手机屏幕的截图

中度可信度描述已自动生成

文本数据挖掘大作业

成员1：李一鸣 202228013229030

成员2：张 兆 202228013229029

成员3：陈国鑫 2022E8013282125

代码开源地址：https://github.com/zhangzhao219/Multimodal-Sentiment-Analysis

2023年05月

目录

[1 引言 1](#_Toc132833827)

[1.1 任务概览 1](#_Toc132833828)

[1.2 数据描述 1](#_Toc132833829)

[1.3 评价指标 2](#_Toc132833830)

[2 基于文本模态的语音情感识别 2](#_Toc132833831)

[2.1 数据处理 2](#_Toc132833832)

[2.2 传统方法 3](#_Toc132833833)

[2.2.1 线性回归 4](#_Toc132833834)

[2.2.2 Logistic回归 4](#_Toc132833835)

[2.2.3 朴素贝叶斯 5](#_Toc132833836)

[2.2.4 决策树 6](#_Toc132833837)

[2.2.5 随机森林 7](#_Toc132833838)

[2.2.6 支持向量机 7](#_Toc132833839)

[2.3 深度学习方法 8](#_Toc132833840)

[2.3.1 TextCNN & BiLSTM 8](#_Toc132833841)

[2.3.2 预训练模型 9](#_Toc132833842)

[2.3.3 Prompt Tuning 10](#_Toc132833843)

[2.4 文本情感分析实验 11](#_Toc132833844)

[3 文本情感识别系统 12](#_Toc132833845)

[4 参考文献 15](#_Toc132833846)

1. 引言
   1. 任务概览

情感分析是自然语言处理领域的一个重要分支，主要指通过对文本中的主观信息进行挖掘，判断文本作者的情感态度。

随着移动互联网的普及，网民已经习惯于在网络上表达意见和建议，比如电商网站上对商品的评价、社交媒体中对品牌、产品、政策的评价等等。这些评价中都蕴含着巨大的商业价值。比如某品牌公司可以分析社交媒体上广大民众对该品牌的评价，如果负面评价忽然增多，就可以快速采取相应的行动。而这种正负面评价的分析就是情感分析的主要应用场景。除此之外，应用场景还包括电影评论、舆情监控等。

文本情感分析指的是利用算法来分析提取文本中表达的情感。例如分析一个句子表达的好、中、坏等判断，高兴、悲伤、愤怒等情绪。如果能将这种文字转为情感的操作让计算机自动完成，就节省了大量的时间。对于目前的海量文本数据来说，这是很有必要的。

* 1. 数据描述

我们将数据集按照7：2：1划分训练集、验证集与测试集。

* + 1. IMDB

我们将数据集按照7：2：1划分训练集、验证集与测试集。

* + 1. Climate

我们将数据集按照7：2：1划分训练集、验证集与测试集。

* + 1. Waimai

我们将数据集按照7：2：1划分训练集、验证集与测试集。

* + 1. IEMOCAP

南加州（USC）语音分析和解释实验室（SAIL）收集了交互式情感二元运动捕捉数据库IEMOCAP[2]，可以进行语音情感识别探究。该数据集包含大约十二个小时数据，记录了来自 10 位演员的二元会话信息，它们被要求在假设场景中即兴对话，旨在引发特定的情绪（快乐、愤怒、悲伤、沮丧和中性等状态），此外，数据集当中还包含演员的面部、手部动作等模态信息。

考虑到本文致力于探索文本情感识别，并考虑到平衡数据类别分布等因素，本文选择从数据库当中抽取音频信息转录的文本信息，由于某些类别存在较为严重的长尾现象，因此，本文选择了“neu”（中性）、“sad”（悲伤）、“hap”快乐、“ang”愤怒等四个类别，并将“exc”（激动）类别划分为“hap”类别。划分后的数据集包含5531条数据，数据分布如下图所示：

图表, 饼图

描述已自动生成

图1 数据集标签分布情况

* 1. 评价指标

本文选取以下的四种评价标准对情感识别进行评价：

（1）Acc.

（2）Pre.

（3）Rec.

（4）F1

1. 基于文本模态的语音情感识别
   1. 数据处理

对于文本来说，与结构化的数值数据不同，首先要进行一些预处理操作，提高文本数据的完整性、正确性和一致性，从而提高模型训练和预测的效率。我们在将文本输入到传统机器学习模型之前进行了一系列的文本预处理操作：

（1）数据清洗。从原始的文本数据中删除标签为空或者标签不合理的数据，同时过滤一些明显长度过短的文本。

（2）分词。对于中文数据来说，文本一般是通过句子的形式给出，因此需要首先将句子划分为词语。在本次任务中我们使用jieba分词工具进行分词。Jieba分词工具性能高效，准确性高，成熟稳定，是中文自然语言处理的重要基石之一。

