Exploring Transformers as Compact, Data-efficient Language Models

Clayton Fields
Department of Computer Science
Boise State University
claytonfields@u.
boisestate.edu

Casey Kennington
Department of Computer Science
Boise State University
caseykennington
boisestate.edu

Abstract

Large scale transformer models, trained with massive datasets have become the standard in natural language processing. The huge size of most transformers make research with these models impossible for those with limited computational resources. Additionally, the enormous pretraining data requirements of transformers exclude pretraining them with many smaller datasets that might provide enlightening results. In this study, we show that transformers can be significantly reduced in size, with as few as 5.7 million parameters, and still retain most of their downstream capability. Further we show that transformer models can retain comparable results when trained on human-scale datasets, as few as 5 million words of pretraining data. Overall, the results of our study suggest transformers function well as compact, data efficient language models and that complex model compression methods, such as model distillation are not necessarily superior to pretraining reduced size transformer models from scratch.

1 Introduction

In the space of a few years, transformers have revolutionized natural language processing. Their success has been driven by increasingly large models and more training data. Sizes of the most powerful language models have ballooned to billions of parameters and are pretrained with (in some cases) *trillions* of tokens of text (Hoffmann et al., 2022; Chowdhery et al., 2022). However, the size and data input requirements of transformers limit their reach as research tools in two key ways:

First, training transformers usually requires access to powerful compute resources. For instance, the creators of the PALM model (Chowdhery et al., 2022), used 6,144 TPUv3 chips for pretraining. At the time of this writing, the on-demand cost of this much compute would be a little less than $20,000 *per hour*.[1] Even the moderately sized BERT (Devlin et al., 2018) model required 16 TPU chips for pretraining, putting such a task beyond the meager means of many researchers. Costs this high make research on end-to-end pretraining impossible for potentially timely and impactful academic research (Togelius and Yannakakis, 2023).

Second, large models require pretraining with large datasets that can generally only be obtained from data extracted from the internet. BERT, for instance, was trained on a 3.3 billion word web-based corpus. In contrast, datasets derived from other sources, human speech for instance, are necessarily much smaller and sometimes contain only a few million words. Using data that is not based on internet text can offer insight into how the nature of language data affects language model performance. Currently, such efforts to create language models from smaller, alternative data sources are of growing interest in computational linguistics (Warstadt et al., 2023; Huebner et al., 2021).[2,3]

Most research for creating efficient transformers has focused on *distillation*, which trains a smaller student model using output from a large, pretrained teacher model (Sanh et al., 2019; Wang et al., 2020; Sun et al., 2020; Jiao et al., 2019). While these efforts have produced more efficient models, they require the same large datasets and the use of larger teacher models which themselves require ample compute power during training, even though the end goal is a smaller model. Remarkably, there has to date been little research into simply reducing the size of transformers, pretraining them from scratch and fine-tuning them on downstream tasks. The process of increasing the size of transformer models and their data inputs are well explored (Kaplan et al., 2020; Hoffmann et al., 2022). However, it

[1] https://cloud.google.com/tpu/pricing
[2] https://babylm.github.io
[3] https://sites.google.com/view/learning-with-small-data/home

Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL), pages 521–531
December 6–7, 2023.

is still an open question to what degree the transformer architecture can function as a lightweight, data-friendly research tool.

In this paper, we offer a preliminary study toward addressing these issues. In contrast to previous studies that have approached these topics, we forego the use of knowledge distillation and other complex compression techniques. Rather we pretrain various configurations of the ELECTRA (Clark et al., 2020) transformer in search of parameter and data efficient models. We conduct all of our experiments using a single 12GB GPU to demonstrate the computational efficiency of the models we train. The main contributions of our study are:

- We show that compact transformers can retain a surprising amount of capability on the GLUE benchmark (Wang et al., 2018) when trained with only 5 million word tokens. Further, we show that when training with such a small dataset, several model dimensions can be significantly reduced with little ill-effect.
- We show that when using such a small dataset we can shrink transformers to as few as 5.7 million parmeters and train them faster, using less compute, while retaining much of the performance of much larger models.
- We show that with suitable changes to model configuration, compact variants of the ELECTRA model trained on the moderately sized OpenWebText (Gokaslan and Cohen) corpus can perform on par with compact transformers trained with complex distillation methods such as DistilBERT (Sanh et al., 2019). Further they can do so with significantly fewer parameters and computational requirements.

