GENERAL BENEFITS OF MONO-LINGUAL PRE-TRAINING IN TRANSFORMERS

Jiacheng Zhang

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

UNIVERSITY OF ARIZONA

2021

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Master's Committee, we certify that we have read the thesis prepared by: Jiacheng Zhang
titled:

and recommend that it be accepted as fulfilling the thesis requirement for the Master's Degree.

Steven Bethard
Steven Bethard Date: Apr 23, 2021

Mihai Surdeanu
Mihai Surdeanu Date: Apr 23, 2021

J Barnard
Jacobus Barnard Date: Apr 23, 2021

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to the Graduate College.

I hereby certify that I have read this thesis prepared under my direction and recommend that it be accepted as fulfilling the Master's requirement.

Steven Bethard
Steven Bethard Date: Apr 23, 2021
Thesis Committee Chair
School of Information / Computer Science

Table of Contents

Abstract

Pre-trained transformer is a class of neural networks behind many recent natural language processing systems. Its success is often attributed to linguistic knowledge injected during the pre-training process. In this work, we make multiple attempts to surgically remove language specific knowledge from BERT. Surprisingly, these interventions often do little damage to BERT's performance on GLUE tasks. By contrasting against non-pre-trained transformers with oracle initialization, we argue that when it comes to explain BERT's working, there is a sizable void below linguistic probing and above model initialization.

1 Introduction

Pre-trained transformer (Devlin et al., 2019; Liu et al., 2019; Raffel et al., 2020; Rogers et al., 2020; Wolf et al., 2020) is a class of neural networks used in many recent natural language processing systems. Transformers breaks the input text segment into word-pieces and maps each of these word-pieces into an embedding vector.

The current adaptations of this model use a pre-train-then-fine-tune approach (Devlin et al., 2019). In the pre-training step, the model is trained on unlabeled data tasks like predicting a masked word from its neighbors. And fine-tuning combines the pre-trained model with a task specific output layer and trains them together on the task of interest. This scheme alone is often sufficient to outperform non-transformer-based systems in prediction quality, and it can be taken further with careful engineering. A key insight of these efforts is that the more relevant the unlabeled training data is to the supervised task, the more helpful the pre-training is expected to be (Gururangan et al., 2020; Lee et al., 2019; Elwany et al., 2019).

In this pre-train-then-fine-tune approach, it is often taken for granted that we should use model pre-trained on the target language. This view is reinforced by the discovery that pre-trained models exhibit different linguistic sensitivity in different components (Tenney et al., 2019; Hewitt and Manning, 2019; Clark et al., 2019).

In this project, we also try to find what transformers learnt during pre-training. However instead of extracting information from model hidden states, we study transformers' response to different interventions.

Our findings are:

1. A significant proportion of performance gain from pre-training remains even with all transformer layers put in reversed order. This is unexpected under the conventional understanding that pre-trained transformer layers forms a pipeline where each layer takes a specialized role.

2. It is easier to learn an English model from a transformer pre-trained on a foreign language than it is to learn an English model from scratch, even when the languages have minimal overlap in word-pieces, e.g., English and Chinese. For some languages and tasks, fine-tuning from a foreign language works as well as fine-tuning from English. This suggests that pre-training can yield benefits that transcend the language used.
3. It is easier to learn from randomly shuffled word-piece embeddings, or from English word-piece embeddings injected into a foreign language model, than it is to learn from scratch. This suggests that transformers find it easier to learn the alignments between word-pieces and existing embeddings than to learn the embeddings themselves. This is consistent with the hypothesis that monolingual models learn embeddings that can be easily projected to other languages (Conneau et al., 2020).
4. It is easier to learn from a foreign-language model than from either of the model modifications explored in item 3. This suggests that transformers learn best from coherent models where all components were learned simultaneously.
5. A significant performance gain is possible when each parameter is randomly initialized with an oracle distribution.
6. It is possible for a pre-trained transformer to discard its last few layers and still perform competitively against the original. This provides an alternative to distilled models (Sanh et al., 2019; Jiao et al., 2020), which are expensive to produce.

The rest of this paper will be structured as following: Section 2 reviews previous works on transformer pre-training. Section 3 motivates the need to examine pre-trained transformers with methods that are beyond linguistic probing. Section 4 explains our core observations and section 5 contains some attempts to rationalize such observations. And finally section 6 is a simple exploit of our discovery in engineering.

2 Literature Review

The understanding of pre-training's effect on transformers is important for refinement of such process. In this section, we define transformer networks and review recent works that are relevant to this end.

