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

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

我々は、Transformersからの双方向エンコーダ表現の略であるBERTと呼ばれる新しい言語表現モデルを紹介する。最近の言語表現モデル(Peters et al., 2018a; Radford et al., 2018)とは異なり、BERTは、すべての層で左右両方の文脈を共同で条件付けることにより、ラベルのないテキストから深い双方向表現を事前学習するように設計されている。その結果、事前学習されたBERTモデルは、1つの追加出力層で微調整され、タスク固有のアーキテクチャを大幅に変更することなく、質問応答や言語推論などの幅広いタスクのための最先端モデルを作成することができます。

BERT は概念的に単純で、経験的に強力である。GLUEスコアを80.5%に引き上げる(7.7%ポイント絶対改善)、MultiNLI精度を86.7%に下げる(4.6%絶対改善)、SQuAD v1.1 質問応答テストF1を93.2に、SQuAD v2.0 テストF1を83.1に(5.1ポイント絶対改善)、11の自然言語処理タスクで新しい最先端結果を得ることができました。

1 Introduction

言語モデルの事前学習は、多くの自然言語処理タスクの改善に有効であることが示されている(Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018)。これらには、自然言語推論(Bowman et al., 2015; Williams et al., 2018)や言い換え(Dolan and Brockett, 2005)などの文レベルのタスクが含まれ、文間の関係を全体的に分析して予測することを目的としているほか、名前付き実体認識や質問応答などのトークンレベルでは、トークンレベルできめ細かい出力を生成することが求められる(Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016)。

下流タスクに事前学習された言語表現を適用するための既存の戦略は、特徴ベースと微調整の2つである。ELMo (Peters et al., 2018a) などの特徴ベースアプローチは、事前学習された表現を追加特徴として含むタスク固有のアーキテクチャを使用する。Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018) などの微調整アプローチは、最小限のタスク固有のパラメータを導入し、すべての事前学習パラメータを単に微調整することで下流タスクで学習させるものである。2つのアプローチは、一般的な言語表現を学習するために一方向言語モデルを使用する、事前学習中に同じ目的関数を共有する。我々は、現在の技術は、特に微調整アプローチにおいて、事前学習された表現の力を制限することを主張する。主な制限は、標準的な言語モデルが一方向であるため、事前学習時に使用できるアーキテクチャの選択が制限されることである。例えば、OpenAI GPTでは、著者らは、すべてのトークンがTransformerの自己注意層で以前のトークンにしか注意を払わない、左-右アーキテクチャを使用している(Vaswani et al.、2017)。このような制約は文レベルのタスクには最適ではなく、質問応答のようなトークンレベルのタスクに微調整に基づくアプローチを適用する際に非常に有害である可能性があり、両方向からのコンテキストを取り入れることが重要である。本論文では、BERTを提案することで、微調整に基づくアプローチを改善する。BERTは、Clozeタスク(Taylor, 1953)に触発された「マスク言語モデル」(MLM)事前学習目的を用いることで、前述の一方向性制約を緩和するものである。

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

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train 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 fine-tuned 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 task-specific architecture modifications.

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).

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).

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* pre-trained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

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-to-right 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 fine-tuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

In this paper, we improve the fine-tuning based approaches by proposing BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. 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

マスク言語モデルは入力からトークンの一部をランダムにマスクし、その目的はマスクされた単語の文脈のみに基づいて元の語彙IDを予測することである。左から右への言語モデルの事前学習とは異なり、MLMの目的は左と右の文脈を融合した表現を可能にし、深い双方向のTransformerの事前学習を可能にする。また、マスク言語モデルに加え、テキストとペアの表現を共同で事前学習する「次文予測」タスクも用いる。本論文の貢献は以下の通りである。

- 言語表現における双方向の事前学習の重要性を実証する。一方向言語モデルを事前学習に用いるRadfordら(2018)とは異なり、BERTはマスクされた言語モデルを用いて事前学習された深い双方向表現を可能にする。これは、独立して訓練された左から右、右から左のLMの浅い連結を用いるPetersら(2018a)とも対照的である。

- 我々は、事前に学習された表現が、多くの重点的に設計されたタスクに特化したアーキテクチャの必要性を低減することを示す。BERTは、文レベルおよびトークンレベルの大規模なタスクにおいて最先端の性能を達成し、多くのタスクに特化したアーキテクチャを凌駕する、最初のファインチューニングベースの表現モデルである。

- BERTは11のNLPタスクの現状を前進させる。コードと学習済みモデルは https://github.com/google-research/bert で公開されている。

2 Related Work

一般的な言語表現の事前学習には長い歴史があり、このセクションで最も広く使われているアプローチを簡単にレビューします。

2.1 Unsupervised Feature-based Approaches

非ニューラル(Brown et al., 1992; Ando and Zhang, 2005; Blitzer et al., 2006)やニューラル(Mikolov et al., 2013; Pennington et al., 2014)手法など、広く適用できる単語表現の学習は数十年にわたり活発な研究分野であった。事前に学習された単語埋め込みは、現代の自然言語処理システムに不可欠なものであり、ゼロから学習した埋め込みよりも大幅に改善されます(Turian et al.、2010)。単語埋め込みベクトルを事前学習するために、左から右への言語モデリング目標(Mnih and Hinton, 2009)や、左右の文脈で正しい単語と間違った単語を識別する目標(Mikolov et al, 2013)が使用されています。

これらのアプローチは、文埋め込み(Kiros et al., 2015; Logeswaran and Lee, 2018)や段落埋め込み(Le and Mikolov, 2014)などの粗い粒度に一般化されてきた。文表現を訓練するために、先行研究は、次の文候補をランク付けする目的(Jernite et al., 2017; Logeswaran and Lee, 2018)、前の文の表現が与えられた次の文の単語の左から右への生成(Kiros et al., 2015)、またはオートエンコーダ由来の目的のノイズ除去(Hill et al., 2016)に使用されてきた。

ELMo and its predecessor (Peters et al., 2017, 2018a)は、従来の単語埋め込み研究を別の次元で一般化したものである。彼らは、左から右、右から左の言語モデルから文脈に応じた特徴を抽出する。各トークンの文脈表現は、左から右、右から左の表現を連結したものである。文脈的な単語埋め込みを既存のタスク固有のアーキテクチャと統合する場合、ELMoは質問応答(Rajpurkarら、2016)、感情分析(Socherら、2013)、名前付きエンティティ認識(Tjong Kim Sang and De Meulder、2003)などのいくつかの主要なNLPベンチマーク(Petersら、2018a)の技術水準を前進させる。Melamudら(2016)は、LSTMを用いて左右両方の文脈から単一の単語を予測するタスクを通じて文脈表現を学習することを提案した。ELMoと同様に、彼らのモデルは特徴ベースであり、深い双方向性はない。Fedusら(2018)は、クローズタスクがテキスト生成モデルの頑健性を向上させるために使用できることを示す。

2.2 Unsupervised Fine-tuning Approaches

特徴ベースアプローチと同様に、この方向で最初に機能するのは、ラベルのないテキストから事前に学習された単語埋め込みパラメータのみである(Collobert and Weston, 2008)。

More recently, sentence or document encoders は、文脈的なトークン表現を生成する、ラベルのないテキストから事前学習され、教師ありの下流タスクのために微調整されている(Dai and Le, 2015; Howard and Ruder, 2018; Radford et al, 2018)。これらのアプローチの利点は、ゼロから学習する必要のあるパラメータが少ないことである。少なくともこの利点のため、OpenAI GPT (Radford et al., 2018) は、GLUEベンチマーク (Wang et al., 2018a) の多くの文レベルのタスクでこれまで最先端の結果を達成しました。

word based only on its context. Unlike left-to-right language model pre-training, the MLM objective enables the representation to fuse the left and the right context, which allows us to pre-train a deep bidirectional Transformer. In addition to the masked language model, we also use a "next sentence prediction" task that jointly pre-trains text-pair representations. The contributions of our paper are as follows:

- 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 pre-trained 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.
- We show that pre-trained representations reduce the need for many heavily-engineered task-specific architectures. BERT is the first fine-tuning 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.
- 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`.

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.

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 pre-train 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).

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 auto-encoder derived objectives (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.

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).

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 sentence-level tasks from the GLUE benchmark (Wang et al., 2018a). Left-to-right language model-

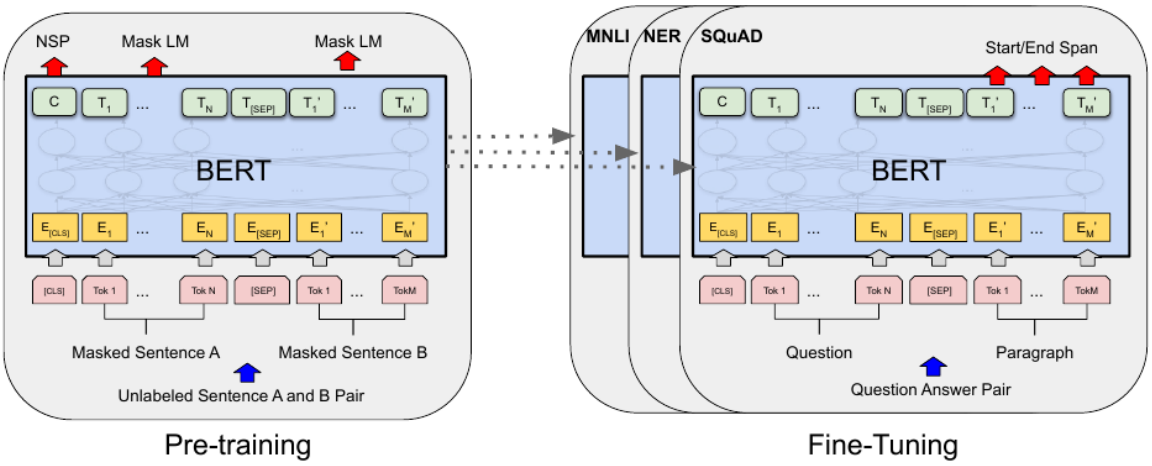

図1:BERTの事前学習と微調整の全体手順。出力層とは別に、事前学習と微調整の両方で同じアーキテクチャが使用されている。同じ事前学習済みモデルパラメータは、異なるダウンストリームタスクのモデルを初期化するために使用される。微調整の間、すべてのパラメータは微調整される。[CLS]はすべての入力例の前に追加された特別な記号であり、[SEP]は特別なセパレータトークン(例:質問/回答の分離)である。

左から右への言語モデルingとオートエンコーダー目的は、そのようなモデルの事前学習に使用されている(Howard and Ruder, 2018; Radford et al., 2018; Dai and Le, 2015)。

2.3 教師ありデータからの転移学習

また、自然言語推論(Conneau et al., 2017)や機械翻訳(McCann et al., 2017)など、大規模データセットによる教師ありタスクからの効果的な転送を示す研究も行われている。コンピュータビジョンの研究では、ImageNetで事前学習したモデルを微調整することが有効なレシピである、大規模な事前学習済みモデルからの転移学習の重要性も実証されている(Deng et al., 2009; Yosinski et al., 2014)。

