General Cross-Architecture Distillation of Pretrained Language Models into Matrix Embeddings

Lukas Galke
Max Planck Institute for Psycholinguistics
Nijmegen, Netherlands
lukas.galke@mpi.nl

Isabelle Cuber, Christoph Meyer, Henrik Ferdinand Nölscher,
Angelina Sonderecker, Ansgar Scherp
University of Ulm, Germany
{isabelle.cuber,christoph-1.meyer,henrik-1.noelscher,angelina.sonderecker,
ansgar.scherp}@uni-ulm.de

2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9892144

***Abstract*—Large pretrained language models (PreLMs) are revolutionizing natural language processing across all benchmarks. However, their sheer size is prohibitive for small laboratories or for deployment on mobile devices. Approaches like pruning and distillation reduce the model size but typically retain the same model architecture. In contrast, we explore distilling PreLMs into a different, more efficient architecture, Continual Multiplication of Words (CMOW), which embeds each word as a matrix and uses matrix multiplication to encode sequences. We extend the CMOW architecture and its CMOW/CBOW-Hybrid variant with a bidirectional component for more expressive power, per-token representations for a general (task-agnostic) distillation during pretraining, and a two-sequence encoding scheme that facilitates downstream tasks on sentence pairs, such as sentence similarity and natural language inference. Our matrix-based bidirectional CMOW/CBOW-Hybrid model is competitive to DistilBERT on question similarity and recognizing textual entailment, but uses only half of the number of parameters and is three times faster in terms of inference speed. We match or exceed the scores of ELMo for all tasks of the GLUE benchmark except for the sentiment analysis task SST-2 and the linguistic acceptability task CoLA. However, compared to previous cross-architecture distillation approaches, we demonstrate a doubling of the scores on detecting linguistic acceptability. This shows that matrix-based embeddings can be used to distill large PreLM into competitive models and motivates further research in this direction.**

I. Introduction

Large pretrained language models [1, 2] (PreLMs) have emerged as de-facto standard methods for natural language processing [3, 4]. The common strategy is to pretrain models on enormous amounts of unlabeled text before fine-tuning them for downstream tasks. However, the drawback of PreLMs is that the models are becoming larger and larger with up to several billion parameters [5]. This comes with high environmental and economic costs [6] and puts development and research in the hands of a few global players only [7, pp. 10-12]. Even though a single pretrained model can be reused for multiple downstream tasks, the sheer model size is often prohibitive. The immense resource requirements prevent the use of these models in small-scale laboratories and on mobile devices, which is tied to privacy concerns [8].

There is a need for more efficient models or compressed versions of large models to make AI research more inclusive and energy-friendly while fostering deployment in applications. Reducing the size of PreLMs using knowledge distillation [9] or model compression [10] is an active area of research [11, 12, 13]. It is reported that companies, such as Google, use distillation of PreLM to deploy their large models for productive use, i. e., for services that have strong requirements in terms of low latency.[1] Both knowledge distillation and model compression can be described as teacher-student setups [9, 10]. The student is trained to imitate the predictions of the teacher while using less resources. Typically, a large PreLM takes the role of the teacher while the student is a smaller version of the same architecture. Sharing the same architecture between the student and the teacher enables the use of dedicated distillation techniques, e. g., aligning the representations of intermediate layers [11, 13].

However, using more efficient architectures as student has already shown promising results, such as the task-specific distillation approaches by Tang et al. [14] and Wasserblatt et al. [15]. In their works, the student models are LSTMs [16] or models based on a continuous bag-of-words representation (CBOW) [17, 18]. On the one hand, LSTMs are difficult to parallelize as they need at least $\mathcal{O}(n)$ sequential steps to encode a sequence of length n. On the other hand, CBOW-based models are *not order-aware*, i. e., cannot distinguish sentences with the same words but in different order ("cat eats mouse" vs. "mouse eats cat" are treated equivalent). There are, however, efficient models such as Mai et al.'s continual multiplication of words (CMOW) that *do capture word order* by representing each token as a matrix [19], instead of a vector as in CBOW. A sequence in CMOW is modeled by the non-commutative matrix multiplication [20], which makes the encoding of a sequence dependent on the word order. We denote such models as *matrix embeddings*.

We extend Mai et al.'s work and investigate how order-aware matrix embeddings can be used as student models in *cross-architecture distillation* from large PreLM teachers. This complements the existing body of works that focused predominantly on *same-architecture distillation* (see discussion in Section II-B). Furthermore, all previous cross-architecture distillation approaches are task-specific, whereas we also explore general distillation. We aim to understand to what

[1] J. Devlin, "Contextual Word Representations with BERT and Other Pre-trained Language Models", https://web.stanford.edu/class/cs224n/slides/Jacob_Devlin_BERT.pdf

extent order-aware embeddings are suitable to capture the teacher signal of a large PreLM such as BERT [1]. To this end, we extend Mai et al.'s CMOW/CBOW-Hybrid model [19], which is a hybrid variant unifying the strength of CBOW and CMOW, with a bidirectional representation of the sequences. Furthermore, we add the ability to emit per-token representations to facilitate the use of a modern masked language model objective [1].

We investigate both task-agnostic *general distillation*, i. e., the distillation is applied during pretraining on unlabeled text, and *task-specific distillation*, when an already fine-tuned PreLM is distilled per task. We further introduce a two-sentence encoding scheme to CMOW so that it can deal with sentence similarity and natural language inference tasks.

Our results show that large PreLMs can be distilled into efficient order-sensitive embedding models, achieving a performance that is competitive to ELMo [21] on the GLUE benchmark. On the QQP and RTE tasks, embedding-based models even challenge other size-reduced BERT models such as DistillBERT. In summary, our contributions are:

- We extend order-aware embedding models with bidirection and make them amenable for masked language model pretraining.
- We explore using order-aware embedding models as student models in a cross-architecture distillation setup with BERT as a teacher and compare general and task-specific distillation.
- We introduce the first encoding scheme that enables CMOW/CBOW-Hybrid to deal with two-sentence tasks (20% increase over the naive approach).
- Our results show that the best distilled embedding models are on-par with more expensive models such as ELMo. We outperform DistilBERT on the QQP and RTE tasks of the GLUE benchmark, while having a much higher encoding speed (thrice as high as DistilBERT).

Below, we introduce our embedding models, our cross-architecture distillation setup, and our two-sequence encoding scheme. The experimental procedure is described in Section IV. The results are reported in Section V and discussed in Section VI, where we also relate our work to the literature.

