Partially-Random Initialization: A Smoking Gun for Binarization Hypothesis of BERT

Arash Ardakani
Noah's Ark Lab
Huawei Technologies Canada
arash.ardakani@huawei.com

Abstract

In the past few years, pre-trained BERT has become one of the most popular deep-learning language models due to their remarkable performance in natural language processing (NLP) tasks. However, the superior performance of BERT comes at the cost of high computational and memory complexity, hindering its envisioned widespread deployment in edge devices with limited computing resources. Binarization can alleviate these limitations by reducing storage requirements and improving computing performance. However, obtaining a comparable accuracy performance for binary BERT w.r.t. its full-precision counterpart is still a difficult task. We observe that direct binarization of pre-trained BERT provides a poor initialization during the fine-tuning phase, making the model incapable of achieving a decent accuracy on downstream tasks. Based on this observation, we put forward the following *hypothesis*: partially randomly-initialized BERT with binary weights and activations can reach to a decent accuracy performance by distilling knowledge from the its full-precision counterpart. We show that BERT with pre-trained embedding layer and randomly-initialized encoder is a smoking gun for this hypothesis. We identify the smoking gun through a series of experiments and show that it yields a new set of state-of-the-art results on the GLUE and SQuAD benchmarks.

1 Introduction

In the past few years, pre-trained language models, particularity BERT (Devlin et al., 2019), have shown a great success across various natural language processing (NLP) tasks. As such, it is envisioned that BERT will be widely deployed in edge devices with limited computing resources in the near future. Such a widespread deployment, however, requires a significant decrease in both memory storage and computational complexity of BERT. Therefore, model compression techniques, such as pruning (McCarley et al., 2019; Gordon et al., 2020), quantization (Shen et al., 2020; Zafrir et al., 2019), parameter sharing (Lan et al., 2020) and distillation (Jiao et al., 2019; Sanh et al., 2019), have been an active field of studies among practitioners to meet the required computing constraints of edge devices. Among the aforementioned compression methods, quantization is considered as an effective edge computing approach that can reduce the model size by up to $32\times$ and replace expensive floating-point multiplications with simpler fixed-point counterparts. However, quantization of BERT comes at the cost of a severe performance degradation due to the low bit-width representation (Qin et al., 2022). To compensate for the performance drop of quantization in BERT, knowledge distillation (KD) is used an auxiliary optimization approach, which encourages the quantized BERT to mimic the behavior of its full-precision counterpart.

Extreme 1-bit quantization, which is commonly referred to as binary representation, is the ideal scenario for edge computing due to the significant decrease in the model size and the computation gain from the bit-wise operations. However, binary representation of weights and activations in BERT while obtaining a comparable performance is still a challenging task even when the binarization process is assisted with KD. In fact, binarization of activations in BERT accounts for most of the performance drop in NLP tasks (Qin et al., 2022). Therefore, previous works mainly focused on binarization of weights in the encoder and word embedding layer of BERT (Zhang et al., 2020; Bai et al., 2021).

In this paper, we first investigate the root cause of the performance drop in BERT with binary weights and activations. Our finding shows that the binarization of pre-trained BERT completely undermines the good initialization point obtained from pre-training, leading to its poor generalization on unseen data. Moreover, the binary representation con-

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2603–2612

straint significantly reduces the learning capacity of BERT, making KD incapable of finding the right optimization direction during the fine-tuning stage. Motivated by these observations, we then propose a hypothesis stating that introducing some degrees of freedom to the binarized network suffices to fully distill knowledge from its full-precision counterpart using KD. We introduce such a freedom by randomly initializing some parts of the binarized network, allowing it to mainly focus on minimizing the total loss with less concern about preserving information from pre-training. We finally search for a smoking gun of this hypothesis and identify it as binary BERT whose weights are initialized randomly except for its embedding weights which are initialized from pre-training.

A smoking gun is an undeniable piece of empirical evidence that proves our hypothesis. We refer to the binary BERT obtained from the smoking gun as SGBERT. We empirically support the proposed hypothesis and the importance of this fortuitous initialization of SGBERT through a series of experiments. The empirical results show that our SGBERT outperforms existing binary BERT models and achieves a new set of state-of-the-art results on the GLUE and SQuAD benchmarks. More specifically, our SGBERT outperforms BiBERT (Qin et al., 2022) by 15% in terms of the average accuracy of the GLUE benchmark. The accuracy performance and the hardware benefits (i.e., the binary representation and the bit-wise operations) of SGBERT show its great potential for adoption in edge computing.

