# Prune Once for All: Sparse Pre-Trained Language Models

**Ofir Zafrir**
Intel Labs, Israel
ofir.zafrir@intel.com

**Ariel Larey**
Intel Labs, Israel
ariel.larey@intel.com

**Guy Boudoukh**
Intel Labs, Israel
guy.boudoukh@intel.com

**Haihao Shen**
Intel Corporation
haihao.shen@intel.com

**Moshe Wasserblat**
Intel Labs, Israel
moshe.wasserblat@intel.com

## Abstract

Transformer-based language models are applied to a wide range of applications in natural language processing. However, they are inefficient and difficult to deploy. In recent years, many compression algorithms have been proposed to increase the implementation efficiency of large Transformer-based models on target hardware. In this work we present a new method for training sparse pre-trained Transformer language models by integrating weight pruning and model distillation. These sparse pre-trained models can be used to transfer learning for a wide range of tasks while maintaining their sparsity pattern. We demonstrate our method with three known architectures to create sparse pre-trained BERT-Base, BERT-Large and DistilBERT. We show how the compressed sparse pre-trained models we trained transfer their knowledge to five different downstream natural language tasks with minimal accuracy loss. Moreover, we show how to further compress the sparse models' weights to 8bit precision using quantization-aware training. For example, with our sparse pre-trained BERT-Large fine-tuned on SQuADv1.1 and quantized to 8bit we achieve a compression ratio of 40X for the encoder with less than $1\%$ accuracy loss. To the best of our knowledge, our results show the best compression-to-accuracy ratio for BERT-Base, BERT-Large, and DistilBERT.

## 1   Introduction

Transformer-based pre-trained language models (LM) such as BERT [Devlin et al., 2019] and RoBERTa [Liu et al., 2019] have become the standard approach for a wide range of natural language processing (NLP) tasks. Recently, we witness the emergence of models, larger by several orders of magnitude, such as GPT-2 [Radford et al., 2019], T-NLG [Rosset, 2020], GPT-3 [Brown et al., 2020], and Switch-C [Fedus et al., 2021]. These models advance the state-of-the-art results in several NLP tasks such as question answering and text classification. However, this trend toward bigger models raises several concerns. As the computational and memory resources required to run inference increase with the model's size, it becomes very expensive and challenging to deploy these models in production environments and on edge devices. Moreover, these large amounts of computational resources incur a steep environmental cost [Strubell et al., 2019].

Model compression of large LM is a growing field of study as a result of these concerns. Weight pruning is a compression method that has been shown to be very effective at reducing the memory footprint of a model [Han et al., 2015, Zhu and Gupta, 2018]. However, weight pruning of large Transformer-based LMs to high sparsity ratios requires specialized pruning methods [Sanh et al.,

2020, Chen et al., 2020, Gordon et al., 2020, Lagunas et al., 2021]. Moreover, most of the pruning methods require task specific modifications and tuning to produce quality results.

Gordon et al. [2020] found that, in terms of accuracy, it does not matter whether BERT is pruned during the pre-training phase or during the transfer learning phase. This suggests that a LM can be pruned once during pre-training and then fine-tuned to any downstream task without task-specific tuning.

In this paper, we present a new method, Prune Once for All (Prune OFA), that leverages weight pruning and model distillation to produce pre-trained Transformer-based language models with a high sparsity ratio. We apply our method to BERT-Base, BERT-Large and DistilBERT [Sanh et al., 2019] to produce sparse pre-trained models for these model architectures. We then show how these sparse models can be fine-tuned to produce task-specific sparse models with minimal accuracy loss for SQuADv1.1 [Rajpurkar et al., 2016] as well as for four tasks from the GLUE Benchmark [Wang et al., 2018]. We also show that it is possible to further compress the models using quantization-aware training to achieve state-of-the-art results in terms of compression-to-accuracy ratio.

The main contributions of this work are threefold: 1) We introduce a new architecture-agnostic method of training sparse pre-trained language models. 2) We demonstrate how to fine-tune these sparse models on downstream tasks to create sparse and quantized models, removing the burden of pruning and tuning for a specific language task. 3) We publish our compression research library with example scripts to reproduce our work for other architectures, along with our sparse pre-trained models presented in this paper.

