Distilling Linguistic Context for Language Model Compression

Geondo Park[1] **Gyeongman Kim**[1] **Eunho Yang**[1,2]
KAIST[1], Daejeon, South Korea
AITRICS[2], Seoul, South Korea
{geondopark, gmkim, eunhoy}@kaist.ac.kr

Abstract

A computationally expensive and memory intensive neural network lies behind the recent success of language representation learning. Knowledge distillation, a major technique for deploying such a vast language model in resource-scarce environments, transfers the knowledge on individual word representations learned without restrictions. In this paper, inspired by the recent observations that language representations are relatively positioned and have more semantic knowledge as a whole, we present a new knowledge distillation objective for language representation learning that transfers the *contextual* knowledge via two types of relationships across representations: *Word Relation* and *Layer Transforming Relation*. Unlike other recent distillation techniques for the language models, our contextual distillation does not have any restrictions on architectural changes between teacher and student. We validate the effectiveness of our method on challenging benchmarks of language understanding tasks, not only in architectures of various sizes, but also in combination with DynaBERT, the recently proposed adaptive size pruning method.

1 Introduction

Since the Transformer, a simple architecture based on attention mechanism, succeeded in machine translation tasks, Transformer-based models have become a new state of the arts that takes over more complex structures based on recurrent or convolution networks on various language tasks, e.g., language understanding and question answering, etc (Devlin et al., 2018; Lan et al., 2019; Liu et al., 2019; Raffel et al., 2019; Yang et al., 2019). However, in exchange for high performance, these models suffer from a major drawback: tremendous computational and memory costs. In particular, it is not possible to deploy such large models on platforms with limited resources such as mobile and wearable devices, and it is an urgent research topic with impact to keep up with the performance of the latest models from a *small-size* network.

As the main method for this purpose, Knowledge Distillation (KD) transfers knowledge from the large and well-performing network (teacher) to a smaller network (student). There have been some efforts that distill Transformer-based models into compact networks (Sanh et al., 2019; Turc et al., 2019; Sun et al., 2019, 2020; Jiao et al., 2019; Wang et al., 2020). However, they all build on the idea that each word representation is independent, ignoring relationships between words that could be more informative than individual representations.

In this paper, we pay attention to the fact that word representations from language models are very *structured* and capture certain types of semantic and syntactic relationships. - Word2Vec (Mikolov et al., 2013) and Glove (Pennington et al., 2014) demonstrated that trained embedding of words contains the linguistic patterns as linear relationships between word vectors. Recently, Reif et al. (2019) found out that the distance between words contains the information of the dependency parse tree. Many other studies also suggested the evidence that contextual word representations (Belinkov et al., 2017; Tenney et al., 2019a,b) and attention matrices (Vig, 2019; Clark et al., 2019) contain important relations between words. Moreover, Brunner et al. (2019) showed the vertical relations in word representations across the transformer layers through word identifiability. Intuitively, although each word representation has respective knowledge, the set of representations of words as a whole is more semantically meaningful, since words in the embedding space are positioned relatively by learning.

Inspired by these observations, we propose a novel distillation objective, termed Contextual Knowledge Distillation (CKD), for language tasks that utilizes the statistics of relationships between word representations. In this paper, we define two types of contextual knowledge: *Word Relation (WR)* and *Layer Transforming Relation (LTR)*. Specifically, WR is proposed to capture the knowledge of *relationships between word representations* and LTR defines *how each word representation changes* as it passes through the network layers.

We validate our method on General Language Understanding Evaluation (GLUE) benchmark and the Stanford Question Answer Dataset (SQuAD), and show the effectiveness of CKD against the current state-of-the-art distillation methods. To validate elaborately, we conduct experiments on task-agnostic and task-specific distillation settings. We also show that our CKD performs effectively on a variety of network architectures. Moreover, with the advantage that CKD has no restrictions on student's architecture, we show CKD further improves the performance of adaptive size pruning method (Hou et al., 2020) that involves the architectural changes during the training.

To summarize, our contribution is threefold:

- (1) Inspired by the recent observations that word representations from neural networks are structured, we propose a novel knowledge distillation strategy, Contextual Knowledge Distillation (CKD), that transfers the relationships across word representations.

- (2) We present two types of complementary contextual knowledge: horizontal Word Relation across representations in a single layer and vertical Layer Transforming Relation across representations for a single word.

- (3) We validate CKD on the standard language understanding benchmark datasets and show that CKD not only outperforms the state-of-the-art distillation methods but boosts the performance of adaptive pruning method.

2 Related Work

Knowledge distillation Since recently popular deep neural networks are computation- and memory-heavy by design, there has been a long line of research on transferring knowledge for the purpose of compression. Hinton et al. (2015) first proposed a teacher-student framework with an objective that minimizes the KL divergence between teacher and student class probabilities. In the field of natural language processing (NLP), knowledge distillation has been actively studied (Kim and Rush, 2016; Hu et al., 2018). In particular, after the emergence of large language models based on pre-training such as BERT (Devlin et al., 2018; Liu et al., 2019; Yang et al., 2019; Raffel et al., 2019), many studies have recently emerged that attempt various knowledge distillation in the pre-training process and/or fine-tuning for downstream tasks in order to reduce the burden of handling large models. Specifically, Tang et al. (2019); Chia et al. (2019) proposed to distill the BERT to train the simple recurrent and convolution networks. Sanh et al. (2019); Turc et al. (2019) proposed to use the teacher's predictive distribution to train the smaller BERT and Sun et al. (2019) proposed a method to transfer individual representation of words. In addition to matching the hidden state, Jiao et al. (2019); Sun et al. (2020); Wang et al. (2020) also utilized the attention matrices derived from the Transformer. Several works including Liu et al. (2020); Hou et al. (2020) improved the performance of other compression methods by integrating with knowledge distillation objectives in the training procedure. In particular, DynaBERT (Hou et al., 2020) proposed the method to train the adaptive size BERT using the hidden state matching distillation. Different from previous knowledge distillation methods that transfer respective knowledge of word representations, we design the objective to distill the contextual knowledge contained among word representations.

Contextual knowledge of word representations Understanding and utilizing the relationships across words is one of the key ingredients in language modeling. Word embedding (Mikolov et al., 2013; Pennington et al., 2014) that captures the context of a word in a document, has been traditionally used. Unlike the traditional methods of giving fixed embedding for each word, the contextual embedding methods (Devlin et al., 2018; Peters et al., 2018) that assign different embeddings according to the context with surrounding words have become a new standard in recent years showing high performance. Xia and Zong (2010) improved the performance of the sentiment classification task by using word relation, and Hewitt and Manning (2019); Reif et al. (2019) found that the distance between contextual representations contains syntactic information of sentences. Recently, Brunner et al. (2019) also experimentally showed that the contextual representations of each token change over the layers. Our research focuses on knowledge distillation using context information between words and between layers, and to our best knowledge, we are the first to apply this context information to knowledge distillation.

3 Setup and background

Most of the recent state-of-the-art language models are stacking Transformer layers which consist of repeated multi-head attentions and position-wise feed-forward networks.

