BERT (Bidirectional Encoder Representation for Transformers): Pre-training of Deep Bidirectional Transformers for Language Understanding (Google AI Language)

February 7, 2020

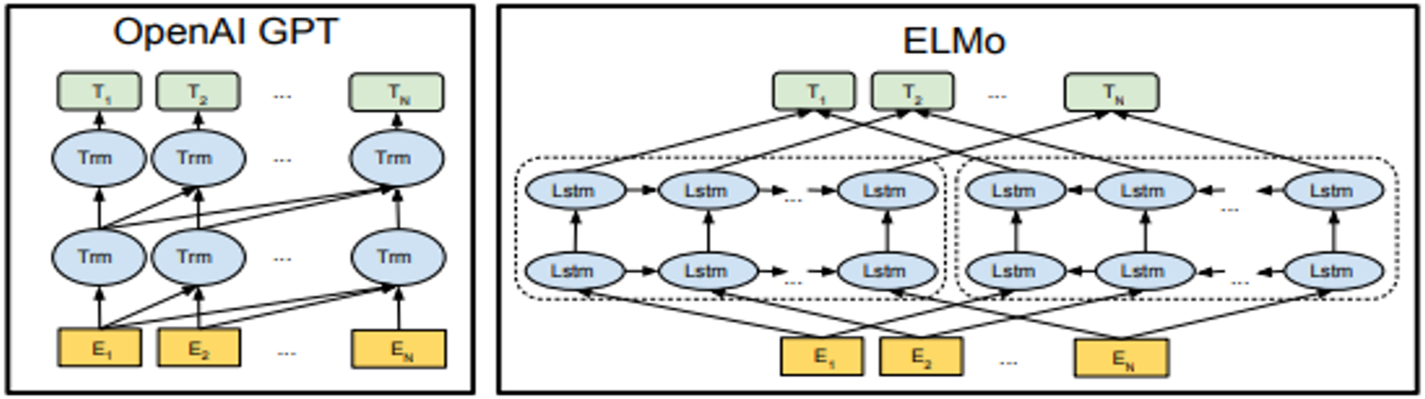
Presenter: Rishab Bamrara

https://qdata.github.io/deep2Read/

Motivation

- When applying pre-trained language representations to downstream task such as Natural Language Inference (NLI), NER or Q/A, two strategies are currently employed:
    - Feature Based (ELMO) → uses task specific architectures
    - Fine-tuning (OpenAI GPT) → fine tunes all pre-trained params.

- However, both of them use unidirectional language models to learn general language representations and this limits the choice of architectures that can be used during pre-training.



Background

- **Masked Language Model (MLM):** This model randomly masks some of the tokens from the input and the objective is to predict the original vocabulary id of the masked word based only on its context. This model also enables the representation to fuse the right and left context.

- **Pretraining:** Here the model is trained on unlabeled data over different pre-training tasks.

- **Fine-Tuning:** Taking pre-trained parameters the model is initialized, and then all the parameters are fine-tuned using labeled data from the downstream tasks.

Related Work

1. **ELMO:** Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In NAACL.
2. **OpenAI GPT:** Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.
3. **Transformer:** Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. **Attention is all you need.**
4. **WordPiece embeddings:** 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.

Claim / Target Task

Authors present BERT to improve fine-tuning based approaches. They claim that BERT alleviates unidirectionality constraint by using a “masked language model” (MLM) pre-training objective.