## 2   Related work

Large language models are over-parameterized and difficult to deploy. Therefore, the problem of compressing these models with minimum accuracy loss for downstream tasks is widely explored. Sanh et al. [2020] suggests the Movement Pruning method designed especially for transfer learning. Neural Magic implements Gradual Magnitude Pruning.[1] Both methods suggest pruning BERT-Base while fine-tuning to downstream tasks paired with model distillation, and present results showing 90% sparsity for several tasks. However, both methods require a long fine-tuning time as well as tuning pruning related hyper-parameters for every task. Our method, on the other hand, requires no tuning of special pruning hyper-parameters per task because we prune the model once for all tasks. Furthermore, we present better or comparable results at a much lower computation budget at the transfer learning phase. Gordon et al. [2020] explored the effect of weight pruning during transfer learning and concluded that pruning BERT-Base at the pre-training phase does not degrade the performance of the model compared to pruning at fine-tuning. We improve upon the suggested method and present better results at a much higher sparsity ratio. Chen et al. [2020] explored the Lottery Ticket Hypothesis [Frankle and Carbin, 2018] for BERT pre-trained models. More specifically, they analyzed the possibility of finding winning tickets in a BERT-Base pre-trained model that transfer to other downstream tasks. The authors concluded that winning tickets found while pre-training on a Masked-LM task, transfer well to other downstream tasks. Lagunas et al. [2021] presented a structured pruning method, removing rows, columns and attention heads, while achieving less than 1% loss in F1 for a BERT architecture on SQuADv1.1. Mishra et al. [2021] performed structured 2:4 pruning on BERT while further pre-training BERT; The method produced a 50% sparse model which can be fine-tuned without accuracy loss. Michel et al. [2019] explored the significance of each head in the multi-head attention mechanism of BERT and presented a method for pruning attention heads with their associated weights.

Other works propose knowledge distillation to compress Transformer models to a smaller dense counter part that can be tuned to downstream tasks [Sanh et al., 2019, Jiao et al., 2020, Sun et al., 2020]. Quantization of Transformer-based language models is also a well known method for compression. Shen et al. [2020] proposes a method to quantize BERT at a different bit-width per layer. Other works implement quantization-aware training to quantize BERT to 8bits [Kim et al., 2021, Zafrir et al., 2019]. Zhang et al. [2020] created a method of producing a ternary weight BERT. Kim and Hassan [2020] presented a compression pipeline for Transformer models that includes model distillation, quantization and head pruning.

---

[1] https://github.com/neuralmagic/sparseml/tree/main/integrations/huggingface-transformers

## 3 Weight pruning

Weight pruning is the process of forcing some of the neural network's weights to zero. Weight pruning can be either unstructured where individual weights are pruned, or structured where structured groups of weights are pruned, e.g. blocks, channels, layers. Weight pruning results in sparse neural networks that reduce the computation and the memory footprint of the trained model.

In this paper we focus on unstructured weight pruning. Zhu and Gupta [2018] presented a method of Gradual Magnitude Pruning (GMP) to gradually prune weights with low magnitude during training. During training, every $f$ steps the lowest magnitude weights are pruned until reaching the temporal sparsity ratio $s_t$ for time step $t$, defined by

$$s_t = s_f + (s_i - s_f) \left(1 - \frac{t - t_s}{t_e - t_s}\right)^3 \qquad (1)$$

where $s_i$ and $s_f$ are the initial and final sparsity ratios, and $t_s$ and $t_e$ are the pruning start and end time steps.

In a recent paper, Renda et al. [2020] presented a pruning algorithm based on IMP (Iterative Magnitude Pruning) [Han et al., 2015] and Learning Rate Rewinding (LRR). IMP consist of two steps: prune a portion of the model and continue fine-tuning it to recover from the induced pruning error. These two steps are repeated until the desired sparsity ratio is achieved. In LRR, the learning rate scheduler is rewound to its state before the pruning step at the beginning of the fine-tune step. We propose to incorporate the principle of learning rate rewinding into GMP by rewinding the learning rate scheduler to its state at time $t_s$ every $f$ steps. After $t_e$ the scheduler continues with its original setting until training ends. Appendix C visualizes how LRR combined with GMP modifies the learning rate scheduler.

## 4 Knowledge distillation

Knowledge distillation, introduced by Hinton et al. [2015], is the process of training a student network to reproduce the behavior of a teacher model. When distillation is used to fit the predictions of the teacher model, soft cross-entropy loss between the student and the teacher soft probabilities is computed as follows:

$$\mathcal{L}_{kd} = -\sum_i t_i \cdot \log\left(s_i\right) \qquad (2)$$

where $s_i$ is the soft-probability estimated by the student, and $t_i$ is its corresponding soft-probability estimated by the teacher for the same input sample. The soft probabilities are calculated using a `softmax` function with temperature $T$.

