微型 BERT 模型的实现及模型泛化下游任务

王 浩 程佳诺

摘 要 在本项目中，我们基于 Standford 2023 Winter CS224 Default Final Projcet 给出的项目框架实现了一个高效的微型 BERT 模型，并探索其在自然语言处理的下游任务中的泛化能力。minBERT 模型基于 Transformer 架构，采用了 Multi-Head Attention 机制，通过对大规模语料库进行预训练，能够捕捉到文本数据中的深层次语义关系。在模型实现阶段，我们使用 adam 优化器对 minBERT 进行了精细调整，以适应具体任务的需求。随后，我们将该模型应用于情感分析任务，采用 SST 数据集，展现了其在该领域的有效性。进一步地，为了验证模型的泛化能力，我们将 minBERT 应用于释义检测和语义文本相似度任务。在这些下游任务的泛化过程中，我们采用了 smart 正则化技术和多任务轮询策略，通过轮询方式对多个任务进行细微调整。此外，为了提升模型在相似任务中的性能，我们设计了一种新的相似任务相关层，通过这一层的结构，可以有效地整合和利用不同任务间的相关性，从而提升模型在各任务中的表现。实验结果表明，我们的 minBERT 模型不仅在特定任务上达到了良好的性能，而且其泛化能力显著，较为优秀的完成了以上自然语言处理领域中的经典任务。

关键词 自然语言处理；微型 BERT 模型；深度学习；模型泛化；情感分析；释义检测；语义文本相似度
中图法分类号 TP391 DOI 号 10.11897/SP.J.1016.01.2023.00001

Implementation of Miniature BERT Model and Its Generalization in Downstream Tasks

Wang Hao Cheng Jianuo

Abstract In this project, we have implemented an efficient miniature BERT model, referred to as minBERT, and explored its generalization capabilities in downstream natural language processing tasks. The minBERT model is based on the Transformer architecture, incorporating the Multi-Head Attention mechanism, and pre-trained on a large-scale corpus to capture deep semantic relationships within text data. During the implementation phase, the adam optimizer was used to fine-tune minBERT to meet the specific requirements of the tasks. Subsequently, we applied the model to the task of sentiment analysis using the Stanford Sentiment Treebank (SST) dataset, demonstrating its effectiveness in this domain. To further validate the model's generalization ability, we extended minBERT to paraphrase detection and semantic textual similarity tasks. In the process of generalizing to these downstream tasks, we employed smart regularization techniques and a multi-task round-robin strategy, making fine adjustments to multiple tasks in a round-robin manner. Additionally, to enhance the model's performance on similar tasks, we designed a new layer that relates similar tasks, enabling effective integration and utilization of inter-task relationships, thereby improving model performance across various tasks. The experimental results indicate that our minBERT model not only performs well on specific tasks but also exhibits significant generalization capabilities, proficiently completing the above classic tasks in the field of natural language processing.

Key words Natural Language Processing; MiniBERT; Deep Learning; Model Generalization; Sentiment Analysis; Paraphrase Detection; Semantic Textual Similarity

1 引言 Introduction

在过去的几年里，变换器模型（Transformers）及其衍生的大型预训练语言模型如 BERT，已经彻底改变了自然语言处理（NLP）的领域。这些模型能够理解和生成人类语言，解答问题，并在多种需要抽象概念理解的任务中取得了前所未有的成就。尤其是在多任务学习（MTL）的背景下，模型通过整合跨多个任务的训练样本，学习共享知识，以此在更为复杂的任务中获益，这一策略已被证明能够显著提高模型的泛化能力。

然而，尽管 BERT 及其变体模型在多个 NLP 任务中展现了卓越的性能，但其在部署和泛化到新任务时仍面临着不小的挑战。这些挑战包括了模型的庞大规模导致的计算资源消耗问题，以及模型在学习新任务时可能会遗忘先前任务的知识。在本项目

收稿日期: 2023-12-17; 修改日期: 2023-12-17 王浩（**通信作者**）, E-mail: 237466265@qq.com. 程佳诺，E-mail: chengjianuo910@gmail.com.

中，我们实现了 minBERT，这是一个高效、精简的 BERT 模型，旨在克服这些挑战，并在自然语言处理的下游任务中测试其泛化能力。

本项目中，我们使用斯坦福大学 CS 224n[1]的 default final project[2]，我们首先介绍了 minBERT 模型的设计与实现过程，该模型基于精简的 Transformer 架构，并采用了 Multi-Head Attention 机制。通过在大规模语料库上的预训练，minBERT 能够捕捉文本数据中的深层语义信息。在模型优化过程中，我们选择了 adam 优化器，并对 minBERT 进行了细致的调整，以适应不同的下游任务。

具体而言，我们将 minBERT 应用于三个核心任务：情感分析、释义检测以及语义文本相似度（STS）分析。通过在情感分析任务上的应用，我们验证了 minBERT 在特定任务中的有效性。进一步地，我们将 minBERT 泛化到了释义检测和 STS 任务。为了进一步提升 minBERT 的泛化能力，我们实施了一系列优化策略。其中，smart 正则化技术帮助模型在训练过程中保持稳定，而多任务轮询策略则确保了模型在各个任务上的均衡学习。此外，我们还引入了针对相似任务的关系层结构优化，这一创新设计有助于模型在处理相关任务时实现信息的有效迁移和整合。