2 Related Work

The excessive compute requirements of transformers has led to the creation of a sizable body of research into reducing their size and memory footprint. The most well explored strategy is knowledge distillation, a process whereby a full-sized teacher network is used to train a smaller student network. DistilBERT (Sanh et al., 2019), TinyBERT (Wang et al., 2020), MiniLM (Jiao et al., 2019) and MobileBERT (Sun et al., 2020) are popular examples of compact transformers distilled using full sized BERT models as teachers. These methods produce effective smaller models, however they don't directly address the amount of input data required and the training process still requires using a full-sized teacher model to train the student model.

Pruning is another popular model compression method in which some fraction of the trained model's parameters are set to zero. Li et al. (2020) and Sanh et al. (2020) use unstructured pruning methods to eliminate a large percentage of weights throughout transformer models with small corresponding reductions in performance. Structured pruning methods such as Fan et al. (2019) set the parameter values of entire regions of the model to zero; in this case whole transformer layers are pruned. Michel et al. (2019) showed that a large percentage of BERT's attention heads can be entirely removed before testing without a significant decrease in performance. However, these techniques are are all premised on pretraining full-sized models and then reducing the model size prior to inference time, therefore still have the same pretraining data and compute requirements.

There has also been some research directly addressing the size of pretraining datasets for transformers. Micheli et al. (2020) and Martin et al. (2019) experimented with reducing the absolute amount of training data in French language models. They showed that full sized French language transformer models can perform well on select tasks with significantly less pretraining data. Warstadt et al. (2020b) and Zhang et al. (2020) investigated the effect of different pretraining data volumes on the grammatical knowledge of the RoBERTa-base model using probing techniques.

Huebner et al. (2021) experimented with using AOCHILDES, the 5 million word dataset composed of child directed speech for pretraining and evaluated their results using a grammatical benchmark based on BLIMP (Warstadt et al., 2020a). This study is notable because the authors used a very small pretraining dataset derived from human speech and opted to use a scaled-down version of the RoBERTa model (Liu et al., 2019) to accommodate it. Unfortunately, the resulting model was only tested on narrow set of grammatical learning tasks, using a specialized dataset for evaluation.

3 Data and Evaluation Criteria

3.1 Pretraining Data

The ELECTRA model was originally pretrained with the 3.3 billion word corpus used to train BERT (Devlin et al., 2018). This dataset, however, is not

Model	Params	COLA	MRPC	QNLI	MNLI	QQP	SST2	STSB	RTE	Avg.
ELECTRA	13.6M	**0.570**	**0.907**	0.883	0.814	**0.894**	0.858	0.822	**0.657**	**0.801**
MobileBERT	15.1M	0.531	0.881	**0.908**	0.814	0.858	**0.901**	**0.874**	0.592	0.794
DistillBERT	67M	0.496	0.869	0.886	**0.824**	0.866	**0.901**	0.864	0.585	0.783

Table 1: Results for downstream tasks with compact, pretrained models downloaded from the Huggingface library.

publicly available. Fortunately, there are a variety of open source alternatives freely available for research purposes. We use a web-sourced, public text corpus, or a subset of it, called OpenWebText (Gokaslan and Cohen) for pretraining in all of our experiments. The OpenWebText corpus was created as a publicly available reproduction of OpenAI's WebText corpus that was used in the training of GPT-2. It consists of over 38GB of text data scraped from over 8 million internet documents. It is a popular choice for pretraining language models. We chose this dataset, specifically because it contains text from a wide variety of sources and will prepare our models for the diverse range of tasks contained in the GLUE benchmark (Wang et al., 2018).

In the first two of our three experiments we aim to test models trained with scarce data, specifically we use approximately 5 million words of pretraining data. 5 million words is a rough estimate of how many words an American child might hear before they begin speaking (Gilkerson et al., 2017). In that sense it represents a realistic size for a human scale dataset. To obtain a corpus of suitable size for this experiment we randomly selected documents from OpenWebText until we had a set with just over 5 million words and 306,462 unique words including names of websites such as "tumblr" and non-English words and phrases. In terms of disk space it requires only 43MB to store. In our third and final experiment we make use of all 38GB of the OpenwebText corpus. The scale and diversity of the full dataset are similar to those used to train models such as BERT and will allow us to compare our compact model variations to other pretrained compact models.