Commonly, the teacher is a large model that achieves high performance, and the student is based on a smaller architecture. In this paper, we propose to leverage the model distillation method for the pruning process. We focus on an approach where both teacher and student share the same architecture, but differ in their sparsity ratio. In this case, the teacher is a dense model that was trained on a target task, and the student is a model with a fixed sparsity or one undergoing pruning. Distillation-during-pruning can be applied to language models during both the pre-training and fine-tuning phases. In the pre-training phase, the teacher is a pre-trained language model, and in the fine-tuning phase, the teacher is a language model fine-tuned to a target task.

## 5 Prune Once for All

The notion of pruning language models such as BERT [Devlin et al., 2019] while pre-training has already been explored by Chen et al. [2020] and Gordon et al. [2020]. However, fine-tuning the sparse model for a specific language task resulted in either poor results or a low sparsity ratio. In this section we will introduce our novel method, Prune OFA, for creating sparse pre-trained language models that can be later fine-tuned to downstream tasks with minimal accuracy loss at high sparsity ratios. A visualization of our method is presented in Figure 1. The method takes as its input a pre-trained language model and outputs a sparse language model of the same architecture. The method consists of two steps, teacher preparation and student pruning. The sparse pre-trained model we trained is the
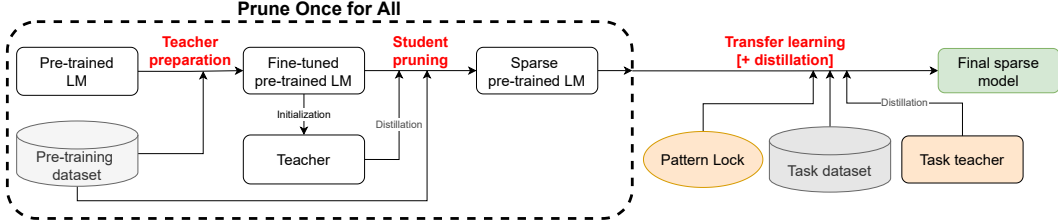
Figure 1: Prune OFA method

model we use for transfer learning while maintaining its sparsity pattern. We call the method Prune Once for All since we show how to fine-tune the sparse pre-trained models for several language tasks while we prune the pre-trained model only once.

**Teacher preparation**    The first step of Prune OFA is to obtain a model optimized on the pre-training dataset for some pre-training task with objective $\mathcal{L}_{\mathcal{PT}}$ as shown in Figure 1.[2] The same dataset will be used for pruning the student in the next step. This model will initialize the student and teacher models in the student pruning step.

**Student pruning**    A student model is initialized from the teacher prepared in the teacher preparation step. The student is then fine-tuned on a linear combination of the pre-training task, from the teacher preparation step, and the knowledge distillation objective $\mathcal{L}_{kd}$:

$$\mathcal{L} = \lambda_{PT}\mathcal{L}_{PT} + \lambda_{kd}\mathcal{L}_{kd} \tag{3}$$

while being pruned with GMP + LRR methods. The output model of this process is a sparse pre-trained LM that can be used without additional pruning for transfer learning to produce sparse models for a specific downstream task.

**Pattern-lock**    We wish to keep the sparsity pattern of the sparse pre-trained model created by Prune OFA in place during the fine-tuning process. We propose a method called pattern-lock that prevents the zeros found in the model from changing while training the model. Pattern-lock is described in more details in Appendix B.

# 6    Experimental setup

**Datasets**    We use the English Wikipedia dataset (2500M words) for training the models on the pre-training task. We split the data into train (95%) and validation (5%) sets. Both sets are pre-processed as described in the models' original papers [Devlin et al., 2019, Sanh et al., 2019]. We process the data to use the maximum sequence length allowed by the models, however, we allow shorter sequences at a probability of 0.1. We evaluate our sparse pre-trained models on several common benchmarks for transfer learning; a question answering task, SQuADv1.1 containing 89K training examples [Rajpurkar et al., 2016], and the following text classification tasks from the GLUE Benchmark: MNLI, QQP, QNLI and SST-2 containing 393K, 364K, 105K, and 67K training examples respectively [Wang et al., 2018, Williams et al., 2018, Iyer et al., 2017, Socher et al., 2013].