通过在一系列下游任务上的实验验证，我们的研究不仅证明了 minBERT 在特定任务上的有效性，更重要的是，展示了其在未知任务上的出色泛化能力。这些成果为深度学习模型在自然语言处理领域的进一步应用和优化提供了有力的证据和新的技术途径。

最终，我们的实验结果证明了 minBERT 模型在特定下游任务上不仅展现了良好的性能，其优化策略也大大增强了模型的泛化能力。通过本研究，我们为深度学习模型在自然语言处理领域的应用和优化提供了新的视角和技术路径。在论文的后续部分，我们将详细介绍相关工作、问题定义、方法论、实验结果，以及对模型未来发展方向的展望与讨论。

2 相关工作 Related work

自然语言处理领域近年来的显著进步，在很大程度上得益于 BERT（Bidirectional Encoder Representations from Transformers）模型的引入。BERT 模型利用多头注意力机制，有效地捕获了词语间复杂的依赖关系，通过预训练在大型语料库上学习丰富的语言表征，从而在多种下游任务中取得了卓越的性能。然而，BERT 在部署和微调过程中遇到的挑战也相应地引起了研究者们的关注，尤其是如何保持预训练权重的价值，同时避免在特定任务上过度拟合。

在优化策略方面，Adam 优化器因其在处理稀疏梯度和不同参数的不同学习速率上的优势而广泛应用于深度学习模型的训练。基于一阶梯度信息，Adam 通过计算梯度的一阶矩和二阶矩的指数移动平均来调整每个参数的学习速率，这使得其在实践中表现出良好的收敛性能。

在解决模型过度拟合的问题上，SMART 正则化技术提出了一种通过对抗性损失和信任区域来控制参数更新的方法。Jiang[1] 等人通过这种方法显著提升了模型在基准测试中的分数，并且减少了对超参数调整的依赖，这对资源有限的研究情境尤其有价值。

除此之外，针对多任务学习的情境，轮询策略（Round-Robin strategy）被提出来确保在微调过程中不同任务之间获得平衡的学习机会。这一策略有助于模型在多个任务上的泛化能力，避免了模型在某个任务上过度优化从而损害其他任务的性能。

最后，为了捕捉在处理相似任务时的共享知识和模式，研究者们设计了相似工作相关层（Rich Relational Layer for similar tasks）。通过在预训练权重之上添加新的网络层来捕获和组合相似任务之间的相关性，这种结构的引入，为提高模型在相关任务中的预测性能提供了一种有效手段。

综上所述，我们的研究工作站在前人的基础上，不仅仅是在模型架构上进行了创新，也在模型优化和多任务学习策略上做出了新的尝试，旨在进一步提升模型的泛化能力和实际应用的有效性。

3 问题定义 Problem definition

3.1 情感分析问题

理解文本的基本任务之一是确定其极性（即，文中表达的观点是积极的、消极的还是中立的）。情感分析可以用来确定个人对特定产品、政治家或新闻报道的感觉。

作为一个具体的数据集示例，斯坦福情感树库

[1] https://web.stanford.edu/class/cs224n/

[2] https://web.stanford.edu/class/cs224n/project/default-final-project-bert-handout.pdf

包含从电影评论中提取的 11,855 个单句。该数据集使用斯坦福解析器解析，并包括了 215,154 个独特的短语，每个短语都由 3 个人工评判者进行注释。每个短语被标记为消极的、有些消极的、中立的、有些积极的或积极的。

- 电影评论：Light, silly, photographed with colour and depth, and rather a good time.
- 情感：4（积极）
- 电影评论：Opening with some contrived banter, cliches and some loose ends, the screenplay only comes into its own in the second half.
- 情感：2（中立）
- 电影评论：... a sour little movie at its core; an exploration of the emptiness that underlay the relentless gaiety of the 1920’s ... The film’s ending has a “What was it all for?”
- 情感：0（消极）

3.2 释义检测问题

释义检测是在大量文段中找到文本释义的任务。释义是“对某人所写或所说内容的改述”；因此释义检测实际上是为了确定某些单词或短语是否传达了相同的语义含义。从研究的角度来看，释义检测是一个有趣的任务，因为它提供了衡量系统如何“理解”语义意义细微差别的尺度。

作为一个具体的数据集示例，网站 Quora 经常收到其他问题的重复问题。为了更好地引导用户并防止不必要的工作，Quora 发布了一个标注了不同问题是否为彼此释义的数据集。

- 问题对：（1）”What is the step by step guide to invest in share market in india?”, (2) ”What is the step by step guide to invest in share market?”
- 是否释义：否
- 问题对：(1) ”I am a Capricorn Sun Cap moon and cap rising...what does that say about me?”, (2) ”I’m a triple Capricorn (Sun, Moon and ascendant in Capricorn) What does this say about me?
- 是否释义：是

3.3 语义文本相似度问题

语义文本相似性（STS）任务旨在捕捉某些文本比其他文本更相似的概念。STS 任务寻求测量文本的语义等价度。STS 与释义不同，在于它不是简单的“是”或“否”的决定；相反，STS 允许相似度的程度。例如，在从 5（完全相同含义）到 0（完全无关）的范围内，以下句子之间具有如下关系：

- （5）两个句子完全等价，因为它们表达了相同的事情：
- The bird is bathing in the sink.
- Birdie is washing itself in the water basin.
- （4）两个句子大体上是等价的，但一些不重要的细节不同：
- In May 2010, the troops attempted to invade Kabul.
- The US army invaded Kabul on May 7th last year, 2010.
- （3）两个句子大致相似，但一些重要信息不同：
- John said he is considered a witness but not a suspect.
- “He is not a suspect anymore.”
- （2）两个句子不等价，但有一些共同细节：
- They flew out of the nest in groups.
- They flew into the nest together.
- （1）两个句子不等价，但主题相同：
- The woman is playing the violin.
- The young lady enjoys listening to the guitar.
- （0）两个句子主题不同：
- John went horseback riding at dawn with a whole group of friends.
- Sunrise at dawn is a magnificent view to take in if you wake up early enough for it.