Transformer based networks. Given an input sentence with n tokens, $\boldsymbol{X} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{d_i \times n}$, most networks (Devlin et al., 2018; Lan et al., 2019; Liu et al., 2019) utilize the embedding layer to map an input sequence of symbol representations $\boldsymbol{X}$ to a sequence of continuous representations $\boldsymbol{E} = [e_1, \ldots, e_n] \in \mathbb{R}^{d_e \times n}$. Then, each l-th Transformer layer of the identical structure takes the previous representations $\boldsymbol{R_{l-1}}$ and produces the updated representations $\boldsymbol{R_l} = [r_{l,1}, r_{l,2}, \ldots, r_{l,n}] \in \mathbb{R}^{d_r \times n}$ through two sublayers: Multi-head Attention (MHA) and position-wise Feed Forward Network (FFN). The input at the first layer ($l = 1$) is simply $\boldsymbol{E}$. In MHA operation where h separate attention heads are operating independently, each input token $r_{l-1,i}$ for each head is projected into a query $q_i \in \mathbb{R}^{d_q}$, key $k_i \in \mathbb{R}^{d_q}$, and value $v_i \in \mathbb{R}^{d_v}$, typically $d_k = d_q = d_v = d_r/h$. Here, the key vectors and value vectors are packed into the matrix forms $\boldsymbol{K} = [k_1, \cdots, k_n]$ and $\boldsymbol{V} = [v_1, \cdots, v_n]$, respectively, and the attention value a_i and output of each head $o_{h,i}$ are calculated as followed:

$$a_i = \text{Softmax}\left(\frac{\boldsymbol{K}^T \cdot q_i}{\sqrt{d_q}}\right) \quad \text{and} \quad o_{h,i} = \boldsymbol{V} \cdot a_i$$

The outputs of all heads are then concatenated and fed through the FFN, producing the single word representation $r_{l,i}$. For clarity, we pack attention values of all words into a matrix form $\boldsymbol{A_{l,h}} = [a_1, a_2, .., a_n] \in \mathbb{R}^{n \times n}$ for attention head h.

Knowledge distillation for Transformer. In the general framework of knowledge distillation, teacher network (T) with large capacity is trained in advance, and then student network (S) with pre-defined architecture but relatively smaller than teacher network is trained with the help of teacher's knowledge. Specifically, given the teacher parameterized by θ_t, training the student parameterized by θ_s aims to minimize two objectives: i) the cross-entropy loss $\mathcal{L}^{\text{CE}}$ between the output of the student network S and the true label y and ii) the difference of some statistics $\mathcal{L}^{\text{D}}$ between teacher and student models. Overall, our goal is to minimize the following objective function:

$$\mathcal{L}(\theta_s) = \mathbb{E}\Big[\mathcal{L}^{\text{CE}} + \lambda \mathcal{L}^{\text{D}}\Big(K^t(\boldsymbol{X};\theta_t), K^s(\boldsymbol{X};\theta_s)\Big)\Big]$$

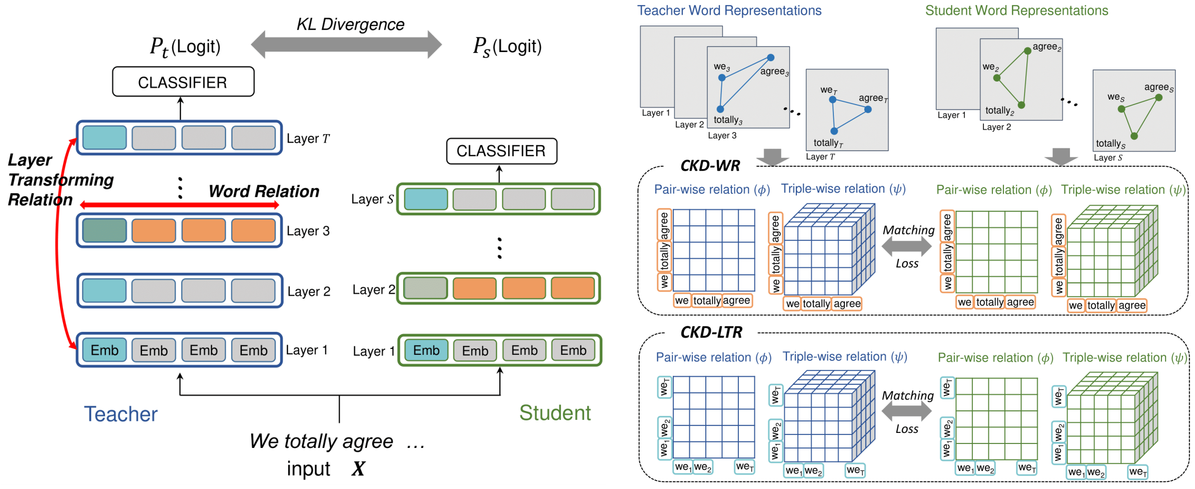
where λ controls the relative importance between two objectives. Here, K characterizes the knowledge being transferred and can vary depending on the distillation methods, and $\mathcal{L}^{\text{D}}$ is a matching loss function such as l_1, l_2 or Huber loss. Recent studies on knowledge distillation for Transformer-based BERT can also be understood in this general framework. In particular, each distillation methods of previous works are summarized in Appendix A.

4 Contextual Knowledge Distillation

We now present our distillation objective that transfers the *structural* or *contextual* knowledge which is defined based on the distribution of word representations. Unlike previous methods distilling each word separately, our method transfers the information contained in relationships between words or between layers, and provides a more flexible way of constructing embedding space than directly matching representations. The overall structure of our method is illustrated in Figure 1(a). Specifically, we design two key concepts of contextual knowledge from language models: *Word Relation*-based and *Layer Transforming Relation*-based contextual knowledge, as shown in Figure 1(b).

4.1 Word Relation (WR)-based Contextual Knowledge Distillation

Inspired by previous studies suggesting that neural networks can successfully capture contextual relationships across words (Reif et al., 2019; Pennington et al., 2014; Mikolov et al., 2013), WR-based CKD aims to distill the contextual knowledge contained in the relationships across words at certain layer. The "relationship" across a set of words can be defined in a variety of different ways. Our work focuses on defining it as the sum of *pair-wise* and *triple-wise* relationships. Specifically, for each input $\boldsymbol{X}$ with n words, let $\boldsymbol{R_l} = [r_{l,1}, \cdots r_{l,n}]$ be the word representations at layer l from the language model (it could be teacher or student), as described in Section 3. Then, the objective of WR-based CKD



(a) (b)

Figure 1: **Overview of our contextual knowledge distillation.** (a) In the teacher-student framework, we define the two contextual knowledge, word relation and layer transforming relation which are the statistics of relation across the words from the same layer (**orange**) and across the layers for the same word (**turquoise**), respectively. (b) Given the pair-wise and triple-wise relationships of WR and LTR from teacher and student, we define the objective as matching loss between them.

is to minimize the following loss:

$$\mathcal{L}_{\text{CKD-WR}} = \sum_{(i,j)\in\chi^2} w_{ij}\,\mathcal{L}^D\Big(\phi(r_i^s, r_j^s), \phi(r_i^t, r_j^t)\Big)$$
$$+\,\lambda_{\text{WR}} \sum_{(i,j,k)\in\chi^3} w_{ijk}\,\mathcal{L}^D\Big(\psi(r_i^s, r_j^s, r_k^s), \psi(r_i^t, r_j^t, r_k^t)\Big) \tag{1}$$

where $\chi = \{1, \ldots, n\}$. The function ϕ and ψ define the pair-wise and triple-wise relationships, respectively and λ_{WR} adjust the scales of two losses. Here, we suppress the layer index l for clarity, but the distillation loss for the entire network is simply summed for all layers. Since not all terms in Eq. (1) are equally important in defining contextual knowledge, we introduce the weight values w_{ij} and w_{ijk} to control the weight of how important each pair-wise and triple-wise term is. Determining the values of these weight values is open as an implementation issue, but it can be determined by the locality of words (i.e. $w_{ij} = 1$ if $|i-j| \le \delta$ and 0, otherwise), or by attention information $\boldsymbol{A}$ to focus only on relationship between related words. In this work, we use the locality of words as weight values.

While functions ϕ and ψ defining pair-wise and triple-wise relationship also have various possibilities, the simplest choices are to use the distance between two words for pair-wise ϕ and the angle by three words for triple-wise ψ, respectively.

