Deep Learning 과정

Pretrained Language Models

Gunhee Kim

Computer Science and Engineering

서 울 대 학 교

SEOUL NATIONAL UNIVERSITY

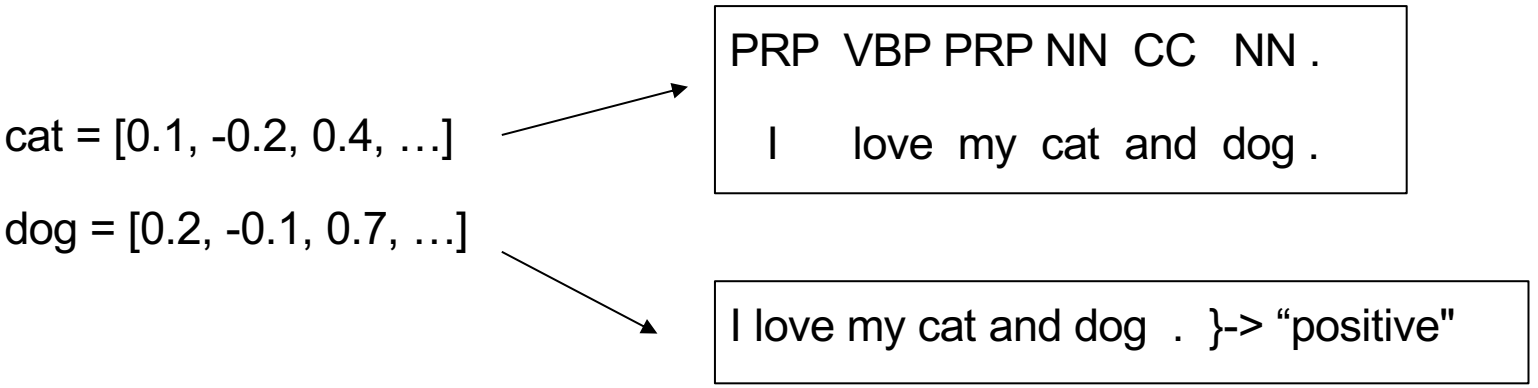
Outline

- Pretrained Language Models
- ELMO
- BERT
- RoBERTa
- ALBERTA

Conventional Text Representations

Word Representations

- Word embedding methods (e.g. word2vec) learn one vector per word



*Excerpt from NAACL 2019 Transfer Learning Tutorial

From Words to Words-in-Context

Word vectors	Sentence/document vectors	Word-in-context vectors
cats = [0.2, -0.3, …]	We have two cats. } [-1.2, 0.0, …]	[1.2, -0.3, …] ⏞ We have two cats.
dogs = [0.4, -0.5, …]	It's raining cats and dogs. } [0.8, 0.9, …]	[-0.4, 0.9, …] ⏞ It's raining cats and dogs.

*Excerpt from NAACL 2019 Transfer Learning Tutorial

How to Get *Contextualized* Word Representations?

Language Model (LM)

Probability distribution over strings of text

- How likely is a given string in a given “language”?
- Computed by factoring out the joint probability:

$$p(x_{1:T}) = p(x_1) \prod_{t=1}^{T} p(x_t | x_{1:t-1})$$

P(its, water, is, so, transparent)

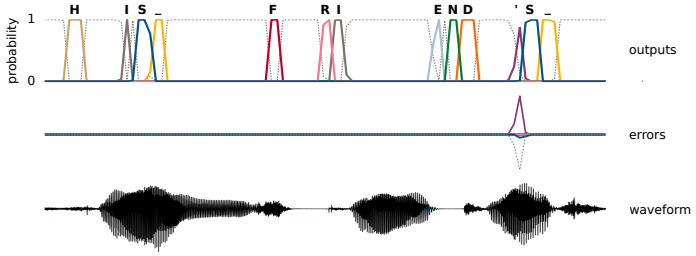
$= P$(its)$\times P$(water|its)$\times P$(is|its, water)$\times P$(so|its, water, is)

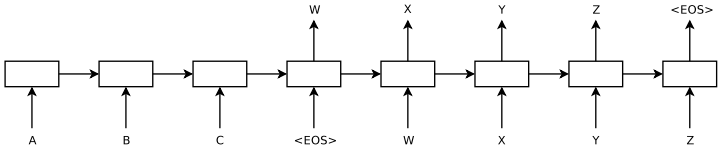
$\times P$(transparent|its, water, is, so)