An Intuitive Figure Showing WHY Claim

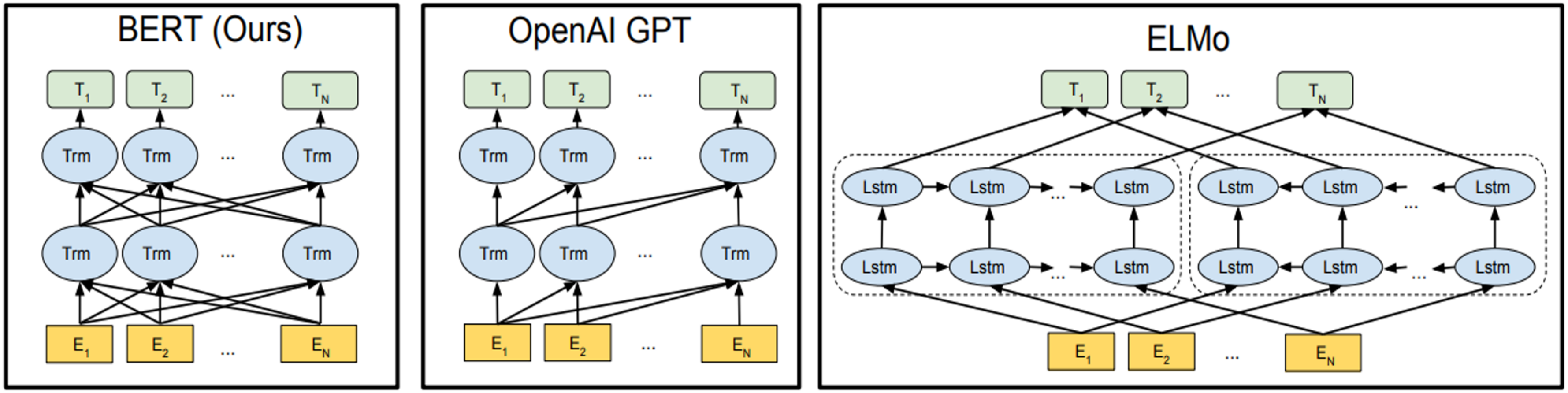


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

Proposed Solution

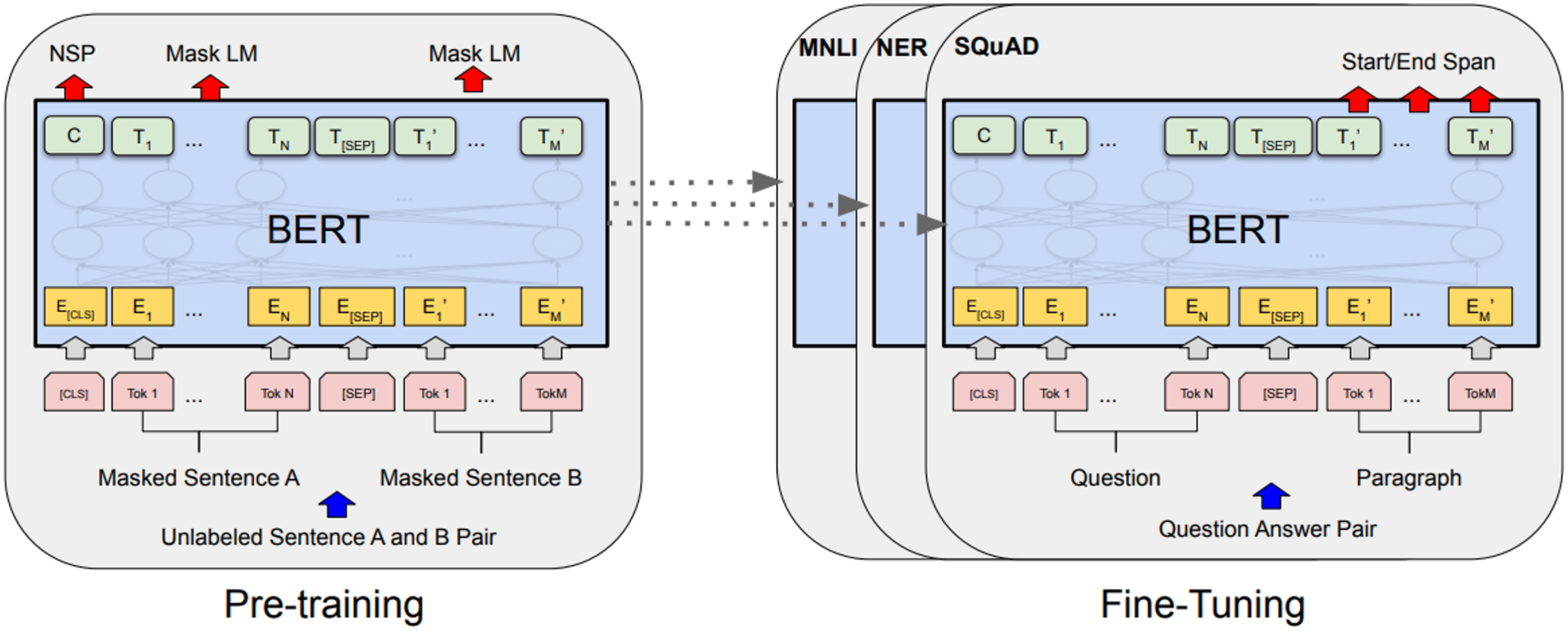


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. `[CLS]` is a special symbol added in front of every input example, and `[SEP]` is a special separator token (e.g. separating questions/answers).