通常在 STS 任务上训练的模型（常使用基于它们词嵌入的余弦相似度指标）需要能够适当地训练模型以预测上述每个句子对的语义内容。

3.4 具体挑战

本项目旨在解决的核心问题是如何设计一个既能保持 BERT 模型深层语义理解能力，同时又能有效减少模型规模以便于部署，且在多任务环境中能够保持或提升泛化能力的 NLP 模型。在此基础上，我们面临以下具体挑战：

1. 模型规模与性能的平衡：BERT 模型虽然性能强大，但其庞大的模型规模使得部署和运算成本高昂。我们需要探索如何通过精简模型结构来减少资源消耗，同时不过度牺牲模型的性能。

2. 泛化能力：传统的微调方法在单一任务上可能表现出色，但在多任务学习场景中，模型的泛化能力往往会下降。如何设计微调策略以提高模型在不同任务上的泛化能力，是我们需要解决的问题。

3. 过拟合问题：在特定下游任务的微调过程中，模型容易过拟合到特定数据集上，失去对新数据的泛化能力。因此，我们要探讨有效的正则化方法以防止过拟合。

4. 训练策略的有效性：如何确保在训练过程中，不同任务能够公平地影响模型，避免某些任务支配学习过程，导致模型偏向于某个特定任务。

综上所述，本研究的目标是实现一个微型 BERT（minBERT）模型，并通过一系列训练策略，如 smart 正则化、多任务轮询策略，以及相似任务相关层的设计，来解决上述挑战。我们期望 minBERT 能够在自然语言处理的多个核心任务上，如情感分析、释义检测和语义文本相似度分析等，展示其优越的性能和良好的泛化能力。

4 方法 Methodology

4.1 minbert 模型的实现

minBERT 模型采用 WordPiece 分词算法处理输入文本，将句子分割成一个个小片段，并将这些片段转化为模型能识别的标识符，同时加入特殊的 [CLS] 和 [PAD] 标记以适应模型对固定长度输入的需求。经过这一处理，文本数据被送入 BERT 模型，该模型由 12 层的 Transformer 编码器构成，每层都包括了能够捕捉不同信息子空间的多头自注意力机制。基于这种深度网络结构，预训练的 BERT 模型进一步通过情感分析任务进行微调，使用 Adam 优化器来有效调整权重，以适应特定任务的需求。微调是通过 BertSentimentClassifier 类实现的，它不仅利用 BERT 输出的嵌入向量来编码句子，还在最终输出上应用 dropout 和线性层来进行情感分类。模型的有效性通过在 SST 和 CFIMDB 数据集上的实验得到了验证，这些实验利用了 BERT 的预训练能力，其中原始 BERT 模型是在维基百科文章上通过掩码语言模型和下一句预测任务进行无监督训练的。

4.2 Adam 优化器

Adam 优化器是一种仅需一阶梯度的高效随机优化方法。该方法通过估计梯度的一阶矩和二阶矩来为不同参数计算自适应学习率。具体来说，算法在每个时间步更新梯度的指数移动平均值 m_t 和梯度的平方 v_t，其中超参数 $\beta_1, \beta_2 \in [0, 1)$ 控制这些平均值的指数衰减。鉴于这些移动平均值在初始时间步被初始化为 0，因此这些平均值会向零偏见。因此，该算法的一个关键方面是在每个时间步进行偏置修正，以获得 $\hat{m}_t$ 和 $\hat{v}_t$。

具体代码体现在 optimizer.py 中的 step 函数，我们通过完善 step 函数实现了 adam 优化器的具体逻辑，另外也可以查看附录的伪代码。

4.3 MultitaskBERT 模型的实现

MultitaskBERT 模型是一种基于预训练的 BERT 架构的多任务学习框架，专门设计用于同时处理多个自然语言处理任务，如情感分析、释义关系识别和语义文本相似度评估。该模型结合了 BERT 的强大语言表示能力，通过在不同的任务上共享学习，以提高对多种文本类型处理的泛化能力。在实现方面，MultitaskBERT 模型通过 multitask_classifier.py 文件中的 MultitaskBERT 类进行定义和训练，该类集成了 BERT 模型的输出，并针对不同的任务提供了专门的预测功能，如 predict_paraphrase、predict_similarity 和 predict_sentiment。模型的训练方法和优化策略灵活多变，允许用户根据具体需求调整，例如通过实现 Adam 优化器的 step() 函数。模型在多个数据集上进行训练和评估，包括 SST、Quora 和 SemEval STS 数据集，以验证其在各种任务上的表现和效果。这种多任务学习方法通过合并不同任务的损失函数，实现了模型更新的综合优化，从而在多个自然语言处理任务上取得良好效果。

