ALBERT paper review

Daumsoft AI Lab
KwangJune Choi

ALBERT

A Lite BERT For Self-Supervised Learning of Language Representations

preview

- BERT의 pre-trained language representation 모델의 크기를 증가시켜 성능을 개선할 시 발생하는 문제를 개선하고 성능을 높여보자

ALBERT

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Apr 06, 2020	SA-Net on Albert (ensemble) *QIANXIN*	**90.724**	**93.011**
2 Apr 05, 2020	Retro-Reader (ensemble) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694	90.578	92.978
3 Mar 12, 2020	ALBERT + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	90.386	92.777
4 Jan 10, 2020	Retro-Reader on ALBERT (ensemble) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694	90.115	92.580
5 Nov 06, 2019	ALBERT + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	90.002	92.425
6 Sep 18, 2019	ALBERT (ensemble model) *Google Research & TTIC* https://arxiv.org/abs/1909.11942	89.731	92.215
6 Feb 25, 2020	Albert_Verifier_AA_Net (ensemble) *QIANXIN*	89.743	92.180
7 Mar 28, 2020	Retro-Reader on ELECTRA (single model) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694	89.562	92.052
7 Mar 27, 2020	albert+KD+transfer (ensemble) *Anonymous*	89.461	92.134
8 Apr 08, 2020	ALBert-LSTM (ensemble) *oppo.tensorlab*	89.224	91.853
8 Apr 21, 2020	albert+KD+transfer+twopass (single) *SPPD*	89.111	91.877
8 Apr 18, 2020	ALBERT + MTDA + SFVerifier (ensemble model) *Senseforth AI Research* https://www.senseforth.ai/	89.235	91.739
9 Apr 15, 2020	ALBERT + SFVerifier (ensemble model) *Senseforth AI Research* https://www.senseforth.ai/	89.133	91.666
9 Apr 23, 2020	ELECTRA+RL+EV (single model) *Hithink RoyalFlush*	89.021	91.765
10 Dec 08, 2019	ALBERT+Entailment DA (ensemble) *CloudWalk*	88.761	91.745
10 Apr 14, 2020	SA-Net on Electra (single model) *QIANXIN*	88.851	91.486
11 Mar 06, 2020	ELECTRA (single model) *Google Brain & Stanford*	88.716	91.365
12 Feb 24, 2020	ALBERT (Single model) *SRCB_DML*	88.592	91.286
12 Feb 20, 2020	Tuned ALBERT (ensemble model) *Group Data & Analytics Cell | Aditya Birla Group)* https://www.adityabirla.com/About/group-data-and-analytics	88.637	91.230
12 Jan 19, 2020	Retro-Reader on ALBERT (single model) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694	88.107	91.419
12 Jul 22, 2019	XLNet + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	88.592	90.859
12 Apr 14, 2020	albert+KD+transfer (single) *HIT master*	88.298	91.078

ALBERT

기존 BERT 모델 크기 증가로 성능 개선시 발생하는 문제

1. memory limitation
2. training time
3. model degradation

ALBERT

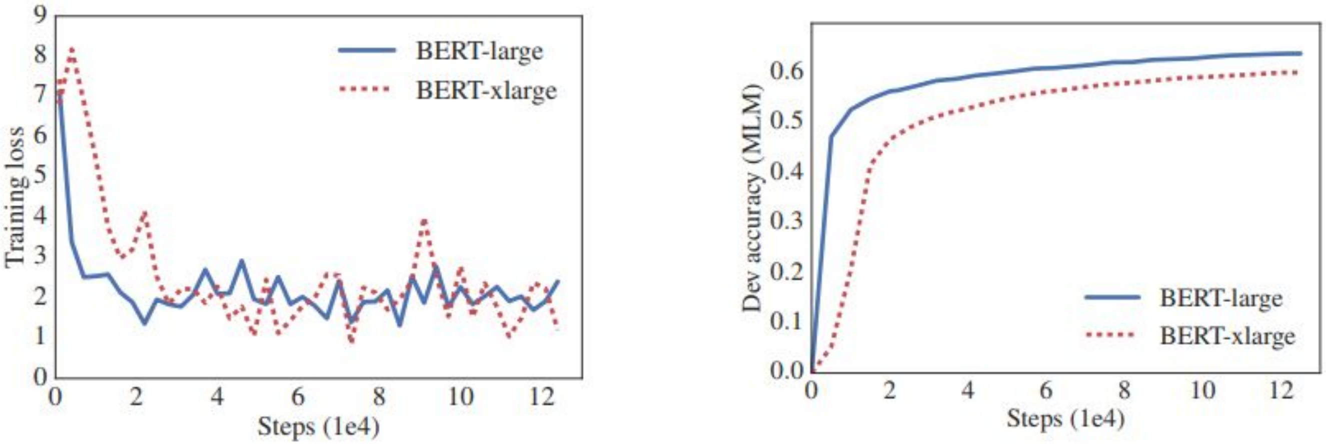

Figure 1: Training loss (left) and dev masked LM accuracy (right) of BERT-large and BERT-xlarge (2x larger than BERT-large in terms of hidden size). The larger model has lower masked LM accuracy while showing no obvious sign of over-fitting.