3 BERT

本節では、BERT とその詳細な実装を紹介する。我々のフレームワークには、事前学習と微調整の2つのステップがある。事前学習では、異なる事前学習タスクにおいて、ラベルのないデータでモデルを学習させる。微調整のために、まずBERTモデルを事前学習したパラメータで初期化し、すべてのパラメータを下流タスクのラベル付きデータを用いて微調整する。各下流タスクは、同じ事前学習済みパラメータで初期化されているにもかかわらず、個別の微調整されたモデルを持っています。図1の質問応答の例は、このセクションの実行例として使用されます。BERTの特徴は、異なるタスク間で統一されたアーキテクチャであることです。

事前学習済みアーキテクチャと最終的な下流アーキテクチャの間には、最小限の差しかない。

モデルアーキテクチャ BERTのモデルアーキテクチャは、Vaswaniら(2017)に記載され、tensor2tensorライブラリでリリースされたオリジナルの実装に基づく多層双方向Transformerエンコーダである。[1] Transformerの使用が一般的になり、我々の実装はオリジナルとほぼ同じであるため、我々はモデルアーキテクチャの網羅的な背景説明を省略し、読者にVaswaniら(2017)と "The Annotated Transformer."^{2}などの優れたガイドを参照することにする。 本研究では、層数(すなわち、Transformerブロック)をL、隠れサイズをH、自己注意ヘッド数をA.^{3}}と表記する。主に二つのモデルサイズに関する結果を報告する。BERT_{BASE} (L=12, H=768, A=12, Total Parameters=110M) と BERT_{LARGE} (L=24, H=1024, A=16, Total ParameBERT$_{BASE}$は、比較のためにOpenAI GPTと同じモデルサイズに選ばれた。しかし、重要なのは、BERT Transformerは双方向の自己注意を使用し、GPT Transformerはすべてのトークンがその左の文脈にしか注意できない制約付き自己注意を使用します。

[1]https://github.com/tensorflow/tensor2tensor

[2]http://nlp.seas.harvard.edu/2018/04/03/attention.html

[3]In all cases we set the feed-forward/filter size to be $4H$, i.e., 3072 for the $H = 768$ and 4096 for the $H = 1024$.

[4]We note that in the literature the bidirectional Trans-

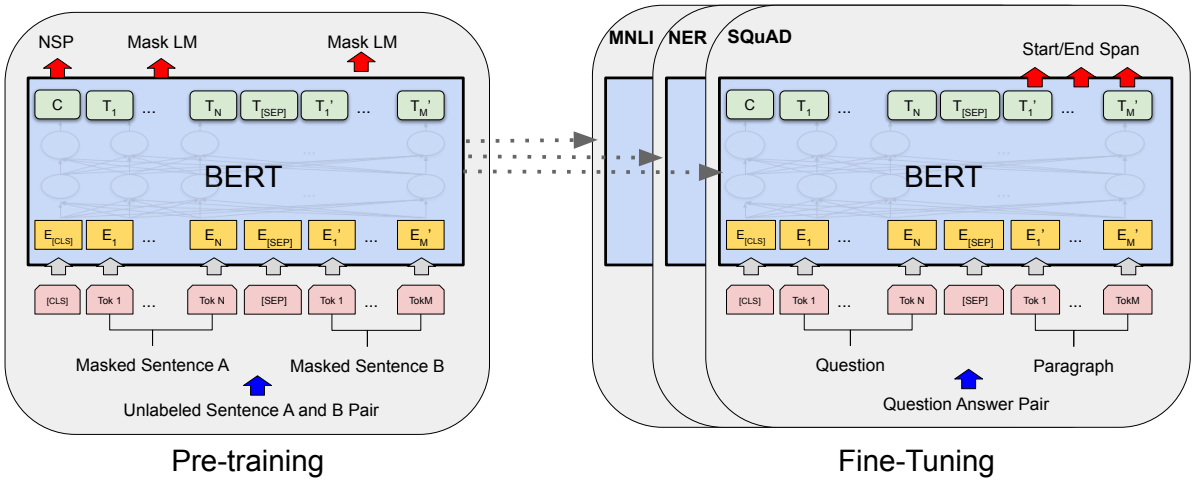

Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are 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).

ing and auto-encoder objectives have been used for pre-training such models (Howard and Ruder, 2018; Radford et al., 2018; Dai and Le, 2015).

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).

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 fine-tuning, 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.

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.

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]

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.[3] We primarily report results on two model sizes: $\textbf{BERT}_{\textbf{BASE}}$ (L=12, H=768, A=12, Total Parameters=110M) and $\textbf{BERT}_{\textbf{LARGE}}$ (L=24, H=1024, A=16, Total Parameters=340M).

$\text{BERT}_{\text{BASE}}$ 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.[4]

[1] https://github.com/tensorflow/tensor2tensor

[2] http://nlp.seas.harvard.edu/2018/04/03/attention.html

[3] In all cases we set the feed-forward/filter size to be $4H$, i.e., 3072 for the $H = 768$ and 4096 for the $H = 1024$.

[4] We note that in the literature the bidirectional Trans-

入力/出力表現 BERTが様々なダウンストリームタスクを処理できるように、我々の入力表現は、単一の文と一対の文(例えば、h Question、Answer i)の両方を一つのトークン列で一義的に表現することができる。この作業を通じて、「文」は実際の言語文ではなく、連続した任意のスパンのテキストであることができる。シーケンス」とは、BERTへの入力トークン列を指し、1つの文または2つの文が一緒に詰まっている場合がある。我々は、3万のトークン語彙を持つWordPiece embeddings (Wu et al., 2016)を使用する。すべてのシーケンスの最初のトークンは常に特別な分類トークン([CLS])である。このトークンに対応する最終的な隠れ状態は、分類タスクのための集約的なシーケンス表現として使用される。文のペアは1つのシーケンスにまとめられる。我々は2つの方法で文を区別する。まず、特別なトークン([SEP])で区切る。次に、学習された埋め込みを、それが文Aに属するか文Bに属するかを示す全てのトークンに追加する。図1に示すように、入力埋め込みをE、特殊[CLS]トークンの最終隠れベクトルをC$\in R^H$、i^{th}入力トークンの最終隠れベクトルを$T_i \in R^H$ と表記する。 与えられたトークンに対して、その入力表現は対応するトークン、セグメント、位置の埋め込みを合計することによって構築される。この構成の視覚化は図2に示す通りである。

3.1 Pre-training BERT

Petersら(2018a)やRadfordら(2018)とは異なり、我々はBERTの事前学習に従来の左から右、または右から左の言語モデルを使用しない。その代わりに、このセクションで説明する2つの教師なしタスクを使用してBERTを事前学習します。このステップは、図 1 の左側に示されている。

タスク1: Masked LM 直感的には、深い双方向モデルは左から右へのモデルや左から右、右から左へのモデルの浅い連結よりも厳密に強力であると考えるのが妥当であろう。残念ながら、標準的な条件付き言語モデルは左から右、右から左のいずれにも学習させることができる。なぜなら、双方向条件付けは各単語を間接的に「自分自身を見る」ことを可能にし、モデルは多層コンテキストでターゲット単語を些細に予測することができるからである。

深い双方向表現を学習するために、入力トークンの何割かをランダムにマスクし、それらのマスクされたトークンを予測するだけである。この手順を「マスクドLM」(MLM)と呼ぶが、文献上ではしばしばClozeタスクと呼ばれる(Taylor, 1953)。この場合、マスクトークンに対応する最終的な隠れベクトルは、標準的なLMと同様に語彙に対する出力ソフトマックスに供給される。すべての実験において、各シーケンスに含まれる全WordPieceトークンの15%をランダムにマスクする。ノイズ除去のオートエンコーダ(Vincent et al., 2008)とは対照的に、我々は入力全体を再構築するのではなく、マスクされた単語のみを予測する。これにより、双方向の事前学習済みモデルを得ることができるが、[MASK]トークンが微調整中に現れないため、事前学習と微調整の間にミスマッチが生じるという欠点がある。これを軽減するために、我々は必ずしも「マスクされた」単語を実際の[MASK]トークンに置き換えるとは限らない。学習データ生成器は、予測のためにトークン位置の15%をランダムに選択する。i番目のトークンが選択された場合、i番目のトークンを(1)80%の確率で[MASK]トークンに置き換える(2)10%の確率でランダムトークンに置き換える(3)10%の確率で変更されていないi番目のトークンに置き換える。そして、T_iを用いて、クロスエントロピー損失で元のトークンを予測する。この手順のバリエーションを付録C.2で比較する。

タスク2:次文予測(NSP) 質問応答(QA)や自然言語推論(NLI)などの多くの重要な下流タスクは、2つの文の関係を理解することに基づいており、言語モデリングでは直接捉えられない。文の関係を理解するモデルを学習するために、任意の単言語コーパスから些細に生成できる二値化次文予測タスクの事前学習を行う。具体的には、各前学習例で文A、Bを選ぶ場合、50%の確率でAに続く次の文(IsNextとラベル付け)、50%の確率でコーパスからランダムに文(NotNextとラベル付け)が選ばれる。図1に示すように、Cは次文予測(NSP)に用いられる。[5] その単純さにもかかわらず、このタスクに対する事前学習がQAとNLIの両方に非常に有益であることをセクション5.1で実証する。

前者はしばしば「トランスフォーマーエンコーダ」と呼ばれ、左文脈のみのバージョンはテキスト生成に使用できるため「トランスフォーマーデコーダ」と呼ばれる。

[5] 最終モデルはNSPで97%-98%の精度を達成した。[6] ベクトルCはNSPで学習したため、微調整を行わないと意味のある文表現にはならない。

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., ⟨ Question, Answer ⟩) 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.

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 $C \in \mathbb{R}^H$, and the final hidden vector for the i^{th} input token as $T_i \in \mathbb{R}^H$.

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.

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.

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-to-right 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.

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.

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, T_i will be used to predict the original token with cross entropy loss. We compare variations of this procedure in Appendix C.2.

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 pre-training 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. [6]

former is often referred to as a "Transformer encoder" while the left-context-only version is referred to as a "Transformer decoder" since it can be used for text generation.

[5] The final model achieves 97%-98% accuracy on NSP.

[6] The vector C is not a meaningful sentence representation without fine-tuning, since it was trained with NSP.