4.4 SMART 正则化

许多微调例程遭受过度拟合的困扰，这导致在下游预测任务的测试集上表现不佳。我们仔细地包含了相关领域的语料文本在预训练步骤中，因此我

们不希望我们的微调过于迅速地偏离预训练权重。因此，我们通过实施 SMART 来使用正则化技术。

为了有效控制大型语言模型的高复杂性，SMART 使用了一种诱导平滑性的对抗性正则化技术。所期望的性质是，当输入 x 被微小扰动时，输出不应该变化太多。为了实现这一点，Jiang 等人通过最小化以下损失函数来优化 $f(\theta)$:

$$\min_{\theta} J(\theta) = L(\theta) + \lambda_s R_s(\theta),$$

其中

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} l(f(x_i;\theta), y_i)$$

为常规损失函数，以及

$$R_s(\theta) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\delta_x\|_p \leqslant \epsilon} l_s(f(x_i+\delta_x;\theta), f(x_i;\theta))$$

为正则化项。

正如 Jiang 等人的论文中所述，我们使用 $l_s(P,Q) = D_{KL}(P\|Q) + D_{KL}(Q\|P)$（对称 KL-散度）用于分类问题，以及平方误差损失 $(p-q)^2$ 用于回归任务。这里，正则化项需要一个可以通过使用投影梯度上升法高效解决的最大化问题。注意，这个正则化项在度量我们模型在对称化的 KL-散度度量下的局部 Lipschitz 连续性。也就是说，如果我们向输入注入一个小扰动（约束为 ϵ 在 p-欧几里得度量中），我们的模型输出不会有太大变化。因此，我们可以鼓励我们的模型 f 在输入的邻域内平滑。这在从事低资源领域任务时特别有帮助。

具体代码体现在 multitask_classifier.py。我们新建了一个参数 smartr 用来控制是否开启 smart 正则化，参数为真表示进行 smart 正则化优化。我们新建了一个函数 symmetric_kl_divergence，用来计算对称 kl 散度。随后，我们新建了两个用于计算正则化项的函数 compute_smart_regularization_kl 和 compute_smart_regularization_mse，分别对分类任务以对称 kl 散度作 l_s 与对回归任务以平方差损失作 l_s。训练过程中在损失函数中添加正则化项进行 smart 正则化优化。超参数 lamada 设置为 5，这是经过一系列的调整所最后确定的。

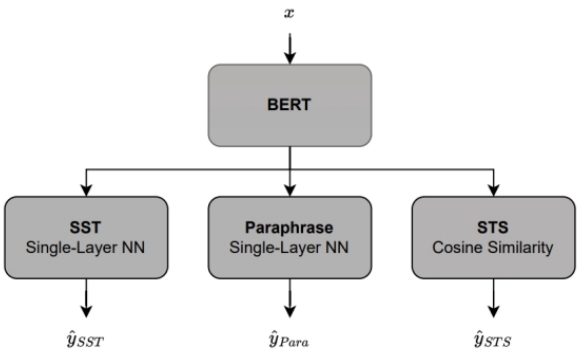
4.5　循环多任务微调

BERT 模型的基础实现原本是假定，仅通过在情感分类任务上进行微调，模型就能有效地适用于释义和相似性预测任务。然而，即便是基础实验也表明这种假设并不成立。为了开发一种能够跨多种任务有效应用的方法，我们采取了一种批量级别的轮流多任务学习（MTL）策略。在加载了预训练的 BERT 权重后，我们轮流处理情感倾向性分析（SST）、释义以及语义文本相似性（STS）任务的数据，并保持固定的批量大小。在每个批次的迭代过程中，我们对每个数据集的相关参数进行一次更新。对于 SST 任务，我们使用一个单层神经网络来输出各个情感类别的逻辑值。对于释义任务的数据，我们使用一个单层神经网络，它不仅处理句子对的嵌入向量 $\mathbf{h}_u$ 和 $\mathbf{h}_v$，还处理它们的绝对差值 $|\mathbf{h}_u - \mathbf{h}_v|$，类似于 SBERT[2] 中的做法。对于 STS 任务，我们发现，计算两个句子嵌入的余弦相似度是一个有效的方法。我们的模型在这三个任务上分别使用交叉熵、二元交叉熵和均方误差作为损失函数。总体来说，这种轮流策略确保了我们的模型能够接触到所有任务的数据。

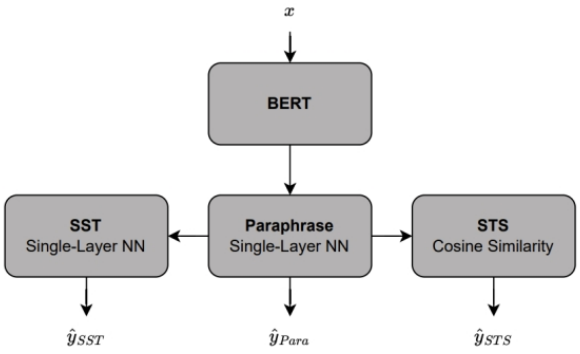
虽然我们期望通过等权重的轮流微调（结合 SMART 技术）能有效地同时学习多个任务，但这种方法仅限于每个任务使用相同数量的数据点。因此，对于数据量较大的数据集（例如 Quora），我们只能使用其中的一部分数据。为了充分利用我们丰富的数据资源，我们尝试了两种调整多任务微调逻辑的策略：一是在 Quora 数据集上进行额外的预训练 (Additional pre-train)，二是采用交错轮询 (Round-robin) 方法。第一种策略首先使用大部分释义数据进行初始训练，然后在最后阶段对所有三个任务实施等量批处理的轮流 MTL。这样做的目的是希望通过先在释义数据上进行学习，为其他任务提供一个良好的起点，并通过使用更丰富的数据来提高相应任务的性能。然而，额外的释义检测预训练可能对其他任务不太相关，甚至可能导致模型遗忘了更具通用性的 BERT 嵌入特征。因此，我们的第二种策略——交错方法，旨在充分利用数据资源，同时保持各任务学习的独立性。具体来说，我们降低了小数据集的批量大小，在每个批次中加入更多比例的 Quora 数据。理想情况下，这将提升释义任务的表现，同时也不会损害情感分类任务的准确度。图 1 对这种轮流模型及其调整进行了可视化展示。

