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

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

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

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

1 Introduction

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

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

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

In this paper, we improve the fine-tuning based approaches by proposing BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. BERT alleviates the previously mentioned unidirectionality constraint by using a "masked language model" (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked

arXiv:1810.04805v2 [cs.CL] 24 May 2019

BERT：基于深度双向变换器的语言理解预训练模型

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

摘要

我们提出一种名为BERT（双向编码器表示）的新型语言表示模型。与近期语言表示模型（Peters et al., 2018a; Radford et al., 2018）不同，BERT通过在所有层中同时联合左右上下文条件，实现了从无标记文本中预训练深度双向表示的能力。因此，预训练的BERT模型仅需添加一个输出层即可进行微调，从而在无需大幅修改任务特定架构的情况下，为问答和语言推理等广泛任务创建最先进模型。

BERT在概念上简单而在实践中强大。它在十一个自然语言处理任务上取得了新的最先进成果，包括将GLUE评分提升至80.5%（绝对提升7.7个百分点），MultiNLI准确率达86.7%（绝对提升4.6%），SQuAD v1.1问答测试F1值达93.2（绝对提升1.5分），SQuAD v2.0测试F1值达83.1（绝对提升5.1分）。

1 引言

语言模型预训练已被证实能有效提升多种自然语言处理任务的性能（Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018）。这些任务包括句级任务（如自然语言推理（Bowman et al., 2015; Williams et al., 2018）和同义替换（Dolan and Brockett, 2005），这类任务通过整体分析句子来预测句间关系；还包括词元级任务如命名实体识别和问答系统，要求模型在词元层面生成精细化输出（Tjong Kim Sang and De Meulder, 2003; Rajpurkar等人，2016）。

将预训练语言表示应用于下游任务的现有策略主要有两种：特征式方法与微调方法。特征式方法（如ELMo模型，Peters等人，2018a）采用特定任务架构，将预训练表示作为附加特征纳入其中。 微调方法（如生成式预训练变换器OpenAI GPT（Radford et al., 2018））仅引入少量任务特异性参数，通过直接微调所有预训练参数实现下游任务训练。两种方法在预训练阶段采用相同的目标函数，即利用单向语言模型学习通用语言表征。

我们认为现有技术限制了预训练表示的能力，尤其体现在微调方法上。 主要局限在于标准语言模型采用单向处理，这限制了预训练阶段可选用的架构。例如OpenAI的GPT模型采用左至右架构，Transformer模型（Vaswani et al., 2017）的自注意力层中每个令牌仅能关注前向令牌。 此类限制对句子级任务效果欠佳，且在将基于微调的方法应用于问答等令牌级任务时可能造成严重损害——这类任务亟需整合双向上下文信息。

本文通过提出BERT（基于Transformer的双向编码器表示）改进微调方法。BERT采用受Cloze任务（Taylor, 1953）启发的"遮蔽语言模型"（MLM）预训练目标，缓解了先前提到的单向性限制。 遮蔽语言模型通过随机遮蔽输入中的部分词元，其目标是预测被遮蔽词元的原始词汇ID。

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

- We demonstrate the importance of bidirectional pre-training for language representations. Unlike Radford et al. (2018), which uses unidirectional language models for pre-training, BERT uses masked language models to enable pre-trained deep bidirectional representations. This is also in contrast to Peters et al. (2018a), which uses a shallow concatenation of independently trained left-to-right and right-to-left LMs.
- We show that pre-trained representations reduce the need for many heavily-engineered task-specific architectures. BERT is the first fine-tuning based representation model that achieves state-of-the-art performance on a large suite of sentence-level *and* token-level tasks, outperforming many task-specific architectures.
- BERT advances the state of the art for eleven NLP tasks. The code and pre-trained models are available at `https://github.com/google-research/bert`.

2 Related Work

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

2.1 Unsupervised Feature-based Approaches

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

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

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

2.2 Unsupervised Fine-tuning Approaches

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

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

仅基于上下文进行词汇处理。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，从而实现深度双向Transformer的预训练。除遮蔽语言模型外，我们还采用"下一句预测"任务联合预训练文本对表示。本文贡献如下：

• 我们证明了双向预训练对语言表征的重要性。与 Radford 等人（ 2018 ）采用单向语言模型进行预训练不同，BERT运用遮蔽语言模型实现深度双向预训练表示。这亦区别于Peters等人（2018a）采用独立训练的左至右与右至左语言模型浅层拼接的方法。

• 我们证明预训练表示能减少对大量高度工程化的任务特定架构的需求。BERT是首个基于微调的表示模型，在大量句子级和词级任务上均达到最先进水平，性能超越众多任务特定架构。

• BERT在十一项自然语言处理任务中实现了技术突破。相关代码及预训练模型可于https://github.com/google-research/bert获取。

2 相关研究

预训练通用语言表示技术历史悠久，本节将简要回顾最广泛采用的方法。

2.1 无监督特征方法

数十年来，学习广泛适用的词汇表示一直是活跃的研究领域，涵盖非神经网络方法（Brown等人，1992；Ando和Zhang，2005；Blitzer等人，2006）与神经网络方法（Mikolov等人，2013；Pennington等人，2014）。预训练词嵌入是现代自然语言处理系统的重要组成部分，相较于从零开始学习的嵌入，其性能显著提升（Turian et al., 2010）。为了预训练词嵌入向量，人们采用了从左到右的语言建模目标（Mnih 和 Hinton，2009），以及从左、右上下文中区分正确和错误词的目标（Mikolov 等，2013）。

这些方法已被推广至更粗粒度的维度，例如句子嵌入（Kiros等人，2015；Logeswaran和Lee，2018）或段落嵌入（Le和Mikolov，2014）。为训练句子表征，先前的研究采用目标函数对候选下一句进行排序（Jernite等人，2017； Logeswaran and Lee, 2018），基于前文表征的从左到右生成下文词序（Kiros et al., 2015），或基于去噪自编码器的目标函数（Hill et al., 2016）。

ELMo及其前身（Peters等人，2017年，2018a年）沿着不同维度对传统词嵌入研究进行了泛化。它们从左至右和右至左的语言模型中提取上下文敏感特征。 每个词的上下文表示是左右向与右向表示的拼接。当将上下文词嵌入与现有特定任务架构整合时， ELMo在多个主要NLP基准测试中实现了突破性进展（Peters et al., 2018a），涵盖问答系统（Rajpurkar et al., 2016）、情感分析（Socher et al., 2013）及命名实体识别（Tjong Kim Sang and De Meulder, 2003）。Melamud 等人（ 2016 ）提出通过任务学习上下文表示，利用LSTM从左右上下文预测单词。该模型与ELMo类似，采用特征表示而非深度双向机制。Fedus等人（2018）证明填空任务可用于提升文本生成模型的鲁棒性。

2.2 无监督微调方法

与基于特征的方法类似，该方向的首项研究仅预训练了来自无标签文本的词嵌入参数（Colbert and Weston, 2008）。

近年来，基于上下文的句/文档编码器通过无标签文本预训练，再针对监督式下游任务进行微调（Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018）。这些方法的优势在于无需从头学习大量参数。至少部分得益于此优势，OpenAI GPT（Radford等人，2018）在GLUE基准测试（Wang等人，2018a）的众多句子级任务中取得了当时最先进的结果。从左到右的语言模型——

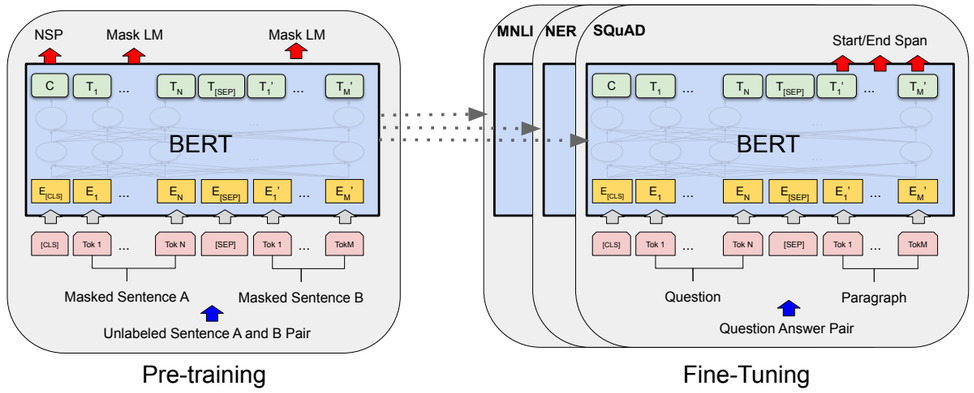


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are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

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

2.3 Transfer Learning from Supervised Data

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

3 BERT

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

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

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

In this work, we denote the number of layers (i.e., Transformer blocks) as L, the hidden size as H, and the number of self-attention heads as A.[3] We primarily report results on two model sizes: **BERT$_{\text{BASE}}$** (L=12, H=768, A=12, Total Parameters=110M) and **BERT$_{\text{LARGE}}$** (L=24, H=1024, A=16, Total Parameters=340M).