Pair-wise ϕ via distance. Given a pair of word representations (r_i, r_j) from the same layer, $\phi(r_i, r_j)$ could be defined as cosine distance: $\cos(r_i, r_j)$ or l_2 distance: $\|r_i - r_j\|_2$.

Triple-wise ψ via angle. Triple-wise relation captures higher-order structure and provides more flexibility in constructing contextual knowledge. One of the simplest forms for ψ is the angle, which is calculated as

$$\psi(r_i, r_j, r_k) = \cos\angle(r_i, r_j, r_k)$$
$$= \left\langle \frac{r_i - r_j}{\|r_i - r_j\|_2}, \frac{r_k - r_j}{\|r_k - r_j\|_2} \right\rangle \tag{2}$$

where $\langle\cdot,\cdot\rangle$ denotes the dot product between two vectors.

Despite its simple form, efficiently computing the angles in Eq. (2) for all possible triples out of n words requires storing all relative representations $(r_i - r_j)$ in a (n, n, d_r) tensor[1]. This incurs an additional memory cost of $\mathcal{O}(n^2 d_r)$. In this case, using locality for w_{ijk} in Eq. (1) mentioned above can be helpful; by considering only the triples within a distance of δ from r_j, the additional memory space required for efficient computation is $\mathcal{O}(\delta n d_r)$, which is beneficial for $\delta \ll n$. It also reduces the computation complexity of computing triple-wise relation from $\mathcal{O}(n^3 d_r)$ to $\mathcal{O}(\delta^2 n d_r)$.

[1]From the equation $\|r_i - r_j\|_2^2 = \|r_i\|_2^2 + \|r_j\|_2^2 - 2\langle r_i, r_j\rangle$, computing the pair-wise distance with the right hand side of equation requires no additional memory cost.

Moreover, we show that measuring angles in local window is helpful in the performance in the experimental section.

4.2 Layer Transforming Relation (LTR) -based Contextual Knowledge Distillation

The second structural knowledge that we propose to capture is on *"how each word is transformed as it passes through the layers"*. Transformer-based language models are composed of a stack of identical layers and thus generate a set of representations for each word, one for each layer, with more abstract concept in the higher hierarchy. Hence, LTR-based CKD aims to distill the knowledge of how *each* word develops into more abstract concept within the hierarchy. Toward this, given a set of representations for a single word in L layers, $[r^s_{1,w}, \cdots, r^s_{L,w}]$ for student and $[r^t_{1,w}, \cdots, r^t_{L,w}]$ for teacher (Here we abuse the notation and $\{1, \dots, L\}$ is not necessarily the entire layers of student or teacher. It is the index set of layers which is defined in alignment strategy; this time, we will suppress the word index below), the objective of LTR-based CKD is to minimize the following loss:

$$\mathcal{L}_{\mathrm{CKD-LTR}} = \sum_{(l,m)\in\rho^2} w_{lm}\, \mathcal{L}^D\Big(\phi(r^s_l, r^s_m), \phi(r^t_l, r^t_m)\Big) + \lambda_{\mathrm{LTR}} \sum_{(l,m,o)\in\rho^3} w_{lmo}\, \mathcal{L}^D\Big(\psi(r^s_l, r^s_m, r^s_o), \psi(r^t_l, r^t_m, r^t_o)\Big) \quad (3)$$

where $\rho = \{1, \dots, L\}$ and λ_{LTR} again adjust the scales of two losses. Here, the composition of Eq. (3) is the same as Eq. (1), but only the objects for which the relationships are captured have been changed from word representations in one layer to representations for a single word in layers. That is, the relationships among representations for a word in different layers can be defined from distance or angle as in Eq. (2): $\phi(r_l, r_m) = \cos(r_l, r_m)$ or $\|r_l - r_m\|_2$ and $\psi(r_l, r_m, r_o) = \langle \frac{r_l - r_m}{\|r_l - r_m\|_2}, \frac{r_o - r_m}{\|r_o - r_m\|_2} \rangle$.

Alignment strategy. When the numbers of layers of teacher and student are different, it is important to determine which layer of the student learns information from which layer of the teacher. Previous works (Sun et al., 2019; Jiao et al., 2019) resolved this *alignment* issue via the *uniform (i.e. skip) strategy* and demonstrated its effectiveness experimentally. For L_t-layered teacher and L_s-layered student, the layer matching function f is defined as

$$f(\mathrm{step_s} \times t) = \mathrm{step_t} \times t, \quad \text{for } t = 0, \dots, g$$

where g is the greatest common divisor of L_t and L_s, $\mathrm{step_t} = L_t/g$ and $\mathrm{step_s} = L_s/g$.

Overall training objective. The distillation objective aims to supervise the student network with the help of teacher's knowledge. Multiple distillation loss functions can be used during training, either alone or together. We combine the proposed CKD with class probability matching (Hinton et al., 2015) as an additional term. In that case, our overall distillation objective is as follows:

$$\mathcal{L} = \mathcal{L}^{\mathrm{D}}_{logit} + \lambda_{\mathrm{CKD}} \Big(\mathcal{L}_{\mathrm{CKD-LTR}} + \mathcal{L}_{\mathrm{CKD-WR}}\Big)$$

where λ_{CKD} is a tunable parameter to balance the loss terms.

4.3 Architectural Constraints in Distillation Objectives

State-of-the-art knowledge distillation objectives commonly used come with constraints in designing student networks since they directly match some parts of the teacher and student networks such as attention matrices or word representations. For example, DistilBERT (Sanh et al., 2019) and PKD (Sun et al., 2019) match each word representation independently using their cosine similarities, $\sum_{i=1}^{n} \cos(r^t_{l,i}, r^s_{l,i})$, hence the embedding size of student network should follow that of given teacher network. Similarly, TinyBERT (Jiao et al., 2019) and MINI-LM (Wang et al., 2020) match the attention matrices via $\sum_{h=1}^{H} \mathbb{KL}(\boldsymbol{A}^t_{l,h}, \boldsymbol{A}^s_{l,h})$. Therefore, we should have the same number of attention heads for teacher and student networks (see Appendix A for more details on diverse distillation objectives).

In addition to the advantage of distilling contextual information, our CKD method has the advantage of being able to select the student network's structure more freely without the restrictions that appear in existing KD methods. This is because CKD matches the pair-wise or triple-wise relationships of words from *arbitrary* networks (student and teacher), as shown in Eq. (1), so it is always possible to match the information of the same dimension without being directly affected by the structure. Thanks to this advantage, in the experimental section, we show that CKD can further improve the performance of recently proposed

Table 1: Comparisons for **task-agnostic** distillation. For the task-agnostic distillation comparison, we do not use task-specific distillation for a fair comparison. The results of TinyBERT and Truncated BERT are ones reported in Wang et al. (2020). Other results are as reported by their authors. We exclude BERT-of-Theseus since the authors do not consider task-agnostic distillation. Results of development set are averaged over 4 runs. "-" indicates the result is not reported in the original papers and the trained model is not released. † marks our runs with the officially released model by the authors.