II. Problem Formulation

We study the problems of transfer learning and knowledge distillation, which we briefly describe below.

A. Transfer Learning: Pretraining and Fine-Tuning

In a pretraining stage, a language model is trained on large amounts of unlabeled text in a self-supervised manner, e. g., by predicting left-out words. The pre-trained model is then transferred to downstream tasks. The parameters of the transferred model are fine-tuned to the respective task. For each task, a fresh copy of the pretrained model is used as a starting point.

- Step 1: Pretraining. Train model f_θ using unlabeled text.
- Step 2: Fine-tuning. For each downstream task $\mathcal{T}$, continue training the model f_θ from pretraining. The model f_θ is extended by a task-specific classification head, which is initialized randomly.

The task $\mathcal{T}$ can be an arbitrary supervised downstream tasks, where paired training data is available. In particular, that includes two-sequence tasks such as natural language inference or textual similarity. The performance measure depends on the respective task. We will use the tasks from the GLUE benchmark [3] to evaluate our models.

B. Knowledge Distillation

The problem of knowledge distillation [9] or model compression [10] refers to learning a smaller model g that imitates the behavior of a larger model f, such that desirably $g(\boldsymbol{x}) \approx f(\boldsymbol{x})$. The smaller model g is called the student and the larger model f is called the teacher.

The distillation is carried out by aligning the teacher's and student's output, e. g., via a loss term $\mathcal{L}(f(\boldsymbol{x}), g(\boldsymbol{x}))$, which we call the teacher signal. There are more techniques to foster distillation such as using the teacher's weights as initialization for the student [11, 12, 13], but those are only applicable when teacher and student are of the same model architecture. When student and teacher models have a different architecture, we call that cross-architecture distillation.

To contextualize knowledge distillation with transfer learning, we adopt the distinction between general distillation and task-specific distillation from Tang et al. [14].

- General distillation: The distillation is carried out *only* in the (self-supervised) pretraining stage. After pretraining, the teacher is not needed anymore. The student can be fine-tuned to downstream tasks independently.
- Task-specific distillation: In task-specific distillation, the teacher model can be consulted during fine-tuning for each of the downstream tasks. The respective task's training data are used for (supervised) distillation.

Note that the model for task-specific distillation can still be initialized with a model obtained by general distillation. However, this also falls under task-specific distillation, because, after all, the teacher has to be consulted during fine-tuning.

Distinguishing between general and task-specific distillation is important because this affects how the methods can be applied in practice. Imagine that we want to fine-tune a model for a downstream task on a mobile device. A model from general distillation would be able to learn new tasks on its own. With a task-specific distillation approach, the larger teacher model would need to be consulted on the mobile device. Thus, both approaches differ in how they can be applied in practice and should be considered separately.

III. Methods

First, we introduce our bidirectional extension to the CMOW/CBOW-Hybrid model. Subsequently, we introduce our approach for cross-architecture distillation that we use during the pretraining and fine-tuning stages. Finally, we introduce a two-sentence encoding scheme for order-aware embedding models that is crucial for fine-tuning on downstream tasks with paired sentences.

A. Extending the Order-Aware Embedding Models

We extend the CMOW/CBOW-Hybrid embeddings of Mai et al. [19] with bidirection and the ability to emit per-token representations as a foundation for cross-architecture distillation. CMOW/CBOW-Hybrid embeddings, our baseline model, are a combination of matrix embeddings and vector embeddings. Compared to vector-only embeddings, the word order can be captured because matrix multiplication is non-commutative. Given a sequence s of n tokens with each token s_j having its corresponding matrix-space embedding $\boldsymbol{X}_j \in \mathbb{R}^{d \times d}$ and vector-space embedding $\boldsymbol{x}_j \in \mathbb{R}^{d_{\text{vec}}}$, the CMOW/CBOW-Hybrid embedding of a sequence of length n is the multiplication of embedding matrices $\boldsymbol{X}_i$ concatenated (symbol $\cdot||\cdot$) to the sum of embedding vectors $\boldsymbol{x}_i$:

$$\begin{aligned}
\boldsymbol{H}^{(\text{CMOW})} &:= \boldsymbol{X}_1^{(\text{CMOW})} \cdot \boldsymbol{X}_2^{(\text{CMOW})} \cdots \boldsymbol{X}_n^{(\text{CMOW})} \\
\boldsymbol{h}^{(\text{CBOW})} &:= \sum_{1 \leq j \leq n} \boldsymbol{x}_j^{(\text{CBOW})} \\
\boldsymbol{h}^{(\text{Hybrid})} &:= \text{flatten}\left(\boldsymbol{H}^{(\text{CMOW})}\right) \,||\, \boldsymbol{h}^{(\text{CBOW})}
\end{aligned}$$

where flatten collapses the matrix into a vector. The original work on CMOW [19] has extensively analyzed the CMOW and CBOW components individually and found that joint training is generally preferable. The CMOW and CBOW components can have different dimensionalities because they are combined by concatenation. We initialize each matrix $\boldsymbol{X}_j$ as identity plus Gaussian noise $\boldsymbol{I}_d + \mathcal{N}(0, \sigma_{\text{init}}^2)$ with $\sigma_{\text{init}} = 0.01$.

a) Proposed Model: Bidirectional CMOW/CBOW-Hybrid: Inspired by the success of bidirectionality in RNNs [22], LSTMs [21], and Transformers [1], we extend CMOW by a bidirectional component. Hence, we introduce a second set of matrix-space embeddings that are multiplied in reverse order. We then have one matrix embedding for the forward direction $\mathbf{X}^{(\text{fw})} \in \mathbb{R}^{n_{\text{vocab}} \times d \times d}$ and one for the backward direction $\mathbf{X}^{(\text{bw})} \in \mathbb{R}^{n_{\text{vocab}} \times d \times d}$. Then we concatenate forward and backward directions. Figure 1 illustrates bidirectional CMOW.

