

4.4 Extremely Small Models

To investigate the effect of our framework on extremely small-sized models, we compare Tutor-KD with our implemented KD (soft label distillation) and MiniLM models on GLUE benchmark tasks. MiniLM models are trained with 12 relation heads, and all models are trained using the same corpus and hyper-parameter settings.

The results are presented in Table 2. Our method is shown to be effective even for extremely small-sized models. Specifically, Tutor-KD improves the performance of the student models with 14M, 9M, and 5.7M parameters by 1.3, 2.6, and 1.3 points on average, respectively. However, we have not con-

Model	#L	#Params	CoLA	MRPC	RTE	SST	QQP	QNLI	MNLI	Avg.
KD	12	14M	52.2	86.8	59.0	90.9	89.4	86.8	79.3	77.8
MiniLM	12	14M	45.8	87.8	64.3	90.6	**90.1**	**89.7**	81.6	78.5
MiniLMv2	12	14M	48.7	**88.0**	65.7	90.9	90.0	**89.7**	82.2	79.0
Tutor-KD (ours)	12	14M	**54.3**	**88.0**	**68.3**	**91.3**	90.0	88.6	**82.3**	**80.3**
KD	6	9M	40.5	85.5	60.0	89.0	89.0	85.2	78.3	75.4
MiniLM	6	9M	33.8	85.9	63.9	89.2	**89.2**	87.9	79.5	75.6
MiniLMv2	6	9M	35.3	87.1	65.7	88.2	88.9	87.9	79.8	76.1
Tutor-KD (ours)	6	9M	**44.5**	**87.4**	**68.9**	**89.3**	**89.2**	**88.4**	**79.9**	**78.7**
KD	2	5.7M	12.2	75.0	58.0	86.1	86.1	81.3	70.7	67.8
MiniLM	2	5.7M	17.4	75.5	58.2	84.6	85.4	78.9	70.1	68.0
MiniLMv2	2	5.7M	13.2	81.1	58.2	85.4	86.0	82.1	**71.1**	69.2
Tutor-KD (ours)*	2	5.7M	**22.0**	**82.5**	**62.0**	**87.0**	**86.5**	**82.5**	70.9	**70.5**

Table 2: Comparison between student models with extremely small-sized architectures distilled from BERT-base. * denotes that the model is trained without attention distillation loss. #L indicates the number of layers.

Model	RTE	QNLI	MNLI
Tutor-KD	**68.9**	**88.4**	**79.9**
w/o R_T	66.7	88.2	79.7
w/o R_S	66.7	88.1	79.7
w/o R_T, R_S	66.2	87.9	79.6

Table 3: Ablation study on the rewards from the teacher and the student for tutor network.

Model	RTE	QNLI	MNLI
Tutor-KD	**68.9**	**88.4**	**79.9**
w/o Mod	67.5	87.7	79.7
w/o Mod, Tutor	65.5	87.2	79.1

Table 4: Ablation study on the logit modification. "w/o Mod" model refers to the Tutor-KD trained by using MLM logit distillation and "w/o Mod, Tutor" is trained with masked tokens.

ducted attention distillation for the 2-layer student models, given that the 2-layer student models show significantly worse performance when using attention representation transfer. We speculate that this is because student models with extremely shallow layers struggle to distill internal representations from the teacher model (Aguilar et al., 2020).

5 Analysis

To better understand the main advantages of our Tutor-KD, we conduct several analysis experiments. We perform all experiments on a 6-layer student model with 9M parameters using the same corpora.

5.1 Ablation Studies

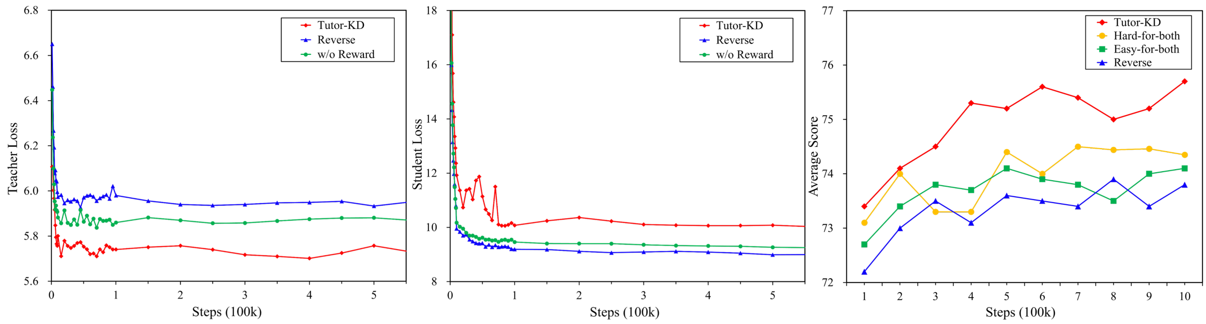
We conduct ablation studies for investigating the contributions of the rewards for the tutor network and the logit modification, respectively. Detailed results are presented in Table 3 and Table 4. We report the evaluation results on three NLI tasks from the GLUE benchmark (RTE, QNLI, and MNLI).

