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

00 Abstract

BERT : Bidirectional Encoder Representation

- **pre-train**
- **bidirectional** : 문맥의 양방향(왼/오) 모두 고려
- 하나의 output layer 추가로 **fine-tuning** 가능

GLUE, SQuAD v1.1/ v2.9 에서 높은 성능을 보임

01 Introduction

Language model pre-training 은 다음 과제에 효과적이다

- sentence level
  - natural language inference
  - paraphrasing
- token level
  - named entity recognition
  - question answering

applying pre-trained model - 접근법

- **feature-based approach** : 선행 학습된 모델의 출력을 추가하여, 특정 작업에 적합한 모델 구조를 사용하는 것. ELMo

  (ELMo는 왼, 오 각 방향의 문맥을 각각 독립적으로 학습하는 반면 BERT는 한번에 양방향 문맥을 학습한다)

- **fine-tuning approach** : 선행 학습된 모델 자체의 파라미터를 추가하여 사용하는 방법. GPT

⇒ 두 접근법 모두 선행 학습 과정에 unidirectional 방식을 사용하고 이것은 선행 학습 구조 선택에 제한이 됨

예를 들어 GPT(left to right, unidirectional)의 경우, 이전 토큰에 대해서 self-attention : 이는 양방향 문맥을 고려하지 못해 question answering 문제 해결에 해가 될 수 있다

BERT → Bidirectional을 위해 **MLM** + **NSP** 사용(사전 학습)

- **MLM (masked language model)** : 무작위로 단어를 마스킹하고, 모델이 문맥을 파악하여 마스킹 된 부분의 본래 단어를 예측하게 하는 방식으로 학습
- **NSP (next sentence prediction)** : 두 문장 간의 관계를 학습. 두 문장의 연속성 예측

02 related work

: pre-training general language representations

02.1 Unsupervised Feature-based Approaches

Word Representation 학습

- non-neural : clustering 등
- neural : 신경망 사용. word2vec, GLoVe 등

Pre-trained Word Embedding : NLP system 성능 향상에 큰 도움을 준다.

- 초기 방식은 left-to-right 방식을 사용함
- 단어를 벡터로 표현

Sentence/Paragraph Embedding 까지 확장

문장 표현의 학습 위해 사용되는 목표 objectives 들

- rank candidate next sentence : 다음에 올 문장 순위 매기기

- left-to-right generation of next sentence : 이전 문장을 바탕으로 다음에 올 문장 생성
- denoising autoencoder : 인위적으로 노이즈를 추가 - 복원 과정 학습

ELMo : context-sensitive feature 추출

- left-to-right & right-to-left → concatenate
- 문맥을 고려하고자 하는 시도

Learning Contextual Representation

- LSTM 사용하여 양방향 문맥 파악 - 하나의 단어 예측

⇒ 두 모델 모두 feature-based 였으며, not deeply bidirectional

02.2 Unsupervised Fine-tuning Approaches

마찬가지로 word-embedding 부터 시작\

최근의 Sentence or Document Encoder

- 더 문맥에 맞는 토큰을 생성
- 비지도 학습으로 선행 학습 후, task에 맞게 supervised fine-tuning
    - 선행 학습에 사용된 objective
    - left-to-right langauge modeling : 이전 단어 참고하여 다음 단어 예측
    - auto-encoder objectives

⇒ 작업에 맞춰 쉽게, 적은 파라미터를 새로 학습하며 재사용에 용이

GPT의 성공 이유가 여기에..

