

# 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 **Bidirectional Encoder Representations from Transformers**. 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: **Bidirectional Encoder Representations from Transformers**. BERT alleviates the previously mentioned unidirectionality constraint by using a “masked language model” (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked

# BERT：用于语言理解的深度双向Transformer预训练

Jacob Devlin 明伟张 Kenton Lee Kristina Toutanova 谷歌 AI 语言

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

## 摘要

我们介绍了一种新的语言表示模型，称为 BERT，它代表来自 Transformer 的双向编码器表示。与最近的语言表示模型（Peters 等人，2018a；Radford 等人，2018）不同，BERT 被设计为通过在所有层中共同调节左右上下文来从未标记文本中预训练深度双向表示。因此，预训练的 BERT 模型只需一个额外的输出层即可微调，以创建适用于各种任务的最先进模型，例如问答和语言推理，而无需进行大量的特定于任务的架构修改。

BERT 的概念很简单，但效果却很强大。它在 11 项自然语言处理任务中取得了新的最先进的结果，包括将 GLUE 分数提升至 80.5%（绝对提升 7.7%），MultiNLI 准确率提升至 86.7%（绝对提升 4.6%），SQuAD v1.1 问答测试 F1 提升至 93.2（绝对提升 1.5%）以及 SQuAD v2.0 测试 F1 提升至 83.1（绝对提升 5.1%）。

## 1 引言

语言模型预训练已被证明可以有效地提高许多自然语言处理任务（Dai 和 Le，2015；Peters 等人，2018a；Radford 等人，2018；Howard 和 Ruder，2018）。这些任务包括句子级任务，例如自然语言推理（Bowman 等人，2015；Williams 等人，2018）和释义（Dolan 和 Brockett，2005），这些任务旨在通过整体分析句子来预测句子之间的关系，以及词语级任务，例如命名实体识别和问答，这些任务要求模型在词语级别生成细粒度的输出（Tjong Kim Sang 和 De Meulder，2003；Rajpurkar 等人，2016）。

目前有两种将预训练语言表示应用于下游任务的策略：*feature-based* 和 *fine-tuning*。基于特征的方法，例如 ELMo（Peters 等人，2018a），使用包含预训练表示作为附加特征的任务特定架构。微调方法，例如生成式预训练 Transformer（OpenAI GPT）（Radford 等人，2018），引入了最少的任务特定参数，并通过简单地微调 *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期间共享相同的目标函数，其中它们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示的能力，特别是对于微调方法。主要限制是标准语言模型是单向的，这限制了预训练期间可使用的架构选择。例如，在 OpenAI GPT 中，作者使用从左到右的架构，其中每个标记只能在 Transformer（Vaswani 等人，2017）的自注意力层中关注之前的标记。这种限制对于句子级任务来说是次优的，并且在将微调方法应用于标记级任务（如问答）时可能非常有害，因为在这些任务中，从两个方向整合上下文至关重要。

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 通过使用受 Cloze 任务（Taylor，1953）启发的“掩码语言模型”（MLM）预训练目标，缓解了之前提到的单向性约束。掩码语言模型随机掩盖输入中的一些标记，目标是预测掩盖的标记的原始词汇 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-

仅根据其上下文来预测一个词。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，我们还使用“下一个句子预测”任务，该任务联合预训练文本对表示。我们论文的贡献如下：

- 我们证明了双向预训练对于语言表示的重要性。与 Radford 等人 (2018) 使用单向语言模型进行预训练不同，BERT 使用掩码语言模型来实现预训练的深度双向表示。这也与 Peters 等人 (2018a) 形成对比，后者使用独立训练的从左到右和从右到左 LM 的浅层串联。

- 我们表明，预训练的表示减少了对许多高度工程化的特定任务架构的需求。BERT 是第一个基于微调的表示模型，它在大量句子级 *and* 词汇级任务上取得了最先进的性能，超越了许多特定任务的架构。

- BERT 在 11 个 NLP 任务上取得了最先进的结果。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，本节将简要回顾最常用的方法。

### ## 2.1 无监督特征方法

学习广泛适用的词表示方法是几十年来的一个活跃研究领域，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)。预训练词嵌入是现代 NLP 系统不可或缺的一部分，与从头开始学习的嵌入相比，它提供了显著的改进 (Turian 等人, 2010)。为了预训练词嵌入向量，人们使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和错误词的目标 (Mikolov 等人, 2013)。

这些方法已被推广到更粗粒度的粒度，例如句子嵌入 (Kiros 等人, 2015; Logeswaran 和 Lee, 2018) 或段落嵌入 (Le 和 Mikolov, 2014)。为了训练句子表示，先前的工作使用了目标来对候选下一个句子进行排序 (Jernite 等人, 2017; Logeswaran 和 Lee, 2018)，给定前一个句子的表示，从左到右生成下一个句子的词 (Kiros 等人, 2015)，或去噪自动编码器衍生的目标 (Hill 等人, 2016)。

ELMo 及其前身 (Peters 等人, 2017, 2018a) 沿不同的维度推广了传统的词嵌入研究。它们从一个从左到右和一个从右到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表示的串联。在将上下文词嵌入与现有的特定任务架构集成时，ELMo 推动了几个主要 NLP 基准 (Peters 等人, 2018a) 的最新技术，包括问答 (Rajpurkar 等人, 2016)、情感分析 (Socher 等人, 2013) 和命名实体识别 (Tjong Kim Sang 和 De Meulder, 2003)。Melamud 等人 (2016) 提出通过一项任务来学习上下文表示，该任务使用 LSTM 从左右上下文预测单个词。与 ELMo 类似，他们的模型是基于特征的，而不是深度双向的。Fedus 等人 (2018) 表明，完形填空任务可用于提高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，该方向上的首批工作仅从未标记文本中预训练词嵌入参数 (Collobert 和 Weston, 2008)。

最近，从无标签文本中预训练并针对监督下游任务进行微调的句子或文档编码器，可以生成上下文词语表示 (Dai 和 Le, 2015; Howard 和 Ruder, 2018; Radford 等人, 2018)。这些方法的优势在于，很少需要从头开始学习参数。至少部分由于这种优势，OpenAI GPT (Radford 等人, 2018) 在 GLUE 基准测试 (Wang 等人, 2018a) 的许多句子级任务上取得了先前最先进的结果。从左到右的语言模型-

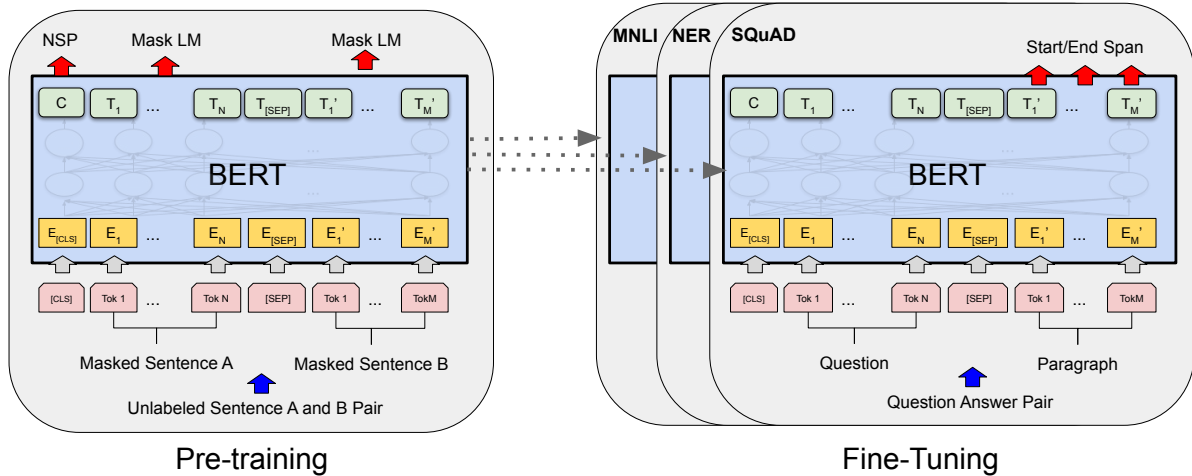


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are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

mal 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 `tensorflow/tensor2tensor` library.<sup>1</sup> 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.”<sup>2</sup>

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  $A$ .<sup>3</sup> We primarily report results on two model sizes: **BERT<sub>BASE</sub>** ( $L=12$ ,  $H=768$ ,  $A=12$ , Total Parameters=110M) and **BERT<sub>LARGE</sub>** ( $L=24$ ,  $H=1024$ ,  $A=16$ , Total Parameters=340M).