Furthermore, we emit one representation per token position i, which allows training with a masked language model objective [1]. Thus, we are able to make use of the BERT teacher signal for pretraining. Since we can reuse computations, $\mathcal{O}(n)$ matrix multiplications are sufficient to encode a sequence of length n. For these intermediate representations, we also modify the CBOW component in a way that it yields partial sums for the forward and backward directions. Formally, we compute the CMOW/CBOW-Hybrid representation as follows:

$$\begin{aligned}
\boldsymbol{H}_i^{(\text{Bidi. CMOW})} &:= \boldsymbol{X}_1^{(\text{fw})} \cdots \boldsymbol{X}_i^{(\text{fw})} \,||\, \boldsymbol{X}_n^{(\text{bw})} \cdots \boldsymbol{X}_i^{(\text{bw})} \\
\boldsymbol{h}_i^{(\text{Bidi. CBOW})} &:= \sum_{j=1}^{i} \boldsymbol{x}_j^{(\text{CBOW})} \,||\, \sum_{j=i}^{n} \boldsymbol{x}_j^{(\text{CBOW})} \\
\boldsymbol{h}_i^{(\text{Bidi. Hybrid})} &:= \text{flatten}\left(\boldsymbol{H}_i^{(\text{Bidi. CMOW})}\right) \,||\, \boldsymbol{h}_i^{(\text{Bidi. CBOW})}
\end{aligned}$$

For fine-tuning on tasks with full sentences as input, e. g., natural language inference, we do not need per-token representations. In this case, we compute the representation of the full token sequence as follows:

$$\begin{aligned}
\boldsymbol{H}^{(\text{Bidi. CMOW})} &:= \boldsymbol{X}_1^{(\text{fw})} \cdot \boldsymbol{X}_2^{(\text{fw})} \cdots \boldsymbol{X}_n^{(\text{fw})} \,||\, \\
&\qquad \boldsymbol{X}_n^{(\text{bw})} \cdot \boldsymbol{X}_{n-1}^{(\text{bw})} \cdots \boldsymbol{X}_1^{(\text{bw})} \\
\boldsymbol{h}^{(\text{CBOW})} &:= \sum_{j=1}^{n} \boldsymbol{x}_j^{(\text{CBOW})} \\
\boldsymbol{h}^{(\text{Bidi. Hybrid})} &:= \text{flatten}\left(\boldsymbol{H}_i^{(\text{Bidi. CMOW})}\right) \,||\, \boldsymbol{h}_i^{(\text{CBOW})}
\end{aligned}$$

Note that the forward and backward directions of the embedding vectors $\boldsymbol{h}^{(\text{CBOW})}$ conflate to equivalent formulas when we encode entire sequences. Thus, we only need to include a single CBOW representation along with the two CMOW components that yield different results for the forward and backward direction. At inference time, the model is parallelizable along the sequential dimension.

For regularization, we apply a mild dropout ($p = 0.1$) on both the embeddings and their aggregated representations during pretraining. Then, we feed them into a linear masked language modeling head, see Figure 1, or an MLP classification head to tackle the downstream tasks.

We have also experimented with linear, LSTM, and CNN classification heads. We chose an MLP because it adds non-linearity to the model without introducing further complexity. The MLP has led to the best average performance across all GLUE tasks. We report the detailed results, also with the other downstream classifiers, in the supplementary material.

B. Cross-Architecture Distillation

A central question of our research is whether we can distill a large PreLM, e. g., BERT, into more efficient, non-transformer architectures such as the proposed bidirectional CMOW/CBOW-Hybrid model. This requires a cross-architecture distillation approach, which we describe below.

In general, the idea of knowledge distillation is to compress the knowledge of a large teacher model into a smaller student model [9, 10]. It involves a loss function $\mathcal{L}$ that is a combination of two loss terms, i. e., $\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{hard}} + (1 - \alpha) \cdot \mathcal{L}_{\text{soft}}$ with weighting parameter α. $\mathcal{L}_{\text{hard}}$ denotes the cross-entropy loss with respect to the ground truth and $\mathcal{L}_{\text{soft}} = \Sigma_i t_i \cdot \log(s_i)$ is the cross-entropy between the student logits $\boldsymbol{s}$ and the teacher signal $\boldsymbol{t}$. Optionally, the softmax within $\mathcal{L}_{\text{soft}}$ is flattened by a temperature parameter T. We distinguish *general distillation*, where BERT's teacher signal is only used during pretraining, and *task-specific distillation*, where the BERT teacher signal is used during fine-tuning for the downstream task (see Section II).

Considering our goal to design a cross-architecture distillation, the general distillation approach has the conceptual benefit that the teacher model is not needed for fine-tuning. Thus, the student model is capable of tackling downstream tasks without the supervision of the large teacher. This has the benefit that one does not need to carry around the BERT model for adaption to every new downstream task. Above, we have introduced the ability to emit per-token representations with lightweight (bidirectional) CMOW/CBOW-Hybrid embedding models. This