3.2 Finetuning Data & Tasks

GLUE To evaluate our pretrained models we fine-tune them on the GLUE tasks introduced in Wang et al. (2018). The GLUE benchmark consists of nine supervised sentence-level tasks and their associated datasets that cover a variety of natural language understanding domains. We chose GLUE as a benchmark because it spans several tasks and

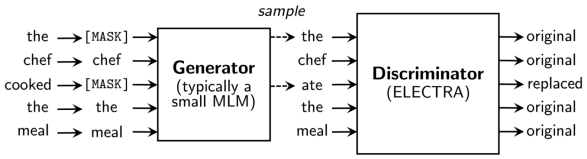

Figure 1: The ELECTRA model is a Generator-Discriminator ensemble. The Discriminator is tasked with determining if the Generator properly guessed a masked word; borrowed from (Clark et al., 2020).

because its popularity in NLP research allows us to directly compare the performance of our models with previously published results. Following Devlin et al. (2018) and Clark et al. (2020) we exclude the WNLI task from our consideration. COLA is a grammatical acceptability task, SST-2 a sentiment classification, QQP, MRPC, STS-B are sentence similarity tasks and MNLI, QNLI, and RTE are inference tasks. Our evaluation metrics are Spearman correlation for STS-B, Matthews correlation for CoLA, F1 score for QQP and MRPC and accuracy for the remainning tasks. All of the reported results were obtained by evaluating on the dev sets of the tasks described and are fine-tuned for 10 epochs. In general, the standard practice for GLUE fine-tuning is to train for 3 epochs with a batch size of 32 and a learning rate of 2e-5. However, Clark et al. (2020) noted that ELECTRA performs better on select GLUE tasks when trained for 10 epochs. We found that since overfitting is not a concern for the small variants of ELECTRA that we trained, our models benefited from training for 10 epochs on all of the GLUE tasks.

4 Language Model: ELECTRA-small

In this section we describe the ELECTRA-small model (Clark et al., 2020) and the rationale behind using it as the basis for our experiments. In place of masked language modeling, ELECTRA pretrains a transformer encoder stack, structurally identical to BERT's, by replacing some input tokens with plausible alternative words sampled from a small generator network. A larger discriminator model then predicts whether or not each input token has

intr size	emb size	lyrs	prms	COLA	MRPC	QNLI	MNLI	QQP	SST2	STSB	RTE	Avg.
1024	128	12	13.6M	0.417	0.825	0.818	0.755	0.836	0.838	0.802	0.596	0.736
768	128	12	12.0M	0.422	0.841	0.823	0.755	0.838	0.849	0.800	0.556	0.736
512	128	12	10.4M	0.425	0.832	0.822	0.757	0.838	0.807	0.800	0.570	0.731
256	128	12	8.82M	0.343	0.833	**0.828**	**0.758**	0.838	0.830	0.794	0.588	0.727
128	128	12	8.03M	0.379	**0.861**	0.819	0.750	0.839	0.825	**0.815**	0.592	0.735
64	128	12	7.64M	0.366	0.852	0.816	0.753	0.838	0.819	0.813	**0.639**	0.737
1024	**96**	12	12.5M	0.362	0.818	0.821	0.752	**0.840**	0.823	0.807	0.588	0.726
1024	**64**	12	11.5M	0.346	0.824	0.820	0.746	0.830	0.820	0.784	0.552	0.715
1024	**32**	12	10.5M	0.246	0.822	0.798	0.725	0.816	0.831	0.757	0.563	0.695
1024	128	**10**	12.0M	0.415	0.834	0.827	0.746	**0.840**	0.844	0.807	0.599	0.739
1024	128	**8**	10.4M	**0.442**	0.851	0.826	0.748	0.839	0.826	0.806	0.585	**0.740**
1024	128	**6**	8.81M	0.367	0.814	0.826	0.746	0.835	**0.852**	0.787	0.516	0.718
1024	128	**4**	7.2M	0.28	0.823	0.819	0.740	0.832	0.818	0.791	0.581	0.710

Table 2: Results for downstream tasks with reduced model dimensions. Note that the top row represents the full-sized ELECTRA-small model. All results were trained with a 5M word subset of openwebtext trained for 100,000 steps with batch size of 128. Reduced parameter settings are shown in bold.

been replaced. See Figure 1 for an illustration of the ELECTRA model. Clark et al. (2020) show that this strategy leads to better results with less data and less compute than causal language modeling or standard masked language modeling. After training, the generator is discarded and the discriminator is used for downstream tasks.

The ELECTRA-small model also has the advantage of beginning with only 13.6 million parameters and performs favorably to similarly sized models. Further, it can be pretrained without the use of model distillation. As the purpose of our study is to train transformers in data and resource scarce settings, it is desirable to use a model that doesn't require a teacher model pretrained on a massive text corpus, and can be trained on a single GPU.

To ensure that ELECTRA-small can produce results on par with other compact models we use the pretrained models from the Huggingface transformer hub and test them on our selected downstream tasks.[4] The results are summarized in Table 1. We tested three pretrained models, ELECTRA, MobileBERT and DistilBERT. Of the three, ELECTRA is the smallest model in terms of absolute number of parameters with only 13.6 million. Despite its small size, it achieves the best average results on GLUE. Notably it does so using only pretraining and fine-tuning without the benefit of knowledge distillation from a larger model. These features harmonize well with the goals of our study and make the ELECTRA-small model the logical choice on which to base our succeeding experiments.

