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

BERT：用于语言理解的深度双向Transformer预训练

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova 谷歌AI语言

{jacobdevlin,mingweichang,kentonl,kristout}@google.com

摘要

我们介绍了一种名为BERT的新语言表示模型，BERT代表来自Transformer的双向编码器表示。与近期的语言表示模型不同（Peters等人，2018a；Radford等人，2018），BERT旨在通过在所有层中联合调节左右上下文，从未标记文本中预训练深度双向表示。因此，预训练的BERT模型仅需添加一个额外的输出层进行微调，即可为广泛的任务（如问答和语言推理）创建最先进的模型，而无需对任务特定架构进行实质性修改。

BERT在概念上简单，但在实证中表现出强大的能力。它在十一个自然语言处理任务中取得了新的最先进成果，包括将GLUE分数提升至80.5%（绝对提升7.7个百分点）、MultiNLI准确率提升至86.7%（绝对提升4.6%）、SQuAD v1.1问答测试F1值达到93.2（绝对提升1.5分）以及SQuAD v2.0测试F1值达到83.1（绝对提升5.1分）。

1 引言

语言模型预训练已被证明能有效提升多种自然语言处理任务的性能（Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018）。这包括句子级任务（如自然语言推理（Bowman et al., 2015; Williams et al., 2018）和复述检测（Dolan and Brockett, 2005）），这类任务通过整体分析句子以预测其关联关系；同时也涵盖词元级任务（如命名实体识别与问答系统），此类任务要求模型在词元级别生成细粒度输出（Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016）。

将预训练语言表示应用于下游任务现有两种策略：*feature-based*和*fine-tuning*。基于特征的方法，如ELMo（Peters等人，2018a），采用包含预训练表示作为附加特征的特定任务架构。而微调方法，如生成式预训练Transformer（OpenAI GPT）（Radford等人，2018），则引入极少的任务特定参数，并通过直接微调*all*预训练参数在下游任务上进行训练。这两种方法在预训练阶段共享相同的目标函数，即使用单向语言模型来学习通用语言表示。

我们认为，当前的技术限制了预训练表征的潜力，尤其是在微调方法中。主要局限在于标准语言模型是单向的，这限制了预训练时可用的架构选择。例如，在OpenAI GPT中，作者采用了从左到右的架构，其中每个标记在Transformer的自注意力层（Vaswani等人，2017）中只能关注先前的标记。这种限制对于句子级任务来说并非最优，而在将基于微调的方法应用于标记级任务（如问答）时可能极为不利，因为在这些任务中，双向上下文的整合至关重要。

在本文中，我们通过提出BERT：来自Transformer的双向编码器表示，改进了基于微调的方法。BERT通过使用“掩码语言模型”（MLM）预训练目标来缓解之前提到的单向性限制，该目标受到完形填空任务（Taylor, 1953）的启发。掩码语言模型随机掩盖输入中的部分标记，其目标是预测被掩盖标记的原始词汇ID。

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-

仅基于其上下文预测单词。与从左到右的语言模型预训练不同，MLM目标使表示能够融合左右上下文，这使我们能够预训练一个深度双向Transformer。除了掩码语言模型外，我们还使用“下一句预测”任务，联合预训练文本对表示。本文的贡献如下：

- 我们证明了双向预训练对于语言表征的重要性。与Radford等人（2018）使用单向语言模型进行预训练不同，BERT采用掩码语言模型来实现预训练的深度双向表征。这也与Peters等人（2018a）形成对比，后者仅将独立训练的左到右和右到左语言模型进行浅层拼接。

- 我们证明，预训练表征减少了对许多精心设计的任务特定架构的需求。BERT是首个基于微调的表征模型，在大量句子级*and*和词元级任务上实现了最先进的性能，超越了众多任务特定的架构。

- BERT在十一个自然语言处理任务上取得了最先进的成果。代码和预训练模型可在https://github.com/google-research/bert获取。

2 相关工作

预训练通用语言表示有着悠久的历史，本节我们将简要回顾最广泛使用的方法。

2.1 基于无监督特征的方法

学习广泛适用的词语表示是几十年来一个活跃的研究领域，包括非神经方法（Brown等人，1992；Ando和Zhang，2005；Blitzer等人，2006）和神经方法（Mikolov等人，2013；Pennington等人，2014）。预训练的词嵌入是现代自然语言处理系统的核心组成部分，相比从零开始学习的嵌入能带来显著提升（Turian等人，2010）。为预训练词嵌入向量，研究者采用了从左到右的语言建模目标（Mnih和Hinton，2009），以及区分左右上下文中正确与错误词语的目标（Mikolov等人，2013）。

这些方法已被推广到更粗的粒度，例如句子嵌入（Kiros等人，2015；Logeswaran与Lee，2018）或段落嵌入（Le与Mikolov，2014）。为了训练句子表示，先前的研究采用了多种目标函数：对候选下一句子进行排序（Jernite等人，2017；Logeswaran与Lee，2018）、在给定前一句子表示的情况下从左到右生成下一句子的词语（Kiros等人，2015），或基于去噪自编码器的衍生目标（Hill等人，2016）。

ELMo及其前身（Peters等人，2017，2018a）从不同维度拓展了传统词嵌入研究。它们从左至右和从右至左的语言模型中提取*context-sensitive*特征，每个标记的上下文表示是左右双向表示的拼接。将上下文词嵌入与现有任务特定架构结合时，ELMo在多项重要NLP基准测试中实现了技术突破（Peters等人，2018a），包括问答系统（Rajpurkar等人，2016）、情感分析（Socher等人，2013）和命名实体识别（Tjong Kim Sang与De Meulder，2003）。Melamud等人（2016）提出通过LSTM从左右上下文预测单个单词的任务来学习上下文表示。与ELMo类似，他们的模型基于特征且非深度双向。Fedus等人（2018）研究表明完形填空任务可用于提升文本生成模型的鲁棒性。

2.2 无监督微调方法

与基于特征的方法一样，这一方向的首批研究仅从无标注文本中预训练词嵌入参数（Collobert和Weston，2008）。

最近，能够生成上下文标记表示的句子或文档编码器已通过无标签文本进行预训练，并针对有监督的下游任务进行微调（Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018）。这类方法的优势在于只需从头学习少量参数。至少部分得益于这一优势，OpenAI GPT（Radford et al., 2018）在GLUE基准测试（Wang et al., 2018a）的许多句子级任务中取得了当时最先进的结果。从左到右的语言模型——

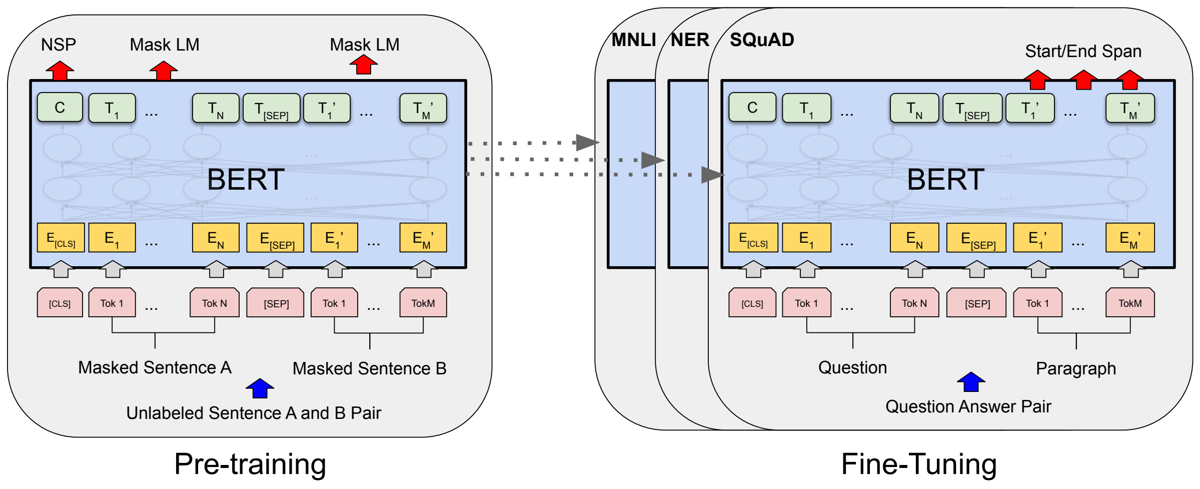


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are 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: **BERT$_{\text{BASE}}$** (L=12, H=768, A=12, Total Parameters=110M) and **BERT$_{\text{LARGE}}$** (L=24, H=1024, A=16, Total Parameters=340M).