2.1 Transformers, Pre-training, Fine-tuning

The core of transformers is self-attention (Vaswani et al., 2017), which can be written as $SOFTMAX(QK^T)V$, where Q, K, V are learnt linear projections of incoming word embedding vectors. Multiple instances of this self-attention mechanism are used in parallel in each transformer layer, and their outputs are concatenated into a single embedding vector. The complete structure of a transformer is

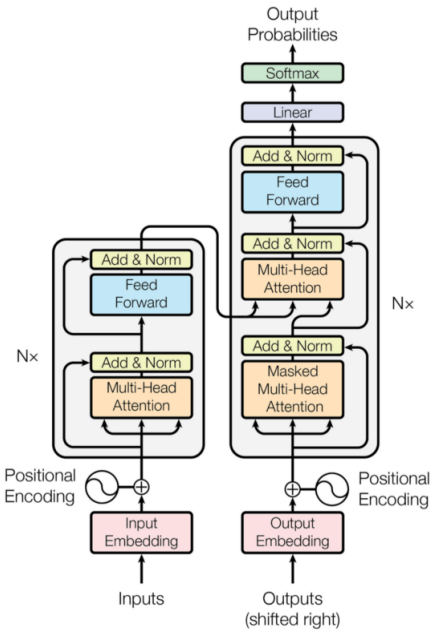


Figure 1: The Transformer - model architecture.

Figure 1: The structure of transformer encoder and decoders (Vaswani et al., 2017).

illustrated in fig. 1

Pre-training is the process of training a model on unlabeled data tasks. One popular task is called "masked language modeling", where a random word is masked and it is the model's objective to recover the masked word from its surrounding text. "Next sentence prediction" is another possible pre-training task, where the model is given a pair of sentences and is asked to decide whether the pair is consecutive from their context. Since such training data require no human labeling, the data used are often massive. For example, English BERT uses around 20GB of data including the English Wikipedia. Training time is also significantly high for this process, which often consumes thousands of GPU hours.

The purpose of pre-training is to reduce time and data requirement of obtaining task specific models. The process of adapting a pre-trained model to a task specific model is called fine-tuning. After adding an extra layer that maps embeddings to the output space, the combined model is trained on labeled data for the task of interest. This process often take less than one hour.

2.2 Linguistic Probing

The cost of pre-training a single transformer is high enough so that it is prohibitive to study the pre-training process via repetition. As a common workaround, linguistic surrogates are used.

The modern classic in this area is Tenney et al. (2019), which placed linear classifiers at the end of each transformer layer of BERT, and fine-tuned the combination to perform linguistic tasks like POS tagging,

parsing, NER, semantic roles, and coreference. The authors discovered that hidden representations at different depth are sensitive to different linguistic features in the order of local to global. Thus they concluded that "BERT rediscovers the classical NLP pipeline".

Other works attempted to develop in various aspects. Hewitt and Manning (2019) attempted to extract syntax tree information from hidden states of BERT. Despite having disagreeing conclusions, Clark et al. (2019) and (Kovaleva et al., 2019) both attempted to seek connections between attention scores and linguistic relationship between words in a sentence. Zhao and Bethard (2020) found a setting where attention quality can be qualitatively measured, and provided insights into attention scores before fine-tuning and during fine-tuning.

2.3 Pre-training on Non-transformer Neural Networks

An interesting approach is used by Papadimitriou and Jurafsky (2020), where pre-training was conducted on LSTM instead of transformers. This makes experiments affordable and repetition feasible. The authors performed pre-training on data from various domains, including non-target human language, music, code, artificial languages (matching parenthesis), and fine-tuned on target human language. It was observed that such pre-training can still benefit target language tasks even when pre-training data are not from the target domain. Baseline models that are pre-trained on random combination of words are used to back the conclusions.

2.4 Training Difficulty of Transformer

There is also a branch of works attempting to explain difficulties of training transformers analytically and empirically. Although pre-training is not addressed directly and the depth of trainable model is often the focus, they can still be relevant to this project as the difficulty of training might be related to the high cost of pre-training.

The first cluster of discoveries was around the position of LayerNorm(Ba et al., 2016) relative to the attention mechanism and residual connections(Chen et al., 2018; Nguyen and Salazar, 2019; Zhang et al., 2019; Xu et al., 2020). The common conclusion was that, the current layer order of $attention \rightarrow dropout \rightarrow residual - add \rightarrow LayerNorm$ (post-LN) causes optimization difficulties. And the solution was to move LayerNorm to the beginning of the sequence, that is $LayerNorm \rightarrow attention \rightarrow dropout \rightarrow residual - add$ (pre-LN).