[4]https://huggingface.co/

5 Experiment 1: Reducing Individual Model Dimensions in a Low Data Setting

In the first set of experiments that we conduct, we test varying the size and configuration of the ELECTRA model using the 5 million word subset of openwebtext described in Section 3.1 as the pretraining dataset for each model variation. We begin by changing only a single dimension of the model's configuration. The goal of this series of experiments is to determine which parts of the model's architecture can be reduced and what effect these reductions have on performance. In the process, we hope to provide some insight into how the size of each dimension of the transformer model contributes to its downstream performance.

5.1 Procedure

The basic architecture of transformer models is best described by Vaswani et al. (2017) and consists of an embedding layer followed by stacked attention layers, each composed of a multi-head attention mechanism followed by a feed-forward neural network sub-layer. All of the stacked layers have the same dimension, but have varying weights. The *embedding size*, *vocab size*, *hidden state size*, the feed-forward network's *intermediate size* and the absolute *number of layers* can all be altered. The number of attention heads per layer and the maximum sequence length can also be varied, though these changes don't affect the overall number of model parameters. We first test the effect of reducing the size of each of the these parameters and then pretrain a given model on our 5 million word

subset of openwebtext. Each model is trained for 100,000 steps with a batch size of 128 and a learning rate of 5e-4. The generator network used for training is one quarter the size of the discriminator network.

The downstream tasks on which we fine-tune and evaluate our resulting models are the GLUE tasks described in Section 3, including an *Average* of all scores (**Avg.**). For each task we fine-tune models for 10 epochs, with a learning rate of 2e-5 and a batch size of 32. Four of these tasks, QQP, QNLI, SST-2 and MNLI are associated with relatively large datasets and the results are fairly robust to changes in the model's size. The remainder of the GLUE tasks on the other hand, use very small training sets leading to wide variation, even scores at the level of chance when model capacity is sufficiently degraded.

5.2 Results

The results for the models with a single dimension reduced are discussed in this section and summarized in Table 2. We had the most success in reducing the intermediate size, the embedding size and the number of attention layers and we provide discussion for each below. The results of modifying the hidden size, vocabulary size and number of attention heads were less successful and are available in Appendix A.

Intermediate Size The intermediate size refers to the dimension of the hidden layer in the feed-forward network (FFN) contained in each attention layer. Following Vaswani et al. (2017), ELECTRA's default intermediate size is 4 times that of the the hidden size, which yields an intermediate size of 1024 for ELECTRA-Small. Our results indicate that the number of these parameters can be dramatically decreased with relatively little effect on the model's capability when training with a small dataset. Downstream performance shows essentially no loss with as few as 64 parameters in each FFNs hidden layer, nearly a 16-fold reduction in size. This is a remarkable result as the model performs nearly identically with 6 million fewer parameters. Most transformer architectures also use an intermediate size 4 times that of the hidden size of the attention layers. These results suggest that the intermediate stage of transformer's FFNs may be over-parameterized. In the final experiment we address how well these results hold for models trained on large-scale datasets.

Embedding Size In transformer models, the embedding size refers to the length of each vocabulary word's embedding vector. The default size of the embedding vectors for ELECTRA is 128. Like the intermediate layer size, the embedding size can be substantially decreased while retaining most of the model's downstream performance. We see from our results that an embedding size of 96 has virtually the same capability as a full-sized model. An embedding size of 64 shows slightly reduced performance on most of the GLUE tasks with a 2 percent drop in average score. This is a notable result and it suggests that the embedding layer may also be over parameterized in a low data setting.

Model Depth Finally, we also reduce the depth of the model, and its number of parameters by simply decreasing the number of attention layers in the model. The number of hidden layers in ELECTRA-small is 12 by default. The results for reducing model depth are included in Table 2. We see that decreasing the number of layers to 10 or 8 actually improves the model's performance on most of our downstream tasks with a 1.5 million and 2.3 million decrease in their respective parameter counts. Further decreasing the number of layers to 6 or 4, with 3-5 million fewer parameters, shows only small decreases in the overall GLUE score.

6 Experiment 2: Reducing Overall Model Size in a Low Data Setting

Guided by our results from the previous experiments, we now aim to find an overall configuration of ELECTRA-small that has significantly fewer parameters than its default of 13.6 million and retains most of its downstream performance. Using the same 5 million word dataset, we train models with any number of their dimensions reduced or modified. In essence, we trained and evaluated a large number of models with various combinations of our most successful modifications from our previous experiments in search of a robust and well functioning model.