**BERT<sub>BASE</sub>** 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.<sup>4</sup>

<sup>1</sup><https://github.com/tensorflow/tensor2tensor>

<sup>2</sup><http://nlp.seas.harvard.edu/2018/04/03/attention.html>

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

<sup>4</sup>We note that in the literature the bidirectional Trans-



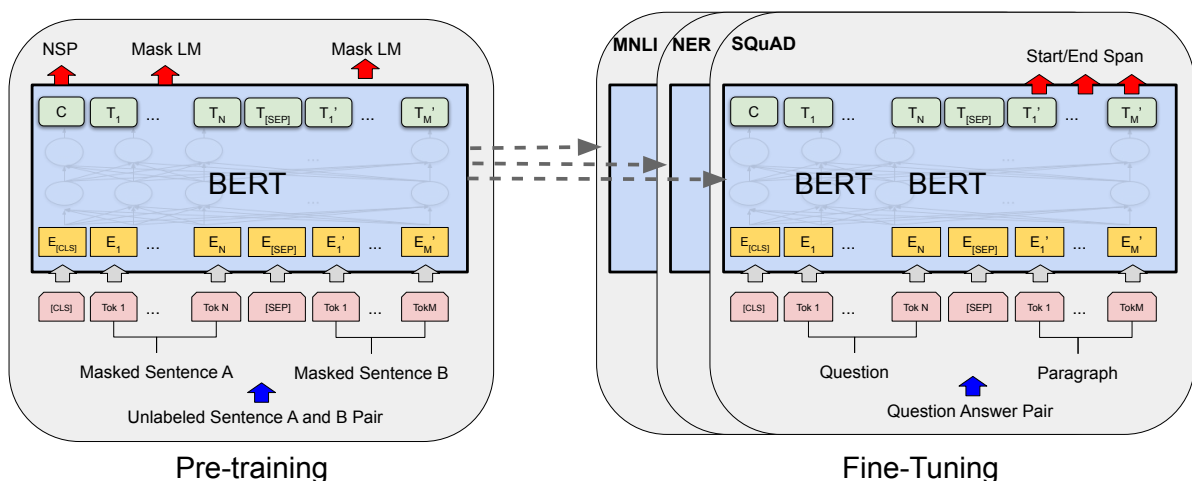


图 1：BERT 的整体预训练和微调流程。除了输出层之外，预训练和微调都使用相同的架构。相同的预训练模型参数用于初始化不同下游任务的模型。在微调期间，所有参数都被微调。[CLS] 是在每个输入示例前面添加的一个特殊符号，[SEP] 是一个特殊的分割标记（例如，分割问题/答案）。

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

### ## 2.3 从监督数据中迁移学习

也有一些工作表明，从具有大型数据集的监督任务中进行有效的迁移，例如自然语言推理（Conneau 等人，2017 年）和机器翻译（McCann 等人，2017 年）。计算机视觉研究也证明了从大型预训练模型中进行迁移学习的重要性，其中一个有效的方案是微调使用 ImageNet 预训练的模型（Deng 等人，2009 年；Yosinski 等人，2014 年）。

### 3 个 BERT

在本节中，我们介绍 BERT 及其详细实现。我们的框架包含两个步骤：*pre-training* 和 *fine-tuning*。在预训练期间，模型在不同预训练任务上使用未标记数据进行训练。对于微调，BERT 模型首先使用预训练参数进行初始化，然后使用下游任务的标记数据对所有参数进行微调。每个下游任务都有单独的微调模型，即使它们使用相同的预训练参数进行初始化。图 1 中的问答示例将作为本节的运行示例。

预训练架构和最终下游架构之间的主要区别。

模型架构 BERT 的模型架构是一个基于 Vaswani 等人 (2017) 中描述的原始实现并在 tensorflow/tensor2tensor 库中发布的多层双向 Transformer 编码器。<sup>1</sup> 由于 Transformer 的使用已变得普遍，并且我们的实现几乎与原始实现相同，因此我们将省略对模型架构的详尽背景描述，并请读者参考 Vaswani 等人 (2017) 以及“带注释的 Transformer”等优秀指南。<sup>2</sup>

在本工作中，我们将层数（即 Transformer 块）记为  $L$ ，隐藏层大小记为  $H$ ，自注意力头数记为  $A$ 。<sup>3</sup> 我们主要报告两种模型尺寸的结果：BERT<sub>BASE</sub> ( $L=12$ ,  $H=768$ ,  $A=12$ , 总参数=110M) 和 BERT<sub>LARGE</sub> ( $L=24$ ,  $H=1024$ ,  $A=16$ , 总参数=340M)。

为了比较目的，选择 BERT<sub>BASE</sub> 与 OpenAI GPT 具有相同的模型大小。然而，至关重要是，BERT Transformer 使用双向自注意力，而 GPT Transformer 使用受限的自注意力，其中每个标记只能关注其左侧的上下文。<sup>4</sup>

BERT 的一个显著特点是它不同任务中统一的架构。

<sup>1</sup><https://github.com/tensorflow/tensor2tensor>

<sup>2</sup><http://nlp.seas.harvard.edu/2018/04/03/attention.html>

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

<sup>4</sup>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.,  $\langle \text{Question, Answer} \rangle$ ) 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 ( $[\text{CLS}]$ ). 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 ( $[\text{SEP}]$ ). 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  $[\text{CLS}]$  token as  $C \in \mathbb{R}^H$ , and the final hidden vector for the  $i^{\text{th}}$  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  $[\text{MASK}]$  token does not appear during fine-tuning. To mitigate this, we do not always replace “masked” words with the actual  $[\text{MASK}]$  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  $[\text{MASK}]$  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  $\text{IsNext}$ ), and 50% of the time it is a random sentence from the corpus (labeled as  $\text{NotNext}$ ). As we show in Figure 1,  $C$  is used for next sentence prediction (NSP).<sup>5</sup> Despite its simplicity, we demonstrate in Section 5.1 that pre-training towards this task is very beneficial to both QA and NLI.<sup>6</sup>

<sup>5</sup>The final model achieves 97%-98% accuracy on NSP.

<sup>6</sup>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 处理各种下游任务，我们的输入表示能够在一个标记序列中明确地表示单个句子和一对句子（例如，〈问题，答案〉）。在本工作中，“句子”可以是任意连续文本跨度，而不是实际的语言句子。“序列”指的是 BERT 的输入标记序列，它可以是单个句子或两个句子打包在一起。

我们使用 WordPiece 嵌入（Wu 等人，2016），词汇量为 30,000 个词。每个序列的第一个词总是特殊的分类词 ([CLS])。与该词对应的最终隐藏状态被用作分类任务的聚合序列表示。句子对被打包成一个单一的序列。我们用两种方式区分句子。首先，我们用一个特殊的词 ([SEP]) 将它们分开。其次，我们为每个词添加一个学习到的嵌入，指示它属于句子 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] 词的最终隐藏向量表示为  $C \in \mathbb{R}^H$ ， $i^{\text{th}}$  输入词的最终隐藏向量表示为  $T_i \in \mathbb{R}^H$ 。

对于给定的词元，其输入表示通过将相应的词元、片段和位置嵌入相加来构建。图 2 展示了这种构建的可视化。

### 3.1 BERT 预训练

与 Peters 等人（2018a）和 Radford 等人（2018）不同，我们没有使用传统的从左到右或从右到左的语言模型来预训练 BERT。相反，我们使用两个无监督任务来预训练 BERT，在本节中描述。此步骤在图 1 的左侧部分中展示。

任务 #1：掩码语言模型 直观地，人们有理由相信，深度双向模型严格来说比左到右模型或左到右模型和右到左模型的浅层连接更强大。不幸的是，标准的条件语言模型只能从左到右或右到左进行训练，因为双向条件允许每个词间接地“看到自己”，模型可以很容易地在多层上下文中预测目标词。