Importance of LMs

Play a key role in several NLP tasks

- Each task-specific model explicitly or implicitly learns a language model for language understanding
- General performance boost for several tasks





Speech Recognition

P(I saw a van) >> P(eyes awe of an)

Machine Translation

P(**high** winds tonite) > P(**large** winds tonite)

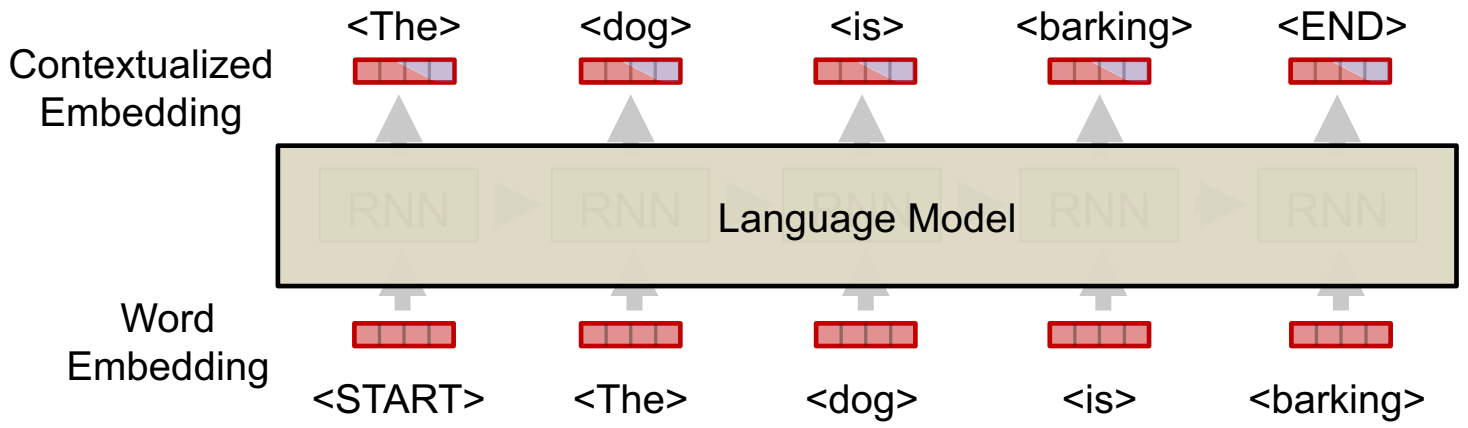
Image Credit: Graves and Jaitly. Towards End-to-End Speech Recognition with Recurrent Neural Networks. ICML 2013

Image Credit: Sutskever et al. Sequence to Sequence Learning with Neural Networks. NIPS 2014

Language Model Pretraining

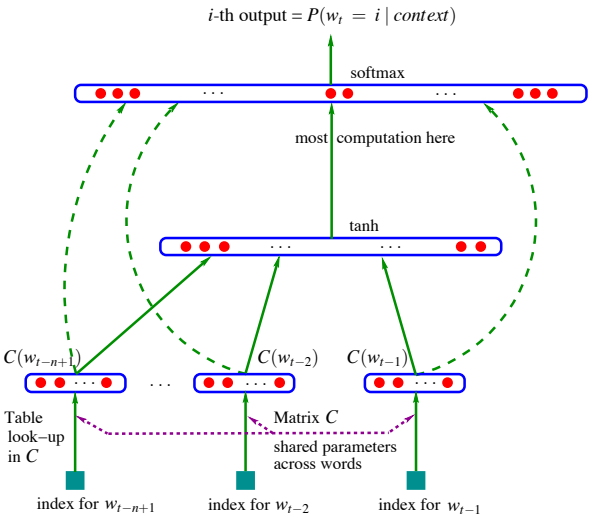
Pretrain deep neural networks for language modeling

- On large-scale corpus (e.g. 30M sentences)
- To improve downstream task performance
- Then? Use deep contextualized word representations from LM networks for each downstream task