We found early in our efforts that simply reducing model size while keeping the dimensions proportionate produced poor results. Our results from the previous section suggest that this may be due to the models low tolerance for decreases in its hidden size. We did however find several alternative configurations with parameter counts that ranged from 5.7 to 10 million that retained most of the performance of the full-sized ELECTRA-small

Model	Hidden Size	Inter Size	Layers	Emb Size	Params	time 100k	time 1M
ELECTRA-small	256	1024	12	128	13.7M	16h26m	6d21h
Model 1	256	1024	**8**	128	10.4M	11h15m	4d17h
Model 2	256	256	**16**	**64**	8.4M	16h11m	6d5h
Model 3	256	**128**	**14**	**96**	7.7M	13h	5d18h
Model 4	256	**64**	12	128	7.6M	11h58m	5d7h
Model 5	**196**	**128**	**18**	**64**	5.7M	13h48m	5d17h

Table 3: **Model Key** Dimensions for 5 smaller model configurations of ELECTRA. Training times for 100k and 1M training steps with a batch size of 128 on a 12GB GPU included. Note that the top row represents the full-sized ELECTRA-small model for reference. Reduced parameter settings are shown in bold.

Model	COLA	MRPC	QNLI	MNLI	QQP	SST2	STSB	RTE	Avg.
ELECTRA-Small	0.417	0.825	0.818	**0.755**	0.836	0.838	0.802	0.596	0.736
Model 1	**0.442**	0.851	0.826	0.748	0.839	0.826	0.806	0.585	**0.740**
Model 2	0.383	0.834	**0.833**	0.752	**0.841**	0.826	0.815	0.614	0.737
Model 3	0.366	**0.852**	0.816	0.753	0.838	0.819	0.813	**0.639**	0.737
Model 4	0.386	0.849	0.832	0.751	0.839	**0.842**	**0.817**	0.567	0.735
Model 5	0.334	0.838	0.819	0.736	0.827	0.815	0.794	0.614	0.722

Table 4: **Low Data Setting** Results for select models trained with the 5M word subset of OpenWebText corpus for 100k steps. Results for MobilBERT and DistilBERT are appended for the sake of comparison.

model trained on our 5 million word set. The most successful configurations modified some combination of intermediate size, embedding size or layer count. We discovered that we could improve performance relative to parameter count by decreasing the width (*intermediate size* and *hidden size*) of the model while increasing its depth (*number of layers*). We had less success increasing the width and decreasing the number of layers. Turc et al. (2019) observed a similar phenomenon leveraging knowledge distillation on pretrained compact models.

Though we trained several dozen model variations, we present only our 5 most succesful. The dimensions and parameter counts of these models are described in Table 3. Two of the models, Model 1 and Model 4, feature only a single modified dimension and were mentioned in the previous experiment. Model 1 has 8 layers and Model 4 has an intermediate size of only 64 parameters. These modifications led to good results in our previous experimental settings and a sizeable reduction in model size. The remaining models feature a decrease in the model width and the size of the embedding layer with an increase in model depth.

Procedure As in our previous experiments where we altered only a single model dimension, we pretrain all of our models using the 5 million word subset of openwebtext for 100,000 steps. We evaluate our models using the same metrics and hyperparameters as the previous experiment in order to compare our results.

Results The downstream results for these models are summarized in Table 4 and discussion is provided below. In this low data setting, our small Models 1-4 have essentially the same performance as the original ELECTRA-Small model configuration trained with the same data and settings. Model 5 performed only slightly worse, despite having only 5.7 million parameters. These results suggest that when using small datasets, small-scale transformers may perform as well as their computationally more expensive larger cousins.

Moreover, the reduction in size can be performed in a variety of ways. The results for Model 1 show that we can also decrease the model depth by 4 layers without ill-effect. Doing so cuts our training time nearly in half and reduces our model size by 3 million parameters. Alternatively, increasing the depth to compensate for loss of width and embedding size was also very effective in lowering overall model size. Models 2, 3 and 5 made use of this strategy to varying degrees and produced similar results. Increasing model depth, however, comes