**Applying Prune Once for All**    We showcase our method by applying Prune OFA on three different architectures of different sizes; BERT-Base, BERT-Large and DistilBERT. Since we don't have the original processed training data used to train BERT-Base, BERT-Large and DistilBERT we run an additional step to fine-tune the pre-trained models using the processed training data we prepared. Next, we execute the student pruning step to obtain our sparse pre-trained models. We prune BERT-Base and DistilBERT to {85%, 90%} sparsity ratios and BERT-Large to a 90% sparsity ratio. Pruning is applied to all Linear layers in the Transformer encoder including the pooler layer if it exists. Exact hyper-parameters and additional details are summarized in Appendix E

---

[2]For example, the pre-training task for BERT-Base is masked language-modeling combined with next sentence prediction.

Table 1: Prune OFA BERT-Base results compared to other pruning methods

| Model | Sparsity | Transfer with KD | SQuAD | | MNLI (m/mm) | | SST-2 | QNLI | QQP | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | | | EM | F1 | Acc | Acc | Acc | Acc | Acc | F1 |
| Reference | 0% | | 80.80 | 88.50 | 84.06 | 84.51 | 92.13 | 91.16 | 91.20 | 88.13 |
| Chen et al. [2020] | 70% | | N/A | 86.54 | **82.59** | N/A | **91.86** | 89.44 | 90.03 | N/A |
| Gordon et al. [2020] | 80% | | N/A | N/A | 75.90 | N/A | 88.10 | 85.30 | 86.90 | N/A |
| Prune OFA | 85% | | **78.59** | **86.63** | 81.67 | 82.53 | 91.34 | **89.95** | 90.69 | 87.41 |
| Fine-tune pruning | 85% | + | 78.00 | 86.16 | 82.45 | 83.05 | 88.82 | 87.79 | 90.87 | 87.65 |
| Prune OFA | | + | **81.10** | **88.42** | **82.71** | **83.67** | **91.46** | **90.34** | **91.15** | **88.00** |
| Prune OFA +QAT | 85% | + | 80.84 | 88.24 | 81.40 | 82.51 | 91.46 | 89.76 | 91.09 | 88.01 |
| Neural Magic[3] | | + | 79.40 | 87.20 | N/A | N/A | N/A | N/A | N/A | N/A |
| Sanh et al. [2020] | 90% | + | 76.60 | 84.90 | 81.20 | 81.80 | N/A | N/A | 90.20 | 86.80 |
| Prune OFA | | + | **79.83** | **87.25** | **81.45** | **82.43** | 90.88 | 89.07 | **90.93** | **87.72** |

Table 2: Prune OFA BERT-Large results

| Model | Sparsity | SQuAD | | MNLI (m/mm) | | SST-2 | QNLI | QQP | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | | EM | F1 | Acc | Acc | Acc | Acc | Acc | F1 |
| Reference | 0% | 83.99 | 90.93 | 86.39 | 86.58 | 93.54 | 92.42 | 91.59 | 88.67 |
| Prune OFA | 90% | 83.35 | 90.20 | 83.74 | 84.20 | 92.95 | 91.39 | 91.48 | 88.43 |
| Prune OFA + QAT | 90% | 83.22 | 90.02 | 83.47 | 84.08 | 92.72 | 91.45 | 91.41 | 88.36 |

**Transfer learning**    After creating our sparse pre-trained models we fine-tune them to the following NLP tasks: SQuADv1.1, QNLI, MNLI, SST-2 and QQP. We use default hyper-parameters for each task and conduct a grid search for learning rate, weight decay, warmup ratio and number of training epochs hyper-parameters. For each task we report the mean of two different runs with different seeds that achieved the best result on the task's development set. We further improve the results of our sparse models by integrating knowledge distillation. For each task and model, we create a task teacher based on the original dense pre-trained model fine-tuned to the task. For SQuADv1.1 and QQP we report the result that maximizes F1, and for MNLI we report the result that maximizes the mismatched accuracy. For exact hyper-parameters and additional details see Appendix E.

**Comparison with fine-tune pruning**    We compare our Prune OFA method with fine-tune pruning where we prune the dense pre-trained model during fine-tuning to a downstream task. For that purpose, we implement GMP pruning coupled with knowledge distillation and run experiments using the same teacher and hyper-parameters used in the Prune OFA transfer learning experiments.

**Quantization**    We implemented quantization-aware training similar to Q8BERT [Zafrir et al., 2019]. For details on the differences between our method and Q8BERT see Appendix D. For each task, we pick the best-performing model for this task and perform quantization-aware training on it. We use slightly different hyper-parameters for this training session as described in Appendix E.2. We report the mean of two different runs with different seeds that achieved the best result.

