# A Primer in BERTology:

# What We Know About How BERT Works

Anna Rogers | Olga Kovaleva | Anna Rumshisky

---

Transformers have been put into much use in NLP systems. BERT is the most famous Transformer-based model. Although everyone appreciates the abilities of BERT, it is still unclear why BERT works better. The paper tries to understand the linguistic and world knowledge learned by BERT and tries to understand how and where it is stored along with a detailing of the technical aspects of BERT.

A simple description of the architecture of BERT would go as follows. BERT is a stack of Transformer encoder layers with self-attention heads. The head computes the key, value, and query vectors to create the weighted representations. Each layer is followed by a normalization. The pretraining involves masked language modeling (MLM) and next sentence prediction (NSP). The finetuning is done by adding fully connected layers on top of the final encoder layer.

Several probing tasks have been made to understand the knowledge captured in BERT weights. The studies show that BERT representations could be more hierarchical than linear. BERT also encodes information about parts of speech, syntax, etc. However, syntactical knowledge is not directly encoded in the weights. It is usually recovered from the BERT token representations. BERT does fill in the gap by taking into account the subject-predicate agreement. BERT is also found to detect the presence of negative polarity items (NPIs) and the words that allow their use than scope violations. This leads us to conclude that BERT does not exactly understand negation, is insensitive to malformed input, and the syntactic knowledge that it gains, which is said to be learned “naturally” is incomplete or is irrelevant to it while performing tasks.

BERT was found to have some knowledge of semantic roles although it shows a preference for incorrect roles that are semantically related to the correct ones. BERT is also seen to struggle with number representations, especially that of floating-point numbers. Considering the world knowledge of BERT, we see that it struggles with pragmatic inference, role-based event knowledge, and visual perceptions. Vanilla BERT was found to generalize well with unseen data and also found to be more reliable in tasks that rely on knowledge bases. Although having said about world knowledge, it must be noted that BERT can not reason based on this information. But despite all this knowledge, we come to the conclusion that the results depend on the probing pattern, the information encoded in the weights. Although studies have also told us that the focus should ideally not be on the amount of information encoded, but on the amount of information that can be extracted.

BERT embeddings are contextualized embeddings. Distilled contextualized embeddings are said to perform well in several studies. There are several ways to do this and training a contextualized embeddings with static embeddings is one of them. The BERT embeddings could be explored further through the eyes of isotropy that is beneficial for static embeddings. The representation of the same word is said to have a much greater cosine similarity in BERT. However, the position of the word in a sentence is said to influence this similarity to a great extent.

Studies have been done to identify the functionality of the BERT self-attention heads. One study said that some heads specialized in syntactic relations. The heads attend to words in obj role more than the positional baselines. But some complex dependencies like dobj are encoded by a combination of heads. Having said that, heads have only partial knowledge and do not carry the complete syntactic tree information. Most self-attention heads do not encode non-trivial linguistic information either.

The first layer has the most information about the linear word order because this is where we input a combination of tokens, segment, and positional embeddings. As we progress, the knowledge of linear word order decreases, and knowledge of the hierarchical sentence structure improves. The middle layers have much syntactic information. The final layers tend to be more task-specific. This explains a lot of behavior of BERT including the reason why restoring the weights of the lower layers after finetuning doesn't change the results much in a task.

In the overall BERT architecture, we see that the number of layers is more important than the number of heads. We can see a significant change in results when we change the number of layers and heads though. A deeper model would mean more information is encoded in the weights. Vanilla BERT seems to learn patterns with its self-attention heads. There are benefits to large batch training. Certain researchers have shown that a batch size of 32,000 does not degrade the performance of BERT but at the same time, it helps reduce the training time. Training can be stabilized by normalization of trained tokens and this improves performance in classification tasks. A warm start is when shallower parts are trained first and the parameters are copied to the deeper layers. This again improves the speed of training.

The original BERT is pre-trained on two tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP). We could have alternate objectives. The categories of these include 1. How to mask, 2. What to mask, 3. Where to mask, 4. Alternatives to masking, 5. NSP alternatives, 6. Others. We can improve the performance further by improving the corpus volume. Pre-training is the most expensive part of the training. But it need not always provide the best results. Some cases exist where random initializing and finetuning gives better results. There is a need to go further to investigate the benefits of pre-training.

Pre-training and finetuning are two important tasks. Pre-training help BERT gain knowledge to perform independently of the task while finetuning helps BERT rely on the representations at hand and perform better for the specific task that it is employed for. Finetuning can be improved with methods such as 1. Taking more layers into account, 2. Intermediate supervised training stage between the traditional pre-training and finetuning approaches, 3. Adversarial token perturbations, 4. Adversarial regularization to prevent overfitting, 5. Mixout regularization. Further, adapters in BERT can be used to reduce the computational complexity or expense of the finetuning process and also for more efficient multi-task learning. It was found that BERT finetuned on GLUE tasks gave significantly better results.

Transformer based models grow by order of magnitudes. At the same time, we can not say that the models make the best use of parameters they have as almost all the transformer heads could be pruned without much loss in performance. This is quite true with BERT as well. Some BERT heads turn out to be redundant based on the task and it becomes necessary to prune the heads to achieve better performance. In general, though, we could say that larger BERT models perform better. But this need not be the case always. BERT can however be compressed efficiently without much accuracy loss and this is done with the help of knowledge distillation, quantization, and pruning. Large portions could be dropped or pruned without significant loss in performance. LayerDrop is one such method. The results of such techniques to improve the performance of BERT have been summarized in the table below. There are other techniques including decomposing embedding matrices into smaller ones, elimination of intermediate encoder outputs, etc.