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-

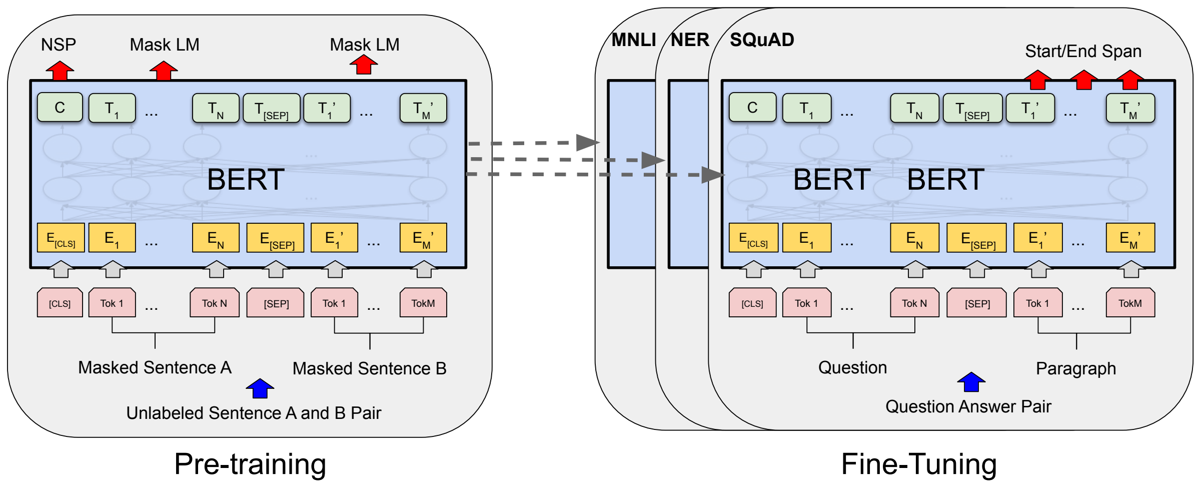


图1：BERT的整体预训练与微调流程。除了输出层外，预训练和微调使用相同的架构结构。相同的预训练模型参数被用于初始化不同下游任务的模型。在微调过程中，所有参数都会被调整。[CLS]是添加在每个输入样本前的特殊符号，而[SEP]是特殊的分隔符标记（例如用于分隔问题/答案）。

语言建模和自编码器目标已被用于此类模型的预训练（Howard和Ruder，2018；Radford等人，2018；Dai和Le，2015）。

2.3 从监督数据中进行迁移学习

已有研究显示，从拥有大型数据集的监督任务（如自然语言推理（Conneau等人，2017）和机器翻译（McCann等人，2017））中可以实现有效的迁移。计算机视觉研究同样证明了从大型预训练模型进行迁移学习的重要性，其中一种有效方法是对基于ImageNet（Deng等人，2009；Yosinski等人，2014）预训练的模型进行微调。

3 BERT

本节将介绍BERT及其具体实现。我们的框架包含两个步骤：*pre-training*和*fine-tuning*。在预训练阶段，模型通过不同的预训练任务在无标注数据上进行训练。在微调阶段，BERT模型首先用预训练参数初始化，随后使用下游任务的标注数据对所有参数进行微调。每个下游任务都有独立的微调模型，即使它们初始化的预训练参数相同。图1中的问答示例将作为本节的运行案例。

BERT的一个显著特点是其跨不同任务的统一架构。有微预训练架构与最终下游架构之间的微小差异。

模型架构 BERT的模型架构是一个基于Vaswani等人（2017）所描述并在tensor2tensor库中发布的原始实现的多层双向Transformer编码器。[1] 由于Transformer的使用已变得普遍，且我们的实现与原始版本几乎相同，我们将省略对模型架构的详尽背景描述，并建议读者参考Vaswani等人（2017）以及《The Annotated Transformer》等优秀指南。[2]

在本工作中，我们将层数（即Transformer块）表示为L，隐藏层大小表示为H，自注意力头数表示为A。[3]我们主要报告两种模型规模的结果：BERT$_{\mathbf{BASE}}$（L=12、H=768、A=12、总参数量=1.1亿），以及BERT$_{\mathbf{LARGE}}$（L=24、H=1024、A=16、总参数量=3.4亿)。

BERT$_{\mathrm{BASE}}$被选择为与OpenAI GPT具有相同的模型大小以便进行比较。然而，关键区别在于，BERT Transformer采用双向自注意力机制，而GPT Transformer则使用受限的自注意力机制，其中每个标记只能关注其左侧的上下文。[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-

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., $\langle$ Question, Answer $\rangle$) 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.

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.

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]

[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.

输入/输出表示　为了使BERT能够处理各种下游任务，我们的输入表示方法能够在一个标记序列中明确表示单个句子和句子对（例如，⟨问题，答案⟩）。在本研究中，“句子”可以是任意连续的文本片段，而非严格意义上的语言学句子。“序列”指的是输入到BERT的标记序列，它可以是单个句子，也可以是打包在一起的两个句子。

我们使用WordPiece嵌入（Wu等人，2016），词汇量为30,000个词元。每个序列的第一个词元始终是一个特殊的分类标记（[CLS]）。该标记对应的最终隐藏状态被用作分类任务的聚合序列表示。句子对被合并成一个单一序列。我们通过两种方式区分句子：首先，用特殊标记（[SEP]）分隔它们；其次，为每个词元添加一个可学习的嵌入，以指示其属于句子A还是句子B。如图1所示，我们将输入嵌入表示为E，特殊[CLS]标记的最终隐藏向量表示为$C \in \mathbb{R}^H$，而i^{th}输入词元的最终隐藏向量表示为$T_i \in \mathbb{R}^H$。

对于给定的标记，其输入表示是通过将对应的标记嵌入、段嵌入和位置嵌入相加来构建的。这种构建的可视化展示可见于图2。

3.1 预训练BERT

与Peters等人（2018a）和Radford等人（2018）不同，我们未使用传统的从左到右或从右到左语言模型对BERT进行预训练。相反，我们采用本节描述的两个无监督任务对BERT进行预训练。该步骤展示在图1的左侧部分。

任务 #1：掩码语言模型　直观上，我们有理由相信，深度双向模型严格优于从左到右的模型或从左到右与从右到左模型的浅层拼接。然而，标准的条件语言模型只能以从左到右 *or* 或从右到左的方式进行训练，因为双向条件作用会使每个词间接地“看到自身”，导致模型在多层上下文中可以轻易预测目标词。

为了训练深度双向表示，我们只需随机掩盖一定比例的输入标记，然后预测这些被掩盖的标记。尽管文献中常将此方法称为*Cloze*任务（Taylor, 1953），我们仍将其称为“掩码语言模型”（MLM）。在此过程中，对应于掩码标记的最终隐藏向量会像标准语言模型那样，被输入词汇表上的输出softmax层。在所有实验中，我们随机掩盖每个序列中15%的WordPiece标记。与去噪自编码器（Vincent等，2008）不同，我们仅预测被掩盖的词汇而非重构整个输入。

尽管这使我们能够获得一个双向预训练模型，但一个缺点是我们在预训练和微调之间造成了不匹配，因为[MASK]标记在微调期间不会出现。为了缓解这个问题，我们并不总是用实际的[MASK]标记替换“被遮蔽”的词语。训练数据生成器会随机选择15%的标记位置进行预测。如果选择了第i个标记，我们将第i个标记替换为：(1) 80%的情况下使用[MASK]标记；(2) 10%的情况下使用随机标记；(3) 10%的情况下保持第i个标记不变。随后，T_i将用于通过交叉熵损失预测原始标记。我们在附录C.2中比较了此过程的不同变体。

任务 #2：下一句预测（NSP）　许多重要的下游任务，如问答（QA）和自然语言推理（NLI），都基于理解两个句子之间的*rela-tionship*，而语言建模无法直接捕捉这种关系。为了训练一个能够理解句子关联的模型，我们针对一个二值化的*next sen-tence prediction*任务进行预训练，该任务可以轻松从任何单语语料库中生成。具体来说，在为每个预训练示例选择句子A和B时，50%的情况下B是实际紧随A的下一句（标记为IsNext），另外50%的情况下B是语料库中的随机句子（标记为NotNext）。如图1所示，C被用于下一句预测（NSP）。[5] 尽管这一任务设计简单，但我们在第5.1节中证明，针对该任务的预训练对QA和NLI都非常有益。[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.