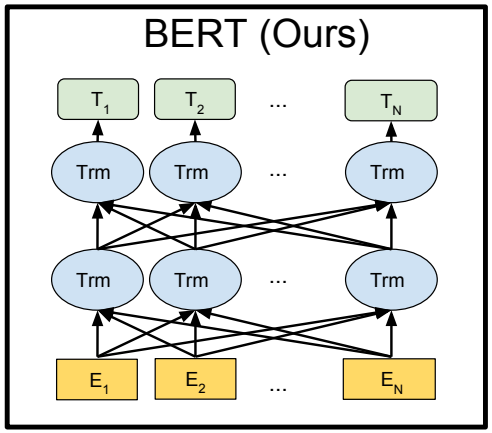


Language Model Pretraining

Pretrain DEEP neural networks for language modeling



1 Layer
[Bengio et al. 2003]



24 Layers
[Devlin et al. 2019]

*Excerpt from NAACL 2019 Transfer Learning Tutorial

Language Model Pretraining

Pretrain DEEP neural networks for language modeling

- Superior performance, even better than human

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jul 22, 2019	XLNet + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	**88.592**	**90.859**
2 Jul 26, 2019	UPM (ensemble) *Anonymous*	88.231	90.713
3 Aug 04, 2019	XLNet + SG-Net Verifier (ensemble) *Shanghai Jiao Tong University & CloudWalk* https://arxiv.org/abs/1908.05147	88.174	90.702
4 Aug 04, 2019	XLNet + SG-Net Verifier++ (single model) *Shanghai Jiao Tong University & CloudWalk* https://arxiv.org/abs/1908.05147	87.238	90.071
5 Jul 26, 2019	UPM (single model) *Anonymous*	87.193	89.934
6 Mar 20, 2019	BERT + DAE + AoA (ensemble) *Joint Laboratory of HIT and iFLYTEK Research*	87.147	89.474

SQuAD 2.0 Leaderboard

*Excerpt from NAACL 2019 Transfer Learning Tutorial

Outline

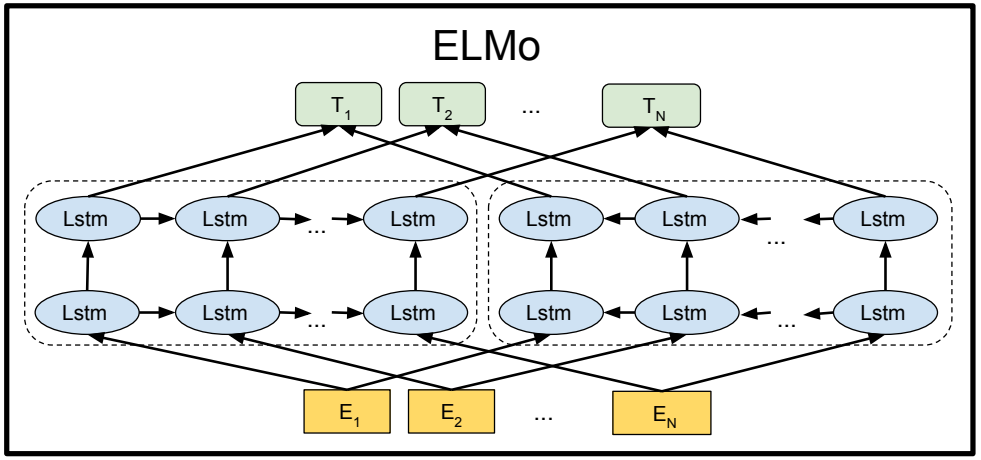
- Pretrained Language Models
- ELMO
- BERT
- RoBERTa
- ALBERTA

M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer. Deep Contextualized Word Representations. NAACL 2018.

ELMo: Embeddings from Language Models

Word Representations from RNN based Bidirectional language model

- Weighted average of all Bidirectional LM (BiLM) layers
- $\mathrm{T}_i = \sum_{j=0}^{L} \alpha_j \, [\overrightarrow{\mathrm{h}_{i,j}}; \overleftarrow{\mathrm{h}_{i,j}}]$

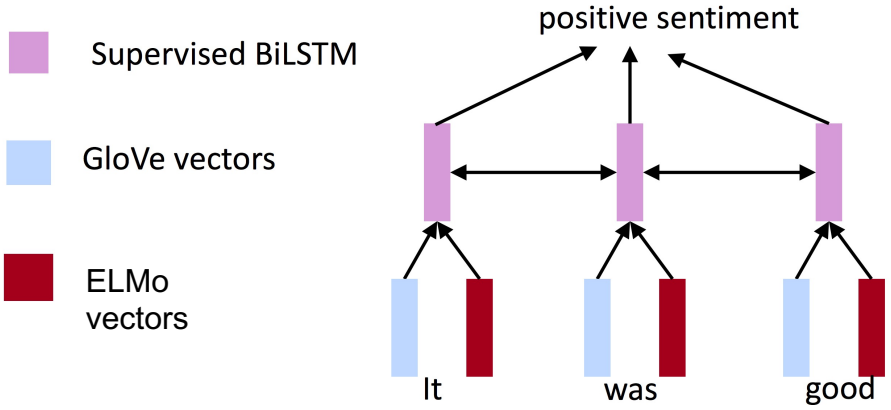


Peters et al. Deep Contextualized Word Representations. NAACL 2018
Image Credit: Delvin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019

ELMo: Embeddings from Language Models

How to use ELMo embeddings?