4.6　共享相似任务的关系层

为了进一步发挥我们数据的潜力，我们对模型进行了调整，使其能够处理跨任务的关联性。具体来说，当多个任务之间具有相似性时，可以通过结



(a) Round-robin 架构



(b) Additional pre-train 结构

图 1　两种不同的 BERT 微调训练架构

合这些任务中的数据信号来促进整体的学习效果。基于这个理念，我们提出在针对特定任务进行微调之前，为相似的任务之间加入一个联合的关联层，这一点类似于 MT-DNN[3]。我们推测，特别是当某些联合任务的数据量不足时，这种方法将会特别有效。本质上，这种做法是迁移学习的一种应用，我们之前在 minBERT 项目中已经采用过，但现在我们将其应用到了相似任务的特殊环境中。

在我们用来进行评估的三个下游任务中，释义任务和语义文本相似性任务有着紧密的联系；这两个任务都是在不同的尺度上评估句子对的相似度。同时，STS 数据集的规模远小于 Quora 数据集。我们希望通过在进入特定任务学习之前先通过一个共享层（采用 LeakyReLU 非线性函数）来处理这两个数据集，从而利用释义数据的丰富性来提升 STS 任务的学习效果。虽然在等权重批次的设置中关联层已经足够有用，但我们更加关注于交错轮询方法，也就是尽可能多地利用释义数据来增强 STS 任务。

值得注意的是，这样一个将嵌入对映射到单个隐藏层嵌入的共享层，不允许我们继续使用余弦相似度来处理 STS 任务。相反，我们采用的是缩放的哈达玛积作为特征输入到关联层：

$$\frac{\mathbf{h}_u \circ \mathbf{h}_v}{\|\mathbf{h}_u\|_2 \|\mathbf{h}_v\|_2 + \epsilon}$$

这里，$\mathbf{h}_u \circ \mathbf{h}_v$ 表示向量 $\mathbf{h}_u$ 和 $\mathbf{h}_v$ 的元素级别乘积，而分母中的 $\|\mathbf{h}_u\|_2$ 和 $\|\mathbf{h}_v\|_2$ 是这两个向量的欧氏范数，ϵ 是为了防止除零错误而添加的一个小的正数。这种方法通过关联层合并嵌入特征，其求和等于余弦相似度。图 2 展示了这一复杂的关联架构。

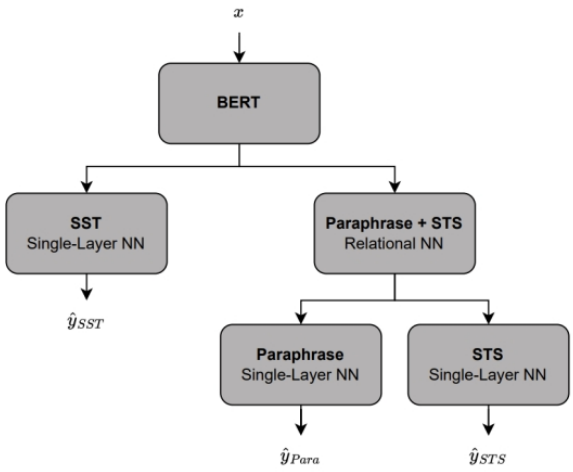


图 2　带有共享关系层的网络架构

5　实验 Experimental Results

5.1　数据集

5.1.1　情感分析数据集

斯坦福情感树库（SST）数据集

斯坦福情感树库数据集包含来自电影评论的 11,855 个单句。该数据集通过斯坦福解析器解析，并包含了总共 215,154 个独特的短语，每个短语由 3 个人工评价者进行注释。每个短语被标记为负面、较负面、中性、较正面或正面。在本项目中，使用 BERT 嵌入来预测这些情感分类标签。

