Improving Language Understanding by Generative Pre-Training

진행 상태	진행 중

25기 분석 인태영

Abstract

NLU 관련 라벨링 되지 않은 데이터들은 풍부하다 하지만 특정 Task를 해결하기 위해 학습에 필요한 라벨링 데이터들은 부족하기 때문에 판별형(discriminative) 모델은 좋은 성능을 내기 쉽지 않다.

이러한 문제를 해결하기 위해 GPT는 비라벨 데이터들을 이용해 generative pre-training을 한 후 특정 Task에 대해 discriminative fine-tuning을 이용한다.

Introduction

> NLP task에서
>
> 수작업 라벨링 데이터가 필요한 분야에서는 대부분의 딥러닝 방법을 사용했을 경우 데이터 부족에 의해 제대로 된 성능을 이끌어 낼 수 없을 것이다.
>
> 비라벨링 데이터를 이용해서 가치있는 정보를 활용할 수 있다면 라벨링 작업에 대해 소요되는 시간과 비용을 절약할 수 있다

Word2Vec 과 같은 pre-trained word embeddings 기법을 이용한 모델들은 위와 같은 상황을 신경쓰며 NLP Task를 해결할 수 있는 가능성을 열어 주었다.

하지만 단어 수준 이상의 표현을 학습하기에는 다음과 같은 두 가지 문제가 있었다.

- 텍스트 표현을 어떤 Task에 적용하냐에 따라 성능이 제각각이다.
- 어떤 Task에 적용할 때 그 적용 방법에 대해서도 명확하지 않다.

이 논문에서는

1. unsupervised pre-training 과 supervised fine-tuning을 결합한 semi-supervised 방식
2. Transformer 아키텍처
3. traversal-style 접근법

과 같은 특성들을 이용해 문제를 해결해 나간다.

Related Word

- Semi supervised learning for NLP

  단어 이상의 의미 정보를 비라벨 데이터로부터 학습하고 다양한 task에 적합한 벡터 표현으로 인코딩

- Unsupervised pre-training

  비지도 사전학습 방법을 이용해 좋은 초기화 지점을 찾고 정규화 역할을 수행한다.
  이러한 사전학습 이후 특정 task에 대해 fine-tuning을 수행해 목표를 달성한다.

- Auxiliary training objectives

  모델의 hidden state를 보조 특징으로 사용하여 성능을 향상 시키는 방법
  다만 이 논문에서는 비지도 사전학습만으로도 충분히 학습할 수 있음을 보여준다.

Framework

Unsupervised pre-training

$$\mathcal{L}_1(U) = \sum_i \log P(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta)$$

어떤 임베딩 토큰 k개에 대해 다음 토큰 u_i의 최대화하는 목적함수이다.

즉, 이전에 오는 토큰들에 대해서 다음에 나올 가장 높은 확률의 단어를 고르는 것이다.

어떻게 이루어지는 지 과정을 살펴보자

먼저 입력에 대해서

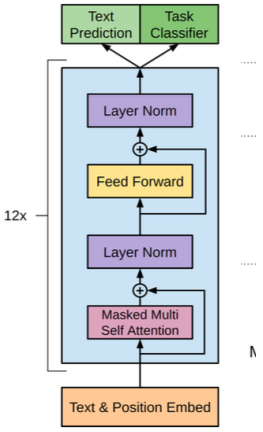
$$h_0 = UW_e + W_p$$

W_e : 단어를 벡터로 변환하는 토큰 임베딩 행렬

W_p : 위치 임베딩 행렬

입력 토큰들은 단어 의미 + 순서 정보가 더해진 벡터로 표현된다.

$$h_l = transformer\_block(h_{l-1})$$



이후 오른쪽과 같은 decoder-only transformer 레이어 12개를 이용한다.

- Multi-Head Self-Attentioin
  미래 단어들은 보지 못하도록 마스킹한 후 어텐션 매커니즘을 이용한다.
- Position-wise Feedforward Layer
  각 위치별로 Feedforward을 적용한다.
- Layer Normalization + Residual Connection
  안정적인 학습과 정보 손실 방지를 위해 적용한다.

출력 단계에서는

$$P(u) = softmax(h_n W_e^T)$$

h_n : 마지막 Transformer 레이어의 출력

W_e^T : 입력 임베딩 행렬을 전치하여 출력층 가중치로 재사용한다

softmax를 적용하여 다음 단어의 확률 분포를 도출한다.

Supervised fine-tuning

$$P(y \mid x_1, \ldots, x_m) = softmax(h_l^m W_y)$$

라벨링된 데이터셋을 입력 시퀀스 $(x_1, \ldots, x_m)$로 하여 사전학습된 Transformer 모델을 통화하며 최종 레이어의 출력 h_l^m를 얻는다.

