ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS
ALBERT：自监督学习语言表示的轻量级 BERT

摘要

当预训练语言模型时，增加模型大小通常会提升模型处理下游任务的性能。然而，由于 GPU/TPU 内存大小和训练时间的限制，进一步增加模型规模变得困难。为了解决这些问题，本文提出了两种参数减少技术，以降低内存消耗并提高 BERT 的训练速度。全面的实验证据表明，本文提出的方法相对于原始 BERT [4] 具有更好的表现。本文还采用了一个自监督损失，侧重于建模句间的一致性，并展示它在处理多句输入的下游任务时能够持续提供帮助。因此，本文的最佳模型在 GLUE、RACE 和 SQuAD 基准上取得了最好的结果，同时相比 BERT-large 拥有更少的参数。

关键词： ALBERT；BERT

1 引言

全网络预训练 [4,8] 在语言表示学习领域取得了一系列突破。许多自然语言处理任务，包括那些训练数据有限的任务，都从这些预训练模型中获益匪浅。其中一个最引人注目的迹象是中国中学和高中英语考试的阅读理解任务——RACE 测试 [10] 在机器准确率方面的发展：该任务最初报告的机器准确率为 44.1%，而最新发布的结果报告了 83.2% 的性能 [11]；本文在此呈现的工作将其推到了更高的水平，达到了 89.4%。

这些进展的证据表明，大型网络对于实现最高性能至关重要 [13]。将大型模型进行预训练并将其提炼为较小的模型！！[16] 已经成为实际应用的常见做法。鉴于模型大小的重要性，本文提出了一个问题：拥有更大的模型是否就等于拥有更好的自然语言处理模型?

回答这个问题的一个障碍是现有硬件的内存限制。考虑到当前最先进的模型通常具有数亿甚至数十亿的参数，继续扩大模型规模时很容易受到这些限制的影响。在分布式训练中，由于通信开销与模型中的参数数量成正比，训练速度也可能受到显著阻碍。

已有的解决方案包括模型并行化 [15] 和巧妙的内存管理 [5]。这些解决方案解决了内存限制的问题，但并未解决通信开销的问题。在本文中，作者通过设计一个比传统 BERT 架构具有显著较少参数的 ALBERT（A Lite BERT）架构来解决所有上述问题。

ALBERT 集成了两种参数减少技术，消除了在扩展预训练模型方面的主要障碍。第一种是分解嵌入参数。通过将大词汇嵌入矩阵分解为两个小矩阵，作者将隐藏层的大小与词汇嵌入的大小分开。这种分离能够增加隐藏层大小，而不显著增加词汇嵌入的参数大小。第二种技术是跨层参数共享。这种技术防止参数随网络深度的增加而增加。这两种技术显著减少了 BERT 的参数数量，并且不会严重影响性能，从而提高了参数效率。与 BERT-large 类似的 ALBERT 配置的参数数量减少了 18 倍，并且训练速度提高了约 1.7 倍。这些参数减少技术还充当一种正则化形式，稳定了训练并有助于泛化。

为了进一步提高 ALBERT 的性能，本文还引入了一个自监督损失，用于句子顺序预测 (SOP)。SOP 主要关注跨句子的一致性，并旨在解决原始 BERT 中提出的下一句子预测 (NSP) 损失的无效性。由于这些设计，本文作者能够扩展更大的 ALBERT 配置，仍然比 BERT-large 的参数更少，但性能显著更好。本文在著名的 GLUE、SQuAD 和 RACE 自然语言理解基准上取得了当时 (2020 年) 的最先进结果。具体而言，RACE 准确率提升到 89.4%，GLUE 基准提升到 89.4，以及 SQuAD 2.0 的 F1 得分提升到 92.2。

2 相关工作

2.1 扩展语言模型的规模

近几年语言模型的研究表明更大的模型规模可以提高性能 [3, 4, 12]。例如， [4] 表明，在三个自然语言理解任务中，使用更大的隐藏大小、更多的隐藏层和更多的注意头总是能够取得更好的性能。然而，由于模型大小和计算成本的限制，他们在隐藏大小为 1024 时停止。