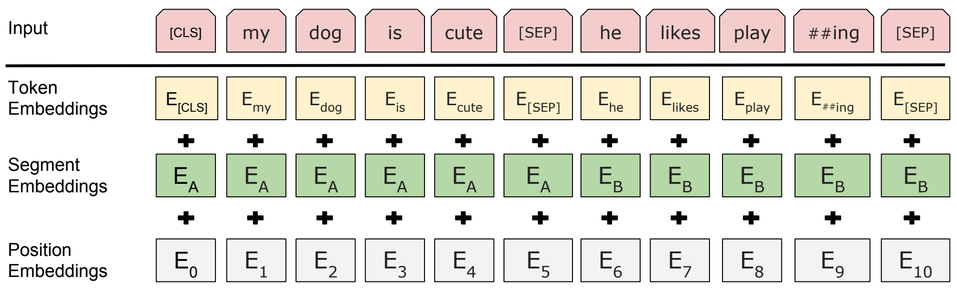

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
E[CLS] Emy Edog Eis Ecute E[SEP] Ehe Elikes Eplay E##ing E[SEP]
Segment Embeddings
EA EA EA EA EA EA EB EB EB EB EB
Position Embeddings
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10

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.

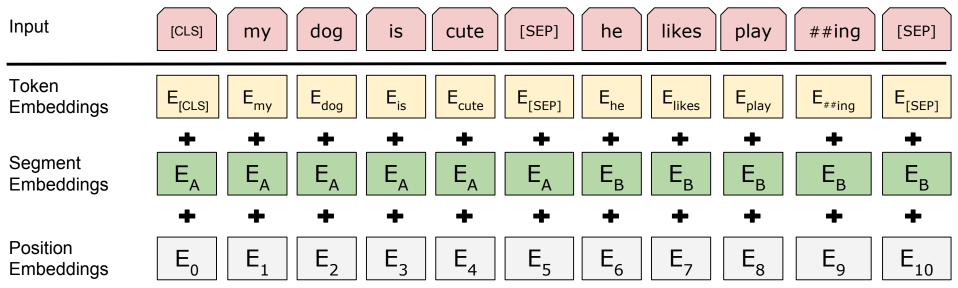
4.1 GLUE

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., $\log(\mathrm{softmax}(CW^T))$.

[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`.



Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

图2：BERT输入表示。输入嵌入是词元嵌入、分段嵌入和位置嵌入的总和。

NSP任务与Jernite等人（2017）以及Logeswaran和Lee（2018）使用的表示学习目标密切相关。然而，在先前的工作中，仅将句子嵌入迁移至下游任务，而BERT则迁移所有参数以初始化终端任务模型参数。

预训练数据 预训练过程主要遵循现有关于语言模型预训练的文献。对于预训练语料库，我们使用了BooksCorpus（8亿词）（Zhu等人，2015年）和英文维基百科（25亿词）。对于维基百科，我们仅提取文本段落，忽略列表、表格和标题。为了提取长连续序列，使用文档级语料库而非如Billion Word Benchmark（Chelba等人，2013年）这类经过打乱的句子级语料库至关重要。

3.2 微调BERT

微调过程十分直接，因为Transformer中的自注意力机制让BERT能够通过替换适当的输入和输出，来建模多种下游任务——无论涉及单个文本还是文本对。对于包含文本对的应用，一种常见模式是在应用双向交叉注意力之前独立编码文本对，例如Parikh等人（2016）和Seo等人（2017）的研究。而BERT则利用自注意力机制统一这两个阶段，因为通过自注意力对拼接文本对进行编码时，实际上已包含两个句子之间的*bidi-rectional*交叉注意力。

对于每个任务，我们只需将任务特定的输入和输出插入BERT，并端到端地微调所有参数。在输入层面，预训练中的句子A和句子B分别对应：（1）释义任务中的句子对，（2）蕴含任务中的假设-前提对，（3）问答任务中的问题-段落对，以及(4) 文本分类或序列标注中的一种退化文本-∅对。在输出端，词元表示被送入输出层以执行词元级任务，如序列标注或问答；而[CLS]表示则被送入输出层以进行分类任务，如蕴含判断或情感分析。

与预训练相比，微调的成本相对较低。本文中的所有结果都可以从完全相同的预训练模型出发，在单个Cloud TPU上至多1小时内复现，或在GPU上花费数小时完成。[7] 我们在第4节的相应小节中描述了任务特定的细节。更多细节可在附录A.5中找到。

4 实验

在本节中，我们展示了BERT在11项NLP任务上的微调结果。

4.1 GLUE

通用语言理解评估（GLUE）基准（Wang等人，2018a）是一个包含多样化自然语言理解任务的集合。GLUE数据集的详细描述见附录B.1。

为了在GLUE上进行微调，我们按照第3节所述的方式表示输入序列（针对单句或句对），并使用与第一个输入标记（[CLS]）对应的最终隐藏向量$C \in \mathbb{R}^H$作为聚合表示。微调过程中引入的唯一新参数是分类层权重$W \in \mathbb{R}^{K\times H}$，其中$K$是标签数量。我们使用$C$和$W$计算标准分类损失，即log(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)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
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
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
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 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 BERT$_{\text{BASE}}$ and 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 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], 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 BERT$_{\text{LARGE}}$ significantly outperforms 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 BERT$_{\text{BASE}}$ and 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	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
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
BERT$_{\mathrm{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\mathrm{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

表1：GLUE测试结果，由评估服务器（https://gluebenchmark.com/leaderboard）评分。每个任务下方的数字表示训练样本的数量。“平均”列与官方GLUE分数略有不同，因为我们排除了有问题的WNLI数据集。[8] BERT和OpenAI GPT是单模型、单任务系统。QQP和MRPC任务报告F1分数，STS-B任务报告斯皮尔曼相关系数，其余任务报告准确率分数。我们排除了使用BERT作为其组成部分的条目。

我们使用32的批次大小，并在所有GLUE任务的数据上进行了3个轮次的微调。针对每个任务，我们在开发集上选择了最佳的微调学习率（从5e-5、4e-5、3e-5和2e-5中选取）。此外，对于BERT$_{\mathrm{LARGE}}$，我们发现微调在小数据集上有时不稳定，因此我们进行了多次随机重启，并在开发集上选择了最佳模型。在随机重启过程中，我们使用相同的预训练检查点，但执行不同的微调数据洗牌和分类层初始化。[9]

结果如表1所示。BERT$_{\mathrm{BASE}}$和BERT$_{\mathrm{LARGE}}$在所有任务上均以显著优势超越所有系统，相较于先前的最优技术分别实现了4.5%和7.0%的平均准确率提升。需注意的是，除注意力掩码机制外，BERT$_{\mathrm{BASE}}$与OpenAI GPT的模型架构几乎完全相同。在规模最大且最常被引用的GLUE任务MNLI上，BERT取得了4.6%的绝对准确率提升。根据官方GLUE排行榜[10]截至撰写时的数据，BERT$_{\mathrm{LARGE}}$获得80.5分，而OpenAI GPT的得分为72.8。

我们发现，BERT$_{\mathrm{LARGE}}$在所有任务上的表现都显著优于BERT$_{\mathrm{BASE}}$，尤其是在训练数据非常有限的任务中。模型规模的影响将在第5.2节进行更深入的探讨。

4.2 SQuAD v1.1

斯坦福问答数据集（SQuAD {v*}1.1）是一个包含10万条众包问答对的数据集（Rajpurkar等人，2016）。给定一个问题及一篇来自包含答案的维基百科文章，任务是在文章中预测答案文本的范围。

如图1所示，在问答任务中，我们将输入问题与文本段落表示为单个打包序列，其中问题使用A嵌入表示，段落使用B嵌入表示。我们仅在微调阶段引入起始向量$S \in \mathbb{R}^H$和结束向量$E \in \mathbb{R}^H$。词语i作为答案跨度起始位置的概率，通过计算T_i与S的点积后，对段落中所有词语进行softmax归一化得到：$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$。答案跨度结束位置的计算采用类似公式。候选跨度从位置i到位置j的得分定义为$S{\cdot}T_i + E{\cdot}T_j$，并以$j \geq i$作为预测依据选取最高得分的跨度。训练目标为正确起始位置与结束位置的对数似然之和。我们采用5e-5的学习率和32的批大小进行3个周期的微调。

表2展示了排行榜前列的条目以及顶尖已发表系统的结果（Seo等人，2017；Clark与Gardner，2018；Peters等人，2018a；Hu等人，2018）。SQuAD排行榜上的顶尖结果目前尚无最新的公开系统描述可用，[11]且这些系统在训练时允许使用任何公开数据。因此，我们在系统中采用了适度的数据增强方法，即先在TriviaQA（Joshi等人，2017）上进行微调，再对SQuAD进行微调。

我们表现最佳的集成系统比排行榜上的顶级系统高出+1.5个F1值，而作为单一系统则高出+1.3个F1值。实际上，我们的单一BERT模型在F1分数上甚至超越了顶级的集成系统。在没有经过TriviaQA微调的情况下——

[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 BERT$_{\mathrm{BASE}}$ and BERT$_{\mathrm{LARGE}}$.

