Efficient Language Model Pre-Training: A Survey

Marcel Braasch
Technische Universität München
marcelbraasch@gmail.com

1 Introduction

With the emergence of modern deep learning methods, natural language processing (NLP) has seen a vital boost in recent years. Pre-trained language models (PLMs) paired with transformers units currently dominate the NLP landscape (Devlin et al., 2018; Radford et al., 2019; Vaswani et al., 2017). PLMs follow a clear paradigm, that is, pre-training followed by a fine-tuning phase. During pre-training, PLMs solve an unsupervised prediction task in an auto-encoding fashion given an abundance of data. This procedure is extremely resource hungry and typically requires training for many days on large GPU clusters. For example, BERT pre-training takes 54 days to train on a single TPU (You et al., 2019b). During fine-tuning, the model is trained end-to-end on an arbitrary task such as natural language inference or question answering. Since datasets at hand are comparably small, fine-tuning PLMs is fairly practicable.

The rationale for this work is manifold, though the guiding principle is clear: there is a need for more efficient deep learning methods in PLMs. In the face of climate change, (**1**) energy used is often not based on renewable resources, and even if they were, (**2**) there might be better alternatives for energy usage. Further, only a selected group of people is able to work on these methods (**3**) posing major ethical concerns in terms of accessibility and (**4**) hindering research in a crucial research directions (Strubell et al., 2019).

2 Approach

This survey is supposed to help overcoming the bottleneck of prohibitive pre-training methods. To the best of my knowledge, there is no such work conducted before. I provide a review of current progress in efficient neural network training with an emphasis on pre-training deep language models (LMs) on a single GPU. It is important to note that not all methods introduced have seen play in PLMs. Yet, this survey shall serve as a starting point to assessing current approaches and possibly adapting these for efficient pre-training. Moreover, some ideas are borrowed from the computer vision community, which has proven to be an effective source of inspiration before.

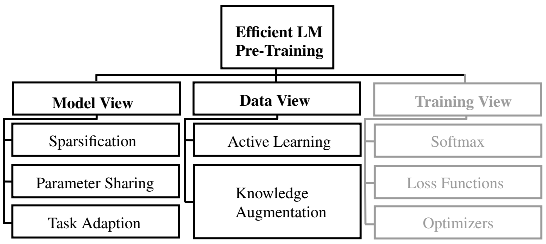

Figure 1: A taxonomy for efficient pre-training approaches introduced in this work. The taxonomy is based on a broad review of literature in the field and covers all (to the best of my knowledge) previously explored directions. The training view is greyed out since it is not covered in this work.

The review is based on a taxonomy covering three different views. First, I discuss the **model view**, comprised of compression, sparsification, parameter sharing and task adaption. Second, I proceed with the **data view** including active learning and knowledge augmentation. Third, the **training view**, including softmax approximations, objective functions and optimizers is of relevance. The last is rather technical and theory-heavy. I mainly focus on empirical methods, thus the training view is not covered in this work. A visualization of the taxonomy can be found in Figure 1.

3 Related Work

The goal of this analysis is locating promising *single GPU* training approaches. Therefore, distributed and federated training approaches are omitted entirely. Nevertheless, I recommend Dean et al. (2012) and Sattler et al. (2019) to the reader if this is of interest. Large batch adapted learning rate algorithms such as LARS, LANS or LAMB are not covered either (You et al., 2019b; Zheng et al., 2020). Quantization methods are mainly applicable at reducing energy cost and require heavy low-level engineering, thus will be omitted (Zafrir et al., 2019; Hubara et al., 2017). Efficient inference methods are already broadly covered by compression algorithms for neural networks (Buciluǎ et al., 2006). Knowledge distillation (Hinton et al., 2015; Gou et al., 2021), parameter pruning (Louizos et al., 2017; Hoefler et al., 2021), quantization (Lin et al., 2016; Gong et al., 2014; Han et al., 2015), as well as low-rank approximation methods (Yu et al., 2017; Sprechmann et al., 2015) are the most common approaches in this field. Efficient methods for fine-tuning are not covered either (Houlsby et al., 2019), since good fine-tuning results can already be obtained in tractable time.

It is important to note that the methods presented hereafter are orthogonal to multi-node procedures, thus can be combined ad libitum.

4 Data View

Examining deep LM pre-training from a data perspective includes all methods which do not directly affect the network architecture or training procedure. This mainly includes choosing the most informative samples as well as augmenting the data at hand or helping the model make more sense of it (though this could be interpreted as belonging to the model view, as well).