Model	#Params	CoLA (Mcc)	MNLI-(m/-mm) (Acc)	SST-2 (Acc)	QNLI (Acc)	MRPC (F1)	QQP (Acc)	RTE (Acc)	STS-B (Spear)	Avg
$\text{BERT}_{\text{BASE}}$ (Teacher)	110M	60.4	84.8/84.6	94.0	91.8	90.3	91.4	70.4	89.5	84.1
Truncated BERT (Sun et al., 2019)	67.5M	41.4	81.2/-	90.8	87.9	82.7	90.4	65.5	-	-
$\text{BERT}_{\text{Small}}$ (Turc et al., 2019)	67.5M	47.1†	81.1/81.7	91.1	87.8	87.9	90.0	63.0	87.5†	79.7
TinyBERT (Jiao et al., 2019)	67.5M	42.8	**83.5**/83.2†	91.6	90.5	88.4	90.6	**72.2**	88.5†	81.3
CKD	67.5M	**52.7**	**83.5/83.4**	**92.4**	**90.7**	**89.1**	**90.8**	70.1	**89.1**	**82.4**

Table 2: Comparisons for **task-specific** distillation. For a fair comparison, all students are 6/768 BERT models, distilled by $\text{BERT}_{\text{BASE}}$ (12/768) teachers. Other results except for TinyBERT and PKD are as reported by their authors. Results of development set are averaged over 4 runs. "-" indicates the result is not reported. Average score is computed excluding the MNLI-mm accuracy.

Model	#Params	CoLA (Mcc)	MNLI-(m/-mm) (Acc)	SST-2 (Acc)	QNLI (Acc)	MRPC (F1)	QQP (Acc)	RTE (Acc)	STS-B (Spear)	Avg
$\text{BERT}_{\text{BASE}}$ (Teacher)	110M	60.4	84.8/84.6	94.0	91.8	90.3	91.4	70.4	89.5	84.1
PD (Turc et al., 2019)	67.5M	-	82.5/83.4	91.1	89.4	89.4	90.7	66.7	-	-
PKD (Sun et al., 2019)	67.5M	45.5	81.3/-	91.3	88.4	85.7	88.4	66.5	86.2	79.2
TinyBERT (Jiao et al., 2019)	67.5M	53.8	83.1/83.4	92.3	89.9	88.8	90.5	66.9	88.3	81.7
BERT-of-Theseus (Xu et al., 2020)	67.5M	51.1	82.3/-	91.5	89.5	89.0	89.6	**68.2**	88.7	81.2
CKD	67.5M	**55.1**	**83.6/84.1**	**93.0**	**90.5**	**89.6**	**91.2**	67.3	**89.0**	**82.4**

Table 3: Comparison of **task-specific** distillation on SQuAD dataset. The results of baselines and ours are reported by performing distillation with their objectives on the top of pre-trained 6-layer BERT (6/768) (Turc et al., 2019).

Model	#Params	SQuAD 1.1v EM	F1
$\text{BERT}_{\text{BASE}}$ (Teacher)	110M	81.3	88.6
PKD(Sun et al., 2019)	67.5M	77.1	85.3
PD(Turc et al., 2019)	67.5M	80.1	87.0
TinyBERT(Jiao et al., 2019)	67.5M	80.4	87.2
CKD	67.5M	**81.8**	**88.7**

DynaBERT (Hou et al., 2020) that involves flexible architectural changes in the training phase.

5 Experiments

We conduct task-agnostic and task-specific distillation experiments to elaborately compare our CKD with baseline distillation objectives. We then report on the performance gains achieved by our method for BERT architectures of various sizes and inserting our objective for training DynaBERT which can run at adaptive width and depth through pruning the attention heads or layers. Finally, we analyze the effect of each component in our CKD and the impact of leveraging locality δ for w_{ijk} in Eq. (1).

Dataset. For task-agnostic distillation which compresses a large pre-trained language model into a small language model on the pre-training stage, we use a document of English Wikipedia. For evaluating the compressed language model on the pre-training stage and task-specific distillation, we use the GLUE benchmark (Wang et al., 2018) which consists of nine diverse sentence-level classification tasks and SQuAD (Rajpurkar et al., 2016).

Setup. For task-agnostic distillation, we use the original BERT without fine-tuning as the teacher. Then, we perform the distillation on the student where the model size is pre-defined. We perform distillation using our proposed CKD objective with class probability matching of masked language modeling for 3 epochs while task-agnostic distillation following the Jiao et al. (2019) and keep other hyperparameters the same as BERT pre-training (Devlin et al., 2018). For task-specific distillation, we experiment with our CKD on top of pre-trained BERT models of various sizes which are released for research in institutions with fewer computational resources[2] (Turc et al., 2019). For the importance weight of each pair-wise and triple-wise terms, we leverage the locality of words, in that $w_{ij} = 1$ if $|i - j| \leq \delta$ and 0, otherwise. For this, we select the δ in (10-21). More details including hyperparameters are provided in Appendix B. The code to reproduce the experimental results is available at https://github.com/GeondoPark/CKD.

[2]https://github.com/google-research/bert

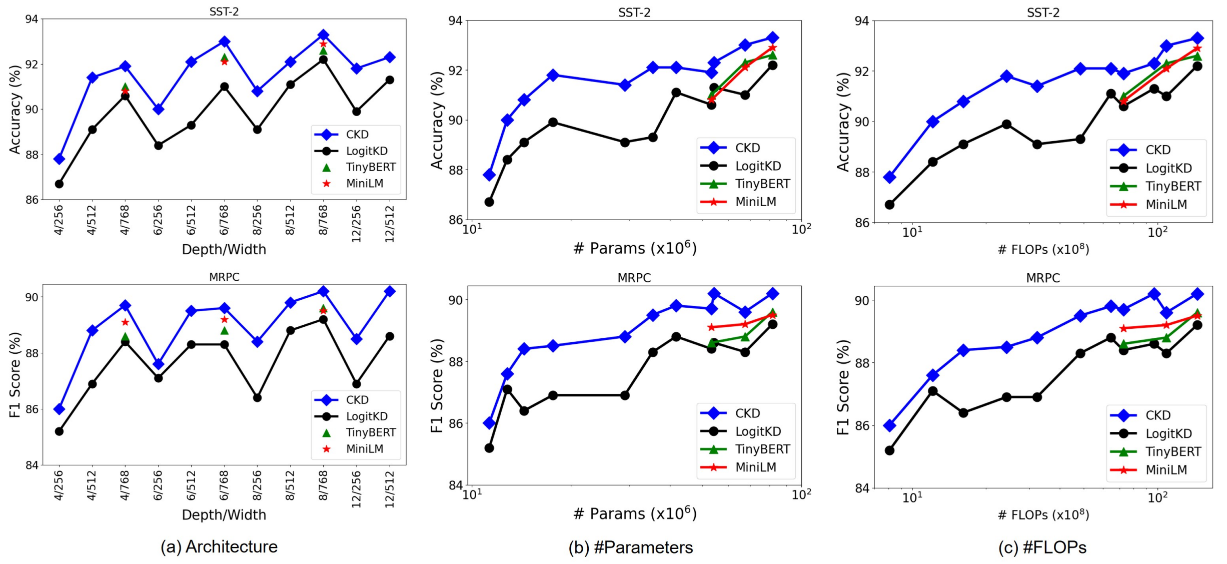


Figure 2: Task specific distillation on various sizes of models. We consider diverse cases by changing (a) the network structures, (b) the number of parameters and (c) the number of FLOPs. All results are averaged over 4 runs on the development set.

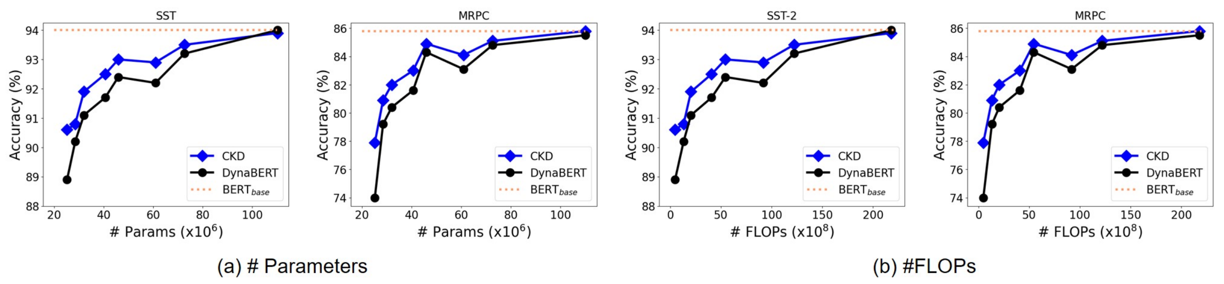


Figure 3: Boosting the performance of DynaBERT via training with CKD. Comparison between the original DynaBERT and CKD-augmented DynaBERT according to (a) the number of parameters and (b) the number of FLOPs. The results are averaged over 4 runs on the development set.