Model	COLA	MRPC	QNLI	MNLI	QQP	SST2	STSB	RTE	Avg.
ELECTRA-Small	**0.591**	**0.908**	0.875	0.812	0.856	0.885	0.857	0.632	**0.802**
Model 1	0.487	0.886	0.865	0.788	0.847	0.894	0.842	0.61	0.777
Model 2	0.504	0.896	0.859	0.784	0.848	0.853	0.841	0.621	0.776
Model 3	0.478	0.881	0.854	0.792	0.847	0.885	0.842	0.632	0.776
Model 4	0.409	0.868	0.846	0.774	0.836	0.858	0.849	0.643	0.760
Model 5	0.444	0.906	0.860	0.784	0.846	0.859	0.834	**0.661**	0.774
MobileBERT	0.510	0.880	**0.908**	**0.831**	**0.873**	**0.917**	**0.874**	0.625	**0.802**
DistilBERT	0.527	0.826	0.889	0.818	0.870	0.896	0.865	0.585	0.785

Table 5: **High Data Setting** Results for select models trained with the full OpenWebText corpus for 1 million steps. Results for MobilBERT and DistilBERT are appended for the sake of comparison.

at the cost of slower training times, presumably because of the increased number of non-linear activation functions. Though smaller, Models 2, 3 and 5 required longer train times than did models 1 and 4, which had fewer layers. Model 2 required nearly as much time to train than ELECTRA-small.

7 Experiment 3: Reducing Model Size in a High Data Setting

In this experiment, we pretrain a selection of models using the full OpenWebText corpus as the pretraing dataset and training for a million steps. We use the same five models described in Table 3. Because the training times in this experiment are much longer, we will not repeat the exhaustive study of the effects of changing individual model dimensions as we did in the low data setting. Rather, we only pretrain and evaluate the 5 models considered in Experiment 2. The goal of this experiment is determine to what degree the results of Experiments 1 and 2 will hold with a full-size dataset trained for an extended time. Given that the models considered contain so few parameters, it is a natural question as to whether or not they can adequately make use of the additional information provided by more data and longer pretraining. The results of this experiment will also be more readily compared to other compact transformers which are also trained on full-sized datasets. We use the same evaluation criteria as that performed in Experiments 1 and 2.

Results The results of fine-tuning these models on the GLUE corpus are summarized in Table 5 and discussion is provided below. As opposed to the scarce data setting, the larger ELECTRA-Small model is able to make greater use of more data and increased training time and outperforms its smaller counterparts to a noticeable degree. This was an expected result given the abundance of training data used. Over the course of a million training steps, the differences in training times are considerable. Model 1 requires almost 2 days fewer to train. Models 3, 4 and 5 all require a day less in training time. The slow training of Model 2 is again on display, requiring over six days of training time.

Of the small models we tested, all had quite similar performance, though Model 4 showed a slight drop relative to the other models. In our high data setting, with longer training time, our smallest model, Model 5 performs as well as the other small model variants. This is a change from the low data setting where it lagged slightly behind. Models 2 and 3 also perform well in this setting suggesting that increasing model depth to offset reductions in other dimensions scales fairly well to larger datasets. Notably Model 1, with 8 layers of the original ELECTRA-small dimensions, had similar performance and a favorable training time. Though it contains more parameters than the other small models, its reduced depth markedly reduces training time, requiring less than 5 days to train for a million steps.

It is not immediately clear why these particular distributions of parameters perform well. Most transformer architectures feature a roughly 2:1 ratio of parameters between their feed-forward networks and their multi-head attention mechanisms. Our results suggest that this ratio might be open to significant modification. The theoretical purpose of the FFN is to introduce non-linearity. The fact that increasing the number of layers, and therefore the number of non-linear activation functions, seems to offset reductions in the size of the FFN lends credence to that theory. MobileBERT also has a long, thin architecture. Its creators, however, felt com-

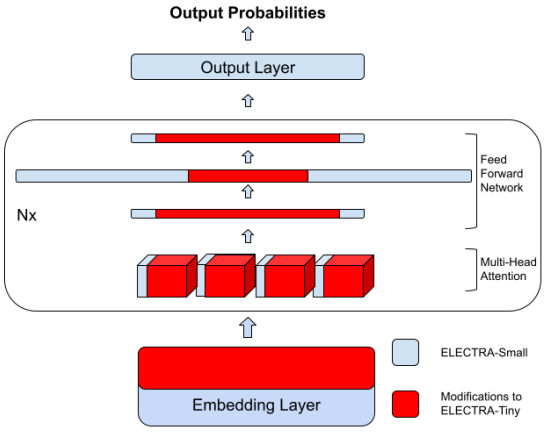

Figure 2: Relative size comparison of Electra-small (blue) with Electra-tiny (red). Electra-tiny has smaller embeddings, hidden size, and intermediate size, but has more hidden layers.

pelled to stack additional FFNs to restore the 2:1 parameter ratio. We suggest that this may not be necessary. In general, we also advocate for a more thorough investigation of how parameters are distributed within the transformer architecture. While the focus of this study was in low data settings and small models, even small improvements in parameter efficiency could be of great consequence for very large models.

