ALBERT & ROBERTA

Google & Toyota Technical Institute at Chicago, Facebook

What are ALBERT and ROBERTA !!

ALBERT -> A Lite BERT architecture

ROBERTA -> Robustly optimized BERT approach

What is BERT

A large language model (proposed in 2018 Devlin et al.) that attained SOTA results on many core downstream nlp tasks.

Intuition -> if we can build a large language model that possess a vast amount of information about the language, we can do well on downstream tasks without spending a lot of time in fine tuning.

Built on the Transformer model proposed by Vaswani et al. 2017

What is BERT

BERT has two training objectives:

1. Predict the masked token
2. Given two sentences, predict whether the second sentence follows the first sentence (NSP)

ROBERTA

RoBERTa is trained with

- dynamic masking,
- FULL-SENTENCES without NSP loss,
- large mini-batches and
- a larger byte-level BPE

Also, experimented with training dataset size and number of training passes through the data

ROBERTA

Static vs. Dynamic Masking

The original BERT implementation performed masking once during data preprocessing, resulting in a single static mask. To avoid using the same mask for each training instance in every epoch, training data was duplicated 10 times so that each sequence is masked in 10 different ways over the 40 epochs of training.

We compare this strategy with dynamic masking where we generate the masking pattern every time we feed a sequence to the model.

ROBERTA(Static vs Dynamic Masking)

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

Table 1: Comparison between static and dynamic masking for $\text{BERT}_{\text{BASE}}$. We report F1 for SQuAD and accuracy for MNLI-m and SST-2. Reported results are medians over 5 random initializations (seeds). Reference results are from Yang et al. (2019).

ROBERTA (Experiment with NSP)

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
$\text{BERT}_{\text{BASE}}$	88.5/76.3	84.3	92.8	64.3
$\text{XLNet}_{\text{BASE}}$ (K = 7)	–/81.3	85.8	92.7	66.1
$\text{XLNet}_{\text{BASE}}$ (K = 6)	–/81.0	85.6	93.4	66.7

Table 2: Development set results for base models pretrained over BOOKCORPUS and WIKIPEDIA. All models are trained for 1M steps with a batch size of 256 sequences. We report F1 for SQuAD and accuracy for MNLI-m, SST-2 and RACE. Reported results are medians over five random initializations (seeds). Results for $\text{BERT}_{\text{BASE}}$ and $\text{XLNet}_{\text{BASE}}$ are from Yang et al. (2019).

ROBERTA (Experiment with Batch Size)

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

Table 3: Perplexity on held-out training data (*ppl*) and development set accuracy for base models trained over BOOKCORPUS and WIKIPEDIA with varying batch sizes (*bsz*). We tune the learning rate (*lr*) for each setting. Models make the same number of passes over the data (epochs) and have the same computational cost.

ROBERTA (Experiment with Text Encoding)

The original BERT implementation uses a character-level BPE vocabulary of size 30K, which is learned after preprocessing the input with heuristic tokenization rules.

ROBERTA trains BERT with a larger byte-level BPE vocabulary containing 50K subword units, without any additional preprocessing or tokenization of the input. This adds approximately 15M and 20M additional parameters for $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$, respectively.

ROBERTA (Effect of Training data and steps)

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

Table 4: Development set results for RoBERTa as we pretrain over more data (16GB → 160GB of text) and pretrain for longer (100K → 300K → 500K steps). Each row accumulates improvements from the rows above. RoBERTa matches the architecture and training objective of $\text{BERT}_{\text{LARGE}}$. Results for $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ are from Devlin et al. (2019) and Yang et al. (2019), respectively. Complete results on all GLUE tasks can be found in the Appendix.

ALBERT

Improves over BERT

- Parameter reduction Technique
- Sentence Order prediction task instead of NSP

Is having better NLP models as easy as having larger models ?

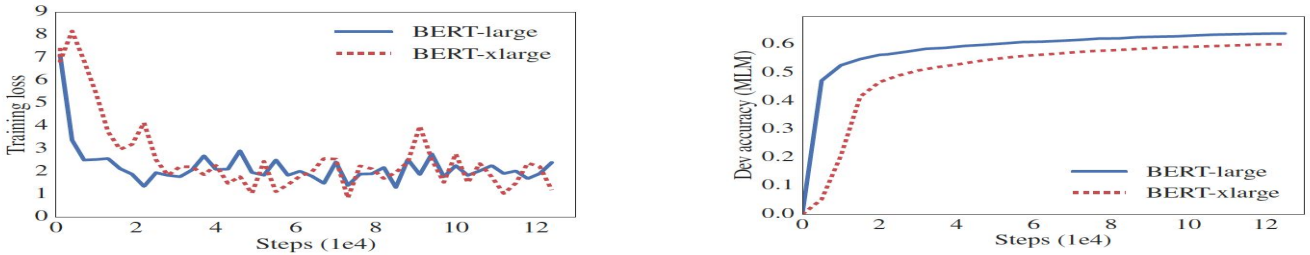


Figure 1: Training loss (left) and dev masked LM accuracy (right) of BERT-large and BERT-xlarge (2x larger than BERT-large in terms of hidden size). The larger model has lower masked LM accuracy while showing no obvious sign of over-fitting.