为了训练一个深度双向表示，我们只需随机屏蔽掉一定比例的输入词元，然后预测这些被屏蔽的词元。我们将此过程称为“掩码语言模型”（MLM），尽管在文献中它通常被称为 Cloze 任务（Taylor, 1953）。在这种情况下，对应于掩码词元的最终隐藏向量被馈送到词汇表上的输出 softmax，就像标准语言模型一样。在我们所有的实验中，我们随机地屏蔽了每个序列中 15% 的所有 WordPiece 词元。与降噪自编码器（Vincent 等人，2008）相比，我们只预测被屏蔽的词，而不是重建整个输入。

虽然这使我们能够获得一个双向预训练模型，但缺点是我们正在创建预训练和微调之间的不匹配，因为 [MASK] 标记在微调期间不会出现。为了缓解这种情况，我们并不总是用实际的 [MASK] 标记替换“屏蔽”的词。训练数据生成器随机选择 15% 的标记位置进行预测。如果选择第  $i$  个标记，我们用 (1) [MASK] 标记替换第  $i$  个标记 80% 的时间 (2) 随机标记 10% 的时间 (3) 不变的第  $i$  个标记 10% 的时间。然后， $T_i$  将用于预测原始标记，并使用交叉熵损失。我们在附录 C.2 中比较了此过程的变体。

任务 #2：下一句预测（NSP）许多重要的下游任务，例如问答（QA）和自然语言推理（NLI），都基于理解两个句子之间的关系，而这并非语言模型直接捕获的。为了训练一个能够理解句子关系的模型，我们针对一个可以从任何单语语料库中轻松生成的二元化任务进行预训练。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 是实际跟随 A 的下一句（标记为 IsNext），而 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， $C$  用于下一句预测（NSP）。<sup>5</sup> 尽管它很简单，但在 5.1 节中我们证明了针对此任务进行预训练对 QA 和 NLI 都非常有益。<sup>6</sup>

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.

<sup>5</sup>The final model achieves 97%-98% accuracy on NSP.

<sup>6</sup>The vector  $C$  is not a meaningful sentence representation without fine-tuning, since it was trained with NSP.





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- $\emptyset$  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.<sup>7</sup> We describe the task-specific details in the corresponding subsections of Section 4. More details can be found in Appendix A.5.

## 4 Experiments

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

### 4.1 GLUE

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

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

<sup>7</sup>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%.

<sup>8</sup>See (10) in <https://gluebenchmark.com/faq>.

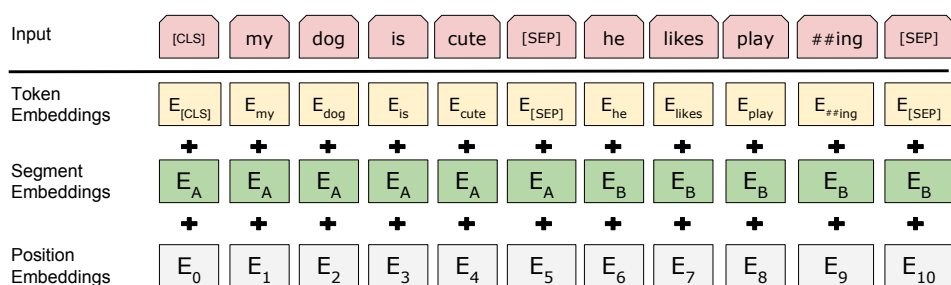


图2：BERT输入表示。输入嵌入是词嵌入、分割嵌入和位置嵌入的总和。

NSP任务与Jernite等人（2017）和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，只有句子嵌入被转移到下游任务，而BERT则将所有参数转移来初始化最终任务模型参数。

**预训练数据** 预训练过程很大程度上遵循现有语言模型预训练文献。对于预训练语料库，我们使用BooksCorpus（8亿词）（Zhu et al., 2015）和英文维基百科（25亿词）。对于维基百科，我们只提取文本段落，忽略列表、表格和标题。为了提取长的连续序列，使用文档级语料库而不是像十亿词基准（Billion Word Benchmark）（Chelba et al., 2013）这样的打乱句子级语料库至关重要。

### 3.2 BERT微调

微调非常简单，因为Transformer中的自注意力机制允许BERT对许多下游任务进行建模——无论它们涉及单文本还是文本对——只需更换合适的输入和输出。对于涉及文本对的应用，一种常见的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）；Seo等人（2017）。BERT使用自注意力机制将这两个阶段统一起来，因为使用自注意力机制对连接的文本对进行编码有效地包含了两个句子之间的交叉注意力。

对于每个任务，我们只需将特定于任务的输入和输出插入BERT并端到端微调所有参数。在输入中，来自预训练的句子A和句子B与（1）释义中的句子对，（2）蕴含中的假设-前提对，（3）问答中的问题-段落对类似。

（4）文本分类或序列标注中简并的文本- $\emptyset$ 对。在输出端，将标记表示馈送到输出层以进行标记级任务，例如序列标注或问答，并将[CLS]表示馈送到输出层以进行分类任务，例如蕴含或情感分析。

与预训练相比，微调相对便宜。论文中的所有结果都可以在一台云TPU上最多1小时内复制，或者在GPU上几小时内复制，起始于完全相同的预训练模型<sup>7</sup>。我们在第4节的相应小节中描述了特定于任务的细节。更多细节可以在附录A.5中找到。

## 4 个实验

在本节中，我们展示了在11个NLP任务上BERT微调的结果。

### 4.1 GLUE

通用语言理解评估（GLUE）基准（Wang等人，2018a）包含各种自然语言理解任务。GLUE数据集的详细描述包含在附录B.1中。

为了在GLUE上微调，我们根据第3节中描述的表示方法表示输入序列（对于单句或句子对），并将对应于第一个输入标记（[CLS]）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 $K$ 是标签的数量。我们使用 $C$ 和 $W$ 计算标准分类损失，即 $\log(\text{softmax}(CW^T))$ 。

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT <sub>BASE</sub>	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT <sub>LARGE</sub>	<b>86.7/85.9</b>	<b>72.1</b>	<b>92.7</b>	<b>94.9</b>	<b>60.5</b>	<b>86.5</b>	<b>89.3</b>	<b>70.1</b>	<b>82.1</b>

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.<sup>8</sup> 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<sub>LARGE</sub> 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.<sup>9</sup>

Results are presented in Table 1. Both BERT<sub>BASE</sub> and BERT<sub>LARGE</sub> 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<sub>BASE</sub> 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<sup>10</sup>, BERT<sub>LARGE</sub> obtains a score of 80.5, compared to OpenAI GPT, which obtains 72.8 as of the date of writing.

We find that BERT<sub>LARGE</sub> significantly outperforms BERT<sub>BASE</sub> 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,<sup>11</sup> 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) before 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-

<sup>9</sup>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<sub>BASE</sub> and BERT<sub>LARGE</sub>.

<sup>10</sup><https://gluebenchmark.com/leaderboard>

<sup>11</sup>QANet is described in Yu et al. (2018), but the system has improved substantially after publication.

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT <sub>BASE</sub>	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT <sub>LARGE</sub>	<b>86.7/85.9</b>	<b>72.1</b>	<b>92.7</b>	<b>94.9</b>	<b>60.5</b>	<b>86.5</b>	<b>89.3</b>	<b>70.1</b>	<b>82.1</b>

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示例的数量。“平均”列与官方GLUE评分略有不同，因为我们排除了有问题的WNLI集。<sup>8</sup> BERT和OpenAI GPT是单模型、单任务模型。QQP和MRPC报告F1分数，STS-B报告Spearman相关性，其他任务报告准确性分数。我们排除了使用BERT作为其组件之一的条目。

我们对所有GLUE任务使用批量大小为32，并在数据上微调3个epoch。对于每个任务，我们在Dev集上选择了最佳的微调学习率（在5e-5、4e-5、3e-5和2e-5之间）。此外，对于BERT<sub>LARGE</sub>，我们发现微调在小型数据集上有时不稳定，因此我们运行了几次随机重启，并在Dev集上选择了最佳模型。在随机重启中，我们使用相同的预训练检查点，但执行不同的微调数据混洗和分类器层初始化。<sup>9</sup>

结果如表1所示。BERT<sub>BASE</sub>和BERT<sub>LARGE</sub>在所有任务上均显著优于所有系统，平均准确率分别比现有技术提高了4.5%和7.0%。需要注意的是，BERT<sub>BASE</sub>和OpenAI GPT在模型架构方面几乎相同，只是注意力掩码不同。对于最大且报道最广泛的GLUE任务MNLI，BERT的准确率绝对提高了4.6%。在GLUE排行榜<sup>10</sup>上，BERT<sub>LARGE</sub>的得分是80.5，而截至撰写本文时，OpenAI GPT的得分是72.8。