[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				
BERT$_{\text{BASE}}$ (Single)	80.8	88.5	-	-
BERT$_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
BERT$_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
BERT$_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
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				
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 $\hat{s_{i,j}} = \texttt{max}_{j \geq i} S{\cdot}T_i + E{\cdot}T_j$. 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
BERT$_{\text{BASE}}$	81.6	-
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. 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

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				
BERT$_{\text{BASE}}$ (Single)	80.8	88.5	-	-
BERT$_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
BERT$_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
BERT$_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
BERT$_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

表2：SQuAD 1.1结果。BERT集成系统包含7个使用不同预训练检查点与微调种子的系统。

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				
BERT$_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

表3：SQuAD 2.0 结果。我们排除了使用BERT作为其组件之一的条目。

在调优数据上，我们仅损失0.1-0.4 F1值，仍以显著优势超越所有现有系统。[12]

4.3 SQuAD v2.0

SQuAD 2.0任务扩展了SQuAD 1.1的问题定义，允许在提供的段落中可能不存在简短答案，从而使问题更加贴近现实。

我们采用一种简单的方法来扩展SQuAD v1.1 BERT模型以完成此任务。我们将没有答案的问题视为其答案跨度起始和结束于[CLS]标记。答案跨度起始和结束位置的概率空间被扩展至包含[CLS]标记的位置。在预测时，我们将无答案跨度的得分：$s_{\tt null} = S{\cdot}C + E{\cdot}C$与最佳非空跨度的得分进行比较。

[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
BERT$_{\text{BASE}}$	81.6	-
BERT$_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

表4：SWAG开发和测试准确率。†人类表现是通过100个样本测量的，如SWAG论文中所述。

$\hat{s_{i,j}}$ = $\mathtt{max}_{j \geq i} S{\cdot}T_i + E{\cdot}T_j$我们预测当$\hat{s_{i,j}} > s_{\tt null} + \tau$时存在非空答案，其中阈值$\tau$是在开发集上为最大化F1值而选定的。该模型未使用TriviaQA数据。我们以5e-5的学习率和48的批量大小进行了2个周期的微调。

与先前排行榜条目及顶尖已发表工作（Sun等人，2018；Wang等人，2018b）相比的结果展示于表3，其中排除了使用BERT作为组件的系统。我们观察到相较于先前最佳系统有+5.1 F1值的提升。

4.4 SWAG

对抗生成情境（SWAG）数据集包含113k个句子对补全示例，用于评估基于常识的推理能力（Zellers等人，2018）。给定一个句子，任务是从四个选项中选择最合理的后续内容。

在SWAG数据集上进行微调时，我们构建四个输入序列，每个序列包含给定句子（句子A）与一个可能后续句（句子B）的拼接。引入的唯一任务特定参数是一个向量，其与[CLS]标记表示C的点积表示每个选项的得分，该得分通过softmax层进行归一化处理。

我们对模型进行了3个周期的微调，学习率为2e-5，批处理大小为16。结果如表4所示。BERT$_{\text{LARGE}}$的表现优于作者基线ESIM+ELMo系统+27.1%，并超过OpenAI GPT 8.3%。

5 消融研究

在本节中，我们对BERT的多个方面进行了消融实验，以更好地理解它们的相对重要性。此外

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
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 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 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, BERT$_{\text{BASE}}$ contains 110M parameters and 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

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：使用BERT$_{BASE}$架构对预训练任务进行消融实验。“No NSP”表示训练时不包含下一句预测任务。“LTR & No NSP”表示像OpenAI GPT一样，作为从左到右的语言模型进行训练，且不包含下一句预测。“+ BiLSTM”表示在微调阶段，在“LTR + No NSP”模型顶部添加一个随机初始化的BiLSTM。

消融研究可在附录C中找到。

5.1 预训练任务的影响

我们通过使用与BERT$_{BASE}$完全相同的预训练数据、微调方案和超参数来评估两个预训练目标，以此证明BERT深度双向性的重要性：

无NSP：一种双向模型，采用“掩码语言模型”（MLM）进行训练，但不包含“下一句预测”（NSP）任务。

LTR & 无NSP：一种仅使用标准从左到右（LTR）语言模型进行训练的纯左上下文模型，而非掩码语言模型。在微调阶段同样应用了仅左向约束，因为取消该约束会导致预训练与微调不匹配，从而降低下游任务性能。此外，该模型在预训练阶段未使用NSP任务。该模型可直接与OpenAI GPT进行对比，但采用了我们更大的训练数据集、输入表示方法及微调方案。

我们首先考察了NSP任务带来的影响。在表5中，我们发现移除NSP会显著降低QNLI、MNLI和SQuAD 1.1的性能。接着，我们通过比较“无NSP”与“仅左向语境 & 无NSP”来评估训练双向表示的影响。左向语境模型在所有任务上的表现均差于MLM模型，其中在MRPC和SQuAD任务上性能下降尤为明显。

对于SQuAD数据集，直观上可以明显看出，LTR模型在词元预测方面表现会很差，因为词元级别的隐藏状态缺乏右侧上下文。为了切实尝试增强LTR系统，我们在其顶部添加了一个随机初始化的BiLSTM。这确实显著提升了SQuAD上的结果，但结果仍然远不如预训练的双向模型。BiLSTM在GLUE任务上的表现反而有所下降。

我们认识到，也可以像ELMo那样训练独立的LTR和RTL模型，并将每个标记表示为两个模型的拼接。然而：(a) 这比单一的双向模型成本高一倍；(b) 对于像QA这样的任务来说，这并不直观，因为RTL模型无法根据问题来调整答案；(c) 这严格来说不如深度双向模型强大，因为后者能在每一层同时利用左右上下文。

5.2 模型规模的影响

在本节中，我们探讨模型规模对微调任务准确性的影响。我们训练了多个具有不同层数、隐藏单元和注意力头数量的BERT模型，同时保持其他超参数和训练流程与先前描述的一致。

在选定的GLUE任务上的结果如表6所示。在此表中，我们报告了5次随机重启微调的平均开发集准确率。可以看出，更大的模型在所有四个数据集上都带来了严格的准确率提升，即使是对于仅有3600个标注训练样本、且与预训练任务存在显著差异的MRPC数据集也是如此。另一个可能令人惊讶的发现是，我们能够在已经相对现有文献相当庞大的模型基础上取得如此显著的改进。例如，Vaswani等人（2017）研究中探索的最大Transformer结构为（L=6, H=1024, A=16），编码器包含1亿参数；而我们在文献中找到的最大Transformer结构为（L=64, H=512, A=2），包含2.35亿参数（Al-Rfou等人，2018）。相比之下，BERT$_{BASE}$包含1.1亿参数，BERT$_{LARGE}$则包含3.4亿参数。

长期以来，人们已经认识到增加模型规模会持续提升机器翻译和语言建模等大规模任务的性能，这一点通过表6所示的训练数据LM困惑度得以验证。然而，我们相信这是首个令人信服地证明：只要模型经过充分预训练，即使扩展到极大规模，也能在极小规模任务上带来显著提升的研究。Peters等人（2018b）的研究表明

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.