总结一下，对于 SST 数据集我们有以下的划分：

- 训练集（8,544 个样本）
- 验证集（1,101 个样本）
- 测试集（2,210 个样本）

CFIMDB 数据集

CFIMDB 数据集包含了 2,434 个极端情感的电影评论。每个评论都有一个二元标签，标记为负面或正面。我们注意到这些评论中的许多都比一句话

要长。在这个项目中，使用 BERT 嵌入来预测这些情感分类。

总结一下，对于 CFIMDB 数据集我们有以下的划分：

- 训练集（1,701 个样本）
- 验证集（245 个样本）
- 测试集（488 个样本）

5.1.2 释义检测数据集

Quora 数据集

Quora 数据集，包括 400,000 个问题对，这些问题对的标签指示了特定实例是否为彼此的释义。我们为您提供了该数据集的一个子集，具有以下划分：

- 训练集（141,506 个样本）
- 开发集（20,215 个样本）
- 测试集（40,431 个样本）

5.1.3 语义相似度数据集

SemEval STS 基准数据集

SemEval STS 基准数据集，包含 8,628 个不同句子对，这些句子对在从 0（无关）到 5（等同含义）的范围内具有不同的相似度。

对于 STS 数据集，我们有以下划分：

- 训练集（6,041 个样本）
- 开发集（864 个样本）
- 测试集（1,726 个样本）

5.2 实验具体参数

所有实验都有一些共通的设置；我们运行了 10 个 epoch，每个 epoch 的学习率为 1e-5，并设置隐藏层 dropout 概率为 0.3。对于我们的预训练默认 minBERT 模型，我们使用了学习率为 1e-3 的预训练 batch 大小为 64。

损失正则化和优化器步骤 [smart] 特定于我们 SMART[2] 实现的超参数值是 $\lambda = 5$，$\epsilon = 1e-5$，$\rho = 1e-5$，$\beta = 0.995$，$\mu = 1$，并且 $\eta = 1e-3$。

轮询多任务微调 [rrobin] 对于加权相等的轮询，我们固定所有数据集的 batch 大小为 64，并计算出每个 epoch 所需的迭代次数为 floor(len(sts_train_data)/$args.batch\_size$)。对于带有额外预训练的轮询 (robin-pre)，我们分别为预训练和轮询部分设置 batch 大小。默认情况下两者都设置为 64，但我们将其减少到了 16，如果运行带有 SMART 的模型。对于交织轮询 (robin-full)，我们固定每个 epoch 的迭代次数，并相应地调整每个数据集的 batch 大小。鉴于最大 batch 大小为 64，对于释义数据，我们推断 SST 和 STS 的 batch 大小分别为 3 和 2。

富关系层结合相似任务 [rlayer] 富关系层方法遵循与交织轮询相同的参数化设置。也就是说，当权重相等时，我们可以使用 batch 大小为 64，而在所有释义数据通过关系层时，我们会使用更小的调整后的 batch 大小为 SST 和 STS。

5.3 实验结果

我们使用准确率来评估情感分析和释义检测任务中不同模型的表现，并用皮尔逊相关系数来评估语义文本相似性（STS）任务。值得注意的是，SST 数据集是一个多类分类问题，而释义检测和 CFIMDB 的任务则是基于二元标签的。在表 1 中，我们比较了基线 BERT 模型（包括预训练和微调阶段）在 Google Colab GPU 上每次训练周期的运行时间以及单任务分类器的开发集准确率。

表 1 基础 BERT 模型（包括预训练和微调）在单任务分类器上的发展集准确率与标准对比，以及每个训练周期所需的 Colab 运行时间

	Sentiment (SST)			Sentiment (CFIMDB)		
Model type	Accuracy	Benchmark	Runtime (s)	Accuracy	Benchmark	Runtime (s)
Pretrain default	0.393	0.390 (0.007)	30	0.788	0.780 (0.002)	45
Fine-tune default	0.522	0.515 (0.004)	120	0.963	0.966 (0.007)	150

在表 1 中，我们报告了针对情感分类任务的基线预训练（最小化的）BERT 模型和经过微调的模型在开发集上的表现，并与项目讲义中的基准数据进行了对比。我们的结果与 CS224N 官方给出的基准值非常接近，这让我们对我们的实现的“正确性”感到自信。预训练的 BERT 模型的表现明显优于随机猜测，这表明其迁移学习在一定程度上是有效的，并且微调显著提升了分类的准确率。于是，接下来我们着手个方面的 BERT 优化工作。

表 2 为我们提供了在 SST、Quora 和 STS 任务上各种模型方案的开发集性能。其中包括三种非多任务模型：即表 1 中的 BERT 基线模型及其微调版，还有采用 SMART 技术微调的 SST 模型。除此之外，我们还对包含 SMART、轮询策略和丰富的关系层模型的多任务学习方法进行了评估。从表 2 我们可以看到，SMART 在单任务模型中的表现最优，在

表 2 单任务和多任务模型在三个微调任务上的开发集准确度。模型分为三组：(i) 单任务方法，(ii) 小规模（等权重批次）方法，(iii) 大规模丰富关系（关联）方法。每个任务的最佳模型用粗体表示。在 AWS EC2 实例上，每个训练周期的运行时间以 Colab GPU 的秒为单位给出。

模型类型	Sentiment (SST)	Paraphrase (Quora)	Similarity (STS)	运行时间 (s)
Pretrain default	0.396	0.380	0.019	9
Fine-tune default	0.525	0.522	0.240	25
Fine-tune smart	0.520	0.501	0.382	161
Fine-tune robin	0.524	0.726	0.580	67
Fine-tune robin+smart	**0.532**	0.741	0.680	464
Fine-tune robin-pre+smart	0.519	0.751	0.690	3,037
Fine-tune robin-full	0.498	0.763	0.762	1,420
Fine-tune robin-full+smart	0.485	0.750	0.714	4,549
Fine-tune robin-full+rlayer	0.501	**0.767**	**0.802**	1,550