我们发现，在所有任务中，BERTv9的性能都显著优于BERTv10，尤其是在训练数据非常少的情况下。模型大小的影响将在5.2节中更深入地探讨。

#### 4.2 SQuAD v1.1

斯坦福问答数据集（SQuAD v1.1）包含10万个由众包提供的问答对（Rajpurkar等人，2016年）。给定一个问题和一段来自

维基百科包含答案，任务是预测答案文本在段落中的跨度。

如图1所示，在问答任务中，我们将输入问题和段落表示为单个打包序列，问题使用A嵌入，段落使用B嵌入。我们仅在微调期间引入起始向量 $S \in \mathbb{R}^H$ 和结束向量 $E \in \mathbb{R}^H$ 。单词 $i$ 作为答案跨度起始的概率计算为 $T_i$ 和 $S$ 之间的点积，然后对段落中的所有单词进行softmax运算： $P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$ 。答案跨度的结束也使用类似的公式。从位置 $i$ 到位置 $j$ 的候选跨度的得分定义为 $S \cdot T_i + E \cdot T_j$ ，其中使用 $j \geq i$ 作为预测的最大得分跨度。训练目标是正确起始位置和结束位置的对数似然的总和。我们以5e-5的学习率和32的批量大小微调3个epoch。

表2显示了排行榜上的顶级参赛作品以及顶级已发表系统的结果（Seo等人，2017；Clark和Gardner，2018；Peters等人，2018a；Hu等人，2018）。SQuAD排行榜上的顶级结果没有提供最新的公共系统描述，<sup>11</sup>并且在训练其系统时可以使用任何公共数据。因此，我们在系统中使用了适度的数据增强，首先在TriviaQA（Joshi等人，2017）上进行微调，然后在SQuAD上进行微调。

我们的最佳系统在集成方面比排行榜上排名第一的系统高出+1.5个F1分数，作为单个系统则高出+1.3个F1分数。事实上，我们的单个BERT模型在F1分数方面优于排名第一的集成系统。无需TriviaQA微调

<sup>11</sup>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 <sub>BASE</sub> (Single)	80.8	88.5	-	-
BERT <sub>LARGE</sub> (Single)	84.1	90.9	-	-
BERT <sub>LARGE</sub> (Ensemble)	85.8	91.8	-	-
BERT <sub>LARGE</sub> (Sgl.+TriviaQA)	<b>84.2</b>	<b>91.1</b>	<b>85.1</b>	<b>91.8</b>
BERT <sub>LARGE</sub> (Ens.+TriviaQA)	<b>86.2</b>	<b>92.2</b>	<b>87.4</b>	<b>93.2</b>

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 <sub>LARGE</sub> (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.<sup>12</sup>

### 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:  $s_{\text{null}} = S \cdot C + E \cdot C$  to the score of the best non-null span

<sup>12</sup>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 <sub>BASE</sub>	81.6	-
BERT <sub>LARGE</sub>	<b>86.6</b>	<b>86.3</b>
Human (expert) <sup>†</sup>	-	85.0
Human (5 annotations) <sup>†</sup>	-	88.0

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

$s_{i,j} = \max_{j \geq i} S \cdot T_i + E \cdot T_j$ . We predict a non-null answer when  $s_{i,j} > s_{\text{null}} + \tau$ , where the threshold  $\tau$  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.

### 4.4 SWAG

The Situations With Adversarial Generations (SWAG) dataset contains 113k sentence-pair completion examples that evaluate grounded common-sense 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<sub>LARGE</sub> 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 <sub>BASE</sub> (Single)	80.8	88.5	-	-
BERT <sub>LARGE</sub> (Single)	84.1	90.9	-	-
BERT <sub>LARGE</sub> (Ensemble)	85.8	91.8	-	-
BERT <sub>LARGE</sub> (Sgl.+TriviaQA)	<b>84.2</b>	<b>91.1</b>	<b>85.1</b>	<b>91.8</b>
BERT <sub>LARGE</sub> (Ens.+TriviaQA)	<b>86.2</b>	<b>92.2</b>	<b>87.4</b>	<b>93.2</b>

表2：SQuAD 1.1 结果。BERT 集成包含 7 个使用不同预训练检查点和微调种子的系统。

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 <sub>LARGE</sub> (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。

在微调数据上，我们仅损失了0.1-0.4的F1值，仍然大幅超越所有现有系统。<sup>12</sup>

#### 4.3 SQuAD v2.0

SQuAD 2.0 任务扩展了 SQuAD 1.1 问题的定义，允许在提供的段落中不存在简短答案的可能性，使问题更贴近实际。

我们采用一种简单的方法来扩展 SQuAD v1.1 BERT 模型以用于此任务。我们将没有答案的问题视为答案跨度起始和结束位置都在 [CLS] 标记处。答案跨度起始和结束位置的概率空间被扩展到包含 [CLS] 标记的位置。在预测中，我们将无答案跨度的分数： $s_{\text{null}} = S \cdot C + E \cdot C$  与最佳非空跨度的分数进行比较

<sup>12</sup>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 <sub>BASE</sub>	81.6	-
BERT <sub>LARGE</sub>	<b>86.6</b>	<b>86.3</b>
Human (expert) <sup>†</sup>	-	85.0
Human (5 annotations) <sup>†</sup>	-	88.0

表4：SWAG开发集和测试集准确率。<sup>†</sup>人类表现根据SWAG论文的报告，通过100个样本进行衡量。

$s_{i,j} = \max_{j \geq i} S \cdot T_i + E \cdot T_j$ 。当 $\hat{s}_{i,j} > s_{\text{null}} + \tau$ 时，我们预测非空答案，其中阈值 $\tau$ 在开发集上选择以最大化F1值。我们没有使用TriviaQA数据来训练此模型。我们以5e-5的学习率和48的批量大小微调了2个epoch。

表3显示了与之前的排行榜条目和顶级已发表作品（Sun et al., 2018; Wang et al., 2018b）相比的结果，其中排除了使用BERT作为其组件之一的系统。我们观察到比之前的最佳系统提高了+5.1的F1值。

#### 4.4 SWAG

SWAG数据集包含113k个句子对完形填空示例，用于评估基于常识的推理（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延续。

在SWAG数据集上微调时，我们构建四个输入序列，每个序列都包含给定句子（句子A）和一个可能的延续（句子B）的串联。唯一引入的特定于任务的参数是一个向量，该向量与[CLS]标记表示 $C$ 的点积表示每个选择的得分，并用softmax层进行归一化。

我们使用2e-5的学习率和16的批量大小，对模型进行了3个epoch的微调。结果如表4所示。BERT<sub>LARGE</sub>的性能优于作者的基线ESIM+ELMo系统+27.1%，优于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，以便更好地理解它们各自的重要性。此外

Tasks	MNLI-m (Acc)	Dev Set			
		QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
BERT <sub>BASE</sub>	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<sub>BASE</sub> 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<sub>BASE</sub>:

**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 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<sub>BASE</sub> contains 110M parameters and BERT<sub>LARGE</sub> 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	MNLI-m (Acc)	Dev Set			
		QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
BERT <sub>BASE</sub>	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<sub>BASE</sub>架构对预训练任务进行消融研究。“No NSP”表示没有进行下一句预测任务的训练。“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，没有进行下一句预测。“+ BiLSTM”在微调期间，在“LTR + No NSP”模型之上添加了一个随机初始化的BiLSTM。

消融研究结果见附录C。

### 5.1 预训练任务的影响

我们通过使用与BERT<sub>BASE</sub>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，从而证明了BERT的深度双向性的重要性：

无NSP：一种使用“掩码语言模型”（MLM）进行训练但无需“下一句预测”（NSP）任务的双向模型。  
LTR和无NSP：一个仅使用左上下文模型，它使用标准的从左到右（LTR）语言模型进行训练，而不是掩码语言模型。仅限左侧的约束也应用于微调，因为去除它会引入预训练/微调不匹配，从而降低下游性能。此外，此模型在预训练时没有使用NSP任务。这可以直接与OpenAI GPT进行比较，但使用了我们更大的训练数据集、我们的输入表示和我们的微调方案。