- Feed the concatenation of conventional word embeddings and ELMo embeddings into task-specific models



Sentimental analysis network with ELMo

ELMo: Embeddings from Language Models

Performance Boost on several tasks

Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

SQuAD: Stanford Question Answering Dataset
SNLI: Stanford Natural Language Inference corpus (textual entailment)
SRL: Semantic Role Labeling
Coref: Coreference resolution
NER: Named Entity Recognition
SST-5: Stanford Sentiment Tree-bank (sentiment analysis)

Outline

- Pretrained Language Models
- ELMO
- BERT
- RoBERTa
- ALBERTA

J. Devlin, M.W. Chang, K. Lee, K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.

Problem with RNN based LMs

Hard to parallelize efficiently

- Difficult to pretrain on large-scale corpus

Back propagation through sequence

- Difficult to optimize due to gradient vanishing/explosion, even with the gating mechanism (LSTM/GRU)

Transmitting local/global information through single vector (hidden state)

- Difficult to model long-term dependency
- Previous work has found that LSTM language models use 200 context words on average, and word order only has an effect within the last 50 words

Khandelwal et al. Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context. ACL 2018

Problem with RNN based LMs

Autoregressive (AR) language modeling

- Unidirectional: left-to-right or right-to-left
- Several real-word applications require bidirectional contexts
- Ex) Question: “Who is the singer of Radiohead?”, Context: Thom Yorke is the singer of Radiohead.”
- Restrict the power of the pre-trained representations

Advancement to Self-Attention

Replace RNN architectures with attention mechanics

- A self-attention mechanism directly models relationships between all words in a sentence, regardless of their respective position

I arrived at the bank after crossing the river

- For a representation of "*bank*", the attention scores are used as weights for a weighted average of all words' representations via a fully-connected network

Advancement to Self-Attention

Self-attention connects all positions with a constant number of sequentially executed operations

- Moreover, $n \ll d$ usually

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

The amount of computation that can be parallelized

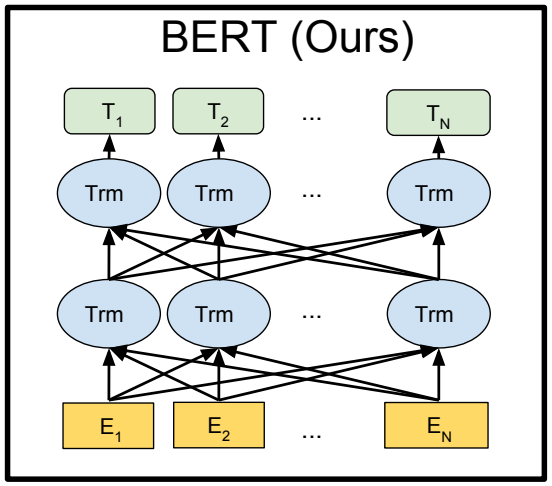
The maximum path length between any two input/output

Image Credit: Vaswani et al. Attention Is All You Need. NIPS 2017

Solution: BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers

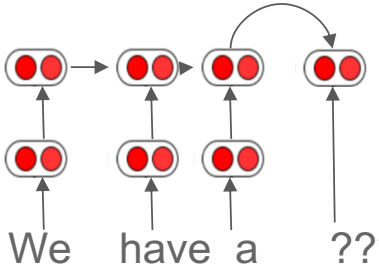
- Use Transformer as a backbone LM network
- Pretrain with two losses: (i) Masked LM and (ii) Next sentence prediction