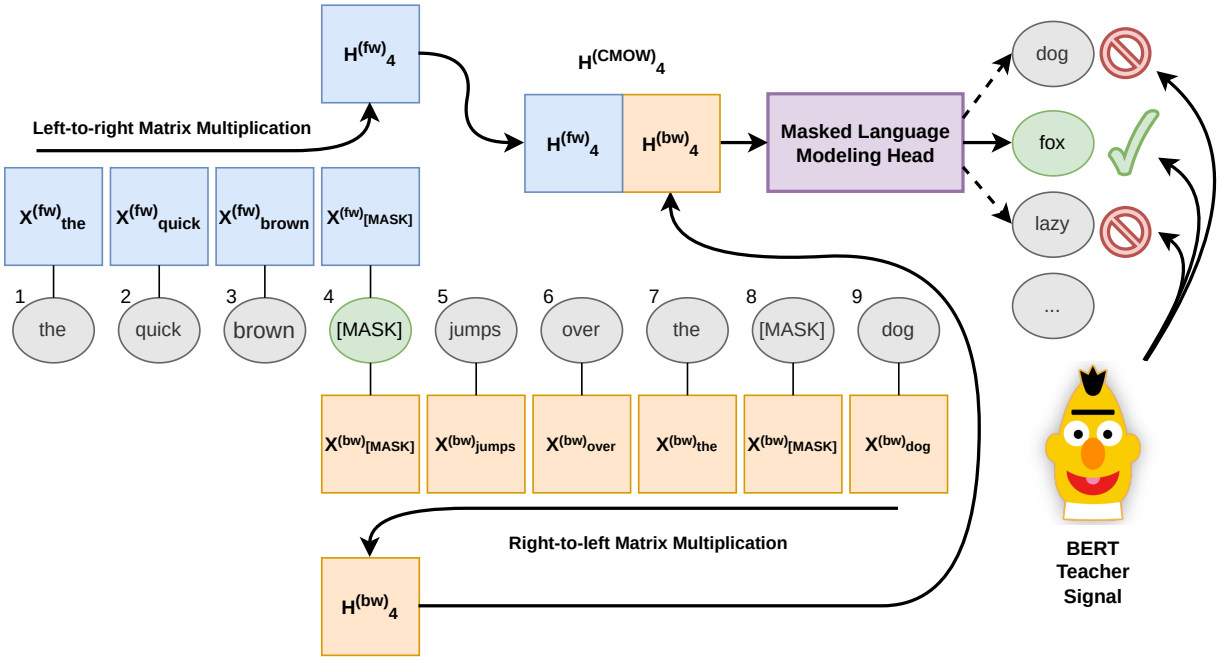


Fig. 1. The bidirectional CMOW component of our proposed architecture during pretraining. In this example, the model predicts the masked token at position 4 ([MASK]) by concatenating forward and backward matrix embeddings, which are then fed into a masked language modeling head.

enables us to use BERT's teacher signal during pretraining together with a masked language modeling objective. In other words, this enables us to perform cross-architecture distillation with matrix embeddings.

We consider three variants of cross-architecture distillation in our experiments: a) When using general distillation, depicted in Figure 1, BERT acts as a teacher during pretraining and the model is fine-tuned to downstream tasks on its own. b) For task-specific distillation, BERT acts as a teacher during fine-tuning as shown in Figure III-C. For this case, we have the option of either b1) starting with pretrained embeddings (from general distillation, i. e., the a) variant), or b2) starting from scratch with randomly initialized embeddings.

C. Two-Sequence Encoding with Matrix Embeddings

When fine-tuning our matrix embeddings to downstream tasks, we can deviate from BERT's input processing, even if BERT is used as a teacher. This is because the distillation loss is computed per sentence (pair) and not per token. The input processing of BERT encodes two sequences by joining them into one sequence. For example, in a natural language inferencing task, there is a sentence A that potentially entails a sentence B, which is encoded together as a single sequence using a special separator token. This encoding scheme is less useful to our matrix embeddings without any attention component, since the order-aware matrix multiplications would blend the representation of the two sequences.

To develop an appropriate two-sequence encoding scheme for matrix embeddings, we take inspiration from the pre-transformer era, e. g., Mou et al. [23], and from Sentence-BERT [24]. The key idea is to encode two sentences A and B separately before combining them. As combination operation, we use the absolute elementwise difference and concatenate it to the representations of A and B, which we denote as DiffCat:

$$\boldsymbol{h}^{(\text{DiffCat})} = \boldsymbol{h}^{(\text{A})} \;||\; |\boldsymbol{h}^{(\text{A})} - \boldsymbol{h}^{(\text{B})}| \;||\; \boldsymbol{h}^{(\text{B})}$$

We illustrate this separate encoding scheme during task-specific distillation in Figure III-C. The rationale for using a concatenation of both sequence representations along with their difference is that we add a component for the similarity of the two sequence representations, without compromising expressive power.

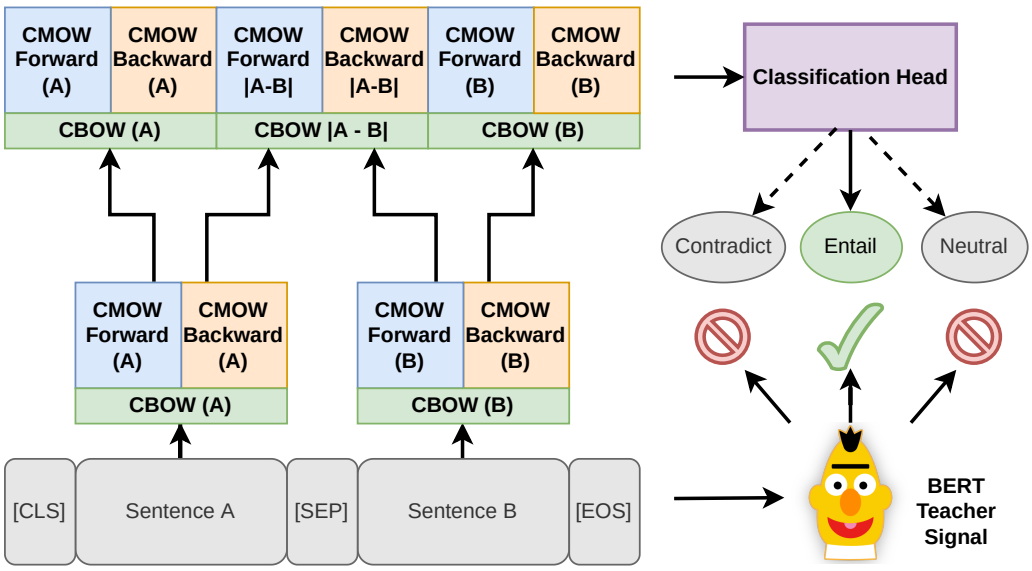
IV. Experimental Procedure

The experimental procedure is divided into pretraining on unlabeled text and fine-tuning on the downstream tasks. We provide the details for these two stages and close the section by outlining the downstream tasks and evaluation measures.

A. Pretraining and General Distillation

In the pretraining stage, as shown in Figure 1, we train our proposed bidirectional CMOW/CBOW-Hybrid model with a masked language modeling objective (MLM) [1] on large amounts of unlabeled text. The MLM objective is to predict left-out words from their context. We put equal weights on the MLM objective and the teacher signal from BERT ($\alpha = 0.5$). As suggested by Liu et al. [25] and Sanh et al. [11], we do not use the next-sentence prediction objective of BERT, but only the MLM objective.

As datasets for pretraining, we use a combination of English Wikipedia and Toronto Books [26], as used in the original BERT. To reduce the environmental footprint of our



experiments, we have only pretrained a single bidirectional CMOW/CBOW-Hybrid model with BERT-base as a teacher on the full unlabeled training data, after pre-experiments on 10% of the training data showed that the selected bidirectional CMOW/CBOW-Hybrid with distillation exceeded the performance of the baseline.

We use matrix embeddings of size $20 \times 20 = 400$ ($d = 20$) for both the CMOW directions (forward and backward) and vector embeddings of size $d_{\text{vec}} = 400$. We use BERT's tokenizer and its vocabulary for both, the teacher and student. The BERT tokenizer relies primarily on the WordPiece algorithm [27], which yields a high coverage while maintaining a small vocabulary.

