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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova

**Abstract**

We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial taskspecific architecture modifications.

우리는 Bidirectional Encoder Representations from Transformers를 나타내는 BERT라는 새로운 언어 표현 모델을 소개합니다. 최근의 언어 표현 모델(Peters et al., 2018a; Radford et al., 2018)과 달리 BERT는 모든 레이어의 왼쪽 및 오른쪽 컨텍스트를 공동으로 조건화하여 레이블이 지정되지 않은 텍스트에서 깊은 양방향 표현을 사전 훈련하도록 설계되었습니다. 결과적으로 사전 훈련된 BERT 모델은 실질적인 작업별 아키텍처 수정 없이도 질문 답변 및 언어 추론과 같은 광범위한 작업을 위한 최첨단 모델을 생성하기 위해 단 하나의 추가 출력 레이어로 미세 조정될 수 있습니다.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement)

BERT는 개념적으로 단순하고 경험적으로 강력합니다. GLUE 점수를 80.5%(7.7% 절대 개선점), MultiNLI 정확도를 86.7%(4.6% 절대 개선점), SQUAD v1.1 정답 테스트 F1에서 93.2(1.5.2 절대 개선점) 및 SQU를 포함한 11개 자연어 처리 작업에 대한 최신 결과를 얻습니다.실행 중입니다.)

**1 Introduction**

Language model pre-training has been shown to be effective for improving many natural language processing tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). These include sentence-level tasks such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), which aim to predict the relationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

언어 모델 사전 훈련은 많은 자연어 처리 작업을 개선하는 데 효과적인 것으로 나타났습니다(Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). 여기에는 자연어 추론(Bowman et al., 2015; Williams et al., 2018) 및 의역(Dolan and Brockett, 2005)과 같은 문장 수준 작업이 포함됩니다. 토큰 수준에서 세분화된 출력을 생성하기 위해 모델이 필요한 명명된 개체 인식 및 질문 답변과 같은 토큰 수준 작업(Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

There are two existing strategies for applying pre-trained language representations to downstream tasks: feature-based and fine-tuning. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning all pretrained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

다운스트림 작업에 사전 훈련된 언어 표현을 적용하기 위한 두 가지 기존 전략이 있습니다. 기능 기반 및 미세 조정입니다. ELMo(Peters et al., 2018a)와 같은 기능 기반 접근 방식은 사전 훈련된 표현을 추가 기능으로 포함하는 작업별 아키텍처를 사용합니다. Generative Pre-trained Transformer(OpenAI GPT)(Radford et al., 2018)와 같은 미세 조정 접근 방식은 최소한의 작업별 매개변수를 도입하고 모든 사전 훈련된 매개변수를 미세 조정하여 다운스트림 작업에 대해 학습합니다. 두 가지 접근 방식은 사전 훈련 동안 동일한 목적 함수를 공유하며, 단방향 언어 모델을 사용하여 일반 언어 표현을 학습합니다.

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The major limitation is that standard language models are unidirectional, and this limits the choice of architectures that can be used during pre-training. For example, in OpenAI GPT, the authors use a left-toright architecture, where every token can only attend to previous tokens in the self-attention layers of the Transformer (Vaswani et al., 2017). Such restrictions are sub-optimal for sentence-level tasks, and could be very harmful when applying finetuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

우리는 현재 기술이 특히 미세 조정 접근 방식에 대해 사전 훈련된 표현의 힘을 제한한다고 주장합니다. 주요 제한 사항은 표준 언어 모델이 단방향이며 이는 사전 훈련 중에 사용할 수 있는 아키텍처 선택을 제한한다는 것입니다. 예를 들어, OpenAI GPT에서 작성자는 모든 토큰이 Transformer의 self-attention 레이어에 있는 이전 토큰에만 관심을 가질 수 있는 왼쪽에서 오른쪽으로 아키텍처를 사용합니다(Vaswani et al., 2017). 이러한 제한은 문장 수준 작업에 대해 차선책이며, 양방향 컨텍스트를 통합하는 것이 중요한 질문 답변과 같은 토큰 수준 작업에 미세 조정 기반 접근 방식을 적용할 때 매우 해로울 수 있습니다.

In this paper, we improve the fine-tuning based approaches by proposing BERT: Bidirectional Encoder Representations from Transformers. BERT alleviates the previously mentioned unidirectionality constraint by using a “masked language model” (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked word based only on its context. Unlike left-toright language model pre-training, the MLM objective enables the representation to fuse the left and the right context, which allows us to pretrain a deep bidirectional Transformer. In addition to the masked language model, we also use a “next sentence prediction” task that jointly pretrains text-pair representations. The contributions of our paper are as follows:

이 논문에서는 BERT: Bidirectional Encoder Representations from Transformers를 제안하여 미세 조정 기반 접근 방식을 개선합니다. BERT는 Cloze 작업(Taylor, 1953)에서 영감을 얻은 "마스킹된 언어 모델"(MLM) 사전 교육 목표를 사용하여 앞에서 언급한 단방향성 제약을 완화합니다. 마스킹된 언어 모델은 입력에서 일부 토큰을 무작위로 마스킹하며 목적은 컨텍스트만 기반으로 마스킹된 단어의 원래 어휘 ID를 예측하는 것입니다. 왼쪽에서 오른쪽으로 언어 모델 사전 훈련과 달리 MLM 목표는 표현이 왼쪽과 오른쪽 컨텍스트를 융합할 수 있도록 하여 깊은 양방향 변환기를 사전 훈련할 수 있습니다. 마스킹된 언어 모델 외에도 텍스트 쌍 표현을 공동으로 사전 훈련하는 "다음 문장 예측" 작업도 사용합니다. 우리 논문의 기여는 다음과 같습니다.

• We demonstrate the importance of bidirectional pre-training for language representations. Unlike Radford et al. (2018), which uses unidirectional language models for pre-training, BERT uses masked language models to enable pretrained deep bidirectional representations. This is also in contrast to Peters et al. (2018a), which uses a shallow concatenation of independently trained left-to-right and right-to-left LMs.

• 우리는 언어 표현을 위한 양방향 사전 교육의 중요성을 보여줍니다. Radford et al. (2018) 사전 훈련을 위해 단방향 언어 모델을 사용하는 BERT는 마스크된 언어 모델을 사용하여 사전 훈련된 깊은 양방향 표현을 가능하게 합니다. 이것은 Peters et al.과도 대조적입니다. (2018a), 독립적으로 훈련된 왼쪽에서 오른쪽 및 오른쪽에서 왼쪽 LM의 얕은 연결을 사용합니다.

• We show that pre-trained representations reduce the need for many heavily-engineered taskspecific architectures. BERT is the first finetuning based representation model that achieves state-of-the-art performance on a large suite of sentence-level and token-level tasks, outperforming many task-specific architectures.

• 사전 훈련된 표현이 고도로 엔지니어링된 많은 작업별 아키텍처의 필요성을 줄여준다는 것을 보여줍니다. BERT는 많은 작업별 아키텍처를 능가하는 대규모 문장 수준 및 토큰 수준 작업 모음에서 최첨단 성능을 달성하는 최초의 미세 조정 기반 표현 모델입니다.

• BERT advances the state of the art for eleven NLP tasks. The code and pre-trained models are available at https://github.com/ google-research/bert..

• BERT는 11개의 NLP 작업에 대한 최첨단 기술을 제공합니다. 코드와 사전 훈련된 모델은 https://github.com/google-research/bert..에서 사용할 수 있습니다.

**2 Related Work**

There is a long history of pre-training general language representations, and we briefly review the most widely-used approaches in this section.

일반 언어 표현을 사전 훈련한 오랜 역사가 있으며 이 섹션에서는 가장 널리 사용되는 접근 방식을 간략하게 검토합니다.