BERT$_{\text{BASE}}$ was chosen to have the same model size as OpenAI GPT for comparison purposes. Critically, however, the BERT Transformer uses bidirectional self-attention, while the GPT Transformer uses constrained self-attention where every token can only attend to context to its left.[4]

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

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

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

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

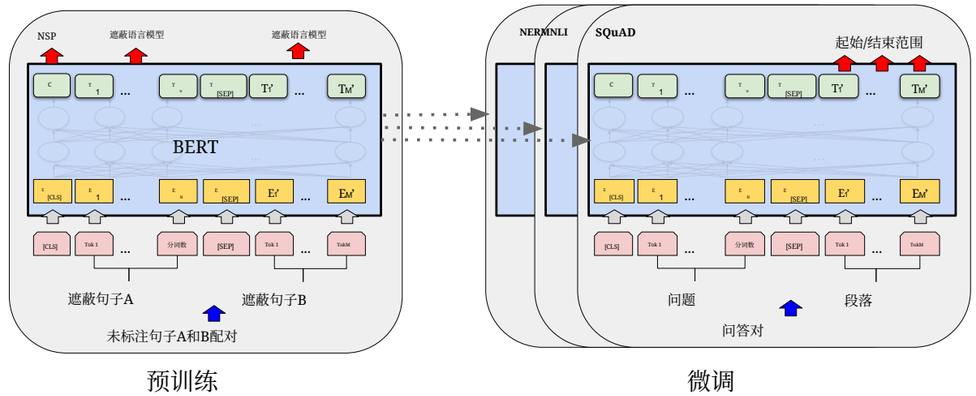


图1：BERT整体预训练与微调流程。除输出层外，预训练与微调采用相同架构。预训练模型参数用于初始化不同下游任务的模型。微调过程中所有参数均参与调整。 [CLS]是添加在每个输入示例前方的特殊符号， [SEP]则是用于分隔（如问题/答案）的特殊分隔令牌。

（Howard和Ruder，2018；Radford等人，2018；Dai和Le，2015）

2.3 基于监督数据的迁移学习

已有研究表明，从大规模数据集的监督任务（如自然语言推理[Conneau et al., 2017]和机器翻译[McCann et al., 2017]）中进行迁移学习效果显著。 计算机视觉研究同样证明了从大型预训练模型进行迁移学习的重要性，其中一种有效方案是对ImageNet预训练模型进行微调（Deng等人，2009；Yosinski等人，2014）。

3 BERT

本节将介绍BERT及其详细实现方案。我们的框架包含两个步骤：预训练与微调。预训练阶段，模型通过不同预训练任务在无标签数据上进行训练。 在微调阶段，首先使用预训练参数初始化BERT模型，随后通过下游任务的标注数据对所有参数进行微调。尽管各任务初始化参数相同，但每个下游任务均拥有独立的微调模型。图1所示的问答任务将作为本节的贯穿性示例。

BERT的独特之处在于其跨任务的统一架构。存在微调——

预训练架构与最终下游架构之间的显著差异。

模型架构BERT的模型架构基于Vaswani等人（2017）描述的原始实现方案，采用多层双向Transformer编码器，该方案已发布于tensor2tensor库。[1] 鉴于Transformer模型已广泛应用且我们的实现与原始版本基本一致，本文将省略模型架构的详尽背景说明，建议读者参阅Vaswani等人（2017）的论文及《注释版Transformer》等优质指南。[2]

本研究中，我们用 L表示层数（即Transformer模块数），用H表示隐藏维度，用 A表示自注意力头数量。[3]我们主要报告两种模型规模的结果：BERT$_{\text{BASE}}$ (L=12, H=768, A=12, 总参数数 =110M) 和 BERT$_{\text{LARGE}}$ (L=24, H =1024, A=16, 总参数数=340M)。

为便于对比，BERT$_{\text{BASE}}$ 被设定为与OpenAI GPT相同的模型规模。但关键区别在于：BERT变换器采用双向自注意力机制，而GPT变换器采用约束自注意力机制——每个标记词仅能关注其左侧上下文。[4]

[1] https://github.com/tensorflow/tensor2tensor [2] http://nlp.seas.harvard.edu/2018/04/03/attention.html[3]所有情况下我们均将前馈/过滤器尺寸设为 $4H$，即 $H = 768$ 采用3072， $H = 1024$采用4096。[4]需注意文献中双向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., ⟨ Question, Answer ⟩) in one token sequence. Throughout this work, a "sentence" can be an arbitrary span of contiguous text, rather than an actual linguistic sentence. A "sequence" refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.

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

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

3.1 Pre-training BERT

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

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

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

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

Task #2: Next Sentence Prediction (NSP) Many important downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) are based on understanding the *relationship* between two sentences, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, we pre-train for a binarized *next sentence prediction* task that can be trivially generated from any monolingual corpus. Specifically, when choosing the sentences `A` and `B` for each pre-training example, 50% of the time `B` is the actual next sentence that follows `A` (labeled as `IsNext`), and 50% of the time it is a random sentence from the corpus (labeled as `NotNext`). As we show in Figure 1, C is used for next sentence prediction (NSP).[5] Despite its simplicity, we demonstrate in Section 5.1 that pre-training towards this task is very beneficial to both QA and NLI. [6]

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

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

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

输入/输出表示法为使BERT处理多样化下游任务，我们的输入表示法能在单一令牌序列中明确呈现单句或双句对（如⟨问题，答案⟩）。本研究中"句子"可指任意连续文本片段，而非实际语言学意义上的句子。"序列"指输入到BERT的令牌序列，可以是单个句子或两个句子组合而成。

我们采用WordPiece嵌入（Wu et al., 2016）并设置30,000词汇量。每个序列的首个词始终为特殊分类词（[CLS]）。对应该词的最终隐藏状态将作为分类任务的聚合序列表示。句子对被打包为单一序列。我们通过两种方式区分句子：首先使用特殊分隔符（[SEP]）进行标记；其次为每个词添加学习得到的嵌入向量，标识其所属句子（A或B）。如图1所示，输入嵌入向量记为 E，特殊分隔符对应的最终隐向量记为 C。其次，为每个标记添加学习得到的嵌入向量，标识其所属句子（A或B）。如图1所示，输入嵌入向量记为 E，特殊标记 [CLS] 的最终隐藏向量记为 $C \in \mathbb{R}^H$，输入标记 i^{th} 的最终隐藏向量记为 $T_i \in \mathbb{R}^H$。

对于给定词元，其输入表示通过累加对应词元、分段及位置嵌入构建而成。该构建过程的可视化示意如图2所示。

3.1 BERT的预训练

与Peters等人（2018a）及Radford等人（2018）不同，我们未采用传统的从左到右或从右到左语言模型对BERT进行预训练。相反，我们通过本节所述的两项无监督任务对BERT进行预训练。此步骤如图1左侧所示。

任务 #1：遮蔽式语言模型直观而言，深度双向模型理应比左至右模型或浅层串联的双向模型更强大。遗憾的是，标准条件式语言模型仅能进行左至右或右至左训练——双向条件处理会使每个词间接"自我参照"，导致模型能在多层上下文中轻易预测目标词。

前者常被称为"Transformer编码器"，而仅处理左侧上下文的版本则称为"Transformer解码器"，因其可用于文本生成。

为训练深度双向表示，我们随机遮蔽输入令牌的特定比例，随后预测这些被遮蔽的令牌。此过程称为"遮蔽语言模型"(MLM)，尽管文献中常将其称为填空任务(Taylor, 1953)。此时，对应掩码令牌的最终隐藏向量将输入词表进行软最大似然输出，与标准语言模型一致。在所有实验中，我们随机遮蔽每个序列中 15 %的WordPiece令牌。与去噪自编码器（Vincent et al., 2008）不同，我们仅预测被遮蔽的单词而非重建完整输入。