(GLUE 벤치마킹 - NLP 모델 평가 테스트에서 최고의 성과)

02.3 Transfer Learning from Supervised Data

Transfer Learning - Supervised Task, Large Dataset을 다룰 때 효과적

선행 학습된 모델을 전이 학습

CV 분야에서 입증됨

03 BERT

- **Pre-training ← unlabeled data**
- **Fine-tuning ← labeled data**

하나의 통합된 모델을 task 별로 미세 조정(fine-tuning) 하여 사용

Model Architecture

기존 Transformer 논문의 인코더 구조를 거의 변화 없이 사용

L : # of layers

H : the hidden size

A : # of self-attention heads

BERTBASE (L=12, H=768, A=12, Total Parameters=110M)

- OpenAI GPT와의 성능 비교 위해 같은 크기로 설정
- OpenAI GPT는 단방향 - 순차적 생성 작업에 유리한 반면(left-to-right 만 고려), 모든 방향을 고려하는 BERT는 문맥 이해에 유리하다.

BERTLARGE (L=24, H=1024,A=16, Total Parameters=340M)

Input/Output Representation

하나 또는 여러 개의 문장으로 구성된 sequence 입력

token → WordPiece Embedding

- [CLS] : 시퀀스의 첫 번째 토큰, 시퀀스 구분

하나의 시퀀스 내 문장 구분

1. [SEP] : 문장이 끝날 때마다 입력
2. 어떤 문장에 속하는지에 대한 learned embedding (segment embedding)

⇒ 두 방식을 모두 사용

각 계산으로 만들어진 Token/Segment/Position Embedding을 모두 합하여 input representation 구성

03.1 Pre-training BERT

기존 방식(left-to-right || right-to-left) 사용하지 않는다 → 두 가지 unsupervised task로 해결

Masked LM (MLM)

기존 양방향 모델의 문제점 - See itself?

: 일반적인 조건부 언어 모델 - 단어 예측 시, 예측할 단어 외의 단어만 참조해야 한다. 하지만 multi-layer, 양방향 모델에서 한번 예측해 만들어낸 단어를 다른 레이어가 알게 되며 무의미한 예측을 수행하는 '참조' 문제가 생길 수 있다.

BERT에서는,

입력의 랜덤 일부를 mask 하고 이를 예측하는 방식으로 모델을 학습

pre-training, fine-tuning 간의 불일치 ← fine-tuning에는 [MASK] 토큰이 나타나지 않기 때문

pre-training에서 [MASK] 토큰을 사용하므로, 두 과정 입력 데이터간의 차이가 발생한다.

⇒ masked word 의 일부에만 [MASK] 토큰을 사용

- 15% 토큰 무작위 선택 (→ 예측)
- [MASK}로 가리는 과정에서
  - **80%의 확률**로 **[MASK] 토큰**으로 대체
  - **10%의 확률**로 **무작위 토큰**으로 대체

- **10%의 확률로 원래 단어 그대로 유지**

⇒ [MASK]에만 집중하지 않도록, 다양한 변형을 접하게 하여 두 단계의 부조화 보완

Next Sentence Prediction (NSP)

: 문장 간의 관계성을 학습하기 위함

QA, NLI(Natural Language Inference 문제에서 중요

IsNext / NotNext : 두 라벨을 구분하는 학습

→ A가 주어졌을 때 문장 B는 다음에 올 문장인가?

이전에는 sentence embedding 만을 downstream task에 넘김

BERT : sentence embedding + 그 외 모든 파라미터 를 전달. 사전 학습된 모델의 파라미터 - 초기 값으로 사용 → fine-tuning

Pre-training Data

- BookCorpus
- English Wikipedia : 글만 사용, 표 등 무시 → 영향이 있을 수 있음

03.2 Fine-tuning BERT

Transformer Self-Attention - 문장 개수와 관계없이 문맥 파악이 가능하게 하여 BERT가 여러 작업에 쉽게 적용될 수 있게 함

(입력, 출력 형식을 변경하여 적용)

- 기존 : 두 문장 각각 인코딩 - 정보 교환
- BERT : 두 문장을 하나의 시퀀스로 concatenate → self-attention 으로 인코딩+정보교환

다양한 input 형태

- paraphrasing : sentence pairs

- Entailment : premise-hypothesis
- QA : quetion-passage
- Text Clf || Sequence Tagging : sentence-(비어있는 텍스트)

output

- token representation : Sequence tagging, QA
- [CLS] representation : 문맥 수준의 작업. Entailment, 감정 분석

Downstream Task에서 fine-tuning은 비교적 적은 비용을 요구

04 Experiments

11 NLP task 에 대해 평가

04.1 GLUE

: The General Language Understanding Evaluation benchmark

- 문장 이해 (NLI, 문장 유사도, 문장 분류)
    - 아홉 가지 평가 작업이 포함되어있다.