**2.1 Unsupervised Feature-based Approaches**

Learning widely applicable representations of words has been an active area of research for decades, including non-neural (Brown et al., 1992; Ando and Zhang, 2005; Blitzer et al., 2006) and neural (Mikolov et al., 2013; Pennington et al., 2014) methods. Pre-trained word embeddings are an integral part of modern NLP systems, offering significant improvements over embeddings learned from scratch (Turian et al., 2010). To pretrain word embedding vectors, left-to-right language modeling objectives have been used (Mnih and Hinton, 2009), as well as objectives to discriminate correct from incorrect words in left and right context (Mikolov et al., 2013).

광범위하게 적용 가능한 단어 표현을 학습하는 것은 비신경(Brown et al., 1992; Ando and Zhang, 2005; Blitzer et al., 2006) 및 신경(Mikolov et al., 2013)을 포함하여 수십 년 동안 활발한 연구 영역이었습니다. ; Pennington et al., 2014) 방법. 사전 훈련된 단어 임베딩은 현대 NLP 시스템의 필수적인 부분으로, 처음부터 학습한 임베딩에 비해 상당한 개선을 제공합니다(Turian et al., 2010). 단어 임베딩 벡터를 사전 훈련하기 위해 왼쪽에서 오른쪽으로 언어 모델링 목표(Mnih and Hinton, 2009)와 왼쪽 및 오른쪽 컨텍스트에서 올바른 단어와 잘못된 단어를 구별하는 목표(Mikolov et al., 2013)가 사용되었습니다.

These approaches have been generalized to coarser granularities, such as sentence embeddings (Kiros et al., 2015; Logeswaran and Lee, 2018) or paragraph embeddings (Le and Mikolov, 2014). To train sentence representations, prior work has used objectives to rank candidate next sentences (Jernite et al., 2017; Logeswaran and Lee, 2018), left-to-right generation of next sentence words given a representation of the previous sentence (Kiros et al., 2015), or denoising autoencoder derived objectives (Hill et al., 2016).

이러한 접근 방식은 문장 임베딩(Kiros et al., 2015; Logeswaran and Lee, 2018) 또는 단락 임베딩(Le and Mikolov, 2014)과 같이 거친 세분성으로 일반화되었습니다. 문장 표현을 훈련하기 위해, 이전 연구에서는 후보 다음 문장(Jernite et al., 2017; Logeswaran and Lee, 2018), 왼쪽에서 오른쪽으로 다음 문장 생성(Kiros et al. 2015) 또는 자동 인코더 파생 목표(Denoising autoencoder deriven)의 순위를 매기기 위해 목표를 사용했습니다(Hill et al., 2016).

ELMo and its predecessor (Peters et al., 2017, 2018a) generalize traditional word embedding research along a different dimension. They extract context-sensitive features from a left-to-right and a right-to-left language model. The contextual representation of each token is the concatenation of the left-to-right and right-to-left representations. When integrating contextual word embeddings with existing task-specific architectures, ELMo advances the state of the art for several major NLP benchmarks (Peters et al., 2018a) including question answering (Rajpurkar et al., 2016), sentiment analysis (Socher et al., 2013), and named entity recognition (Tjong Kim Sang and De Meulder, 2003). Melamud et al. (2016) proposed learning contextual representations through a task to predict a single word from both left and right context using LSTMs. Similar to ELMo, their model is feature-based and not deeply bidirectional. Fedus et al. (2018) shows that the cloze task can be used to improve the robustness of text generation models.

ELMo와 그 전신(Peters et al., 2017, 2018a)은 다른 차원의 전통적인 단어 임베딩 연구를 일반화합니다. 왼쪽에서 오른쪽으로 및 오른쪽에서 왼쪽으로 언어 모델에서 상황에 맞는 기능을 추출합니다. 각 토큰의 상황별 표현은 왼쪽에서 오른쪽으로 및 오른쪽에서 왼쪽으로의 표현을 연결한 것입니다. ELMo는 상황별 단어 임베딩을 기존 작업별 아키텍처와 통합할 때 질문 답변(Rajpurkar 등, 2016a), 감정 분석(Socher 등, 2013), 명명된 개체 인식(Tjong Kim Sang 및 Deulder 2003)을 포함한 여러 주요 NLP 벤치마크(Peters 등, 2018a)의 최첨단 기술을 발전시킵니다. Melamud et al. (2016)는 LSTM을 사용하여 좌우 문맥에서 하나의 단어를 예측하는 과제를 통해 문맥 표현을 학습할 것을 제안했습니다. ELMo와 유사하게, 그들의 모델은 특징 기반이며 깊은 양방향은 아닙니다. Fedus et al. (2018)는 클로즈 태스크를 사용하여 텍스트 생성 모델의 견고성을 개선할 수 있음을 보여줍니다.

**2.2 Unsupervised Fine-tuning Approaches**

As with the feature-based approaches, the first works in this direction only pre-trained word embedding parameters from unlabeled text (Collobert and Weston, 2008).

기능 기반 접근 방식과 마찬가지로 첫 번째 방법은 레이블이 지정되지 않은 텍스트에서 사전 훈련된 단어 임베딩 매개변수만 이 방향으로 작동합니다(Collobert and Weston, 2008).

More recently, sentence or document encoders which produce contextual token representations have been pre-trained from unlabeled text and fine-tuned for a supervised downstream task (Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018). The advantage of these approaches is that few parameters need to be learned from scratch. At least partly due to this advantage, OpenAI GPT (Radford et al., 2018) achieved previously state-of-the-art results on many sentencelevel tasks from the GLUE benchmark (Wang et al., 2018a). Left-to-right language modeling and auto-encoder objectives have been used for pre-training such models (Howard and Ruder, 2018; Radford et al., 2018; Dai and Le, 2015).

보다 최근에는 컨텍스트 토큰 표현을 생성하는 문장 또는 문서 인코더가 레이블이 지정되지 않은 텍스트에서 사전 훈련되고 감독되는 다운스트림 작업을 위해 미세 조정되었습니다(Dai 및 Le, 2015; Howard 및 Ruder, 2018; Radford et al., 2018). 이러한 접근 방식의 장점은 처음부터 학습해야 하는 매개변수가 거의 없다는 것입니다. 적어도 부분적으로 이러한 이점으로 인해 OpenAI GPT(Radford et al., 2018)는 GLUE 벤치마크(Wang et al., 2018a)의 많은 문장 수준 작업에서 이전의 최첨단 결과를 달성했습니다. 왼쪽에서 오른쪽으로 언어 모델링 및 자동 인코더 목표는 이러한 모델을 사전 훈련하는 데 사용되었습니다(Howard and Ruder, 2018; Radford et al., 2018; Dai and Le, 2015).

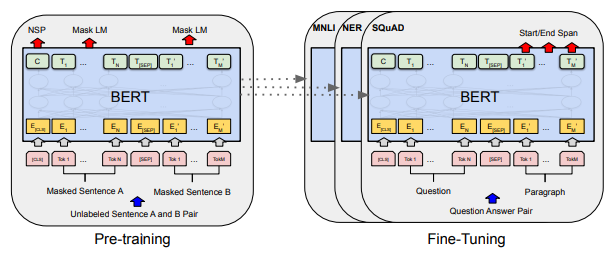


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

그림 1: BERT에 대한 전반적인 사전 교육 및 미세 조정 절차. 출력 레이어를 제외하고 사전 훈련과 미세 조정 모두에 동일한 아키텍처가 사용됩니다. 동일한 사전 훈련된 모델 매개변수가 다양한 다운스트림 작업에 대한 모델을 초기화하는 데 사용됩니다. 미세 조정 중에는 모든 매개변수가 미세 조정됩니다. [CLS]는 모든 입력 예제 앞에 추가되는 특수 기호이고 [SEP]는 특수 구분 토큰(예: 질문/답변 구분)입니다.

**2.3 Transfer Learning from Supervised Data**

There has also been work showing effective transfer from supervised tasks with large datasets, such as natural language inference (Conneau et al., 2017) and machine translation (McCann et al., 2017). Computer vision research has also demonstrated the importance of transfer learning from large pre-trained models, where an effective recipe is to fine-tune models pre-trained with ImageNet (Deng et al., 2009; Yosinski et al., 2014).