虽然这使我们能够获得双向预训练模型，但其弊端在于造成预训练与微调阶段的错位——因为 [MASK] 令牌在微调过程中不会出现。为缓解此问题，我们并非始终用实际 [MASK] 令牌替换"遮蔽"词。训练数据生成器会随机选取15%的令牌位置进行预测。若选中第 i个令牌，则按以下概率替换第 i个令牌：(1)80%概率替换为 [MASK] 令牌；(2)10%概率替换为随机令牌；(3)10%概率保留原始令牌。随后，T_i 将用于通过交叉熵损失预测原始令牌。我们在附录C.2中比较了该流程的变体。

任务 #2：下句预测（NSP）诸如问答（QA）和自然语言推理（NLI）等重要下游任务，均基于对两句之间关系的理解——而语言模型无法直接捕捉这种关系。为训练理解句子关系的模型，我们针对二值化下一句预测任务进行预训练——该任务可从任意单语语料库中轻松生成。具体而言，在为每个预训练样本选择句子A和B时，50%的情况下B是紧接在A之后的实际下一句（标记为IsNext），50%的情况下则是语料库中随机抽取的句子（标记为NotNext）。如图1所示，C 用于下一句预测（NSP）。[5] 尽管该任务简单，我们在第5.1节证明针对此任务的预训练对QA和NLI均大有裨益。 [6]

[5]最终模型在NSP任务上达到97%-98%的准确率。[6]该向量 C 未经微调时并非有效的句子表示，因其采用NSP进行训练。

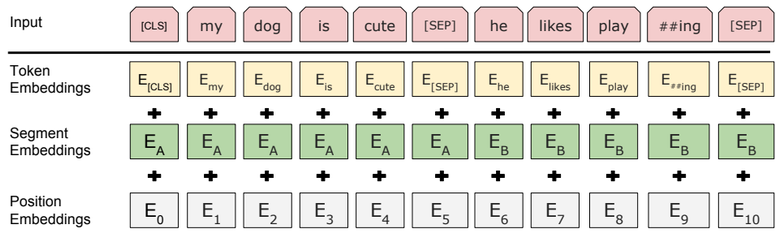


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

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

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

3.2 Fine-tuning BERT

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

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

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

4 Experiments

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

4.1 GLUE

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

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

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

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

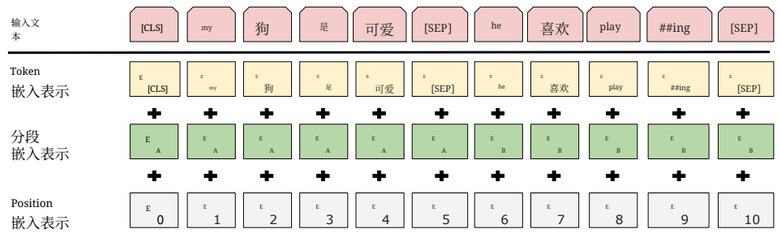


图2：BERT输入表示。输入嵌入是词嵌入、分段嵌入和位置嵌入的总和。

NSP任务与Jernite等人（2017）及Logeswaran和Lee（2018）研究中采用的表征学习目标密切相关。但先前研究仅将句子嵌入转移至下游任务，而BERT则转移全部参数以初始化终端任务模型参数。

预训练数据预训练流程基本遵循现有语言模型预训练文献。预训练语料库采用BooksCorpus（8亿词）（Zhu et al., 2015）及英文维基百科（25亿词）。针对维基百科数据，我们仅提取文本段落，忽略列表、表格及标题。采用文档级语料库而非Billion Word Benchmark（Chelba等人，2013）这类打乱的句子级语料库至关重要，这能有效提取长连续序列。

3.2 BERT的微调

由于Transformer中的自注意力机制，BERT可通过替换相应输入输出，轻松建模多种下游任务——无论涉及单文本还是文本对。对于涉及文本对的应用，常见模式是先独立编码文本对再应用双向交叉注意力机制，如Parikh等人（2016）和Seo等人（2017）的研究。而BERT则利用自注意力机制统一这两个阶段——通过自注意力对拼接后的文本对进行编码，实质上实现了两句之间的双向交叉注意力。

针对每项任务，我们只需将特定任务的输入输出数据接入BERT模型，并进行端到端参数微调。在输入端，预训练阶段的句子A与句子B分别对应：(1) 转述任务中的句子对，(2) 蕴含任务中的假设-前提对，(3) 问答任务中的问题-段落对，以及(4) 在文本分类或序列标注中出现退化文本-$\varnothing$对。输出端，令词元表示输入输出层处理词元级任务（如序列标注或问答），而［CLS］表示则输入分类任务（如蕴含判断或情感分析）的输出层。

相较于预训练，微调成本相对较低。所有实验结果均可在单台Cloud TPU上1小时内复现，或在GPU上数小时内完成（基于完全相同的预训练模型）。[7] 具体任务细节详见第4节对应子章节，更多内容参见附录A.5。

4 实验验证

本节展示BERT在11项自然语言处理任务上的微调结果。

4.1 GLUE

通用语言理解评估（GLUE）基准（Wang et al., 2018a）是一组多样化的自然语言理解任务集合。GLUE数据集的详细描述详见附录B.1。

在GLUE数据集上进行微调时，我们按第3节所述表示输入序列（单句或句对），并采用首个输入令牌（［CLS］）对应的最终隐藏向量 $C \in \mathbb{R}^H$ 作为聚合表示。微调过程中新增的参数仅为分类层权重 $W \in \mathbb{R}^{K \times H}$，其中 K 表示标签数量。我们通过 C 和 W 计算标准分类损失，即log(softmax(CW^T))。

[7] 例如，BERT SQuAD模型可在单个云TPU上约30分钟内完成训练，实现91.0%的开发集F1分数。[8] 详见https://gluebenchmark.com/faq中的(10)。

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

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

We use a batch size of 32 and fine-tune for 3 epochs over the data for all GLUE tasks. For each task, we selected the best fine-tuning learning rate (among 5e-5, 4e-5, 3e-5, and 2e-5) on the Dev set. Additionally, for $\text{BERT}_{\text{LARGE}}$ we found that fine-tuning was sometimes unstable on small datasets, so we ran several random restarts and selected the best model on the Dev set. With random restarts, we use the same pre-trained checkpoint but perform different fine-tuning data shuffling and classifier layer initialization.[9]

Results are presented in Table 1. Both $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ outperform all systems on all tasks by a substantial margin, obtaining 4.5% and 7.0% respective average accuracy improvement over the prior state of the art. Note that $\text{BERT}_{\text{BASE}}$ and OpenAI GPT are nearly identical in terms of model architecture apart from the attention masking. For the largest and most widely reported GLUE task, MNLI, BERT obtains a 4.6% absolute accuracy improvement. On the official GLUE leaderboard[10], $\text{BERT}_{\text{LARGE}}$ obtains a score of 80.5, compared to OpenAI GPT, which obtains 72.8 as of the date of writing.

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

4.2 SQuAD v1.1

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

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

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

Our best performing system outperforms the top leaderboard system by +1.5 F1 in ensembling and +1.3 F1 as a single system. In fact, our single BERT model outperforms the top ensemble system in terms of F1 score. Without TriviaQA fine-

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

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

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

系统	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**平均** -
OpenAI问世前的顶尖水平	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

表1：GLUE测试结果（由评估服务器https://gluebenchmark.com/leaderboard评分）。各任务下方数字表示训练样本数量。"平均"列与官方GLUE评分略有差异，因我们排除了存在问题的WNLI数据集。[8] BERT和OpenAI GPT均为单模型单任务。QQP和MRPC任务采用F1分数，STS-B任务采用斯皮尔曼相关系数，其余任务采用准确率评分。我们排除了将BERT作为组件之一的研究成果。

我们采用批量大小32，针对所有GLUE任务的数据进行3个 epoch 的微调。每个任务均在开发集上选取最佳微调学习率（选项包括5e-5、4e-5、3e-5和2e-5）。此外，针对$\text{BERT}_{\text{LARGE}}$模型，我们发现小规模数据集上的微调过程有时不够稳定，因此进行了多次随机重启，并从开发集上选取最佳模型。在随机重启过程中，我们使用相同的预训练检查点，但执行不同的微调数据洗牌和分类器层初始化操作。[9]