4.1 Knowledge Augmentation

Inspired by cognitive systems and computer architecture, knowledge augmentation (KA) methods have seen early play in deep learning (Graves et al., 2014). Generally, a model is extended by a knowledge source. This source augments the patterns induced by the data. Often, this leads to a gain in accuracy or performance. In pre-trained LMs various flavors have recently been introduced.

Many approaches extend models by external memories. They may be differentiable, accessible and updatable at all times, similar to a short-term memory (Févry et al., 2020; Guu et al., 2020; Graves et al., 2014). Other work suggests to induce external resources uncoupled from the network and data such as ontologies or parser outputs (Xiong et al., 2019; Peters et al., 2019; Wang et al., 2020).

Clearly, high entropy is crucial for fast convergence. But, increasing model complexity introduces overhead. If information comes at the cost of model complexity efficiency may be impeded and convergence gains nullified. Therefore it is substantial to find a trade-off between complexity and information, in favor of information. Most of the work shown above does not consider efficiency at all and aims at increasing accuracy. *Taking Notes on the Fly (TNF)* is one of the only works aiming at reducing time-complexity in PLMs.

4.1.1 Taking Notes on the Fly (TNF)

While most current LM approaches focus on increased accuracy, few focus on efficient LM pre-training. Clark et al. (2020)'s ELECTRA model changes the MLM task at hand from a generative to a discriminative setting. This substantially lowers complexity to gain a significant speed-up (see Section 5.3). Augmenting difficult samples by contextual knowledge can be interpreted as reducing task complexity, too.

Wu et al. (2020) suspect that rare words significantly contribute to a more challenging training setting. Words follow a heavy-tail distribution, i.e., most words have low occurrence while few words dominate (Piantadosi, 2014). Interpreted differently, low-occurrence-words are inherently more difficult to predict due to their nature, hence the slower model convergence. Their approach is motivated by human-behavior when trying to remember critical information.

To make sure humans remember critical information they take notes. Once the information is needed they can look it up. For example, assume a model shall predict a missing word given the sentence "COVID-19 has cost many ____". In this case, *COVID-19* represents the rare word. The correct answer would be *lives*. Euros, dollars, puppies and tomatoes are viable options too. If the model can look up in which contexts COVID-19 has occurred previously, it will realize it is closely related to *pandemic* and *global crisis* (assuming it has appeared in this context before). This induces *lives* to be the far better fitting option than *tomato* or *euro*.

Each rare word is initialized into a note dictionary, where the key is the word and the value a semantic vector representation of the word and its context. Once a rare word is encountered during training, the dictionary is accessed and its information is concatenated with the word embedding. Lastly, the dictionary entry of the word is updated according to its current context. The context is a weighted average over words occurring in a context window and the previous vector. It is important to note that once pre-training has finished the dictionary will be discarded. Exact mathematical definitions are omitted and can be found in Wu et al. (2020). The authors report that both BERT and ELECTRA coupled with TNF result in a 60% lesser training time. Figure 2 illustrates the high-level idea.

4.2 Active Learning

Investigating efficient LM pre-training from a data perspective suggests seeking more informing samples. If the training signal is more informing, the model is more likely to converge faster. Some samples may not be as informative as others, some may even yield no informational gain, few likely cause the biggest gradient change.

This suggests to skip samples entirely, even though data may be available at abundance. Active learning (AL) has not seen much play in efficient LM pre-training. Though underlying assumptions (between AL and LM pre-training) vary, interpretations can be superimposed. AL assumes an infinite amount of unlabeled data, where labeling instances is very costly. Instances are selected according to some (informational) metric and labeled by an oracle. For LM pre-training the assumption is similar. Though data is not getting labelled, one can interpret training time using additional samples, which may not be informative, as cost. In the following, I will shed light on promising approaches and try to build a bridge toward efficient pre-training methods (disclaimer: this *bridge* is an idea and has not been examined within the scope of this work).

Active Learning (AL) approaches can be broadly clustered into three categories. **(1)** *Uncertainty Sampling* chooses samples which the model is least confident about, **(2)** *Expected Model Change* chooses samples which yield the largest expected gradient change and **(3)** *Diversity Sampling* tries to cover the data distribution as broadly as possible (Dor et al., 2020).

Least Confidence AL (1): Selects instances the model is least confident about according to the max-entropy or variation-ratio decision rule (Lewis and Gale, 1994).

Deep Bayesian AL (1): Selects instances the model is least confident about according to Monte Carlo Dropout inference cycles paired with max-entropy acquisition function (Gal and Ghahramani, 2016).