2 Our Binary BERT Architecture

BERT architecture is consisted of three main parts: embedding layer, encoder and classifier. The encoder module itself contains two main components: multi-head attention (MHA) module and feed-forward network (FFN). We binarize the weights of the aforementioned components using the sign function as follows:

$$\alpha_w = \frac{1}{n}||\mathbf{W}||_1,$$

$$\mathbf{W}_b = \alpha_w \text{sign}(\mathbf{W} - \mu(\mathbf{W})), \tag{1}$$

where $\mathbf{W}$ and $\mathbf{W}_b$ denote full-precision weight and its binary counterpart, respectively. As suggested in (Qin et al., 2022), we redistribute the weight to zero-mean during the bianarization of weights to retain their representation information. The mean value of weight is denoted by $\mu(\cdot)$. We also apply the scaling factor α_w to minimize the binarization error.

Activations in BERT are also binarized in a similar manner to weights using the sign function in the forward path and its gradient is computed using the straight-through estimator (STE) (Courbariaux et al., 2015) in the backward propagation, i.e.,

$$\mathbf{X}_b = \alpha_x \text{sign}(\mathbf{X}),$$

$$\frac{\partial \text{sign}}{\partial \mathbf{X}} = \begin{cases} 1, & \text{if } |\mathbf{X}| \leq 1 \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where $\mathbf{X}$ and $\mathbf{X}_b$ denotes full-precision weight and its binary representation, respectively. We consider an sclaing factor for binarization of activations, which is denoted as α_x. The scaling factor for activations is computed similar to that of weights, i.e., using the averaged ℓ_1 norm. However, such an approach requires on-the-fly computation of ℓ_1 norm for each minibatch, which undermines the efficiency of the binary computations. To address this issue, we estimate the scaling factor for activations using the exponential running average similar to (Ardakani et al., 2022) such that

$$\alpha_{x_{t+1}} = (1-m) \times \alpha_{x_t} + m \times \frac{1}{n}||\mathbf{X}||_1, \tag{3}$$

where m is the momentum controlling the update rate of the scaling factor. We use $m = 0.01$ in our experiments throughout this paper.

In BERT, the inputs are first passed into the embedding layers among which the word embedding layer is binarized only as it contains most of the parameters while the position and type embedding layers are kept in full-precision (Qin et al., 2022). In MHA module, queries $\mathbf{Q}$, keys $\mathbf{K}$ and values $\mathbf{V}$ are computed in a binary manner according to Eq. (1) and Eq. (2). It is worth mentioning that the scaling factors for both weights and activations are combined together and severs as a single scaling factor during inference. Since the direct binarization of softmax results in a significant performance drop based on our empirical experiments, we adopted a boolean function similar to (Qin et al., 2022) where the attention computation is performed by

$$\text{Attention}(\mathbf{Q}_b, \mathbf{K}_b, \mathbf{V}_b) = \text{bool}\left(\frac{\mathbf{Q}_b \odot \mathbf{K}_b^T}{\sqrt{d}}\right) \boxdot \mathbf{V}_b, \tag{4}$$

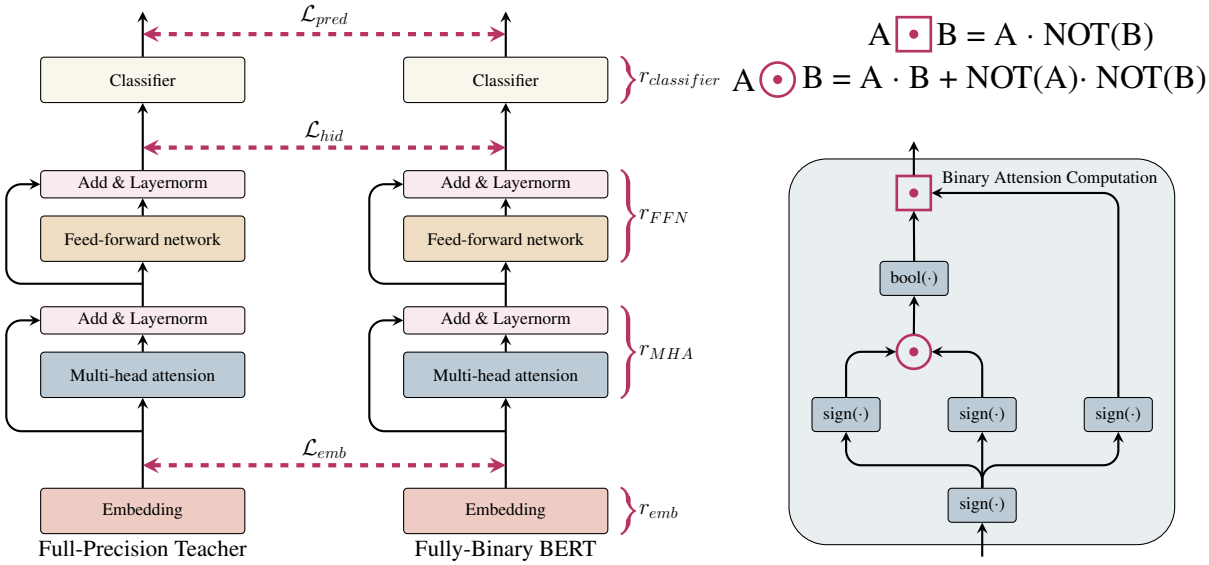


Figure 1: The overall training paradigm for our binary BERT is shown on the left. On the right side, the binary attention computations are illustrated where multiplications are performed using bit-wise operations.