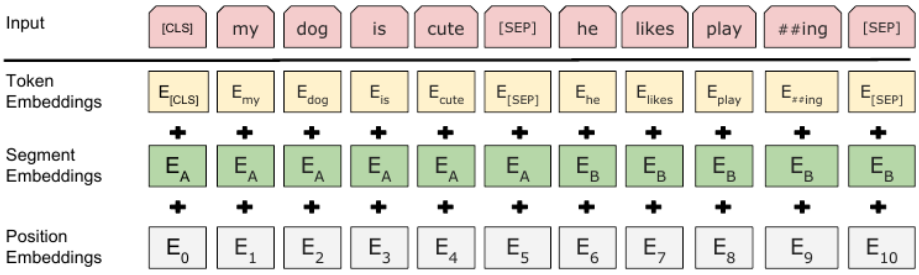

図2:BERTの入力表現。入力埋め込みは、トークン埋め込み、セグメンテーション埋め込み、位置埋め込みの和である。

NSPタスクは、Jerniteら(2017)およびLogeswaran and Lee(2018)で用いられた表現学習目的と密接に関連している。しかし、先行研究では、文の埋め込みのみがダウンストリームタスクに転送され、BERTはすべてのパラメータを転送してエンドタスクモデルのパラメータを初期化する。

事前学習データ 事前学習手順は、言語モデルの事前学習に関する既存の文献にほぼ準拠している。事前学習コーパスにはBooksCorpus (800M words) (Zhu et al., 2015) とEnglish Wikipedia (2,500M words)を使用する。Wikipediaについては、テキストパッセージのみを抽出し、リスト、表、ヘッダは無視する。長い連続したシーケンスを抽出するためには、Billion Word Benchmark (Chelba et al., 2013) のようなシャッフルされた文レベルのコーパスではなく、文書レベルのコーパスを使用することが重要である。

3.2 Fine-tuning BERT

Transformerの自己注意メカニズムは、適切な入力と出力を入れ替えることで、BERTが多くの下流タスク(単一のテキストまたはテキストペアを含むかどうかにかかわらず)をモデル化することができるので、微調整は簡単です。テキストペアを含むアプリケーションでは、Parikhら(2016);Seoら(2017)のような双方向クロスアテンションを適用する前に、テキストペアを独立してエンコードすることが一般的なパターンである。BERTは、自己注意で連結されたテキストペアを符号化することで、2つの文間の双方向交差注意を効果的に含むため、代わりに自己注意メカニズムを使用してこれら2つのステージを統一する。各タスクについて、タスク固有の入力と出力をBERTにプラグインし、すべてのパラメータをエンドツーエンドで微調整するだけである。

入力において、事前学習による文Aと文Bは、(1)言い換えの文ペア、(2)含意の仮説-前提ペア、(3)質問応答の質問-パッセージペア、(4)テキスト分類やシーケンスタギングにおける退化テキスト-*ペアに類似している。出力では、トークン表現はシーケンスタギングや質問応答などのトークンレベルのタスクのために出力層に供給され、[CLS]表現は含意や感情分析などの分類のために出力層に供給される。事前学習と比較して、微調整は比較的安価である。本論文の結果はすべて、全く同じ事前学習済みモデルから始めて、単一のCloud TPUで最大1時間、GPUで数時間再現することができます。

4 Experiments

本節では、11のNLPタスクに対するBERTの微調整の結果を示す。

4.1 GLUE

一般言語理解評価(GLUE)ベンチマーク(Wang et al., 2018a)は、多様な自然言語理解タスクのコレクションである。GLUE データセットの詳細な説明は付録 B.1 に含まれている。GLUE上で微調整を行うために、セクション3で説明したように入力シーケンス(単文または文のペアについて)を表現し、最初の入力トークン([CLS])に対応する最終隠れベクトル$C \in R^H$を集約表現として使用する。微調整の際に導入される唯一の新しいパラメータは分類層の重み$W \in R^{K \times H}$ 、ここでKはラベルの数である。C と W 、すなわち log(softmax(CW^T)) を用いて標準的な分類損失を計算する。

[7] 例えば、BERT SQuADモデルは、91.0%のDev F1スコアを達成するために、単一のCloud TPUで約30分で学習することができます。[8] https://gluebenchmark.com/faq.の(10)を参照。

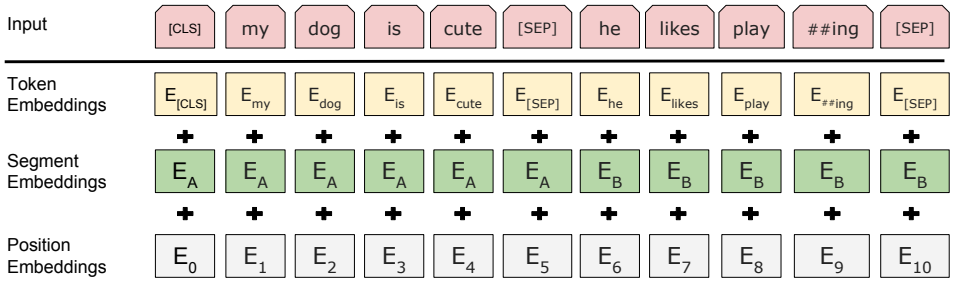

Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

The NSP task is closely related to representation-learning 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.

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.

3.2 Fine-tuning BERT

Fine-tuning is straightforward since the self-attention 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.

For each task, we simply plug in the task-specific inputs and outputs into BERT and fine-tune 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-$\varnothing$ pair in text classification or sequence tagging. At the output, the token representations are fed into an output layer for token-level 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.

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.

4 Experiments

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

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.

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 $C \in \mathbb{R}^H$ corresponding to the first input token (`[CLS]`) as the aggregate representation. The only new parameters introduced during fine-tuning are classification layer weights $W \in \mathbb{R}^{K \times H}$, where K is the number of labels. We compute a standard classification loss with C and W, i.e., $\log(\text{softmax}(CW^T))$.

[7] For example, the BERT SQuAD model can be trained in around 30 minutes on a single Cloud TPU to achieve a Dev F1 score of 91.0%.

[8] See (10) in `https://gluebenchmark.com/faq`.

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**

表 1:評価サーバ(https://gluebenchmark.com/leaderboard)で採点した GLUE テスト結果。各タスクの下の数字は、学習例数を表す。平均」列は、問題のあるWNLIセットを除外しているため、公式GLUEスコアとは若干異なっている。BERTとOpenAI GPTはシングルモデル、シングルタスクである。F1スコアはQQPとMRPC、Spearman相関はSTS-B、精度スコアはその他のタスクで報告されている。BERTを構成要素の一つとして使用するエントリは除外している。

バッチサイズは32を使用し、全てのGLUEタスクのデータに対して3エポック分の微調整を行った。各タスクにおいて、Devセット上で最適な微調整学習率(5e-5、4e-5、3e-5、2e-5)を選択した。さらに、BERT_{LARGE}では、小さなデータセットで微調整が不安定になることがあることがわかったので、いくつかのランダムリスタートを実行し、Devセットで最適なモデルを選択しました。ランダムリスタートでは、同じ事前学習済みチェックポイントを使用しますが、異なる微調整データのシャッフルと分類器層の初期化を実行します。BERT_{BASE}とBERT_{LARGE}は共に全てのタスクで全てのシステムを大幅に上回り、先行技術水準に対する平均精度がそれぞれ4.5%と7.0%向上した。なお、BERT_{BASE}とOpenAI GPTは、アテンションマスキングを除けば、モデルアーキテクチャの点でほぼ同じである。最大かつ最も広く報告されているGLUEタスクであるMNLIにおいて、BERTは4.6%の絶対精度向上を得ることができました。公式のGLUE leaderboard[10]では、BERT_{LARGE}は、執筆時点で72.8を得たOpenAI GPTと比較して、80.5のスコアを獲得しています。BERT_{LARGE}は、すべてのタスク、特に学習データが非常に少ないタスクにおいて、BERT_{BASE}を大幅に上回ることがわかりました。モデルサイズの効果については、5.2節でより詳細に検討する。

4.2 SQuAD v1.1

スタンフォード質問応答データセット(SQuAD v1.1)は、100kのクラウドソーシングされた質問/回答ペアのコレクションである(Rajpurkar et al.、2016)。からの質問と一節が与えられるとWikipediaに答えが含まれている場合、その文章中の答えのテキストスパンを予測することがタスクとなります。

As shown in Figure 1, in the question answer-タスクでは、入力された質問と文章を1つのパックシーケンスとして表現し、質問にはA埋め込みを、文章にはB埋め込みを用いる。微調整の際には、開始ベクトル$S \in R^H$と終了ベクトル$E \in R^H$のみを導入する。単語iが回答スパンの開始となる確率は、T_iとSの内積として計算され、その後、段落内の全ての単語に対するソフトマックスが行われる。$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$.

回答スパンの終了には、類似の式が使用される。位置iから位置jまでの候補スパンのスコアは、$S \cdot T_i + E \cdot T_j$ 、および$j \geq i$を予測として使用する最大スコアリングスパンと定義される。学習目的は、正しい開始位置と終了位置の対数尤度の和である。学習率5e-5、バッチサイズ32で3エポックの微調整を行う。

Table 2 shows top leaderboard entries as well は、トップが公開したシステム(Seo et al., 2017; Clark and Gardner, 2018; Peters et al., 2018a; Hu et al., 2018)の結果として、。SQuADリーダーボードのトップ結果は、最新の公開システム説明がない[11]、およびそのシステムを訓練する際に任意の公開データを使用することが許可されています。したがって、我々は、SQuAD上の微調整のために、まずTriviaQA(Joshi et al.、2017)beで微調整することによって、我々のシステムで控えめなデータ増強を使用します。

Our best performing system outperforms the top を用いた場合、アンサンブルで+1.5 F1、シングルシステムで+1.3 F1の向上が見られた。実際、我々の単一BERTモデルは、F1スコアの点でトップアンサンブルシステムを上回った。TriviaQAを用いない場合

[9] GLUEデータセットの配布にはTestラベルが含まれず、BERT_{BASE}とBERT_{LARGE}それぞれについてGLUE評価サーバを1台ずつ提出したのみである.

[10] https://gluebenchmark.com/leaderboard

[11] QANetはYuら(2018)に記載されているが、発表後に大幅に改善されたシステムである。

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.

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 $\text{BERT}_{\text{LARGE}}$ we found that fine-tuning 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.[9]

Results are presented in Table 1. Both $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ 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 $\text{BERT}_{\text{BASE}}$ 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 leaderboard[10], $\text{BERT}_{\text{LARGE}}$ obtains a score of 80.5, compared to OpenAI GPT, which obtains 72.8 as of the date of writing.

We find that $\text{BERT}_{\text{LARGE}}$ significantly outperforms $\text{BERT}_{\text{BASE}}$ across all tasks, especially those with very little training data. The effect of model size is explored more thoroughly in Section 5.2.

4.2 SQuAD v1.1

The Stanford Question Answering Dataset (SQuAD v1.1) is a collection of 100k crowd-sourced 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.

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 $S \in \mathbb{R}^H$ and an end vector $E \in \mathbb{R}^H$ during fine-tuning. The probability of word i being the start of the answer span is computed as a dot product between T_i and S followed by a softmax over all of the words in the paragraph: $P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$. 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 $S \cdot T_i + E \cdot T_j$, and the maximum scoring span where $j \geq i$ 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.

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.

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 fine-

[9] The GLUE data set distribution does not include the Test labels, and we only made a single GLUE evaluation server submission for each of $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$.

[10] https://gluebenchmark.com/leaderboard

[11] QANet is described in Yu et al. (2018), but the system has improved substantially after publication.

