BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

2019.07.26

김민수

Abstract

- BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers.

- BERT is designed to pretrain **deep bidirectional representations** from unlabeled text by jointly conditioning on **both left and right context** in all layers.

- The pre-trained BERT model can be **finetuned with just one additional output layer** to create state-of-the-art models for a wide range of tasks

- It obtains **new state-of-the-art results** on eleven natural language processing tasks

Introduction & Related Work

- Language model **pre-training** has been shown to be effective for improving many natural language processing tasks.
- There are two existing strategies for applying pre-trained language representations to downstream tasks: **feature-based** and **fine-tuning**.
- The **feature-based approach**, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features.
- The **fine-tuning approach**, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning all pretrained parameters.
- The two approaches share the same objective function during pre-training, where they use **unidirectional** language models to learn general language representations.

Introduction & Related Work

- Unlike left-to-right language model pre-training, the MLM objective enables the representation to fuse the left and the right context, which allows us to pretrain a **deep bidirectional Transformer.**

- BERT alleviates the previously mentioned unidirectionality constraint by using a **"masked language model" (MLM)** pre-training objective.

    - The masked language 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.

- In addition to the masked language model, we also use a **"next sentence prediction"** task that jointly pretrains text-pair representations.

Introduction & Related Work

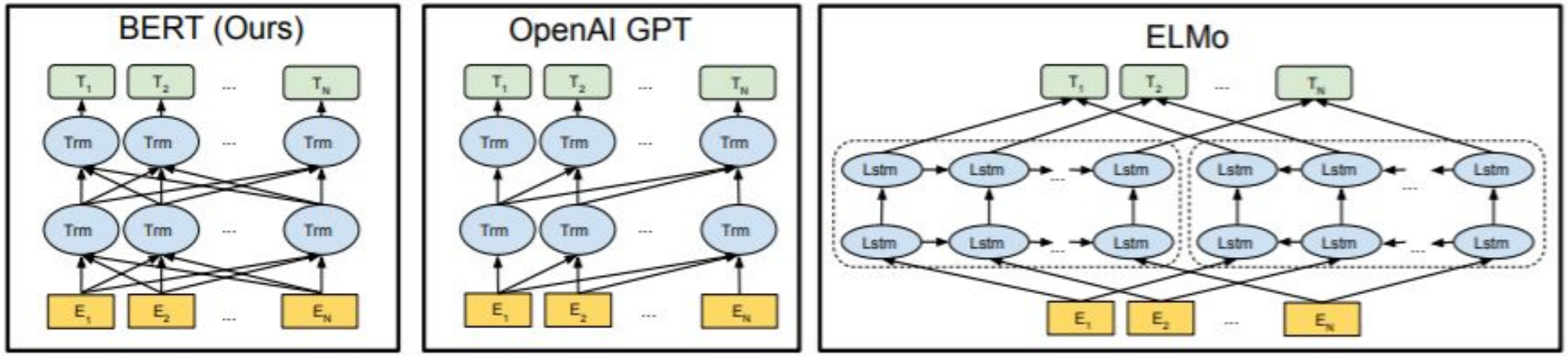


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT 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.

BERT

- Model Architecture
  - Multi-layer bidirectional Transformer encoder.
  - BERT Transformer uses **bidirectional** self-attention.
  - $\text{BERT}_{\text{BASE}}$ (L=12, H=768, A=12, Total Parameters=110M)

    $\text{BERT}_{\text{LARGE}}$ (L=24, H=1024, A=16, Total Parameters=340M).

* L : the number of transformer layers
H : hidden size
A : the number of self-attention heads