$$\text{bool}(\mathbf{X}) = \begin{cases} 1, & \text{if } \mathbf{X} \geq 0 \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

where $\mathbf{Q}_b$, $\mathbf{K}_b$ and $\mathbf{V}_b$ denote binary query, key and value matrices, respectively. d denotes the dimension of the features. The gradient of bool function is computed using the STE similar to the gradient of sign function. The bit-wise multiplications between binary query and key matrices are computed using XNOR which is denoted by $\odot$. It is worth mentioning that 0 in the binary representation denotes the value of -1 and 0 as the output of the sign function and the bool function, respectively. The former representation is referred to as bipolar whereas the latter one is called unipolar. Since the result of the multiplication between the bipolar and unipolar formats can take three different values (i.e., -1, 0, and 1), the result of such multiplication requires two bits in the two's complement format. Given the unipolar input A, the bipolar input B, and the result of multiplication as C, the least significant bit of C is equal to A and the most significant bit of C is equal to $A \cdot \text{NOT}(B)$ where $\cdot$ and NOT denote the bit-wise AND and logical complement operations, respectively. Figure 1 shows the details of the attention computations in the binary BERT. The rest of the encoder module contains linear layers which are binarized using Eq. (1) and Eq. (2).

3 Knowledge Distillation

In this section, we describe the KD method used in this paper. In general, we use KD for the embedding output, hidden states and logits.

3.1 Intermediate KD

We define $f_t^l(x)$ and $f_s^l(x)$ as the output of the l^{th} layer for the teacher network and the student network given a batch of data (x, y), respectively. The student model is binary BERT whereas the teacher model is its fine-tuned full-precision counterpart. Since the embedding layer contains information about unseen data, it is important to accurately deliver such information to the rest of the network using KD. We use the root mean squared errors (RMSE) as the embedding loss function $\mathcal{L}_{emb}$ to measure the distance between the embedding layer's output of student model (i.e., E_s) and that of teacher model (i.e., E_t) by

$$\mathcal{L}_{emb} = \left\| \frac{E_t}{\|E_t\|_2} - \frac{E_s}{\|E_s\|_2} \right\|_2. \tag{6}$$

Early hierarchical layers of fine-tuned full-precision BERT provides generic linguistic patterns whereas the latter layers extract task-specific patterns (Devlin et al., 2019; Merchant et al., 2020). To make sure that binary BERT also learns similar patterns, we distill knowledge from the l^{th} hidden states of the teacher model H_t^l to that of the student model H_s^l using the RMSE as the loss function for

Table 1: The role of pre-training in the performance of GLUE tasks on the development set. We measure the performance using Spearman correlation for STS-B, Matthews correlation for CoLA and accuracy for RTE, MRPC, SST-2 QQP, and MNLI (matched).

Method	Pre-training	MNLI-m	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE
BERT	✓	84.9	91.4	92.1	93.2	59.7	90.1	86.3	72.2
BERT (re-initialization)[1]	✗	84.8	91.5	91.9	92.9	60.2	90.2	86.4	71.7
BERT (Random initialization)[2]	✗	68.1	84.7	61.3	81.4	16.4	21.5	72.1	54.9
Binary BERT[3]	✓	38.4	68.1	64.9	78.9	0.1	41.1	70.8	54.9

[1] The parameters of each task are initialized by the parameters of fine-tuned BERT from another task. The parameters of fine-tuned BERT on SST-2 is used for CoLA and MRPC, CoLA for SST-2 and QQP, RTE for STS-B and QNLI, MRPC for RTE, and STS-B for MNLI. The parameters of the classifier is still randomly initialized.
[2] The parameters of each task are randomly initialized using PyTorch default.
[3] The parameters of binary BERT are initialized from pre-training.

hidden states (i.e., $\mathcal{L}_{hid}$) such that

$$\mathcal{L}_{hid} = \left\| \frac{H_t^l}{\|H_t^l\|_2} - \frac{H_s^l}{\|H_s^l\|_2} \right\|_2. \tag{7}$$

3.2 Prediction-Layer Distillation

As the last term in our KD scheme, we conduct prediction-layer distillation where soft cross-entropy (SCE) between teacher logits Y_t and student logits Y_s is minimized, i.e.,

$$\mathcal{L}_{pred} = SCE(Y_t, Y_s). \tag{8}$$

Given the aforementioned KD losses, the total distillation loss can be written as follow:

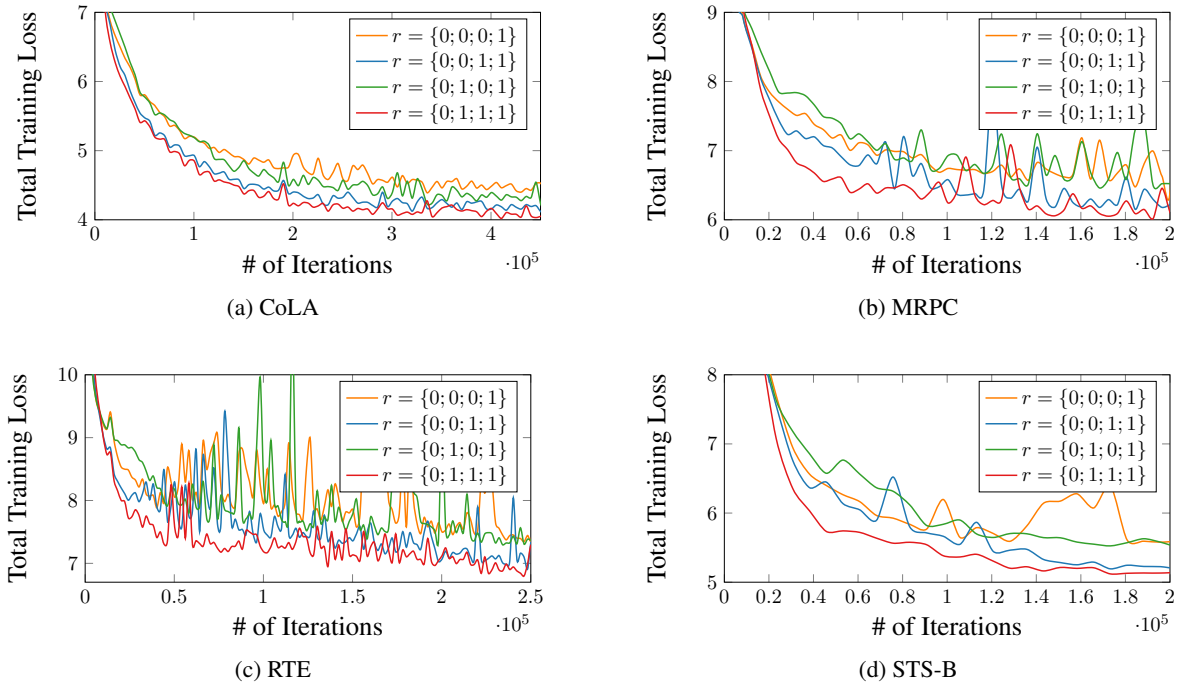
$$\mathcal{L}_{KD} = \mathcal{L}_{emb} + \mathcal{L}_{hid} + \mathcal{L}_{pred}. \tag{9}$$

4 The Proposed Binarization Method

The unsupervised pre-training on enormous unlabeled data is the key factor in the remarkable success of BERT in the context of few-shot learning across various NLP tasks. In fact, the pre-training process provides a good initial point for BERT that leads to a better generalization on unseen data of downstream tasks. The fine-tuning procedure of BERT is performed by adjusting the pre-training weights to minimize the loss of downstream tasks. It is worth mentioning that these adjustments are usually small such that even fine-tuned parameters of BERT on a specific task can be used as an initialization point for another task while still achieving a decent performance. For instance, Table 1 shows the performance of downstream tasks from the GLUE benchmark where the initialization point of each task was obtained from a fined-tuned BERT on a different task. This experiment shows that fine-tuned BERT maintains its information from pre-training. On the other hand, while training from scratch can reach to the same loss (or even better in some cases) as fine-tuning BERT, the randomly-initialized model results in a poor performance on the unseen data (see Table 1).

Since the good performance of BERT relies on its parameters obtained from pre-training, binarization has been always integrated into the fine-tuning stage in previous studies (Qin et al., 2022; Zhang et al., 2020; Bai et al., 2021). However, fully binarization of BERT (i.e., binarization of its weights and activations) using this approach results in a huge performance drop (Qin et al., 2022). We attribute this issue to disruption of the pre-training initialization incurred by fully binarization of BERT. In other words, the fully binary model cannot hold on to its information from pre-training with binary weights and activations. For instance, the binary BERT in Table 1, whose weights are initialized from pre-training, results in a significant accuracy drop, confirming the disruptive effect of binarization on the pre-training initialization. That is why the fine-tuning procedure of quantized BERT models is assisted with KD (Qin et al., 2022).