Zhang et al. (2019) discovered that variance is high for the second momentum of optimizer at the beginning of training, and consequently undermines the training process if a warm-up learning rate

schedule is not used. The proposed solution was to scale randomly initialized parameter by a certain factor. The authors' evaluation suggests their initialization scheme would allow training of much deeper transformer models and the warm-up learning rate is not required.

Liu et al. (2020) on the other hand argued that while gradient variances are unbalanced across parameters, the update rates are still balanced because of adaptive learning rate of the Adam optimizer (Kingma and Ba, 2014). The authors' analysis eventually attributed the difficulty in training to high variance of model outputs across training steps, and proposed to ease the problem by adding an extra trainable scaling factor before each parameter.

3 Motivation: The Reversed BERT Model

Suppose BERT relies on the pipeline discovered by Tenney et al. (2019), then we should expect that reversing such pipeline would undo pre-training. We test this hypothesis by reversing the order of pre-trained transformer layers in BERT, without re-learning any model parameters.

Table 1 compares such reversed model against pre-trained transformer (BERT) and non-pre-trained (clean) transformer model. In contradict to the hypothesis' prediction, it is clear that reversed models always outperform clean models. In tasks like MNLI, SST-2, and QQP the reversed model beat the clean model by more than 30 points and scored below pre-trained models by less than 10 points.

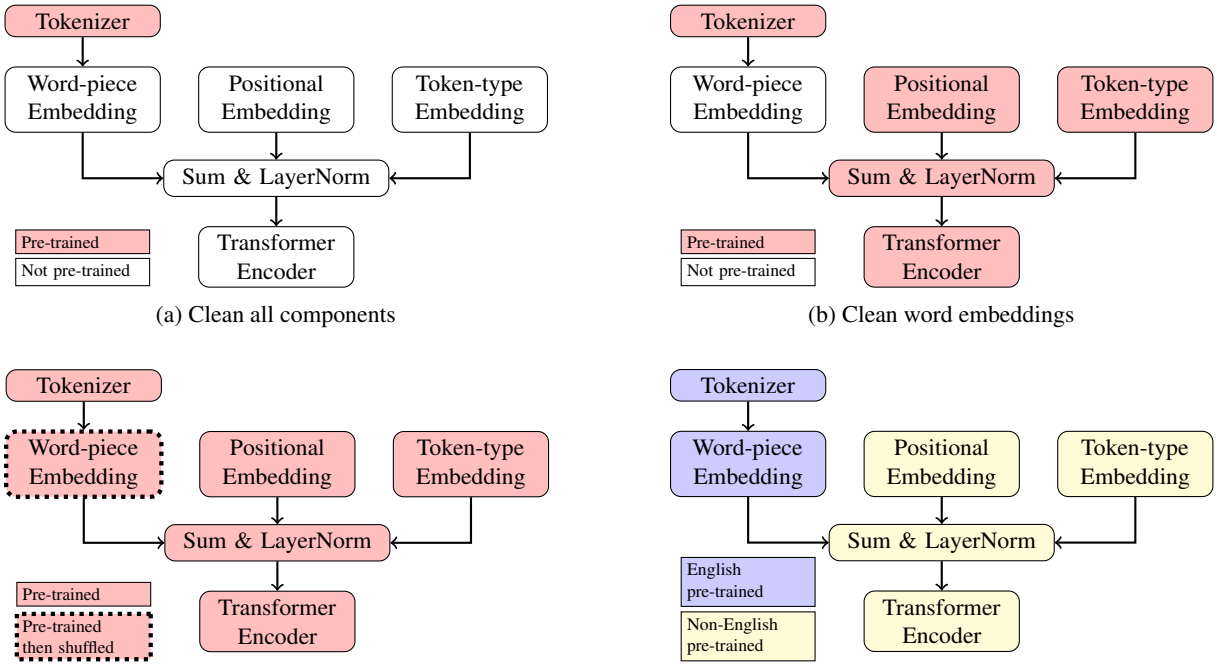
These observations raises two possible scenarios.