结果见表1。$\text{BERT}_{\text{BASE}}$ 和$\text{BERT}_{\text{LARGE}}$ 在所有任务上均以显著优势超越现有系统，较前沿水平分别提升4.5%和7.0%的平均准确率。需注意$\text{BERT}_{\text{BASE}}$ 与OpenAI GPT在模型架构上除注意力遮蔽机制外几乎完全一致。 在规模最大且报道最广泛的GLUE任务MNLI中，BERT实现了4.6%的绝对准确率提升。在官方GLUE排行榜[10]上，$\text{BERT}_{\text{LARGE}}$ 获得80.5分，而截至撰稿时OpenAI GPT的得分仅为72.8分。

我们发现$\text{BERT}_{\text{LARGE}}$ 在所有任务中均显著优于$\text{BERT}_{\text{BASE}}$，尤其在训练数据极少的场景下表现突出。模型规模的影响将在第5.2节进行更深入的探讨。

4.2 SQuAD v1.1

斯坦福问答数据集（SQuAD v1.1）包含10万组众包问答对（Rajpurkar等，2016）。给定一个问题及其对应的文本段落，维基百科包含答案，任务是在该段落中预测答案文本片段。

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一打包序列，其中问题采用A嵌入，段落采用B嵌入。 在微调阶段，我们仅引入起始向量 $S \in \mathbb{R}^H$ 和结束向量 $E \in \mathbb{R}^H$ 。单词 i 作为答案区段起点的概率，通过计算 T_i 与 S 的点积，再对段落中所有单词进行softmax归一化获得： $P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$ 答案区间的末端采用类似公式计算。候选区间从位置 i 到位置 j 的得分定义为 $S \cdot T_i + E \cdot T_j$，其中得分最高的区间 $j \geq i$ 被用作预测结果。训练目标函数为正确起始位置与结束位置对数似然值之和。 采用学习率5e-5、批量大小32的参数进行3个 epoch 的微调训练。

表2展示了排行榜顶尖系统及已发表顶尖系统的结果（Seo等人，2017；Clark和Gardner，2018；Peters等人，2018a；Hu等人，2018）。 SQuAD排行榜的顶尖结果尚未提供最新的公开系统描述，[11] 且允许在训练系统时使用任何公开数据。因此我们在系统中采用适度的数据增强策略：先在TriviaQA（Joshi等人，2017）上进行微调，再转至SQuAD进行微调。

我们的最佳系统在集成模式下比排行榜首系统高出 +1.5 F1值，作为单一系统则高出+1.3 F1值。事实上，我们的单一BERT模型在F1分数上已超越顶级集成系统。若未进行TriviaQA微调——

[9]GLUE数据集分发不包含测试标签，我们仅为$\text{BERT}_{\text{BASE}}$ 和$\text{BERT}_{\text{LARGE}}$各提交了一次GLUE评估服务器。

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

[11]QANet在Yu等人（2018）的研究中被提出，但该系统在发表后已得到显著改进。

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

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

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
$\text{BERT}_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

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

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

4.3 SQuAD v2.0

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

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

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

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

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

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 commonsense inference (Zellers et al., 2018). Given a sentence, the task is to choose the most plausible continuation among four choices.

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

We fine-tune the model for 3 epochs with a learning rate of 2e-5 and a batch size of 16. Results are presented in Table 4. $\text{BERT}_{\text{LARGE}}$ outperforms the authors' baseline ESIM+ELMo system by +27.1% and OpenAI GPT by 8.3%.

5 Ablation Studies

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

系统	开发者 EM	开发者 F1	测试 EM	测试 F1
顶尖排行榜系统（2018年12月10日）				
Human			82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 集成模型 - QANet	-	-	84.5	90.5
已发布				
BiDAF+ELMo (单词)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
我们的				
$\text{BERT}_{\text{BASE}}$ (单一版本)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (单一版本)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$(Sgl.+TriviaQA)	84.2	91.1	85.1	91.8
$\text{BERT}_{\text{LARGE}}$(Ens.+TriviaQA)	86.2	92.2	87.4	93.2

表2：SQuAD 1.1测试结果。BERT集成模型包含7个子系统，它们采用不同的预训练检查点和微调种子。

系统	开发者 EM	开发者 F1	测试 EM	测试 F1
顶尖排行榜系统（2018年12月10日）				
Human			86.3	89.0
#1 单任务 - MIR-MRC（F-Net）	-	-	74.8	78.0
#2 单词 - nlnet	-	-	74.2	77.1
已发布				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (单一)	-	-	71.4	74.4
我们的				
$\text{BERT}_{\text{LARGE}}$ (单一版本)	78.7	81.9	80.0	83.1

表3：SQuAD 2.0测试结果。我们排除那些将BERT作为组件之一的研究条目。

在使用微调数据时，我们仅损失0.1-0.4的F1值，仍以显著优势超越所有现有系统。[12]

4.3 SQuAD v2.0

SQuAD 2.0任务在SQuAD 1.1问题定义基础上进行了扩展，允许给定段落中不存在简短答案的可能性，从而使问题更具现实意义。

我们采用简易方法扩展SQuAD v1.1的BERT模型以完成此任务。对于无答案的问题，将其答案跨度起点与终点均设为［CLS］标记。答案跨度起止位置的概率空间扩展至包含［CLS］标记的位置。预测时，我们将无答案区间的得分（$s_{\texttt{null}} = S{\cdot}C + E{\cdot}C$）与最佳非空答案区间的得分进行比较。

[12]我们使用的TriviaQA数据集由TriviaQA-Wiki段落组成，这些段落取自文档的前400个词元，且至少包含一个给定的备选答案。

系统	开发测试
ESIM+GloVe	51.9 52.7
ESIM+ELMo	59.1 59.2
OpenAI GPT	78.0
$\text{BERT}_{\text{BASE}}$	81.6
$\text{BERT}_{\text{LARGE}}$	**86.6 86.3**
Human (expert)†	85.0
人类（5条注释）†	88.0

表4：SWAG开发集与测试集准确率。 †人类表现采用SWAG论文所述的100个样本进行评估。

$\hat{s}_{i,j} = \max_{j \geq i} S{\cdot}T_i + E{\cdot}T_j$。当满足 $\hat{s}_{i,j} > s_{\texttt{null}} + \tau$ 时，我们预测非空答案，其中阈值 τ 在开发集上选取以最大化F1值。本模型未使用TriviaQA数据集，采用5e-5学习率与48批量大小进行2个 epoch 的微调。

与先前排行榜条目及顶尖发表成果（Sun et al., 2018; Wang et al., 2018b）的对比结果见表3，其中排除将BERT作为组件的系统。我们观察到相较于先前最佳系统，F1值提升了 +5.1。

4.4 SWAG

对抗性生成情境（SWAG）数据集包含11.3万组句子对补全范例，用于评估基于现实的常识推理能力（Zellers等人，2018）。任务要求根据给定句子，从四个备选项中选择最合理的后续内容。

在SWAG数据集上进行微调时，我们构建了四种输入序列，每种序列都包含给定句子（句子A）与可能的续句（句子B）的拼接。唯一引入的任务特定参数是一个向量，其与［CLS］标记表示 C 的点积表示每个选项的得分，该得分通过softmax层进行归一化处理。

我们采用学习率2e-5、批量大小16的参数对模型进行3个 epoch 的微调。结果见表4。$\text{BERT}_{\text{LARGE}}$ 的性能超越作者基线系统ESIM+ELMo +27.1%，并领先OpenAI GPT 8.3%。

5 消融研究

本节通过对BERT多个维度的消融实验，深入解析其各组件的相对重要性。

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $\text{BERT}_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

ablation studies can be found in Appendix C.