## 7   Results

In Table 1 we present our experimental results for pruning BERT-Base to a 85% and 90% sparsity ratio using Prune OFA. We also present results of other pruning methods applied to BERT-Base as well as results of the fine-tune pruning experiments we conducted. Results not marked in the column Transfer with KD do not use model distillation in the transfer learning phase. The best result in each category is marked with bold font. We observe that our method achieves better results than other previous pruning works while pre-training at a higher sparsity ratio. When comparing our Prune OFA method against other fine-tune pruning methods, we observe that our method produces the best results at 85% and 90% sparsity ratios. Moreover, we show accuracy degradation lower than

---

[3]Results taken from Neural Magic's sparse model zoo: `https://sparsezoo.neuralmagic.com/`

Table 3: Prune OFA DistilBERT results compared to fine-tune pruning

| Model | Sparsity | SQuAD | | MNLI (m/mm) | | SST-2 | QNLI | QQP | |
|-------|----------|-------|------|-------------|------|-------|------|------|------|
| | | EM | F1 | Acc | Acc | Acc | Acc | Acc | F1 |
| Reference | 0% | 77.70 | 85.80 | 82.20 | N/A | 91.30 | 89.20 | N/A | 88.50 |
| Fine-tune pruning | 85% | 76.16 | 84.55 | 81.22 | 81.92 | 88.88 | 86.60 | 90.18 | 86.80 |
| Prune OFA | | **78.10** | **85.82** | **81.35** | **82.03** | **90.60** | **88.31** | **90.29** | **86.97** |
| Prune OFA +QAT | 85% | 77.03 | 85.13 | 80.66 | 81.14 | 88.93 | 87.97 | 90.22 | 86.92 |
| Fine-tune pruning | 90% | 74.63 | 83.42 | 80.47 | 81.32 | 88.25 | 84.91 | 89.97 | 86.57 |
| Prune OFA | | **76.91** | **84.82** | **80.68** | **81.47** | **90.02** | **87.66** | **90.05** | **86.67** |
| Prune OFA +QAT | 90% | 75.62 | 83.87 | 78.80 | 80.40 | 88.47 | 87.20 | 89.97 | 86.63 |

$1\%$ relative to the results of the dense pre-trained model at $85\%$ sparsity with the exception of the MNLI-matched benchmark. Note that for MNLI, the reported results were selected based on the best model's mismatched accuracy found in our grid-search; when searching for the best matched result we reduce the accuracy gap to $\sim 1\%$ accuracy loss at the expense of increased accuracy loss for mismatched: 83.09/83.36 (m/mm).

The results for pruning BERT-Large to a $90\%$ sparsity ratio are presented in Table 2. These results fall within the range of $1\%$ accuracy loss for all tasks but the MNLI task. We conclude that the $90\%$ sparse BERT-Large (30.2M non-zero parameters) model we trained has better accuracy in comparison to dense BERT-Base (85M non-zero parameters).

Our results for pruning DistilBERT to a $85\%$ and $90\%$ sparsity ratio are presented in Table 3 with our results for the fine-tune pruning experiments we conducted. In both sparsity ratios our method produces better accuracy results compared to fine-tune pruning (the best result in each category is marked with bold font). Furthermore, at the $85\%$ sparsity ratio our results are within the range of $1\%$ relative accuracy loss in all tasks but QQP.

Tables 1, 2 and 3 present quantization results, designated with a +QAT suffix. Applying quantization-aware training on our resultant sparse models decreases the accuracy of the model further by an average of $0.67\%$ relative to the full precision model's accuracy. The results for the $85\%$ sparse model +QAT are better than for the $90\%$ sparse model with full precision in all the tasks for BERT-Base and in 3/5 tasks for DistilBERT. Furthermore, the $85\%$ sparse and quantized model are smaller than the $90\%$ sparse model by a factor of $0.375$.

An ablation study was conducted to test how each component of the Prune OFA method affects the ability of the pre-trained model to transfer its knowledge to downstream tasks, as described in Appendix A.

# 8 Conclusion and future work

We introduced Prune OFA, an architecture-agnostic method for producing sparse pre-trained language models. We also showed how these sparse models can be used to obtain fine-tuned sparse models without the burden of task-specific pruning. Our results suggest that using these sparse pre-trained models for transfer learning produces results with minimal performance degradation loss w.r.t their dense counterpart for a variety of NLP tasks. We further demonstrated that integrating quantization can lead to more efficient sparse and quantized models at a small cost to the model's accuracy.