1. Such pipeline of specialized steps is not necessary for BERT to work, and the loss of performance comparing to the pre-trained model is caused by damaged model integrity. That is, layer 11 expects inputs come from previous layers but gets outputs of layer 12 in the reversed model. We explore this possibility in section 4.1
2. Such pipeline still exists and works in the reversed model because each step indifferently combines its inputs across contexts. In other words, it was never transformer encoder's role to carry any "knowledge". Since static embeddings are the only component remaining in BERT after transformer encoder, these static embeddings would be the actual carrier of "language knowledges". We test this scenario in section 4.2 and 4.3

These observations also suggest the risk of knowledge uncovered in linguistic probing being irrelevant to the model's operation. This motivated us to avoid surrogates and assess BERT as a totality in its production setting for the rest of this project.

	Pre-trained	Layers Reversed	Clean
CoLA	0.60	0.14	0.00
MNLI-m	0.76	0.68	0.32
MNLI-mm	0.75	0.68	0.32
MRPC	0.89	0.82	0.81
QNLI	0.87	0.66	0.49
QQP	0.85	0.72	0.54
RTE	0.62	0.55	0.53
SST-2	0.92	0.83	0.51
STS-B	0.85	0.47	0.14
WNLI	0.58	0.56	0.56

Table 1: GLUE scores of pre-trained (BERT), BERT with all transformers layers in reverse order, clean (non-pre-trained) transformer. Reversed models always outperform clean models. In tasks like MNLI and SST-2 the reversed model beat the clean model by more than 30 points and scored below pre-trained models by less than 10 points.



(a) Clean all components

(b) Clean word embeddings

(c) Shuffle word embeddings

(d) English word embeddings

Figure 2: Types of modifications to transformer components.

4 More Modified BERT Models

We introduce three more experiments in this section to further investigate two scenarios raised at the end of section 3.

These three experiments contain the following modifications to BERT.

Clean all components All of the word-piece embeddings, positional embeddings, token-type embeddings, layer norms, and transformer encoder (i.e., self-attention layers) are randomly initialized. See fig. 2a.

Clean word embeddings The word-piece embeddings are randomly initialized, but all other components keep their pre-trained values. See fig. 2b.

Shuffle word embeddings All components start with their pre-trained values. The word-piece to embedding alignments (i.e., the word-piece indices into the embedding matrix) are shuffled. Special token alignments are preserved, and we only map English tokens to English embedding vectors. See fig. 2c.

English word embeddings This modification is applied only to non-English pre-trained models. The tokenizer and word-piece embeddings from the non-English pre-trained model are replaced with the ones from the English pre-trained model. See fig. 2d.

4.1 Non-English BERT on English Tasks

BERT models pre-trained on non-English corpus are indeed pre-trained and intact in contrast to the reversed models. Since they had no substantial exposure to English, they are not supposed to be have English-specific pipeline or "knowledge". Suppose the performance drop in the reversed model is caused by damaged by model integrity, then non-English model should perform well.

Table 3 shows that for almost all GLUE tasks, transformers pre-trained on non-English data achieve nearly the performance of pre-training on English. For example, on SST-2 starting with an English pre-trained model yields 0.92 accuracy, while training on German, Chinese, Spanish, and Japanese yield 0.88, 0.88, 0.89, and 0.86, respectively. At the same time, training from scratch, where all model parameters have been randomly initialized, generally yields much lower performance. Again, taking SST-2 as an example, randomly initializing all components yields 0.51 accuracy.

These trends suggest that the non-English pre-trained models are learning something general about language that is useful when fine-tuning for English tasks. Without these basics of language pre-trained into the model, a randomly initialized model struggles to learn most of the GLUE tasks. Note that we see a benefit even from languages like Chinese, where the vocabulary shares little with our target language, English.

There are a few outliers to these trends. In WNLI, all models achieve the same performance, likely because it has only 634 training samples. In CoLA, the non-English pre-trained models yield much lower performance than the English pre-trained model, likely because this task focuses on English grammaticality judgments.

In regard to the possibility of non-English corpus containing substantial English data, we conducted one measurement on tokenizer vocabulary of different models and another on English-like contexts in

Language	shared with EN BERT	English-like
DE	5,775	25,266
ZH	2,672	3,365
ES	5,396	24,984
JP	830	2,548

Table 2: The number of English-like word-pieces in every non-English BERT and their overlap with English BERT. The English BERT has 30,522 word-pieces, and 26,503 word-pieces matching the English-like RegEx.

training data of different models.

For tokenizer vocabulary, we count the overlap between non-English models and the English model. We define "English-like" token using regular expression (##)?[A-Za-z]+, and the numbers are shown in Table 2. There are few word-pieces shared between non-English and English BERTs, especially in the ZH and JP variant.