5.1 Effect of Pre-training Tasks

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

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

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

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

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

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

5.2 Effect of Model Size

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

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

It has long been known that increasing the model size will lead to continual improvements on large-scale tasks such as machine translation and language modeling, which is demonstrated by the LM perplexity of held-out training data shown in Table 6. However, we believe that this is the first work to demonstrate convincingly that scaling to extreme model sizes also leads to large improvements on very small scale tasks, provided that the model has been sufficiently pre-trained. Peters et al. (2018b) presented

任务	开发集 MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
不使用空格分隔符	83.9	84.9	86.5	92.6	87.9
LTR & 无NSP	82.1	84.3	77.5	92.1	77.8
+ 双向长短期记忆网络	82.1	84.1	75.7	91.6	84.9

表5：基于$\text{BERT}_{\text{BASE}}$ 架构的预训练任务消融实验。"无NSP"指未进行下句预测任务的训练。"LTR & 无NSP"采用类似OpenAI GPT的左至右语言模型训练方式，不包含下句预测。"+BiLSTM"在"LTR + 无NSP"模型基础上，于微调阶段添加随机初始化的双向长短期记忆网络（BiLSTM）。

切除研究详见附录C。

5.1 预训练任务的影响

我们通过评估两个预训练目标，采用与BERTv1完全相同的预训练数据、微调方案和超参数，展示了BERT深度双向性的重要性：

无NSP：采用"遮蔽语言模型"(MLM)训练的双向模型，但未包含"下句预测"(NSP)任务。

LTR &无NSP：采用标准左到右（LTR）语言模型（LM）训练的左上下文限定模型，而非多任务学习（MLM）。微调阶段同样保持左限定约束，因解除该约束会导致预训练与微调不匹配，从而降低下游任务性能。 此外，该模型预训练时未包含NSP任务。这使其可与OpenAI GPT直接对比，但采用我们更大的训练数据集、输入表示方案及微调机制。

我们首先考察NSP任务带来的影响。 表5显示，移除NSP会显著降低QNLI、MNLI和SQuAD 1.1任务的性能。随后通过"无NSP"与"LTR & 无NSP"对比评估双向表示训练的影响。LTR模型在所有任务上表现逊于MLM模型，尤其在MRPC和SQuAD任务中出现大幅性能下降。

对于SQuAD数据集，直观而言基于左到右（LTR）的模型在词元预测中表现不佳，因为词元级别的隐藏状态缺乏右侧上下文。为切实强化LTR系统，我们在模型顶部添加了随机初始化的双向LSTM（BiLSTM）。此举显著提升了SQuAD测试结果，但其结果仍远逊于预训练的双向模型。BiLSTM在GLUE任务中的表现受到损害。

我们意识到也可像ELMo那样分别训练左至右（LTR）和右至左（RTL）模型，将每个词元表示为两个模型的拼接结果。 然而：(a) 此方案成本是单一双向模型的两倍；(b) 对问答等任务缺乏直观性——右到左模型无法将答案与问题关联；(c) 其性能必然逊于深度双向模型，因后者可在每层同时利用左右上下文。

5.2 模型规模的影响

本节探讨模型规模对微调任务准确率的影响。我们训练了多个层数、隐藏单元数和注意力头数各异的BERT模型，其余超参数及训练流程均与前文所述保持一致。

表6展示了选定GLUE任务的测试结果。该表记录了5次随机重启微调后在开发集上的平均准确率。 可见更大规模模型在全部四个数据集上均带来显著精度提升，即便在仅含3600个标注训练样本的MRPC任务上亦然——该任务与预训练任务存在本质差异。更令人惊讶的是，相较于现有文献中已相当庞大的模型，我们仍能实现如此显著的性能提升。例如Vaswani等人的研究（2017）中探索的最大Transformer模型参数为(L=6, H=1024, A=16)，编码器参数达1亿，而文献中发现的最大Transformer模型参数为1亿。 (2017)研究的最大Transformer模型为(L=6, H=1024, A=16)，编码器参数达1亿；文献中发现的最大Transformer模型为(L=64, H=512, A=2)，参数达2.35亿（Al-Rfou等人，2018）。 相比之下，$\text{BERT}_{\text{BASE}}$包含1.1亿个参数，$\text{BERT}_{\text{LARGE}}$ 包含3.4亿个参数。

长期以来，人们已知模型规模的扩大将持续提升机器翻译和语言建模等大规模任务的性能，这从表6所示的保留训练数据的LM困惑度中可见一斑。然而我们认为，本研究首次有力证明：只要模型经过充分预训练，即使扩展至极端规模，也能在极小规模任务上实现显著提升。 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)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

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. $\text{BERT}_{\text{LARGE}}$ performs competitively with state-of-the-art methods. The best performing method concatenates the token representations from the top four hidden layers of the pre-trained Transformer, which is only 0.3 F1 behind fine-tuning the entire model. This demonstrates that BERT is effective for both fine-tuning and feature-based approaches.

6 Conclusion

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

将预训练双向语言模型（bi-LM）层数从两层增至四层对下游任务效果的影响存在矛盾结果。Melamud等人（2016）曾提及隐含维度从200增至600有所帮助，但进一步增至1000并未带来额外提升。 这两项先前的研究都采用了基于特征的方法——我们假设，当模型直接在下游任务上进行微调，且仅使用极少数量随机初始化的附加参数时，即使下游任务数据非常少，特定任务模型也能从更大、更具表现力的预训练表示中获益。

5.3 基于BERT的特征化方法

迄今所有BERT研究成果均采用微调方法：在预训练模型上添加简单分类层，并针对下游任务对所有参数进行联合微调。然而基于特征的方法——即从预训练模型中提取固定特征——具有特定优势。 首先，并非所有任务都能轻松通过Transformer编码器架构实现，因此需要添加特定任务的模型架构。其次，预先计算一次训练数据的高成本表示，再基于该表示运行多个低成本模型实验，可带来显著的计算效益。

本节通过将BERT应用于CoNLL-2003命名实体识别（NER）任务（Tjong Kim Sang和De Meulder, 2003）来比较两种方法。在BERT输入中，我们采用保留大小写的WordPiece模型，并包含数据提供的最大文档上下文。 遵循标准实践，我们将任务定义为标注任务，但未采用条件随机框架（CRF）。

超参数				开发集准确率		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

表6：BERT模型规模的消融实验。#L = 表示层数；#H =表示隐藏维度； #A = 表示注意力头数量。" LM (ppl)"为保留测试数据的遮蔽语言模型困惑度。

系统	开发集F1值	测试集F1值
ELMo（Peters等人，2018a）	95.7	92.2
CVT（Clark等人，2018）	-	92.6
CSE（Akbik等人，2018）	-	**93.1**
微调方法		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
基于特征的方法（$\text{BERT}_{\text{BASE}}$）		
嵌入表示	91.0	-
倒数第二个隐藏层	95.6	-
最后隐藏层	94.9	-
加权求和最后四个隐藏层	95.9	-
连接最后四个隐藏层	96.1	-
加权求和所有12层	95.5	-

表7：CoNLL-2003命名实体识别结果。超参数选取基于开发集，报告的开发集与测试集分数均采用该超参数下5次随机重启的平均值。

在输出层中，我们将首个子标记的表示作为输入，应用于命名实体识别标签集的标记级分类器。

为验证微调方法，我们采用基于特征的方法：从一个或多个层中提取激活值而不微调BERT的任何参数。这些上下文嵌入被用作随机初始化的两层768维双向长短期记忆网络（BiLSTM）的输入，该网络位于分类层之前。

结果如表7所示。$\text{BERT}_{\text{LARGE}}$在性能上与最先进方法不相上下。表现最佳的方法是将预训练Transformer模型前四层隐藏层的词元表示进行拼接，其F1值仅比对整个模型进行微调低0.3。这表明BERT在微调和基于特征的方法中均表现出色。

6 结论

基于语言模型的迁移学习近期取得的实证改进表明，丰富的无监督预训练已成为众多语言理解系统不可或缺的组成部分。尤其值得注意的是，这些成果使低资源任务也能受益于深度单向架构。 我们的主要贡献在于将这些发现进一步推广至深度双向架构，使同一预训练模型能够成功处理广泛的自然语言处理任务。

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

参考文献

Alan Akbik, Duncan Blythe, 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, DokookChoe, Noah Constant, Mandy Guo, and Llion Jones. 2018. Character-level language modeling with deeper self-attention. arXiv preprint arXiv:1808.04444.