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**

表 2: SQuAD 1.1 の結果。BERTアンサンブルは、異なる事前学習用チェックポイントと微調整用シードを使用する7xシステムである。

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

表 3: SQuAD 2.0 の結果。BERTを構成要素の一つとして使用するエントリーは除外している。

チューニングデータでは、0.1-0.4 F1しか失わず、既存のすべてのシステムを大幅に上回ります。

4.3 SQuAD v2.0

SQuAD 2.0タスクは、SQuAD 1.1の問題定義を拡張し、提供された段落に短い答えが存在しない可能性を考慮し、問題をより現実的なものにするものである。我々は、このタスクのためにSQuAD v1.1 BERTモデルを拡張する簡単なアプローチを使用する。我々は、答えを持たない質問を、[CLS]トークンで開始と終了を持つ答えスパンを持つものとして扱う。開始と終了の回答スパン位置の確率空間は、[CLS]トークンの位置を含むように拡張される。予測のために、無回答スパンのスコア:s_{null} = S-C + E-Cと最良の非Nullスパンのスコアを比較する。

[12] 使用したTriviaQAデータは、TriviaQA-Wikiから、提供された可能な答えの少なくとも1つを含む、文書中の最初の400トークンから成る段落で構成されています。

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

表 4: SWAG Dev とテスト精度.† 人間の性能は、SWAG 論文で報告されているように、100 サンプルで測定されています。

s $\triangle_{i,j}$ = $\max_{j \geq i}$ S-T_i + E-T_j. ここで、閾値τはF1を最大化するようにdevセット上で選択される。このモデルにはTriviaQAデータを使用しない。学習率5e-5、バッチサイズ48で2エポックの微調整を行った。先行するリーダーボードエントリーやトップ公開の仕事(Sun et al., 2018; Wang et al., 2018b)と比較した結果を表3に示すが、その構成要素の一つとしてBERTを使用するシステムは除外している。以前のベストシステムに対して+5.1のF1の改善が観察される。

4.4 SWAG

SWAG(Situations With Adversarial Generations)データセットには、grounded commonsense推論を評価する113k文対補完例が含まれている(Zellers et al.、2018)。文が与えられると、タスクは4つの選択肢の中から最も妥当な継続を選択することである。SWAGデータセットで微調整を行う場合、与えられた文の連結(文A)と可能な継続(文B)を含む4つの入力シーケンスを構築する。導入される唯一のタスク固有のパラメータは、[CLS]トークン表現Cとのドット積が各選択肢のスコアを表し、ソフトマックス層で正規化されたベクトルのみである。学習率2e-5、バッチサイズ16で3エポック分のモデル微調整を行った。結果は表 4 に示す。BERT_{LARGE}は、著者のベースラインESIM+ELMoシステムを+27.1%、OpenAI GPTを8.3%上回った。

5 Ablation Studies

本節では、BERT の相対的な重要性をより理解するために、BERT のいくつかの側面についてアブレーション実験を行う。

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.

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.

tuning data, we only lose 0.1-0.4 F1, still outperforming all existing systems by a wide margin.[12]

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.

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: $s_{\texttt{null}} = S{\cdot}C + E{\cdot}C$ to the score of the best non-null span $\hat{s_{i,j}} = \max_{j \geq i} S{\cdot}T_i + E{\cdot}T_j$. We predict a non-null answer when $\hat{s_{i,j}} > s_{\texttt{null}} + \tau$, where the threshold τ 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.

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.

[12] The TriviaQA data we used consists of paragraphs from TriviaQA-Wiki formed of the first 400 tokens in documents, that contain at least one of the provided possible answers.

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.

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.

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.

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. $\text{BERT}_{\text{LARGE}}$ outperforms the authors' baseline ESIM+ELMo system by +27.1% and OpenAI GPT by 8.3%.

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

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$BERT_{BASE}$	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

表5:$BERT_{BASE}$アーキテクチャを用いた事前学習タスクに対するアブレーション。「No NSP ”は、次の文の予測タスクを行わずに学習させたものである。「LTR & No NSP ”は、OpenAI GPTのように、次の文の予測を行わず、左から右へのLMとして学習させたものである。”+ BiLSTM ”は、微調整の際に ”LTR + No NSP ”モデルの上にランダムに初期化されたBiLSTMを追加する。

アブレーション研究の概要は付録Cを参照されたい。

5.1 Effect of Pre-training Tasks

さらに、全く同じ事前学習データ、微調整スキーム、およびハイパーパラメータを $BERT_{BASE}$ として使用して 2 つの事前学習目標を評価することにより、BERT の深い双方向性の重要性を実証する。

NSPなし:「マスク付きLM」(MLM)を用いて学習するが、「次文予測」(NSP)タスクを用いない双方向モデル。LTR & No NSP: MLMではなく、標準的なLTR(Left-to-Right)LMを用いて学習させた左文脈のみのモデル。また、左のみの制約を除去すると、事前学習/微調整のミスマッチが発生し、下流の性能が低下するため、微調整の際にも制約を適用した。さらに、このモデルはNSPタスクなしで事前学習された。これはOpenAI GPTと直接似ていますが、より大きな学習データセット、入力表現、微調整のスキームを使用しています。まず、NSPタスクがもたらす影響について検証します。表5では、NSPを削除すると、QNLI、MNLI、SQuAD 1.1において、性能が大きく低下することを示しています。次に、「NSPなし」と「LTR & No NSP」を比較し、双方向表現の学習の影響を評価する。LTRモデルは全てのタスクでMLMモデルより性能が悪く、MRPCとSQuADでは大きく低下している。SQuADの場合、トークンレベルの隠れ状態には右側の文脈がないため、LTRモデルはトークン予測で性能が低下することが直感的にわかる。そこで、LTRシステムを強化するために、ランダムに初期化したBiLSTMを上に追加しました。

これにより、SQuADの結果は大幅に改善されましたが、事前学習した双方向モデルの結果よりもまだはるかに悪い結果となっています。BiLSTMはGLUEタスクの性能を低下させる。

We recognize that it would also be possible to は、LTRとRTLを別々に学習し、ELMoと同様に各トークンを2つのモデルの連結として表現する。しかし、(a)これは単一の双方向モデルの2倍のコストである。(b)RTLモデルは質問に対する答えを条件付けることができないので、QAのようなタスクではこれは直感的でない。 (c)これはすべての層で左右両方の文脈を使用できるので、深い双方向モデルよりも厳密には強力でない。

5.2 Effect of Model Size

本節では、モデルサイズがタスクの微調整精度に与える影響について検討する。我々は、層数、隠れユニット、注意ヘッドが異なる多数のBERTモデルを、それ以外は前述と同じハイパーパラメータと学習手順を用いて学習させた。

Results on selected GLUE tasks are shown in 表 6. この表では、微調整の5回のランダムな再スタートによる平均Dev Set精度を報告しています。3,600のラベル付き学習例しかなく、事前学習タスクと大きく異なるMRPCでも、より大きなモデルは4つのデータセットすべてにおいて厳密な精度向上をもたらすことがわかります。また、既存の文献と比較して既にかなり大きいモデルの上に、このような大幅な改善を達成できることは驚くべきことかもしれません。例えば、Vaswaniら(2017)で探求された最大のTransformerは、エンコーダのパラメータが100Mの(L=6、H=1024、A=16)であり、文献で見つかった最大のTransformerは、235Mのパラメータを持つ(L=64、H=512、A=2)です(Al-Rfouら、2018)。対照的に、$BERT_{BASE}$は110Mのパラメータを含み、$BERT_{LARGE}$は340Mのパラメータを含んでいます。

It has long been known that increasing the モデルサイズは、機械翻訳や言語モデリングなどの大規模タスクにおいて継続的な改善をもたらすことが、表6に示す学習データのLMパープレキシティによって実証されている。しかし、極端なモデルサイズに拡張することで、モデルが十分に事前学習されていれば、非常に小規模なタスクでも大きな改善につながることを説得的に示したのは、これが最初の仕事であると我々は考えている。

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.

ablation studies can be found in Appendix C.

5.1 Effect of Pre-training Tasks

We demonstrate the importance of the deep bidirectionality of BERT by evaluating two pre-training objectives using exactly the same pre-training data, fine-tuning scheme, and hyperparameters as $\text{BERT}_{\text{BASE}}$:

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

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.

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.

For SQuAD it is intuitively clear that a LTR model will perform poorly at token predictions, since the token-level hidden states have no right-side 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 pre-trained bidirectional models. The BiLSTM hurts performance on the GLUE tasks.

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.

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.

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, $\text{BERT}_{\text{BASE}}$ contains 110M parameters and $\text{BERT}_{\text{LARGE}}$ contains 340M parameters.

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

Petersら(2018b)は、事前学習したbi-LMサイズを2層から4層に増やした場合の下流タスクの影響について様々な結果を示し、Melamudら(2016)は、隠れ次元サイズを200から600に増やしても効果があったが、さらに1000に増やしてもさらなる改善はもたらさなかったと一応言及している。これらの先行研究はいずれも特徴ベースのアプローチを用いている。我々は、モデルを下流のタスクで直接微調整し、ランダムに初期化された非常に少数の追加パラメータのみを使用する場合、タスク固有のモデルは、下流のタスクデータが非常に小さい場合でも、より大きく表現力の高い事前学習済み表現の恩恵を受けることができると仮定している。

5.3 Feature-based Approach with BERT

これまで紹介したBERTの結果はすべて、事前学習済みモデルに単純な分類層を追加し、下流のタスクですべてのパラメータを共同で微調整するファインチューニングアプローチを用いている。しかし、特徴量ベースのアプローチでは、事前学習済みモデルから固定的な特徴を抽出するため、一定の利点がある。第一に、全てのタスクがTransformerエンコーダアーキテクチャで容易に表現できるわけではないため、タスクに特化したモデルアーキテクチャを追加する必要がある。第二に、一度学習データの高価な表現を事前に計算し、その表現の上に安価なモデルで多くの実験を実行することには、大きな計算上の利点がある。本節では、CoNLL-2003 Named Entity Recognition (NER)タスクにBERTを適用し、2つのアプローチを比較する(Tjong Kim Sang and De Meulder, 2003)。BERTへの入力では、ケース保存WordPieceモデルを使用し、データから提供される最大文書コンテキストを含める。標準的な手法に従い、タグ付けタスクとして定式化するが、CRFは使用しない。

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

表 6:BERT モデルサイズに対するアブレーション。#L = 層数、#H = 隠れサイズ、#A = 注意ヘッドの数。”LM (ppl)”は、ホールドアウトされた学習データのマスクされたLMパープレキシティである。

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	-

表 7: CoNLL-2003 名称付き固有表現認識結果。ハイパーパラメータはDevセットを用いて選択した。報告されたDevとTestのスコアは、これらのハイパーパラメータを用いた5回のランダムリスタートの平均値である。

層の出力である。NERラベルセットに対するトークンレベル分類器の入力として、最初のサブトークンの表現を使用する。

微調整のアプローチをアブレーションするために、BERTのパラメータを微調整することなく、1つ以上の層から活性度を抽出することにより、特徴ベースのアプローチを適用します。これらの文脈埋め込みは、分類層の前にランダムに初期化された2層768次元BiLSTMの入力として使用されます。

結果は表 7 に示されている。BERT_{LARGE}は、最先端の手法と競争力のある性能を発揮する。最も性能の良い方法は、事前に学習したTransformerの上位4つの隠れ層からトークン表現を連結するもので、モデル全体の微調整に0.3F1しか差がない。これは、BERTが微調整と特徴ベースのアプローチの両方に有効であることを示している。

6 Conclusion

近年の言語モデルによる転移学習による経験的な改善により、教師なし事前学習が多くの言語理解システムに不可欠であることが実証されている。特に、これらの結果は、低リソースタスクでさえ、深い一方向アーキテクチャの恩恵を受けることを可能にする。我々の主な貢献は、これらの知見を深い双方向アーキテクチャにさらに一般化し、同じ事前学習済みモデルが幅広いNLPタスクにうまく取り組むことを可能にすることである。

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 feature-based 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 task-specific models can benefit from the larger, more expressive pre-trained representations even when downstream task data is very small.

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 pre-trained 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.

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.

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.

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.

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.

Results are presented in Table 7. $\text{BERT}_{\text{LARGE}}$ 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 fine-tuning and feature-based approaches.

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.

References

アラン・アクベック、ダンカン・ブライス、ローランド・ヴォルグラフ。2018. シーケンスラベリングのための文脈的文字列埋め込み。第27回国際計算言語学会議論文集，ページ1638-1649.