BERT-xlarge는 BERT-large에 비해 dev acc가 오히려 떨어질 뿐더러 학습 과정에서 overfitting으로 의심되는 뚜렷한 정황도 없음

RACE 테스트에서도 BERT-xlarge가 BERT-large보다 성능이 떨어지는 것이 확인됨

ALBERT

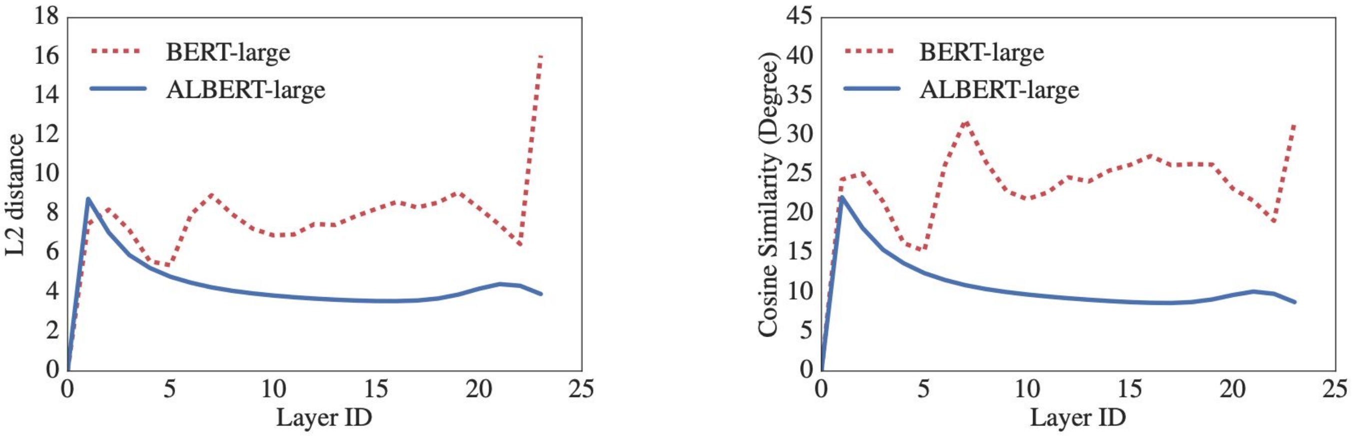

Figure 1: The L2 distances and cosine similarity (in terms of degree) of the input and output embedding of each layer for BERT-large and ALBERT-large.

- BERT-large와 ALBERT-large의 각 layer input & output embedding 사이의 L2 distance(왼쪽)와 Cosine Similariry(오른쪽)를 측정
- BERT-large의 경우 수렴할 수록 내려가야하는데 오히려 진동하는 형태가 확인, 이는 BERT가 불안정하게 학습한다는 것

ALBERT

목표

- 모델 크기를 줄이고 성능을 개선해보자
    1. Factorized Embedding Parameterization
    2. Cross-layer Parameter Sharing
    3. Sentence-Order Prediction (Inter-Sentence Coherence Loss)

ALBERT

Factorized Embedding Parameterization

- Token embedding size(E)를 hidden layer embedding size(H)보다 작게 설정하여 parameter 수를 줄임
- 기존 BERT, XLNet, RoBert 등 모두 E=H로 모델 구성
- H는 contextualized representation이지만 E는 context independent representation이므로 E < H 이어도 모델이 성능이 떨어지지 않을 것이라는 가정

ALBERT

Factorized Embedding Parameterization

- BERT에서는 Vocabulary size(V) x H를 활용하여 token embedding
- V x E matrix로 Token embedding 후 E x H matrix를 활용하여 최종 input 생성
    - O(VxH) -> O(VxE + ExH)
    - 일반적인 BERT에서는 V가 30,000이라 매우 크므로 이러한 방식으로 parameter size를 줄임

ALBERT

Cross-layer Parameter Sharing

- Layer 간 같은 parameter를 사용(recursive transformation)
    - attention, FFN 등 의 모든 parameter를 layer 간 공유
    - layer 수가 늘어나더라도 parameter수가 늘지않음