B. Fine-tuning and Task-specific Distillation

In the fine-tuning stage, as shown in Figure III-C, the pretrained model is adapted for each downstream task individually. The training objective for fine-tuning is either cross-entropy with the ground truth (in general distillation) or a mixture of the ground truth loss and cross-entropy with respect to the teacher's logits (in task-specific distillation). Again, we put equal weight on ground truth and teacher signal along ($\alpha = 0.5$) with unit temperature. To facilitate distillation on regression tasks, we follow Raffel et al. [2] and cast STS-B from regression to classification by binning the scores into intervals of 0.2. To encode the inputs for two-sequence tasks, we use a sequential encoding similar to BERT and the proposed DiffCat encoding (see Section III-C).

For task-specific distillation, we employ an uncased BERT-base model[2] from the Huggingface repository that has already been fine-tuned for each task of the GLUE benchmark. We have fine-tuned the BERT model ourselves on the tasks STS-B, where we applied binning, and MNLI, where the pretrained model led to subpar results. We use the same fine-tuned BERT model as a teacher for all experiments with task-specific distillation.

We seek a fair comparison between the unidirectional CMOW/CBOW-Hybrid baseline model and our bidirectional model. As such, we allow both models to equally benefit from BERT's teacher signal during fine-tuning. For our comparisons regarding specific components (two-sentence encoding and bidirectionality), we use random initialization for both models because Mai et al. 's pretrained embeddings [19] came with a different vocabulary that covered only 53% of the one of BERT. Throughout the other experiments, we initialize our bidirectional CMOW/CBOW-Hybrid with the pretrained embeddings from general distillation, while we isolate the effect of task-specific distillation in a dedicated experiment.

For hyperparameter optimization, we tune learning rates in the range of $[10^{-3}, 10^{-6}]$. In total, we have conducted 306 training and evaluation runs for hyperparameter optimization of each GLUE task. To determine the best model, we use each task's evaluation measure on the development set. We run each model for 20 epochs with early stopping (5 epochs patience). We select appropriate batch sizes on the basis of preliminary experiments and training data sizes.

In the supplementary material, we provide a more detailed discussion of the hyperparameters. We also report the hyperparameter values of the best-performing models. We have also experimented with data augmentation and using exclusively the teacher signal during task-specific distillation ($\alpha = 1$). In some tasks, we could further increase the results by a small margin, but found no consistent improvement. These additional experiments can also be found in the supplementary materials.

C. Downstream Tasks and Measures

We use the GLUE benchmark [3] to evaluate our models. The GLUE benchmark consists of nine tasks for English language comprehension [3]. These tasks comprise natural language inference (MNLI-m, QNLI, WNLI, RTE), sentence similarity

[2]https://huggingface.co/textattack

(QQP, STS-B, MRPC), linguistic acceptibility (CoLA), and sentiment analysis (SST-2). All tasks are based on pairs of sentences except CoLA and SST-2, which are single-sentence tasks. The GLUE benchmark explicitly encourages the use of different fine-tuning strategies for different tasks. For our evaluation, we use the GLUE development set along with its task-specific measures. As such, the performance on all four NLI tasks as well as SST-2 is measured with accuracy. CoLA is evaluated by Matthews correlation coefficient. Similarity tasks are measured by the average Pearson and Spearman correlation for the STS-B task, and as the average accuracy and F_1-score for MRPC and QQP.

V. Results

We present the results along the design choices introduced in Section III, namely bidirection, cross-architecture distillation approaches, and two-sequence encoding scheme. We compare our best embedding methods with ELMo and BERT distillates from the literature. Finally, we report the inference times and the number of parameters of the models.

A. DiffCat Encoding vs. Joint Encoding

First, we compare the encoding schemes for two-sentence tasks. On the one hand, we have the BERT-like encoding that encodes the two sentences together, separated by a special token. On the other hand, we have the proposed DiffCat encoding, which encodes each sentence separately before combining the representations. For a fair comparison, we use a randomly initialized unidirectional CMOW/CBOW-Hybrid model under task-specific distillation.

Table I shows that DiffCat encoding improves the results consistently with the largest margin on STS-B. The most remarkable improvement is the improvement from 18.5 to 58.6 in the STS-B sentence similarity task when encoding the input of the sentence pair through DiffCat. The average improvement across the two-sentence GLUE tasks is 20%, when the DiffCat encoding is used over a BERT-like joint encoding of the sentence pairs. In subsequent experiments, we only report scores with DiffCat encoding.

B. Bidirectional vs. Unidirectional CMOW/CBOW-Hybrid

To isolate the effect of the bidirectional component, we compare unidirectional CMOW/CBOW-Hybrid with bidirectional CMOW/CBOW-Hybrid under equal conditions. We train both variants from scratch for the downstream tasks, while using a BERT's teacher signal. Table II shows the results of the comparison of unidirectional Hybrid embeddings with the proposed bidirectional Hybrid embeddings. Bidirection helps on the tasks MNLI, MRPC, QNLI, SST-2, STS-B, and WNLI. On the other tasks, the difference is marginal. We have an average improvement of 1% of the bidirectional model over the unidirectional model across all tasks of the GLUE benchmark.

C. General Distillation vs. Task-Specific Distillation

Next, we compare general distillation with task-specific distillation. As shown in Table III, using general distillation leads to better results for five tasks (MNLI, MRPC, QQP, STS-B, and RTE) compared to task-specific distillation. For the other four tasks (CoLA, QNLI, SST-2, and WNLI), task-specific distillation achieves higher scores. The average score of general distillation is higher than with task-specific distillation in both pretrained and randomly initialized cases.

D. Comparing Our Best Models to the Literature

Table IV shows the results of the best bidirectional CMOW/CBOW-Hybrid variants using any of the three distillation methods considered. As described by Wasserblatt et al. [15], a model needs to capture context and linguistic structure to perform well on CoLA. We doubled the results for CoLA and SST-2 compared to the best previously reported cross-architecture distillation approaches by Wasserblatt et al. [15]. Our best models scored higher than ELMo [21] on the tasks MRPC, QNLI, QQP, RTE, and WNLI. We achieve higher scores than DistilBERT on RTE and WNLI.