Model Architecture

- BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in Vaswani et al. (2017) and released in the tensor2tensor library.

- Used 2 model sizes:

  - **$\mathbf{BERT_{BASE}}$**: (L=12, H=768, A=12, Total Parameters=110M)

  - **$\mathbf{BERT_{LARGE}}$**: (L=24, H=1024, A=16, Total Parameters=340M)

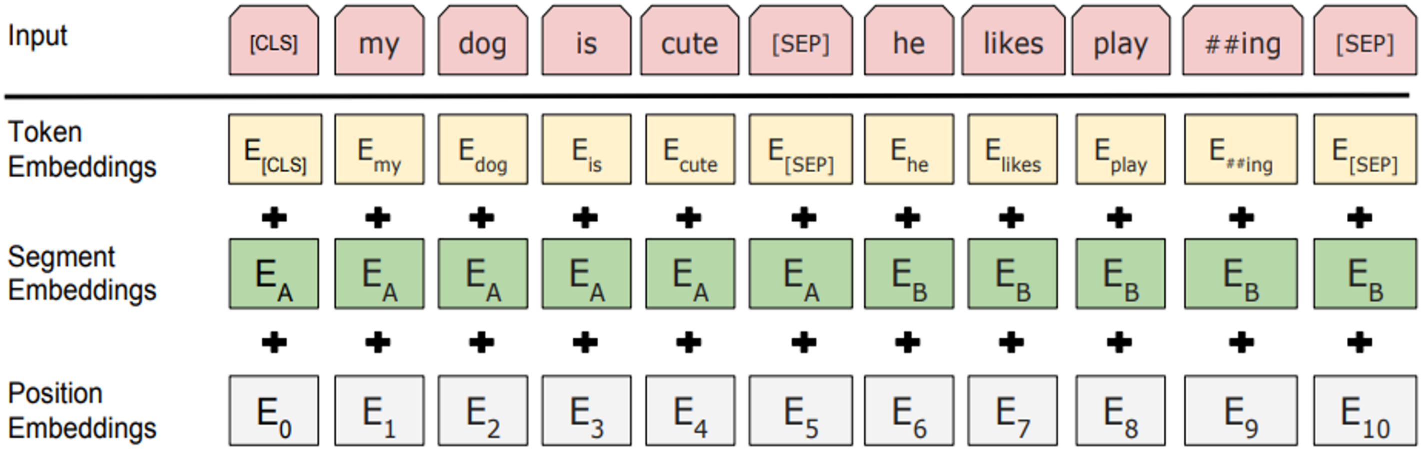
  Here, **L:** number of layers (i.e., Transformer blocks), **H:** the hidden size, **A:** and the number of self-attention heads.

- **$\mathbf{BERT_{BASE}}$** was chosen to have the same model size as OpenAI GPT for comparison purposes.

Input/Output Representations

- Used token sequence to represent both a single sentence and a pair of sentences.
- WordPiece Embeddings with a 30k token vocab.
- First token of every sequence is a special classification token [CLS]. The final hidden state of this token is used for classification tasks.
- Sentences are seperated with special token [SEP] and a learned embedding to every token indicating whether it belongs to sentence A or B.
- Similar to Transformers, input representation is constructed by summing the corresponding token, segment with positional embeddings.

Input/Output Representations (Fig.)



Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[SEP]}$	E_{he}	E_{likes}	E_{play}	$E_{\#\#ing}$	$E_{[SEP]}$
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

Pre-training BERT

Pre-training was done using two unsupervised tasks: MLM and NSP.

Task 1: MLM

- Simply mask some percentage of the input tokens at random, and then predict those masked tokens. The final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary.
- A downside is that we are creating a mismatch between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning.
- The training data generator chooses 15% of the token positions at random for prediction. If the i-th token is chosen, we replace the i-th token with (1) the [MASK] token 80% of the time (2) a random token 10% of the time (3) the unchanged i-th token 10% of the time.