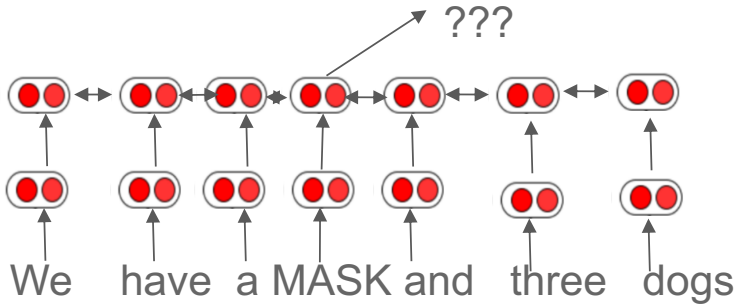
Solution: BERT

Comparison with ELMo

ELMo: Autoregressive LM



BERT: Masked LM



*Excerpt from NAACL 2019 Transfer Learning Tutorial

BERT – Pre-training

Masked LM: bidirectional conditioning

- Mask some percentage (15%) of the input tokens at random
- Predict the original value of the masked words, based on the context provided by the other, non-masked words in the sequence
- Mismatch between pre-training and fine-tuning (No masked tokens appearing during fine-tuning)
  - Solution: replace the chosen token with
    - (i) 80% of the time: the mask token
    - (ii) 10% of the time: a random token
    - (iii) 10% of the time: the original token (unchanged)

BERT – Pre-training

Next Sentence Prediction (NSP)

- The loss for the tasks that require understanding relationship between two sentences (e.g. QA, NLI)
- Pre-train for a *binarized NSP task*
  - (i) 50% of the time: B is the actual next sentence of A
  - (ii) 50% of the time: B is a random sentence

$$\text{Input} = \texttt{[CLS] the man went to [MASK] store [SEP]}$$
$$\texttt{he bought a gallon [MASK] milk [SEP]}$$
$$\text{Label} = \texttt{IsNext}$$

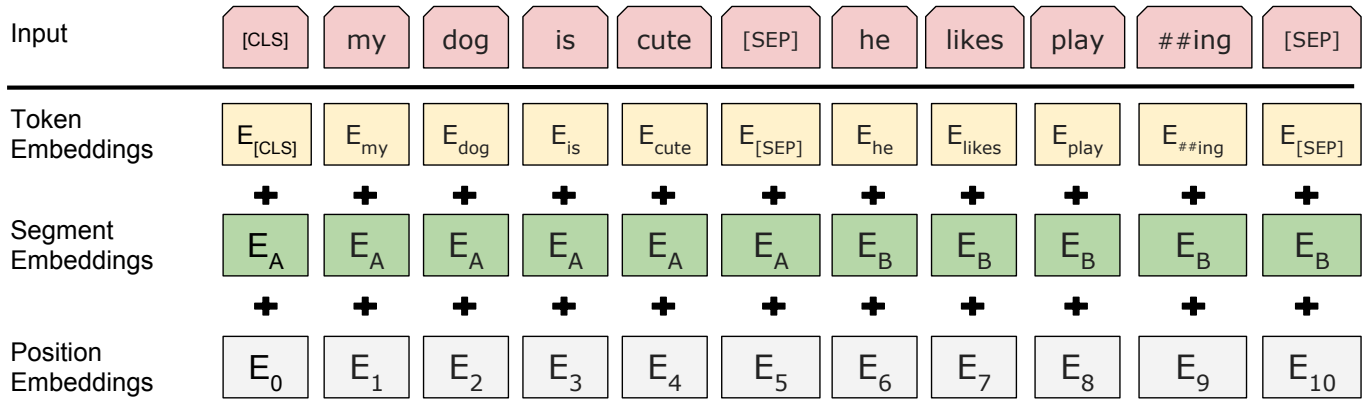
$$\text{Input} = \texttt{[CLS] the man [MASK] to the store [SEP]}$$
$$\texttt{penguin [MASK] are flight ##less birds [SEP]}$$
$$\text{Label} = \texttt{NotNext}$$

