Suleiman Khan, Ph.D. Follow

Sep 4, 2019 · 5 min read ★ · Listen

BERT, RoBERTa, DistilBERT, XLNet — which one to use?

varying improvements over BERT have been shown — and here I will contrast the main similarities and differences so you can choose which one to use in your research or application.

BERT is a bi-directional transformer for pre-training over a lot of unlabeled textual data to learn a language representation that can be used to fine-tune for specific machine learning tasks. While BERT outperformed the NLP state-of-the-art on several challenging tasks, its performance improvement could be attributed to the bidirectional transformer, novel pre-training tasks of Masked Language Model and Next Structure Prediction along with a lot of data and Google's compute power. If you are not yet familiar with BERT's basic technology, I recommend reading this 3-minute blog post quickly.

> *Lately, several methods have been presented to improve BERT on either its prediction metrics or computational speed, but not both.*

XLNet and RoBERTa improve on the performance while DistilBERT improves on the inference speed. The table below compares them for what they are!

	BERT	RoBERTa	DistilBERT	XLNet
Size (millions)	**Base**: 110 **Large**: 340	**Base**: 110 **Large**: 340	**Base**: 66	**Base**: ~110 **Large**: ~340
Training Time	**Base**: 8 x V100 x 12 days* **Large**: 64 TPU Chips x 4 days (or 280 x V100 x 1 days*)	**Large**: 1024 x V100 x 1 day; 4-5 times more than BERT.	**Base**: 8 x V100 x 3.5 days; 4 times less than BERT.	**Large**: 512 TPU Chips x 2.5 days; 5 times more than BERT.
Performance	Outperforms state-of-the-art in Oct 2018	2-20% improvement over BERT	3% degradation from BERT	2-15% improvement over BERT
Data	16 GB BERT data (Books Corpus + Wikipedia). 3.3 Billion words.	[illegible] additional)	16 GB BERT data. 3.3 Billion words.	**Base**: 16 GB BERT data **Large**: 113 GB (16 GB BERT data + 97 GB additional). 33 Billion words.
Method	BERT (Bidirectional Transformer with MLM and NSP)	BERT without NSP**	BERT Distillation	Bidirectional Transformer with Permutation based modeling

** uses larger mini-batches, learning rates and step sizes for longer training along with differences in masking procedure.

*** Numbers as given in the original publications, unless specified otherwise.

XLNet is a large bidirectional transformer that uses improved training methodology, larger data and more computational power to achieve better than BERT prediction metrics on 20 language tasks.

To improve the training, XLNet introduces permutation language modeling, where all tokens are predicted but in random order. This is in contrast to BERT's masked language model where only the masked (15%) tokens are predicted. This is also in contrast to the traditional language models, where all tokens were predicted in *sequential order* instead of *random order*. This helps the model to learn bidirectional relationships and therefore better handles dependencies and relations between words. In addition, Transformer XL was used as the base architecture, which showed good performance even in the absence of permutation-based training.

XLNet was trained with over 130 GB of textual data and 512 TPU chips running for 2.5 days, both of which ar e much larger than BERT.

RoBERTa. Introduced at Facebook, Robustly optimized BERT approach RoBERTa, is a retraining of BERT with improved training methodology, 1000% more data and compute power.

To improve the training procedure, RoBERTa removes the Next Sentence Prediction (NSP) task from BERT's pre-training and introduces dynamic masking so that the masked token changes during the training epochs. Larger batch-training sizes were also found to be

News dataset (63 million articles, 76 GB), Web text corpus (38 GB) and Stories from Common Crawl (31 GB). This coupled with whopping 1024 V100 Tesla GPU's running for a day, led to pre-training of RoBERTa.

As a result, RoBERTa outperforms both BERT and XLNet on GLUE benchmark results:

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$BERT_{LARGE}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$XLNet_{LARGE}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

Performance comparison from RoBERTa.

On the other hand, to reduce the computational (training, prediction) times of BERT or related models, a natural choice is to use a smaller network to approximate the performance. There are many approaches that can be used to do this, including pruning, distillation and quantization, however, all of these result in lower prediction metrics.

DistilBERT learns a distilled (approximate) version of BERT, retaining 97% performance but using only half the number of parameters (paper). Specifically, it does not has token-type embeddings, pooler and retains only half of the layers from Google's BERT. DistilBERT uses a technique called distillation, which approximates the Google's BERT, i.e. the large neural network by a smaller one. The idea is that once a large neural network has been trained, its full output distributions can be approximated using a smaller network. This is in some sense similar to posterior approximation. One of the key

Note: In Bayesian statistics, we are approximating the true posterior (from the data), whereas with distillation we are just approximating the posterior learned by the larger network.

So which one to use?

> If you really need a faster inference speed but can compromise few-% on prediction metrics, DistilBERT is a starting reasonable choice, however, if you are looking for the best prediction metrics, you'll be better off with Facebook's RoBERTa.
>
> Theoratically, XLNet's permutation based training should handle dependencies well, and might work better in longer-run.
>
> However, Google's BERT does serve a good baseline to work with and if you don't have any of the above critical needs, you can keep your systems running with BERT.

Conclusion

Most of the performance improvements (including BERT itself!) are either due to increased data, computation power, or training procedure. While these do have a value of

Update

Part-2 of the blog discussing latest methods in 2020 and 2021 can be found here.

The writer tweets at @SuleimanAliKhan. —We are Hiring —