A possible direction for future research is to explore whether a large and sparse pre-trained model is better at capturing and transferring natural language knowledge than a smaller dense model of the same architecture with similar non-zero parameters count.

We hope that the release of our code and sparse pre-trained models to the community will help develop more efficient models.

# References

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, Z. Wang, and M. Carbin. The lottery ticket hypothesis for pre-trained bert networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15834–15846. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/b6af2c9703f203a2794be03d443af2e3-Paper.pdf`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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, 2019.

W. Fedus, B. Zoph, and N. Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

M. Gordon, K. Duh, and N. Andrews. Compressing bert: Studying the effects of weight pruning on transfer learning. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 143–155, 2020.

S. Han, J. Pool, J. Tran, and W. J. Dally. Learning both weights and connections for efficient neural network. In *NIPS*, pages 1135–1143, 2015. URL `http://papers.nips.cc/paper/5784-learning-both-weights-and-connections-for-efficient-neural-network`.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL `http://arxiv.org/abs/1503.02531`.

S. Iyer, N. Dandekar, K. Csernai, et al. First quora dataset release: Question pairs. *data. quora. com*, 2017.

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 *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4163–4174, 2020.

S. Kim, A. Gholami, Z. Yao, M. W. Mahoney, and K. Keutzer. I-bert: Integer-only bert quantization. *ICML*, 2021.

Y. J. Kim and H. Hassan. Fastformers: Highly efficient transformer models for natural language understanding. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 149–158, 2020.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

F. Lagunas, E. Charlaix, V. Sanh, and A. M. Rush. Block pruning for faster transformers, 2021.

Q. Lhoest, A. V. del Moral, P. von Platen, T. Wolf, Y. Jernite, A. Thakur, L. Tunstall, S. Patil, M. Drame, J. Chaumond, J. Plu, J. Davison, S. Brandeis, T. L. Scao, V. Sanh, K. C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, S. Liu, S. Lesage, L. Debut, T. Matussière, C. Delangue, and S. Bekman. huggingface/datasets: 1.11.0, July 2021. URL `https://doi.org/10.5281/zenodo.5148649`.

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. *arXiv preprint arXiv:1907.11692*, 2019.

P. Michel, O. Levy, and G. Neubig. Are sixteen heads really better than one? *Advances in Neural Information Processing Systems*, 32:14014–14024, 2019.

A. Mishra, J. A. Latorre, J. Pool, D. Stosic, D. Stosic, G. Venkatesh, C. Yu, and P. Micikevicius. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*, 2021.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, 2016.

A. Renda, J. Frankle, and M. Carbin. Comparing rewinding and fine-tuning in neural network pruning. *ICLR*, 2020.

C. Rosset. Turing-nlg: A 17-billion-parameter language model by microsoft. 2020. URL https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

V. Sanh, T. Wolf, and A. Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33, 2020.

S. Shen, Z. Dong, J. Ye, L. Ma, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821, 2020.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

E. Strubell, A. Ganesh, and A. McCallum. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, 2019.

Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, 2020.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. 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, 2018.

A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2018*, pages 1112–1122. Association for Computational Linguistics (ACL), 2018.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

O. Zafrir, G. Boudoukh, P. Izsak, and M. Wasserblat. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, pages 36–39, 2019. doi: 10.1109/EMC2-NIPS53020.2019.00016.

Table 4: Prune OFA 85% sparse BERT-Base ablation study results

| Teacher preparation | LRR | Pre-train distillation | Transfer distillation | SQuAD EM | F1 | MNLI (m/mm) Acc | Acc |
|---|---|---|---|---|---|---|---|
| | | | | 78.11 | 86.13 | 81.14 | 81.74 |
| + | | | | 78.00 | 86.31 | 81.22 | 82.01 |
| + | + | | | 78.41 | 86.51 | 81.39 | 82.01 |
| + | | + | | 78.30 | 86.41 | 81.57 | 82.13 |
| + | + | + | | 78.59 | 86.63 | 81.67 | 82.53 |
| + | | | + | 80.77 | 88.08 | 82.20 | 82.83 |
| + | + | + | + | 81.10 | 88.42 | 82.71 | 83.67 |

W. Zhang, L. Hou, Y. Yin, L. Shang, X. Chen, X. Jiang, and Q. Liu. Ternarybert: Distillation-aware ultra-low bit bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, 2020.