我们首先检查NSP任务带来的影响。表5显示，移除NSP会显著损害QNLI、MNLI和SQuAD 1.1上的性能。接下来，我们通过比较“无NSP”和“LTR & 无NSP”来评估训练双向表示的影响。LTR模型在所有任务上的性能都比MLM模型差，在MRPC和SQuAD上的下降幅度很大。

对于SQuAD，直观上很清楚，LTR模型在词元预测上的表现会很差，因为词元级别的隐藏状态没有右侧上下文。为了认真尝试加强LTR系统，我们在其顶部添加了一个随机初始化的BiLSTM。这确实显著改善了SQuAD的结果，但是

结果仍然远不如预训练的双向模型。BiLSTM在GLUE任务上的性能受到影响。

我们认识到，也可以训练单独的LTR和RTL模型，并将每个标记表示为两个模型的串联，就像ELMo一样。但是：（a）这比单个双向模型贵一倍；（b）对于像QA这样的任务，这并不直观，因为RTL模型无法根据问题来确定答案；（c）它严格弱于深度双向模型，因为它可以在每一层都使用左右上下文。

### 5.2 模型大小的影响

在本节中，我们探讨了模型大小对微调任务准确性的影响。我们训练了一系列具有不同层数、隐藏单元和注意力头的BERT模型，同时使用与前面描述相同的超参数和训练过程。

表6显示了在选定GLUE任务上的结果。在此表中，我们报告了微调5次随机重启的平均开发集准确率。我们可以看到，更大的模型导致所有四个数据集的准确率都严格提高，即使对于只有3600个标记训练样本的MRPC也是如此，并且与预训练任务大相径庭。也许令人惊讶的是，我们能够在现有文献中已经相当大的模型之上取得如此显著的改进。例如，Vaswani等人（2017）探索的最大Transformer是（L=6，H=1024，A=16），编码器有1亿个参数，而我们在文献中发现的最大Transformer是（L=64，H=512，A=2），有2.35亿个参数（Al-Rfou等人，2018）。相比之下，BERT<sub>BASE</sub>包含1.1亿个参数，BERT<sub>LARGE</sub>包含3.4亿个参数。

长期以来，人们都知道，增加模型规模将导致在机器翻译和语言建模等大型任务上持续改进，表6中的保留训练数据的LM困惑度证明了这一点。然而，我们相信，这是第一个令人信服地证明，扩展到极端模型规模也会导致在非常小的规模任务上取得巨大改进的工作，前提是模型已经进行了充分的预训练。Peters等人（2018b）提出

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

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)	-	<b>93.1</b>
Fine-tuning approach		
BERT <sub>LARGE</sub>	96.6	92.8
BERT <sub>BASE</sub>	96.4	92.4
Feature-based approach (BERT <sub>BASE</sub> )		
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.

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.

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. BERT<sub>LARGE</sub> 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.

关于增加预训练双向语言模型层数（从两层到四层）对下游任务影响的结果喜忧参半，Malamud等人（2016）顺便提到，将隐藏维度大小从200增加到600有所帮助，但进一步增加到1000并没有带来进一步的改进。这两项先前的工作都使用了基于特征的方法——我们假设，当模型直接在下游任务上微调，并且只使用极少量的随机初始化的额外参数时，即使下游任务数据非常小，特定任务模型也可以从更大、更具表达能力的预训练表示中获益。

### 5.3 基于BERT的特征方法

到目前为止，所有展示的BERT结果都使用了微调方法，其中一个简单的分类层被添加到预训练模型中，并且所有参数都在下游任务上联合微调。然而，基于特征的方法（其中从预训练模型中提取固定特征）具有一定的优势。首先，并非所有任务都能轻易地用Transformer编码器架构表示，因此需要添加特定于任务的模型架构。其次，预先计算训练数据的昂贵表示一次，然后在此表示之上运行许多更便宜的模型，这具有主要的计算优势。

在本节中，我们将通过将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，并且我们包含数据提供的最大文档上下文。按照标准做法，我们将此表述为标记任务，但不使用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模型大小消融实验。#L = 层数；#H = 隐藏层大小；#A = 注意力头数量。“LM (ppl)”是保留的训练数据的掩码语言模型困惑度。

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)	-	<b>93.1</b>
Fine-tuning approach		
BERT <sub>LARGE</sub>	96.6	92.8
BERT <sub>BASE</sub>	96.4	92.4
Feature-based approach (BERT <sub>BASE</sub> )		
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。超参数使用验证集选择。报告的验证集和测试集分数是在使用这些超参数的5次随机重启的平均值。

输出层中的层。我们将第一个子词的表示作为命名实体识别标签集上词级别分类器的输入。

为了去除微调方法的影响，我们采用基于特征的方法，从一个或多个层*without*中提取激活值，而不对BERT的任何参数进行微调。这些上下文嵌入用作随机初始化的两层768维BiLSTM在分类层之前的输入。

结果如表7所示。BERT<sub>LARGE</sub>的性能与最先进的方法具有竞争力。性能最佳的方法将预训练Transformer的顶层四个隐藏层的标记表示连接起来，其F1值仅比微调整个模型低0.3。这表明BERT对于微调和基于特征的方法都非常有效。

## 6 结论

最近，利用语言模型进行迁移学习的经验改进表明，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，这些结果使即使是低资源任务也能从深度单向架构中受益。我们的主要贡献是将这些发现进一步推广到深度*bidirectional*架构，允许同一个预训练模型成功地处理各种NLP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 Sonntag. 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。在 *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. 基于更深层自注意力机制的字符级语言建模。 *arXiv preprint arXiv:1808.04444*.
- 久保田理惠, 张童. 2005. 从多个任务和未标记数据中学习预测结构的框架。 *Journal of Machine Learning Research*, 6(十一月):1817 – 1853.
- Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang和Danilo Giampiccolo. 2009. 第五届PASCAL文本蕴含识别挑战赛。In *TAC*. NIST.
- John Blitzer, Ryan McDonald和Fernando Pereira. 2006. 基于结构对应学习的领域自适应。见 *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. 《用于学习自然语言推理的大型标注语料库》。 *EMNLP*. 计算语言学协会。
- Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra和Jennifer C Lai. 1992. 《基于类别的n-gram自然语言模型》。 *Computational linguistics*, 18(4):467 – 479.
- Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio和Lucia Specia. 2017. 《SemEval-2017任务1: 语义文本相似性多语言和跨语言焦点评估》。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, Qi Ge, Thorsten Brants, Phillipp Koehn和Tony Robinson. 2013. 用于衡量统计语言模型进展的十亿词基准。 *arXiv preprint arXiv:1312.3005*.
- 陈卓, 张华, 张翔, 赵亮. 2018. Quora问题对。Kevin Clark, Minh-Thang Luong, Christopher D Manning和Quoc Le. 2018. 基于跨视角训练的半监督序列建模。In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914 – 1925.
- Ronan Collobert和Jason Weston. 2008. 一种统一的自然语言处理架构: 具有多任务学习的深度神经网络。In *Proceedings of the 25th international conference on Machine learning*, 第160-167页。ACM。
- Alexis Conneau, Douwe Kiela, Holger Schwenk, L c Barrault和Antoine Bordes. 2017. 《从自然语言推理数据中监督学习通用句子表示》。载于 *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。
- Dai Andrew M.和Le Quoc V. 2015. 半监督序列学习。In *Advances in neural information processing systems*, 第3079-3087页。
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li和L. Fei-Fei. 2009. ImageNet: 一个大型分层图像数据库。In *CVPR09*.
- William B Dolan和Chris Brockett. 2005年. 自动构建一个句子释义语料库。In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.
- William Fedus, Ian Goodfellow和Andrew M Dai. 2018. MaskGAN: 通过填充生成更好的文本。  $\$v^{\wedge}*\$$ .
- Dan Hendrycks和Kevin Gimpel. 2016. 用高斯误差线性单元桥接非线性与随机正则化器。 *CoRR*, abs/1606.08415.
- Felix Hill, Kyunghyun Cho和Anna Korhonen. 2016. 从未标记数据中学习句子的分布式表示。In *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和Sebastian Ruder. 2018. 用于文本分类的通用语言模型微调。In *ACL*. 计算语言学协会。
- 胡明浩, 彭玉兴, 黄震, 邱锡鹏, 魏福如, 周明. 2018. 用于机器阅读理解的强化记忆阅读器。In *IJCAI*.
- Yacine Jernite, Samuel R. Bowman和David Sontag. 2017. 《基于话语的快速无监督句子表示学习目标》。 *CoRR*, abs/1705.00557.
- 克里斯托弗·克拉克和卡特·加德纳. 2018. 简单有效的段落阅读理解。In *ACL*.