久保田里惠、安藤和张桐。2005。《基于多任务与无标签数据学习预测结构的框架》。《机器学习研究期刊》，6(11月)：1817–1853。

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo. 2009. 第五届PASCAL文本蕴含识别挑战赛. 载于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. 基于类的自然语言n-gram模型。《计算语言学》，18(4):467–479。Daniel Cer、Mona Diab、Eneko Agirre、Inigo Lopez-Gazpio与Lucia Specia。2017。《Semeval-2017任务1：多语言与跨语言语义文本相似度聚焦评估》。收录于第11届语义评估国际研讨会（SemEval-2017）论文集，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与Tony Robinson。2013。用于衡量统计语言建模进展的十亿词基准。arXiv预印本arXiv:1312.3005。Z. Chen、H. Zhang、X. Zhang和L. Zhao。2018。Quora问题对。Christopher Clark和Matt Gardner。2018。简单而有效的多段落阅读理解。载于ACL。

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

Ronan Collobert 与 Jason Weston. 2008.《自然语言处理的统一架构：基于多任务学习的深度神经网络》。载于《第25届国际机器学习会议论文集》，第160–167页。ACM出版。

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 与 Quoc V Le. 2015. 半监督序列学习. 载于《神经信息处理系统进展》，第3079–3087页。

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 与 Chris Brockett. 2005. 自动构建句子同义句语料库. 收录于第三届同义句国际研讨会论文集（IWP2005）. William Fedus, Ian Goodfellow 与 Andrew M Dai. 2018. MaskGAN：通过填充空缺实现更优文本生成。arXiv预印本arXiv:1801.07736。Dan Hendrycks与Kevin Gimpel. 2016. 通过高斯误差线性单元连接非线性与随机正则化器。CoRR, abs/1606.08415。 Felix Hill、Kyunghyun Cho与Anna Korhonen. 2016.《从无标签数据学习句子的分布式表示》。收录于《2016年北美计算语言学协会会议论文集：人类语言技术》。计算语言学协会出版。Jeremy Howard与Sebastian Ruder. 2018. 通用语言模型微调在文本分类中的应用。收录于ACL。计算语言学协会。胡明昊、彭玉兴、黄振、邱希鹏、魏福如、周明。2018。强化记忆阅读器在机器阅读理解中的应用。发表于 IJCAI。雅辛·杰尔尼特、塞缪尔·鲍曼和大卫·桑塔格。2017年。基于话语的快速无监督句子表示学习目标。CoRR，abs/1705.00557。

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

曼达尔·乔希、崔恩硕、丹尼尔·S·韦尔德与卢克·泽特莫耶。2017。《TriviaQA：大规模远距离监督阅读理解挑战数据集》。载于ACL会议论文集。

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, pp. 3294–3302.

Quoc Le 与 Tomas Mikolov. 2014.《句子与文档的分布式表示》。载于《国际机器学习会议》，第1188–1196页。

赫克托·J·勒维斯克、欧内斯特·戴维斯与莱奥拉·莫根斯坦。2011。《温诺格拉德模式挑战》。载于AAAI春季研讨会：常识推理的逻辑形式化，第46卷，第47页。

Lajanugen Logeswaran 与 Honglak Lee. 2018. 学习句子表示的高效框架. 载于《国际学习表示会议》。

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. 词与短语的分布式表示及其组合性. 载于《神经信息处理系统进展26》，第3111–3119页. Curran Associates, Inc. Andriy Mnih与 Geoffrey E Hinton. 2009. 可扩展分层分布式语言模型。载于D. Koller、D. Schuurmans、Y. Bengio和L. Bottou编著的《神经信息处理系统进展21》，第1081–1088页。 Curran Associates, Inc. Ankur P Parikh, Oscar T¨ackstr¨om, Dipanjan Das, and Jakob Uszkoreit. 2016. 可分解注意力模型在自然语言推理中的应用. 收录于EMNLP. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove：词表示的全球向量。收录于《自然语言处理中的经验方法》（EMNLP），页码1532–1543。 Matthew Peters、Waleed Ammar、Chandra Bhagavatula与Russell Power。2017年。基于双向语言模型的半监督序列标注。收录于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, KarthikNarasimhan, 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 tree-bank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642.

傅孙、李林阳、邱希鹏、刘阳。2018。《U-net：通过无法回答的问题实现机器阅读理解》。arXiv预印本arXiv:1810.06638。威尔逊·L·泰勒。1953。《填空法：衡量可读性的新工具》。《新闻学公报》，30(4):415–433。 Erik F Tjong Kim Sang 和 Fien De Meulder. 2003. Conll-2003共享任务导论：语言无关命名实体识别. 收录于CoNLL会议论文集. Joseph Turian, Lev Ratinov, 和 Yoshua Bengio. 2010. 词表示：半监督学习的简单通用方法。载于第48届计算语言学协会年会论文集，ACL '10，第384–394页。 阿什什·瓦斯瓦尼、诺姆·沙泽尔、尼基·帕玛、雅各布·乌斯克雷特、利昂·琼斯、艾丹·N·戈麦斯、卢卡斯·凯泽、伊利亚·波洛苏金。 2017. 注意力机制即你所需。载于《神经信息处理系统进展》，第6000–6010页。作者：Pascal Vincent, Hugo Larochelle, Yoshua Bengio, Pierre-Antoine Manzagol。2008.《基于去噪自编码器的鲁棒特征提取与组合》。收录于《第25届国际机器学习会议论文集》，第1096–1103页。ACM出版。 Alex Wang、Amanpreet Singh、Julian Michael、Felix Hill、Omer Levy与Samuel Bowman。2018a。《Glue：多任务基准测试与分析平台》

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

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

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

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

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

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

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

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

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

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

We organize the appendix into three sections:

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

  We present additional ablation studies for BERT including:

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

A Additional Details for BERT

A.1 Illustration of the Pre-training Tasks

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

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

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

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

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

用于自然语言理解。载于《*2018*年*EMNLP*研讨会论文集：*BlackboxNLP*——自然语言处理神经网络的分析与解释》，第*353–355*页。

Wei Wang, Ming Yan, and Chen Wu. 2018b. Multi-granularity hierarchical attention fusion networks f or 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. 神经网络可接受性判断. arXiv预印本 arXiv:1805.12471.

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

吴永辉、Mike Schuster、陈志锋、Quoc V Le、Mohammad Norouzi、Wolfgang Macherey、Maxim Krikun、曹远、高秦、Klaus Macherey 等。2016。《谷歌神经机器翻译系统：弥合人类与机器翻译的鸿沟》。arXiv预印本 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 WeiYu, 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).

朱玉坤、Ryan Kiros、Rich Zemel、Ruslan Salakhutdinov、Raquel Urtasun、Antonio Torralba与Sanja Fidler。2015。《书籍与电影的对齐：通过观影与阅读实现故事化视觉解释》。收录于IEEE国际计算机视觉会议论文集，第19-27页。

《BERT：深度双向Transformer模型在语言理解领域的预训练》附录

我们把附录分为三个部分：

• BERT的更多实现细节详见附录A；

•实验的详细信息详见附录B；

• 更多消融研究详见附录C。我们针对BERT进行了额外消融研究，包括：– 训练步数的影响；以及 – 不同遮蔽策略的消融分析。

BERT的补充说明

A.1 预训练任务示例

我们将在下文提供预训练任务的示例。

遮蔽式语言模型与遮蔽流程假设待处理句子为"my dog is hairy"，在随机遮蔽过程中选定第4个词元（对应"hairy"），其遮蔽流程可通过以下方式进一步说明：

• 80%的情况下：用[MASK]令牌替换该词，例如：my dog is hairy →my dog is [MASK]

• 10%的概率：用随机词替换目标词，例如：my dog is hairy → my dog is apple

• 10%的情况下：保留单词原貌，例如 my dog is hairy → my dog is hairy。此举旨在使表示更倾向于实际观察到的单词。

该方法的优势在于：Transformer编码器无法预知需预测的词汇或已被随机词替换的词汇，因此被迫为每个输入令牌保留分布式上下文表示。 此外，由于随机替换仅发生在1.5%的令牌上（即15%的10%），这似乎并未损害模型的语言理解能力。在C.2节中，我们将评估此过程的影响。

相较于标准语言模型训练，遮蔽式语言模型仅对每批次15%的词元进行预测，这表明模型可能需要更多预训练步骤。

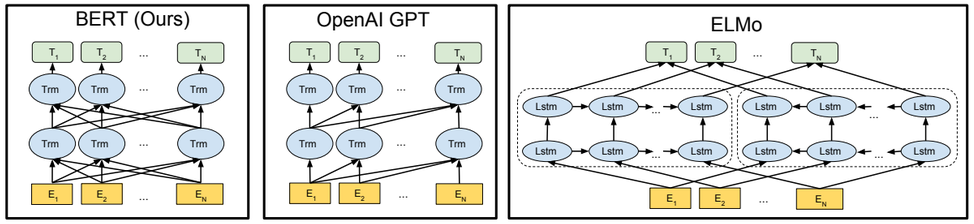


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT 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 ≤ 512 tokens. The LM masking is applied after WordPiece tokenization with a uniform masking rate of 15%, and no special consideration given to partial word pieces.