자연어 추론(Conneau et al., 2017) 및 기계 번역(McCann et al., 2017)과 같은 대규모 데이터 세트가 있는 감독 작업에서 효과적인 전송을 보여주는 작업도 있었습니다. 컴퓨터 비전 연구는 또한 ImageNet으로 사전 훈련된 모델을 미세 조정하는 것이 효과적인 방법인 사전 훈련된 대규모 모델에서 전이 학습의 중요성을 보여주었습니다(Deng et al., 2009; Yosinski et al., 2014).

**3 BERT**

We introduce BERT and its detailed implementation in this section. There are two steps in our framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For finetuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters. The question-answering example in Figure 1 will serve as a running example for this section.

이 섹션에서는 BERT와 그 세부 구현을 소개합니다. 프레임워크에는 사전 훈련과 미세 조정의 두 단계가 있습니다. 사전 훈련 동안 모델은 다양한 사전 훈련 작업에 대해 레이블이 지정되지 않은 데이터에 대해 훈련됩니다. 미세 조정을 위해 BERT 모델은 먼저 사전 훈련된 매개변수로 초기화되고 모든 매개변수는 다운스트림 작업의 레이블이 지정된 데이터를 사용하여 미세 조정됩니다. 각 다운스트림 작업에는 사전 훈련된 동일한 매개변수로 초기화된 경우에도 별도의 미세 조정된 모델이 있습니다. 그림 1의 질문 답변 예제는 이 섹션의 실행 예제 역할을 합니다.

A distinctive feature of BERT is its unified architecture across different tasks. There is minimal difference between the pre-trained architecture and the final downstream architecture.

BERT의 독특한 특징은 다양한 작업에 걸친 통합 아키텍처입니다. 사전 훈련된 아키텍처와 최종 다운스트림 아키텍처 사이에는 최소한의 차이가 있습니다.

Model Architecture BERT’s model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in Vaswani et al. (2017) and released in the tensor2tensor library.1 Because the use of Transformers has become common and our implementation is almost identical to the original, we will omit an exhaustive background description of the model architecture and refer readers to Vaswani et al. (2017) as well as excellent guides such as “The Annotated Transformer.”2

모델 아키텍처 BERT의 모델 아키텍처는 Vaswani 등 (2017)에서 설명되고 텐서2텐서 라이브러리에 공개된 원래 구현을 기반으로 하는 다층 양방향 변압기 인코더입니다.1 트랜스포머의 사용이 보편화되고 구현이 원본과 거의 동일하기 때문에 모델 아키텍처에 대한 자세한 배경 설명은 생략하고 Vaswani 등(2017)과 "The Annoted Transformer"와 같은 우수한 가이드를 참조하도록 하겠습니다.2

In this work, we denote the number of layers (i.e., Transformer blocks) as L, the hidden size as H, and the number of self-attention heads as A. We primarily report results on two model sizes: BERTBASE (L=12, H=768, A=12, Total Parameters=110M) and BERTLARGE (L=24, H=1024, A=16, Total Parameters=340M).

본 연구에서는 레이어 수(즉, 트랜스포머 블록)를 L로, 숨겨진 크기를 H로, 자가 주의 헤드의 수를 A로 나타냅니다. BERTBASE (L=12, H=768, A=12, 총 매개변수=110M) 및 H=Large (Large)의 두 가지 모델 크기에 대한 결과를 주로 보고합니다.

BERTBASE was chosen to have the same model size as OpenAI GPT for comparison purposes. Critically, however, the BERT Transformer uses bidirectional self-attention, while the GPT Transformer uses constrained self-attention where every token can only attend to context to its left.

BERTBASE는 Open과 동일한 모델 크기를 가지도록 선택되었습니다.비교를 위한 AI GPT입니다. 그러나 중요한 것은 BERT Transformer는 양방향 셀프 어텐션을 사용하는 반면 GPT Transformer는 모든 토큰이 왼쪽의 컨텍스트에만 주의를 기울일 수 있는 제한된 셀프 어텐션을 사용하는 것입니다.

Input/Output Representations To make BERT handle a variety of down-stream tasks, our input representation is able to unambiguously represent both a single sentence and a pair of sentences (e.g., h Question, Answeri) in one token sequence. Throughout this work, a “sentence” can be an arbitrary span of contiguous text, rather than an actual linguistic sentence. A “sequence” refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.

입력/출력 표현 BERT가 다양한 다운스트림 작업을 처리할 수 있도록 입력 표현은 단일 문장과 한 쌍의 문장(예: h 질문, 응답)을 하나의 토큰 시퀀스로 모호하지 않게 표현할 수 있습니다. 이 작업을 통해 "문장"은 실제 언어적인 문장이 아니라 연속된 텍스트의 임의 범위일 수 있습니다. "시퀀스"는 BERT에 대한 입력 토큰 시퀀스를 의미하며, 한 문장 또는 두 문장이 함께 채워질 수 있습니다.

We use WordPiece embeddings (Wu et al., 2016) with a 30,000 token vocabulary. The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks. Sentence pairs are packed together into a single sequence. We differentiate the sentences in two ways. First, we separate them with a special token ([SEP]). Second, we add a learned embedding to every token indicating whether it belongs to sentence A or sentence B. As shown in Figure 1, we denote input embedding as E, the final hidden vector of the special [CLS] token as , and the final hidden vector for the input token as .

30,000개의 토큰 어휘와 함께 WordPiece 임베딩(Wu et al., 2016)을 사용합니다. 모든 시퀀스의 첫 번째 토큰은 항상 특수 분류 토큰([CLS])입니다. 이 토큰에 해당하는 최종 숨겨진 상태는 분류 작업에 대한 집계 시퀀스 표현으로 사용됩니다. 문장 쌍은 단일 시퀀스로 함께 묶입니다. 우리는 두 가지 방법으로 문장을 구별합니다. 먼저 특수 토큰([SEP])으로 구분합니다. 둘째, 모든 토큰에 학습된 임베딩을 추가하여 그것이 문장 A에 속하는지 아니면 문장 B에 속하는지 여부를 나타냅니다. 그림 1에서 볼 수 있듯이 입력 임베딩을 E로 표시하고 특수 [CLS] 토큰의 최종 은닉 벡터를 로 표시합니다. 번째 입력 토큰에 대한 최종 은닉 벡터는 입니다.

For a given token, its input representation is constructed by summing the corresponding token, segment, and position embeddings. A visualization of this construction can be seen in Figure 2.

주어진 토큰의 입력 표현은 해당 토큰, 세그먼트 및 위치 임베딩을 합산하여 구성됩니다. 이 구성의 시각화는 그림 2에서 볼 수 있습니다.

**3.1 Pre-training BERT**

Unlike Peters et al. (2018a) and Radford et al. (2018), we do not use traditional left-to-right or right-to-left language models to pre-train BERT. Instead, we pre-train BERT using two unsupervised tasks, described in this section. This step is presented in the left part of Figure 1

Peters et al. (2018a) 및 Radford et al. (2018), 우리는 BERT를 사전 훈련하기 위해 전통적인 왼쪽에서 오른쪽 또는 오른쪽에서 왼쪽 언어 모델을 사용하지 않습니다. 대신 이 섹션에 설명된 두 가지 비지도 작업을 사용하여 BERT를 사전 훈련합니다. 이 단계는 그림 1의 왼쪽 부분에 나와 있습니다.

Task #1: Masked LM Intuitively, it is reasonable to believe that a deep bidirectional model is strictly more powerful than either a left-to-right model or the shallow concatenation of a left-toright and a right-to-left model. Unfortunately, standard conditional language models can only be trained left-to-right or right-to-left, since bidirectional conditioning would allow each word to indirectly “see itself”, and the model could trivially predict the target word in a multi-layered context.