A new approach has been highlighted that uses pruning to analyze a model and see which parts are actually useful and what is the knowledge that is utilized properly etc. However, this can't be restricted to the removal of heads or layers as studies have shown that there are representations that are spread across the network.

BERTology has shown that BERT has still a long way to go and is nowhere near to actually "understanding" human speech. Further research can be done on developing benchmarks for linguistic competence or methods can be developed to help models learn to reason etc.

Tables:

| | | Compression | Performance | Speedup | Model | Evaluation |
|---|---|---|---|---|---|---|
| | BERT-base (Devlin et al., 2019) | ×1 | 100% | ×1 | $BERT_{12}$ | All GLUE tasks, SQuAD |
| | BERT-small | ×3.8 | 91% | - | $BERT_4$† | All GLUE tasks |
| Distillation | DistilBERT (Sanh et al., 2019a) | ×1.5 | 90%§ | ×1.6 | $BERT_6$ | All GLUE tasks, SQuAD |
| | $BERT_6$-PKD (Sun et al., 2019a) | ×1.6 | 98% | ×1.9 | $BERT_6$ | No WNLI, CoLA, STS-B; RACE |
| | $BERT_3$-PKD (Sun et al., 2019a) | ×2.4 | 92% | ×3.7 | $BERT_3$ | No WNLI, CoLA, STS-B; RACE |
| | Aguilar et al. (2019), Exp. 3 | ×1.6 | 93% | - | $BERT_6$ | CoLA, MRPC, QQP, RTE |
| | BERT-48 (Zhao et al., 2019) | ×62 | 87% | ×77 | $BERT_{12}$*† | MNLI, MRPC, SST-2 |
| | BERT-192 (Zhao et al., 2019) | ×5.7 | 93% | ×22 | $BERT_{12}$*† | MNLI, MRPC, SST-2 |
| | TinyBERT (Jiao et al., 2019) | ×7.5 | 96% | ×9.4 | $BERT_4$† | No WNLI; SQuAD |
| | MobileBERT (Sun et al., 2020) | ×4.3 | 100% | ×4 | $BERT_{24}$† | No WNLI; SQuAD |
| | PD (Turc et al., 2019) | ×1.6 | 98% | ×2.5‡ | $BERT_6$† | No WNLI, CoLA and STS-B |
| | WaLDORf (Tian et al., 2019) | ×4.4 | 93% | ×9 | $BERT_8$†‖ | SQuAD |
| | MiniLM (Wang et al., 2020b) | ×1.65 | 99% | ×2 | $BERT_6$ | No WNLI, STS-B, $MNLI_{mm}$; SQuAD |
| | MiniBERT(Tsai et al., 2019) | ×6** | 98% | ×27** | $mBERT_3$† | CoNLL-18 POS and morphology |
| | BiLSTM-soft (Tang et al., 2019) | ×110 | 91% | ×434‡ | $BiLSTM_1$ | MNLI, QQP, SST-2 |
| Quantization | Q-BERT-MP (Shen et al., 2019) | ×13 | 98%¶ | - | $BERT_{12}$ | MNLI, SST-2, CoNLL-03, SQuAD |
| | BERT-QAT (Zafrir et al., 2019) | ×4 | 99% | - | $BERT_{12}$ | No WNLI, MNLI; SQuAD |
| | GOBO(Zadeh and Moshovos, 2020) | ×9.8 | 99% | - | $BERT_{12}$ | MNLI |
| Pruning | McCarley et al. (2020), ff2 | ×2.2‡ | 98%‡ | ×1.9‡ | $BERT_{24}$ | SQuAD, Natural Questions |
| | RPP (Guo et al., 2019) | ×1.7‡ | 99%‡ | - | $BERT_{24}$ | No WNLI, STS-B; SQuAD |
| | Soft MvP (Sanh et al., 2020) | ×33 | 94%¶ | - | $BERT_{12}$ | MNLI, QQP, SQuAD |
| | IMP (Chen et al., 2020), rewind 50% | ×1.4–2.5 | 94–100% | - | $BERT_{12}$ | No MNLI-mm; SQuAD |
| Other | ALBERT-base (Lan et al., 2020b) | ×9 | 97% | - | $BERT_{12}$† | MNLI, SST-2 |
| | ALBERT-xxlarge (Lan et al., 2020b) | ×0.47 | 107% | - | $BERT_{12}$† | MNLI, SST-2 |
| | BERT-of-Theseus (Xu et al., 2020) | ×1.6 | 98% | ×1.9 | $BERT_6$ | No WNLI |
| | PoWER-BERT (Goyal et al., 2020) | N/A | 99% | ×2–4.5 | $BERT_{12}$ | No WNLI; RACE |

Table 1: Comparison of BERT compression studies. Compression, performance retention, inference time speedup figures are given with respect to $BERT_{base}$, unless indicated otherwise. Performance retention is measured as a ratio of average scores achieved by a given model and by $BERT_{base}$. The subscript in the model description reflects the number of layers used. *Smaller vocabulary used. †The dimensionality of the hidden layers is reduced. ‖Convolutional layers used. ‡Compared to $BERT_{large}$. **Compared to mBERT. §As reported in (Jiao et al., 2019). ¶In comparison to the dev set.