5.1 Main Results

To verify the effectiveness of our CKD objective, we compare the performance with previous distillation methods for BERT compression including task-agnostic and task-specific distillation. Following the standard setup in baselines, we use the $\text{BERT}_{\text{BASE}}$ (12/768)[3] as the teacher and 6-layer BERT (6/768) as the student network. Therefore, the student models used in all baselines and ours have the same number of parameters (67.5M) and inference FLOPs (10878M) and time.

Task-agnostic Distillation. We compare with three baselines: 1) Truncated BERT which drop top 6 layers from $\text{BERT}_{\text{base}}$ proposed in PKD (Sun et al., 2019), 2) $\text{BERT}_{\text{small}}$ which trained using the Masked LM objectives provided in PD (Turc et al., 2019), 3) TinyBERT (Jiao et al., 2019) which propose the individual word representation and attention map matching. Since MobileBERT (Sun et al., 2020) use the specifically designed teacher and student network which have 24-layers with an inverted bottleneck structure, we do not compare with. DistilBERT (Sanh et al., 2019) and MINI-LM (Wang et al., 2020) use the additional BookCorpus dataset which is no longer publicly available. Moreover, the authors do not release the code, making it hard to reproduce. Thus we do not compare in the main paper for a fair comparison. The comparisons with those methods are available in Appendix C. Results of task-agnostic distillation on GLUE dev sets are presented Table 1. The result shows that CKD surpasses all baselines. Comparing with TinyBERT which transfers the knowledge of individual representations, CKD outdoes in all scores except for the RTE. These results empirically demonstrate that distribution-based knowledge works better than individual representation knowledge.

[3]In notation (a/b), a means the number of layers and b denotes a hidden size in intermediate layers. The number of attention heads is defined as $b/64$.

Table 4: Ablation study about the impact of each component of CKD. '- *' denotes to eliminate *, the component of CKD.

Objectives	MNLI-(m/-mm) (Acc)	SST-2 (Acc)	QNLI (Acc)	MRPC (F1)	QQP (Acc)	STS-B (Spear)
CKD	**80.7/80.8**	**91.4**	**88.1**	**88.8**	**90.3**	**87.9**
- WR	80.1/80.6	90.6	87.5	88.5	89.7	87.5
- LTR	79.9/80.3	91.1	87.8	88.3	**90.3**	87.6
- WR - LTR	79.2/79.9	89.1	87.4	88.1	89.2	86.8

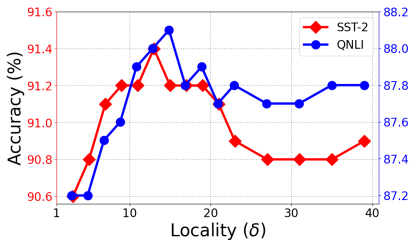


Figure 4: Effect of local window size.

Task-specific Distillation. Here, we compare with four baselines that do not perform distillation in the pre-training: 1) PD (Turc et al., 2019) which do pre-training with Masked LM and distills with Logit KD in task-specific fine-tuning process. 2) PKD (Sun et al., 2019) which uses only 6 layers below BERT$_{\text{base}}$, and distillation is also performed only in task-specific fine-tuning. The GLUE results on dev sets of PKD are taken from (Xu et al., 2020). 3) TinyBERT (Jiao et al., 2019). For the TinyBERT, we also perform distillation only in the task-specific fine-tuning with their objectives on the top of the pre-trained model provided by Turc et al. (2019) for a fair comparison. 4) BERT-of-Theseus (Xu et al., 2020) which learn a compact student network by replacing the teacher layers in a fine-tuning stage. Results of task-specific distillation on GLUE dev sets and SQuAD datasets are presented in Table 2 and 3, respectively. Note that briefly, the CKD also outperforms all baselines for all GLUE datasets and SQuAD dataset except for RTE for task-specific distillation, convincingly verifying its effectiveness. These results consistently support that contextual knowledge works better than other distillation knowledge.

5.2 Effect of CKD on various sizes of models

For the knowledge distillation with the purpose of network compression, it is essential to work well in more resource-scarce environments. To this end, we further evaluate our method on various sizes of architectures. For this experiment, we perform distillation on a task-specific training process on top of various size pre-trained models provided by Turc et al. (2019). We compare CKD with three baselines: 1) LogitKD objective used by Sanh et al. (2019); Turc et al. (2019). 2) TinyBERT (Jiao et al., 2019) objective which includes individual word representations and attention matrix matching. 3) MINI-LM (Wang et al., 2020) objective which includes attention matrix and value-value relation matching. We implement the baselines and runs for task-specific distillation. We note that MINI-LM and TinyBERT objective are applicable only to models (*/768) which have the same number of attention heads with the teacher model (12/768). Figure 2 illustrate that our CKD consistently exhibits significant improvements in the performance compared LogitKD. In addition, for task-specific distillation, we show that CKD works better than all baselines on (*/768) student models. The results on more datasets are provided in Appendix E.

5.3 Incorporating with DynaBERT

DynaBERT (Hou et al., 2020) is a recently proposed adaptive-size pruning method that can run at adaptive width and depth by removing the attention heads or layers. In the training phase, DynaBERT uses distillation objectives which consist of LogitKD and individual word representations matching to improve the performance. Since the CKD objective has no constraints about architecture such as embedding size or the number of attention heads, we validate the objective by replacing it with CKD. The algorithm of DynaBERT and how to insert CKD are provided in Appendix D. To observe just how much distillation alone improves performance, we do not use data augmentation and an additional fine-tuning process. We note that objectives proposed in MINI-LM (Wang et al., 2020) and TinyBERT (Jiao et al., 2019) cannot be directly applied due to constraints of the number of attention heads. As illustrated in Figure 3, CKD consistently outperforms the original DynaBERT on dynamic model sizes, supporting the claim that distribution-based knowledge is more helpful than individual word representation knowledge. The results on more datasets are provided in Appendix E.

5.4 Ablation Studies

We provide additional ablation studies to analyze the impact of each component of the CKD and the introduced locality ($w_{i,j} = \delta$) in Eq. (1) as the weight of how important each pair-wise and triple-wise term is. For these studies, we fix the student network with 4-layer BERT (4/512) and report the results as an average of over 4 runs on the development set.

Impact of each component of CKD. The proposed CKD transfers the word relation based and layer transforming relation based contextual knowledge. To isolate the impact on them, we experiment successively removing each piece of our objective. Table 4 summarizes the results, and we observe that WR and LTR can bring a considerable performance gain when they are applied together, verifying their individual effectiveness.

Locality as the importance of relation terms. We introduced the additional weights (w_{ij}, w_{ijk}) in Eq. (1) for CKD-WR (and similar ones for CKD-LTR) to control the importance of each pair-wise and triple-wise term and suggested using the locality for them as one possible way. Here, we verify the effect of locality by increasing the local window size (δ) on the SST-2 and QNLI datasets. The result is illustrated in Figure 4. We observe that as the local window size increases, the performance improves, but after some point, the performance is degenerated. From this ablation study, we set the window size (δ) between 10-21.