과제 #1: 마스크 LM 직관적으로 심층 양방향 모델이 왼쪽에서 오른쪽으로 모델 또는 왼쪽에서 왼쪽으로 모델의 얕은 연결보다 훨씬 강력하다고 믿는 것이 합리적입니다. 불행하게도, 표준 조건부 언어 모델은 왼쪽에서 오른쪽으로 또는 오른쪽에서 왼쪽으로만 훈련될 수 있습니다. 왜냐하면 양방향 조건부 모델은 각 단어가 간접적으로 "자신을 볼 수 있도록" 허용하고 모델은 다중 계층 컨텍스트에서 대상 단어를 세심하게 예측할 수 있기 때문입니다.

In order to train a deep bidirectional representation, we simply mask some percentage of the input tokens at random, and then predict those masked tokens. We refer to this procedure as a “masked LM” (MLM), although it is often referred to as a Cloze task in the literature (Taylor, 1953). In this case, the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary, as in a standard LM. In all of our experiments, we mask 15% of all WordPiece tokens in each sequence at random. In contrast to denoising auto-encoders (Vincent et al., 2008), we only predict the masked words rather than reconstructing the entire input.

깊은 양방향 표현을 훈련하기 위해 우리는 단순히 입력 토큰의 일부를 무작위로 마스킹한 다음 마스킹된 토큰을 예측합니다. 우리는 이 절차를 "masked LM"(MLM)이라고 부르지만 문헌에서는 종종 Cloze 작업이라고 합니다(Taylor, 1953). 이 경우 마스크 토큰에 해당하는 최종 은닉 벡터는 표준 LM에서와 같이 어휘에 대한 출력 소프트맥스에 공급됩니다. 모든 실험에서 무작위로 각 시퀀스의 모든 WordPiece 토큰의 15%를 마스킹합니다. 잡음 제거 자동 인코더(Vincent et al., 2008)와 달리 우리는 전체 입력을 재구성하지 않고 마스킹된 단어만 예측합니다.

Although this allows us to obtain a bidirectional pre-trained model, a downside is that we are creating a mismatch between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning. To mitigate this, we do not always replace “masked” words with the actual [MASK] token. The training data generator chooses 15% of the token positions at random for prediction. If the i-th token is chosen, we replace the i-th token with (1) the [MASK] token 80% of the time (2) a random token 10% of the time (3) the unchanged i-th token 10% of the time. Then, will be used to predict the original token with cross entropy loss. We compare variations of this procedure in Appendix C.2.

이를 통해 양방향 사전 학습 모델을 얻을 수 있지만 미세 조정 중에는 [MASK] 토큰이 나타나지 않기 때문에 사전 학습과 미세 조정 간에 불일치가 발생한다는 단점이 있습니다. 이를 완화하기 위해 "마스킹된" 단어를 실제 [MASK] 토큰으로 항상 교체하지는 않습니다. 훈련 데이터 생성기는 예측을 위해 토큰 위치의 15%를 무작위로 선택합니다. i번째 토큰이 선택되면 i번째 토큰을 (1) [MASK] 토큰으로 80% 확률로 (2) 확률로 10% 임의 토큰으로 (3) 변경되지 않은 i번째 토큰 시간의 10%. 그런 다음 는 교차 엔트로피 손실이 있는 원래 토큰을 예측하는 데 사용됩니다. 부록 C.2에서 이 절차의 변형을 비교합니다.

Task #2: Next Sentence Prediction (NSP) Many important downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) are based on understanding the relationship between two sentences, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, we pre-train for a binarized next sentence prediction task that can be trivially generated from any monolingual corpus. Specifically, when choosing the sentences A and B for each pretraining example, 50% of the time B is the actual next sentence that follows A (labeled as IsNext), and 50% of the time it is a random sentence from the corpus (labeled as NotNext). As we show in Figure 1, C is used for next sentence prediction (NSP).5 Despite its simplicity, we demonstrate in Section 5.1 that pre-training towards this task is very beneficial to both QA and NLI.

작업 #2: NSP(다음 문장 예측) QA(질문 응답) 및 NLI(자연어 추론)와 같은 많은 중요한 다운스트림 작업은 언어 모델링으로 직접 캡처되지 않는 두 문장 간의 관계 이해를 기반으로 합니다. 문장 관계를 이해하는 모델을 훈련하기 위해 우리는 단일 언어 코퍼스에서 간단하게 생성할 수 있는 이진화된 다음 문장 예측 작업을 사전 훈련합니다. 특히, 각 사전 훈련 예제에 대해 문장 A와 B를 선택할 때 시간 B의 50%는 A 다음에 오는 실제 다음 문장이고(IsNext로 레이블 지정), 시간의 50%는 말뭉치(레이블이 지정됨)에서 임의의 문장입니다. NotNext로). 그림 1에서 볼 수 있듯이 C는 다음 문장 예측(NSP)에 사용됩니다.5 단순함에도 불구하고 섹션 5.1에서 이 작업에 대한 사전 교육이 QA와 NLI 모두에 매우 유익하다는 것을 보여줍니다.

키보드, 전자기기이(가) 표시된 사진

자동 생성된 설명

Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

그림 2: BERT 입력 표현. 입력 임베딩은 토큰 임베딩, 분할 임베딩 및 위치 임베딩의 합입니다.

The NSP task is closely related to representationlearning objectives used in Jernite et al. (2017) and Logeswaran and Lee (2018). However, in prior work, only sentence embeddings are transferred to down-stream tasks, where BERT transfers all parameters to initialize end-task model parameters.

NSP 작업은 Jernite et al.에서 사용된 표현 학습 목표와 밀접하게 관련되어 있습니다. (2017) 및 Logeswaran 및 Lee(2018). 그러나 이전 작업에서는 문장 임베딩만 다운스트림 작업으로 전송되며, 여기서 BERT는 최종 작업 모델 매개변수를 초기화하기 위해 모든 매개변수를 전송합니다.

Pre-training data The pre-training procedure largely follows the existing literature on language model pre-training. For the pre-training corpus we use the BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia (2,500M words). For Wikipedia we extract only the text passages and ignore lists, tables, and headers. It is critical to use a document-level corpus rather than a shuffled sentence-level corpus such as the Billion Word Benchmark (Chelba et al., 2013) in order to extract long contiguous sequences.

사전 훈련 데이터 사전 훈련 절차는 언어 모델 사전 훈련에 대한 기존 문헌을 크게 따릅니다. 사전 훈련 말뭉치의 경우 BooksCorpus(8억 단어)(Zhu et al., 2015)와 영어 Wikipedia(2,500M 단어)를 사용합니다. Wikipedia의 경우 텍스트 구절만 추출하고 목록, 테이블 및 헤더는 무시합니다. 긴 연속 시퀀스를 추출하려면 Billion Word Benchmark(Chelba et al., 2013)와 같이 섞인 문장 수준 말뭉치보다 문서 수준 말뭉치를 사용하는 것이 중요합니다.

**3.2 Fine-tuning BERT**

Fine-tuning is straightforward since the selfattention mechanism in the Transformer allows BERT to model many downstream tasks— whether they involve single text or text pairs—by swapping out the appropriate inputs and outputs. For applications involving text pairs, a common pattern is to independently encode text pairs before applying bidirectional cross attention, such as Parikh et al. (2016); Seo et al. (2017). BERT instead uses the self-attention mechanism to unify these two stages, as encoding a concatenated text pair with self-attention effectively includes bidirectional cross attention between two sentences.

Transformer의 자가 주의 메커니즘을 통해 BERT는 적절한 입력과 출력을 교환하여 단일 텍스트 또는 텍스트 쌍을 포함하는 많은 다운스트림 작업을 모델링할 수 있으므로 미세 조정은 간단합니다. 텍스트 쌍을 포함하는 응용 프로그램의 경우 일반적인 패턴은 Parikh et al.과 같이 양방향 교차 주의를 적용하기 전에 텍스트 쌍을 독립적으로 인코딩하는 것입니다. (2016); Seo et al. (2017). BERT는 self-attention으로 연결된 텍스트 쌍을 인코딩하는 것이 두 문장 사이의 양방향 교차 주의를 효과적으로 포함하기 때문에 대신 self-attention 메커니즘을 사용하여 이러한 두 단계를 통합합니다.