이후 linear output layer W_y를 이용하여 라벨을 예측하고 역시 softmax를 이용해 출력을 얻는다.

$$\mathcal{L}_2(C) = \sum_{(x,y)} \log P(y \mid x_1, \ldots, x_m)$$

또한 cross-entropy를 적용하여 정답 라벨 y의 확률을 최대화 한다.

$$\mathcal{L}_3(C) = \mathcal{L}_2(C) + \lambda \cdot \mathcal{L}_1(C)$$

추가로 Auxiliary LM Objective를 추가했을 때 더 잘 학습되는 모습을 보인다. 다만 이는 데이터양에 따라서 차이가 존재하긴 한다.

Experiments

Task	Datasets
Natural language inference	SNLI [5], MultiNLI [66], Question NLI [64], RTE [4], SciTail [25]
Question Answering	RACE [30], Story Cloze [40]
Sentence similarity	MSR Paraphrase Corpus [14], Quora Question Pairs [9], STS Benchmark [6]
Classification	Stanford Sentiment Treebank-2 [54], CoLA [65]

위와 같은 데이터셋을 이용했으며 모델 세팅은 다음과 같다.

- **구조**
  - **12-layer Decoder-only Transformer**
  - Masked Self-Attention
  - Hidden state 차원: **768**
  - Attention head: **12**
  - Feed-forward 차원: **3072**
- **최적화(Optimization)**
  - Optimizer: **Adam**

    - 최대 학습률: **2.5e-4**
    - 학습률 스케줄:
        - 초기 2000 step 동안 **0 → max까지 선형 증가**
        - 이후 **cosine schedule** 로 감소
    - Epoch: **100**
    - Batch: **64 sequences × 512 tokens (연속 토큰 시퀀스)**
- **정규화 & 초기화**
    - LayerNorm 사용
    - 가중치 초기화: N(0,0.02)
    - Dropout: **0.1** (Residual, Embedding, Attention)
    - L2 정규화: [37] 변형 사용, w=0.01(bias, gain 제외)
    - 활성 함수: **GELU**

Finetuning시에는

- 기본적으로 **사전학습 하이퍼파라미터 재사용**
- 추가로 적용:
    - Classifier에 Dropout **0.1**
    - Learning rate: **6.25e-5**
    - Batch size: **32**
    - 학습 epoch: **3** (빠른 수렴)
    - 학습률 스케줄: 선형 감소 + warmup (전체 step의 **0.2%**)
    - Auxiliary LM Loss 가중치: λ=0.5\lambda = 0.5λ=0.5

성능 비교

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

GPT-1 vs 기존 SOTA 성능 비교 (절대 개선폭)

- **MNLI**: **+1.5%**
- **SciTail**: **+5.0%**
- **QNLI**: **+5.8%**
- **SNLI**: **+0.6%**
- **RTE**: **56% (기존 multi-task BiLSTM 61.7%보다 낮음)**

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	77.6	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	60.2	50.3	53.3
Finetuned Transformer LM (ours)	**86.5**	**62.9**	**57.4**	**59.0**

1. Question Answering & Commonsense Reasoning

- **RACE** (중·고등학교 영어 시험 기반, 추론 중심)
  - GPT-1: 기존 SOTA 대비 **+5.7%** 향상
- **Story Cloze Test** (이야기 결말 선택)
  - GPT-1: 기존 SOTA 대비 **+8.9%** 향상
- 해석: **긴 문맥(long-range context)** 추론 능력에서 강점 확인

Method	Classification		Semantic Similarity			GLUE
	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	
Sparse byte mLSTM [16]	-	**93.2**	-	-	-	-
TF-KLD [23]	-	-	**86.0**	-	-	-
ECNU (mixed ensemble) [60]	-	-	-	81.0	-	-
Single-task BiLSTM + ELMo + Attn [64]	35.0	90.2	80.2	55.5	66.1	64.8
Multi-task BiLSTM + ELMo + Attn [64]	18.9	91.6	83.5	72.8	63.3	68.9
Finetuned Transformer LM (ours)	**45.4**	91.3	82.3	**82.0**	**70.3**	**72.8**