The limitation of this analysis is that, the overlapping between word-pieces is not necessarily a good indicator for the overlapping of training data or language. On one hand, different data orders would produce different sets of word-piece for the same training data, one the other hand, the same word-piece can mean different things in different languages models. For example, when fetching the English word-piece “a” in the Spanish pre-trained model, we will get back a vector that means something like the English “to” (the translation of Spanish “a”). For languages like Chinese, English-like word-pieces will be even more scattered and less meaningful (ZH tokenizer produces 50% longer sequence than EN on average). During fine-tuning, the model must learn better representations for these word-pieces in English contexts.

For the reason above, we also measured the percentage of "English-like context" in corpus used to train non-English BERTs. We define every 10-word-piece window as a context, and consider a context to be English-like when it contains more than 5 English-like word-pieces. Under this measurement, English-like contexts are 1.45% of the ZH pre-training data (Chinese Wikipedia) and 1.12% of the JA pre-training data (Japanese Wikipedia). We do not perform this analysis for DE and ES, since most word-pieces in these languages already match the English-like RegEx.

4.2 Shuffled Word-Piece Embeddings

Suppose transformer layers combine their inputs indifferently and word-piece embeddings are the actual carrier of “language knowledge”, then shuffling the mapping from token to embedding vector would invalidate such “knowledge” and collapse the pre-trained model.

Table 4 shows that for almost all GLUE tasks, regardless of the pre-training language, transformers

	EN	Foreign language				clean
		DE	ZH	ES	JP	
CoLA	0.60	0.15	0.13	0.26	0.15	0.0
MNLI-m	0.76	0.74	0.66	0.77	0.71	0.32
MNLI-mm	0.75	0.73	0.65	0.78	0.72	0.32
MRPC	0.89	0.89	0.87	0.89	0.86	0.81
QNLI	0.87	0.82	0.81	0.86	0.82	0.49
QQP	0.85	0.82	0.78	0.84	0.82	0.54
RTE	0.62	0.63	0.58	0.62	0.58	0.53
SST-2	0.92	0.88	0.88	0.89	0.86	0.51
STS-B	0.85	0.83	0.76	0.85	0.83	0.14
WNLI	0.58	0.56	0.56	0.56	0.56	0.56

Table 3: Results of using non-English pre-trained models (DE, ZH, ES, JP) or cleaning all components (clean) when fine-tuning for the English GLUE tasks.

	EN			DE			ZH			ES			JP		
	base	shuff	clean	base	shuff	clean	base	shuff	clean	base	shuff	clean	base	shuff	clean
CoLA	0.60	0.19	0.14	0.15	0.14	0.13	0.13	0.14	0.00	0.26	0.17	0.09	0.15	0.18	0.08
MNLI-m	0.76	0.66	0.56	0.74	0.69	0.51	0.66	0.62	0.39	0.77	0.70	0.56	0.71	0.68	0.52
MNLI-mm	0.75	0.67	0.53	0.73	0.69	0.46	0.65	0.62	0.45	0.78	0.70	0.53	0.72	0.69	0.56
MRPC	0.89	0.86	0.81	0.89	0.85	0.82	0.87	0.85	0.82	0.89	0.86	0.82	0.86	0.84	0.82
QNLI	0.87	0.79	0.51	0.82	0.78	0.60	0.81	0.79	0.60	0.86	0.81	0.61	0.82	0.78	0.60
QQP	0.85	0.80	0.60	0.82	0.79	0.65	0.78	0.76	0.62	0.84	0.79	0.67	0.82	0.80	0.61
RTE	0.62	0.55	0.54	0.63	0.56	0.55	0.58	0.58	0.55	0.62	0.59	0.55	0.58	0.58	0.54
SST-2	0.92	0.83	0.82	0.88	0.83	0.79	0.88	0.83	0.79	0.89	0.82	0.79	0.86	0.81	0.80
STS-B	0.85	0.77	0.19	0.83	0.79	0.16	0.76	0.74	0.11	0.85	0.80	0.20	0.83	0.73	0.17
WNLI	0.58	0.56	0.56	0.56	0.56	0.56	0.56	0.56	0.59	0.56	0.56	0.56	0.56	0.56	0.56

Table 4: Effects of shuffling the alignment between word-pieces and embeddings (shuff), as compared to the original unshuffled model (base), and a model with randomly initialized embeddings (clean). Shuffled embedding model still outperforms clean model significantly in most tasks.

with shuffled word-piece embeddings (shuff) perform slightly worse than non-shuffled transformers (base), and substantially better than training embeddings from scratch (clean). For example, in STS-B, we observed baseline-shuffled-clean performance of 0.85-0.77-0.19 (English), 0.83-0.79-0.16 (German), 0.76-0.74-0.11 (Chinese), 0.85-0.80-0.20 (Spanish), and 0.83-0.73-0.17 (Japanese). The only outliers to this trend are CoLA and RTE with Chinese and Japanese, where shuffled performance is just as good as non-shuffled (and still better than random initialization), and WNLI, where almost all models perform the same.