아홉 개의 평가에서 모두 BERT - outperform

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.

04.2 SQuAD

: Stanford Question Answering Dataset

- QA
  - 질문과 본문이 주어지고, 본문에서 정답을 찾는다. - 하나의 시퀀스로 입력
  - 질문-본문의 관계 파악 + 본문 내 정확한 위치에서 답변 추출(A가 있는 시작/끝 위치 예측)
  - F1 score

BERT - 사람 수준의 성능을 보임

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.

04.3 SQuAD v2.0

: SQuAD 업그레이드 버전-Null Answer(답이 없는 질문) 포함됨

→ CLS 토큰 활용하여 Null Answer 표현

- f1score

마찬가지로 높은 성능 보임

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.

04.4 SWAG

: Situations With Adversarial Generations

- 상황 추론 (Situational Reasoning)
  - 상황 → 자연스럽게 연결되는 문장 추론
  - 네 개의 선택지 중 정답으로 판단한 문장 선택
  - 문장의 연결성 평가
- 질문-선택지 문장을 하나의 시퀀스로 입력
- accuracy

BERT - 사람의 능력에 가까운 성능을 보임

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.

05 Ablation studies

: BERT의 구성 요소가 성과에 기여하는 방식을 설명

05.1 Effect of Pre-training Tasks

: the deep bidirectionality의 중요성

NO NSP : MLM, ~~NSP~~

LTR & NO NSP : Left-to-right only, ~~NSP.~~ 파인튜닝에서도 방향 그대로 제한

NO NSP 의 경우 성능이 떨어졌고, LTR까지 사용했을 때 성능은 더욱 낮아졌다.

SQuAD 에서 NSP를 지우는 것이 성능을 저하시킨다는 것이 명확히 보임

양방향 ELMod와의 비교

- 두 배의 비용이 든다. 양 방향으로 두번씩 계산인 반면 BERT는 양 방향을 한번에 계산
- non-intuitive
- less powerful

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.

05.2 Effect of Model Size

layers, hidden units, attention heads 의 개수별 성능 비교

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.

- 5 random restarts of fine-tuning
- BERTLARGE가 항상 더 높은 정확도를 보인다

  ⇒ 모델 크기가 클수록 성능 향상
- 작은 데이터셋(MRPC) 에서도 큰 모델이 더 좋은 성능 - 일반화 가능

- 이전 연구에서는 큰 모델이 downstream task에서도 좋은 결과를 만들지 않았지만, BERT는 fine-tuning 방식을 통해 모델 크기를 키웠을 때 더 좋은 성능을 기대할 수 있음

05.3 Feature-based Approach with BERT

: fine-tuning approach와 feature-based approach에서 BERT의 성능 비교

- fine-tuning : 사전 학습된 BERT에 분류 레이어 추가 → 작업에 맞춰 모든 파라미터 파인튜닝
- feature-based : 사전 학습된 BERT에서 activation 추출. 파라미터는 고정 → BiLSTM 인풋으로 사용

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.

⇒ BERTLARGE : 두 방식에서 모두 높은 성능을 보였다.

feature-based approach의 장점

- 작업에 따라 특별한 모델이 필요할 때 - transformer의 인코더로 표현하기 어려울 때
- 반복 사용 - 비용 절감

06 Conclusion

- unsupervised pre-training → transfer learning
- Bidirectional : 전이 학습의 이점을 확장 → 다양한 작업에 적용 가능