6 Conclusion

We proposed a novel distillation strategy that leverages contextual information efficiently based on word relation and layer transforming relation. To our knowledge, we are the first to apply this contextual knowledge which is studied to interpret the language models. Through various experiments, we show not only that CKD outperforms the state-of-the-art distillation methods but also the possibility that our method boosts the performance of other compression methods.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grants (2018R1A5A1059921, 2019R1C1C1009192) and Institute of Information & Communications Technology Planning & Evaluation (IITP) grants (No.2017-0-01779, A machine learning and statistical inference framework for explainable artificial intelligence, No.2019-0-01371, Development of brain-inspired AI with human-like intelligence, No.2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)) funded by the Korea government (MSIT).

References

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. *arXiv preprint arXiv:1908.04211*.

Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2019. Transformer to cnn: Label-scarce distillation for efficient text classification. *arXiv preprint arXiv:1909.03508*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

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

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*.

Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, Dongsheng Li, Nan Yang, and Ming Zhou. 2018. Attention-guided answer distillation for machine reading comprehension. *arXiv preprint arXiv:1808.07644*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut.

2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. 2020. Fastbert: a self-distilling bert with adaptive inference time. *arXiv preprint arXiv:2004.02178*.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems*, pages 8594–8603.

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

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

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

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Jesse Vig. 2019. Visualizing attention in transformer-based language representation models. *arXiv preprint arXiv:1904.02679*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

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. *arXiv preprint arXiv:2002.10957*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Rui Xia and Chengqing Zong. 2010. Exploring the use of word relation features for sentiment classification. In *Coling 2010: Posters*, pages 1336–1344.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. *arXiv preprint arXiv:2002.02925*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

A Explanation of previous methods and their constraints

Table 5 present the details of knowledge distillation objectives of previous methods and their constraints.

DistilBERT (Sanh et al., 2019) uses logit distillation loss (Logit KD), masked language modeling loss, and cosine loss between the teacher and student word representations in the learning process. The cosine loss serves to align the directions of the hidden state vectors of the teacher and student. Since the cosine of the two hidden state vectors is calculated in this process, they have the constraint that the embedding size of the teacher and the student model must be the same.

PKD (Sun et al., 2019) transfers teacher knowledge to the student with Logit KD and patient loss. The patient loss is the mean-square loss between the normalized hidden states of the teacher and student. To calculate the mean square error between the hidden states, they have a constraint that the dimensions of hidden states must be the same between teacher and student.

TinyBERT (Jiao et al., 2019) uses additional loss that matches word representations and attention matrices between the teacher and student. Although they acquire flexibility on the embedding size, using an additional parameter, since the attention matrices of the teacher and student are matched through mean square error loss, the number of attention heads of the teacher and student must be the same.

MobileBERT (Sun et al., 2020) utilizes a similar objective with TinyBERT (Jiao et al., 2019) for task-agnostic distillation. However, since they match the hidden states with $l2$ distance and attention matrices with $\mathcal{KL}$ divergence between teacher and student, they have restrictions on the size of hidden states and the number of attention heads.

MiniLM (Wang et al., 2020) proposes distilling the self-attention module of the last Transformer layer of the teacher. In self-attention module, they transfer attention matrices such as TinyBERT and MobileBERT and Value-Value relation matrices. Since they match the attention matrices of the teacher and student in a one-to-one correspondence, the number of attention heads of the teacher and student must be the same.

The methods introduced in Table 5 have constraints by their respective knowledge distillation objectives. However, our CKD method which utilizes the relation statistics between the word representations (hidden states) has the advantage of not having any constraints on student architecture.

B Details of experiment setting

This section introduces the experimental setting in detail. We implemented with PyTorch framework and huggingface's transformers package (Wolf et al., 2019).

Task-agnostic distillation We use the pre-trained original BERT$_{\text{base}}$ with masked language modeling objective as the teacher and a document of English Wikipedia as training data. We set the max sequence length to 128 and follow the preprocess and WordPiece tokenization of Devlin et al. (2018). Then, we perform the distillation for 3 epochs. For the pre-training stage, we use the CKD objective with class probability matching of masked language modeling and keep other hyper-parameters the same as BERT pre-training (Devlin et al., 2018).

Task-specific distillation Our contextual knowledge distillation proceeds in the following order. First, from pre-trained BERT$_{\text{base}}$, task-specific fine-tuning is conducted to serve as a teacher. Then, prepare the pre-trained small-size architecture which serves as a student. In this case, pre-trained models of various model sizes provided by Turc et al. (2019) are employed. Finally, task-specific distillation with our CKD is performed.

To reduce the hyperparameters search cost, λ_{WR} in Eq. (1) and λ_{LTR} in Eq. (3) are used with same value. For the importance weights introduced for pair-wise and triple-wise terms, the locality is applied only to the importance weight w of the word relation (WR)-based CKD loss. The importance weight w of the layer transforming relation (LTR)-based CKD loss is set to 1. In this paper, we report the best result among the following values to find the optimal hyperparameters of each dataset:

- Alpha (α) : 0.7, 0.9
- Temperature (T) : 3, 4
- λ_{WR}, λ_{LTR} : 1, 10, 100, 1000
- λ_{CKD} : 1, 10, 100, 1000

Other training configurations such as batch size, learning rate and warm up proportion are used following the BERT (Devlin et al., 2018).

Table 5: Overview of distillation objectives used for language model compression and their constraint on architecture. $\mathbb{S}_k$ means scaled softmax function across the kth-dimension.

	Knowledge Distillation Objectives	**Constraint**
DistilBERT (Sanh et al., 2019)	$\sum_{i=1}^{n} \cos(r_{l,i}^{t}, r_{l,i}^{s}),\ \mathcal{L}_{Logit}^{\mathrm{D}}$	Embedding size
PKD (Sun et al., 2019)	$\sum_{i=1}^{n} \Big[\mathtt{MSE}(\frac{r_{l,i}^{t}}{\lVert r_{l,i}^{t}\rVert_2} - \frac{r_{l,i}^{s}}{\lVert r_{l,i}^{s}\rVert_2})\Big],\ \mathcal{L}_{Logit}^{\mathrm{D}}$	Embedding size
TinyBERT (Jiao et al., 2019)	$\sum_{i=1}^{n} \Big[\mathtt{MSE}(r_{l,i}^{t} - W_r r_{l,i}^{s})\Big], \sum_{h=1}^{H} \Big[\mathtt{MSE}(\boldsymbol{A}_{l,h}^{t} - \boldsymbol{A}_{l,h}^{s})\Big],\ \mathcal{L}_{Logit}^{\mathrm{D}}$	Attention head
Mobile-BERT (Sun et al., 2020)	$\sum_{i=1}^{n} \Big[\mathtt{MSE}(r_{l,i}^{t} - r_{l,i}^{s})\Big], \sum_{h=1}^{H} \Big[\mathbb{KL}\big(\boldsymbol{A}_{l,h}^{t}, \boldsymbol{A}_{l,h}^{s}\big)\Big],\ \mathcal{L}_{Logit}^{\mathrm{D}}$	Embedding size Attention head
MiniLM (Wang et al., 2020)	$\sum_{h=1}^{H} \Big[\mathbb{KL}\big(\boldsymbol{A}_{l,h}^{t}, \boldsymbol{A}_{l,h}^{s}\big)\Big], \sum_{h=1}^{H} \Big[\mathbb{KL}\big(\mathbb{S}_2(\boldsymbol{V}_{l,h}^{t} \cdot {\boldsymbol{V}_{l,h}^{t}}^{T}), \mathbb{S}_2(\boldsymbol{V}_{l,h}^{s} \cdot {\boldsymbol{V}_{l,h}^{s}}^{T})\Big]$	Attention head

Table 6: Full comparison of task-agnostic distillation comparing our CKD against the baseline methods. For the task-agnostic distillation comparison, we do not use task-specific distillation for a fair comparison. The results of TinyBERT cited as reported by Wang et al. (2020). Other results are as reported by their authors. Results of the development set are averaged over 4 runs. "-" means the result is not reported and the trained model is not released. † marks our runs with the officially released model.