Model	Hidden Size	Parameters	RACE (Accuracy)
BERT-large (Devlin et al., 2019)	1024	334M	72.0%
BERT-large (ours)	1024	334M	73.9%
BERT-xlarge (ours)	2048	1270M	54.3%

Table 1: Increasing hidden size of BERT-large leads to worse performance on RACE.

BERT Issues

BERT -> using larger hidden size, more hidden layers, and more attention heads always leads to better performance. (hidden size 1024)
ALBERT -> under the same setting, increasing the hidden size to 2048 leads to model degradation and hence worse performance. Therefore, scaling up representation learning for natural language is not as easy as simply increasing model size.

In addition, it is difficult to experiment with large models due to computational constraints, especially in terms of GPU/TPU memory limitations.

MODEL ARCHITECTURE CHOICES of ALBERT

- Factorized embedding parameterization
- Cross-layer parameter sharing

Factorized Embedding Parameterization

WordPiece embedding size E is tied with the hidden layer size H , i.e., $E \equiv H$. This decision appears suboptimal for both modeling and practical reasons.

From a modeling perspective,WordPiece embeddings are meant to learn context independent representations, whereas hidden-layer embeddings are meant to learn context-dependent representations.

From a practical perspective, natural language processing usually require the vocabulary size V to be large. If $E \equiv H$, then increasing H increases the size of the embedding matrix, which has size $V * E$.

Factorized Embedding Parameterization

ALBERT uses a factorization of the embedding parameters, decomposing them into two smaller matrices.

Instead of projecting the one-hot vectors directly into the hidden space of size H,

- first project them into a lower dimensional embedding space of size E
- then project it to the hidden space.

By using this decomposition, the embedding parameters can be reduced from $O(V * H)$ to $O(V * E + E * H)$. This parameter reduction is significant when $H >> E$.

Cross-layer parameter sharing

BERT -> every layer has own Multi-head self attention mechanism and feed forward network

For ALBERT, cross-layer parameter sharing has been proposed as another way to improve parameter efficiency. There are multiple ways to share parameters, e.g., only sharing feed-forward network (FFN) parameters across layers, or only sharing attention parameters.
The default decision for ALBERT is to share all parameters across layers.

Cross-layer parameter sharing

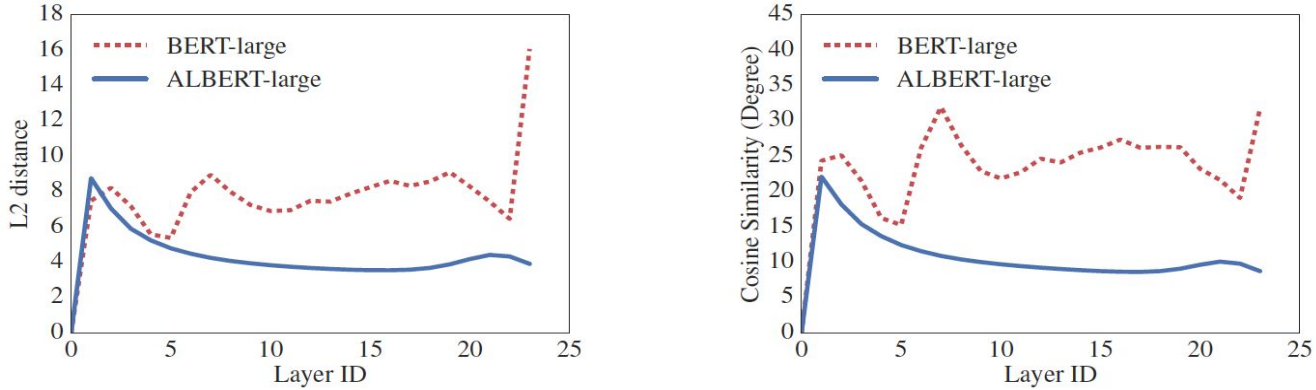


Figure 2: The L2 distances and cosine similarity (in terms of degree) of the input and output embedding of each layer for BERT-large and ALBERT-large.