NSP Examples

BERT

BERT input representation is a sum of the following

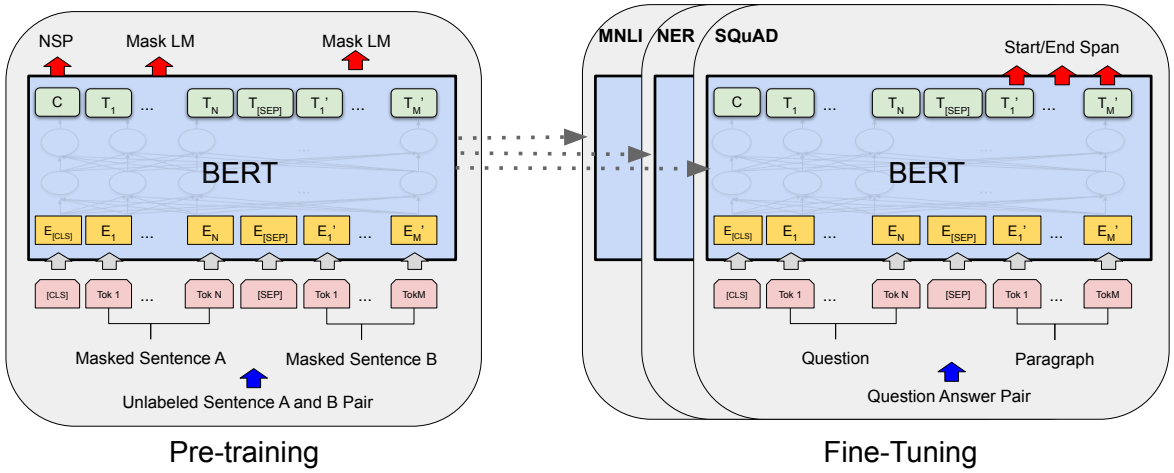
- (1) Token embedding (WordPiece with 30K vocabularies)
- (2) Segment embedding: distinguish between sentence A/B
- (3) (Absolute) Position embedding



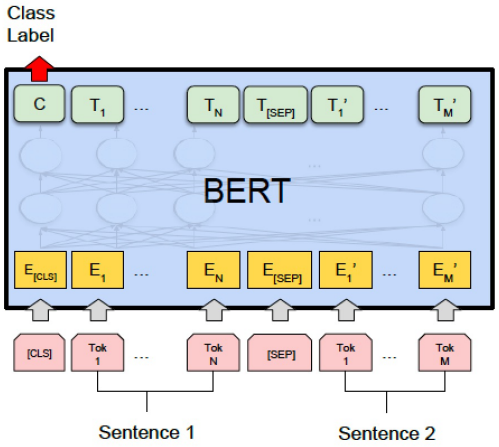
BERT

How to use BERT embeddings?

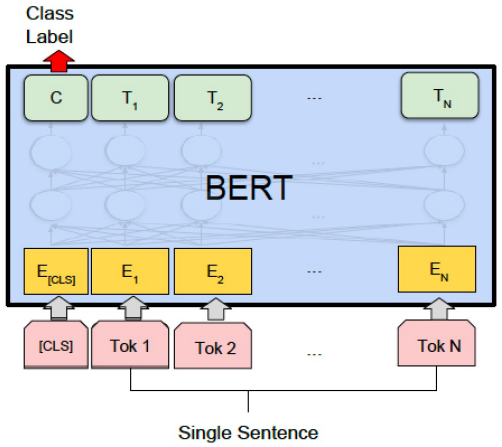
- Pretrain on the BooksCorpus (800M words) and English Wikipedia (2,500M words)
- Finetune for downstream tasks with few task-specific layers on top of BERT



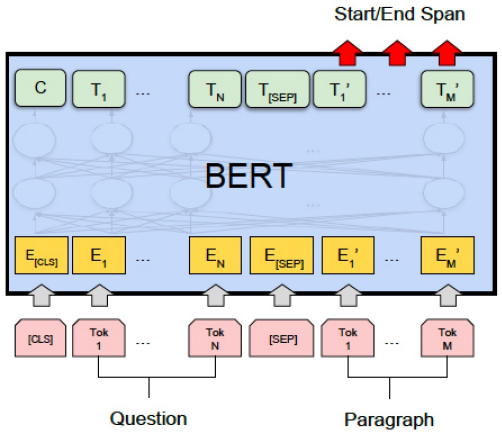
BERT – Fine-tuning



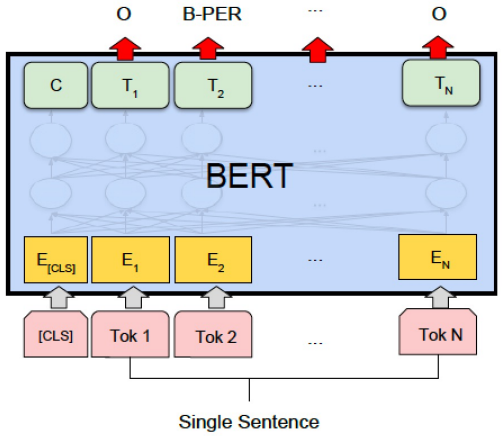
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