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

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

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

A.3 Fine-tuning Procedure

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

- **Batch size**: 16, 32

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

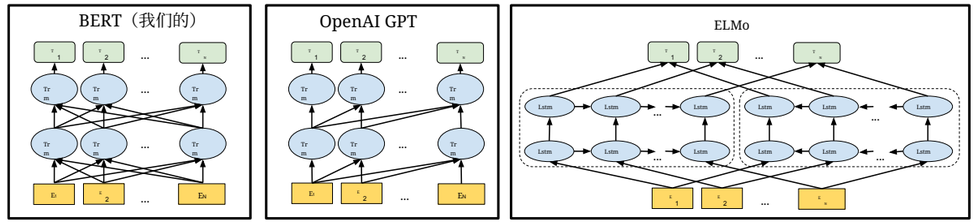


图3：预训练模型架构差异。BERT采用双向Transformer架构，OpenAI GPT采用从左到右的Transformer架构，ELMo则通过串联独立训练的左到右与右到左LSTM网络生成下游任务特征。三者中仅BERT在所有层级中同时对左右上下文进行联合条件处理。 除架构差异外，BERT与OpenAI GPT采用微调方法，而ELMo属于基于特征的方法。

在C.1节中，我们证明了MLM模型确实会收敛——在33亿词规模的语料库上训练30个epoch。虽然其收敛速度略慢于从左到右的模型（该模型预测每个词元），但MLM模型的经验性改进远大于其增加的训练成本。

下一句预测下一句预测任务可通过以下示例说明：

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

A.2 预训练流程

为生成每个训练输入序列，我们从语料库中抽取两段文本片段——尽管这些片段通常远长于单个句子（但也可能较短），我们仍将其称为"句子"。 首句接收A嵌入向量，次句接收B嵌入向量。50%情况下B为A的实际后续句子，50%情况下为随机句子——此机制用于实现"下一句预测"任务。采样时确保组合长度为 ≤ 512 个词元。 在WordPiece分词后应用语言模型遮蔽，采用15%的统一遮蔽率，且不考虑部分词片段的特殊性。

我们采用批量大小为256序列（256序列×512个令牌 = 128，000令牌/批）进行1,000,000步训练，约相当于40

采用Adam学习率0.0001， $\beta_1 = 0.9$， $\beta_2 = 0.999$，L2正则化权重衰减0.01，前10,000步学习率预热，学习率线性衰减。所有层应用0.1的dropout概率。 遵循OpenAI GPT架构，我们采用Gelu激活函数（Hendrycks and Gimpel, 2016）替代标准ReLU。训练损失函数为掩码语言模型平均似然值与下句预测平均似然值之和。

$\text{BERT}_{\text{BASE}}$ 的训练在4台Cloud TPU上以Pod配置（共16个TPU芯片）完成。[13] $\text{BERT}_{\text{LARGE}}$ 的训练在16台Cloud TPU上（共64个TPU芯片）完成。每次预训练耗时4天。

长序列的计算成本呈非线性增长，因注意力机制的复杂度与序列长度呈二次方关系。为加速预训练进程，我们在实验中采用90%训练步长使用128字符序列进行预训练，随后以512字符序列完成剩余10%训练步长，以学习位置嵌入向量。

A.3 微调流程

在微调过程中，除批量大小、学习率和训练 epoch 数外，多数模型超参数与预训练阶段保持一致。dropout 概率始终维持在 0.1。虽然最优超参数取值因任务而异，但我们发现以下参数范围在所有任务中均表现良好：

- **批量大小：16, 32**

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

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

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

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

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

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

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

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

A.5 Illustrations of Fine-tuning on Different Tasks

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

B Detailed Experimental Setup

B.1 Detailed Descriptions for the GLUE Benchmark Experiments.

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

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

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

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

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

我们还发现，与小规模数据集相比，大规模数据集（例如10万个标注训练样本）对超参数选择的敏感度显著降低。由于微调过程通常非常快速，因此可直接对上述参数进行穷举搜索，并选择在开发集上表现最佳的模型。

A.4 BERT、ELMo与OpenAI GPT的比较

本文研究了近期流行的表征学习模型（包括ELMo、OpenAI GPT和BERT）之间的差异。模型架构的对比关系如图3所示。需注意除架构差异外，BERT和OpenAI GPT属于微调方法，而ELMo则是基于特征的方法。

与BERT最接近的现有预训练方法是OpenAI的GPT，其通过大型文本语料库训练从左到右的Transformer语言模型。事实上，BERT的许多设计决策都是刻意使其尽可能接近GPT，以便两者能进行最小差异比较。本研究的核心论点是：第3.1节所述的双向性与双预训练任务共同促成了绝大多数实证改进。但我们注意到BERT与GPT在训练方式上还存在若干差异：

- GPT在BooksCorpus（8亿词）上训练；BERT在BooksCorpus（8亿词）和维基百科（25亿词）上训练。
- GPT在微调阶段引入句子分隔符（[SEP]）和分类标记（[CLS]）；而BERT在预训练阶段即学习[SEP]，[CLS]及句子A/B嵌入向量。
- GPT采用32,000词批量大小训练100万步；BERT采用128,000词批量大小训练100万步。
- GPT在所有微调实验中均采用5e-5的统一学习率；BERT则选择针对特定任务的微调学习率，该学习率在开发集上表现最佳。

为隔离这些差异的影响，我们在第5.1节进行了消融实验，结果表明绝大多数改进实际上源自两个预训练任务及其赋予的双向性。

A.5 不同任务上的微调示例

图4展示了BERT在不同任务上的微调示意图。我们的任务专用模型通过在BERT基础上添加一层输出层构建而成，因此仅需从零学习极少量参数。其中(a)和(b)属于序列级任务，(c)和(d)属于令牌级任务。图中 E 表示输入嵌入，T_i表示令牌 i的上下文表示，[CLS] 是分类输出的特殊符号，[SEP] 用于分隔非连续令牌序列。

B 详细实验设置

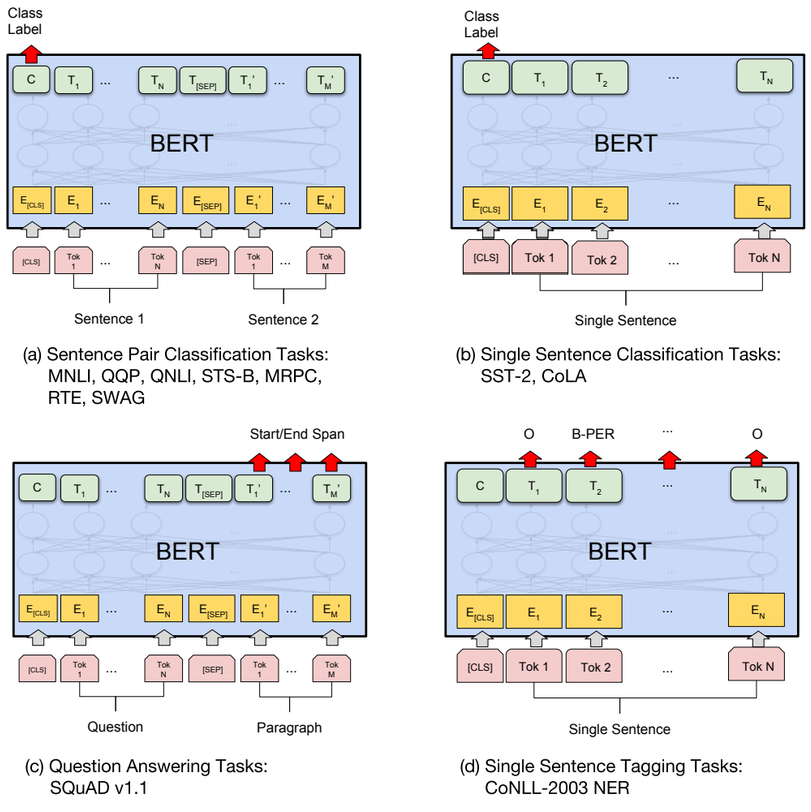
B.1 GLUE基准实验的详细说明。

表1中的GLUE测试结果源自 https://gluebenchmark.com/ leaderboard及 https://blog.openai.com/language-unsupervised。GLUE基准包含以下数据集，其描述最初由Wang et al. (2018a)总结：