Inter-sentence coherence loss (Issues with NSP)

BERT used next-sentence prediction (NSP).
NSP is a binary classification loss for predicting whether two segments appear consecutively in the original text, as follows: positive examples are created by taking consecutive segments from the training corpus; negative examples are created by pairing segments from different documents; positive and negative examples are sampled with equal probability.
NSP conflates topic prediction and coherence prediction in a single task. However, topic prediction is easier to learn compared to coherence prediction, and also overlaps more with what is learned using the MLM loss

Inter-sentence coherence loss

ALBERT, we use a sentence-order prediction (SOP) loss, which avoids topic prediction and instead focuses on modeling inter-sentence coherence.
The SOP loss uses as positive examples the same technique as BERT (two consecutive segments from the same document), and as negative examples the same two consecutive segments but with their order swapped.
This forces the model to learn finer-grained distinctions about discourse-level coherence properties.

Model Setup

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	59M	24	2048	128	True
	xxlarge	233M	12	4096	128	True

Table 2: The configurations of the main BERT and ALBERT models analyzed in this paper.

Experiments

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.5/83.3	80.3/77.3	84.1	91.7	68.3	82.1	17.7x
	large	334M	92.4/85.8	83.9/80.8	85.8	92.2	73.8	85.1	3.8x
	xlarge	1270M	86.3/77.9	73.8/70.5	80.5	87.8	39.7	76.7	1.0
ALBERT	base	12M	89.3/82.1	79.1/76.1	81.9	89.4	63.5	80.1	21.1x
	large	18M	90.9/84.1	82.1/79.0	83.8	90.6	68.4	82.4	6.5x
	xlarge	59M	93.0/86.5	85.9/83.1	85.4	91.9	73.9	85.5	2.4x
	xxlarge	233M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	1.2x

Table 3: Dev set results for models pretrained over BOOKCORPUS and Wikipedia for 125k steps. Here and everywhere else, the Avg column is computed by averaging the scores of the downstream tasks to its left (the two numbers of F1 and EM for each SQuAD are first averaged).

ALBERT - Ablation Study

FACTORIZED EMBEDDING PARAMETERIZATION

Model	E	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base not-shared	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base all-shared	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

Table 4: The effect of vocabulary embedding size on the performance of ALBERT-base.

ALBERT - Ablation Study

FACTORIZED EMBEDDING PARAMETERIZATION

- Under the non-shared condition (BERT-style), larger embedding sizes give better performance, but not by much.
- Under the all-shared condition (ALBERT-style), an embedding of size 128 appears to be the best.

ALBERT - Ablation Study

CROSS-LAYER PARAMETER SHARING

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base E=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base E=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

Table 5: The effect of cross-layer parameter-sharing strategies, ALBERT-base configuration.

ALBERT - Ablation Study

SOP task

	Intrinsic Tasks			Downstream Tasks					
SP tasks	MLM	NSP	SOP	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
None	54.9	52.4	53.3	88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0
NSP	54.5	90.5	52.0	88.4/81.5	77.2/74.6	81.6	**91.1**	62.3	79.2
SOP	54.0	78.9	86.5	**89.3/82.3**	**80.0/77.1**	**82.0**	90.3	**64.0**	**80.1**

Table 6: The effect of sentence-prediction loss, NSP vs. SOP, on intrinsic and downstream tasks.

Effect of Increasing Depth

Number of layers	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
1	18M	31.1/22.9	50.1/50.1	66.4	80.8	40.1	52.9
3	18M	79.8/69.7	64.4/61.7	77.7	86.7	54.0	71.2
6	18M	86.4/78.4	73.8/71.1	81.2	88.9	60.9	77.2
12	18M	89.8/83.3	80.7/77.9	83.3	91.7	66.7	81.5
24	18M	**90.3/83.3**	**81.8/79.0**	83.3	91.5	**68.7**	**82.1**
48	18M	90.0/83.1	**81.8/78.9**	**83.4**	**91.9**	66.9	81.8

Table 7: The effect of increasing the number of layers for an ALBERT-large configuration.

Effect of Increasing Width

Hidden size	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
1024	18M	79.8/69.7	64.4/61.7	77.7	86.7	54.0	71.2
2048	59M	83.3/74.1	69.1/66.6	79.7	88.6	58.2	74.6
4096	223M	**85.0/76.4**	**71.0/68.1**	**80.3**	**90.4**	**60.4**	**76.3**
6144	497M	84.7/75.8	67.8/65.4	78.1	89.1	56.0	74.0

Table 8: The effect of increasing the hidden-layer size for an ALBERT-large 3-layer configuration.