Pre-training BERT

Task 2: NSP

- Beneficial for many downstream tasks such as QA and NLI which are based on relationship between two sentences.
- For this, they chose two sentences A and B for each pre-training example. 50% of the time, the second sentence is logical continuation of first (labeled as *IsNext*), and 50% of the times the second sentence is any arbitrary sentence (labeled as *NotNext*).
- Final hidden state of the [CLS] token, “C”, is used for this task as it is a binary level classification task.

Pre-training Data

- Follow the existing literature on language model pre-training.
- BookCorpus (800M words) and English Wikipedia (2,500M words). For Wikipedia, only text passages were extracted and list, tables and headers were ignored.
- Critical to use a document-level corpus rather than a sentence-level corpus in order to extract long contiguous sequences.

Fine-tuning BERT

- Very straightforward as Transformer allows BERT to model many downstream tasks containing a single text or text pairs.

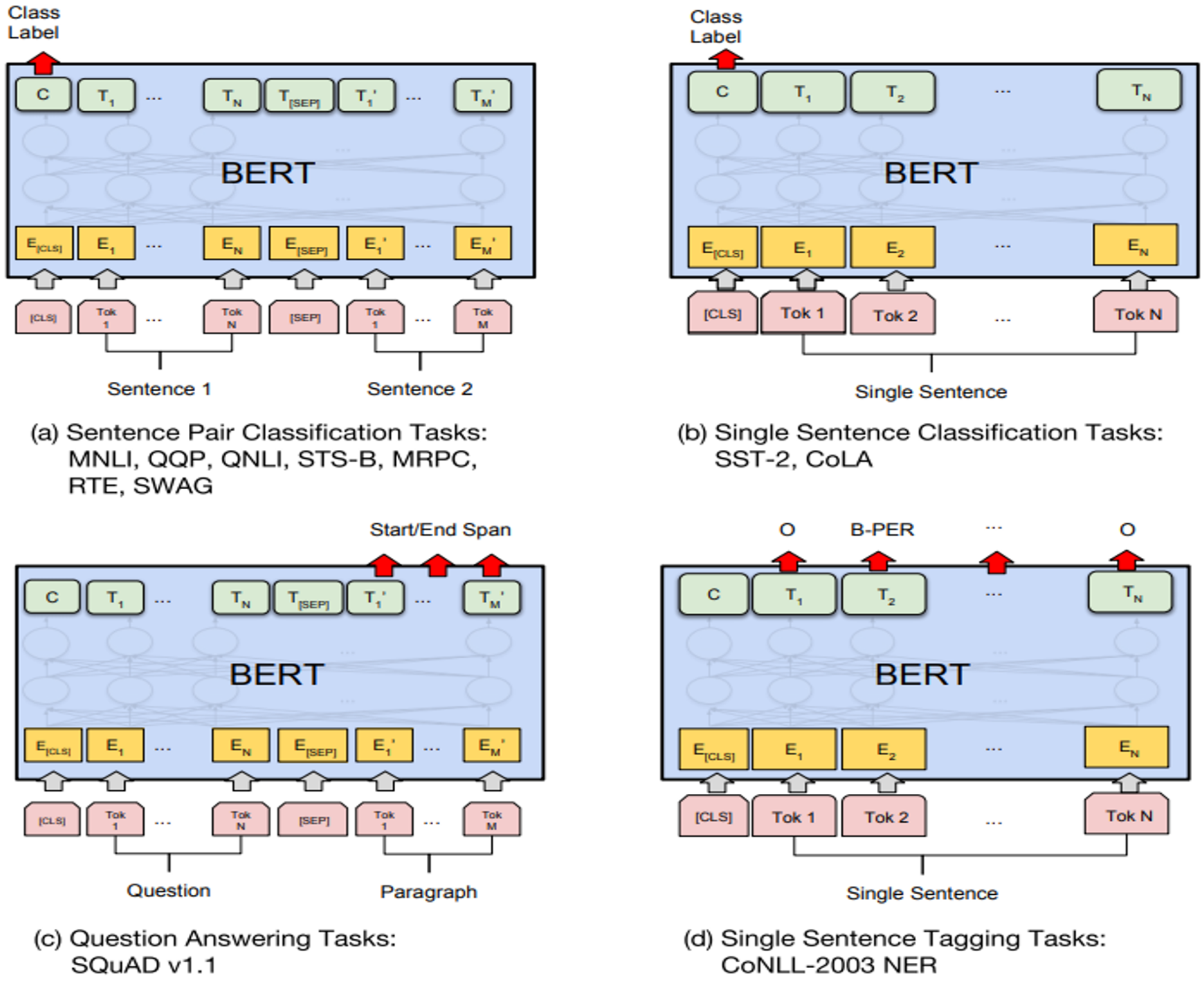
- For text pairs, independently encode text pairs before applying bi-directional cross attention.