考虑到当前最先进的模型通常有数亿甚至数十亿个参数，我们很容易触及内存限制。为了解决这个问题， [2] 提出了一种称为梯度检查点的方法，以减少内存要求，代价是需要额外的前向传播。 [5] 提出了一种方法，从下一层重新构建每一层的激活，以便它们无需存储中间激活。这两种方法通过牺牲速度来减少内存消耗。 [14] 提出使用模型并行化来训练一个巨大的模型。相比之下，本文的参数减少技术减少了内存消耗并提高了训练速度。

2.2 跨层参数共享

在 Transformer 架构中，先前已经探索过跨层参数共享的概念 [17]，但这之前的工作主要集中在标准编码器-解码器任务的训练，而不是预训练/微调的设置。与 Google 团队的观察不同， [8] 表明，具有跨层参数共享（Universal Transformer）的网络在语言建模和主谓一致性方面表现比标准 Transformer 更好。最近， [1] 提出了一个深度平衡模型（DQE）用于 Transformer 网络，并展示 DQE 可以达到一个平衡点，其中某一层的输入嵌入和输出嵌入保持不变。 [7] 将一个具有参数共享的 Transformer 与标准 Transformer 结合使用，进一步增加了标准 Transformer 的参数数量。

2.3 句子排序目标函数

ALBERT 采用了一种基于预测两个相邻文本片段顺序的预训练损失函数。多位研究人员已经尝试了与篇章连贯性相关的预训练目标。篇章中的连贯性和凝聚性已经得到广泛研究，已经确定了许多与相邻文本片段相关的现象 [6]。在实践中发现，大多数有效的目标都相当简单。

本文的损失与 [9] 的句子排序目标最相似，其中学习句子嵌入以确定两个相邻句子的顺序。然而，与上述工作不同的是，本文的损失是在文本片段而不是句子上定义的。BERT 采用了一种基于预测一对中的第二个片段是否与另一篇文档中的片段交换的损失。作者在实验中与这个损失进行了比较，并发现句子排序是一项更具挑战性的预训练任务，并对某些下游任务更有用。与本文的工作同时进行的，[18] 也尝试预测两个相邻文本片段的顺序，但他们将其与原始的下一句预测结合在一个三分类任务中，而不是在实证中比较这两种方法。

3 本文方法

这篇文章中主要提出了对 BERT 的三点改进，缩小了整体的参数量，加快了训练速度，提升了模型效果。

嵌入矩阵分解。在 BERT、XLNet、RoBERTa 中，词表的 embedding size (E) 和模型的 hidden size(H) 都是相等的，ALBERT 选择将 E 和 H 分开，这样做有两方面优点：1）从建模角度来讲，wordpiece 向量应该是不依赖于当前内容的，而 transformer 所学习到的表示应该是依赖内容的。所以把 E 和 H 分开可以更高效地利用参数，因为理论上存储了上下文信息的 H 要远大于 E。2）从实践角度来讲，NLP 任务中的词表大小本来就很大，如果 E=H 的话，embedding matrix 参数量就容易很大 $V \times E$，而且 embedding 在实际的训练中更新地也比较稀疏。因此，ALBERT 使用嵌入参数的分解，将其分解为两个较小的矩阵。与直接将一向量投影到大小为 H 的隐藏空间不同，ALBERT 首先将其投影到大小为 E 的较低维度嵌入空间，然后再将其投影到隐藏空间。通过使用这种分解，ALBERT 将嵌入参数从 $O(V \times H)$ 减少到 $O(V \times E + E \times H)$。

参数共享。ALBERT 提出了跨层参数共享作为减少参数数量的另一种方式。有多种参数共享的方式，例如，仅在层之间共享前馈网络（Feed-Forward Network）参数，或仅在层之间共享注意力参数。ALBERT 默认共享不同层的全部参数。