During the past few years, distillation–in isolation–has been used as a compression technique where a small student model (which is in full-precision) is trained to mimic the behavior of a larger teacher model. In this compression method, the student model can have a similar structure (Sanh et al., 2019; Jiao et al., 2019; Sun et al., 2019) to or a different architecture (Chen et al., 2020) from its teacher model. For instance, in BERT-PKD (Sun et al., 2019), the student model only differs in the number layers from its teacher model (i.e., BERT) and KD was used to distill knowledge from the BERT's intermediate layer during the fine-tuning phase. When the student model's structure is similar to its teacher model, it is a common approach to initialize the student model's parameters from the pre-trained BERT (e.g., BERT-PKD). On



(a) CoLA (b) MRPC

(c) RTE (d) STS-B

Figure 2: The training loss curves of four different tasks of the GLUE benchmark for four combinations of the initialization identifier. All binary models are trained under the same training setting.

the other hand, when the student model's structure is different from its teacher model, its parameters are initialized either partially from the pre-trained BERT or randomly (e.g., AdaBERT (Chen et al., 2020)). In either case (i.e., having a similar or different structure), the student model can acquire sufficient knowledge from its teacher model to perform well on downstream tasks. Based on these distillation works, we put forward the binarization hypothesis.

4.1 The Binarization Hypothesis

A binary BERT contains sub-networks that if initialized randomly, it can reach to a decent performance on NLP downstream tasks when trained in isolation using KD.

Let's consider a BERT model $f(x; \theta)$ with the parameter set of $\theta = \{\theta_{emb}; \theta_{MHA}; \theta_{FFN}; \theta_{classifier}\}$ with the embedding parameter θ_{emb}, multi-head attention parameter θ_{MHA}, feed-forward network parameter θ_{FFN}, and classifier parameter $\theta_{classifier}$ obtained from pre-training. f reaches the test accuracy a when fine-tuning with binary weights and activations on a downstream task. Now, let's consider a BERT model $f(x; \theta', r)$ with the parameter set of θ' and initialization identifier $r = \{r_{emb}; r_{MHA}; r_{FFN}; r_{classifier}\} \in \{0, 1\}^4$. If the initialization identifier is equal to one, its associated parameters are randomly initialized; otherwise, they are initialized from pre-training. When $f(x; \theta', r)$ is trained on a downstream task, it yields the test accuracy a'. Our hypothesis predicts that there exists r such that $a' > a$.

4.2 The Smoking Gun of the Binarization Hypothesis

Due to the small size of the initialization identifier, we perform a brute-force search to identify supporting examples for our hypothesis. Before doing so, however, we can still make the search space smaller. Of course the classifier of BERT is always initialized randomly, which makes $r_{classifier}$ equal to 1. Moreover, the embedding layer is the key element in the fine-tuning process since it contains information about unseen data. Random initialization of the embedding layer drastically degrades the accuracy performance on downstream tasks. Therefore, we set r_{emb} to 0.

To decide about the remaining initialization identifiers (i.e., r_{MHA} and r_{FFN}), we run an experiment on four different tasks of the GLUE benchmark for all four possible initialization combinations as illustrated in Figure 2. According to Figure 2, the binary BERT with $r = \{0; 1; 1; 1\}$ shows a smoother and faster convergence across four different tasks and reaches to a lower training loss among the four possible combinations. Therefore,

the binary BERT with $r = \{0; 1; 1; 1\}$ constitutes the smoking gun.

The smoking gun that we found allows BERT to be binarized in a similar way to the binarization of networks without fine-tuning any parameters. For example, in the binarization process of a convolutional neural network on CIFAR10, which involves no fine-tuning, the goal is to minimize the final loss of the network with any possible values for activations and weights. It is worth mentioning that the final loss of the network can also include a KD loss. Similarly, the smoking gun suggests that weights and activations of MHA and FFN modules can take any values as long as they minimize the final loss of the network. Of course the embedding layer is still being fine-tuned which serves as a feature extractor for the rest of the network. Therefore, the performance of our binary BERT mainly relies on the minimization of the total loss which includes KD losses.

5 Experiments

In this section, we perform empirical experiments to verify the effectiveness of the identified smoking gun as the supporting example for our proposed hypothesis on the GLUE (Wang et al., 2018) and SQuAD (Rajpurkar et al., 2016) benchmarks. We first define our experimental setup. We then present our main experimental results on the aforementioned benchmarks and compare them with state-of-the-art quantized models. Finally, we conduct an ablation study to further discuss the importance of initialization when binarizing BERT.

5.1 Experimental Setup

5.1.1 Dataset and Metrics

We assess the language understanding and generalization capabilities of our proposed binary BERT using eight datasets from the GLUE benchmark. More precisely, we consider four different types of NLP tasks for our experiments: sentiment classification (SST-2), natural language inference (RTE, QNLI, MNLI), paraphrase detection (MRPC, QQP, STS-B), and linguistic acceptability (CoLA). To evaluate the performance of these tasks, we use Spearman correlation for STS-B, Matthews correlation for CoLA and accuracy for RTE, MRPC, SST-2 QQP, MNLI-m (matched) and MNLI-mm (mismatched). To evaluate the reading comprehension of our binary BERT, we perform the question answering task on SQuAD v2.0, where its performance is measured using the EM (i.e., exact match) and F1 score.