在增加预训练双向语言模型规模从两层到四层对下游任务的影响方面，结果不一；而Melamud等人（2016）曾简要提及，将隐藏层维度从200增至600有所助益，但继续提升至1000并未带来进一步改善。这两项先前研究均采用基于特征的方法——我们推测，当模型直接在下游任务上进行微调，且仅使用极少量随机初始化的附加参数时，即使下游任务数据量非常有限，任务专用模型仍能受益于更庞大、表达能力更强的预训练表征。

5.3 基于BERT的特征方法

迄今为止展示的所有BERT结果都采用了微调方法，即在预训练模型基础上添加简单的分类层，并针对下游任务对所有参数进行联合微调。然而，基于特征的方法——从预训练模型中提取固定特征——具有特定优势。首先，并非所有任务都能轻易用Transformer编码器架构表示，因此需要添加任务特定的模型架构。其次，预计算训练数据的高成本表征具有显著计算优势：只需计算一次昂贵的数据表征，即可基于该表征使用更轻量的模型进行大量实验。

在本节中，我们通过将BERT应用于CoNLL-2003命名实体识别（NER）任务（Tjong Kim Sang和De Meulder，2003）来比较这两种方法。在BERT的输入中，我们使用保留大小写的WordPiece模型，并包含数据提供的最大文档上下文。遵循标准做法，我们将其表述为标注任务，但不使用条件随机场（CRF）。

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模型规模的消融实验。#L = 层数；#H = 隐藏层大小；#A = 注意力头数量。“LM (ppl)”指掩码语言模型在预留训练数据上的困惑度。

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		
BERT$_{LARGE}$	96.6	92.8
BERT$_{BASE}$	96.4	92.4
Feature-based approach (BERT$_{BASE}$)		
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命名实体识别结果。超参数通过开发集选定。所报告的开发集和测试集分数是使用这些超参数进行5次随机重启后的平均值。

在输出层中，我们使用第一个子标记的表示作为输入，通过NER标签集上的标记级分类器进行处理。

为了消融微调方法，我们采用基于特征的方法，通过从一个或多个层提取激活值*without*，而不微调BERT的任何参数。这些上下文嵌入被用作随机初始化的两层768维BiLSTM的输入，然后接入分类层。

结果如表7所示。BERT$_{LARGE}$与最先进方法相比表现出竞争力。最佳性能的方法将预训练Transformer顶部四个隐藏层的标记表示进行拼接，其结果仅比微调整个模型低0.3个F1值。这表明BERT对于微调和基于特征的方法都同样有效。

6 结论

最近，基于语言模型的迁移学习带来的实证改进表明，丰富的无监督预训练是许多语言理解系统不可或缺的一部分。特别是，这些成果使得即使是低资源任务也能从深度单向架构中受益。我们的主要贡献在于进一步将这些发现推广到深度*bidirectional*架构，使得同一个预训练模型能够成功应对广泛的自然语言处理任务。

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.

参考文献

Alan Akbik、Duncan Blythe 和 Roland Vollgraf。2018。用于序列标注的上下文字符串嵌入。收录于 *Proceedings of the 27th International Conference on Computational Linguistics*，第 1638–1649 页。

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo 和 Llion Jones。2018。基于更深层次自注意力的字符级语言建模。*arXiv preprint arXiv:1808.04444*。

安藤理惠和张彤。2005年。从多任务和无标签数据中学习预测结构的框架。*Journal of Machine Learning Research*，6(11月):1817–1853。

Luisa Bentivogli、Bernardo Magnini、Ido Dagan、Hoa Trang Dang和Danilo Giampiccolo。2009年。第五届PASCAL文本蕴含识别挑战赛。收录于*TAC*。NIST。

John Blitzer, Ryan McDonald和Fernando Pereira。2006年。基于结构对应学习的领域自适应。载于 *Proceedings of the 2006 conference on empirical methods in natural language processing*，第120–128页。计算语言学协会。

Samuel R. Bowman、Gabor Angeli、Christopher Potts 和 Christopher D. Manning。2015。一个用于学习自然语言推理的大规模标注语料库。载于 *EMNLP*。计算语言学协会。

彼得·F·布朗、彼得·V·德索萨、罗伯特·L·默瑟、文森特·J·德拉彼得拉与詹妮弗·C·赖。1992年。基于类的自然语言n元模型。*Computational linguistics*，18(4)：467–479。

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 任务 1：语义文本相似性多语言与跨语言重点评估。载于 *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*，第 1–14 页，加拿大温哥华。计算语言学协会。

Ciprian Chelba、Tomas Mikolov、Mike Schuster、齐格、Thorsten Brants、Phillipp Koehn与Tony Robinson。2013年。用于衡量统计语言建模进展的十亿词基准测试。*arXiv preprint arXiv:1312.3005*。

Z. Chen, H. Zhang, X. Zhang, and L. Zhao. 2018. Quora问题对。

克里斯托弗·克拉克和马特·加德纳。2018年。简单有效的多段落阅读理解。于*ACL*。

凯文·克拉克、Minh-Thang Luong、克里斯托弗·D·曼宁与郭去疾。2018。基于交叉视角训练的半监督序列建模。发表于*Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*，第1914–1925页。

Ronan Collobert 和 Jason Weston。2008。自然语言处理的统一架构：采用多任务学习的深度神经网络。于 *Proceedings of the 25th international conference on Machine learning*，第160–167页。ACM。

Alexis Conneau、Douwe Kiela、Holger Schwenk、Lo¨c Barrault与Antoine Bordes。2017。基于自然语言推理数据的通用句子表示监督学习。收录于 *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*，第670–680页，丹麦哥本哈根。计算语言学协会。

Andrew M Dai 和 Quoc V Le。2015。半监督序列学习。于 *Advances in neural information processing systems*，第3079–3087页。

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet：一个大规模分层图像数据库。于 *CVPR09*。

William B Dolan 与 Chris Brockett。2005。自动构建句子复述语料库。发表于 *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*。

William Fedus、Ian Goodfellow 和 Andrew M Dai。2018。Maskgan：通过填充 {v*} 实现更好的文本生成。_ *arXiv preprint arXiv:1801.07736*。

Dan Hendrycks 与 Kevin Gimpel。2016。通过高斯误差线性单元连接非线性与随机正则化器。*CoRR*，abs/1606.08415。

Felix Hill、Kyunghyun Cho 和 Anna Korhonen。2016。从未标注数据中学习句子的分布式表示。于 *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*。计算语言学协会。

Jeremy Howard与Sebastian Ruder。2018。面向文本分类的通用语言模型微调。发表于*ACL*。计算语言学协会。

胡明浩、彭宇星、黄震、邱锡鹏、韦福如、周明。2018。用于机器阅读理解任务的强化记忆网络。发表于 *IJCAI*。

Yacine Jernite、Samuel R. Bowman 与 David Sontag。2017。基于语篇的目标用于快速无监督句子表示学习。*CoRR*，abs/1705.00557。

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

Mandar Joshi、Eunsol Choi、Daniel S Weld和Luke Zettlemoyer。2017。Triviaqa：一个用于阅读理解的大规模远程监督挑战数据集。于*ACL*。Ryan Kiros、Yukun Zhu、Ruslan R Salakhutdinov、Richard Zemel、Raquel Urtasun、Antonio Torralba和Sanja Fidler。2015。Skip-thought向量。于 *Advances in neural information processing systems*，第3294–3302页。Quoc Le和Tomas Mikolov。2014。句子和文档的分布式表示。于*International Conference on Machine Learning*，第1188–1196页。Hector J Levesque、Ernest Davis和Leora Morgenstern。2011。Winograd模式挑战。于 *Aaai spring symposium: Logical formalizations of commonsense reasoning*，第46卷，第47页。Lajanugen Logeswaran和Honglak Lee。2018。一种学习句子表示的高效框架。于 *International Conference on Learning Representations*。Bryan McCann、James Bradbury、Caiming Xiong和Richard Socher。2017。在翻译中学习：上下文化的词向量。于*NIPS*。Oren Melamud、Jacob Goldberger和Ido Dagan。2016。context2vec：使用双向LSTM学习通用上下文嵌入。于 *CoNLL*。Tomas Mikolov、Ilya Sutskever、Kai Chen、Greg S Corrado和Jeff Dean。2013。词和短语的分布式表示及其组合性。于 *Advances in Neural Information Processing Systems 26*，第3111–3119页。Curran Associates, Inc.Andriy Mnih和Geoffrey E Hinton。2009。一种可扩展的层次化分布式语言模型。于D. Koller、D. Schuurmans、Y. Bengio和L. Bottou编辑，*Advances in Neural Information Processing Systems 21*，第1081–1088页。Curran Associates, Inc.Ankur P Parikh、Oscar Täckström、Dipanjan Das和Jakob Uszkoreit。2016。一种用于自然语言推理的可分解注意力模型。于*EMNLP*。Jeffrey Pennington、Richard Socher和Christopher D. Manning。2014。Glove：用于词表示的全局向量。于 *Empirical Methods in Natural Language Processing (EMNLP)*，第1532–1543页。Matthew Peters、Waleed Ammar、Chandra Bhagavatula和Russell Power。2017。使用双向语言模型的半监督序列标注。于*ACL*。Matthew Peters、Mark Neumann、Mohit Iyyer、Matt Gardner、Christopher Clark、Kenton Lee和Luke Zettlemoyer。2018a。深度上下文化的词表示。于*NAACL*。