Model	#Params	CoLA (Mcc)	MNLI-(m/-mm) (Acc)	SST-2 (Acc)	QNLI (Acc)	MRPC (F1)	QQP (Acc)	RTE (Acc)	STS-B (Spear)
BERT$_{\text{BASE}}$ (Teacher)	110M	60.4	84.8/84.6	94.0	91.8	90.3	91.4	70.4	89.5
Truncated BERT (Sun et al., 2019)	67.5M	41.4	81.2/-	90.8	87.9	82.7	90.4	65.5	-
BERT$_{\text{Small}}$ (Turc et al., 2019)	67.5M	47.1†	81.1/81.7	91.1	87.8	87.9	90.0	63.0	87.5†
DistilBERT (Sanh et al., 2019)	67.5M	51.3	82.2/-	91.3	89.2	87.5	88.5	59.9	86.9
TinyBERT (Jiao et al., 2019)	67.5M	42.8	83.5/83.2†	91.6	90.5	88.4	90.6	**72.2**	88.5†
MINI-LM (Wang et al., 2020)	67.5M	49.2	**84.0**/-	92.0	**91.0**	88.4	**91.0**	71.5	-
CKD	67.5M	**52.7**	83.5/**83.4**	**92.4**	90.7	**89.1**	90.8	70.1	**89.1**

C Additional comparison on task-agnostic distillation

We report the fair comparison of our method and baselines about the task-agnostic distillation in Section 5.1 of the main paper. Several works (Sanh et al., 2019; Wang et al., 2020) use the additional BookCorpus dataset which is no longer publicly available. Here, we present the full comparison of CKD and baselines including DistilBERT (Sanh et al., 2019) and MINI-LM (Wang et al., 2020). As shown in Table 6, even though we do not use the BookCorpus dataset, we outperform all baselines on four datasets and obtain comparable performance on the rest of the datasets.

D Applying CKD to DynaBERT

In this section, we describe how we apply our CKD objective to DynaBERT (Hou et al., 2020). Training DynaBERT consists of three stages: 1) Rewire the model according to the importance and then 2) Go through the two-stage of adaptive pruning with distillation objective. Since we suppress some details of DynaBERT for clarity, refer to the paper (Hou et al., 2020) for more information.

We summarize the training procedure of DynaBERT with CKD in algorithm 1. To fully exploit the capacity, more important attention heads and neurons must be shared more across the various sub-networks. Therefore, we follow phase 1 in DynaBERT to rewire the network by calculating the loss and estimating the importance of each attention head in the Multi-Head Attention (MHA) and neuron in the Feed-Forward Network (FFN) based on gradients. Then, they train the DynaBERT by accumulating the gradient varying the width and depth of BERT. In these stages, they utilize distillation objective which matches hidden states and logits to improve the performance. We apply our CKD at these stages by replacing their objective with CKD as shown in algorithm 1 (Blue). Since CKD has no restrictions on student's architecture, it can be easily applied.

E More Results

Due to space limitations in the main paper, we only report the results on a subset of GLUE datasets for experiments about the effect of model size for CKD and boosting the DynaBERT with CKD. Here, we report all datasets of GLUE except for CoLA for two experiments. We exclude the CoLA dataset since the distillation losses are not converged properly in the very small-size models.

Here, we present the results of three experiments

Algorithm 1: Train DynaBERT with CKD

```
Phase 1: Rewire the network.
    input : Development set, trained BERT on downstream task.
    Calculate the importance of attention heads and neurons with gradients.
    Rewire the network according to the importance.
Phase 2: Train DynaBERT$_W$ with adaptive width.
    input : Training set, width multiplier list $widthList$.
    initialize a fixed teacher model and a trainable student model with rewired net.
    for $iter = 1, \dots, T_{train}$ do
        Get the logits $\boldsymbol{y}$, hidden states $\boldsymbol{R}$ from teacher model.
        for width multiplier $m_w$ in $widthList$ do
            Get the logits $\boldsymbol{y}^{(m_w)}$, hidden states $\boldsymbol{R}^{(m_w)}$ from student model.
            Compute distillation loss.
            $\mathcal{L}_{\text{DynaBERT}} = \text{SCE}(\boldsymbol{y}^{(m_w)}, \boldsymbol{y}) + \lambda_1 \cdot \sum_{l=0}^{L} \text{MSE}(\boldsymbol{R}_l^{(m_w)}, \boldsymbol{R}_l)$
            $\mathcal{L}_{\text{CKD}} = \text{SCE}(\boldsymbol{y}^{(m_w)}, \boldsymbol{y}) + \lambda_1 \cdot \mathcal{L}_{\text{CKD-WR}}(\boldsymbol{R}^{(m_w)}, \boldsymbol{R}) + \lambda_2 \cdot \mathcal{L}_{\text{CKD-LTR}}(\boldsymbol{R}^{(m_w)}, \boldsymbol{R})$
            Accumulate gradients $\mathcal{L}.backward()$.
        Update with the accumulated gradients.
Phase 3: Train DynaBERT with adaptive width and depth.
    input : Training set, width multiplier list $widthList$, depth multiplier list $depthList$.
    initialize a fixed DynaBERT$_W$ as teacher model and a trainable student model with the DynaBERT$_W$.
    for $iter = 1, \dots, T_{train}$ do
        for width multiplier $m_w$ in $widthList$ do
            Get the logits $\boldsymbol{y}^{(m_w)}$, $\boldsymbol{R}^{(m_w)}$ from teacher model.
            for depth multiplier $m_d$ in $depthList$ do
                Get the logits $\boldsymbol{y}^{(m_w,m_d)}$, hidden states $\boldsymbol{R}^{(m_w,m_d)}$ from student model.
                Compute distillation loss.
                $\mathcal{L}_{\text{DynaBERT}} = \text{SCE}(\boldsymbol{y}^{(m_w,m_d)}, \boldsymbol{y}^{(m_w)}) + \lambda_1 \cdot \sum_{l,l' \in L_S, L_T} \text{MSE}(\boldsymbol{R}_l^{(m_w,m_d)}, \boldsymbol{R}_{l'}^{(m_w)})$
                $\mathcal{L}_{\text{CKD}} = \text{SCE}(\boldsymbol{y}^{(m_w,m_d)}, \boldsymbol{y}^{(m_w)}) + \lambda_1 \cdot \mathcal{L}_{\text{CKD-WR}}(\boldsymbol{R}^{(m_w,m_d)}, \boldsymbol{R}^{(m_w)})$
                $+\lambda_2 \cdot \mathcal{L}_{\text{CKD-LTR}}(\boldsymbol{R}^{(m_w,m_d)}, \boldsymbol{R}^{(m_w)})$
                Accumulate gradients $\mathcal{L}.backward()$.
        Update with the accumulated gradients.
```

on additional datasets in order. 1) Effect of CKD on various sizes of models. 2) Boosting the performance of DynaBERT when CKD is applied.

Effect of CKD on various sizes of models. Figure 5 illustrates the performance of task-specific distillation on various sizes of models. Again, we note that MINI-LM and TinyBERT objectives are applicable only to models (*/768), which have the same number of attention heads as the teacher model (12/768). As shown in Figure 5, our CKD consistently exhibits significant improvements in the performance compared LogitKD for all model sizes. Compared to TinyBERT and MINI-LM, CKD shows higher performance on all datasets for almost all model sizes (*/768).