ラミ・アル・ルフー、ドゥクック・チョ、ノア・コンスタント、マンディ・グオ、リリオン・ジョーンズ。2018. より深い自己注意を用いた文字レベルの言語モデリング.arXiv preprint arXiv:1808.0444.

安藤理恵・張同. 2005. 複数のタスクとラベル無しデータから予測構造を学習するためのフレームワーク. 機械学習研究、6(Nov):1817-1853.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. 第5回PASCALによるテキスト含意の課題認識。TACにて。NISTにて。

ジョン・ブリッツァー、ライアン・マクドナルド、フェルナンド・ペレイラ。2006. 構造的対応学習による領域適応。自然言語処理における経験的手法に関する2006年会議論文集，ページ120-128. 計算言語学協会.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. 自然言語推論を学習するための大規模アノテーションコーパス。EMNLPにて。計算言語学協会.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. 自然言語のクラスベースN-gramモデル. 計算言語学, 18(4):467-479.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez Gazpio, and Lucia Specia. 2017. Semeval-2017タスク1:意味的テキスト類似性多言語・異言語集中評価. 第11回意味評価に関する国際ワークショップ(SemEval-2017)の議事録、ページ1-14、バンクーバー、カナダ。Association for Computational Linguistics(計算言語学会)。

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013.

Z. Chen, H. Zhang, X. Zhang, and L. Zhao. 2018. Quora question pairs.

クリストファー・クラーク、マット・ガードナー。2018. シンプルで効果的なマルチパラグラフ読解。ACLにて。

ケビン・クラーク、ミン・チャン・ルオン、クリストファー・D・マニング、クオック・レ. 2018. クロスビュー学習による半教師付きシーケンスモデリング。自然言語処理における経験的方法に関する2018年会議の議事録、ページ1914-1925にて。

Ronan Collobert と Jason Weston. 2008. 自然言語処理のための統一アーキテクチャ。マルチタスク学習によるディープニューラルネットワーク. 第25回機械学習国際会議論文集，ページ160-167. ACM.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Lo_wc Barrault, and Antoine Bordes. 2017. 自然言語推論データからの普遍文表現の教師あり学習。2017年自然言語処理における経験的方法に関する会議の議事録、ページ670-680、コペンハーゲン、デンマーク。Association for Computational Linguistics(計算言語学会)。

アンドリュー・M・ダイ、クオック・V・レ。2015. 半教師付きシーケンス学習. 神経情報処理システムの進歩，ページ 3079-3087 にて。

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. FeiFei. 2009. ImageNet: 大規模階層的画像データベース. CVPR09にて。

ウィリアム・B・ドーラン、クリス・ブロケット。2005. 文言言い換えのコーパスを自動的に構築する。第3回言い換えに関する国際ワークショップ(IWP2005)予稿集.

ウィリアム・フェドゥス、イアン・グッドフェロー、アンドリュー・M・ダイ。2018. Maskgan:. arXiv preprint arXiv:1801.07736を埋めることによるより良いテキスト生成。

ダン・ヘンドリクスとケビン・ギンペル. 2016. 非線形性と確率的正則化をガウス誤差線形単位で橋渡しする。CoRR, abs/1606.08415.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. ラベルのないデータから文の分散表現を学習する。計算言語学会の北米支部の2016年大会の議事録にて。Human Language Technologies. Association for Computational Linguistics.

ジェレミー・ハワードとセバスチャン・ルーダー. 2018. テキスト分類のための普遍的な言語モデルの微調整。ACLにて。Association for Computational Linguistics(計算言語学会)。

胡明浩、Peng Yuxing、黄正、斉新栄、Wei Furu、周明。2018. 機械読解のための強化されたニーモニックリーダー。IJCAIにて。

Yacine Jernite, Samuel R. Bowman, and David Sontag. 2017. 教師なし文表現学習のための談話ベースの目的. CoRR, abs/1705.00557.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2018. Character-level language modeling with deeper self-attention. *arXiv preprint arXiv:1808.04444*.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*. NIST.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*. Association for Computational Linguistics.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Z. Chen, H. Zhang, X. Zhang, and L. Zhao. 2018. Quora question pairs.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *ACL*.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: Better text generation via filling in the_. *arXiv preprint arXiv:1801.07736*.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*. Association for Computational Linguistics.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In *IJCAI*.

Yacine Jernite, Samuel R. Bowman, and David Sontag. 2017. Discourse-based objectives for fast unsupervised sentence representation learning. *CoRR*, abs/1705.00557.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: 読解のための大規模な遠隔監視付きチャレンジデータセット. ACLにて。

ライアン・キロス、朱玉邦、ラスラン・R・サラフティノフ、リチャード・ゼメル、ラケル・ウルタスン、アントニオ・トルラバ、サンジャ・フィドラー。2015. スキップソートベクトル。神経情報処理システムの進歩、ページ3294-3302にて。

クオック・レとトマス・ミコロフ. 2014. 文と文書の分散表現. In International Conference on Machine Learning, pages 1188-1196.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. Winogradスキーマの課題。青井春季シンポジウムにて。コモンセンス推論の論理的形式化、第46巻、47ページ。

ラジャンゲン・ロジェスワラン、ホンラック・リー。2018. 文の表現を学習するための効率的なフレームワーク。In International Conference on Learning Representations.

ブライアン・マッキャン、ジェームズ・ブラッドベリー、カイミン・シオン、リチャード・ソッチャー. 2017. 並進で学習する。文脈に応じた単語ベクトル。NIPSにて。

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: 双方向LSTMを用いた汎用コンテキスト埋め込みの学習. CoNLLにて。

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. 単語とフレーズの分散表現とその構成性。神経情報処理システムにおける進歩 26 ページ 3111-3119. Curran Associates, Inc.

Andriy Mnih and Geoffrey E Hinton. 2009. スケーラブルな階層的分散言語モデル。このような場合,そのような言語モデルを利用することができる。Curran Associates, Inc.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. 自然言語推論のための分解可能な注意モデル。EMNLPにて。

ジェフリー・ペニントン、リチャード・ソッチャー、クリストファー・D・マニング. 2014. Glove: 単語表現のためのグローバルベクトル. 自然言語処理における経験的方法(EMNLP)、ページ1532-1543にて。

マシュー・ピータース、ウォルエ・アムマル、チャンドラ・バガヴァトゥーラ、ラッセル・パワー. 2017. 双方向言語モデルによる半教師付きシーケンスタギング。ACLにて。

マシュー・ピータース、マーク・ノイマン、モフィット・アイイヤー、マット・ガードナー、クリストファー・クラーク、ケントン・リー、ルーク・ゼトルマイヤー。2018a. ディープコンテクスト化された単語表現。NAACLにて。

マシュー・ピータース、マーク・ノイマン、ルーク・ゼットルモイヤー、ウェンタウ・イー・。2018b. 文脈的な単語埋め込みを解剖する。アーキテクチャと表現。自然言語処理における経験的方法に関する2018年会議の議事録、ページ1499-1509にて。

アレック・ラドフォード、カルティク・ナラシムハン、ティム・サリマンス、イリヤ・スツケバー。2018. 教師なし学習による言語理解の改善。テクニカルレポート、OpenAI。

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: テキストの機械理解のための100,000以上の質問。自然言語処理における経験的方法に関する2016年会議の議事録、ページ2383から2392に記載されています。

瀬尾明男、Aniruddha Kembhavi、Ali Farhadi、Hannaneh Hajishirzi. 2017. 機械理解のための双方向注意フロー。ICLRにて。

リチャード・ソッチャー、アレックス・ペレリギン、ジャン・ウー、ジェイソン・チャン、クリストファー・D・マニング、アンドリュー・Ng、クリストファー・ポッツ。2013. センチメントツリーバンク上の意味的構成性のための再帰的なディープモデル。自然言語処理における経験的手法に関する2013年会議論文集, ページ1631-1642.

孫 麗 麗 暁・斉 陽. 2018. U-net: 回答不能な質問による機械読解.arXiv preprint arXiv:1810.06638.

Wilson L Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Bulletin*, 30(4):415–433.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. conll-2003共有タスクの紹介。言語非依存型名前付き固有表現認識. CoNLLにて。

ジョセフ・トゥリアン、レフ・ラチノフ、ヨシュア・ベンジオ。2010. 単語表現。半教師付き学習のためのシンプルで一般的な方法。第48回計算言語学会年次大会予稿集, ACL '10, pages 384-394.

アシシュ・ヴァスワニ、ノーム・シャゼール、ニキ・パルマー、ヤコブ・ウズコレイト、リオン・ジョーンズ、アダン・N・ゴメス、ルカスツァイザー、イリア・ポロスチン。2017. アテンションがあればいい。神経情報処理システムの進歩、ページ6000-6010にて。

パスカル・ヴィンセント、ヒューゴ・ラロシェル、ヨシュア・ベンジオ、ピエール・アントイネ・マンザゴール。2008. ノイズ除去オートエンコーダを用いたロバストな特徴の抽出と合成。第25回機械学習国際会議講演論文集, ページ 1096-1103. ACM.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In *Aaai spring symposium: Logical formalizations of commonsense reasoning*, volume 46, page 47.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *NIPS*.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *CoNLL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1081–1088. Curran Associates, Inc.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *ACL*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *NAACL*.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Fu Sun, Linyang Li, Xipeng Qiu, and Yang Liu. 2018. U-net: Machine reading comprehension with unanswerable questions. *arXiv preprint arXiv:1810.06638*.

Wilson L Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Bulletin*, 30(4):415–433.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. Glue: A multi-task benchmark and analysis platform

自然言語理解のための Proceedings of the 2018 EMNLP Workshop BlackboxNLP: NLPのためのニューラルネットワークの解析と解釈、ページ353-355にて。

Samuel Bowman. 2018a. Glue: マルチタスクベンチマークと解析プラットフォーム Wei Wang, Ming Yan, and Chen Wu. 2018b. 読解と質問応答のための多階層階層的注意融合ネットワーク。Association for Computational Linguistics(計算言語学会)。

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. ニューラルネットワークの受容性判断.arXiv preprint arXiv:1805.12471.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. 推論による文理解のための広範なカバレッジのチャレンジコーパス。NAACLにて。

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al.2016年. Googleのニューラル機械翻訳システム。人間翻訳と機械翻訳のギャップを埋める。

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. ディープニューラルネットワークにおける特徴量はどの程度転送可能か?神経情報処理システムの進歩, ページ 3320-3328.

アダムス・ウェイ・ユイ、デビッド・ドーハン、ミン・チャン・ルオン、ルイ・ザオ、カイ・チェン、モハマド・ノロウジ、クオック・V・レ. 2018. QANet: 読解のための局所畳み込みとグローバル自己注意の組合せ. ICLRにて。

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: 接地されたコモンセンス推論のための大規模な敵対的データセット。自然言語処理における経験的方法(EMNLP)に関する2018年会議の議事録にて。

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. 本と映画のアライメント。映画を見たり本を読んだりすることで、ストーリーのような視覚的説明に向けて。IEEE国際コンピュータビジョン会議論文集, ページ19-27.

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

付録を3つのセクションに整理する。

- BERT の追加実装の詳細は、付録 A に記載されている。
- 実験の詳細は付録 B に示す。
- Additional ablation studies are presented in Appendix C.

  We present additional ablation studies for BERT including:

  - 学習ステップ数の影響、および、異なるマスキング手順に対するアブレーション。

    dures.