BERT

- Model Architecture

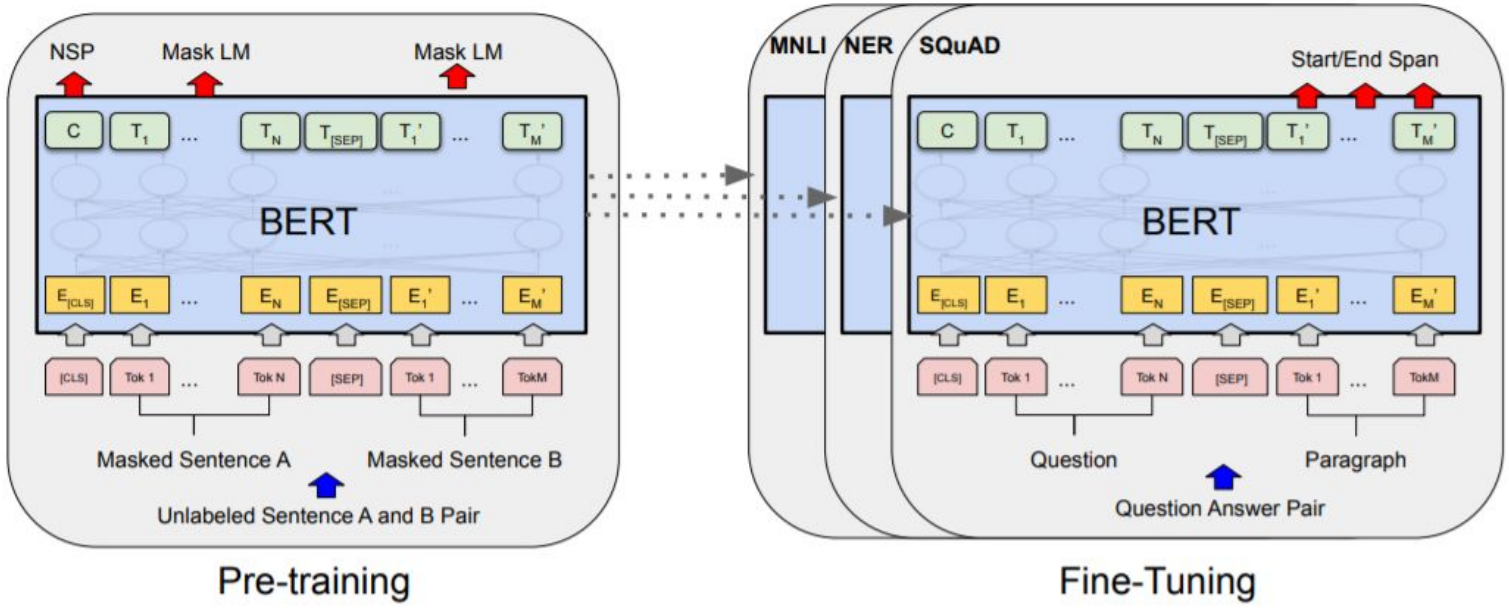


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are 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).

BERT

- Input/Output Representations
  - To make BERT handle a variety of down-stream tasks, our input representation is able to unambiguously represent **both a single sentence and a pair of sentences** (e.g., Question, Answer) in one token sequence.
  - "**sentence**" can be an arbitrary span of contiguous text, rather than an actual linguistic sentence.
  - "**sequence**" refers to the input token sequence to BERT, which may be a **single sentence or two sentences packed together.**
  - We use **WordPiece embeddings** with a **30,000** token vocabulary.

BERT

- Input/Output Representations
  - The **first token** of every sequence is always a special classification token (**[CLS]**).
    - The final hidden state corresponding to this token is used as the aggregate sequence representation **for classification tasks.**
  - We differentiate the sentences in two ways. First, we **separate** them with a special token (**[SEP]**). Second, we add a learned embedding to every token indicating whether it belongs to sentence A or sentence B.
  - For a given token, its input representation is constructed by **summing the corresponding token, segment, and position embeddings.**