Effect of Training Hours

Models	Steps	Time	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
BERT-large	400k	34h	93.5/87.4	86.9/84.3	87.8	94.6	77.3	87.2
ALBERT-xxlarge	125k	32h	**94.0/88.1**	**88.3/85.3**	87.8	**95.4**	**82.5**	**88.7**

Table 9: The effect of controlling for training time, BERT-large vs ALBERT-xxlarge configurations.

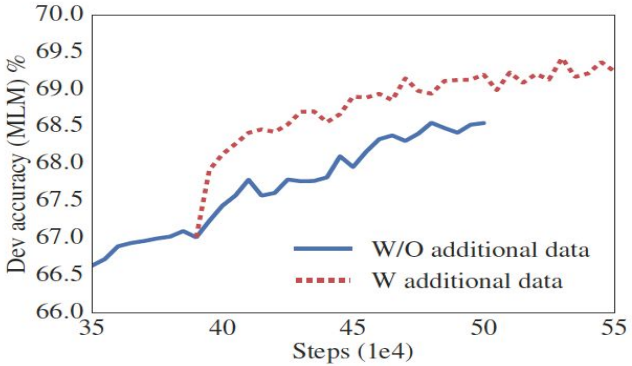
Effect of Varying both Depth and Width

ALBERT-large (H =1024), the difference between a 12-layer and a
24-layer configuration is small. Does this result still hold for much wider ALBERT configurations

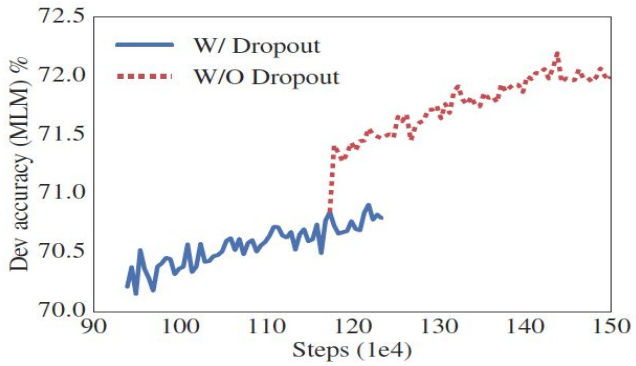
Number of layers	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
12	94.0/88.1	88.3/85.3	87.8	95.4	82.5	88.7
24	94.1/88.3	88.1/85.1	88.0	95.2	82.3	88.7

Table 10: The effect of a deeper network using an ALBERT-xxlarge configuration.

ADDITIONAL TRAINING DATA AND DROPOUT EFFECTS



(a) Adding data



(b) Removing dropout

Figure 3: The effects of adding data and removing dropout during training.

ADDITIONAL TRAINING DATA AND DROPOUT EFFECTS

	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
No additional data	**89.3/82.3**	**80.0/77.1**	81.6	90.3	64.0	80.1
With additional data	88.8/81.7	79.1/76.3	**82.4**	**92.8**	**66.0**	**80.8**

Table 11: The effect of additional training data using the ALBERT-base configuration.

	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
With dropout	94.7/89.2	89.6/86.9	90.0	96.3	85.7	90.4
Without dropout	**94.8/89.5**	**89.9/87.2**	**90.4**	**96.5**	**86.1**	**90.7**

Table 12: The effect of removing dropout, measured for an ALBERT-xxlarge configuration.

CURRENT STATE-OF-THE-ART ON NLU TASKS

The single-model ALBERT configuration incorporates the best-performing settings =>

ALBERT-xxlarge configuration (Table 2) using combined MLM and SOP losses, and no dropout.
The checkpoints that contribute to the final ensemble model are selected based on development set performance; the number of checkpoints considered for this selection range from 6 to 17, depending on the task. For the GLUE (Table 13) and RACE (Table 14) benchmarks, we average the model predictions for the ensemble models, where the candidates are fine-tuned from different training steps using the 12-layer and 24-layer architectures. For SQuAD (Table 14), we average the prediction scores for those spans that have multiple probabilities; we also average the scores of the “unanswerable” decision