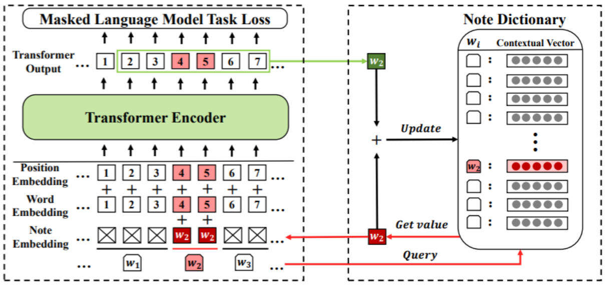

Figure 2: High-level idea of Taking Notes on the Fly (Wu et al., 2020).

Perceptron Ensemble AL (1): Selects instances the model is least confident about according to an ensemble of lightweight models which vote for hardness. Alternatively, inter-model agreement is a viable metric, too (Melville and Mooney, 2004).

Expected Gradient Length (2): Selects instances which the largest expected gradient norm with respect to the posterior distribution of labels (Huang et al., 2016).

Core-set AL (3): Greedily selects diverse instances which best approximate the continuous latent representation space according to the core-set algorithm (Sener and Savarese, 2017). However, this method has been shown to be prone to outliers.

Discriminative AL (DAL, 3): Similar to core-set, DAL selects instances which best cover the representation state, but makes up for the bias towards sparse regions. The procedure follows a simple binary classification task which asks whether a given sample belongs to the set of samples which have already been classified. If not, the sample is likely informative (Gissin and Shalev-Shwartz, 2019).

To date, active learning has not been applied to LM pre-training, likely, because there is no labelling effort involved and almost unlimited data is available. The only restriction is the resource hungry setting. A successful application of the procedure, however, could significantly boost pre-training. For example, skipping only 15% of all samples with a low-overhead selecting procedure would speed up BERT training by one week. While there is a variety of paradigms to choose from, the exact mode of application can be interpreted as a hyperparameter which needs to be found (if the method works at all). One could consider skipping samples for the entire training procedure. More auspiciously, one would select samples interdependently and bound on the current batch or iteration. Intuitively, the more diverse the sample, the less noisy the gradient update. This intuition suggests taking a closer look at diversity sampling.

5 Model View

5.1 Sparsification

Large LMs like BERT are heavily overparametrized leading to vital research in compression algorithms (Jiao et al., 2019; Sanh et al., 2019). However, most known algorithms of this type will only boost inference time. Until recently, it was unclear whether sparse models could be trained to achieve high accuracy, or can only be obtained by compressing an already trained, dense model. In the following, a variety of methods is presented which mainly aim at sparsifying LMs *during* training.

5.1.1 The Lottery Ticket Hypothesis

The *Lottery Ticket Hypothesis* (LTH) states that for a parametrized function $f(x; \theta_0)$ there exists a binary mask m such that the sub-network $f(x; \theta_0 \odot m)$ is as performant (or even better) as the super-network (Frankle et al., 2019). To find such a sub-network, Iterative Magnitude Pruning (IMP) is employed, that is; **(1)** Set $m = \mathbf{1}$, while wished sparsity is not reached **(2)** train $f(x; \theta_0 \odot m)$, **(3)** set $m_i = 0$ if $(\theta_0 \odot m)_i < \tau$ where τ is a number close to 0 indicating small magnitude (Chen et al., 2020). Finally, $\theta_0 \odot m$ is called a *winning ticket*.

In various CV tasks, You et al. (2019a) find that *winning tickets* emerge early in the training procedure if the learning rate is adopted accordingly. Chen et al. (2020) adapt these findings to BERT pre-training utilizing Network Slimming (NS) (Liu et al., 2017). NS relies on the learnable batch normalization scaling parameter γ utilized in convolutional neural networks (CNNs), which is clearly not applicable to transformer networks.

To overcome this, Chen et al. (2020) introduce a parameter c to the attention heads and feed-forward networks (FFN) of BERT. c is regularized by the ℓ_1-norm driving coefficients towards 0. An attention head $h_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ is defined such that $\tilde{h}_i = c_i^h \cdot h_i$. Similarly, FFNs are defined as $FFN(x) = c^f \cdot max(0, xW_1 + b_1)W_2 + b_2$. Since c is just a learnable scaling parameter it indicates the importance of the respective layer. Previous findings suggest that very few attention heads do most of predictive work (Michel et al., 2019; Voita et al., 2019). Thus, we can employ IMP to structurally prune attention head i if $c_i^h < \tau$ until m converges. Once m has converged, we can start pre-training a pruned version of BERT, called *Early-BERT*. It is important to note that *structured sparsity* is crucial for computational efficiency (Wen et al., 2016), hence dropping entire heads. The authors report a comparable performance with 40% lesser training time.