- 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：一个大规模远程监督阅读理解挑战数据集。In *ACL*。Ryan Kiros、Yukun Zhu、Ruslan R Salakhutdinov、Richard Zemel、Raquel Urtasun、Antonio Torralba和Sanja Fidler。2015. Skip-thought向量。In *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。In *International Conference on Machine Learning*，第1188 – 1196页。Hector J Levesque、Ernest Davis和Leora Morgenstern。2011. Winograd模式挑战。In *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有效框架。In *International Conference on Learning Representations*。Bryan McCann、James Bradbury、Caiming Xiong和Richard Socher。2017. 翻译中学习：上下文化词向量。In *NIPS*。Oren Melamud、Jacob Goldberger和Ido Dagan。2016. context2vec：使用双向LSTM学习通用上下文嵌入。In *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。In *Advances in Neural Information Processing Systems 26*，第3111 – 3119页。Curran Associates, Inc. Andriy Mnih和Geoffrey E Hinton。2009。一种可扩展的分层分布式语言模型。In D. Koller、D. S chuurmans、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。In *EMNLP*。Jeffrey Pennington、Richard Socher和Christopher D. Manning。2014. Glove：词表示的全局向量。In *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。In *ACL*。Matthew Peters、Mark Neumann、Mohit Iyyer、Matt Gardner、Christopher Clark、Kenton Lee和Luke Zettlemoyer。2018a. 深度上下文化词表示。In *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。在 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*，第1499 – 1509页。Alec Radford、Karthik Narasimhan、Tim Salimans和Ilya Sutskever。2018. 利用无监督学习改进语言理解。技术报告，OpenAI。Pranav Rajpurkar、Jian Zhang、Konstantin Lopyrev和Percy Liang。2016. SQuAD：100,000+个用于文本机器理解的问题。在 *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*，第2383 – 2392页。Minjion Seo、Aniruddha Kembhavi、Ali Farhadi和Hannah Hajishirzi。2017. 用于机器理解的双向注意力流。在 *ICLR*。Richard Socher、Alex Perelygin、Jean Wu、Jason Chuang、Christopher D Manning、Andrew Ng和Christopher Potts。2013. 基于情感树库的语义组合递归深度模型。在 *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：包含不可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：与语言无关的命名实体识别。在 *CoNLL*。Joseph Turian、Lev Ratinov和Yoshua Bengio。2010. 词表示：一种用于半监督学习的简单而通用的方法。在 *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. 注意力是你所需要的一切。在 *Advances in Neural Information Processing Systems*，第6000 – 6010页。Pascal Vincent、Hugo Larochelle、Yoshua Bengio和Pierre-Antoine Manzagol。2008. 使用去噪自动编码器提取和组合鲁棒特征。在 *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：一个多任务基准和分析平台

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 language 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* , 第353-355页。

王巍, 颜明, 吴辰. 2018b. 用于阅读理解和问答的多粒度层次注意融合网络. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 计算语言学协会.

Alex Warstadt, Amanpreet Singh和Samuel R Bowman. 2018. 神经网络可接受性判断. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia和Samuel R Bowman. 2018年. 一个用于通过推理进行句子理解的广覆盖挑战语料库. In *NAACL*.

吴永辉, 迈克·舒斯特, 陈志峰, 黎 quoc v, 穆罕默德·诺鲁齐, 沃尔夫冈·马切雷, 马克西姆·克里昆, 曹元, 高秦, 克劳斯·马切雷, 等. 2016. 谷歌神经机器翻译系统: 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和Hod Lipson. 2014. 深度神经网络中的特征迁移性如何? 在 *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: 结合局部卷积和全局自注意力进行阅读理解. In *ICLR*.

罗文·泽勒斯, 约纳坦·比斯克, 罗伊·施瓦茨和崔艺珍. 2018. SWAG: 一个用于常识推理的大规模对抗数据集. 见 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

朱玉昆, 瑞安·基罗斯, 里奇·泽梅尔, 鲁斯兰·萨拉赫乌季诺夫, 拉奎尔·乌尔塔松, 安东尼奥·托拉尔巴和桑娅·菲德勒. 2015年. 《对齐书籍和电影: 通过观看电影和阅读书籍实现类似故事的视觉解释》. 见 *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

我们将附录分为三个部分:

- BERT 的其他实现细节在附录 A 中给出;

• 附录 B 提供了我们实验的更多细节; 以及

- 附录 C 中提供了额外的消融研究。

我们对BERT进行了额外的消融研究, 包括: – 训练步数的影响; 以及 – 不同掩码过程的消融研究。

## BERT 的附加细节

### A.1 预训练任务示例

以下是预训练任务的示例。

掩码语言模型和掩码过程 假设未标记的句子是 my dog is hairy, 并且在随机掩码过程中我们选择了第 4 个标记 (对应于 hairy), 我们的掩码过程可以进一步说明为

- 80% 的时间: 用 [MASK] token 替换单词, 例如, my dog is hairy → my dog is [MASK]
- 10% 的情况下: 将单词替换为随机单词, 例如, 我的狗毛茸茸的 → 我的狗是苹果
- 10% 的时间: 保持单词不变, 例如, my dog is hairy → my dog is hairy。这样做是为了使表示偏向于实际观察到的单词。

这种方法的优点是, Transformer编码器不知道它会被要求预测哪些词, 也不知道哪些词被随机词替换了, 因此它被迫保留 *every* 输入 token 的分布式上下文表示。此外, 由于随机替换只发生在所有 token 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, 我们评估了此过程的影响。

与标准语言模型训练相比, 掩码语言模型每次批次只对 15% 的词元进行预测, 这表明模型可能需要更多预训练步骤。

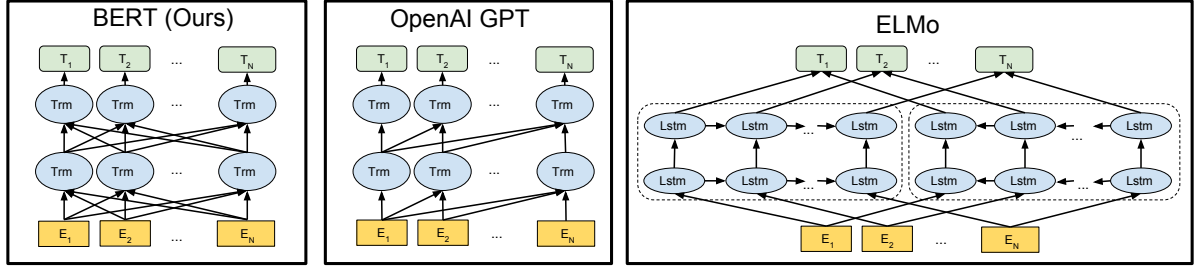


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT 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<sub>BASE</sub> was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total).<sup>13</sup> Training of BERT<sub>LARGE</sub> 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 pretraining 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

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

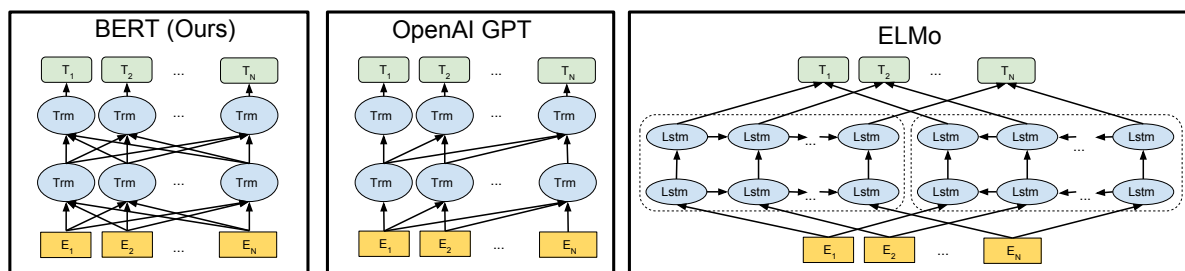


图3：预训练模型架构的差异。BERT使用双向Transformer。OpenAI GPT使用从左到右的Transformer。ELMo使用独立训练的从左到右和从右到左的LSTMs的串联来生成下游任务的特征。在这三个模型中，只有BERT的表示在所有层中都联合依赖于左右上下文。除了架构差异之外，BERT和OpenAI GPT是微调方法，而ELMo是基于特征的方法。