For each task, we simply plug in the taskspecific inputs and outputs into BERT and finetune all the parameters end-to-end. At the input, sentence A and sentence B from pre-training are analogous to (1) sentence pairs in paraphrasing, (2) hypothesis-premise pairs in entailment, (3) question-passage pairs in question answering, and (4) a degenerate text-∅ pair in text classification or sequence tagging. At the output, the token representations are fed into an output layer for tokenlevel tasks, such as sequence tagging or question answering, and the [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis.

각 작업에 대해 작업별 입력 및 출력을 BERT에 연결하고 모든 매개변수를 종단 간 미세 조정하기만 하면 됩니다. 입력에서 사전 훈련의 문장 A와 문장 B는 (1) 의역에서 문장 쌍, (2) 수반에서 가설-전제 쌍, (3) 질문 답변에서 질문-구절 쌍, (4) a와 유사합니다. 텍스트 분류 또는 시퀀스 태깅에서 텍스트-∅ 쌍을 퇴화합니다. 출력에서 토큰 표현은 시퀀스 태깅 또는 질문 응답과 같은 토큰 수준 작업을 위한 출력 레이어에 공급되고 [CLS] 표현은 수반 또는 감정 분석과 같은 분류를 위해 출력 레이어에 공급됩니다.

Compared to pre-training, fine-tuning is relatively inexpensive. All of the results in the paper can be replicated in at most 1 hour on a single Cloud TPU, or a few hours on a GPU, starting from the exact same pre-trained model.7 We describe the task-specific details in the corresponding subsections of Section 4. More details can be found in Appendix A.5.

사전 훈련에 비해 미세 조정은 비교적 저렴합니다. 백서의 모든 결과는 동일한 사전 훈련된 모델에서 시작하여 단일 Cloud TPU에서 최대 1시간 또는 GPU에서 몇 시간 만에 복제할 수 있습니다.7 해당하는 작업별 세부 정보를 설명합니다. 섹션 4의 하위 섹션. 자세한 내용은 부록 A.5에서 찾을 수 있습니다.

**4 Experiments**

In this section, we present BERT fine-tuning results on 11 NLP tasks.

이 섹션에서는 11개의 NLP 작업에 대한 BERT 미세 조정 결과를 제시합니다.

**4.1 GLUE**

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018a) is a collection of diverse natural language understanding tasks. Detailed descriptions of GLUE datasets are included in Appendix B.1.

GLUE(일반 언어 이해 평가) 벤치마크(Wang et al., 2018a)는 다양한 자연어 이해 작업의 모음입니다. GLUE 데이터 세트에 대한 자세한 설명은 부록 B.1에 포함되어 있습니다.

To fine-tune on GLUE, we represent the input sequence (for single sentence or sentence pairs) as described in Section 3, and use the final hidden vector corresponding to the first input token ([CLS]) as the aggregate representation. The only new parameters introduced during fine-tuning are classification layer weights , where K is the number of labels. We compute a standard classification loss with C and W, i.e., log(softmax( )).

GLUE를 미세 조정하기 위해 섹션 3에서 설명한 대로 입력 시퀀스(단일 문장 또는 문장 쌍의 경우)를 나타내고 첫 번째 입력 토큰([CLS])에 해당하는 최종 은닉 벡터 를 다음으로 사용합니다. 집계 표현. 미세 조정 중에 도입된 유일한 새 매개변수는 분류 계층 가중치 이며, 여기서 K는 레이블 수입니다. C와 W, 즉 log(softmax( ))로 표준 분류 손실을 계산합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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 singlemodel, 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.

표 1: 평가 서버에서 채점한 GLUE 테스트 결과(https://gluebenchmark.com/leaderboard). 각 작업 아래의 숫자는 훈련 예제의 수를 나타냅니다. "Average" 열은 문제가 있는 WNLI 세트를 제외하기 때문에 공식 GLUE 점수와 약간 다릅니다. 8 BERT 및 OpenAI GPT는 단일 모델, 단일 작업입니다. F1 점수는 QQP 및 MRPC에 대해 보고되고 Spearman 상관 관계는 STS-B에 대해 보고되며 정확도 점수는 다른 작업에 대해 보고됩니다. BERT를 구성 요소 중 하나로 사용하는 항목은 제외됩니다.

We use a batch size of 32 and fine-tune for 3 epochs over the data for all GLUE tasks. For each task, we selected the best fine-tuning learning rate (among 5e-5, 4e-5, 3e-5, and 2e-5) on the Dev set. Additionally, for BERTLARGE we found that finetuning was sometimes unstable on small datasets, so we ran several random restarts and selected the best model on the Dev set. With random restarts, we use the same pre-trained checkpoint but perform different fine-tuning data shuffling and classifier layer initialization.

32의 배치 크기를 사용하고 모든 GLUE 작업에 대한 데이터에 대해 3개의 에포크에 대해 미세 조정합니다. 각 작업에 대해 Dev 세트에서 최고의 미세 조정 학습률(5e-5, 4e-5, 3e-5 및 2e-5)을 선택했습니다. 또한 BERTLARGE의 경우 소규모 데이터 세트에서 미세 조정이 불안정한 경우가 있으므로 무작위로 여러 번 다시 시작하고 Dev 세트에서 최상의 모델을 선택했습니다. 무작위 재시작으로 동일한 사전 훈련된 체크포인트를 사용하지만 다른 미세 조정 데이터 셔플링 및 분류기 계층 초기화를 수행합니다.

Results are presented in Table 1. Both and outperform all systems on all tasks by a substantial margin, obtaining 4.5% and 7.0% respective average accuracy improvement over the prior state of the art. Note that and OpenAI GPT are nearly identical in terms of model architecture apart from the attention masking. For the largest and most widely reported GLUE task, MNLI, BERT obtains a 4.6% absolute accuracy improvement. On the official GLUE leaderboard10, obtains a score of 80.5, compared to OpenAI GPT, which obtains 72.8 as of the date of writing.

결과는 표 1에 제시되어 있습니다. 및 는 모든 작업에서 모든 시스템보다 상당한 차이로 성능이 향상되어 이전 기술 상태에 비해 각각 평균 ​​정확도가 4.5% 및 7.0% 향상되었습니다. 와 OpenAI GPT는 어텐션 마스킹을 제외하고는 모델 아키텍처 측면에서 거의 동일합니다. 가장 크고 가장 널리 보고된 GLUE 작업인 MNLI의 경우 BERT는 4.6%의 절대 정확도 향상을 얻습니다. 공식 GLUE 리더보드10에서 는 작성일 현재 72.8점을 얻은 OpenAI GPT와 비교하여 80.5점을 받았습니다.

We find that significantly outperforms across all tasks, especially those with very little training data. The effect of model size is explored more thoroughly in Section 5.2.

우리는 〖BERT〗\_LARGE가 모든 작업, 특히 훈련 데이터가 매우 적은 작업에서 〖BERT〗\_BASE를 훨씬 능가한다는 것을 발견했습니다. 모델 크기의 영향은 섹션 5.2에서 더 자세히 살펴봅니다.

**4.2 SQuAD v1.1**

The Stanford Question Answering Dataset (SQuAD v1.1) is a collection of 100k crowdsourced question/answer pairs (Rajpurkar et al., 2016). Given a question and a passage from Wikipedia containing the answer, the task is to predict the answer text span in the passage.

Stanford Question Answering Dataset(SQuAD v1.1)은 크라우드소싱된 100,000개의 질문/답변 쌍의 모음입니다(Rajpurkar et al., 2016). 질문과 답변이 포함된 Wikipedia의 구절이 주어지면 작업은 해당 구절의 답변 텍스트 범위를 예측하는 것입니다.