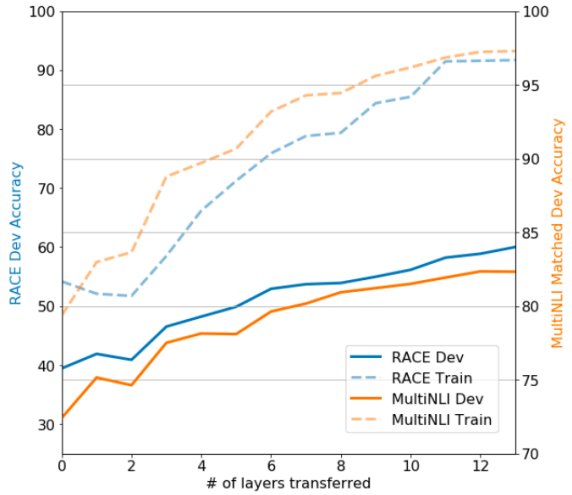
2. Semantic Similarity (Paraphrase Detection)

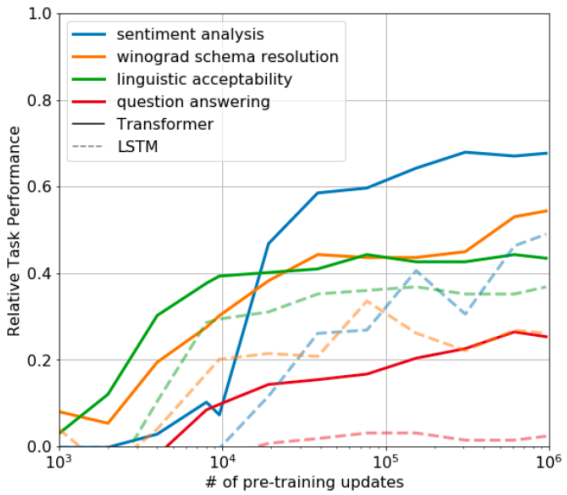
- **MRPC**: 경쟁력 있음
- **QQP (Quora Question Pairs)**: 기존 BiLSTM+ELMo+Attn 대비 **+4.2%** 향상
- **STS-B**: **+1점** 절대 성능 향상
- 해석: 단순 단어 수준이 아닌 **문장 의미 이해**에서 효과적

3. Classification

- **CoLA (문법 수용성 판단)**
  - GPT-1: **45.4점**, 기존 최고(35.0) 대비 **+10.4점** → 언어적 편향 학습 능력 입증
- **SST-2 (감정 분류)**
  - GPT-1: **91.3% 정확도**, 기존 SOTA와 동급
- **GLUE 종합 점수**
  - GPT-1: **72.8점**, 기존 최고(68.9) 대비 **+3.9점**

Analysis





왼쪽 그래프를 확인하면 비지도 사전학습에서 각 Transformer 레이어를 추가적으로 전이할 때마다 성능이 개선되는 점을 확인 할 수 있다.

오른쪽 그래프를 확인하면 Zero-shot Task 에 대해 Transformer와 LSTM의 차이를 알 수 있다.

우선 Transformer 구조의 장기기억성이 LSTM보다 전이에 유리한 점을 알 수 있으며 이에 따라 점진적으로 안정적이게 성능이 향상하는 모습을 알 수 있다.

Method	Avg. Score	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	MNLI (acc)	QNLI (acc)	RTE (acc)
Transformer w/ aux LM (full)	74.7	45.4	91.3	82.3	82.0	**70.3**	**81.8**	**88.1**	**56.0**
Transformer w/o pre-training	59.9	18.9	84.0	79.4	30.9	65.5	75.7	71.2	53.8
Transformer w/o aux LM	**75.0**	**47.9**	**92.0**	**84.9**	**83.2**	69.8	81.1	86.9	54.4
LSTM w/ aux LM	69.1	30.3	90.5	83.2	71.8	68.1	73.7	81.1	54.6

추가로 세가지 Ablation studies를 진행하였다.

1. 사전학습 제거
   사전학습이 없으면 모든 태스크에서 성능이 평균적으로 14.8% 저하되는 것을 알 수 있다. 즉 사전학습의 중요성을 보여준다.
2. Fine-tuning 시 보조 LM 목표 제거
   큰 데이터셋에서는 성능이 상승하였지만 작은 데이터셋에서는 효과가 미미하다.
3. Transformer 구조를 LSTM으로 변경
   Transformer가 평균적으로 훨씬 강력하다. 예외적으로 MRPC에서는 LSTM이 더 나은 결과를 보여준다.

Conclusion

pre-training + fine-tuning 방식의 단일 모델로도 12개의 데이터셋 중 9개에서 SOTA의 성과를 냈다.

Transformer 구조를 이용하여 장기 기억 능력을 가지게 되었으므로 긴 연속 텍스트에서도 좋은 학습을 유지할 수 있다.

이러한 비지도 학습이 실제 성능 향상에 기여함을 알 수 있으며 앞으로의 NLP 및 다른 도메인의 연구에도 중요한 단서가 될 수 있다.