The Lottery Ticket Hypothesis is an active field of research. For example, (Zhou et al., 2019) make a series of interesting observations. First, they claim that masking can be interpreted as training a network. This provides an alternative view onto classical gradient based training procedures. Second, there exist supermasks which can be found with low effort, i.e., no training, and exceed random performance by a large margin. This implies that weight initialization, or mask initialization, is crucial for efficient training. Third, they show that solely the sign of the initial weight has significant impact on its relevance, not its magnitude.

Prasanna et al. (2020) find the following facts. There exist "good" and "bad" subnetworks. The "good" are on par with their supernetwork, however, are not stable across fine-tuning tasks. This leaves room for stabilizing the masking procedure. In fine-tuning, "bad" networks forfeit some performance, but still achieve reasonable accuracy with respect to a strong baseline. This implies that most (if not all) subnetworks are viable instantiations.

5.1.2 Structured Dropout

Alike the prior, in an earlier paper Fan et al. (2019) propose a structured pruning technique called *LayerDrop*. Similar to conventional dropout (Srivastava et al., 2014), *LayerDrop* aims at dropping entire layers of a transformer model. In their work, a variety of pruning strategies is introduced, but simply dropping "every other" layer appears to be the most convenient in terms of simplicity and perplexity. Though *LayerDrop* has been developed to boost inference time, the authors report that this method can be used to efficiently pre-train deep LMs as well. However, they do not report any results or go into detail. Moreover, Fan et al. (2019) claim that pruning *attention heads only* results in the second lowest perplexity while pruning single heads plus entire layers results in lowest perplexity. For the sake of simplicity they vote in favor of pruning single heads only.

5.1.3 Dynamically Skipping Layers

Wang et al. (2019) showcase an interesting approach. Following their dynamic inference idea for CNNs (Wang et al., 2018), they extend their idea to dynamically skipping layers during training. A low-cost recurrent neural network (RNN) between all layers is employed, allowing for an input-dependent selective layer update (SLU). Despite this approach being borrowed from the CV community, it is possibly applicable to transformers. Though their aim is energy efficiency, the idea is possibly extendable to time efficiency as these often correlate. Clearly, applying this idea to Transformers is speculative and needs further, careful examination.

5.2 Parameter Sharing

Sparsification methods try to approximate an unknown hypothesis by only training a structured portion of the available parameters. Parameter sharing approaches, too, try to exploit structural conveniences. The authors of (Gong et al., 2019) find two interesting facts while training BERT. They investigate a shallow and a deep model to frame commonalities and differences. They find that for both models **(1)** attention layers are a mixture of two distributions. One encodes local information and the other focuses on the start-of-sentence token. Most importantly, they note that **(2)** attention weights across both architectures are quite similar and claim **(3)** attention distributions between top layers (e.g., 8, 10, 12) and bottom layers (e.g., 2, 4, 6) are structurally congruent to a large extent. This suggests that parameters can be shared between models or layers. Viewing it as an *intra-transfer learning* approach the authors propose the *progressive stacking* algorithm, that is; **(1)** Initialize BERT with $L/^k$ layers, **(2)** Train BERT, **(3)** repeat k times: stack the current BERT model (i.e. double its size) until L layers are reached. The algorithm is reported to achieve a 25% shorter training while maintaining comparable performance.

5.3 Task Adaption

Most of the work presented here refers to masked language modelling (MLM), of which BERT is an instance of. MLM is typically viewed as a generative task. Given a sentence, a portion of $p\%$ of words is masked. The missing words are to be predicted, or generated, by the MLM. This procedure poses two major issues, that is **(1)** only $p\%$ of a given sample contributes to the learning and **(2)** the generative task at hand is quite complex considering the MLM must find the correct out of $|V|$ words (where V is the vocabulary and typically very large). Consequently, Clark et al. (2020) propose an approach called "**E**fficiently **L**earning an **E**ncoder that **C**lassifies **T**oken **R**eplacements **A**ccurately", short ELECTRA. The model is quite similar to GANs (Goodfellow et al., 2014), though it is not trained adversarially.