E. Runtime Performance and Parameter Count

To compare runtime performance, we generate 1,024 batches with 256 random sequences of length 64 and measure the inference time (no gradient computation) of the models to encode the sequences. As shown in Table V, both bidirectional CMOW/CBOW-Hybrid and, notably, TinyBERT are more than 6 times faster than BERT-base and more than 3 times faster than DistilBERT. Bidirectional CMOW/CBOW-Hybrid uses only half of DistilBERT's parameters.

The inference speed of CMOW/CBOW-Hybrid could be increased even further because the $\mathcal{O}(n)$ steps to encode a sequence of length n can be parallelized into $\mathcal{O}(\log n)$ *sequential* steps, since matrix multiplication is associative.

VI. Discussion and Related Work

a) Key Results: We have shown that BERT can be distilled into efficient matrix embedding models during pretraining by emitting intermediate representations. We have also introduced a bidirectional component and a separate two-sequence encoding scheme for CMOW-style models. We have observed that the general distillation approach, i. e., using the BERT teacher only during pretraining, leads to results that are oftentimes even better than those achieved with task-specific distillation. This is an interesting result because all previous works on cross-architecture distillation relied on task-specific distillation. Our proposed model offers an encoding speed at inference time that is three times faster than DistilBERT and more than five times faster than MobileBERT.

b) Reflection to Same-Architecture Approaches: Recall that in general distillation, a PreLM is distilled into a student model during pretraining. DistilBERT [11] is such a general-purpose language model that has been distilled from BERT. Apart from masked language modeling and distillation objectives, the authors also introduced a cosine loss term to align the student's and teacher's hidden states (layer transfer). Furthermore, the student is initialized with selected layers of the teacher. MobileBERT [13] introduced a bottleneck to

TABLE I
DIFFCAT ENCODING VS. JOINT BERT-LIKE ENCODING. BOTH VARIANTS USE RANDOMLY INITIALIZED UNIDIRECTIONAL CMOW/CBOW-HYBRID EMBEDDINGS WITH MLP UNDER TASK-SPECIFIC DISTILLATION. DIFFCAT ENCODING IMPROVES THE AVG. SCORE ACROSS TWO-SENTENCE TASKS BY 20%.

Two-Sentence Encoding	Avg.	MNLI-m	MRPC	QNLI	QQP	RTE	STS-B	WNLI
BERT-like Joint Encoding	55.8	50.0	73.0	60.4	78.6	53.8	18.5	**56.3**
DiffCat Separate Encoding	**66.8**	**62.5**	**74.3**	**71.5**	**86.6**	**58.1**	**58.6**	**56.3**

TABLE II
BIDIRECTIONAL VERSUS UNIDIRECTIONAL CMOW/CBOW-HYBRID UNDER TASK-SPECIFIC DISTILLATION WITH DIFFCAT ENCODING.

Model Type	Score	CoLA	MNLI-m	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
Hybrid, rand. init.	62.5	**13.1**	62.5	74.3	71.5	**86.6**	**58.1**	83.1	58.6	56.3
Bidirectional Hybrid, rand. init.	**63.2**	13.0	**63.3**	**75.7**	**72.6**	86.1	57.4	**83.3**	**59.7**	**57.7**

TABLE III
COMPARISON OF TASK-SPECIFIC VS. GENERAL DISTILLATION USING BIDIRECTIONAL CMOW/CBOW-HYBRID EMBEDDINGS.

Distillation Type	Score	CoLA	MNLI-m	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
General	**66.6**	16.7	**66.6**	**79.7**	71.7	**87.2**	**61.0**	82.9	**76.9**	56.3
Task-specific, rand. init	63.2	13.0	63.3	75.7	**72.6**	86.1	57.4	**83.3**	59.7	**57.7**
Task-specific, pretrained	64.6	**23.3**	61.8	75.0	72.0	86.3	59.9	82.9	62.9	**57.7**

TABLE IV
COMPARISON OF BEST EMBEDDING-BASED METHODS (IN BOLD) WITH METHODS FROM THE LITERATURE ON THE GLUE VALIDATION SET. THE ⋆-SYMBOL INDICATES NUMBERS ON THE OFFICIAL GLUE TEST SET

Method	Score	CoLA	MNLI-m	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
BERT-base (our teacher model)	78.9	57.9	84.2	84.6	91.4	89.7	67.9	91.7	88.0	54.9
ELMo [21]	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
DistilBERT [11]	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3
⋆ MobileBERT [13]	—	51.1	84.3	88.8	91.6	70.5	70.4	92.6	84.8	—
⋆ TinyBERT (4 layers) [12]	—	44.1	82.5	86.4	87.7	71.3	66.6	92.6	80.4	—
Hybrid [19]	—	—	—	—	—	—	—	79.6	63.4	—
Word2rate [28]	—	—	—	—	—	—	—	65.7	53.1	—
CBOW [15]	—	10.0	—	—	—	—	—	79.1	—	—
BiLSTM [15]	—	10.0	—	—	—	—	—	80.7	—	—
Bidi. Hybrid + MLP (ours)	**68.0**	**23.3**	**66.6**	**80.9**	**72.6**	**87.2**	**61.0**	**84.0**	**76.9**	**59.2**

TABLE V
NUMBER OF PARAMETERS AND INFERENCE TIME OF THE MODELS. INFERENCE SPEED ON THE TRAINED MODEL COMPUTED USING AN NVIDIA A100-SXM4-40GB CARD

Model	# Parameters	Inference speed (sent./sec)
ELMo	94M	1.1k
BERT-base	109M	4.6k
DistilBERT-base	66M	9.2k
MobileBERT	25M	5.5k
TinyBERT (4 layer)	**14M**	**30.0k**
Bidi. Hyrid	37M	**30.0k**

BERT such that layers can be transferred to student models with smaller dimensions.

In task-specific distillation, the teacher signal is used during fine-tuning. Sun et al. [29] use layer-wise distillation objectives and initialize with teacher weights to train BERT students with fewer layers. TinyBERT [12] applies knowledge distillation in both stages, pretraining and fine-tuning. LadaBERT [30] combines knowledge distillation with pruning and matrix factorization. Other approaches consider distillation in multi-lingual [31] or multi-task settings [32].

Bidirectional CMOW/CBOW-Hybrid yields high throughput rates comparable to a 4-layer TinyBERT [12]. In the original TinyBERT [12] work, the authors report a speed-up of 2x with 6 layers compared to BERT-base, and 9.4x with 4 layers. However, TinyBERT requires to have the teacher model available for fine-tuning. We have shown that CMOW/CBOW-Hybrid is better even when using only general distillation compared to using task-specific distillation. TinyBERT further augments the training data, which we have also considered, but we found no consistent improvement.