We also report hardware performance of our binary BERT using the model size in terms of megabyte (MB) and the number of operations in terms of FLOPs for a single inference run. To measure the number of operations for quantized models including our binary BERT, we approximate the multiplication between an m-bit weight and an n-bit activation as $m \times n/64$ FLOPs for a CPU with instruction size of 64-bit similar to (Bai et al., 2021; Zhou et al., 2016; Qin et al., 2022).

5.1.2 Architecture

The backbone architecture of our binary BERT is similar to that of BERT-base. More specifically, the number of transformer layers is set to 12, the hidden size to 768 and the number of attention heads in MHA to 12. We also use a fine-tuned BERT-base as the teacher model for knowledge distillation during the training phase. Unlike previously-proposed binarization methods where the binarization is performed on full-precision weights from pre-training, we use the pre-trained parameters as an initial point for the embedding module only whereas the rest of the network (i.e., MHA, FFN and classifier modules) are randomly initialized according to the smoking gun. We refer to our binary BERT with the aforementioned configuration as SGBERT.

5.1.3 Settings

For training our binary BERT, we use the AdamW optimizer with the weight decay of 1e-2, learning rate of 1e-4, gradient clipping of 2.5 and batch size of 16 for all datasets. It is worth mentioning that no data augmentation is used in our experiments. To allow a sufficient training for our binary model, we adopt more training epochs for each task, which is a common approach for training quantized models. More specifically, we use 1000 epochs for RTE, SST-2, CoLA, STS-B and MRPC, and 100 epochs for MNLI, QNLI, QQP and SQuAD v2.0. We also used a linear learning rate decay with the warmup portion of 0.1. We report the best validation accuracy obtained during the training phase in this paper.

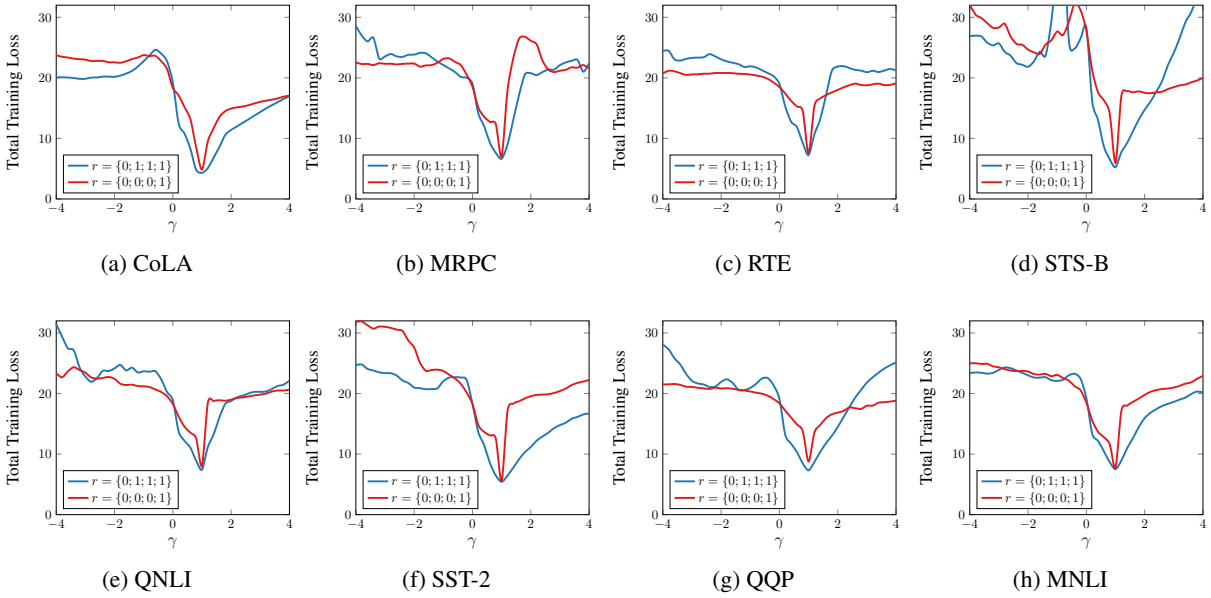
5.2 Experimental Results

5.2.1 Results on GLUE Benchmark

Table 2 demonstrates the performance results of SGBERT in terms of accuracy and efficiency on

Table 2: The comparison with quantized BERT models on the development set of GLUE.