A Additional Details for BERT

A.1 事前学習タスクの説明図

以下に、事前学習タスクの例を示す。

マスキングLMとマスキング手順 ラベルのない文が私の犬の毛であると仮定し、ランダムマスキング手順で4番目のトークン(毛に対応する)を選びましたが、我々のマスキング手順はさらに次のように説明できます。

- 80%の確率で 単語を[MASK]トークンに置き換える、例:私の犬は毛深い →。

  `my dog is [MASK]`

- 10%の確率で 単語をランダムな単語に置き換える 例:私の犬は毛深い → 私の

  `dog is apple`

- 10%の確率で 単語を変更しない、例えば、私の犬は毛深い → 私の犬は毛深い。その目的は、実際に観測された単語に表現を偏らせることである。

この方法の利点は、Transformerエンコーダがどの単語を予測するよう求められるのか、あるいはどの単語がランダムな単語に置き換えられたのかを知らないため、すべての入力トークンの分布的文脈表現を維持せざるを得ないことである。さらに、ランダムな置換は全トークンの1.5%(すなわち15%のうち10%)しか起こらないので、これはモデルの言語理解能力を損なわないようである。セクションC.2において、この手順の影響を評価する。標準的な言語モデルの学習と比較して、マスクされたLMは各バッチのトークンの15%に対してのみ予測を行うため、モデルが収束するためにはより多くの事前学習ステップが必要である可能性が示唆される。

for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Wei Wang, Ming Yan, and Chen Wu. 2018b. Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. In *ICLR*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

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

We organize the appendix into three sections:

- Additional implementation details for BERT are presented in Appendix A;
- Additional details for our experiments are presented in Appendix B; and
- Additional ablation studies are presented in Appendix C.

  We present additional ablation studies for BERT including:

  – Effect of Number of Training Steps; and
  – Ablation for Different Masking Procedures.

A Additional Details for BERT

A.1 Illustration of the Pre-training Tasks

We provide examples of the pre-training tasks in the following.

Masked LM and the Masking Procedure Assuming the unlabeled sentence is `my dog is hairy`, and during the random masking procedure we chose the 4-th token (which corresponding to `hairy`), our masking procedure can be further illustrated by

- 80% of the time: Replace the word with the `[MASK]` token, e.g., `my dog is hairy` → `my dog is [MASK]`
- 10% of the time: Replace the word with a random word, e.g., `my dog is hairy` → `my dog is apple`
- 10% of the time: Keep the word unchanged, e.g., `my dog is hairy` → `my dog is hairy`. The purpose of this is to bias the representation towards the actual observed word.

The advantage of this procedure is that the Transformer encoder does not know which words it will be asked to predict or which have been replaced by random words, so it is forced to keep a distributional contextual representation of *every* input token. Additionally, because random replacement only occurs for 1.5% of all tokens (i.e., 10% of 15%), this does not seem to harm the model's language understanding capability. In Section C.2, we evaluate the impact this procedure.

Compared to standard langauge model training, the masked LM only make predictions on 15% of tokens in each batch, which suggests that more pre-training steps may be required for the model

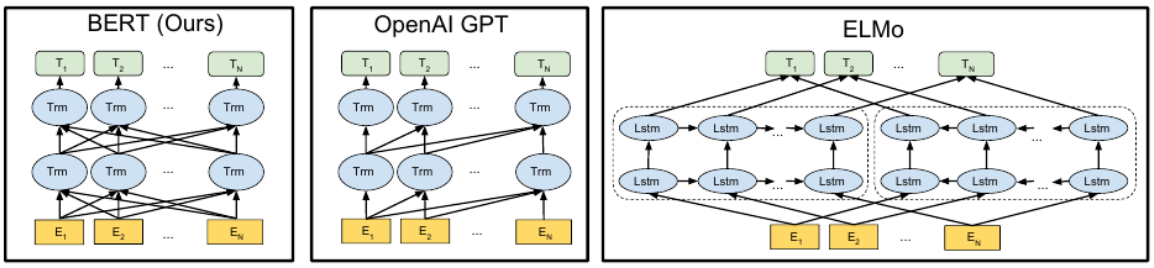

図3:事前学習モデルアーキテクチャの違い。BERTは双方向のTransformerを使用している。OpenAI GPTは左から右へのTransformerを使用している。ELMoは、独立に学習した左右のLSTMと右から左へのLSTMを連結して、下流タスクの特徴を生成する。この3つのうち、BERT表現のみが、全層で左右両方の文脈を条件としている。アーキテクチャの違いに加え、BERTとOpenAI GPTは微調整のアプローチであり、ELMoは特徴ベースのアプローチである。

セクションC.1では、MLMが左から右へのモデル(すべてのトークンを予測する)よりもわずかに遅い収束を示すが、MLMモデルの経験的改善は学習コストの増加をはるかに上回るものである。

次文予測 次文予測タスクは、以下の例で説明することができる。

Input = [CLS] MASK]店に行った。
彼はガロン(MASK)の牛乳を[SEP]で購入した。
Label = IsNext

Input = [CLS] 男[MASK]から店[SEP]へ
lka(MASK)は無兵鳥(SEP)です。
Label = NotNext

A.2 Pre-training Procedure

各トレーニング入力列を生成するために、コーパスから2つのスパンのテキストをサンプリングする。これは、一般的に単一文よりずっと長い(しかし、より短いこともある)にもかかわらず、「文」と呼ぶことにする。最初の文はAの埋め込みを受け、2番目の文はBの埋め込みを受ける。Bの50%はAに続く実際の次の文であり、50%はランダムな文であるが、これは「次の文の予測」タスクのために行われる。これらは、結合された長さが≦512トークンになるようにサンプリングされる。LMマスキングはWordPieceトークン化後に一律に15%のマスキング率で適用され、部分的な単語片には特に考慮されない。バッチサイズ256シーケンス(256シーケンス×512トークン=128,000トークン/バッチ)で100万ステップの学習を行い、これは33億語コーパスに対して約40エポックである。学習率1e-4、$\beta_1 = 0.9$, $\beta_2 = 0.999$, L2 weight decay of 0のAdamを使用する。01、最初の10,000ステップでの学習率ウォームアップ、学習率の線形減衰。全ての層でドロップアウト確率を0.1とする。OpenAI GPTに準じ、標準的なreluではなく、gelu activation (Hendrycks and Gimpel, 2016)を使用する。学習損失は、平均マスクLM尤度と平均次文予測尤度の和である。BERT$_{BASE}$の学習は、Pod構成の4つのCloud TPU(合計16 TPUチップ)に対して行った。[13] BERT$_{LARGE}$の学習は、16のCloud TPU(合計64 TPUチップ)に対して実施した。各プリトレーニングは4日間で完了した。長い配列は、注意が配列長の2次式であるため、不釣り合いに高価である。我々の実験では、プリトレートを高速化するために、配列長128で90%のステップでモデルを事前学習させた。そして、512の配列の残りの10%のステップを学習し、位置の埋め込みを学習する。

A.3 Fine-tuning Procedure

微調整のため、バッチサイズ、学習率、学習エポック数を除き、ほとんどのモデルのハイパーパラメータは事前学習と同じである。ドロップアウト確率は常に0.1に保たれた。最適なハイパーパラメータ値はタスクに特化したものですが、以下の範囲で全てのタスクでうまく機能することがわかりました。

- **Batch size**: 16, 32

[13] https://cloudplatform.googleblog.com/2018/06/Cloud-TPU-now-offers-preemptible-pricing-and-global-availability.html

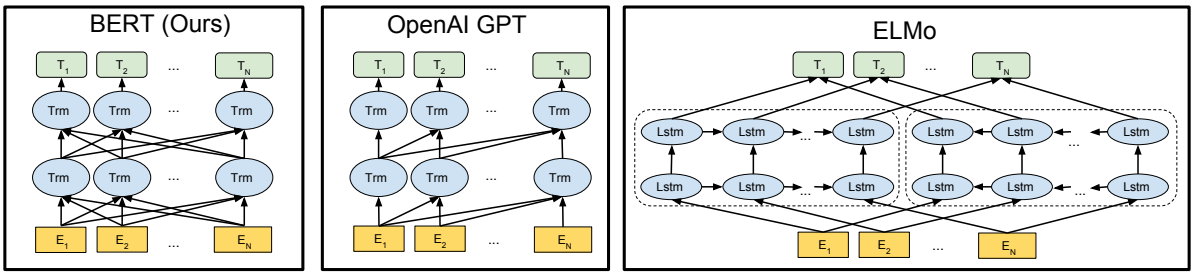

Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

to converge. In Section C.1 we demonstrate that MLM does converge marginally slower than a left-to-right model (which predicts every token), but the empirical improvements of the MLM model far outweigh the increased training cost.

Next Sentence Prediction The next sentence prediction task can be illustrated in the following examples.

Input = `[CLS] the man went to [MASK] store [SEP]`
`he bought a gallon [MASK] milk [SEP]`
Label = `IsNext`

Input = `[CLS] the man [MASK] to the store [SEP]`
`penguin [MASK] are flight ##less birds [SEP]`
Label = `NotNext`

A.2 Pre-training Procedure

To generate each training input sequence, we sample two spans of text from the corpus, which we refer to as "sentences" even though they are typically much longer than single sentences (but can be shorter also). The first sentence receives the `A` embedding and the second receives the `B` embedding. 50% of the time `B` is the actual next sentence that follows `A` and 50% of the time it is a random sentence, which is done for the "next sentence prediction" task. They are sampled such that the combined length is $\leq$ 512 tokens. The LM masking is applied after WordPiece tokenization with a uniform masking rate of 15%, and no special consideration given to partial word pieces.

We train with batch size of 256 sequences (256 sequences * 512 tokens = 128,000 tokens/batch) for 1,000,000 steps, which is approximately 40 epochs over the 3.3 billion word corpus. We use Adam with learning rate of 1e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$, L2 weight decay of 0.01, learning rate warmup over the first 10,000 steps, and linear decay of the learning rate. We use a dropout probability of 0.1 on all layers. We use a `gelu` activation (Hendrycks and Gimpel, 2016) rather than the standard `relu`, following OpenAI GPT. The training loss is the sum of the mean masked LM likelihood and the mean next sentence prediction likelihood.

Training of $\text{BERT}_{\text{BASE}}$ was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total).[13] Training of $\text{BERT}_{\text{LARGE}}$ was performed on 16 Cloud TPUs (64 TPU chips total). Each pre-training took 4 days to complete.

Longer sequences are disproportionately expensive because attention is quadratic to the sequence length. To speed up pretraing in our experiments, we pre-train the model with sequence length of 128 for 90% of the steps. Then, we train the rest 10% of the steps of sequence of 512 to learn the positional embeddings.