7.1 Our Smallest Model: ELECTRA-tiny

The smallest model configuration we found that didn't experience large reductions in performance was Model 5. It had a hidden size of 196, reduced from 256, intermediate size of 128, decreased from 1024, an embedding size of 64, decreased from 128 and 18 layers, increased from 12. We call this model ELECTRA-Tiny and it contains just 5.7 million parameters. Figure 2 shows visually how ELECTRA-Tiny compares to ELECTRA-Small. ELECTRA-Tiny is an extremely small given the model's performance on a diverse set of tasks such as GLUE. When training the ELECTRA-Small model, the largest batch size that our 12GB GPU could accommodate was 128. Because of the small size of ELECTRA-Tiny, we could train at batch sizes of up to 256; alternatively we might have trained ELECTRA-Tiny at a batch size 128 on an even smaller GPU. The compactness, low compute requirements and favorable training times make a model like this ideal for researchers without access to multiple GPUs. The model weights from Experiment 2, trained with the full OpenWebText for 1 million steps, are available on the Huggingface.[5]

For the sake of comparison, we have added the results of two compact transfomers trained with distillation to Table 5, DistilBERT (Sanh et al., 2019) and MobileBERT (Sun et al., 2020). We again downloaded the pretrained weights for these models from Huggingface. This time however, we finetuned the models for 10 epochs and the same fine-tuning parameters as the previous experiments in order to fairly compare the results to the compact ELECTRA variants we trained. Though differences in training data and training times make this comparison somewhat inexact, the results are still illuminating. We see that ELECTRA-Tiny produces scores only slightly below that of the DistilBERT model, despite being a tenth of the size and being trained without complex distillation losses. MobileBERT performs slightly better, on par with the ELECTRA-Small model. MobileBERT has 15 million parameters, slightly more than ELECTRA-Small and 3 times as many as ELECTRA-Tiny. All told, our data suggest that complex compression techniques like distillation may be less profitable than simply starting with much smaller models and pretraining them on a suitable training corpus with a data efficient proxy task such as the discriminative loss of ELECTRA.

8 Conclusion

In this study, we have shown that the transformers, specifically ELECTRA, can function as compact data-efficient models. Our results suggest that when training with small datasets, the intermediate size, embedding size and number of layers can all be reduced with little ill-effect. Additionally, we presented the GLUE results for 5 model variations that significantly reduce the overall size of the ELECTRA-Small model. In the final phase of our experiments, we tested the same five models trained with the full OpenWebText corpus. We showed that several compact transformer architectures can function on par with larger models trained using complex distillation methods. Finally, we present a compact configuration of ELECTRA we call ELECTRA-Tiny with just 5.7 million parameters that performs remarkably well on the GLUE benchmark given its small size, requires less compute and can be trained end to end on a single 12GB GPU.

[5] https://huggingface.co/claytonfields/electra-tiny/tree/main

Limitations

One of the primary limitations of our study was that of computational resources. Had we had more compute, we would to have been able to conduct more exhaustive studies of our models in high data scenarios with extended training times. There are several model compression methods such as quantization (Bondarenko et al., 2022) and adaptive sequence length reduction (Guskin et al., 2021) that would have been compatible with the models that we trained. An exhaustive study of these techniques applied to the type of small models we used in this study could potentially have produced even more efficient models.

Ethics Statement

There are no compelling ethical conflicts to report for this study.

Acknowledgements

Thanks to the anonymous reviewers for their very useful feedback. This material is based upon work supported by the National Science Foundation under Grant No. 2140642.

References

Alexander Bondarenko, Magdalena Wolska, Stefan Heindorf, Lukas Blübaum, Axel-Cyrille Ngonga Ngomo, Benno Stein, Pavel Braslavski, Matthias Hagen, and Martin Potthast. 2022. CausalQA: A benchmark for causal question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3296–3308, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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*.

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*.

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

Jill Gilkerson, Jeffrey A Richards, Steven F Warren, Judith K Montgomery, Charles R Greenwood, D Kimbrough Oller, John HL Hansen, and Terrance D Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American journal of speech-language pathology*, 26(2):248–265.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus.

Shira Guskin, Moshe Wasserblat, Ke Ding, and Gyuwan Kim. 2021. Dynamic-TinyBERT: Boost TinyBERT's inference efficiency by dynamic sequence length.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training Compute-Optimal large language models.

Philip A Huebner, Elior Sulem, Fisher Cynthia, and Dan Roth. 2021. Babyberta: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 624–646.