Matthew Peters、Mark Neumann、Luke Zettlemoyer 和 Wen-tau Yih。2018b。剖析上下文词嵌入：架构与表示。于 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*，第1499–1510页。 Alec Radford、Karthik Narasimhan、Tim Salimans 和 Ilya Sutskever。2018。通过无监督学习提升语言理解能力。技术报告，OpenAI。 Pranav Rajpurkar、Jian Zhang、Konstantin Lopyrev 和 Percy Liang。2016。SQuAD：面向文本机器理解的10万+个问题。于 *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*，第2383–2392页。 Minjoon Seo、Aniruddha Kembhavi、Ali Farhadi 和 Hannaneh Hajishirzi。2017。用于机器理解的双向注意力流。于 *ICLR*。 Richard Socher、Alex Perelygin、Jean Wu、Jason Chuang、Christopher D Manning、Andrew Ng 和 Christopher Potts。2013。基于情感树库语义组合性的递归深度模型。于 *Proceedings of the 2013 conference on empirical methods in natural language processing*，第1631–1642页。 Fu Sun、Linyang Li、Xipeng Qiu 和 Yang Liu。2018。U-Net：针对无法回答问题的机器阅读理解。*arXiv preprint arXiv:1810.06638*。 Wilson L Taylor。1953。完形填空程序：一种衡量可读性的新工具。*Journalism Bulletin*，30(4):415–433。 Erik F Tjong Kim Sang 和 Fien De Meulder。2003。CONLL-2003共享任务介绍：语言无关的命名实体识别。于 *CoNLL*。 Joseph Turian、Lev Ratinov 和 Yoshua Bengio。2010。词表示：一种简单通用的半监督学习方法。于 *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*，ACL '10，第384–394页。 Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Lukasz Kaiser 和 Illia Polosukhin。2017。注意力机制就是全部所需。于 *Advances in Neural Information Processing Systems*，第6000–6010页。 Pascal Vincent、Hugo Larochelle、Yoshua Bengio 和 Pierre-Antoine Manzagol。2008。通过去噪自编码器提取与组合鲁棒特征。于 *Proceedings of the 25th international conference on Machine learning*，第1096–1103页。ACM。 Alex Wang、Amanpreet Singh、Julian Michael、Felix Hill、Omer Levy 和 Samuel Bowman。2018a。GLUE：一个多任务基准测试与分析平台

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

用于自然语言理解。在*Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*中，第353至355页。

王伟、严明和吴晨。2018b。用于阅读理解和问答的多粒度分层注意力融合网络。于*Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*。计算语言学协会。

亚历克斯·沃斯塔特、阿曼普里特·辛格和塞缪尔·R·鲍曼。2018年。神经网络可接受性判断。*arXiv preprint arXiv:1805.12471*。

Adina Williams、Nikita Nangia与Samuel R Bowman。2018。一个用于通过推理理解句子的广覆盖挑战语料库。于*NAACL*。

吴永辉、Mike Schuster、陈智峰、Quoc V Le、Mohammad Norouzi、Wolfgang Macherey、Maxim Krikun、曹原、高勤、Klaus Macherey 等。2016。谷歌的神经机器翻译系统：弥合人类与机器翻译之间的差距。*arXiv preprint arXiv:1609.08144*。

Jason Yosinski、Jeff Clune、Yoshua Bengio 与 Hod Lipson。2014。深度神经网络中的特征可迁移性如何？载于 *Advances in neural information processing systems*，第 3320–3328 页。

Adams Wei Yu、David Dohan、Minh-Thang Luong、Rui Zhao、Kai Chen、Mohammad Norouzi 和 Quoc V Le。2018。QANet：结合局部卷积与全局自注意力机制用于阅读理解。于 *ICLR*。

Rowan Zellers, Yonatan Bisk, Roy Schwartz 和 Yejin Choi。2018。Swag：一个用于基础常识推理的大规模对抗性数据集。发表于 *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 和 Sanja Fidler。2015。对齐书籍与电影：通过观看电影与阅读书籍实现故事化视觉解释。发表于 *Proceedings of the IEEE international conference on computer vision*，第 19–27 页。

《BERT：用于语言理解的深度双向Transformer预训练》附录

我们将附录组织为三个部分：

- BERT的额外实现细节在附录A中给出；
- 我们实验的更多细节在附录B中给出；并且
- 额外的消融研究见附录C。

我们针对BERT进行了额外的消融研究，包括：– 训练步数的影响；以及 – 不同掩码处理方法的消融分析。

BERT 的额外细节

A.1 预训练任务示意图

我们在下面提供预训练任务的示例。

掩码语言模型与掩码过程假设未标记的句子是“my dog is hairy”，在随机掩码过程中我们选择了第4个标记（对应“hairy”），我们的掩码过程可以进一步通过以下方式说明:

- 80% 的情况下：将单词替换为 [MASK] 标记，例如，my dog is hairy → my dog is [MASK]
- 10% 的情况下：用随机词替换该词，例如，我的狗毛茸茸的 → 我的狗是苹果
- 10% 的情况下：保持单词不变，例如，my dog is hairy → my dog is hairy。这样做的目的是使表征偏向实际观察到的单词。

该方法的优势在于，Transformer编码器不知道哪些词将被要求预测，或者哪些词已被随机词替换，因此它必须为每个输入标记保留一个分布式的上下文表示。此外，由于随机替换仅发生在所有标记的1.5%（即15%的10%），这似乎不会损害模型的语言理解能力。在C.2节中，我们评估了该程序的影响。

与标准语言模型训练相比，掩码语言模型仅对每个批次中15%的标记进行预测，这表明模型可能需要更多的预训练步骤。

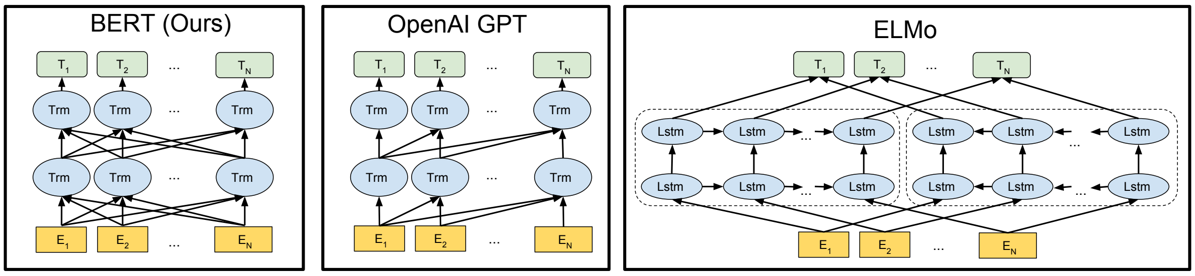


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT 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 BERT$_{\text{BASE}}$ was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total).[13] Training of 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

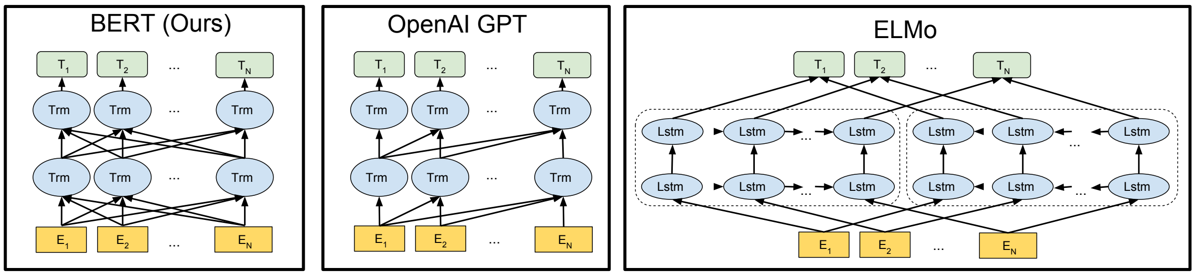


图3：预训练模型架构的差异。BERT使用双向Transformer。OpenAI GPT使用从左到右的Transformer。ELMo使用独立训练的从左到右和从右到左LSTM的拼接来为下游任务生成特征。在这三者中，只有BERT表示在所有层中同时以左右上下文为条件。除了架构差异外，BERT和OpenAI GPT是微调方法，而ELMo是基于特征的方法。