(유사연구 _ Universal Transformer, Deep Equilibrium models)

ALBERT

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

각 layer의 feed forward network를 공유하지 않을 때 더 좋은 성능이 나타남

attention만 sharing 했을 때는 not-shared와 성능차이가 크지않음

ALBERT

Inter-sentence coherence loss

- **next sentence prediction**(NSP) 대신 2개의 sentence의 order가 맞는지 여부를 예측하는 방식으로 학습 (**sentence order prediction**, SOP)
- SOP를 예측하는 것이 inter-sentence coherence를 학습하는데 더 적합
- NSP는 문장의 순서와 상관없이 유사한 주제를 다루기만 한다면 next sentence로 예측할 수 있음 -> topic prediction으로 볼 수 있음
- SOP로 예측 시 같은 topic이라도 문장 간 순서를 고려하므로 <u>discourse level coherence를 반영</u>한다고 볼 수 있음

ALBERT

Inter-sentence coherence loss

<Next Sentence Prediction> (BERT)

- QA나 NLI와 같은 NLP에서 중요한 downstream task들은 두 문장 사이의 relationship을 이해하는 것에 기반을 둠
- 두 문장 사이의 relationship은 일반적인 language model로는 직관적으로 포착되지않음
- 이를 위해 단일 언어 corpus에서 생성된 binarized next sentence prediction task를 pre-train

ALBERT

Inter-sentence coherence loss

‹Next Sentence Prediction› (BERT)

- 각 step 별 50% 확률로 B는 A의 실제 다음 문장 IsNext라고 라벨링,
  그리고 50% 확률로 B를 corpus의 임의의 문장으로 추출하여 NotNext로 라벨링

vector C는 다음 sentence prediction을 위해 사용됨 (NSP에서 학습됐기 때문에 fine-tuning이 없으면 의미있는 문장 표현을 내포하지 못함)

ALBERT

Inter-sentence coherence loss

- Sentence Order prediction으로 예측 시 같은 topic이라도 문장 간 순서를 고려하므로 discourse level coherence를 반영한다고 볼 수 있음

- positive pair를 뽑는 것은 BERT와 같으나 negative pair의 경우 연속된 문장을 뽑되 순서를 바꾸어서 배치

ALBERT

SP tasks	Intrinsic Tasks MLM	NSP	SOP	Downstream Tasks 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 5: The effect of sentence-prediction loss, NSP vs. SOP, on intrinsic and downstream tasks.

NSP 학습 시 SOP를 거의 하지 못함

- NSP는 너무 난이도가 낮고 downstream tasks에서 최선의 선택이 아님
- SOP는 NSP에 비해 난이도가 높고 성능 개선에 효과적

ALBERT

Experiment

- book corpus, wikipedia -16GB (same as BERT)
- batch size = 4,096
- 125,000 steps training
- TPU v3 (model의 크기에 따라 64-1024 chips 활용)

ALBERT

Experiment

- Dataset
  - GLUE 9 tasks
  - SQuAD 1.1, 2.0
  - RACE

ALBERT

Experiment

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

layer, depth에 따른 성능 차이 비교시 12 layer이상부터는 큰 차이가 없었음

ALBERT

Experiment

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	60M	83.3/74.1	69.1/66.6	79.7	88.6	58.2	74.6
4096	225M	**85.0/76.4**	**71.0/68.1**	**80.3**	**90.4**	**60.4**	**76.3**
6144	499M	84.7/75.8	67.8/65.4	78.1	89.1	56.0	74.0

Table 12: The effect of increasing the hidden-layer size for an ALBERT-large 3-layer configuration.

hidden size를 증가시켰을 시 오히려 성능이 떨어짐

ALBERT

Experiment

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	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

Table 1: The configurations of the main BERT and ALBERT models analyzed in this paper.

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	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

xxlarge ALBERT는 6-layer로 미리 학 습 한 후 12 layer로 추가 training -> converge가 더 잘됨

ALBERT

Experiment

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

유사한 학습 시간으로 학습 시 성능 차이 비교

ALBERT

Experiment

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 (76.6/70.1)
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	-
DCMN+	-	-	-	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 10: State-of-the-art results on the SQuAD and RACE benchmarks.

Appendix

dates	model	authors
2018/02	ELMo	Allen AI & UW
2018/05	GPT-1	OpenAI
2018/10	BERT	Google
2019/07	XLNet	CMU & Google Brain
2019/07	RoBERTa	FAIR
2019/09	ALBERT	Google & TTIC
2019/10	T5	Google

Thank You

contact : +82-10-5500-7977

Daumsoft AILab

kwmme797@gmail.com
https://github.com/kwangjunechoi7