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



(c) Question Answering Tasks: SQuAD v1.1

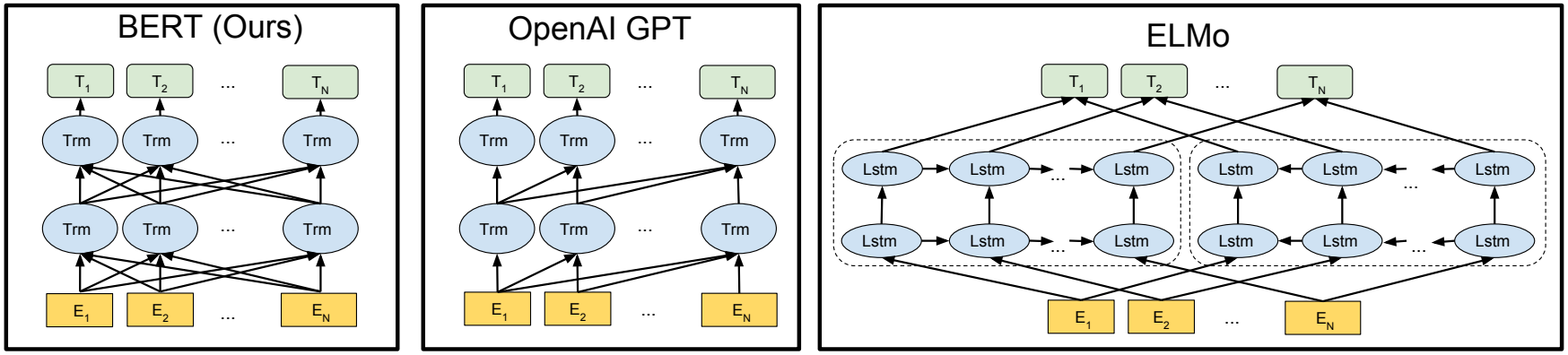


(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

BERT

Comparison with other pretraining LM architectures

- OpenAI GPT: Transformer with unidirectional (left-to-right) LM loss



Radford et al. Improving Language Understanding with Unsupervised Learning. Technical report, OpenAI, 2018

BERT

Performance boost on several tasks

- GLUE (General Language Understanding Evaluation): a collection of diverse natural language understanding tasks

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.

QOP (Quora Question Pairs, binary, two Questions are semantically equal or not)
QNLI (Question NLI, binary) SST (Stanford Sentiment treebank)
CoLA (Corpus of linguistic acceptability, binary classification, acceptable or not)
STS-B (Semantic Textural Similarity Benchmark) MRPC (MSR Paraphrase Copus)
RTE (Recognizing Textual Entailment)

BERT

Performance boost on several tasks

- SQuAD 1.1 / SQuAD 2.0

System	Dev		Test	
	EM	F1	EM	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		Test	
	EM	F1	EM	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.

BERT

Performance boost on several tasks

- SWAG (Situations with Adversarial Generations): given a sentence, choose the most plausible continuation among four choices

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.

BERT

Effect of the model size

- BERT is a really **BIG** model!!

Model	Layers (Transformer Blocks)	Hidden Size	Self-Attention Heads	Feed Forward/Filter Size	Total Parameters
BERT-Base	12	768	12	3072	110M
BERT-Large	24	1024	16	4096	340M

- Fine-tuned on a single Cloud TPU with 64GB of RAM
- Not reproducible from conventional GPUs with 12GB - 16GB of RAM (due to small batch sizes)

Outline

- Pretrained Language Models
- ELMO
- BERT
- RoBERTa
- ALBERTA

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. Arxiv 2019

RoBERTa

Robustly **o**ptimized **BERT** **a**pproach

- Proposed by Facebook
- Some minor technical updates with much more training!

1. Dynamic masking

- Static masking (BERT): The same mask for each training instance in every epoch
- Dynamic masking (proposed): 10 different masks per instance for 40 epochs (i.e. the same mask seen four times)

Masking	SQuAD 2.0	MNLI-m	SST-2
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

RoBERTa

2. Remove NSP loss

- Removing matches or slightly improves downstream task performance

3. Training with large batches

- Using a larger batch with less steps is better (with the same training budget)
- BERT: 256 bsz with 1M steps vs. RoBERTa: 8K with 31K

4. Text encoding

- A larger vocabulary for Byte-Pair Encoding (BPE)
- Use 50K sub-word units (vs 30K in BERT) with no preprocessing of the input

RoBERTa

10x more training time and data!