As shown in Figure 1, in the question answering task, we represent the input question and passage as a single packed sequence, with the question using the A embedding and the passage using the B embedding. We only introduce a start vector and an end vector during fine-tuning. The probability of word i being the start of the answer span is computed as a dot product between and S followed by a softmax over all of the words in the paragraph: . The analogous formula is used for the end of the answer span. The score of a candidate span from position i to position j is defined as , and the maximum scoring span where j ≥ i is used as a prediction. The training objective is the sum of the log-likelihoods of the correct start and end positions. We fine-tune for 3 epochs with a learning rate of 5e-5 and a batch size of 32

그림 1과 같이 질문 답변 작업에서 입력 질문과 구절을 A 임베딩을 사용하는 질문과 B 임베딩을 사용하는 구절을 사용하여 단일 묶음 시퀀스로 표현합니다. 미세 조정 중에 시작 벡터 S ∈ R^H와 끝 벡터 E ∈R^H만 도입합니다. 단어 i가 답변 범위의 시작일 확률은 와 S 사이의 내적으로 계산된 다음 단락의 모든 단어에 대한 softmax가 뒤따릅니다. . 유사한 공식이 응답 범위의 끝에 사용됩니다. 위치 i에서 위치 j까지의 후보 범위의 점수를 , 로 정의하고, j ≥ i인 최대 점수 범위를 예측으로 사용합니다. 훈련 목표는 올바른 시작 위치와 끝 위치의 로그 가능성의 합입니다. 학습률이 5e-5이고 배치 크기가 32인 3 Epoch에 대해 미세 조정합니다.

Table 2 shows top leaderboard entries as well as results from top published systems (Seo et al., 2017; Clark and Gardner, 2018; Peters et al., 2018a; Hu et al., 2018). The top results from the SQuAD leaderboard do not have up-to-date public system descriptions available,11 and are allowed to use any public data when training their systems. We therefore use modest data augmentation in our system by first fine-tuning on TriviaQA (Joshi et al., 2017) befor fine-tuning on SQuAD.

표 2는 상위 순위표 항목과 상위 게시 시스템의 결과를 보여줍니다(Seo et al., 2017; Clark and Gardner, 2018; Peters et al., 2018a; Hu et al., 2018). SQuAD 리더보드의 상위 결과에는 사용 가능한 최신 공개 시스템 설명이 없으며11 시스템을 교육할 때 공개 데이터를 사용할 수 있습니다. 따라서 SQuAD를 미세 조정하기 전에 먼저 TriviaQA(Joshi et al., 2017)를 미세 조정하여 시스템에서 적당한 데이터 증대를 사용합니다.

Our best performing system outperforms the top leaderboard system by +1.5 F1 in ensembling and +1.3 F1 as a single system. In fact, our single BERT model outperforms the top ensemble system in terms of F1 score. Without TriviaQA finetuning data, we only lose 0.1-0.4 F1, still outperforming all existing systems by a wide margin.

우리의 최고 성능 시스템은 앙상블에서 +1.5 F1, 단일 시스템에서 +1.3 F1만큼 상위 리더보드 시스템보다 성능이 뛰어납니다. 사실, 우리의 단일 BERT 모델은 F1 점수 측면에서 상위 앙상블 시스템을 능가합니다. TriviaQA 미세 조정 데이터가 없으면 0.1-0.4 F1만 잃게 되며 여전히 모든 기존 시스템보다 훨씬 뛰어난 성능을 보입니다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

표 2: SQuAD 1.1 결과. BERT 앙상블은 다양한 사전 훈련 체크포인트와 미세 조정 시드를 사용하는 7x 시스템입니다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

표 3: SQuAD 2.0 결과. BERT를 구성 요소 중 하나로 사용하는 항목은 제외됩니다.

**4.3 SQuAD v2.0**

The SQuAD 2.0 task extends the SQuAD 1.1 problem definition by allowing for the possibility that no short answer exists in the provided paragraph, making the problem more realistic.

SQuAD 2.0 작업은 제공된 단락에 단답형이 없을 가능성을 허용함으로써 SQuAD 1.1 문제 정의를 확장하여 문제를 보다 현실적으로 만듭니다.

We use a simple approach to extend the SQuAD v1.1 BERT model for this task. We treat questions that do not have an answer as having an answer span with start and end at the [CLS] token. The probability space for the start and end answer span positions is extended to include the position of the [CLS] token. For prediction, we compare the score of the no-answer span: snull = S·C + E·C to the score of the best non-null span . We predict a non-null answer when , where the threshold is selected on the dev set to maximize F1. We did not use TriviaQA data for this model. We fine-tuned for 2 epochs with a learning rate of 5e-5 and a batch size of 48.

우리는 이 작업을 위해 SQuAD v1.1 BERT 모델을 확장하기 위해 간단한 접근 방식을 사용합니다. 답변이 없는 질문은 [CLS] 토큰에서 시작과 끝이 있는 답변 범위가 있는 것으로 처리합니다. 시작 및 종료 응답 범위 위치에 대한 확률 공간은 [CLS] 토큰의 위치를 ​​포함하도록 확장됩니다. 예측을 위해 우리는 무응답 범위의 점수를 비교합니다. snull = S·C + E·C 와 최고의 널이 아닌 범위 span . 일 때 null이 아닌 응답을 예측합니다. 여기서 임계값 τ는 F1을 최대화하도록 설정된 dev에서 선택됩니다. 이 모델에는 TriviaQA 데이터를 사용하지 않았습니다. 학습률이 5e-5이고 배치 크기가 48인 2 Epoch에 대해 미세 조정했습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 4: SWAG Dev and Test accuracies. †Human performance is measured with 100 samples, as reported in the SWAG paper.

표 4: SWAG 개발 및 테스트 정확도. †인간의 성과는 SWAG 논문에 보고된 대로 100개의 샘플로 측정되었습니다.

The results compared to prior leaderboard entries and top published work (Sun et al., 2018; Wang et al., 2018b) are shown in Table 3, excluding systems that use BERT as one of their components. We observe a +5.1 F1 improvement over the previous best system.

BERT를 구성 요소 중 하나로 사용하는 시스템을 제외하고 이전 순위표 항목 및 상위 게시 작업(Sun et al., 2018; Wang et al., 2018b)과 비교한 결과가 표 3에 나와 있습니다. 우리는 이전 최고의 시스템에 비해 +5.1 F1 개선을 관찰했습니다.

**4.4 SWAG**

4.4 SWAG The Situations With Adversarial Generations (SWAG) dataset contains 113k sentence-pair completion examples that evaluate grounded commonsense inference (Zellers et al., 2018). Given a sentence, the task is to choose the most plausible continuation among four choices.

4.4 SWAG SWAG(Situations With Adversarial Generations) 데이터 세트에는 근거가 있는 상식 추론을 평가하는 113k 문장 쌍 완성 예제가 포함되어 있습니다(Zellers et al., 2018). 주어진 문장에서 과제는 네 가지 선택 중에서 가장 그럴듯한 연속을 선택하는 것입니다.

When fine-tuning on the SWAG dataset, we construct four input sequences, each containing the concatenation of the given sentence (sentence A) and a possible continuation (sentence B). The only task-specific parameters introduced is a vector whose dot product with the [CLS] token representation C denotes a score for each choice which is normalized with a softmax layer.

SWAG 데이터 세트를 미세 조정할 때 각각 주어진 문장(문장 A)과 가능한 연속(문장 B)의 연결을 포함하는 4개의 입력 시퀀스를 구성합니다. 도입된 유일한 작업별 매개변수는 [CLS] 토큰 표현 C가 있는 내적이 softmax 레이어로 정규화되는 각 선택에 대한 점수를 나타내는 벡터입니다.

We fine-tune the model for 3 epochs with a learning rate of 2e-5 and a batch size of 16. Results are presented in Table 4. BERTLARGE outperforms the authors’ baseline ESIM+ELMo system by +27.1% and OpenAI GPT by 8.3%.