Turc et al. [33] analyze the interaction between pre-training and fine-tuning with BERT models and find that pretrained distillation works well, which agrees with our findings on the importance of pretraining with CMOW-style models.

c) Reflection w.r.t. Cross-Architecture Approaches: The distillations of BERT described above assume that the teacher and student share the same architecture. However, the student model does not need to have the same architecture as the teacher, which is what we call cross-architecture distillation. For example, Wasserblatt et al. [15] use a simple feed forward network with CBOW embeddings and a bidirectional LSTM model as students. Both models perform well in several downstream tasks. Tang et al. [14] explore the distillation of BERT into a single-layer BiLSTM without using additional training data or modifications to the teacher architecture. Their distillation-based approach yields improvements compared to a plain BiLSTM without teacher signal: about 4 points on all reported tasks (QQP, MNLI, and SST-2). This has motivated us to investigate whether even more efficient models can be used as students of a BERT teacher.

d) Reflection w.r.t. Pruning and Quantization: Other techniques for reducing the size of a model are pruning and quantization. Pruning approaches such as in Sanh et al. [8] reduce the number of parameters. Still, the resulting smaller models use the same architecture as their larger counterparts and, thus, pruning does not necessarily improve inference speed. Quantization is a common post-processing step to reduce model size by decreasing the floating point precision of the weights [34, 35]. Pruning and quantization can be applied in conjunction with knowledge distillation [8, 13]. Aside from techniques for reducing the model size, there is also a tremendous effort to improve the efficiency of the transformer architecture in the first place [36].

e) Threats to Validity: For a fair comparison, we have ensured that our baselines and the proposed extensions have equal conditions. When testing the effect of bidirectional component and the separate-encoding scheme, we start with random initialization with for the baseline model and for the extended models. Moreover, we have put the same effort into hyperparameter tuning for the baselines and the proposed extensions, as well as when comparing general and task-specific distillation with the proposed model.

Currently, cross-architecture distillation approaches still fall behind other BERT distillates, such as MobileBERT and TinyBERT, in many of the downstream tasks. In particular, detecting linguistic acceptability remains a challenge for non-transformer methods, even though we improve upon previous cross-architecture distillation approaches. So far, we have not analyzed the trade-off between embedding size and downstream performance. We rely on general arguments for the benefits of increased dimensionality [37].

f) Future Work: One could further improve the efficiency by applying pruning [8] and/or quantization [35] techniques on the learned matrices to allow sparse matrix multiplication during encoding. Future work could explore what components would be necessary to improve scores on particularly challenging downstream tasks such as detecting linguistic acceptability. This might include the introduction of a small attention module like that of gMLP [38] to CMOW/CBOW-Hybrid. Finally, matrix embeddings could also be used in other domains where sequences of discrete elements need to be encoded, e. g., events, or gene sequences.

VII. Conclusion

This research contributes to the development of simpler and more efficient models for natural language processing in general. We have introduced three extensions to the CMOW/CBOW-Hybrid model: a bidirectional component, a separate two-sequence encoding scheme, and the ability to emit per-token representations. These per-token representations allow us to distill BERT into CMOW/CBOW-Hybrid already during pretraining with a masked language modeling objective. Our results show that a separate encoding scheme improves the performance of CMOW/CBOW-Hybrid on two-sentence GLUE tasks by 20%, while bidirection improves the performance by 1% compared to the unidirectional model. Furthermore, we have shown that general distillation seems to be sufficient, and task-specific distillation is not necessary for most GLUE tasks. In comparison to more expensive language models reported in the literature, our embedding-based approach achieves scores that match or exceed the scores of ELMo and are competitive to DistilBERT on QQP and RTE with only half of its parameters and thrice its encoding speed. While linguistic acceptability remains a challenge for non-transformer models, our approach yields notably higher scores than previous cross-architecture distillation approaches.

Ethics Statement

The drawback of PreLMs, as discussed in the introduction, is that they are growing larger and larger, which leads to higher environmental and economic costs [6] and shifts the development and training of PreLMs into the hands of a few global players [7]. Large language models have also been criticized for having biases because they are trained on a large amount of weakly curated training data [39]. Moreover, in vision, the reduction of model sizes has also been reported to amplify bias [40]. This is particularly a problem when the model is not explainable. In our proposed method, we distill large-scale PreLMs into models that do not have a nonlinearity (except for the final classification head). Since linear models are easier to explain, we hope to contribute to more explainable language models with our work, even though explainability is not the focus of the present work. In fact, models conceptually similar to the ones presented here have been used to explain black-box representations [41].

Reproducibility Statement

We provide the code and pretrained models: https://github.com/lgalke/cross-architecture-distillation
Furthermore, supplementary material with extended results are available: https://arxiv.org/abs/2109.08449.

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT (1)*. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[2] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *ArXiv*, vol. abs/1910.10683, 2020. [Online]. Available: https://arxiv.org/abs/1910.10683

[3] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *BlackboxNLP@EMNLP*. Association for Computational Linguistics, 2018, pp. 353–355. [Online]. Available: https://doi.org/10.18653/v1/w18-5446

[4] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "SuperGLUE: A stickier benchmark for general-purpose language understanding systems," in *NeurIPS*, 2019, pp. 3261–3275. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/4496bf24afe7fab6f046bf4923da8de6-Abstract.html

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, and others, "Language models are few-shot learners," *ArXiv*, vol. abs/2005.14165, 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[6] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," in *ACL (1)*. Association for Computational Linguistics, 2019, pp. 3645–3650. [Online]. Available: https://doi.org/10.18653/v1/p19-1355

[7] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, and others, "On the opportunities and risks of foundation models," *ArXiv*, vol. abs/2108.07258, 2021. [Online]. Available: https://arxiv.org/abs/2108.07258

[8] V. Sanh, T. Wolf, and A. M. Rush, "Movement pruning: Adaptive sparsity by fine-tuning," in *NeurIPS*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/eae15aabaa768ae4a5993a8a4f4fa6e4-Abstract.html

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *ArXiv*, vol. abs/1503.02531, 2015. [Online]. Available: https://arxiv.org/abs/1503.02531

[10] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *KDD*. ACM, 2006, pp. 535–541. [Online]. Available: https://doi.org/10.1145/1150402.1150464