	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.	160 GB (16 GB BERT data + 144 GB 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

Suleiman Khan's BERT, RoBERTa, DistilBERT, XLNet — which one to use?

RoBERTa

Training data

- 16GB of Books Corpus and English Wikipedia (BERT) + CommonCrawl News dataset (63 million articles, 76 GB), Web text corpus (38 GB) and Stories from Common Crawl (31 GB)

GLUE benchmark results

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	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**

Outline

- Pretrained Language Models
- ELMO
- BERT
- RoBERTa
- ALBERTA

Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. ICLR 2020

ALBERT

A Lite **B**ERT

- Proposed by Google Research
- Two parameter-reduction techniques and a self-supervised loss
- Significantly fewer parameters: 1.27B (BERT x-large) vs 59 M (ALBERT x-large)

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
	xlarge	1270M	24	2048	2048	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

ALBERT

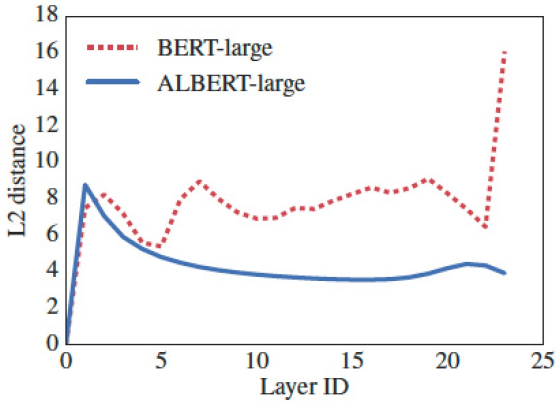
1. Factorization embedding parameterization

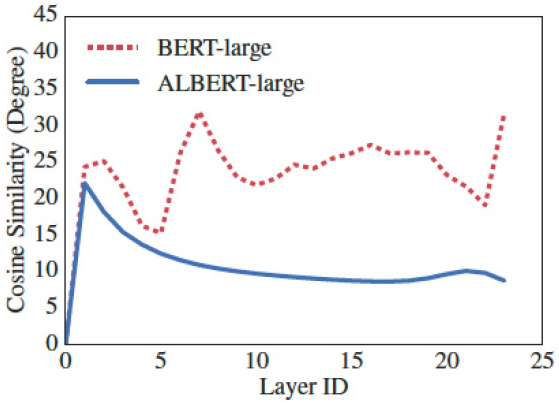
- WordPiece embeddings are meant to learn context-independent representations
- Hidden-layer embeddings: context-dependent representations. Thus, $H \gg E$ is reasonable
- BERT uses $E \equiv H$ (i.e. directly embed one-hot vector into H)
- ALBERT uses a much small $E \ll H$. It first projects one-hot vector into E and then project it again H
- Reduction from $O(V \times H)$ to $O(V \times E + E \times H)$
- Size notation: E: WordPiece embedding, H: hidden layer, V: vocabulary

ALBERT

2. Cross-layer parameter sharing

- In default, share all parameters across layers!
- Including network parameters + attention parameters
- The L2 distances and cosine similarity of the input and output embedding of each layer
- The transitions from layer to layer are much smoother for ALBERT than for BERT





ALBERT

3. Inter-sentence coherence loss

- Subsequent studies (e.g. RoBERTa, XLNet) ignore NSP
- Hypothesize NSP's ineffectiveness is lack of difficulty
- NSP conflates topic prediction and coherence prediction in a single task
- ALBERT uses a sentence-order prediction (SOP) loss to focus only on coherence
- Positive examples are the same with BERT (two consecutive segments from the same document)
- Negative examples the same two consecutive segments but with their order swapped

ALBERT

Results

- Less parameters, higher performance and faster training data

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	17.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	3.8x
	xlarge	1270M	86.4/78.1	75.5/72.6	81.6	90.7	54.3	76.6	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	21.1x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	6.5x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	2.4x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	1.2x

- See the papers about more experimental results
- e.g. Removing dropout, adding data improves performance

Summary

Pretrained Language Models (LMs) help improve downstream task performance

To fully model bidirectional context based representation, BERT introduces the masked LM loss, with the Next Sentence Prediction loss