Table 4 also shows that, except for CoLA, shuffled English word-piece embeddings is similar in performance to non-shuffled non-English word-piece embeddings (EN-shuff column vs. DE-base, ZH-base, ES-base, or JP-base). For example, the first five EN-shuff values are 0.19, 0.56, 0.77, 0.83, 0.55, and the first five JP-base values are 0.15, 0.56, 0.83, 0.86, 0.58. In other words, forcing the model to re-learn English word-piece to English embedding alignments is similar in difficulty to forcing the model to learn English word-piece to non-English embedding alignments.

These trends suggest that transformers find it easier to learn alignments between word-pieces and

	EN			DE			ZH			ES			JP		
	base	english	clean	base	english	clean	base	english	clean	base	english	clean	base	english	clean
CoLA	0.60	NA	0.14	0.15	0.14	0.13	0.13	0.15	0.00	0.26	0.16	0.09	0.15	0.18	0.08
MNLI-m	0.76	NA	0.56	0.74	0.72	0.51	0.66	0.49	0.39	0.77	0.74	0.56	0.71	0.74	0.52
MNLI-mm	0.75	NA	0.53	0.73	0.73	0.46	0.65	0.56	0.45	0.78	0.75	0.53	0.72	0.75	0.56
MRPC	0.89	NA	0.81	0.89	0.83	0.82	0.87	0.82	0.82	0.89	0.83	0.82	0.86	0.81	0.82
QNLI	0.87	NA	0.51	0.82	0.81	0.60	0.81	0.70	0.60	0.86	0.82	0.61	0.82	0.79	0.60
QQP	0.85	NA	0.60	0.82	0.82	0.65	0.78	0.66	0.62	0.84	0.82	0.67	0.82	0.82	0.61
RTE	0.62	NA	0.54	0.63	0.56	0.55	0.58	0.55	0.55	0.62	0.58	0.55	0.58	0.57	0.54
SST-2	0.92	NA	0.82	0.88	0.85	0.79	0.88	0.84	0.79	0.89	0.87	0.79	0.86	0.85	0.80
STS-B	0.85	NA	0.19	0.83	0.81	0.16	0.76	0.13	0.11	0.85	0.79	0.20	0.83	0.80	0.17
WNLI	0.58	NA	0.56	0.56	0.58	0.56	0.56	0.62	0.59	0.56	0.56	0.56	0.56	0.56	0.56

Table 5: Effects of replacing foreign-language tokenizer and word-piece embeddings with English ones (english), as compared to the original unshuffled model (base), and a model with randomly initialized embeddings (clean).

pre-trained embeddings than to learn embeddings from scratch. This is consistent with the hypothesis that monolingual models learn embeddings that can be easily projected to other languages, as was suggested by Conneau et al. (2020).

4.3 English Word-Piece Embeddings in Non-English BERT

This experiment complements the shuffled embedding experiment in section 4.2. Suppose word-piece embeddings are the primary carrier of "knowledge" and transformer encoder are generic, then combining English embeddings with a non-English transformer encoder bears an opportunity to produce an English model out of a non-English transformer encoder. In other words, we should expect the mixed model to perform better than the original non-English model.

Table 5 shows that the results with English word-piece embeddings in non-English transformers are similar to the results from shuffling word-piece embeddings: transformers pre-trained on non-English data and then injected with English word-piece embeddings (english) outperform models where the word-piece embeddings are trained from scratch (clean), and approach the performance of models with their original embeddings (base).

These trends again suggest that there is something in monolingually-learned embeddings that is useful to fine-tuning transformers, regardless of the language of the fine-tuning task.

It is worth noting that the "base" columns are generally a bit higher than the same-language "shuff" or "english" columns in both tables 4 and 5. This suggests that learning from non-English shuffled or English-injected models is a bit more difficult than learning from unmodified non-English models whose learned parameters are presumably more coherent.