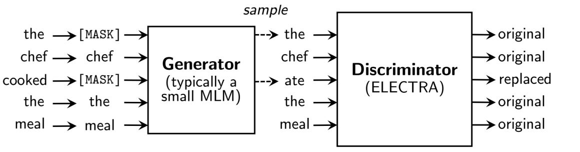

Figure 3: High-level illustration of the ELECTRA architecture (Clark et al., 2020)

A generator (typically a small MLM) and a discriminator (ELECTRA) are trained jointly. The generator receives masked samples (i.e., $p\%$ is masked) and tries to reconstruct the unknown words. Next, ELECTRA receives the output of the generator and tries to discriminate whether a word in the given sample is *original* or has been *replaced*. Figure 3 illustrates the procedure accordingly. There are a few things which are important to note. **(1)** Intuitively, since we train a generator and a discriminator, training should take longer, however, **(2)** task complexity decreases substantially since ELECTRA only needs to choose between 2 instead of $|V|$ options resulting in lesser training time. Lastly, **(3)** the generator is usually a MLM of half of the capacity of its counterpart. Though a model of half of the capacity takes less time to train, it clearly poses some overhead. Finally, the authors report to have achieved comparable results to other MLMs while using 75% less compute power.

Investigating the language modelling objective at hand seems like a promising path. Clark et al. (2020) extend their experiments by interpolating between BERT and ELECTRA. E.g., Instead of inferring *original* or *replaced* on all tokens ELECTRA 15%, just as BERT, only predicts 15%. There are other approaches tested, however, for this I refer to the paper. Essential to note is, the authors of Xu et al. (2020) find that speed-up does not significantly improve when predicting all in contrast to only $p\%$ of the tokens. It is reported that the task complexity at hand is responsible for training PLMs efficiently, yielding a crucial design choice.

5.4 MC-BERT

Clearly, discriminative training is much less informative than generative training. The performance advantage of ELECTRA over BERT is mostly in tasks where syntactic knowledge is more important than semantics. This motivates to design a more informative, but not too complex, pre-training task, which proves to increase performance on semantic tasks. Subsequently, Xu et al. (2020) extend ELECTRA's approach by changing the two-class approach (*original* or *replace*) to a k-class discriminative task. The generator receives a masked sentence. Each masked token is predicted by the generator's best guess. Furthermore, the generator samples k possibilities for every token. For each token, the discriminator decides which of the k possibilities is the correct one. Clearly, the correct word may not be present, thus appending a *"None of the above"* ([NOTA]) token to be selected. The authors do not report a speed-up compared to ELECTRA, since their objective was to design a better discriminative training to enhance semantics tasks.

5.5 Train large, then compress

Typically, deep language models are trained until convergence. The larger the model, the higher the computational cost. This makes large models less viable in resource constrained settings. Li et al. (2020) challenge the status quo. They claim that it is more efficient to train large models, stop early and finally compress to ensure fast inference. The faster convergence rate not only compensates computational overhead but outpaces it. Neither depth nor width explain fast convergence rate well. The number of parameters is the strongest explanatory variable for convergence rate. However, depending on the fine-tune task at hand, adapting width or depth accordingly can be beneficial. In Figure 4 effects of changing hidden size can be observed. The effects of the layer amount acts similarly. One can see that a hidden size of 1024 combined with 24 layers leads to fastest convergence. These results coincide with the initial configuration proposed in the RoBERTa paper (Liu et al., 2019). The key take away is that when pre-training a large LM one needs to find the sweet spot where fast convergence exceeds computational overhead. Even being counter intuitive at first, large models serve best for this purpose.

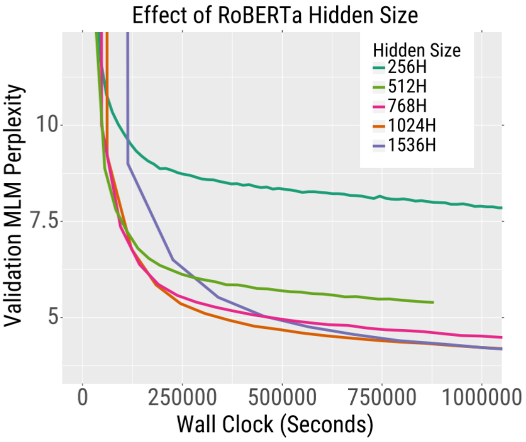

Figure 4: Validation perplexity vs. time as a function of hidden size (Li et al., 2020).

6 Conclusion

In this seminar paper, a variety of efficient pre-training methods for deep language models where explored. This most promising approaches are comprised of the following.