BERT

- Input/Output Representations

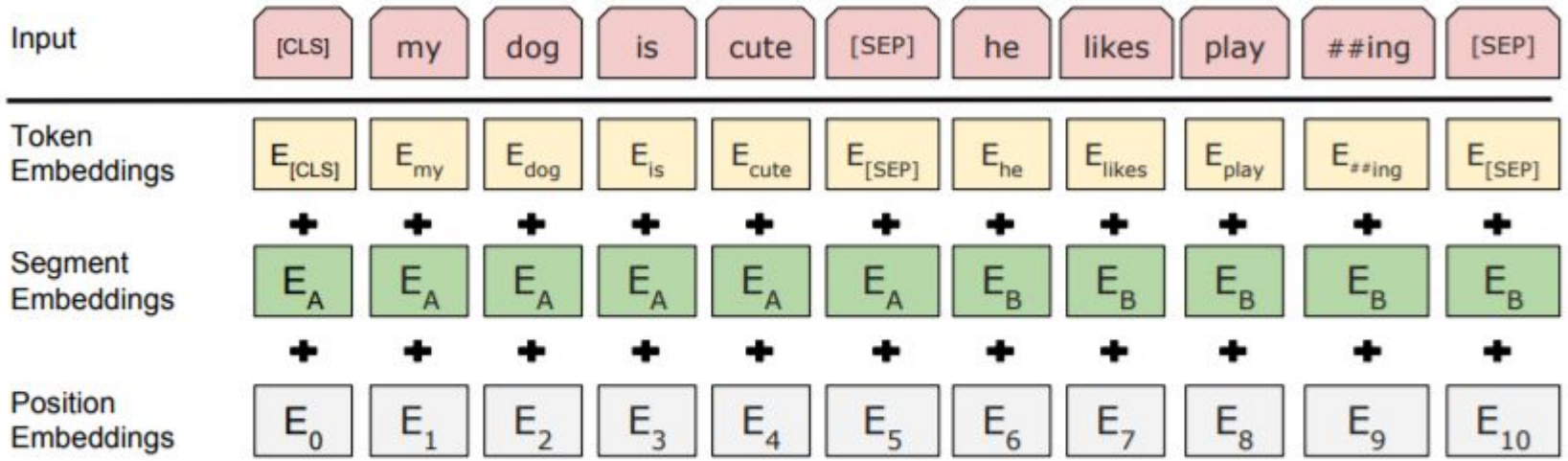


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT

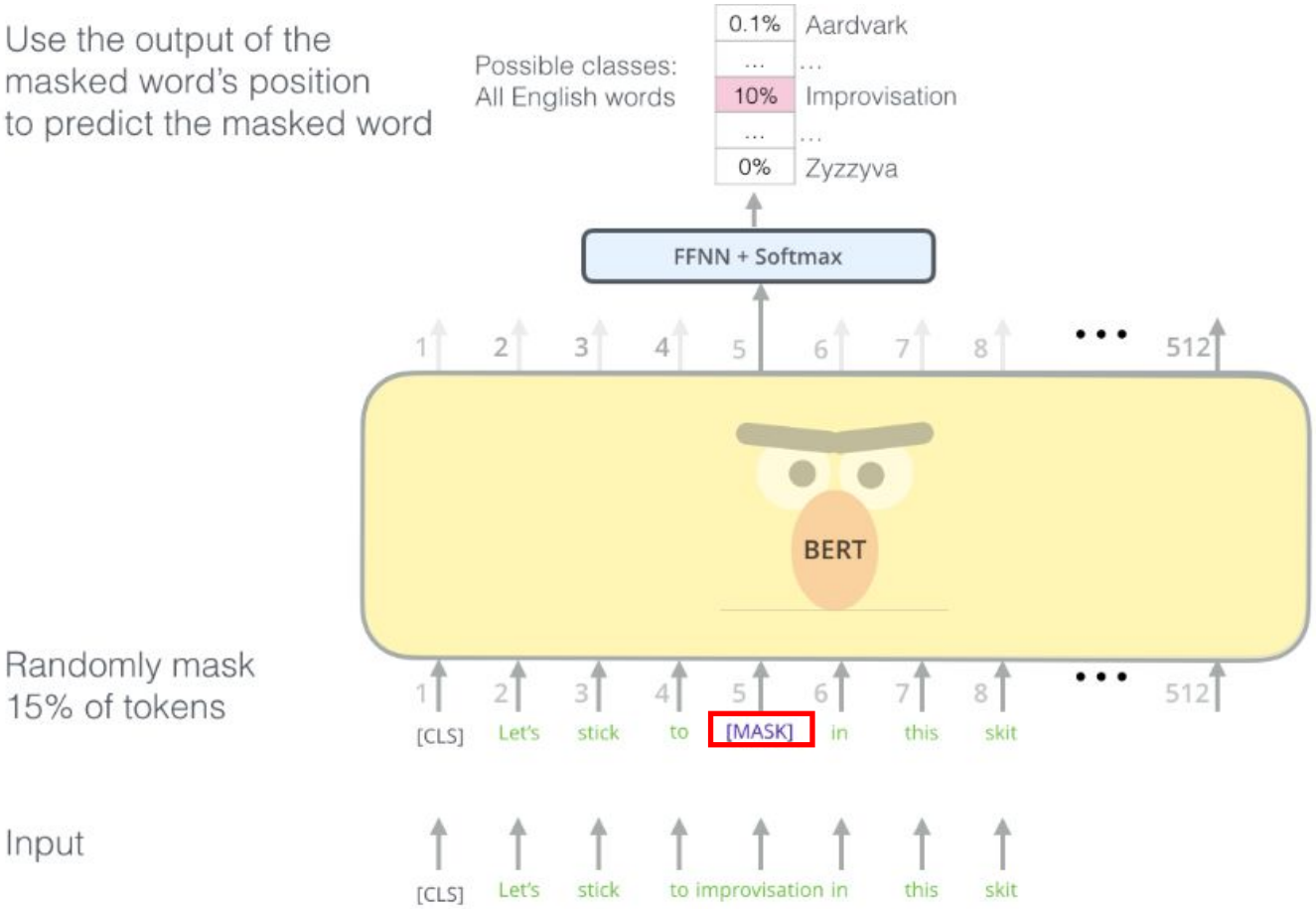
- Pre-training BERT
  - we do not **use traditional left-to-right or right-to-left** language models to pre-train BERT.
  - we pre-train BERT using two unsupervised tasks
    - Masked LM
    - Next Sentence Prediction (NSP)