학습 속도 2e-5와 배치 크기 16으로 3에폭 모델을 미세 조정합니다. 결과는 표 4에 나와 있습니다. BERTLARGE는 저자의 기준 ESIM+ELMo 시스템을 +27.1% 능가하며 OpenAI GPT가 8.3% 증가했습니다.

**5 Ablation Studies**

In this section, we perform ablation experiments over a number of facets of BERT in order to better understand their relative importance. Additional ablation studies can be found in Appendix C.

이 섹션에서는 상대적 중요성을 더 잘 이해하기 위해 BERT의 여러 측면에 대해 절제 실험을 수행합니다. 추가 절제 연구는 부록 C에서 찾을 수 있습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

Table 5: Ablation over the pre-training tasks using the 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.

표 5: BERT\_BASE 아키텍처를 사용한 사전 훈련 작업에 대한 제거. "No NSP"는 다음 문장 예측 작업 없이 훈련됩니다. "LTR & No NSP"는 OpenAI GPT와 같이 다음 문장 예측 없이 왼쪽에서 오른쪽으로 LM으로 훈련됩니다. "+ BiLSTM"은 미세 조정 중에 "LTR + No NSP" 모델 위에 무작위로 초기화된 BiLSTM을 추가합니다.

**5.1 Effect of Pre-training Tasks**

We demonstrate the importance of the deep bidirectionality of BERT by evaluating two pretraining objectives using exactly the same pretraining data, fine-tuning scheme, and hyperparameters as :

와 정확히 동일한 사전 훈련 데이터, 미세 조정 체계 및 하이퍼파라미터를 사용하여 두 가지 사전 훈련 목표를 평가하여 BERT의 깊은 양방향성의 중요성을 보여줍니다.

No NSP: A bidirectional model which is trained using the “masked LM” (MLM) but without the “next sentence prediction” (NSP) task.

NSP 없음: "마스킹된 LM"(MLM)을 사용하여 훈련되지만 "다음 문장 예측"(NSP) 작업 없이 훈련되는 양방향 모델입니다.

LTR & No NSP: A left-context-only model which is trained using a standard Left-to-Right (LTR) LM, rather than an MLM. The left-only constraint was also applied at fine-tuning, because removing it introduced a pre-train/fine-tune mismatch that degraded downstream performance. Additionally, this model was pre-trained without the NSP task. This is directly comparable to OpenAI GPT, but using our larger training dataset, our input representation, and our fine-tuning scheme.

LTR 및 NSP 없음: MLM이 아닌 표준 LTR(왼쪽에서 오른쪽으로) LM을 사용하여 훈련되는 왼쪽 컨텍스트 전용 모델입니다. 왼쪽 전용 제약 조건을 제거하면 다운스트림 성능을 저하시키는 사전 훈련/미세 조정 불일치가 발생하기 때문에 미세 조정 시에도 적용되었습니다. 또한 이 모델은 NSP 작업 없이 사전 학습되었습니다. 이것은 OpenAI GPT와 직접 비교할 수 있지만 더 큰 훈련 데이터 세트, 입력 표현 및 미세 조정 체계를 사용합니다.

We first examine the impact brought by the NSP task. In Table 5, we show that removing NSP hurts performance significantly on QNLI, MNLI, and SQuAD 1.1. Next, we evaluate the impact of training bidirectional representations by comparing “No NSP” to “LTR & No NSP”. The LTR model performs worse than the MLM model on all tasks, with large drops on MRPC and SQuAD.

먼저 NSP 작업이 가져오는 영향을 조사합니다. 표 5에서 NSP를 제거하면 QNLI, MNLI 및 SQuAD 1.1에서 성능이 크게 저하된다는 것을 알 수 있습니다. 다음으로 "No NSP"를 "LTR & No NSP"와 비교하여 양방향 표현 교육의 영향을 평가합니다. LTR 모델은 모든 작업에서 MLM 모델보다 성능이 좋지 않으며 MRPC 및 SQuAD가 크게 떨어집니다.

For SQuAD it is intuitively clear that a LTR model will perform poorly at token predictions, since the token-level hidden states have no rightside context. In order to make a good faith attempt at strengthening the LTR system, we added a randomly initialized BiLSTM on top. This does significantly improve results on SQuAD, but the results are still far worse than those of the pretrained bidirectional models. The BiLSTM hurts performance on the GLUE tasks.

SQuAD의 경우 토큰 수준 숨겨진 상태에는 오른쪽 컨텍스트가 없기 때문에 LTR 모델이 토큰 예측에서 제대로 수행되지 않을 것이라는 점은 직관적으로 분명합니다. LTR 시스템을 강화하기 위해 선의의 시도를 하기 위해 무작위로 초기화된 BiLSTM을 맨 위에 추가했습니다. 이것은 SQuAD의 결과를 크게 개선하지만 결과는 여전히 사전 훈련된 양방향 모델의 결과보다 훨씬 나쁩니다. BiLSTM은 GLUE 작업의 성능을 저하시킵니다.

We recognize that it would also be possible to train separate LTR and RTL models and represent each token as the concatenation of the two models, as ELMo does. However: (a) this is twice as expensive as a single bidirectional model; (b) this is non-intuitive for tasks like QA, since the RTL model would not be able to condition the answer on the question; (c) this it is strictly less powerful than a deep bidirectional model, since it can use both left and right context at every layer.

우리는 ELMo가 하는 것처럼 별도의 LTR 및 RTL 모델을 훈련하고 각 토큰을 두 모델의 연결로 나타내는 것도 가능하다는 것을 알고 있습니다. 그러나: (a) 이것은 단일 양방향 모델보다 두 배 비쌉니다. (b) RTL 모델이 질문에 대한 답변을 조건화할 수 없기 때문에 이것은 QA와 같은 작업에 대해 직관적이지 않습니다. (c) 이것은 모든 계층에서 왼쪽 및 오른쪽 컨텍스트를 모두 사용할 수 있기 때문에 깊은 양방향 모델보다 엄격하게 덜 강력합니다.

**5.2 Effect of Model Size**

In this section, we explore the effect of model size on fine-tuning task accuracy. We trained a number of BERT models with a differing number of layers, hidden units, and attention heads, while otherwise using the same hyperparameters and training procedure as described previously.

이 섹션에서는 모델 크기가 미세 조정 작업 정확도에 미치는 영향을 살펴봅니다. 우리는 이전에 설명한 것과 동일한 하이퍼파라미터와 훈련 절차를 사용하면서 다른 수의 레이어, 은닉 유닛 및 주의 헤드를 사용하여 여러 BERT 모델을 훈련했습니다.

Results on selected GLUE tasks are shown in Table 6. In this table, we report the average Dev Set accuracy from 5 random restarts of fine-tuning. We can see that larger models lead to a strict accuracy improvement across all four datasets, even for MRPC which only has 3,600 labeled training examples, and is substantially different from the pre-training tasks. It is also perhaps surprising that we are able to achieve such significant improvements on top of models which are already quite large relative to the existing literature. For example, the largest Transformer explored in Vaswani et al. (2017) is (L=6, H=1024, A=16) with 100M parameters for the encoder, and the largest Transformer we have found in the literature is (L=64, H=512, A=2) with 235M parameters (Al-Rfou et al., 2018). By contrast, contains 110M parameters and contains 340M parameters.

선택된 GLUE 작업에 대한 결과는 표 6에 나와 있습니다. 이 표에서는 미세 조정을 무작위로 다시 시작한 5번의 평균 Dev Set 정확도를 보고합니다. 3,600개의 레이블이 지정된 훈련 예제만 있고 사전 훈련 작업과 상당히 다른 MRPC의 경우에도 더 큰 모델은 4개의 데이터 세트 모두에서 엄격한 정확도 향상으로 이어진다는 것을 알 수 있습니다. 또한 기존 문헌에 비해 이미 상당히 큰 모델을 기반으로 이러한 상당한 개선을 달성할 수 있다는 사실도 놀랍습니다. 예를 들어, Vaswani et al.에서 조사된 가장 큰 변압기. (2017)은 인코더에 대한 100M 매개변수가 있는 (L=6, H=1024, A=16)이고 문헌에서 찾은 가장 큰 변압기는 (L=64, H=512, A=2) 235M입니다. 매개변수(Al-Rfou et al., 2018). 대조적으로 BERT\_BASE에는 1억 1천만 개의 매개변수가 포함되고 BERT\_LARGE에는 3억 4천만 개의 매개변수가 포함됩니다.