A.3 Fine-tuning Procedure

For fine-tuning, most model hyperparameters are the same as in pre-training, with the exception of the batch size, learning rate, and number of training epochs. The dropout probability was always kept at 0.1. The optimal hyperparameter values are task-specific, but we found the following range of possible values to work well across all tasks:

- **Batch size**: 16, 32

[13] https://cloudplatform.googleblog.com/2018/06/Cloud-TPU-now-offers-preemptible-pricing-and-global-availability.html

- **Learning rate (Adam)**: 5e-5, 3e-5, 2e-5
- **Number of epochs**: 2, 3, 4

また、大規模なデータセット(例えば、100k以上のラベル付き学習例)は、小規模なデータセットよりもハイパーパラメータの選択に対する感度がはるかに低いことが観察された。微調整は一般的に非常に速いので、上記のパラメータを網羅的に探索し、開発セットで最も良い性能を発揮するモデルを選択するのが合理的である。

A.4 Comparison of BERT, ELMo ,and OpenAI GPT

ここでは、ELMo、OpenAI GPT、BERT などの最近の一般的な表現学習モデルの違いについて研究している。モデルアーキテクチャ間の比較を図 3 に視覚的に示す。アーキテクチャの違いに加え、BERTとOpenAI GPTはファインチューニングアプローチであり、ELMoは特徴ベースアプローチであることに注意されたい。BERTと最も比較可能な既存の事前学習方法はOpenAI GPTであり、大規模なテキストコーパスに対して左から右への変換器LMを学習させるものである。実際、BERTの設計上の決定の多くは、GPTにできるだけ近づけるように意図的に行われており、2つの方法を最小限に比較することができました。本研究の中核となる論点は、セクション 3.1 で示した双方向性と 2 つの事前学習タスクが経験的改善の大部分を占めるということですが、BERT と GPT の学習方法には他にもいくつかの違いがあることに留意してください。

- GPTはBooksCorpus(800Mワード)、BERTはBooksCorpus(800Mワード)、Wikipedia(2,500Mワード)で学習させる。

- GPTは、微調整時にのみ導入される文区切り([SEP])と分類器トークン([CLS])を使用し、BERTは事前学習時に[SEP]、[CLS]、文A/Bの埋め込みを学習する。

- GPTはバッチサイズ32,000ワードで1Mステップ、BERTはバッチサイズ128,000ワードで1Mステップの学習を行った。

- GPTはすべての微調整実験に同じ5e-5の学習率を使用した。BERTは開発セットで最も良い性能を発揮するタスク固有の微調整学習率を選択した。

これらの違いの影響を分離するために、セクション5.1でアブレーション実験を行い、改善の大部分が実際には2つの事前学習タスクとそれらが可能にする双方向性からもたらされていることを実証した。

A.5 Illustrations of Fine-tuning on Different Tasks

図 4 は、異なるタスクにおける BERT の微調整の説明図である。我々のタスク固有のモデルは、1つの追加出力層を持つBERTを組み込むことで形成されているため、最小限のパラメータをゼロから学習する必要がある。タスクのうち、(a)と(b)はシーケンスレベルのタスクであり、(c)と(d)はトークンレベルのタスクである。図中、Eは入力の埋め込み、T_iはトークンiの文脈表現、[CLS]は分類出力の特殊記号、[SEP]は非連続トークン列を分離する特殊記号である。

B Detailed Experimental Setup

B.1 Detailed Descriptions for the GLUE Benchmark Experiments.

Our GLUE results in Table1 are obtained from https://gluebenchmark.com/leaderboard and https://blog.openai.com/language-unsupervised. The GLUE benchmark includes the following datasets, the descriptions of which were originally summarized in Wang et al. (2018a):

MNLI Multi-Genre Natural Language Inferenceは、大規模なクラウドソーシングによる含意分類タスクである(Williams et al.、2018)。一対の文が与えられたとき、ゴールは2番目の文が最初の文に関して含意、矛盾、または中立であるかどうかを予測することである。

QQP Quora Question Pairsは、Quoraで尋ねられた2つの質問が意味的に等価であるかどうかを判断することを目的とした二値分類タスクである(Chen et al.、2018)。

QNLI Question Natural Language Inferenceは、Stanford Question Answering Dataset (Rajpurkar et al., 2016) のバージョンで、二値分類タスク (Wang et al., 2018a) に変換されています。正例は正解を含む(質問、文)ペア、負例は答えを含まない同じ段落からの(質問、文)である。

- **Learning rate (Adam)**: 5e-5, 3e-5, 2e-5
- **Number of epochs**: 2, 3, 4

We also observed that large data sets (e.g., 100k+ labeled training examples) were far less sensitive to hyperparameter choice than small data sets. Fine-tuning is typically very fast, so it is reasonable to simply run an exhaustive search over the above parameters and choose the model that performs best on the development set.

A.4 Comparison of BERT, ELMo ,and OpenAI GPT

Here we studies the differences in recent popular representation learning models including ELMo, OpenAI GPT and BERT. The comparisons between the model architectures are shown visually in Figure 3. Note that in addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

The most comparable existing pre-training method to BERT is OpenAI GPT, which trains a left-to-right Transformer LM on a large text corpus. In fact, many of the design decisions in BERT were intentionally made to make it as close to GPT as possible so that the two methods could be minimally compared. The core argument of this work is that the bi-directionality and the two pre-training tasks presented in Section 3.1 account for the majority of the empirical improvements, but we do note that there are several other differences between how BERT and GPT were trained:

- GPT is trained on the BooksCorpus (800M words); BERT is trained on the BooksCorpus (800M words) and Wikipedia (2,500M words).
- GPT uses a sentence separator (`[SEP]`) and classifier token (`[CLS]`) which are only introduced at fine-tuning time; BERT learns `[SEP]`, `[CLS]` and sentence `A`/`B` embeddings during pre-training.
- GPT was trained for 1M steps with a batch size of 32,000 words; BERT was trained for 1M steps with a batch size of 128,000 words.
- GPT used the same learning rate of 5e-5 for all fine-tuning experiments; BERT chooses a task-specific fine-tuning learning rate which performs the best on the development set.

To isolate the effect of these differences, we perform ablation experiments in Section 5.1 which demonstrate that the majority of the improvements are in fact coming from the two pre-training tasks and the bidirectionality they enable.

A.5 Illustrations of Fine-tuning on Different Tasks

The illustration of fine-tuning BERT on different tasks can be seen in Figure 4. Our task-specific models are formed by incorporating BERT with one additional output layer, so a minimal number of parameters need to be learned from scratch. Among the tasks, (a) and (b) are sequence-level tasks while (c) and (d) are token-level tasks. In the figure, E represents the input embedding, T_i represents the contextual representation of token i, [CLS] is the special symbol for classification output, and [SEP] is the special symbol to separate non-consecutive token sequences.

B Detailed Experimental Setup

B.1 Detailed Descriptions for the GLUE Benchmark Experiments.

Our GLUE results in Table1 are obtained from `https://gluebenchmark.com/leaderboard` and `https://blog.openai.com/language-unsupervised`. The GLUE benchmark includes the following datasets, the descriptions of which were originally summarized in Wang et al. (2018a):

MNLI Multi-Genre Natural Language Inference is a large-scale, crowdsourced entailment classification task (Williams et al., 2018). Given a pair of sentences, the goal is to predict whether the second sentence is an *entailment*, *contradiction*, or *neutral* with respect to the first one.

QQP Quora Question Pairs is a binary classification task where the goal is to determine if two questions asked on Quora are semantically equivalent (Chen et al., 2018).

QNLI Question Natural Language Inference is a version of the Stanford Question Answering Dataset (Rajpurkar et al., 2016) which has been converted to a binary classification task (Wang et al., 2018a). The positive examples are (question, sentence) pairs which do contain the correct answer, and the negative examples are (question, sentence) from the same paragraph which do not contain the answer.

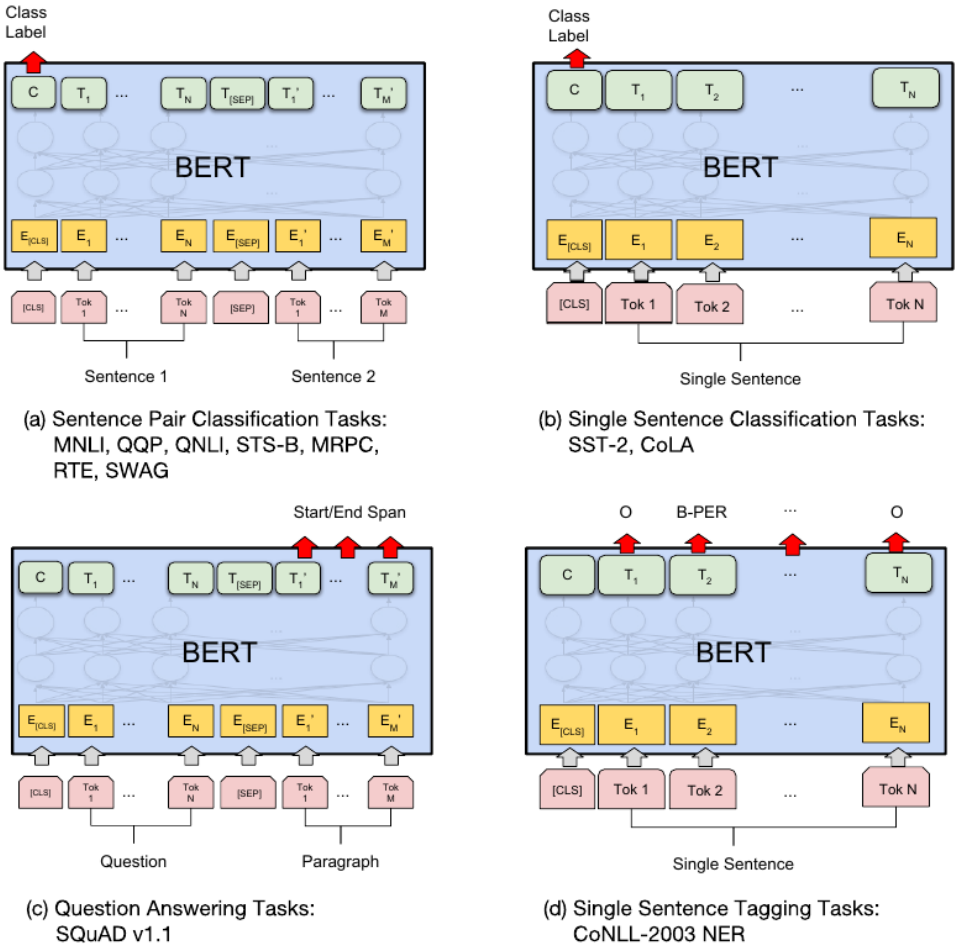

図4:異なるタスクにおけるBERTの微調整の説明図。

SST-2 Stanford Sentiment Treebankは、映画レビューから抽出された文とその人間の感情のアノテーションからなる2値1文分類タスクである(Socher et al.、2013)。

CoLA 言語的受容性のコーパスは、英語の文が言語的に「受容可能」かどうかを予測することを目的とした、二値単文分類タスクである(Warstadt et al.、2018)。

STS-B Semantic Textual Similarity Benchmarkは、ニュースヘッドラインや他のソースから抽出した文のペアのコレクションである(Cer et al.、2017)。彼らは、2つの文が意味的な意味においてどの程度似ているかを示す1から5までのスコアでアノテーションされた。

MRPC Microsoft Research Paraphrase Corpus は、オンラインニュースソースから自動的に抽出された文ペアと、そのペアの文が意味的に等価であるかどうかを人間がアノテーションしたものである (Dolan and Brockett, 2005)。

RTE 認識テキストエンタイルメントは、MNLI と同様の二値含意タスクであるが、学習データが非常に少ない (Bentivogli et al., 2009).[14].