Incorporating with DynaBERT Figure 6 shows the performance of the original DynaBERT and when CKD is applied. As illustrated in Figure 6, CKD further improves the original DynaBERT on dynamic width and depth size, convincingly verifying its effectiveness. The results also present the possibility that our method boosts the performance of other compression methods.

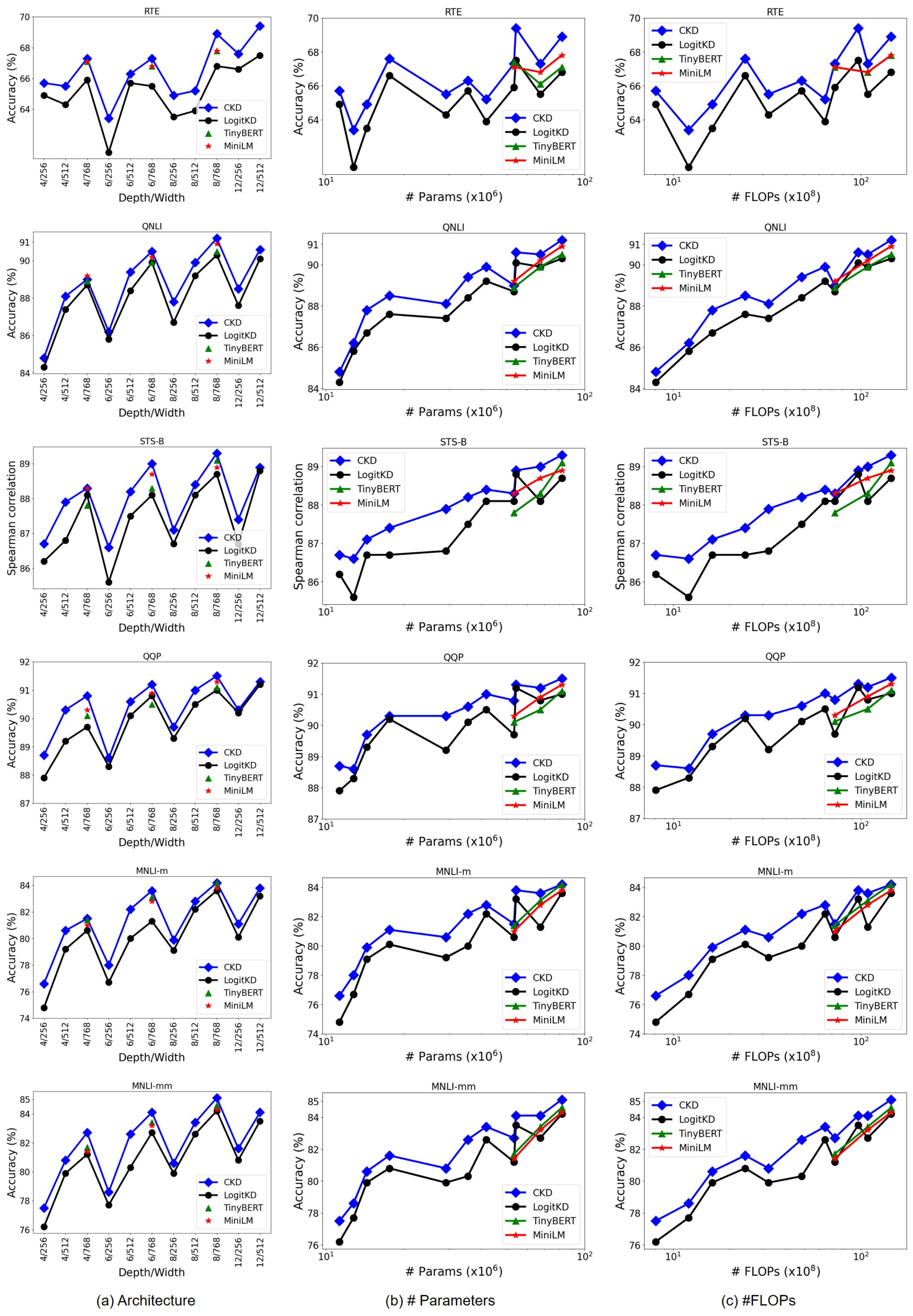


Figure 5: The efficiency of various sizes of models for CKD compared to baselines. The performance graph according to (a) network structure (b) the number of parameters (c) the number of FLOPs. The results are averaged over 4 runs on the development set.

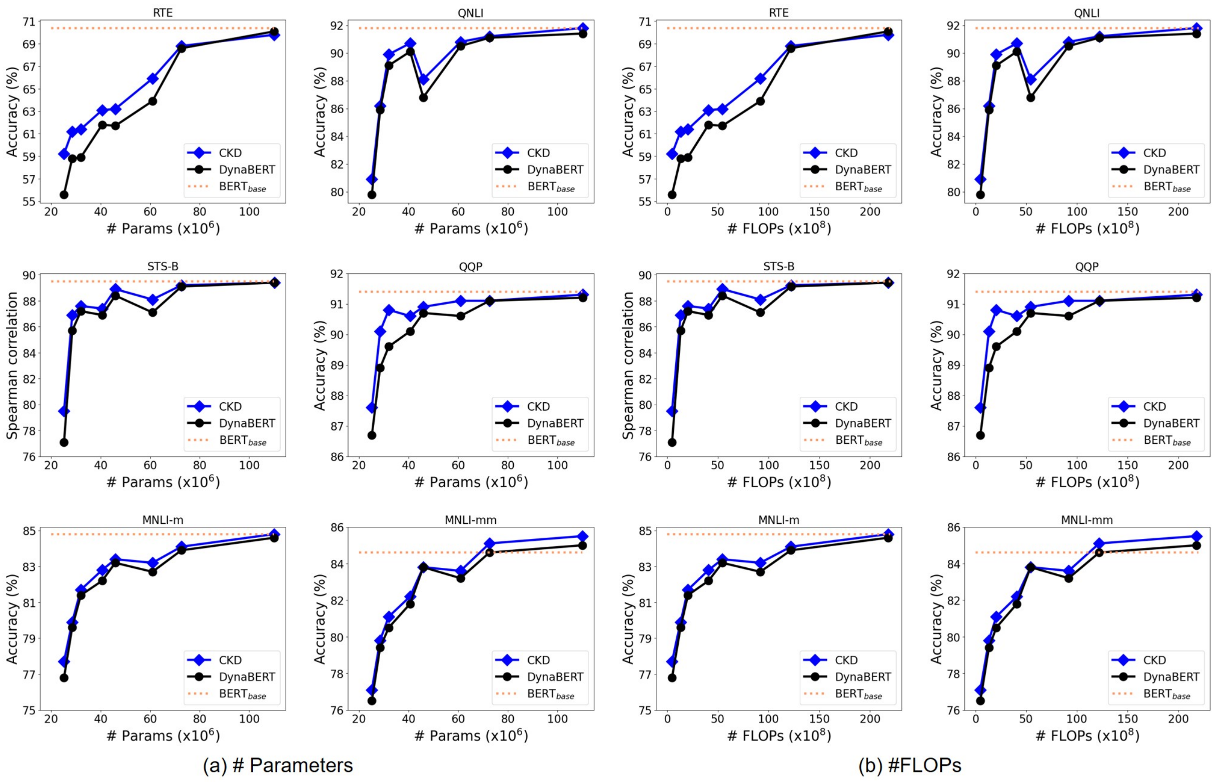


Figure 6: Boosting the performance of DynaBERT when CKD is applied. The performance graph for comparison of original DynaBERT and CKD according to (a) the number of parameters and (b) the number of FLOPs. The results are averaged over 4 runs on the development set.