BERT

- Pre-training BERT – Masked LM
  - we simply mask some percentage of the input tokens at random, and then predict those **masked tokens**.
  - The training data generator chooses **15%** of the token positions at random for prediction.
    - 80% of the time: Replace the word with the [MASK] token,
      ex) my dog is hairy
      → my dog is **[MASK]**
    - 10% of the time: Replace the word with a random word,
      ex) my dog is hairy
      → my dog is **apple**
    - 10% of the time: Keep the word unchanged,
      ex) my dog is hairy
      → my dog is **hairy.**

BERT

- Pre-training BERT – Masked LM



BERT's clever language modeling task masks 15% of words in the input and asks the model to predict the missing word.

BERT

- Pre-training BERT – Masked LM
  - The purpose of this is to bias the representation towards the actual observed word. The advantage of this procedure is that the Transformer encoder does not know which words it will be asked to predict or which have been replaced by random words, so it is forced to keep a **distributional contextual representation** of every input token.
  - because random replacement only occurs for **1.5%** of all tokens (i.e., 10% of 15%), this does not seem to harm the model's language understanding capability.

BERT

- Pre-training BERT - Next Sentence Prediction (NSP)
  - **Question Answering (QA) and Natural Language Inference (NLI)** are based on understanding the **relationship between two sentences.**
  - In order to train a model that understands sentence relationships, we pre-train for a binarized next sentence prediction task that can be trivially generated from any monolingual corpus.
  - when choosing the sentences A and B for each pretraining example, **50%** of the time B is the actual next sentence that follows A (**labeled as IsNext**), and **50%** of the time it is a random sentence from the corpus (**labeled as NotNext**).

Input = [CLS] the man went to [MASK] store [SEP]
he bought a gallon [MASK] milk [SEP]
Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
penguin [MASK] are flight ##less birds [SEP]
Label = NotNext

BERT

- Fine-tuning BERT
  - Our **task-specific** models are formed by incorporating BERT with one additional output layer, so a minimal number of parameters need to be learned from scratch.
  - the **token representations** are fed into an output layer for **tokenlevel tasks**, such as sequence tagging or question answering. the **[CLS] representation** is fed into an output layer for classification, such as entailment or sentiment analysis.