CURRENT STATE-OF-THE-ART ON NLU TASKS

Models	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-large	90.2	94.7	**92.2**	86.6	96.4	**90.9**	68.0	92.4	-	-
ALBERT (1M)	90.4	95.2	92.0	88.1	96.8	90.2	68.7	92.7	-	-
ALBERT (1.5M)	**90.8**	**95.3**	**92.2**	**89.2**	**96.9**	**90.9**	**71.4**	**93.0**	-	-
Ensembles on test (from leaderboard as of Sept. 16, 2019)										
ALICE	88.2	95.7	**90.7**	83.5	95.2	92.6	**69.2**	91.1	80.8	87.0
MT-DNN	87.9	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2	98.6	90.3	86.3	96.8	93.0	67.8	91.6	90.4	88.4
RoBERTa	90.8	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0	88.5
Adv-RoBERTa	91.1	98.8	90.3	88.7	96.8	93.1	68.0	92.4	89.0	88.8
ALBERT	**91.3**	**99.2**	90.5	**89.2**	**97.1**	**93.4**	69.1	**92.5**	**91.8**	**89.4**

Table 13: State-of-the-art results on the GLUE benchmark. For single-task single-model results, we report ALBERT at 1M steps (comparable to RoBERTa) and at 1.5M steps. The ALBERT ensemble uses models trained with 1M, 1.5M, and other numbers of steps.

CURRENT STATE-OF-THE-ART ON NLU TASKS

Models	SQuAD1.1 dev	SQuAD2.0 dev	SQuAD2.0 test	RACE test (Middle/High)
Single model (from leaderboard as of Sept. 23, 2019)				
BERT-large	90.9/84.1	81.8/79.0	89.1/86.3	72.0 (79.6/70.1)
XLNet	94.5/89.0	88.8/86.1	89.1/86.3	81.8 (85.5/80.2)
RoBERTa	94.6/88.9	89.4/86.5	89.8/86.8	83.2 (86.5/81.3)
UPM	-	-	89.9/87.2	-
XLNet + SG-Net Verifier++	-	-	90.1/87.2	-
ALBERT (1M)	94.8/89.2	89.9/87.2	-	86.0 (88.2/85.1)
ALBERT (1.5M)	**94.8/89.3**	**90.2/87.4**	**90.9/88.1**	**86.5 (89.0/85.5)**
Ensembles (from leaderboard as of Sept. 23, 2019)				
BERT-large	92.2/86.2	-	-	-
XLNet + SG-Net Verifier	-	-	90.7/88.2	-
UPM	-	-	90.7/88.2	
XLNet + DAAF + Verifier	-	-	90.9/88.6	-
DCMI+	-	-	-	84.1 (88.5/82.3)
ALBERT	**95.5/90.1**	**91.4/88.9**	**92.2/89.7**	**89.4 (91.2/88.6)**

Table 14: State-of-the-art results on the SQuAD and RACE benchmarks.

Datasets

GLUE GLUE is comprised of 9 tasks, namely Corpus of Linguistic Acceptability (CoLA; Warstadt et al., 2018), Stanford Sentiment Treebank (SST; Socher et al., 2013), Microsoft Research Paraphrase Corpus (MRPC; Dolan & Brockett, 2005), Semantic Textual Similarity Benchmark (STS; Cer et al., 2017), Quora Question Pairs (QQP; Iyer et al., 2017), Multi-Genre NLI (MNLI; Williams et al., 2018), Question NLI (QNLI; Rajpurkar et al., 2016), Recognizing Textual Entailment (RTE; Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009) and Winograd NLI (WNLI; Levesque et al., 2012). It focuses on evaluating model capabilities for natural language understanding. When reporting MNLI results, we only report the "match" condition (MNLI-m). We follow the finetuning procedures from prior work (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019) and report the held-out test set performance obtained from GLUE submissions. For test set submissions, we perform task-specific modifications for WNLI and QNLI as described by Liu et al. (2019) and Yang et al. (2019).

Datasets

SQuAD SQuAD is an extractive question answering dataset built from Wikipedia. The answers are segments from the context paragraphs and the task is to predict answer spans. We evaluate our models on two versions of SQuAD: v1.1 and v2.0. SQuAD v1.1 has 100,000 human-annotated question/answer pairs. SQuAD v2.0 additionally introduced 50,000 unanswerable questions. For SQuAD v1.1, we use the same training procedure as BERT, whereas for SQuAD v2.0, models are jointly trained with a span extraction loss and an additional classifier for predicting answerability (Yang et al., 2019; Liu et al., 2019). We report both development set and test set performance.

Datasets

RACE RACE is a large-scale dataset for multi-choice reading comprehension, collected from English examinations in China with nearly 100,000 questions. Each instance in RACE has 4 candidate answers. Following prior work (Yang et al., 2019; Liu et al., 2019), we use the concatenation of the passage, question, and each candidate answer as the input to models. Then, we use the representations from the "[CLS]" token for predicting the probability of each answer. The dataset consists of two domains: middle school and high school. We train our models on both domains and report accuracies on both the development set and test set.