[11] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of bert: smaller, faster, cheaper and lighter," *ArXiv*, vol. abs/1910.01108, 2020. [Online]. Available: https://arxiv.org/abs/1910.01108

[12] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for natural language understanding," in *EMNLP (Findings)*. Association for Computational Linguistics, 2020, pp. 4163–4174. [Online]. Available: https://doi.org/10.18653/v1/2020.findings-emnlp.372

[13] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "MobileBERT: a compact task-agnostic BERT for resource-limited devices," in *ACL*. Association for Computational Linguistics, 2020, pp. 2158–2170. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.195

[14] R. Tang, Y. Lu, L. Liu, L. Mou, O. Vechtomova, and J. Lin, "Distilling task-specific knowledge from bert into simple neural networks," *ArXiv*, vol. abs/1903.12136, 2019. [Online]. Available: https://arxiv.org/abs/1903.12136

[15] M. Wasserblat, O. Pereg, and P. Izsak, "Exploring the boundaries of low-resource BERT distillation," in *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*. Association for Computational Linguistics, Nov. 2020, pp. 35–40. [Online]. Available: https://aclanthology.org/2020.sustainlp-1.5

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[17] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," in *ICML*. ACM, 2008, pp. 160–167. [Online]. Available: https://doi.org/10.1145/1390156.1390177

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *ArXiv*, vol. abs/1301.3781, 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[19] F. Mai, L. Galke, and A. Scherp, "CBOW is not all you need: Combining CBOW with the compositional matrix space model," in *ICLR (Poster)*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=H1MgjoR9tQ

[20] S. Rudolph and E. Giesbrecht, "Compositional matrix-space models of language," in *ACL*. The Association for Computer Linguistics, 2010, pp. 907–916. [Online]. Available: https://aclanthology.org/P10-1093/

[21] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL-HLT*. Association for Computational Linguistics, 2018, pp. 2227–2237. [Online]. Available: https://doi.org/10.18653/v1/n18-1202

[22] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997. [Online]. Available: https://doi.org/10.1109/78.650093

[23] L. Mou, R. Men, G. Li, Y. Xu, L. Zhang, R. Yan, and Z. Jin, "Natural language inference by tree-based convolution and heuristic matching," in *ACL (2)*. The Association for Computer Linguistics, 2016. [Online]. Available: https://aclanthology.org/P16-2022/

[24] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *EMNLP/IJCNLP (1)*. Association for Computational Linguistics, 2019, pp. 3980–3990. [Online]. Available: https://doi.org/10.18653/v1/D19-1410

[25] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[26] Y. Zhu, R. Kiros, R. S. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *ICCV*. IEEE Computer Society, 2015, pp. 19–27. [Online]. Available: https://ieeexplore.ieee.org/document/7410368

[27] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, and others, "Google's neural machine translation system: Bridging the gap between human and machine translation," *ArXiv*, vol. abs/1609.08144, 2016. [Online]. Available: https://arxiv.org/abs/1609.08144

[28] G. Phua, S. Lin, and D. Poletti, "Word2rate: training and evaluating multiple word embeddings as statistical transitions," *ArXiv*, vol. abs/2104.08173, 2021. [Online]. Available: https://arxiv.org/abs/2104.08173

[29] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for BERT model compression," in *EMNLP/IJCNLP (1)*. Association for Computational Linguistics, 2019, pp. 4322–4331. [Online]. Available: https://doi.org/10.18653/v1/D19-1441

[30] Y. Mao, Y. Wang, C. Wu, C. Zhang, Y. Wang, Q. Zhang, Y. Yang, Y. Tong, and J. Bai, "Ladabert: Lightweight adaptation of BERT through hybrid model compression," in *COLING*. International Committee on Computational Linguistics, 2020, pp. 3225–3234. [Online]. Available: https://doi.org/10.18653/v1/2020.coling-main.287

[31] H. Tsai, J. Riesa, M. Johnson, N. Arivazhagan, X. Li, and A. Archer, "Small and practical BERT models for sequence labeling," in *EMNLP/IJCNLP (1)*. Association for Computational Linguistics, 2019, pp. 3630–3634. [Online]. Available: https://aclanthology.org/D19-1374/

[32] Z. Yang, L. Shou, M. Gong, W. Lin, and D. Jiang, "Model compression with multi-task knowledge distillation for web-scale question answering system," *Arxiv*, vol. abs/1904.09636, 2019. [Online]. Available: https://arxiv.org/abs/1904.09636

[33] I. Turc, M. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: The impact of student initialization on knowledge distillation," *Arxiv*, vol. abs/1908.08962, 2019. [Online]. Available: https://arxiv.org/abs/1908.08962

[34] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *ICML*, ser. JMLR Workshop and Conference Proceedings, vol. 37. JMLR.org, 2015, pp. 1737–1746. [Online]. Available: http://proceedings.mlr.press/v37/gupta15.html

[35] H. Wu, P. Judd, X. Zhang, M. Isaev, and P. Micikevicius, "Integer quantization for deep learning inference: Principles and empirical evaluation," *ArXiv*, vol. abs/2004.09602, 2020. [Online]. Available: https://arxiv.org/abs/2004.09602

[36] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *ArXiv*, vol. abs/2009.06732, 2020. [Online]. Available: https://arxiv.org/abs/2009.06732

[37] J. Wieting and D. Kiela, "No training required: Exploring random encoders for sentence classification," in *ICLR (Poster)*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=BkgPajAcY7

[38] H. Liu, Z. Dai, D. R. So, and Q. V. Le, "Pay attention to MLPs," *ArXiv*, vol. abs/2105.08050, 2021. [Online]. Available: https://arxiv.org/abs/2105.08050

[39] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in *FAccT*. ACM, 2021, pp. 610–623. [Online]. Available: https://doi.org/10.1145/3442188.3445922

[40] S. Hooker, N. Moorosi, G. Clark, S. Bengio, and E. Denton, "Characterising bias in compressed models," *Arxiv*, vol. abs/2010.03058, 2020. [Online]. Available: https://arxiv.org/abs/2010.03058

[41] P. Soulos, R. T. McCoy, T. Linzen, and P. Smolensky, "Discovering the compositional structure of vector representations with role learning networks," in *BlackboxNLP@EMNLP*. Association for Computational Linguistics, 2020, pp. 238–254. [Online]. Available: https://doi.org/10.18653/v1/2020.blackboxnlp-1.23