收敛。在C.1节中，我们证明了MLM的收敛速度略慢于从左到右的模型（预测每个token），但MLM模型的经验改进远远超过了增加的训练成本。

下一句预测 下一句预测任务可以用以下例子来说明。

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 预训练过程

为了生成每个训练输入序列，我们从语料库中采样两个文本片段，我们称之为“句子”，即使它们通常比单个句子长得多（但也可能更短）。第一个句子接收A嵌入，第二个句子接收B嵌入。50%的情况下，B是实际跟随A的下一个句子，50%的情况下它是一个随机句子，这是为了进行“下一个句子预测”任务。它们的采样方式使得组合长度为 $\leq 512$ 个词元。LM屏蔽在WordPiece分词后应用，均匀屏蔽率为15%，并且没有对部分词元进行特殊考虑。

我们使用批量大小为256个序列（256个序列 \* 512个标记 = 128,000个标记/批）进行训练，共训练1,000,000步，大约40

在包含33亿词的语料库上进行了多个轮次的训练。我们使用Adam优化器，学习率为 $1e-4$ ， $\beta_1 = 0.9$ ， $\beta_2 = 0.999$ ，L2权重衰减为0.01，并在前10000步进行学习率预热，然后线性衰减学习率。我们在所有层上使用0.1的dropout概率。我们遵循OpenAI GPT的做法，使用gelu激活函数（Hendrycks和Gimpel，2016）而不是标准的relu激活函数。训练损失是平均掩码语言模型似然和平均下一句预测似然的总和。

BERT<sub>BASE</sub>的训练在Pod配置的4个Cloud TPU（共16个TPU芯片）上进行。<sup>13</sup> BERT<sub>LARGE</sub>的训练在16个Cloud TPU（共64个TPU芯片）上进行。每个预训练都需要4天才能完成。

较长的序列成本过高，因为注意力机制与序列长度的平方成正比。为了加快我们实验中的预训练速度，我们在90%的步骤中使用128的序列长度对模型进行预训练。然后，我们使用512的序列长度训练剩余10%的步骤以学习位置嵌入。

## A.3 微调过程

微调时，大多数模型超参数与预训练时相同，批大小、学习率和训练轮数除外。丢弃概率始终保持在0.1。最佳超参数值是特定于任务的，但我们发现以下范围内的值在所有任务中都能很好地工作：

- 批次大小：16，32

<sup>13</sup><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 study 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 Table 1 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) :  $5e-5$ 、 $3e-5$ 、 $2e-5$  •
- 轮数 : 2、3、4

我们还观察到，大型数据集（例如，100k+ 个带标签的训练样本）对超参数的选择远不如小型数据集敏感。微调通常非常快，因此对上述参数进行穷举搜索并选择在开发集上性能最佳的模型是合理的。

#### A.4 BERT、ELMo和OpenAI GPT的比较

这里我们研究了最近流行的表示学习模型（包括ELMo、OpenAI GPT和BERT）的差异。模型架构之间的比较如图3所示。需要注意的是，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。

与BERT最相似的现有预训练方法是OpenAI GPT，它在一个大型文本语料库上训练了一个从左到右的Transformer语言模型。事实上，BERT的许多设计决策都是有意使其尽可能接近GPT，以便能够对这两种方法进行最小化比较。这项工作的核心论点是，第3.1节中提出的双向性和两个预训练任务占据了大部分经验改进，但我们也注意到，BERT和GPT的训练方式之间还存在其他一些差异：

- GPT 在 BooksCorpus（8 亿词）上进行训练；BERT 在 BooksCorpus（8 亿词）和维基百科（25 亿词）上进行训练。
- GPT 使用句子分隔符 ([SEP]) 和分类器标记 ([CLS])，这些标记仅在微调时引入；BERT 在预训练期间学习 [SEP]、[CLS] 和句子 A/B 嵌入。
- GPT 使用批量大小为 32,000 词的训练步数为 100 万步；BERT 使用批量大小为 128,000 词的训练步数为 100 万步。
- GPT 对所有微调实验使用相同的学习率  $5e-5$ ；BERT 选择特定于任务的微调学习率，该学习率在开发集上性能最佳。

为了隔离这些差异的影响，我们在5.1节进行了消融实验，结果表明，大部分改进实际上来自两个预训练任务及其所带来的双向性。

#### A.5 不同任务的微调示例

图4展示了在不同任务上微调BERT的示例。我们的特定任务模型是通过将BERT与一个额外的输出层结合而成的，因此只需要学习少量参数。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。在图中， $E$ 表示输入嵌入， $T_i$ 表示词元 $i$ 的上下文表示，[CLS]是用于分类输出的特殊符号，[SEP]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中给出的GLUE结果取自<https://gluebenchmark.com/排行榜>和<https://blog.openai.com/language-unsupervised>。GLUE基准包含以下数据集，其描述最初总结于Wang等人的论文(2018a)：

MNLI（多体裁自然语言推理）是一个大规模的、众包的蕴含分类任务（Williams等人，2018）。给定一对句子，目标是预测第二句相对于第一句是*entailment*、*contradiction*还是*neutral*。

QQP（Quora问题对）是一个二元分类任务，其目标是确定Quora上提出的两个问题在语义上是否等价（Chen等人，2018）。

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



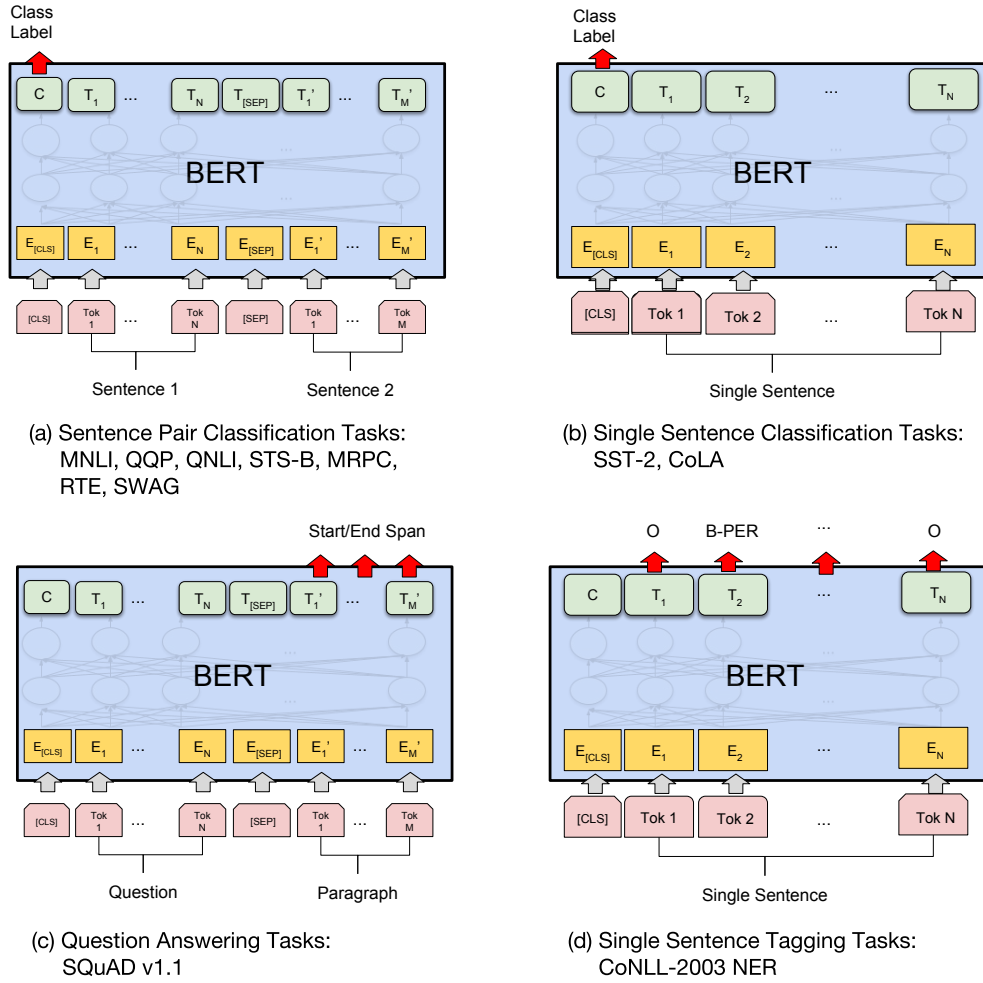


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

**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).<sup>14</sup>

**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,<sup>15</sup> 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-

<sup>14</sup>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.