MNLI（多体裁自然语言推理）是一项大规模众包蕴含分类任务（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否构成蕴含、矛盾或中性关系。

QQP（Quora问题对）是一种二分类任务，其目标是判断Quora平台上两个问题的语义等价性（Chen et al., 2018）。

QNLI（问答自然语言推理）是斯坦福问答数据集（Rajpurkar et al., 2016）的二分类任务版本（Wang et al., 2018a）。正例为包含正确答案的(问题,句子)对，负例则为同一段落中不包含答案的(问题,句子)对。



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

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

(c) Question Answering Tasks: SQuAD v1.1

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

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

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

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

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

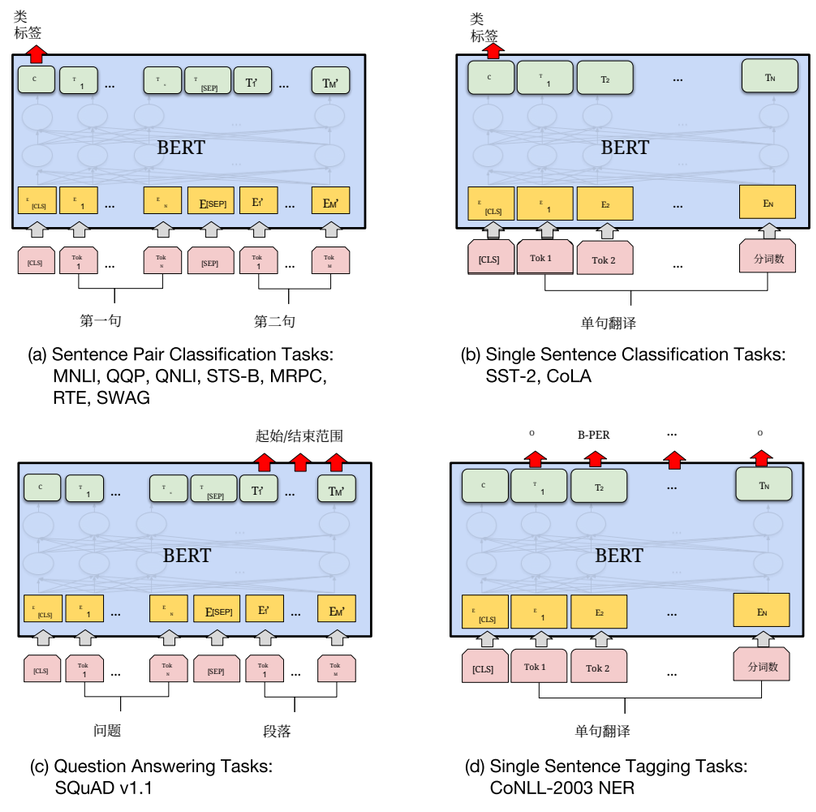
MRPC Microsoft Research Paraphrase Corpus consists of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent (Dolan and Brockett, 2005).

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

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

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

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



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

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

(c) Question Answering Tasks: SQuAD v1.1

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

图4：BERT在不同任务上进行微调的示意图。

SST-2（斯坦福情感树库）是一项二元单句分类任务，由电影评论中提取的句子构成，并附有人类标注的情感标签（Socher等人，2013）。

CoLA（语言学可接受性语料库）是一个二元单句分类任务，其目标是预测英语句子是否在语言学上"可接受"（Warstadt等人，2018）。

STS-B语义文本相似性基准数据集（Cer et al., 2017）由新闻标题及其他来源抽取的句子对组成。这些句子对被标注了1至5分的相似度评分，用于衡量两句在语义层面的相似程度。

MRPC（微软研究院同义句语料库）由在线新闻源自动提取的句子对组成，并附有人工标注。

用于判断句对是否具有语义等价性（Dolan and Brockett, 2005）。

RTE（文本蕴含识别）是一种二元蕴含任务，与MNLI类似，但训练数据量少得多（Bentivogli等人，2009）。[14]

WNLI Winograd NLI是一个小型自然语言推理数据集（Levesque等人，2011）。GLUE官网指出该数据集存在构建问题，[15] 且所有提交至GLUE的训练系统在预测多数类时均低于65.1%的基准准确率。为确保对OpenAI GPT的公平性，我们排除该数据集。在GLUE提交中，我们始终预测多数类。

[14]需注意本文仅报告单任务微调结果。多任务微调方法有望进一步提升性能。例如，我们观察到通过MNLI的多任务训练在RTE任务上获得了显著提升。[15]

https://gluebenchmark.com/faq

jority class.

C Additional Ablation Studies

C.1 Effect of Number of Training Steps

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

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

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

C.2 Ablation for Different Masking Procedures

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

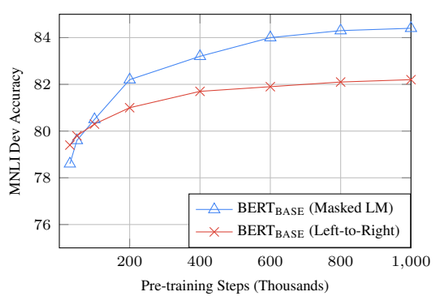


Figure 5: Ablation over number of training steps. This shows the MNLI accuracy after fine-tuning, starting from model parameters that have been pre-trained for k steps. The x-axis is the value of k.

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

Masking Rates			Dev Set Results		
			MNLI	NER	
MASK	SAME	RND	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.

多数类。

C 补充消融研究

C.1 训练步数的影响

图5展示了从预训练 k 步的检查点进行微调后，MNLI开发集的准确率。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规模的预训练（128,000 词/批次 * 1,000,000 步）才能实现高精度的微调效果?
   答案：是的，在MNLI任务上，$\text{BERT}_{\text{BASE}}$ 模型经过100万步训练时，相较于50万步训练，准确率提升近1.0%。

2. 问题：由于每次批处理中仅预测15%的词而非全部词汇，MLM预训练的收敛速度是否慢于LTR预训练？答案：MLM模型收敛速度确实略慢于LTR模型。然而就绝对准确率而言，MLM模型几乎从一开始就表现优于LTR模型。

C.2 不同遮蔽策略的消融实验

在第3.1节中，我们提到BERT采用混合策略对目标词进行遮蔽，以实现基于遮蔽语言模型（MLM）目标的预训练。以下是评估不同遮蔽策略效果的消融研究。

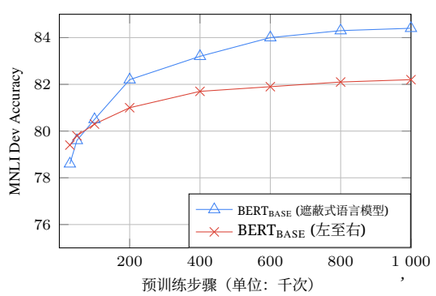


图5：训练步数消融实验。展示从预训练k 步的模型参数开始，经过微调后的MNLI准确率。x轴表示 k的数值。

需注意遮蔽策略旨在减少预训练与微调阶段的偏差，因 [MASK] 符号在微调阶段从未出现。 我们报告了MNLI和NER的开发集结果。对于NER任务，我们同时报告了微调方法和基于特征的方法，因为我们预期基于特征的方法会因模型无法调整表示而放大这种不匹配现象。

遮蔽率			开发集结果		
			MNLI	命名实体识别（NER）	
MASK	SAME	RND	微调	微调	基于特征的
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：不同遮蔽策略的消融实验。

结果如表8所示。表中MASK表示在MLM训练中将目标词替换为 [MASK] 符号；SAME表示保留目标词原样；RND表示将目标词替换为随机词。

表格左侧的数值代表多任务学习预训练中采用的具体策略概率（BERT采用80%、10%、10%）。论文右侧展示开发集测试结果。对于基于特征的方法，我们将BERT最后4层串联作为特征输入，该方法在第5.3节中被证实为最优方案。

从表格可见，微调对不同遮蔽策略表现出惊人的鲁棒性。但正如预期，仅采用MASK策略时，基于特征的命名实体识别（NER）方法存在问题。值得注意的是，仅使用RND策略的表现也远逊于我们的策略。