Method	#Bits(E-W-A)	Size(MB)	FLOPs(G)	MNLI(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
BERT	32-32-32	418	22.5	84.9/85.5	91.4	92.1	93.2	59.7	90.1	86.3	72.2	83.9
Q-BERT (AAAI'20)	2-8-8	43	6.5	76.6/77.0	–	–	84.6	–	–	68.3	52.7	–
Q2BERT (EMC-2'19)	2-8-8	43	6.5	47.2/47.3	67.0	61.3	80.6	0	4.4	68.4	52.7	47.7
TinaryBERT (EMNLP'20)	2-2-8	28	6.4	83.3/83.3	90.1	–	–	50.7	–	87.5	68.2	–
BinaryBERT (ACL'21)	1-1-4	16.5	1.5	83.9/84.2	91.2	90.9	92.3	44.4	87.2	83.3	65.3	79.9
TinaryBERT (EMNLP'20)	2-2-2	28	1.5	40.3/40.0	63.1	50.0	80.7	0	12.4	68.3	54.5	45.5
BinaryBERT (ACL'21)	1-1-1	16.5	0.4	35.6/35.3	66.2	51.5	53.2	0	6.1	68.3	52.7	41.0
BiBERT (ICLR'22)	1-1-1	13.4	0.4	66.1/67.5	84.8	72.6	88.7	25.4	33.6	72.5	57.4	63.2
SGBERT (ours)	1-1-1	13.4	0.4	**74.3/75.2**	**86.6**	**82.7**	**91.4**	**36.9**	**70.1**	**77.2**	**61.7**	**72.9**



(a) CoLA (b) MRPC (c) RTE (d) STS-B

(e) QNLI (f) SST-2 (g) QQP (h) MNLI

Figure 3: The one dimensional training loss curves for the binary BERT with $r = \{0; 0; 0; 1\}$ and $r = \{0; 1; 1; 1\}$ on the GLUE benchmark. The models are trained under the same training setting.

the development set of each task in the GLUE benchmark. Our SGBERT trained with its fortuitous initialization significantly outperforms its qauntized counterparts such as BiBERT (Qin et al., 2022), BinaryBERT (Bai et al., 2021), TernaryBERT (Zhang et al., 2020), Q-BERT (Shen et al., 2020) and Q2BERT (Zafrir et al., 2019).

Q-BERT and Q2BERT are early quantization methods that compress the model parameters to low bit-width representation without KD assistant. Due to the disruptive effect of quantization on the initialization of BERT, these methods result in a large performance drop in accuracy performance and fall behind the existing quantization methods.

TernaryBERT uses both approximation-based and loss-aware ternarization methods to quantize weights into 2 bits. BinaryBERT, on the other hand, introduced a ternary weight splitting method for quantization of BERT. Both BinaryBERT and TernaryBERT use the minimum of 4 bits for representation of activations. For the sake of comparison with fully binary BERT, their official code was used in (Qin et al., 2022) to quantize activations into binary and ternary values. In Table 2, we report the performance of BinaryBERT and TernaryBERT from (Qin et al., 2022) for a fair comparison. Given the reported numbers, our binary BERT is far ahead of both BinaryBERT and TernaryBERT in terms of accuracy performance on each task.

BiBERT uses direction-matching distillation for accurate optimization of the binarization process. The accuracy performance of BiBERT is similar to that of our binary BERT when trained with pre-training weights, i.e., $r = \{0; 0; 0; 1\}$ (see Section 5.3). However, compared to the smoking gun of the hypothesis (i.e., $r = \{0; 1; 1; 1\}$), our SGBERT significantly outperforms BiBERT on each task and accordingly by a large margin in the average accuracy on the GLUE benchmark, leading to a new set of state-of-the-art results among fully binary BERT models.

Table 3: The experimental results on the development set (EM/F1) of SQuAD.

Method	#Bits(E-W-A)	Size(MB)	FLOPs(G)	SQuAD v2.0
BERT	32-32-32	418	22.5	74.4/77.6
BiBERT	1-1-1	13.4	0.4	49.9/49.9
SGBERT (ours)	1-1-1	13.4	0.4	**54.6/55.2**

Table 4: The development set results of binary BERT with different initializations on the GLUE benchmark.

$\{r_{emb}; r_{MHA}; r_{FFN}; r_{classifier}\}$	CoLA	STS-B	MRPC	RTE
$r = \{0; 0; 0; 1\}$	28.6	45.6	70.3	57.7
$r = \{0; 0; 1; 1\}$	32.3	67.1	75.8	61.0
$r = \{0; 1; 0; 1\}$	29.7	40.8	70.3	58.2
$r = \{0; 1; 1; 1\}$	36.9	70.1	77.2	61.7

5.2.2 Results on SQuAD Benchmark

Table 3 summarizes the results on the development set of SQuAD v2.0. We have also used the original code for BinaryBERT, TernaryBERT and BiBERT to measure the EM and F1 scores of these works on SQuAD v2.0. Our binary BERT outperforms BiBERT by a large margin on this task. Note that we use the official code of BiBERT to measure its performance on the SQuAD benchmark.