收敛。在C.1节中，我们证明MLM的收敛速度确实略慢于从左到右的模型（后者预测每个词元），但MLM模型带来的实证改进远远超过了训练成本的增加。

下一句预测　下一句预测任务可以通过以下示例进行说明。

输入 = `[CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]` 标签 = `IsNext`
输入 = `[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight ##less birds [SEP]` 标签 = `NotNext`

A.2 预训练流程

为了生成每个训练输入序列，我们从语料库中采样两段文本（尽管它们通常比单个句子长得多，但也可以更短，我们仍将其称为“句子”）。第一句接收A嵌入，第二句接收B嵌入。在50%的情况下，B是实际接在A之后的下一句，另外50%的情况下它是一个随机句子，这是为“下一句预测”任务而设计的。采样时确保组合长度不超过≤ 512个标记。在应用WordPiece分词后，以15%的统一掩码率进行语言模型掩码处理，且不对部分词片段作特殊考虑。

我们以256个序列的批次大小（256个序列 * 512个标记 = 128,000个标记/批次）训练了1,000,000步，这大约相当于40在33亿单词的语料库上训练多个周期。我们采用Adam优化器，学习率为1e-4，β_1 =为0.9，β_2 =为0.999，L2权重衰减为0.01，学习率在前10,000步进行预热，之后线性衰减。所有层使用0.1的dropout概率。遵循OpenAI GPT的方法，我们使用gelu激活函数（Hendrycks和Gimpel，2016）而非标准的relu。训练损失由掩码语言模型似然的均值与下一句预测似然的均值相加构成。

$\text{BERT}_{\text{BASE}}$的训练在4个Cloud TPU上以Pod配置（共16个TPU芯片）进行。[13] $\text{BERT}_{\text{LARGE}}$的训练在16个Cloud TPU（共64个TPU芯片）上完成。每次预训练需要4天时间。

较长的序列成本不成比例地高昂，因为注意力机制的计算复杂度与序列长度呈平方关系。为了在我们的实验中加速预训练，我们先用128的序列长度预训练模型90%的步骤，然后用512的序列长度训练剩余10%的步骤，以学习位置嵌入。

A.3 微调程序

对于微调，大多数模型超参数与预训练时保持一致，除了批大小、学习率和训练周期数。dropout概率始终保持在0.1。最佳超参数值因任务而异，但我们发现以下取值范围在所有任务中都能取得良好效果：

- 批次大小：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

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.

- 学习率（Adam）：5e-5、3e-5、2e-5
- 训练轮数：2、3、4

我们还观察到，大型数据集（例如10万+个带标签的训练样本）对超参数选择的敏感度远低于小型数据集。微调通常非常快，因此合理做法是直接对上述参数进行穷举搜索，并选择在开发集上表现最佳的模型。

A.4 BERT、ELMo 与 OpenAI GPT 的对比

这里我们研究了近期流行的表示学习模型之间的差异，包括ELMo、OpenAI GPT和BERT。模型架构之间的对比在图3中直观展示。需要注意的是，除了架构差异外，BERT和OpenAI GPT采用微调方法，而ELMo则是基于特征的方法。

与BERT最为接近的现有预训练方法是OpenAI GPT，后者在大规模文本语料上训练了一个从左到右的Transformer语言模型。实际上，BERT的许多设计决策都刻意保持与GPT高度一致，以便两种方法能在最小变量下进行对比。本文的核心论点是：双向性特征及第3.1节提出的两项预训练任务构成了实证改进的主要来源，但我们也注意到BERT与GPT在训练方式上还存在若干其他差异：

- GPT在BooksCorpus（8亿单词）上训练；BERT在BooksCorpus（8亿单词）和维基百科（25亿单词）上训练。

- GPT在微调时引入了句子分隔符（[SEP]）和分类器标记（[CLS]）；而BERT在预训练期间就学习了[SEP]、[CLS]以及句子A/B嵌入。

- GPT以32,000词的批量大小训练了100万步；BERT以128,000词的批量大小训练了100万步。

- GPT在所有微调实验中均采用5e-5的相同学习率；BERT则选择在开发集上表现最佳的任务特定微调学习率。

为了分离这些差异的影响，我们在5.1节进行了消融实验，结果表明大部分改进实际上源于两个预训练任务及其所实现的双向性。

A.5 不同任务上的微调示例

图4展示了在不同任务上对BERT进行微调的示意图。我们的任务特定模型通过将BERT与一个额外的输出层结合而形成，因此只需从头学习最少数量的参数。在这些任务中，(a)和(b)是序列级任务，而(c)和(d)是词元级任务。图中，E代表输入嵌入，T_i代表词元i的上下文表示，[CLS]是分类输出的特殊符号，[SEP]则是用于分隔非连续词元序列的特殊符号。

B 详细实验设置

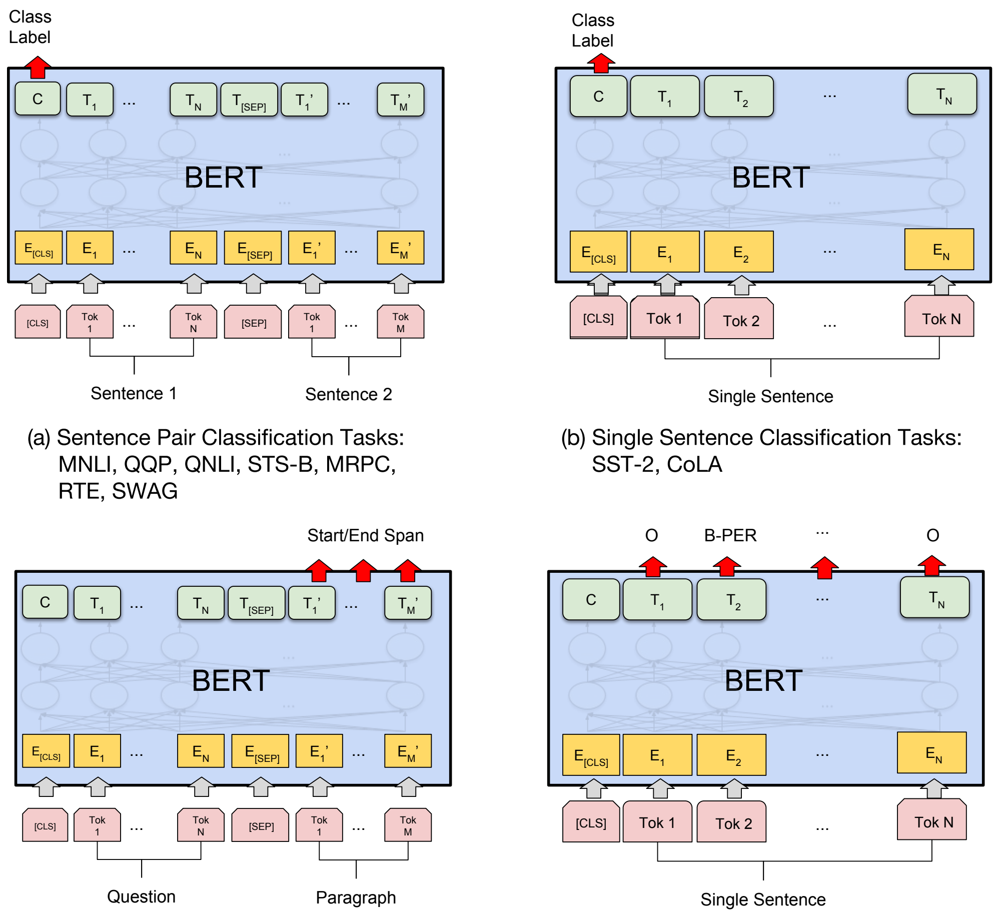
B.1 GLUE基准实验的详细描述。

我们在表1中的GLUE结果来自https://gluebenchmark.com/排行榜和https://blog.openai.com/language-unsupervised。GLUE基准包含以下数据集，其描述最初由Wang等人（2018a）总结：

MNLI多类型自然语言推理是一项大规模众包蕴含关系分类任务（Williams等人，2018）。给定一对句子，其目标是预测第二句话相对于第一句话是*entailment*、*contradiction*还是*neutral*。

QQP Quora问题对是一项二元分类任务，其目标是判断Quora上提出的两个问题在语义上是否等价（Chen等人，2018年）。

QNLI（问题自然语言推理）是斯坦福问答数据集（Rajpurkar等人，2016）的一个版本，该数据集已被转化为二元分类任务（Wang等人，2018a）。正例是包含正确答案的（问题，句子）对，负例则来自同一段落中不包含答案的（问题，句子）对。



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

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.