- For each task, simply plug the task-specific inputs and outputs into BERT and fine-tune all parameters end-to-end.

- At input, sentence A and B from pretraining are analogous to 0 (1) sentence pairs in paraphrasing, (2) hypothesis-premise pairs in entailment, (3) question-passage pairs in question answering, and (4) a degenerate text-∅ pair in text classification or sequence tagging.

- At the output, token representations are fed into an output layer for token-level tasks, such as sequence tagging or QA, and [CLS] representation is used for classification such as entailment or sentiment analysis.

Fine-tuning BERT



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

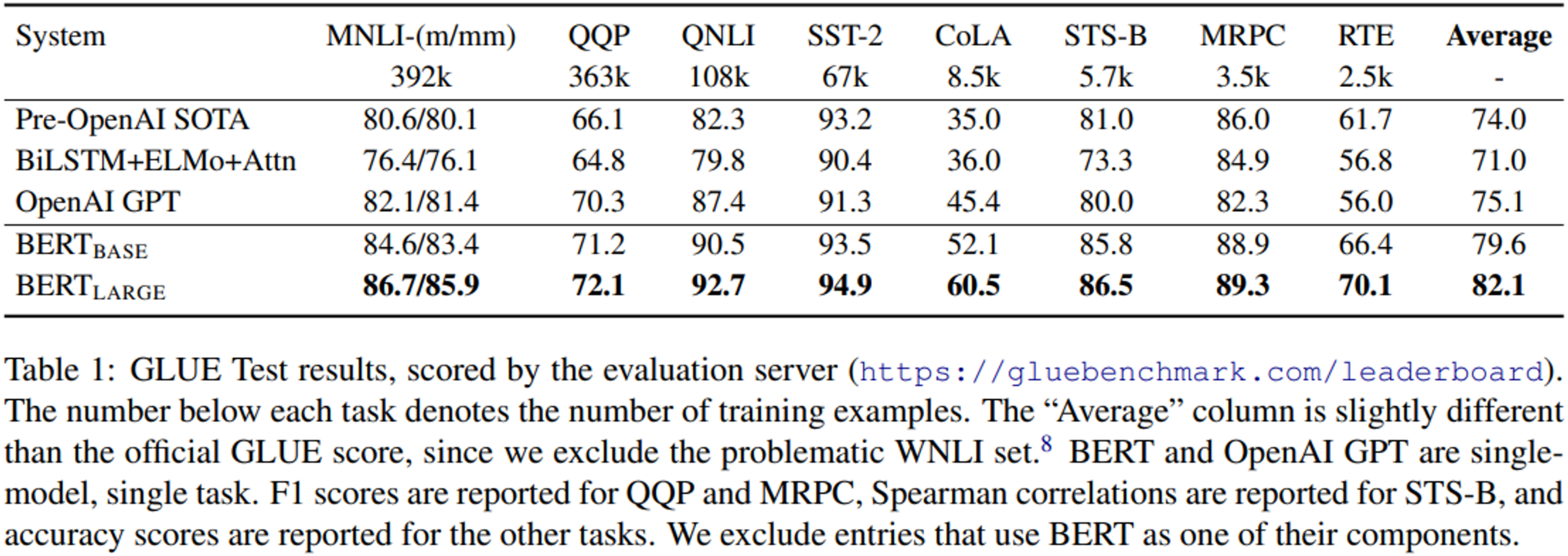
(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