句子顺序判断。BERT 引入了一种二元分类损失，称为 Next Sentence Prediction(NSP)。NSP 是一个二元分类损失，用于预测两个片段是否在原始文本中连续出现，具体如下：正例是通过从训练语料库中取连续的片段创建的；负例是通过将来自不同文档的片段配对创建的；正例和负例是以相等的概率进行抽样的。

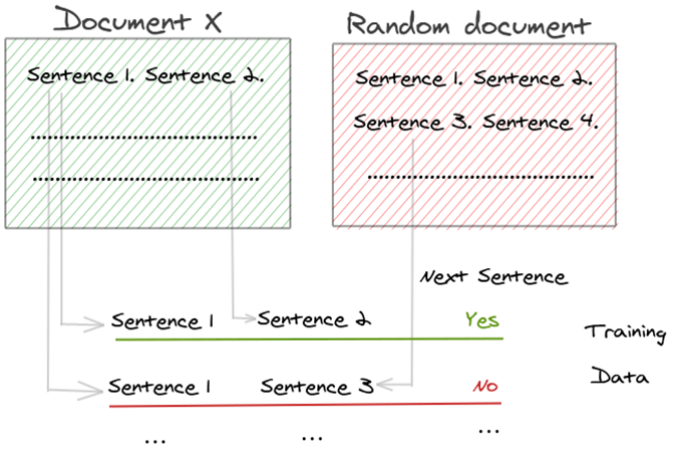

图 1. NSP 使用的正例与反例

4 复现细节

4.1 与已有开源代码对比

本文使用 HuggingFace 提供的基于 Transformers 工具的 Albert V2 版本的预训练模型。HuggingFace 的 Transformers 支持 PyTorch、TensorFlow 和 JAX 等不同框架，提供了在模型生命周期的每个阶段使用不同框架的灵活性。使用 HuggingFace 的 Albert 模型有利于我们后期根据工作需要切换不同的框架。

Albert 是一个在大型英语语料库上以自监督方式预训练的模型。这意味着它仅在原始文本上进行了预训练，没有以任何方式由人员进行标注。Albert V2 是基础模型的第二个版本。V2 版本的模型的模型架构与原论文中的完全相同，区别主要在于不同的丢失率、额外的训练数据和更长的训练时间。根据作者提供的表 1显示，对于 ALBERT-base、ALBERT-large 和 ALBERT-xlarge，V2 要比 V1 好得多。

	Average	SQuAD1.0	SQuAD2.0	MNLI	SST-2	RACE
V2						
ALBERT-base	82.3	90.2/83.2	82.1/79.3	84.6	92.9	66.8
ALBERT-large	85.7	91.8/85.2	84.9/81.8	86.5	94.9	75.2
ALBERT-xlarge	87.9	92.9/86.4	87.9/84.1	87.9	95.4	80.7
ALBERT-xxlarge	90.9	94.6/89.1	89.8/86.9	90.6	96.8	86.8
V1						
ALBERT-base	80.1	89.3/82.3	80.0/77.1	81.6	90.3	64.0
ALBERT-large	82.4	90.6/83.9	82.3/79.4	83.5	91.7	68.5
ALBERT-xlarge	85.5	92.5/86.1	86.1/83.1	86.4	92.4	74.8
ALBERT-xxlarge	91.0	94.8/89.3	90.2/87.4	90.8	96.9	86.5