Rewards. We first explore the impact brought by the teacher's reward (R_T). As shown in Table 3, we observe that removing the teacher's reward (w/o R_T) significantly hurts the performance on all three NLI tasks. We also evaluate the impact of the student's reward (R_S) by comparing "w/o R_T" to "w/o R_T, R_S". Among different ablation settings, "w/o R_T, R_S" model performs worse than the "w/o R_T" model. Moreover, removing all rewards leads to a 2.7, 0.5, and 0.3 points performance drop on all three benchmark tasks. These results indicate that our tutor network with the reward schema can generate more useful samples for distillation.

Logit Modification. To examine the impact of modified logits, we first compare with the Tutor-KD results neglecting modification. As shown in Table 4, Tutor-KD without modification shows consistently worse performances on all three NLI tasks. These results demonstrate that our modified logits are conducive to more effective knowledge transfer. Nevertheless, we observe that "w/o Mod" can yet perform surpassing results over "w/o Mod, Tutor", which verify the improved effectiveness of using our tutor network on distilling knowledge.

5.2 Effect of Tutor Network

As aforementioned, we have demonstrated the effectiveness of our tutor network with impressive



(a) Loss curves of the teacher and student

(b) Student performance

Figure 3: Comparison results among various sampling methods based on (a) loss curves of the teacher and student and (b) the student performance on several GLUE tasks. "Reverse" refers to the sampling strategy for extracting samples that are hard for the teacher but easy for the student and "w/o Reward" represents the sampling from the tutor without policy gradient training. "Hard-for-both" represents the sampling strategy for generating samples that are difficult for both the teacher and student, while "Easy-for-both" works as the same logic with generating easy samples instead.

results on NLI tasks. To further investigate the role of the tutor network acting in Tutor-KD, we report the loss curves with respect to training steps. We evaluate the mean loss using 8000 examples randomly sampled from the Wikipedia corpus and compare the three sampling strategies.

The results are shown in Figure 3 (a). Considering that most training samples are difficult for the student model at the early stage of training, we observe that the tutor tends to generate easy samples for the teacher. However, as the student gradually obtains enough knowledge, the tutor starts generating more difficult samples, matching with the knowledge level of the teacher and student accordingly. In addition, compared with other baselines, Tutor-KD consistently generates samples that have relatively low teacher loss and high student loss. These results demonstrate that our tutor network successfully generates samples that are easy for the teacher but difficult for the student simultaneously.

5.3 Effect of Sampling Strategy

We compare Tutor-KD with three baseline sampling strategies[4] to verify the effectiveness of our sampling strategy. We report average scores on five GLUE benchmark tasks (CoLA, MRPC, SST, RTE, and STS) with respect to training steps.

The results are shown in Figure 3 (b). We observe that our sampling strategy consistently performs the best over several GLUE tasks. These results show that training samples that are easy for the teacher but difficult for the student are more effective for language model distillation. Conversely, the student model trained using the reversed strategy performs the worst. We presume this is because the teacher model transfers incorrect signals to the student.

[4]The rewards are designed as reflecting the degree of difficulty level for the teacher and student. Therefore, we implement baseline strategies by applying the negative sign to the corresponding rewards.

Task	λ Hyperparameter				
	0.0	0.25	0.5	0.75	1.0
RTE	66.6	66.7	**68.9**	66.5	66.7
QNLI	88.1	88.2	**88.4**	87.9	88.1
MNLI	79.7	79.8	**79.9**	79.8	79.7

Table 5: Sensitivity analysis of λ. The larger the λ value, the stronger the influence of the feedback from the teacher model. If $\lambda = 0.5$, the weights of rewards from the teacher and the student are the same.

5.4 Sensitivity Analysis

To investigate the weight of the rewards from the teacher and the student, we report the results of students distilled with different ratios of λ. The results are presented in Table 5. The student model trained with the teacher and student rewards having the same effectiveness achieves the best results. In addition, reducing either the teacher or student reward shows negative effects on the student model. These results demonstrate that the balance between teacher and student rewards is sensitive to the performance of the student model.