<sup>15</sup><https://gluebenchmark.com/faq>

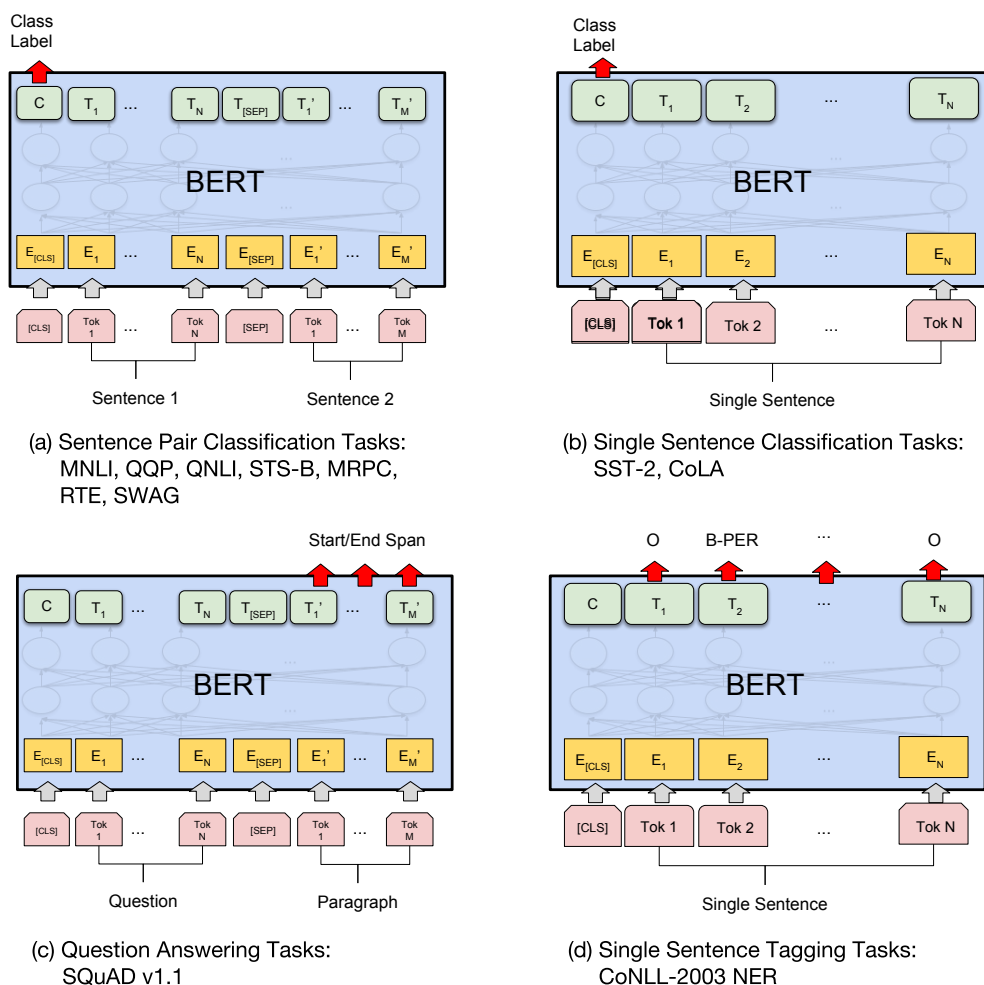


图4：在不同任务上微调BERT的示例图。

SST-2 斯坦福情感树库是一个二元单句分类任务，由从影评中提取的句子及其人工标注的情感构成（Socher等人，2013）。

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

STS-B 语义文本相似性基准测试是一个句子对的集合，这些句子对取自新闻标题和其他来源（Cer等人，2017年）。它们被标注了1到5的分数，表示两个句子在语义含义上的相似程度。

MRPC 微软研究院释义语料库由从在线新闻来源自动提取的句子对组成，并带有人工标注

用于判断句对在语义上是否等价（Dolan和Brockett，2005）。

RTE（识别文本蕴涵）是一个二元蕴涵任务，类似于MNLI，但训练数据少得多（Bentivogli等人，2009）。<sup>14</sup>

WNLI Winograd自然语言推理数据集是一个小型数据集（Levesque等人，2011年）。GLUE网页指出该数据集的构建存在问题，<sup>15</sup>以及每个提交到GLUE的训练系统，其性能都比预测多数类的65.1%基准准确率差。因此，为了公平起见，我们排除了这个数据集，以保证对OpenAI GPT的公平性。对于我们的GLUE提交，我们总是预测多数类。

<sup>14</sup>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.

<sup>15</sup><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<sub>BASE</sub> 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.

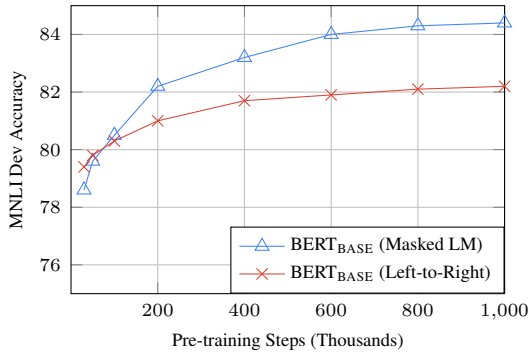


Figure 5: Ablation over number of training steps. This shows the MNLI accuracy after fine-tuning, starting from model parameters that have been pre-trained for  $k$  steps. The x-axis is the value of  $k$ .

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 附加消融研究

### C.1 训练步数的影响

图5展示了在经过 $k$ 步预训练的检查点微调后，MNLI Dev的准确率。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数据（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<sub>v2</sub>在训练100万步后，MNLI的准确率提高了近1.0%。

2. 问题：由于每个批次只预测15%的单词而不是每个单词，因此MLM预训练是否比LTR预训练收敛速度更慢？

答案：MLM 模型的收敛速度略慢于 LTR 模型。然而，就绝对精度而言，MLM 模型几乎立即开始优于 LTR 模型。

### C.2 不同掩码程序的消融实验

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，采用了一种混合策略来掩码目标token。以下是评估不同掩码策略效果的消融研究。

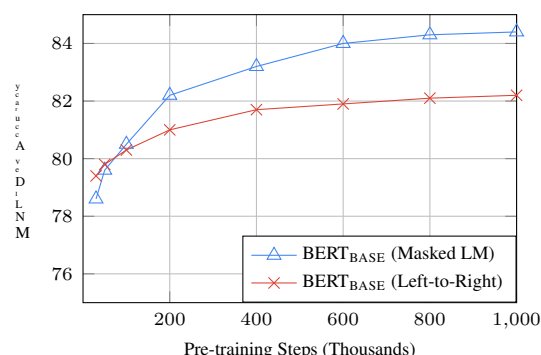


图5：训练步数消融实验。该图显示了微调后的MNLI准确率，起始模型参数已预训练 $k$ 步。x轴表示 $k$ 的值。

需要注意的是，掩码策略的目的是减少预训练和微调之间的不匹配，因为[MASK]符号在微调阶段从未出现过。我们报告了MNLI和NER的Dev结果。对于NER，我们报告了微调和基于特征的方法的结果，因为我们预计基于特征的方法的不匹配将会被放大，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token替换为[MASK]符号用于MLM；SAME表示我们将目标token保持不变；RND表示我们将目标token替换为另一个随机token。

表的左侧数字代表MLM预训练期间使用的特定策略的概率（BERT使用80%、10%、10%）。本文的右侧代表Dev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5.3节中被证明是最佳方法。

从表中可以看出，微调对不同的掩码策略具有惊人的鲁棒性。然而，正如预期的那样，当将基于特征的方法应用于NER时，仅使用MASK策略是有问题的。有趣的是，仅使用RND策略的性能也比我们的策略差得多。