STS 任务上获得了最高分。不过意料之中的是，所有多任务模型在释义和 STS 的得分上都轻松超过了单任务基线模型。在仅使用限定数据量的多任务模型中（轮询微调、轮询 +SMART 微调），SMART 策略再次表现出色，尤其是在 STS 得分方面。这似乎说明，在使用有限数据量的情况下，如果不采取规则化手段，该任务在训练时容易出现过拟合。当使用整个 Quora 数据集时，释义的准确率显著提升，数值超过了 75%。在这种情况下，SMART 的额外优势似乎不复存在，即非 SMART 模型的 STS 得分超过了采用 SMART 的模型。

总体来说，表现最好的是采用丰富关系层和交错轮询策略的模型，其在测试集上的性能分别达到了 0.503（SST）、0.765（释义）和 0.789（STS）。这个模型在 STS 评分上的出色表现确认了我们的假设：通过共享层挖掘释义和 STS 数据之间的关联，能够提升相似任务的性能。在运行时间方面，当与更多数据配合使用时，SMART 是最消耗资源的方法。

5.4 结果分析

我们的实验结果及其分析表明，在单一下游任务上应用 BERT 的微调能显著提高性能，如表 1 所示。然而，这样单一任务的微调模型并不适合泛化到其他预测任务上，正如我们在表 2 中的单任务方法观察到的那样。这促使我们转向多任务学习方法。与只在 SST 数据集上训练的单任务模型相比，我们发现轮询式的多任务学习方法（表 2 中的 fine-tune rrobin）能显著提升我们的实验结果。这种方法通过均衡地引入其他任务的微调，避免了在情感分析性能上的损失。但在这一过程中，我们也遇到了过拟合问题。

这引导我们采用了 SMART 策略。SMART 的高效性在于它能够保持预训练中获得的结构，防止微调数据集可能引起的波动。特别是在数据集较小或分布不均时，这一点尤为重要。因为这样的数据集中稀疏的特征空间往往会产生尖锐的决策边界，这些边界可能无法泛化，导致在新样本上的表现不佳。我们的结果似乎验证了这一假设。在默认微调模型中引入 SMART 后，我们的 STS 性能提高了约 14%，而在轮询微调模型中引入 SMART（fine-tune rrobin+smart）使 STS 得分提高了约 17%。同时，其他两项任务的性能保持稳定。在数据受限的环境中，这种结果是有意义的，因为我们的训练样本数量受限于三个任务中最小的数据集。

鉴于我们的默认模型和等量轮询模型没有利用所有可用数据，我们决定在数据更丰富的环境中进行学习。首先，我们尝试通过在额外的" 预训练" 层使用完整的 Quora 数据集，然后继续执行多任务学习（rrobin-pre+smart）。正如表 2 所展示的，这一做法在我们的里程碑性能上，为释义任务和语义文本相似度任务都带来了实质性的提升。因此，我们认识到 Quora 数据集对于释义任务很有用，同时也对文本相似度任务有益。这激发了我们去验证这样的假设：引入更多数据的好处大于不平衡微调的潜在风险。我们采用了一种更复杂的多任务方法，该方法通过使用变化的批量大小将较大的 Quora 数据集纳入到微调过程中（rrobin-full）。如预期的那样，这进一步提升了性能，特别是 STS 的表现。

值得注意的是，表 2 还显示，向该模型添加 SMART 规则化（rrobin-full+smart）并没有提高性

能，这可能说明了随着使用的数据量增加，规则化的边际效益在减小，规则化甚至可能在此背景下过度限制了学习。也有可能是由于在丰富数据的 SMART 环境下降低的批量大小导致了过大的学习波动。

当我们在包含完整 Quora 数据集的模型中观察到 STS 得分的提升，以及在第 5.1 节中对句子对进行深入分析时，我们认识到释义任务和 STS 任务之间存在某种关联。为了利用这种关联，我们设计了一个丰富而

关联的架构（图 2），该架构在共享相似任务的同时使用所有数据（rrobin-full+rlayer）。这个模型证明是我们最好的模型，其释义任务和 STS 得分都超过了 0.75。需要注意的是，对于所有利用丰富数据的模型，SST 的准确率略有下降。这很可能是由于添加的 Quora 数据对情感任务无关或有干扰作用所致。不过，SST 准确率的下降相对于释义任务和 STS 得分的增益来说是小的，这让我们认为 SST 任务与其他任务在某种程度上是‘正交’的。

关于关联性和正交性的概念促使我们设计了我们的训练步骤。具体来说，我们首先将三个预测任务的损失函数加起来，并在每次训练迭代中对这个组合损失执行单个反向和优化步骤。然而，我们发现，按预测任务顺序进行单独的步骤可以提高性能。这可以通过 STS 和释义任务的梯度更新非常一致，或者与 SST 任务相对正交来解释。此外，这样我们可以采取更多的优化步骤，并且在将不同数量级的损失相加时不会面临规模问题。

6 总结与展望 Conclusion

本项目成功实现了微型 BERT 模型（minBERT）的设计与优化，并在多个自然语言处理任务上验证了其性能。通过精简的 Transformer 架构和多头注意力机制，minBERT 展示了其在情感分析、释义检测以及语义文本相似度（STS）任务上的强大能力。特别是，我们采用的 smart 正则化技术有效防止了过拟合，而轮询多任务微调策略确保了模型在不同任务之间的平衡学习。此外，通过引入富关系层，minBERT 在处理相似任务时展现了更丰富的语义理解。

然而，尽管我们取得了一定的成果，但未来的研究仍有广阔的天地。首先，模型的进一步精简以适应更严苛的资源限制是一个潜在的研究领域。其次，考虑到不同语言和领域的适应性，拓展 minBERT 以处理跨语言和跨领域的任务将是有益的尝试。最后，深入探索不同类型的自然语言理解任务之间的相互作用，以及它们对模型学习的影响，将帮助我们设计更为通用和高效的多任务学习策略。

随着模型和算法的不断进步，我们期望 minBERT 能够在自然语言处理的广阔天地中发挥更大的作用，尤其是在资源受限的应用场景中。未来工作的重点将放在提高模型的通用性和可解释性上，以及在现实世界应用中的可行性测试。

参考文献

[1] JIANG H, HE P, CHEN W, et al. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization[J]. CoRR, 2019, abs/1911.03437.