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*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

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*.

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*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot. 2019. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Vincent Micheli, Martin d'Hoffschmidt, and François Fleuret. 2020. On the importance of pre-training data

volume for compact language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7853–7858, Online. 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*.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389.

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

Julian Togelius and Georgios N Yannakakis. 2023. Choose your weapon: Survival strategies for depressed ai academics. *arXiv preprint arXiv:2304.06035*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Alex Warstadt, Leshem Choshen, Aaron Mueller, Adina Williams, Ethan Wilcox, and Chengxu Zhuang. 2023. Call for papers – the BabyLM challenge: Sample-efficient pretraining on a developmentally plausible corpus.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020a. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Yian Zhang, Haau-Sing Li, Haokun Liu, and Samuel R Bowman. 2020b. Learning which features matter: Roberta acquires a preference for linguistic generalizations (eventually). *arXiv preprint arXiv:2010.05358*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yian Zhang, Alex Warstadt, Haau-Sing Li, and Samuel R Bowman. 2020. When do you need billions of words of pretraining data? *arXiv preprint arXiv:2011.04946*.

A Additional Results

A.1 Experiment 2: Hidden Size

We found that small reductions in hidden size result in significantly fewer model parameters and notable effects on downstream performance. Lowering the hidden size from 256 to 192 results in tolerable losses in performance, even on our low data tasks COLA and BLiMP. However, further reductions show sizable drops in downstream performance, especially for COLA and BLiMP. As was mentioned in section 6.2, the effect of decreasing hidden size can be offset by increasing mode depth.

A.2 Experiment 2: Vocabulary Size

Altering the vocabulary size is somewhat more involved than changing the other dimensions. The vocab is produced by the WordPiece algorithm (Wu et al., 2016) and must be trained on a corpus of text. The number of words in the vocab is chosen prior to training and the algorithm determines the optimum choice of word pieces. In order to form a fair comparison with the original vocabulary we elected to train various tokenizers on a large fraction of the openwebtext data. In contrast to embedding size, we see significant effect from lowering the vocab size relative to the decrease in parameter count. As such, decreased vocabulary size did not figure into our most effective reduced model configurations.

A.3 Experiment 2: Attention Heads

Finally, we tried altering the number of attention heads per layer from the defualt number of 4. Since the number of attention heads does not affect the number of parameters in the model, we also tried increasing the number to 8 (the number of attention heads must evenly divide the attention layer hidden size). Our results show that doing so did not greatly impact model performance.

hid size	voc size	atn hds	Prms	COLA	MRPC	QNLI	MNLI	QQP	SST2	STSB	RTE	Avg.
256	30,522	4	13.6M	0.417	0.825	0.818	0.755	0.836	0.838	0.802	0.596	0.736
192	30,522	4	10.6M	0.369	0.824	0.824	0.752	0.839	0.833	0.789	0.567	0.725
128	30,522	4	7.9M	0.284	0.828	0.824	0.738	0.826	0.815	0.716	0.534	0.696
64	30,522	4	5.8M	0.176	0.815	0.773	0.696	0.79	0.803	-0.107	0.505	0.556
32	30,522	4	4.8M	0.0	0.812	0.657	0.655	0.753	0.763	-0.139	0.52	0.503
256	**28,672**	4	13.3M	0.339	0.841	0.811	0.74	0.832	0.807	0.789	0.585	0.718
256	**24,576**	4	12.8M	0.275	0.838	0.818	0.745	0.836	0.813	0.765	0.552	0.705
256	**20,480**	4	12.3M	0.294	0.842	0.813	0.744	0.837	0.828	0.794	0.599	0.719
256	**16,384**	4	11.7M	0.33	0.821	0.821	0.742	0.837	0.821	0.779	0.578	0.716
256	**12,288**	4	11.2M	0.335	0.818	0.824	0.74	0.839	0.798	0.809	0.534	0.712
256	**8,192**	4	10.7M	0.279	0.844	0.819	0.735	0.84	0.817	0.807	0.545	0.711
256	30,522	**8**	13.5M	0.381	0.828	0.824	0.749	0.842	0.831	0.765	0.552	0.722
256	30,522	**2**	13.5M	0.385	0.848	0.815	0.752	0.839	0.844	0.801	0.581	0.733
256	30,522	**1**	13.5M	0.401	0.803	0.819	0.746	0.838	0.838	0.788	0.574	0.726

Table 6: Additional results for downstream tasks with reduced model dimensions. Note that the top row represents the full-sized ELECTRA-small model. All results were trained with a 5M word subset of openwebtext trained for 100,000 steps with batch size of 128. Modified parameter settings are shown in bold.