4.4 Analysis

In terms of the two possibilities raised at the end of section 3, all of the three experiments affirmed the importance of model integrity as non-English models performed well while modified models in section 4.2

and section 4.3 all experienced performance loss. It is also evident that the word-piece embeddings are not the only "knowledge" carrier in BERT. When contrasted with clean embedding models, it is clear that pre-training of word-piece embedding is still necessary, and certain level of coherency between word-piece embeddings and transformer encoder is needed.

More importantly, we found that pre-training grants BERT certain quality that transcends the language used for pre-training.

We find that transformers pre-trained on non-English data do surprisingly well when fine-tuned on English GLUE tasks. The performance of non-English-pre-trained models is often close to the performance of English-pre-trained models, and is well above the performance of randomly initialized models. We further find that randomly shuffling the word embeddings of the English pre-trained model leads to similar performance as when a non-English pre-trained model is used instead.

We take these findings to suggest that monolingually pre-trained transformers are learning something about language in general that is useful beyond their source language. It is likely that at least part of what is being learned is a set of word-piece embeddings that are a good starting point for any language. The exact word-piece-to-embedding alignments must be learned for each new language (i.e., by fine-tuning), but that appears to be a much easier task than constructing word-piece embeddings for a language from scratch.

5 Initialization

This section contains two of our attempts to study the effect of initialization on transformer performance.

5.1 Variance and Magnitude

Variance and magnitude of parameters can influence the variance and magnitude of gradients, thus affect the quality of updates and convergence (Glorot and Bengio, 2010). It would be helpful to know whether or not a changed magnitude or variance of parameters contributes to the working of pre-trained transformers. The experimentation is straightforward: for each pre-trained weight matrix, we note its magnitude (or variance), randomly re-initialize its values, and scale it to match pre-trained magnitude (or variance). We did not observe significant differences among them and the default initialization.

5.2 Oracle Distribution

From the other side, we attempted to establish the "speed of light" performance for random initialization by initialize each parameter to exactly the same distribution as that of its pre-trained counterpart. That is,

	Pre-trained	Shuffled	Oracle	Clean
CoLA	0.60	0.19	0.16	0.00
MNLI-m	0.76	0.66	0.32	0.32
MNLI-mm	0.75	0.67	0.53	0.32
MRPC	0.89	0.86	0.81	0.81
QNLI	0.87	0.79	0.62	0.49
QQP	0.85	0.80	0.65	0.54
RTE	0.62	0.55	0.53	0.53
SST-2	0.92	0.83	0.84	0.51
STS-B	0.85	0.77	0.22	0.14
WNLI	0.58	0.56	0.56	0.56

Table 6: BERT, shuffled embedding BERT, Oracle initialization, and randomly initialization. By re-permute elements within each weight matrix, we can establish the "speed of light" performance of randomly initialized model, since the resultant parameter matrix has the same element distribution as its pre-trained counterpart.

for each weight matrix, we shuffle its elements.

Results are in table 6. It is clear that these shuffled-parameter model can still outperform clean model, as we can see in CoLA, MNLI-mm, QNLI, QQP, SST-2, and STS-B. When compared with other disturbed models, the shuffled-parameter model can be hit or miss. That is, for instance, the shuffled-parameter model is worse than the reversed model in STS-B (0.22 vs 0.47), but they performed equally well in SST-2.

When contrasted with the variance and magnitude experiments, these results indicate the importance of the shape of element distribution when it comes to parameter initialization. However, it was unclear what exactly is this distribution and how to obtain it without an already pre-trained model. It is also noticeable that the shuffled-parameter model never come close to non-English pre-trained model, which suggests the benefit of pre-training does not stop at the distribution of model parameters.

6 Example Application

The understanding that the transformer receives generic benefits from pre-training could promote more liberty in engineering. Here we demonstrate one possible use of such knowledge toward the goal of shrinking the size of pre-trained transformer models.