[2] REIMERS N, GUREVYCH I. Sentence-bert: Sentence embeddings using siamese bert-networks[J]. arXiv preprint arXiv:1908.10084, 2019.

[3] HE P, CHEN W, LIU X, et al. Improving multi-task deep neural networks via knowledge distillation for natural language understanding[J]. arXiv preprint arXiv:1904.09482, 2019.

[4] AGIRRE E, CER D, DIAB M, et al. SemEval 2013 shared task: Semantic textual similarity[C]//Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity: volume 1. [S.l.: s.n.], 2013: 32-43.

[5] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems. [S.l.: s.n.], 2017: 5998-6008.

[6] DEVLIN J, CHANG M W, LEE K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[7] LIU Y, OTT M, GOYAL N, et al. Roberta: A robustly optimized bert pretraining approach[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. [S.l.: s.n.], 2019: 4221-4231.

[8] LAN Z, CHEN M, GOODMAN S, et al. Albert: A lite bert for self-supervised learning of language representations[C]//International Conference on Learning Representations. [S.l.: s.n.], 2019.

[9] CLARK K, LUONG M T, LE Q V, et al. Electra: Pre-training text encoders as discriminators rather than generators[C]//International Conference on Learning Representations. [S.l.: s.n.], 2020.

[10] HEINZINGER M, ELNAGGAR A, WANG Y, et al. Bertology meets biology: Interpreting attention in protein language models[J]. Nature Methods, 2021, 18(4):397-406.

附录 A 成员分工

共同完成了使得斯坦福大学 CS224N 期末项目 min-BERT 和 MultitaskBERT 正常工作的代码；之后程佳诺完成了 SMART 正则化，王浩完成了额外预训练 (additional pretrianing), 交织轮询 (round-robin) 和共享关系层 (shared relational layer)；共同完成模型的测试和论文撰写；github 链接：https://github.com/water-00/NKU-NLP-project

附录 B SMART 正则化

算法 1 SMART 正则化

Require: f: 模型, $\{x_i\}_{i=1}^{Q}$: 模型嵌入, $\{z_j\}_{j=1}^{B}$: 批量输入, σ^2: 噪声方差, $\eta = 10^{-3}$: 对抗性更新的学习率, $\epsilon = 10^{-5}$: 最大扰动

f 设置为评价模式

$y_j \leftarrow f(z_j, x)$

for x_i in Q **do**

$\tilde{x}_i \leftarrow x_i + \nu_i$, 其中 $\nu_i \sim \mathcal{N}(0, \sigma^2 I)$

end for

for y_j in B **do**

$l_j \leftarrow \nabla_{l_s}(f(z_j, x))$

end for

$g_i \leftarrow \left(\frac{1}{B}\sum_j l_j\right)\left(\frac{1}{\|\frac{1}{B}\sum_j \hat{l}_j\|_\infty}\right)^{-1}$

for x_i in Q **do**

$\tilde{x}_i \leftarrow \Pi_{x_i+\eta g_i, \|\cdot\|_2 \leqslant \epsilon}(\tilde{x}_i + \eta g_i)$

end for

$y_j^{adv} \leftarrow f(\tilde{z}_j, \tilde{x})$

$L_{adv,classification} \leftarrow \frac{1}{B}\sum_j D_{KL}(y_j \| y_j^{adv}) + D_{KL}(y_j^{adv} \| y_j)$

$L_{adv,regression} \leftarrow \frac{1}{B}\sum_j \|y_j - y_j^{adv}\|_2^2$

f 设置回训练模式

附录 C Adam 最优化

算法 2 Adam 算法：$\hat{g}_t^2$ 表示元素级的平方 $g_t \circ g_t$。所有向量上的操作都是元素级的。用 $\boldsymbol{B}_1^t$ 和 $\boldsymbol{B}_2^t$，我们表示 $\boldsymbol{B}_1$ 和 $\boldsymbol{B}_2$ 的 t 次幂。

Require: α: 步长

Require: $\beta_1, \beta_2 \in [0, 1)$: 动量估计的指数衰减率

Require: $f(\theta)$: 带有参数 θ 的随机目标函数

Require: θ_0: 初始参数向量

$m_0 \leftarrow 0$ (初始化第一动量向量)

$v_0 \leftarrow 0$ (初始化第二动量向量)

$t \leftarrow 0$ (初始化时间步)

while θ_t 未收敛 **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla f(\theta_{t-1})$ (在时间步 t 获取相对于随机目标函数的梯度)

$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (更新有偏第一动量估计)

$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (更新有偏第二原始动量估计)

$\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (计算校正后的第一动量估计)

$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (计算校正后的第二动量估计)

$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (更新参数)

end while

return θ_t (最终参数)