5.5 Case Study

We conduct a qualitative analysis by presenting a few selected samples from Wikipedia. The results are shown in Table 6. The first two examples show

		Examples	Prediction
✓	Original	the [texas] longhorns play home games in the state's ...	N/A
	KD	the [MASK] longhorns play home games in the state's ...	long
	Tutor-KD	the [holy] longhorns play home games in the state's ...	texas
✓	Original	ended with ratnasimha's [defeat] against the delhi sultanate ...	N/A
	KD	ended with ratnasimha's [MASK] against the delhi sultanate ...	victory
	Tutor-KD	ended with ratnasimha's [defeat] against the delhi sultanate ...	defeat
✗	Original	he was invited as a [linguist] to the first turkish language congress ...	N/A
	KD	he was invited as a [MASK] to the first turkish language congress ...	speaker
	Tutor-KD	he was invited as a [guest] to the first turkish language congress ...	guest

Table 6: Examples of the token replaced by baseline KD and Tutor-KD respectively, regarding the same original input sequence. Prediction represents the prediction by the teacher model for the masked or replaced tokens.

that Tutor-KD generates samples as a replacement for the ease of correct teacher prediction. Specifically, we observe that the tutor network replaces the masked tokens with either implausible or original tokens to ease the difficulty level of the problems for the teacher. However, as shown in the third example, despite giving the modified samples by the tutor network, overly difficult tokens such as *linguist*, remains challenging and are mispredicted by the teacher model.

6 Conclusion

We have presented Tutor-KD, a novel KD framework that controls the difficulty of training examples. With the carefully designed rewards on the policy gradient method, our tutor network is trained to generate training examples that are easy for the teacher but difficult for the student. Through extensive experiments, we have verified that Tutor-KD significantly improves KD effectiveness. Specifically, our student models outperform the state-of-the-art KD baselines with various sizes of models on the GLUE benchmark. Furthermore, we have demonstrated that our tutor network can generate effective samples for training student models, resulting in consistent performance improvements.

7 Limitations

While we show that Tutor-KD successfully improves the effectiveness of KD, there are some limitations existed. First, we mainly focus on improving the effectiveness of the KD for the BERT-base model. However, it is an open question whether our framework can improve KD for larger teacher models. Although it is known for the adversely affected distillation efficacy with the widening capacity gap between the teacher and student (Mirzadeh et al., 2020), one recent approach reveals that the effect brings to the student performance by the capacity gap can be alleviated by gradually transferring more difficult knowledge to the student (Li et al., 2021a). Likewise, as Tutor-KD generates samples with gradually increasing difficulty levels for students, we believe that Tutor-KD is highly expected to contribute to KD on larger teacher models.

Second, despite the fact that our adopted training method, the policy gradient (Williams, 1992), for discrete sampling can generate samples that maximize the target rewards (Yu et al., 2017; Clark et al., 2020) in various NLP tasks, it usually suffers from high variances. To further improve our current work, we plan to explore techniques for training the tutor network that can reduce the high variance problems of policy gradient.

8 Acknowledgement

We thank the anonymous reviewers for their helpful comments. This work was supported by the Basic Research Program through the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2021R1A2C3010430), National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2020R1A4A1018309), and Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2019-0-00079, Artificial Intelligence Graduate School Program (Korea University)).

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. In *The Thirty-*

Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7350–7357. AAAI Press.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 1–14. Association for Computational Linguistics.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

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.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Md Akmal Haidar, Mehdi Rezagholizadeh, Abbas Ghaddar, Khalil Bibi, Philippe Langlais, and Pascal Poupart. 2022. Cilda: Contrastive data augmentation using intermediate layer knowledge distillation. *arXiv preprint arXiv:2204.07674*.

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

Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, Dongsheng Li, Nan Yang, and Ming Zhou. 2018. Attention-guided answer distillation for machine reading comprehension. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2077–2086. Association for Computational Linguistics.

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

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lei Li, Yankai Lin, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2021a. Dynamic knowledge distillation for pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 379–389. Association for Computational Linguistics.

Tianda Li, Ahmad Rashid, Aref Jafari, Pranav Sharma, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021b. How to select one among all? an extensive empirical study towards the robustness of knowledge distillation in natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

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.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2020. Improved knowledge distillation via teacher assistant. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The*

Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 5191–5198. AAAI Press.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

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.

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. *CoRR*, abs/1910.01108.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis, 4th International Conference, IDA 2001, Cascais, Portugal, September 13-15, 2001, Proceedings*, volume 2189 of *Lecture Notes in Computer Science*, pages 309–318. Springer.

Burr Settles. 2009. Active learning literature survey.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. 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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. Association for Computational Linguistics.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 4322–4331. 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, ACL 2020, Online, July 5-10, 2020*, pages 2158–2170. Association for Computational Linguistics.

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

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*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Trans. Assoc. Comput. Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel R. 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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. 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 5754–5764.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press.

Minjia Zhang, Uma-Naresh Niranjan, and Yuxiong He. 2022. Adversarial data augmentation for task-specific knowledge distillation of pre-trained transformers. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11685–11693. AAAI Press.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. 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 1441–1451. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.