M. Zhu and S. Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *ICLR*, 2018.

# A  Ablation study

In this section we analyze how each step of the Prune OFA method affects the final results. We compare the models in the same fashion as in Section 7, by comparing the different results of the sparse pre-trained models on downstream tasks. In the ablation study we focus on BERT-Base pruned to 85% fine-tuned to SQuADv1.1 and MNLI. All the results from the ablation study are present in Table 4.

**Teacher preparation**   The teacher preparation step is only done in the case the original processed training data of the pre-trained model is not available. Since our objective is to prune the model it is always better to start from a model that is better optimized to the data used for pruning, hence the teacher preparation step. To measure the effect of the teacher preparation step we prune two models, a model that uses BERT-Base pre-trained model as initialization, and a model that uses the output of the teacher preparation step as initialization. Then, we fine-tune them both to SQuADv1.1 and MNLI tasks and compare their results. We see notable improvement when executing with the teacher preparation step in both tasks.

**Student pruning**   We compare the results of a model pruned with LRR to a model that was pruned without LRR, meaning the learning rate schedule remained the default linear decay with warmup schedule. For SQuADv1.1 we observe a significant improvement in both benchmarks. However, in MNLI case we don't see any improvement in the mismatched accuracy which we try to maximize, but there is a significant improvement in the matched accuracy. We observe that applying knowledge distillation during the student pruning step improves both tasks results. Knowledge distillation seems less significant in SQuADv1.1 case and more significant in MNLI case. In addition, we see that combining LRR and knowledge distillation achieves better results than either method separately. We conclude that applying LRR while pruning improves fine-tuning results and therefore a crucial part of our algorithm.

**Transfer learning with knowledge distillation**   We saw that using knowledge distillation while fine-tuning to downstream tasks improves the results significantly. We test whether our method still improves accuracy results of sparse models when fine-tuned with model distillation. From the results at the bottom of Table 4 we deduce that our method is orthogonal to knowledge distillation while fine-tuning and improves the accuracy results of both tasks further.
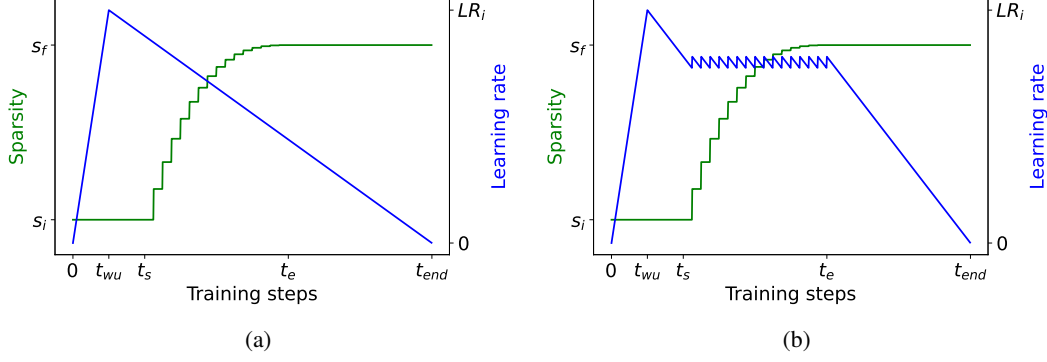
Figure 2: Learning rate and sparsity scheduler. Both figures show a linear decay learning rate scheduler with $t_{wu}$ warmup steps against a sparsity scheduler defined by Equation 1. (a) learning scheduler without rewinding. (b) learning scheduler with rewinding

## B  Pattern-lock details

Following is a detailed description of the Pattern-lock method used when fine-tuning our sparse pre-trained models. Before training, Pattern-lock method initializes a mask $M^l$ for each sparse layer $l$ with weight $W^l$, representing the layer's sparsity pattern.

$$M_{uv}^l = \begin{cases} 1 & W_{uv}^l \neq 0 \\ 0 & W_{uv}^l = 0 \end{cases} \tag{4}$$

Then, while training, the loss $\mathcal{L}$ gradient w.r.t the weights is modified to

$$\overline{\frac{\partial \mathcal{L}}{\partial W_{uv}^l}} = \begin{cases} \frac{\partial \mathcal{L}}{\partial W_{uv}^l} & M_{uv}^l = 1 \\ 0 & M_{uv}^l = 0 \end{cases} \tag{5}$$

ensuring that a weight that was initially 0 will stay 0 through-out fine-tuning.