It has long been known that increasing the model size will lead to continual improvements on large-scale tasks such as machine translation and language modeling, which is demonstrated by the LM perplexity of held-out training data shown in Table 6. However, we believe that this is the first work to demonstrate convincingly that scaling to extreme model sizes also leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained. Peters et al. (2018b) presented mixed results on the downstream task impact of increasing the pre-trained bi-LM size from two to four layers and Melamud et al. (2016) mentioned in passing that increasing hidden dimension size from 200 to 600 helped, but increasing further to 1,000 did not bring further improvements. Both of these prior works used a featurebased approach — we hypothesize that when the model is fine-tuned directly on the downstream tasks and uses only a very small number of randomly initialized additional parameters, the taskspecific models can benefit from the larger, more expressive pre-trained representations even when downstream task data is very small.

모델 크기를 늘리면 기계 번역 및 언어 모델링과 같은 대규모 작업에서 지속적으로 개선된다는 사실은 오랫동안 알려져 왔으며, 이는 표 6에 표시된 보류된 훈련 데이터의 LM 혼란으로 입증됩니다. 그러나 우리는 믿습니다. 이것은 모델이 충분히 사전 훈련된 경우 극단적인 모델 크기로 확장하면 매우 작은 규모의 작업에서도 큰 개선으로 이어진다는 것을 설득력 있게 보여주는 첫 번째 작업입니다. Peters et al. (2018b)는 사전 훈련된 bi-LM 크기를 2개에서 4개 레이어로 늘리는 다운스트림 작업 영향에 대해 혼합된 결과를 제시했으며 Melamud et al. (2016) 전달에서 언급한 숨겨진 차원 크기를 200에서 600으로 늘리면 도움이 되지만 1,000으로 더 늘려도 더 이상 개선되지 않습니다. 이러한 이전 작업은 모두 기능 기반 접근 방식을 사용했습니다. 모델이 다운스트림 작업에서 직접 미세 조정되고 무작위로 초기화된 매우 적은 수의 추가 매개변수만 사용하는 경우 작업별 모델이 더 크고 표현력 있는 사전의 이점을 얻을 수 있다고 가정합니다. - 다운스트림 작업 데이터가 매우 작은 경우에도 훈련된 표현.

**5.3 Feature-based Approach with BERT**

All of the BERT results presented so far have used the fine-tuning approach, where a simple classification layer is added to the pre-trained model, and all parameters are jointly fine-tuned on a downstream task. However, the feature-based approach, where fixed features are extracted from the pretrained model, has certain advantages. First, not all tasks can be easily represented by a Transformer encoder architecture, and therefore require a task-specific model architecture to be added. Second, there are major computational benefits to pre-compute an expensive representation of the training data once and then run many experiments with cheaper models on top of this representation.

지금까지 제시된 모든 BERT 결과는 사전 훈련된 모델에 간단한 분류 계층이 추가되고 모든 매개변수가 다운스트림 작업에서 공동으로 미세 조정되는 미세 조정 접근 방식을 사용했습니다. 그러나 사전 훈련된 모델에서 고정된 특징을 추출하는 특징 기반 접근 방식에는 몇 가지 장점이 있습니다. 첫째, 모든 작업을 Transformer 인코더 아키텍처로 쉽게 표현할 수 있는 것은 아니므로 작업별 모델 아키텍처를 추가해야 합니다. 둘째, 훈련 데이터의 값비싼 표현을 한 번 미리 계산한 다음 이 표현 위에 더 저렴한 모델로 많은 실험을 실행하는 주요 계산 이점이 있습니다.

In this section, we compare the two approaches by applying BERT to the CoNLL-2003 Named Entity Recognition (NER) task (Tjong Kim Sang and De Meulder, 2003). In the input to BERT, we use a case-preserving WordPiece model, and we include the maximal document context provided by the data. Following standard practice, we formulate this as a tagging task but do not use a CRF layer in the output. We use the representation of the first sub-token as the input to the token-level classifier over the NER label set.

이 섹션에서는 BERT를 CoNLL-2003 NER(Named Entity Recognition) 작업에 적용하여 두 가지 접근 방식을 비교합니다(Tjong Kim Sang and De Meulder, 2003). BERT에 대한 입력에서 대소문자를 유지하는 WordPiece 모델을 사용하고 데이터에서 제공하는 최대 문서 컨텍스트를 포함합니다. 표준 관행에 따라 이것을 태깅 작업으로 공식화하지만 출력에서 ​​CRF 레이어를 사용하지 않습니다. 첫 번째 하위 토큰의 표현을 NER 레이블 세트에 대한 토큰 수준 분류기에 대한 입력으로 사용합니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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.

표 6: BERT 모델 크기에 대한 절제. #L = 레이어 수 #H = 숨겨진 크기; #A = 주의 헤드의 수. "LM(ppl)"은 보류된 훈련 데이터의 마스크된 LM 혼란도입니다.

테이블이(가) 표시된 사진

자동 생성된 설명

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.

표 7: ConLL-2003 명명된 엔터티 인식 결과. Dev 집합을 사용하여 하이퍼파라미터를 선택했습니다. 보고된 개발 및 테스트 점수는 해당 하이퍼파라미터를 사용하여 무작위로 5회 다시 시작하여 평균을 낸 것입니다.

To ablate the fine-tuning approach, we apply the feature-based approach by extracting the activations from one or more layers without fine-tuning any parameters of BERT. These contextual embeddings are used as input to a randomly initialized two-layer 768-dimensional BiLSTM before the classification layer.

미세 조정 접근 방식을 제거하기 위해 BERT의 매개 변수를 미세 조정하지 않고 하나 이상의 레이어에서 활성화를 추출하여 기능 기반 접근 방식을 적용합니다. 이러한 컨텍스트 임베딩은 분류 계층 전에 무작위로 초기화된 2계층 768차원 BiLSTM에 대한 입력으로 사용됩니다.

Results are presented in Table 7. BERTLARGE performs competitively with state-of-the-art methods. The best performing method concatenates the token representations from the top four hidden layers of the pre-trained Transformer, which is only 0.3 F1 behind fine-tuning the entire model. This demonstrates that BERT is effective for both finetuning and feature-based approaches.

결과는 표 7에 나와 있습니다. BERTLARGE는 최첨단 방법으로 경쟁적으로 수행합니다. 최고의 성능을 발휘하는 방법은 사전 훈련된 Transformer의 상위 4개 은닉 레이어의 토큰 표현을 연결합니다. 이는 BERT가 미세 조정 및 기능 기반 접근 방식 모두에 효과적이라는 것을 보여줍니다.

**6 Conclusion**

Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems. In particular, these results enable even low-resource tasks to benefit from deep unidirectional architectures. Our major contribution is further generalizing these findings to deep bidirectional architectures, allowing the same pre-trained model to successfully tackle a broad set of NLP tasks.

언어 모델을 사용한 전이 학습으로 인한 최근의 경험적 개선은 풍부하고 감독되지 않은 사전 훈련이 많은 언어 이해 시스템의 필수적인 부분임을 보여주었습니다. 특히 이러한 결과를 통해 리소스가 적은 작업도 깊은 단방향 아키텍처의 이점을 누릴 수 있습니다. 우리의 주요 기여는 이러한 결과를 심층 양방향 아키텍처로 일반화하여 동일한 사전 훈련된 모델이 광범위한 NLP 작업 세트를 성공적으로 처리할 수 있도록 하는 것입니다.