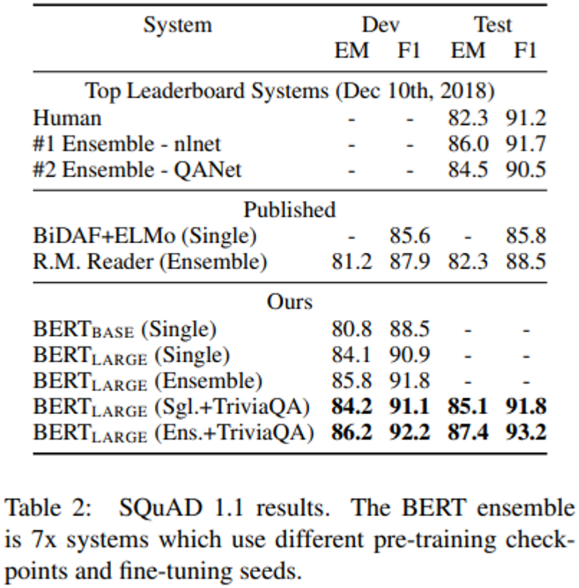
Experimental Results and Analysis



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

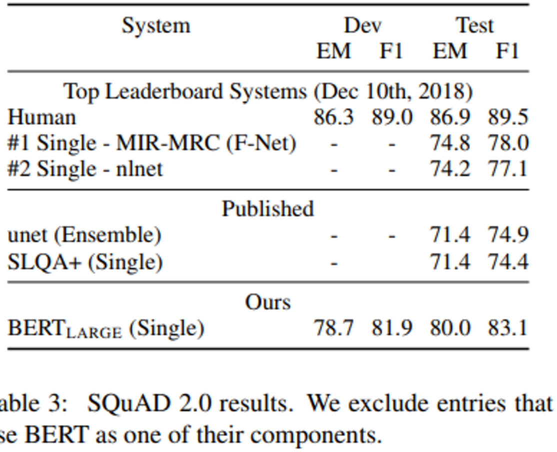
Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Experimental Results and Analysis



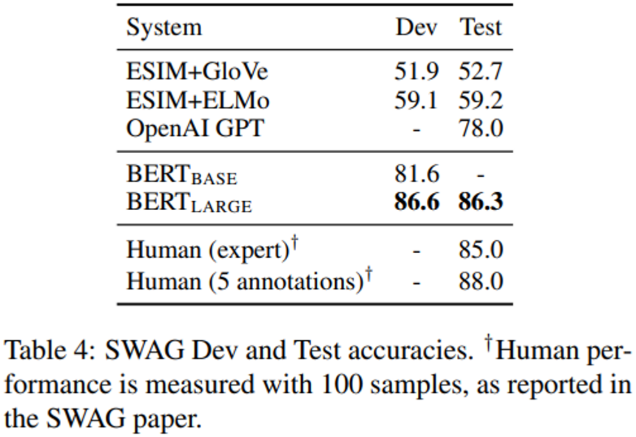
System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
BERT$_{\text{BASE}}$ (Single)	80.8	88.5	-	-
BERT$_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
BERT$_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
BERT$_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
BERT$_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.



System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
BERT$_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