## C  Visualization of Learning Rate Rewinding with Gradual Magnitude Pruning

Figure 2 demonstrates how a linear decay learning rate scheduler with warmup is modified with LRR against the same scheduler without LRR.

## D  Quantization method differences from Q8BERT

We have implemented our own version of quantization-aware training which is similar to Q8BERT with the following differences: 1) Activations are quantized using asymmetric quantization instead of symmetric quantization. 2) Embedding vectors are not quantized and represented in full precision. 3) Models are quantized after fine-tuning to a downstream task in a seperate learning session.

## E  Reproducibility

### E.1  Implementation

Our Prune OFA method, GMP, model distillation and quantization-aware training are implemented in our Model Compression Research Package using `PyTorch` [Paszke et al., 2019].[4] Our library offers several architecture agnostic pruning and other compression methods that can be plugged into any training session with a few lines of code. We invite the researches community to use our library to accelerate their research in pruning and neural networks compression.

---

[4]`https://github.com/IntelLabs/Model-Compression-Research-Package`

Table 5: Hyper-parameters used with Prune OFA

| Hyper-parameter | Value |
|---|---|
| Warmup ratio | 0.01 |
| Batch size | 256 |
| Weight decay | 0.01 |
| Max steps | 100k |
| Learning rate decay | Linear + LRR |
| Sequence length | 512 |
| $\lambda_{PT}$ | 0.5 |
| $\lambda_{kd}$ | 0.5 |
| Temperature | 2.0 |
| Pruning start | 0 |
| Pruning policy end | 50k |
| Pruning end | 80k |
| Pruning interval | 1k |

Table 6: Hyper-parameters used for transfer learning

| Hyper-parameter | SQuAD | GLUE |
|---|---|---|
| Learning rate | {1.5e-4, 1.8e-4} | {1e-4, 1.2e-4, 1.5e-5} |
| Batch size | 12 | 32 |
| Weight decay | {0, 0.01} | |
| Epochs | 8 | {3, 6, 9} |
| Learning rate decay | Linear | |
| Warmup ratio | {0, 0.01, 0.1} | |
| Sequence length | 384 | 128 |
| $\lambda_{PT}$ | 0.0 | |
| $\lambda_{kd}$ | 1.0 | |
| Temperature | 2.0 | |

We use the `HuggingFace/transformers` library and the available example scripts to train our Transformer-based models [Wolf et al., 2020]. We have modified the example scripts to include our methods and make them available in our library's examples.

All the datasets mentioned in the paper are downloaded and processed using the `HuggingFace/datasets` library [Lhoest et al., 2021].

### E.2 Training details & hyper-parameters

**Teacher preparation** We execute the teacher preparation step on all models. The pre-training objectives for both BERT models and DistilBERT are the same as in the original paper. For BERT models, the objectives are masked language-modeling (MLM) and next sentence prediction (NSP), and for DistilBERT the objective is MLM only. The hyper-parameters used are detailed in Table 5. We use Adam optimizer [Kingma and Ba, 2015] with learning rates {5e-5, 1e-4, 1e-4} for {BERT-Base, BERT-Large, DistilBERT}.

**Student pruning** We run student pruning with the same objectives, hyper-parameters and optimizer we used at the teacher preparation step (Table 5) with learning rates {1.5e-4, 1e-4, 1.5e-4} for {BERT-Base, BERT-Large, DistilBERT}.

**Transfer learning** For transfer learning experiments of either Prune OFA or fine-tune pruning we use the hyper-parameters in Table 6 coupled with Adam optimizer. When combining knowledge distillation in the transfer learning phase, in our experiments we found that it is best to optimize only on knowledge distillation objective and ignore the ground truth labels.

Table 7: Hyper-parameters used for quantization-aware training

| Hyper-parameter | SQuAD | GLUE |
|---|---|---|
| Learning rate | {1e-6, 1e-5} | {5e-8, 1e-7, 1e-6, 1e-5} |
| Batch size | 12 | 32 |
| Weight decay | {0, 0.01} | |
| Epochs | 2 | 3 |
| Learning rate decay | Linear | |
| Warmup ratio | {0, 0.01, 0.1} | |
| Sequence length | 384 | 128 |
| $\lambda_{PT}$ | 0.0 | |
| $\lambda_{kd}$ | 1.0 | |
| Temperature | 2.0 | |

**Quantization** For quantization-aware training experiments of Prune OFA we use the hyper-parameters in Table 7 coupled with Adam optimizer.