WNLI Winograd NLIは、小規模な自然言語推論データセットである(Levesque et al., 2011)。GLUEのウェブページでは、このデータセットの構築に問題がある[15]と、GLUEに提出されたすべての学習システムは、多数決クラスを予測する65.1ベースライン精度より悪い結果を示していることを指摘しています。そのため、OpenAI GPTではこのセットを公平に除外しています。GLUE提出では、常にma-を予測した。

[14] なお、本論文ではシングルタスクの微調整の結果のみを報告する。マルチタスク微調整のアプローチは、性能をさらに押し上げる可能性がある。例えば、MNLIを用いたマルチタスク学習により、RTEの大幅な向上が確認された。

[15] https://gluebenchmark.com/faq

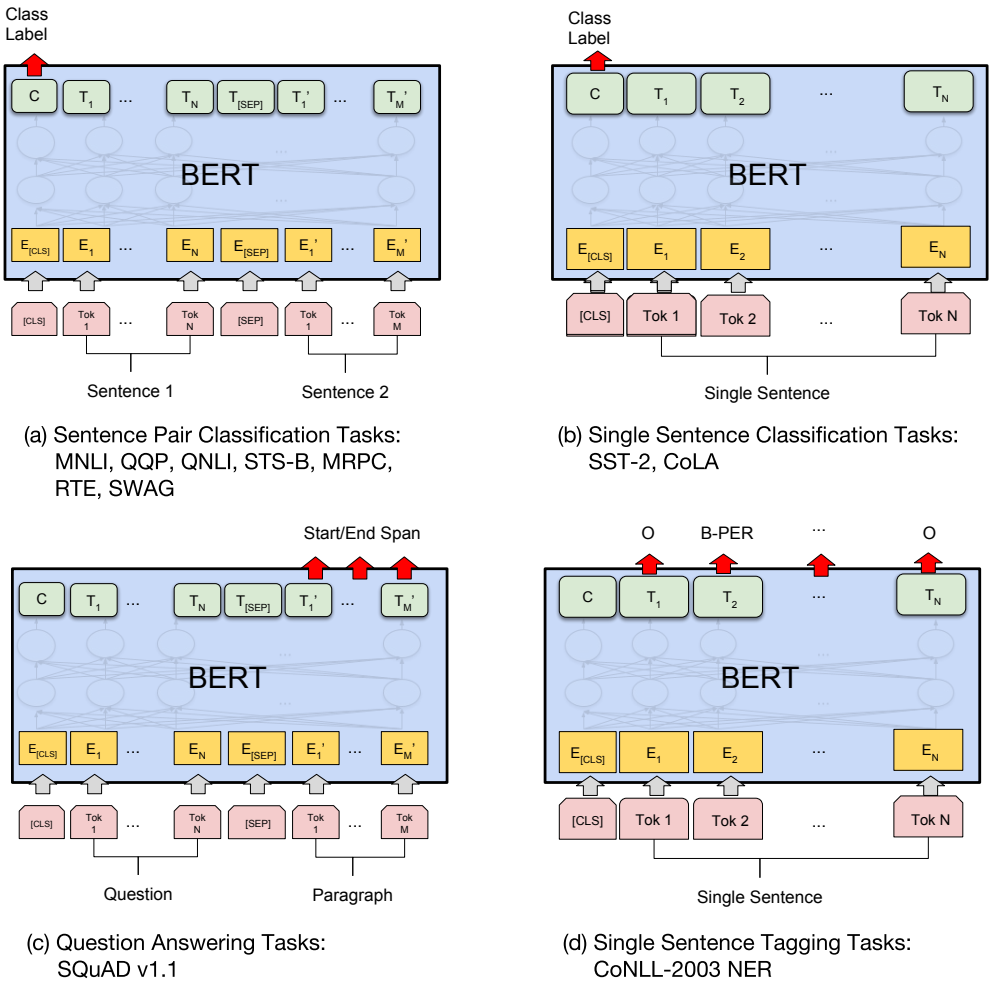

Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

SST-2 The Stanford Sentiment Treebank is a binary single-sentence classification task consisting of sentences extracted from movie reviews with human annotations of their sentiment (Socher et al., 2013).

CoLA The Corpus of Linguistic Acceptability is a binary single-sentence classification task, where the goal is to predict whether an English sentence is linguistically "acceptable" or not (Warstadt et al., 2018).

STS-B The Semantic Textual Similarity Benchmark is a collection of sentence pairs drawn from news headlines and other sources (Cer et al., 2017). They were annotated with a score from 1 to 5 denoting how similar the two sentences are in terms of semantic meaning.

MRPC Microsoft Research Paraphrase Corpus consists of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent (Dolan and Brockett, 2005).

RTE Recognizing Textual Entailment is a binary entailment task similar to MNLI, but with much less training data (Bentivogli et al., 2009).[14]

WNLI Winograd NLI is a small natural language inference dataset (Levesque et al., 2011). The GLUE webpage notes that there are issues with the construction of this dataset, [15] and every trained system that's been submitted to GLUE has performed worse than the 65.1 baseline accuracy of predicting the majority class. We therefore exclude this set to be fair to OpenAI GPT. For our GLUE submission, we always predicted the ma-

[14]Note that we only report single-task fine-tuning results in this paper. A multitask fine-tuning approach could potentially push the performance even further. For example, we did observe substantial improvements on RTE from multi-task training with MNLI.

[15]https://gluebenchmark.com/faq

jority class.

C Additional Ablation Studies

C.1 トレーニングステップ数の効果

図5は、kステップの事前学習済みのチェックポイントから微調整を行った後のMNLI Devの精度を示しています。これにより、以下のような疑問に答えることができる。

1. 質問 BERTは、高い微調整精度を得るために、本当に大量の事前学習(128,000ワード/バッチ×100,000ステップ)が必要なのでしょうか?回答 はい、BERT_{BASE}は、500kステップに対して1Mステップで学習した場合、MNLIでほぼ1.0%の精度を追加で達成しています。

2. 質問です。MLMの事前学習はLTRの事前学習よりも収束が遅いか?なぜなら、全ての単語ではなく、各バッチで予測される単語は15%に過ぎないからである。回答 MLMモデルはLTRモデルより若干遅く収束する。しかし、絶対精度の面では、MLMモデルはLTRモデルよりほぼ即座に性能が向上し始める。

C.2 Ablation for Different Masking Procedures

セクション3.1では、BERTがマスク言語モデル(MLM)目的語を用いて事前学習する際に、ターゲットトークンのマスキングに混合戦略を用いていることに言及する。以下は、異なるマスキング戦略の効果を評価するためのアブレーションスタディである。

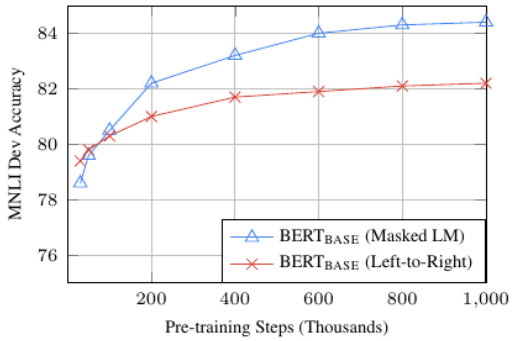

図5:学習ステップ数に対するアブレーション。これは、kステップの事前学習済みのモデルパラメータから、微調整後のMNLI精度を示すものである。x軸はkの値である。

なお、マスキング戦略の目的は、[MASK]シンボルが微調整の段階で現れないため、事前学習と微調整のミスマッチを減らすことである。MNLIとNERの両方についてDevの結果を報告する。NERについては、モデルが表現を調整する機会を持たないため、特徴ベースアプローチではミスマッチが増幅されると予想されるため、微調整と特徴ベースアプローチの両方を報告する。

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

表8:異なるマスキング戦略に対するアブレーション。

結果は表 8 に示すとおりである。表中、M ASKはターゲットトークンをMLMの[MASK]記号に置き換えることを意味し、S AMEはターゲットトークンをそのまま残すことを意味し、R NDはターゲットトークンを別のランダムトークンに置き換えることを意味する。表の左側の数字は、MLMの事前学習時に使用した特定の戦略の確率を表しています(BERTは80%、10%、10%を使用)。本論文の右側は、Devセットの結果を表しています。特徴ベースアプローチでは、BERTの最後の4層を特徴として連結し、セクション5.3で最良のアプローチであることが示された。表から、微調整は異なるマスキング戦略に対して驚くほどロバストであることがわかる。しかし、予想通り、NERに特徴ベースアプローチを適用する場合、M ASK戦略のみを使用することは問題であった。興味深いことに、R ND戦略のみを使用すると、我々の戦略よりもはるかに悪いパフォーマンスを示す。

jority class.

C Additional Ablation Studies

C.1 Effect of Number of Training Steps

Figure 5 presents MNLI Dev accuracy after fine-tuning from a checkpoint that has been pre-trained for k steps. This allows us to answer the following questions:

1. Question: Does BERT really need such a large amount of pre-training (128,000 words/batch * 1,000,000 steps) to achieve high fine-tuning accuracy?
   Answer: Yes, $\text{BERT}_{\text{BASE}}$ achieves almost 1.0% additional accuracy on MNLI when trained on 1M steps compared to 500k steps.

2. Question: Does MLM pre-training converge slower than LTR pre-training, since only 15% of words are predicted in each batch rather than every word?
   Answer: The MLM model does converge slightly slower than the LTR model. However, in terms of absolute accuracy the MLM model begins to outperform the LTR model almost immediately.

C.2 Ablation for Different Masking Procedures

In Section 3.1, we mention that BERT uses a mixed strategy for masking the target tokens when pre-training with the masked language model (MLM) objective. The following is an ablation study to evaluate the effect of different masking strategies.

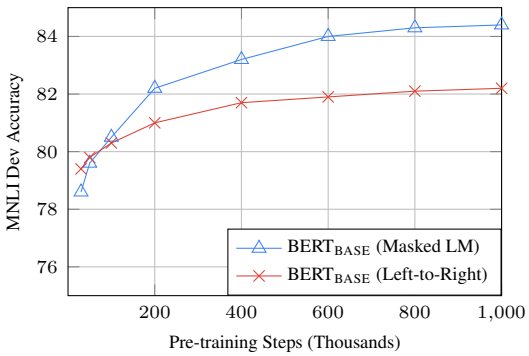

Figure 5: Ablation over number of training steps. This shows the MNLI accuracy after fine-tuning, starting from model parameters that have been pre-trained for k steps. The x-axis is the value of k.

Note that the purpose of the masking strategies is to reduce the mismatch between pre-training and fine-tuning, as the `[MASK]` symbol never appears during the fine-tuning stage. We report the Dev results for both MNLI and NER. For NER, we report both fine-tuning and feature-based approaches, as we expect the mismatch will be amplified for the feature-based approach as the model will not have the chance to adjust the representations.

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

Table 8: Ablation over different masking strategies.

The results are presented in Table 8. In the table, MASK means that we replace the target token with the `[MASK]` symbol for MLM; SAME means that we keep the target token as is; RND means that we replace the target token with another random token.

The numbers in the left part of the table represent the probabilities of the specific strategies used during MLM pre-training (BERT uses 80%, 10%, 10%). The right part of the paper represents the Dev set results. For the feature-based approach, we concatenate the last 4 layers of BERT as the features, which was shown to be the best approach in Section 5.3.

From the table it can be seen that fine-tuning is surprisingly robust to different masking strategies. However, as expected, using only the MASK strategy was problematic when applying the feature-based approach to NER. Interestingly, using only the RND strategy performs much worse than our strategy as well.