表 1. V1 与 V2 的测试对比

我在 Albert-base V2 预训练模型的基础上训练了一个判断电影评论正面还是负面的情感分类模型。情感分类模型的目标是自动识别文本中的情感倾向，通常包括正面、负面或中性情感。这类模型主要应用于文本数据，例如用户评论、社交媒体帖子、电影评论等，以帮助人们了解大众对某个话题或产品的感受。在情感分类中，使用的数据集包括许多已标记的文本，其中每个文本都附有相应的情感标签。这些标签可以是二元的（正面或负面）或多元的（正面、负面、中性等）。训练中我使用了 IMDB 的数据集，IMDB 是大型电影评论数据集，这是一个用于二元情感分类的数据集。IMDB 的情感分类数据库包含了大量的电影评论，这些评论已经被标记为正面或负面情感。选择 IMDB 是因为它提供了丰富的文本数据，并且标签较为准确。本文代码生成的模型在 IMDB 测试集的正确率可以达到 93.12%。

本文在 Albert-base V2 基础上利用 IMDb 的数据集训练一个情感分类模型的步骤如下：首先，下载 IMDb 情感分类数据集，包含训练集和测试集。进行数据预处理，包括文本分词、去除停用词。加载预训练的 Albert 模型，在原模型的基础上构建情感分类模块，添加适当的分类器。定义损失函数和优化器，本报告使用的是交叉熵损失和 Adam 优化器。利用训练集

对模型进行训练，调整参数以最小化损失。使用测试集评估模型性能，采用正确率作为主要指标。本次训练总轮数设置为 3，分类器的权重衰减为 0.01。

4.2 实验环境搭建

首先，根据表 2安装 Python 以及 Transformers 等相关工具，推荐在 conda 虚拟环境中安装。接下来，利用 python 运行项目代码文件夹中的训练脚本 albert_base_imdb_train.py，其中包含 ALBERT 模型的初始化、数据加载与预处理、训练循环和保存模型的代码。用户可以根据自己的硬件调整批量大小，训练轮数等参数。最后，使用测试脚本 test.py 测试情感分类模型的效果。这个测试脚本会测试模型在 imdb 测试集 5000 条数据上的正确率。

操作系统	Ubuntu 20.04.5
CPU	Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
GPU	NVIDIA GeForce RTX 3090
Python 版本	3.80
CUDA 版本	12.2
PyTorch 版本	2.1.2

表 2. 训练环境

4.3 创新点

我在 Albert-base V2 预训练模型的基础上训练了一个判断电影评论正面还是负面的情感分类模型, 以帮助人们了解大众对某部电影的感受。训练中我使用了 IMDB 的数据集，IMDB 是大型的二元情感分类的数据集，本文代码生成的模型在 IMDB 测试集的正确率可以达到 93.12%。

5 实验结果分析

本模型在 IMDB 测试集的正确率为 93.12%，由于硬件限制，本文使用的是原作者提供的 base 模型，模型参数规模为 12M。未来使用更大规模的 Albert 预训练模型预计能取得更好的效果。

6 总结与展望

ALBERT 虽然参数比 BERT 少，但取得了显著更好的结果。未来，一个重要的方法是通过稀疏注意力和块注意力等方法加快 ALBERT 的训练和推理速度。另一条研究线索包括硬例挖掘和更高效的语言建模训练，这可能提供模型额外的表示能力。

参考文献

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

[2] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

[3] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28, 2015.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. The reversible residual network: Backpropagation without storing activations. *Advances in neural information processing systems*, 30, 2017.

[6] Barbara J Grosz, Aravind K Joshi, and Scott Weinstein. Centering: A framework for modelling the local coherence of discourse. 1995.

[7] Jie Hao, Xing Wang, Baosong Yang, Longyue Wang, Jinfeng Zhang, and Zhaopeng Tu. Modeling recurrence for transformer. *arXiv preprint arXiv:1904.03092*, 2019.

[8] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[9] Yacine Jernite, Samuel R Bowman, and David Sontag. Discourse-based objectives for fast unsupervised sentence representation learning. *arXiv preprint arXiv:1705.00557*, 2017.

[10] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

[11] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[12] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[13] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[15] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[16] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*, 2019.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[18] Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Jiangnan Xia, Liwei Peng, and Luo Si. Structbert: Incorporating language structures into pre-training for deep language understanding. *arXiv preprint arXiv:1908.04577*, 2019.