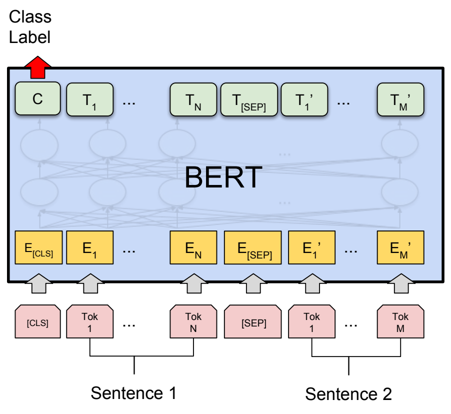
MRPC Microsoft Research Paraphrase Corpus consists of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent (Dolan and Brockett, 2005).

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

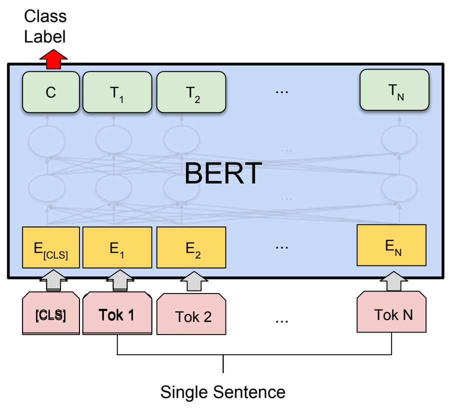
WNLI Winograd NLI is a small natural language inference dataset (Levesque et al., 2011). The GLUE webpage notes that there are issues with the construction of this dataset, [15] and every trained system that's been submitted to GLUE has performed worse than the 65.1 baseline accuracy of predicting the 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 multitask training with MNLI.

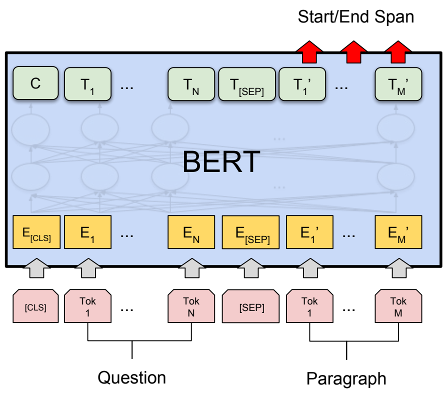
[15]https://gluebenchmark.com/faq



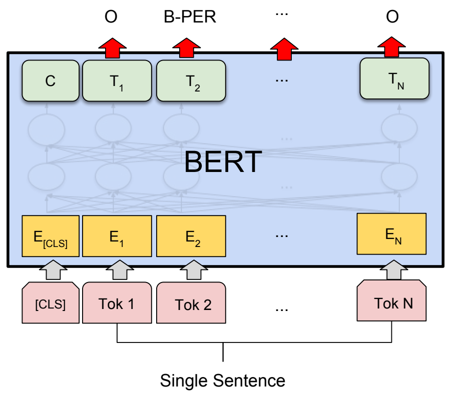
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1



(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

图4：在不同任务上微调BERT的示意图。

SST-2 斯坦福情感树库是一个二元单句分类任务，由从电影评论中提取的句子组成，并包含人工标注的情感标签（Socher et al., 2013）。

CoLA（语言可接受性语料库）是一个二元的单句分类任务，其目标是预测一个英语句子在语言学上是否“可接受”（Warstadt等人，2018年）。

STS-B 语义文本相似性基准测试集是一个从新闻标题和其他来源收集的句子对集合（Cer等人，2017年）。这些句子对通过1到5的评分进行标注，以表示两个句子在语义上的相似程度。

MRPC 微软研究释义语料库由从在线新闻源自动提取的句子对组成，并经过人工标注。对于句子对中的句子是否语义等价（Dolan和Brockett，2005年）。

RTE（Recognizing Textual Entailment，文本蕴含识别）是一项与MNLI类似的二元蕴含任务，但其训练数据量要少得多（Bentivogli等人，2009年）。[14]

WNLI Winograd NLI 是一个小型自然语言推理数据集（Levesque 等人，2011）。GLUE 网页指出该数据集的构建存在问题，{v*} 并且所有提交至 GLUE 的已训练系统的表现都低于预测多数类别的 65.1% 基线准确率。因此，为公平对待 OpenAI GPT，我们排除了该数据集。在我们的 GLUE 提交结果中，始终预测多

[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 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, 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.

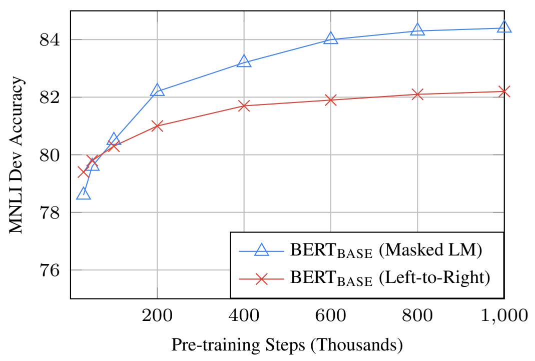


Figure 5: Ablation over number of training steps. This shows the MNLI accuracy after fine-tuning, starting from model parameters that have been pre-trained for k steps. The x-axis is the value of k.

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.

jority class.

C 额外的消融研究

C.1 训练步数的影响

图5展示了从经过k步预训练的检查点进行微调后的MNLI开发集准确率。这使我们能够回答以下问题：

1. 问题：BERT是否真的需要如此大量的预训练（128,000词/批次 * 1,000,000步）才能达到较高的微调精度？

   答案：是的，与训练50万步相比，BERT $_{\text{BASE}}$在训练100万步时，在MNLI上获得了近1.0%的额外准确率。

2. 问题：由于每个批次中只有15%的单词被预测，而不是每个单词都被预测，MLM预训练是否比LTR预训练收敛得更慢？

   答案：MLM模型的收敛速度确实略慢于LTR模型。然而，在绝对准确度方面，MLM模型几乎立即开始超越LTR模型。

C.2 不同掩码程序的消融研究

在3.1节中我们提到，BERT在使用掩码语言模型（MLM）目标进行预训练时，对目标标记采用了混合掩码策略。以下是一项消融研究，旨在评估不同掩码策略的效果。

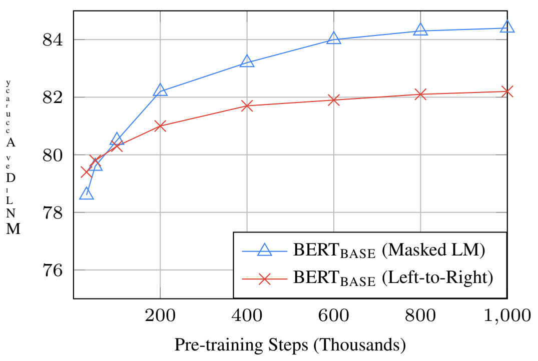


图5：训练步数的消融实验。这展示了从预训练了k步的模型参数开始微调后，MNLI的准确率。x轴表示k的值。

需要注意的是，掩码策略的目的是减少预训练与微调之间的不匹配，因为[MASK]符号在微调阶段从未出现。我们报告了MNLI和NER的Dev结果。对于NER，我们同时报告了微调和基于特征的方法，因为我们预计这种不匹配在基于特征的方法中会被放大，因为模型将没有机会调整表示。

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：不同掩码策略的消融实验。

结果如表8所示。表中，MASK表示我们将目标标记替换为[MASK]符号以进行MLM；SAME表示我们保持目标标记不变；RND表示我们将目标标记替换为另一个随机标记。

表格左侧的数字代表了在MLM预训练期间使用的特定策略的概率（BERT使用80%、10%、10%）。论文的右侧部分代表了开发集的结果。对于基于特征的方法，我们将BERT的最后4层连接起来作为特征，这在第5.3节中被证明是最佳方法。

从表中可以看出，微调对不同掩码策略表现出惊人的鲁棒性。然而正如预期，在将基于特征的方法应用于NER时，仅使用MASK策略会产生问题。有趣的是，仅使用RND策略的表现也远逊于我们的策略。