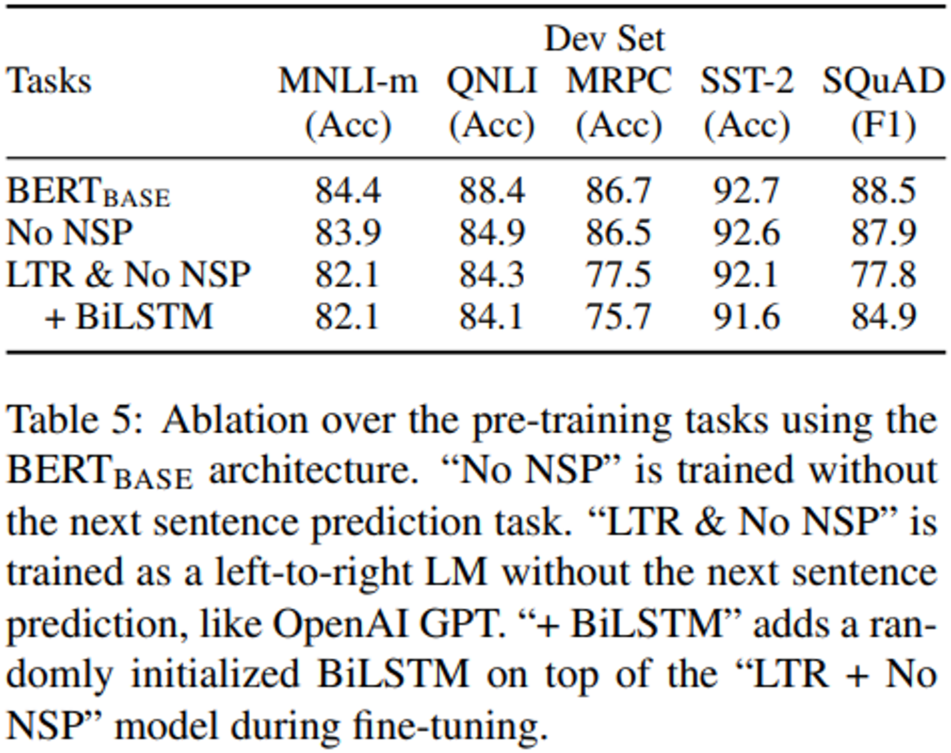


System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
BERT$_{\text{BASE}}$	81.6	-
BERT$_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. †Human performance is measured with 100 samples, as reported in the SWAG paper.

Ablation Studies

(Effect of Pre-training Tasks)

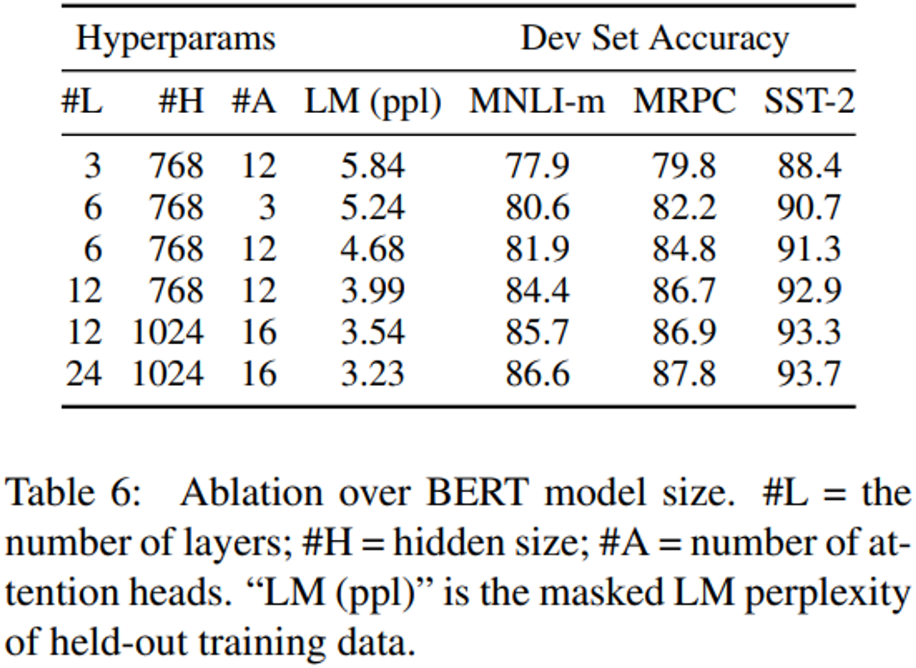


Tasks	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
	Dev Set				
BERT$_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the BERT$_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

- Removing NSP hurts performance significantly on QNLI, MNLI, and SQuAD 1.1.
- The LTR model performs worse than the MLM model on all tasks, with large drops on MRPC and SQuAD.
- Even after adding a random BiLSTM on top of LTR and No NSP, it hurts performance on the GLUE tasks.

Ablation Studies
(Effect of Model Size)



Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.

- It has long been known that increasing the model size will lead to continual improvements on large-scale tasks such as machine translation and language modeling

- This is the first work to demonstrate convincingly that scaling to extreme model sizes also leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained.

Conclusion and Future Work

Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems.

Major contribution is further generalizing these findings to deep bidirectional architectures, allowing the same pre-trained model to successfully tackle a broad set of NLP tasks.

References

- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In NAACL.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. **Attention is all you need.**
- 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.
- Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2018. Character-level language modeling with deeper self-attention. arXiv preprint arXiv:1808.04444.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th international conference on Machine learning, pages 160–167. ACM.
- William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005).
- Yacine Jernite, Samuel R. Bowman, and David Sontag. 2017. Discourse-based objectives for fast unsupervised sentence representation learning. CoRR, abs/1705.00557.
- Z. Chen, H. Zhang, X. Zhang, and L. Zhao. 2018. Quora question pairs.