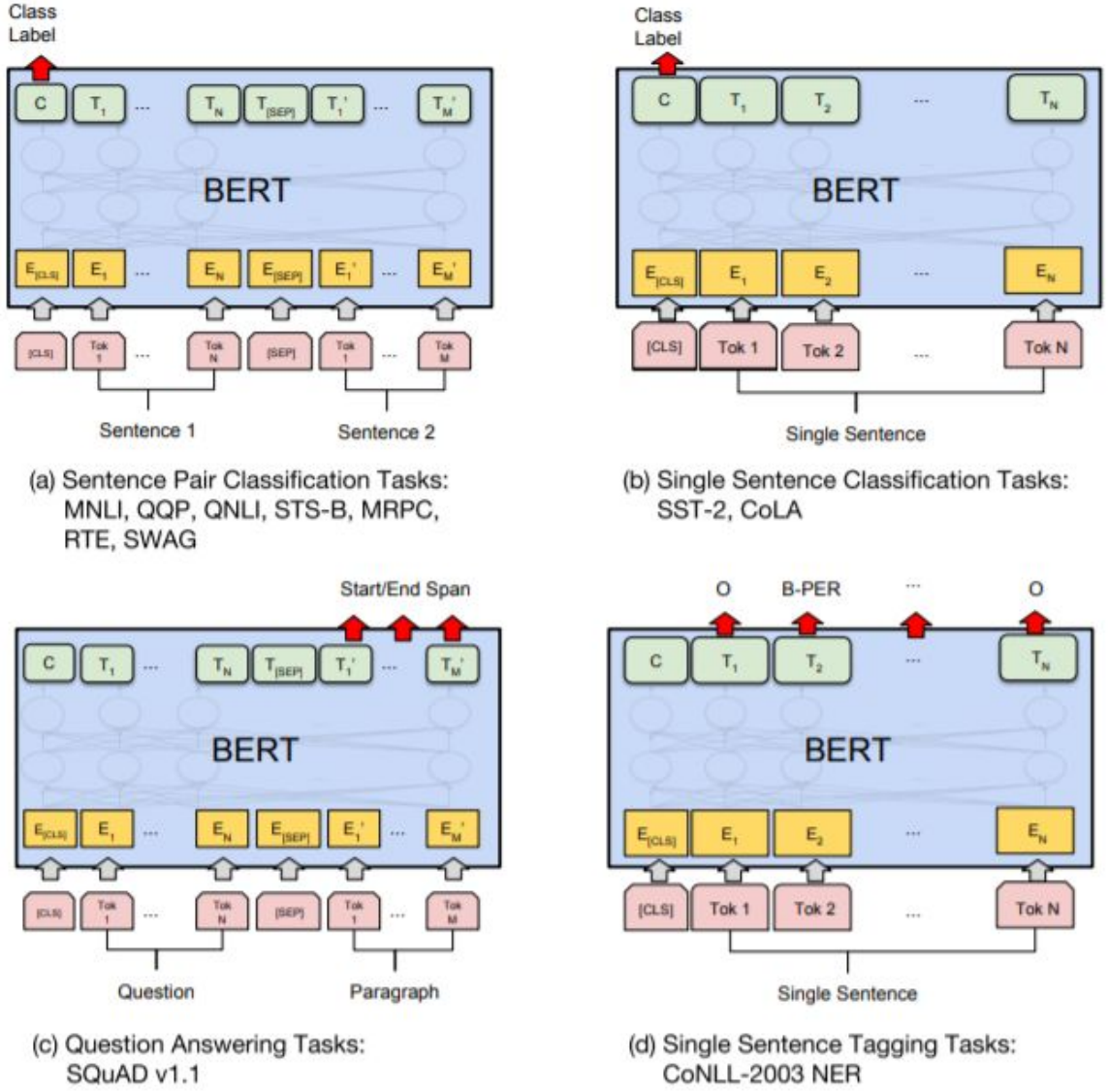


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

Experiments

- BERT fine-tuning results on 11 NLP tasks. (**SOTA**)

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
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.

Experiments

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				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (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				
$\text{BERT}_{\text{LARGE}}$ (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
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**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 Sudies

- Effect of Pre-training Tasks
  - No NSP
    - Using MLM without the NSP task
  - LTR & No NSP
    - Using a standard Left-to-Right(LTR) LM
    - Pre-trained without the NSP task

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	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 $\text{BERT}_{\text{BASE}}$ 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.

Ablation Sudies

- 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.

Ablation Sudies

- Feature-based Approach with BERT

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

Conclusion

- 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.

- In particular, these results enable even low-resource tasks to benefit from deep unidirectional architectures.

- Our 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.

Reference

- https://arxiv.org/pdf/1810.04805.pdf
- https://github.com/google-research/bert
- http://jalammar.github.io/illustrated-bert/
- https://medium.com/@_init_/why-bert-has-3-embedding-layers-and-their-implementation-details-9c261108e28a