Because smaller models run faster and consume less memory and storage, it is desirable to shrink the size of model while preserving prediction accuracy. The most conventional solution is call "distilation" (Sanh et al., 2019; Jiao et al., 2020), where a smaller "student" model is trained to imitate the original "teacher" model. One such example is DistilBERT, which halved the number of parameters in BERT and took 90 hours on 8 V100 GPUs to train. As an alternative to distillation, we opt to simply remove half

	BERT	Distil	Remove
Cost	-	720 GPU hours	1 CPU second
Speed	1×	2×	2×
# param(M)	110	66	66

Table 7: Number of parameters and cost to make for different models.

	base	distil	remove		clean
			Last 6	Every 2	
CoLA	0.60	0.56	0.41	0.41	0.0
MNLI-m	0.76	0.81	0.66	0.65	0.32
MNLI-mm	0.75	0.82	0.69	0.66	0.32
MRPC	0.89	0.90	0.85	0.84	0.81
QNLI	0.87	0.89	0.85	0.83	0.49
QQP	0.85	0.87	0.86	0.85	0.54
RTE	0.62	0.63	0.60	0.57	0.53
SST-2	0.92	0.90	0.91	0.91	0.51
STS-B	0.85	0.86	0.85	0.81	0.14
WNLI	0.58	0.53	0.54	0.49	0.56

Table 8: GLUE(dev) performance of Baseline BERT, DistilBERT, BERT with its last 6 layers removed, BERT with every second of its layers removed, clean BERT. Model produced by removing last 6 layers resembled the performance of BERT and DistilBERT closely in tasks like STS-B, SST-2, RTE, QQP, QNLI, MRPC.

of layers from BERT. In particular, we removed the last 6 layers in one instance, and removed the every second layer in another.

Table 8 shows that the models produced by removing layers, especially by removing the last 6 layers, resembled the performance of BERT and DistilBERT closely in tasks like STS-B, SST-2, RTE, QQP, QNLI, MRPC. In STS-B, our model performed as good as BERT and beat non-pre-trained transformer by more than 70 points. For the remaining tasks (CoLA, MNLI-m, and MNLI-mm), our model is still good enough to perform closer to BERT than to clean transformers. The only outlier task is WNLI, which has only a few hundred training samples. We think it is reasonably respectable performance, considering it only took less than 1 second to produce such model in contrast to hundreds of GPU hours when it comes to distillation.

7 Result Reliability

Each experiment in section 3 and section 4 is performed three times with no fixed random seed and the best epoch of the best run is reported. Each experiment in section 5 and section 6 is performed once and the best epoch is reported.

For experiment configurations that are repeated for three times, we observed typical difference of 3

points from the highest run to the lowest run. In addition, each kind of experiment is replicated on different tasks and different variants of BERT as shown in tables, which we believe demonstrates stability of our results.

8 Conclusion

We find that transformers pre-trained on non-English data do surprisingly well when fine-tuned on English GLUE tasks. The performance of non-English-pretrained models is often close to the performance of English-pretrained models, and is well above the performance of randomly initialized models. We further find that randomly shuffling the word embeddings of the English pre-trained model leads to similar performance as when a non-English pre-trained model is used instead.

We take these findings to suggest that pre-trained transformers receive quality that transcends the source language. In search of description of such benefit, we attempted to synthesize a model that can perform well without pre-training. While such attempts were not successful, the oracle experiment shows the benefit of a proper initialization.

We also demonstrated such knowledge can guide more liberated approach in engineering though the model shrinking example, where our model only consume trivial resources and perform competitively to counterparts that take days to produce.

Some limitations of our current work suggest future directions. We focused on GLUE tasks and BERT models, but it would be interesting to replicate our results on other tasks, such as named-entity tagging or machine translation, and with other models, such as GPT-2 (Radford et al., 2019), XLNet (Yang et al., 2019), or T5 (Raffel et al., 2020). The cause of reported phenomenons is still unclear, and our effort toward this direction is limited compared to the large space of possibilities. We nonetheless believe our research provides an interesting direction for future work: attempting to describe the quality transformers obtained from pre-training, searching for causes of such qualities, and adopting such qualities in engineering practices.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

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

Emad Elwany, Dave Moore, and Gaurav Oberoi. 2019. Bert goes to law school: Quantifying the competitive advantage of access to large legal corpora in contract understanding. *arXiv preprint arXiv:1911.00473*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tiny-BERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *CoRR*, abs/1901.08746.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763, Online. Association for Computational Linguistics.

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. *CoRR*, abs/1907.11692.

Toan Q. Nguyen and Julian Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. *CoRR*, abs/1910.05895.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning Music Helps You Read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839, Online. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. 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. Association for Computational Linguistics.

Hongfei Xu, Qiuhui Liu, Josef van Genabith, Deyi Xiong, and Jingyi Zhang. 2020. Lipschitz constrained parameter initialization for deep transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 397–402, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 898–909, Hong Kong, China. Association for Computational Linguistics.

Yiyun Zhao and Steven Bethard. 2020. How does BERT's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4729–4747, Online. Association for Computational Linguistics.