(1) Taking Notes on the Fly A memory-augmentation strategy resulting in a reported training time reduction of 60% (Wu et al., 2020).

(2) The Lottery Ticket Hypothesis and Early-BERT Making use of finding sparse subnetworks early during training, resulting in a reported training time reduction of 40% (Chen et al., 2020; Frankle et al., 2019).

(3) Stacked Parameter Sharing Starting with a shallow network and doubling layers, resulting in a reported training time reduction of 25% (Gong et al., 2019).

(4) ELECTRA Adapting the task objective at hand from a generative to discriminative setting to reduce complexity, resulting in a reported training time reduction of 75% (Clark et al., 2020).

Besides, other methods such as active learning, dropout, structural pruning other data augmentation methods have been explored. Many of these have not found application in efficient pre-training, yet yield promising ideas to be further explored.

Efficient pre-training methods are not as active of a field as I would have hoped. This may lie in the nature of the problem at hand. While methods developed possibly yield a significant speed up, even if training time is halved, BERT pre-training would take one month. This poses major issues, as further exploration requires suitable hardware. Nevertheless, I am hoping that deep language modelling becomes available to a wide range of people eventually.

References

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. 2020. Earlybert: Efficient bert training via early-bird lottery tickets. *arXiv preprint arXiv:2101.00063*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. 2012. Large scale distributed deep networks. *Advances in neural information processing systems*, 25:1223–1231.

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

Liat Ein Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for bert: An empirical study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. *arXiv preprint arXiv:2004.07202*.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. 2019. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

Daniel Gissin and Shai Shalev-Shwartz. 2019. Discriminative active learning. *arXiv preprint arXiv:1907.06347*.

Linyuan Gong, Di He, Zhuohan Li, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Efficient training of bert by progressively stacking. In *International Conference on Machine Learning*, pages 2337–2346. PMLR.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

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

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. 2021. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *arXiv preprint arXiv:2102.00554*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Jiaji Huang, Rewon Child, Vinay Rao, Hairong Liu, Sanjeev Satheesh, and Adam Coates. 2016. Active learning for speech recognition: the power of gradients. *arXiv preprint arXiv:1612.03226*.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898.

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*.

David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers. *arXiv preprint arXiv:2002.11794*.

Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. 2016. Fixed point quantization of deep convolutional networks. In *International conference on machine learning*, pages 2849–2858. PMLR.

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

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744.

Christos Louizos, Karen Ullrich, and Max Welling. 2017. Bayesian compression for deep learning. *arXiv preprint arXiv:1705.08665*.

Prem Melville and Raymond J Mooney. 2004. Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 74.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *arXiv preprint arXiv:1905.10650*.

Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*.

Steven T Piantadosi. 2014. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic bulletin & review*, 21(5):1112–1130.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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*.

Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. 2019. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Pablo Sprechmann, Alexander M Bronstein, and Guillermo Sapiro. 2015. Learning efficient sparse and low rank models. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1821–1833.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. 2018. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424.

Yue Wang, Ziyu Jiang, Xiaohan Chen, Pengfei Xu, Yang Zhao, Yingyan Lin, and Zhangyang Wang. 2019. E2-train: Training state-of-the-art cnns with over 80% energy savings. *arXiv preprint arXiv:1910.13349*.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29:2074–2082.

Qiyu Wu, Chen Xing, Yatao Li, Guolin Ke, Di He, and Tie-Yan Liu. 2020. Taking notes on the fly helps bert pre-training. *arXiv preprint arXiv:2008.01466*.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2019. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. *arXiv preprint arXiv:1912.09637*.

Zhenhui Xu, Linyuan Gong, Guolin Ke, Di He, Shuxin Zheng, Liwei Wang, Jiang Bian, and Tie-Yan Liu. 2020. Mc-bert: Efficient language pre-training via a meta controller. *arXiv preprint arXiv:2006.05744*.

Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard G Baraniuk, Zhangyang Wang, and Yingyan Lin. 2019a. Drawing early-bird tickets: Towards more efficient training of deep networks. *arXiv preprint arXiv:1909.11957*.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2019b. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. 2017. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*.

Shuai Zheng, Haibin Lin, Sheng Zha, and Mu Li. 2020. Accelerated large batch optimization of bert pretraining in 54 minutes. *arXiv preprint arXiv:2006.13484*.

Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. 2019. Deconstructing lottery tickets: Zeros, signs, and the supermask. *arXiv preprint arXiv:1905.01067*.