5.3 Ablation Study

So far, we have introduced a hypothesis stating that compression by reducing bit-width representation of BERT is not different from KD-based compression approaches; in either cases, the final performance of the network relies on the accurate knowledge transfer from the teacher model to the student model. In KD-based approaches, the compressed model (e.g., AdaBERT (Chen et al., 2020)) has a different shape compared to the original BERT model. As such, its parameters are usually initialized randomly and the information about unseen data is learned by KD. We showed that binary BERT can be trained in a similar manner, i.e., by relying on KD. We also showed that the key to obtain a decent performance is to randomly initialize the MHA and FFN modules along with the classifier. We referred to this supporting example for our hypothesis as the smoking gun that provides the best explanation why our hypothesis is true. The smoking gun suggests that initialization of binary BERT from full-precision pre-training leads to a bad global minima and the optimizer cannot avoid it, which is compatible with the finding of the experiments in (Liu et al., 2020). In other words, initialization of binary BERT from full-precision pre-training constructs a bad initialization without the knowledge of the true loss landscape. Such an undesirable priori justifies the priori preference for a better generalization of binary BERT as suggested by our hypothesis.

To illustrate the generalization capability of the smoking gun versus the binary BERT initialization from full-precision pre-training, we plot one-dimensional training loss curves for these two models similar to (Hao et al., 2019). Such one-dimensional curve represents the cross section of two-dimensional loss surface and the optimization direction. The loss curve is plotted by linear interpolation between θ_0 and θ_1 using the curve function $g(\gamma)$ as follows:

$$g(\gamma) = \mathcal{L}(\theta_0 + \gamma\delta), \tag{10}$$

where $\delta = \theta_1 - \theta_0$ is the optimization direction, $\mathcal{L}(\theta)$ is the loss function for the model parameters θ, θ_0 is the initialized parameters, θ_1 is the trained parameters and $\gamma \in [-4, 4]$ is a scaling factor. We initialize θ_0 with the initialization identifier $r = \{0; 1; 1; 1\}$ for the smoking gun and $r = \{0; 0; 0; 1\}$ for the binary BERT with initialization from pre-training. Figure 3 shows the one-dimensional training loss curves for these two binary models for 40 sampling points. The loss curves show more flatness for the optima of the smoking gun while those of binary BERT with initialization from pre-training shows more sharpness for the optima. As shown in previous studies (Hao et al., 2019; Li et al., 2018; Keskar et al., 2016), the flatness of the local optima is highly correlated with the generalization capability, confirming the better generalization of the smoking gun.

We have also studied the impact of different initialization on the accuracy performance of the GLUE benchmark as summarized in Table 4. The empirical results show that the binary BERT with $r = \{0; 1; 1; 1\}$ (i.e., SGBERT) performs significantly better compared to other initialization configurations.

6 Conclusion

In this paper, we showed that fully binarization of BERT initialized with parameters from pre-training generalizes poorly. Based on this observation, we put forward a hypothesis stating that partially-random initialization of BERT leads to a better generalization in the context of binarization. We identified the best initialization approach for training

binary BERT is to initialize the embedding layer's parameters from pre-training while randomly initializing the remaining modules (i.e., the MHA, FFN and classifier modules). We referred to this fortuitous initialization as the smoking gun that supports our hypothesis. Given the smoking gun, we managed to outperform existing fully binary models and achieve a new set of state-of-the-art results on both GLUE and SQuAD v2.0 benchmarks.

References

Amir Ardakani, Arash Ardakani, Brett Meyer, James J Clark, and Warren J Gross. 2022. Standard deviation-based quantization for deep neural networks. *arXiv preprint arXiv:2202.12422*.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael R. Lyu, and Irwin King. 2021. Binarybert: Pushing the limit of bert quantization. In *ACL/IJCNLP (1)*, pages 4334–4348.

Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2020. Adabert: Task-adaptive bert compression with differentiable neural architecture search. In *IJCAI*, pages 2463–2469.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, 28.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mitchell A Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing bert: Studying the effects of weight pruning on transfer learning. *arXiv preprint arXiv:2002.08307*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and understanding the effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4143–4152, Hong Kong, China. Association for Computational Linguistics.

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*.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31.

Shengchao Liu, Dimitris Papailiopoulos, and Dimitris Achlioptas. 2020. Bad global minima exist and sgd can reach them. *Advances in Neural Information Processing Systems*, 33:8543–8552.

JS McCarley, Rishav Chakravarti, and Avirup Sil. 2019. Structured pruning of a bert-based question answering model. *arXiv preprint arXiv:1910.06360*.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to bert embeddings during fine-tuning? *arXiv preprint arXiv:2004.14448*.

Haotong Qin, Yifu Ding, Mingyuan Zhang, Qinghua YAN, Aishan Liu, Qingqing Dang, Ziwei Liu, and Xianglong Liu. 2022. BiBERT: Accurate fully binarized BERT. In *International Conference on Learning Representations*.

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

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*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018.

GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. Ternary-BERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*.