（3）去除停用词。将一些经常出现在文本中但是语义不强的词汇，如“的”、“是”、“了”、“the”等进行去除，防止其数量过多稀释了重要的模型特征权重，影响模型性能。在本次任务中使用百度的停用词表进行过滤。

（4）保留某些标点符号。一般来说，对于文本的预处理是要移除掉所有的标点符号的，且标点符号也可以作为分词的一个依据。但是对于情感识别任务来说，问号、感叹号等标点符号在一定程度上反映了当前人的一些心理状态。因此需要保留这些比较特殊的标点符号作为文本的一种类别的特征。

* 1. 基于情感词典的方法
  2. 传统机器学习方法

除了进行一系列的数据预处理对文本数据进行清洗外，还需要将文本的特征提取出来，构造模型可以识别的数值型输入，才可以使用机器学习模型进行情感识别。

将文本转换为数字特征的方式有很多，例如词袋模型，n-gram模型，词嵌入等等。在本次任务的传统方法部分，我们使用了TF-IDF提取文本的特征。

TF-IDF的主要思想是：如果一个词或短语在一篇文章中出现的频率高，同时在其他文章中很少出现，则认为此词或短语对该文章具有很强的相关性，具有很好的区分度。这体现了词语在语义上的重要性，是信息检索与文本挖掘中常用的评价指标之一。

TF-IDF通过以下两个指标来衡量一个词语的重要性：

1. 词频（Term Frequency, TF）:某个词语在一篇文档中出现的次数。词频可以衡量一个词语在该文档中的重要程度，出现次数越多，该词可能越重要。

2. 逆文档频率（Inverse Document Frequency, IDF）：某个词语在整个文档集合中的词频，用来衡量该词语在所有文档中的常见程度。如果一个词在很多文档中出现，那么该词的区分度较小，IDF值会较小。相反，如果一个词只在少数文档出现，该词的IDF值会较大，说明其具有很好的区分度。

TF-IDF的值由这两个指标共同决定，通常使用以下公式计算：

其中，词频TF衡量词语在某个文档中的重要性，IDF通过惩罚在许多文档中出现的常用词，而提高只在少数文档中出现的词的权重，起到平衡的作用。

TF-IDF可以有效地衡量一个词语的重要性。由于它考虑了词语在该文档和所有文档中的频率信息，所以可以过滤掉常见但语义较弱的词，更强调比较具有区分度的词语。这使得TF-IDF特别适用于文本的特征提取与关键词识别等任务。

* + 1. 线性回归

线性回归是一种最简单的机器学习算法，假设特征与标签之间存在线性关系。线性回归的模型表达为：

其中，到是个特征，到是每个特征对应的权重，是模型的偏置。对任意一个特征，如果其权重为正，那么对有正相关影响；如果为负，那么对有负相关影响。权重的大小表示该特征对模型预测的重要性。对于本次的任务来说，特征即为通过TF-IDF提取的文本特征，标签即为各类别的实际标签。

线性回归旨在找到一组权重和偏置，可以使得模型对训练数据的预测与真实标签尽量接近。这是一个优化问题，常用的损失函数是均方误差，通过梯度下降等方法不断调整和的值以最小化损失函数，从而得到最优的模型。

线性回归方法比较简单，容易理解，且计算效率高；缺点是假定了线性关系，因此无法处理非线性问题，学习能力有限，且容易过拟合，对异常值比较敏感。对于本次任务来说，由于标签是离散的值，因此只能将经过模型后输出的值与标签绝对距离进行比较，实际上线性回归并不适用于本次的任务。

* + 1. Logistic回归

逻辑回归是一种广泛使用的机器学习算法，用于解决二分类问题，也可以进行扩展从而解决多分类问题。

逻辑回归模型表达为：

其中，到是n个特征，到是每个特征对应的权重，是模型的偏置。

逻辑回归训练的目的是最大化训练数据被正确分类的概率。它采用最大似然估计法，通过梯度上升等优化方法不断更新权重和的值，使模型对训练数据的分类结果与真实标签尽可能吻合。

多类逻辑回归模型的原理如下：

1. 假设类别之间是互斥的，即每个实例只能属于一个类别。对每个类别,引入表示该实例属于类别的概率。

2. 为每个类别构建一个逻辑回归模型,预测属于该类别的概率。具体而言:

这里是类别对应的权重向量。

3. 对新输入，在每个逻辑回归模型中计算属于对应类别的概率，然后选择概率最大的类别作为预测类别。

多类逻辑回归的优点在于原理简单，易于理解和实现，并且可以在不加修改的情况下，扩展到更多类别。缺点是无法学习类别之间的相关性，且计算量比较大，训练代价高。

* + 1. 朴素贝叶斯

朴素贝叶斯是一种简单实用的机器学习算法，用于解决分类问题。原理如下：

1. 假设特征之间相互独立。这是朴素贝叶斯算法的核心假设，该假设简化了模型使其易于计算。

2. 根据贝叶斯定理计算后验概率。对于某个类别和特征向量，贝叶斯定理为：

其中是后验概率，是联合概率，是先验概率

3. 构建分类器。对于新输入，计算它属于每个类别的后验概率，然后预测类别为先验最大的那个类别。

朴素贝叶斯模型表达为：

其中到是个特征，是类别的先验概率，是特征在类别下的条件概率。

朴素贝叶斯的“朴素”是因为它假定特征相互独立。这个假设简化了模型,使其易计算,但也限制了其表达能力。然而即便特征之间存在依赖,朴素贝叶斯仍然表现良好。这是因为贝叶斯定理在分类效果上更为重要。朴素贝叶斯通过训练统计特征-类别共现频次，获得先验概率和条件概率。预测阶段直接应用贝叶斯定理,以获得每个类别的后验概率。

朴素贝叶斯的优点是计算简单，易理解且易实现。缺点是其对属性独立的假设限制了模型的表达能力，且对小训练集表现不佳。

* + 1. 决策树

决策树是一种流行的机器学习算法，用于解决分类和回归问题。其原理如下：

1. 以树形结构表示决策过程。决策树由根节点、内部节点(非叶子节点)和叶子节点组成。

2. 根节点代表完整训练数据集。内部节点表示数据的某个特征，每个子节点代表该特征的一个取值。

3. 叶子节点代表数据集的分类结果或回归结果。分类树的叶子节点包含类别标签，回归树的叶子节点包含连续值。

4. 决策树通过递归地对数据进行切分，直到达到停止条件，逐步生成。

5. 对新数据实例，从根节点开始，依据其特征取值递归地向下移动，最终到达叶子节点，获得预测结果。

决策树主要学习算法有：ID3、C4.5、CART。其中CART构建决策树通过贪心算法，按照特征选择和数据切分的方式生成二叉树。其具体步骤为：

1. 选择最优特征和切分点：计算各个特征和各个切分点的代价，选择代价最小的特征和切分点进行切分。

2. 数据切分：将节点的数据切分成子节点，根据最优切分点的取值将数据划分到左子节点或右子节点。

3. 生成子节点：为左右子节点重复以上步骤，直到达到停止条件。

4. 剪枝：采用CART算法可以剪去一些子节点，防止决策树模型过拟合。

决策树算法结果易于理解，对异常值不太敏感。但可能导致过拟合，学习结果随训练数据的变化而变化大。决策树是最简单,使用最广的机器学习算法之一。

* + 1. 随机森林

随机森林是一种流行的机器学习算法，用于解决分类和回归问题。它的原理如下：

1. 随机森林由多个决策树组成。每个决策树独立进行训练，最后将多个决策树的结果综合得到最终输出。

2. 随机森林使用bagging的思想，通过有放回地从训练数据中抽样获得子数据集来训练每个决策树。

3. 构建每个决策树时，随机选取特征子集。这增加了随机森林的多样性，避免过拟合。

4. 对新输入实例，将其输入到所有的决策树，并统计各类结果出现的次数，最终预测出现次数最多的类。

由于数据、特征的随机抽取，每棵决策树不同，随机森林具有较强的多样性，从而减少了过拟合的风险。随机森林准确性高、对异常值鲁棒、易于理解和实现。缺点是计算开销大、训练时间长。

* + 1. 支持向量机

支持向量机（SVM）是一种用于解决分类和回归问题的机器学习算法。它的原理如下：

1. SVM通过寻找超平面将不同类的训练数据分开，并且使得该超平面与最近的训练数据点之间的距离尽可能大。

2. SVM使用核技巧隐式地将数据映射到高维空间，使数据在高维空间线性可分，而在原始空间不可分。常用有线性核、多项式核和RBF核等。

3. SVM使用最大间隔分类器，寻找能将不同类的训练数据分开的超平面，使得该超平面与最近的数据点的距离最大。

4. SVM因此得到一个分类器，能够对新数据判断类别。对于非线性可分数据集，SVM可以通过核技巧映射得到其在高维空间的表现，仍然使用最大间隔分类器构建超平面，从而实现非线性分类。

SVM理论基础坚实、泛化能力强，通过核技巧可处理非线性问题，对中间空间数据不敏感。缺点是核的选择和超参数的调优比较困难。

* 1. 深度学习方法
     1. TextCNN & BiLSTM

文本情感分类是一项重要的自然语言处理任务，常常需要利用深度学习技术进行处理。在这些深度学习技术中，TextCNN和双向长短期记忆网络（BiLSTM）已被证明可以有效地应用于文本情感分类任务。

1. TextCNN：TextCNN 是一种具有多通道输入、多尺度卷积和全局最大池化等特征的卷积神经网络模型。它能够充分利用文本的局部和全局信息，并且能够高效地捕捉文本中的关键特征。因此，在情感分类任务中，TextCNN 能够有效地提高分类准确率和鲁棒性，并且具有良好的效率和可扩展性。TextCNN的网络架构示意图如下：

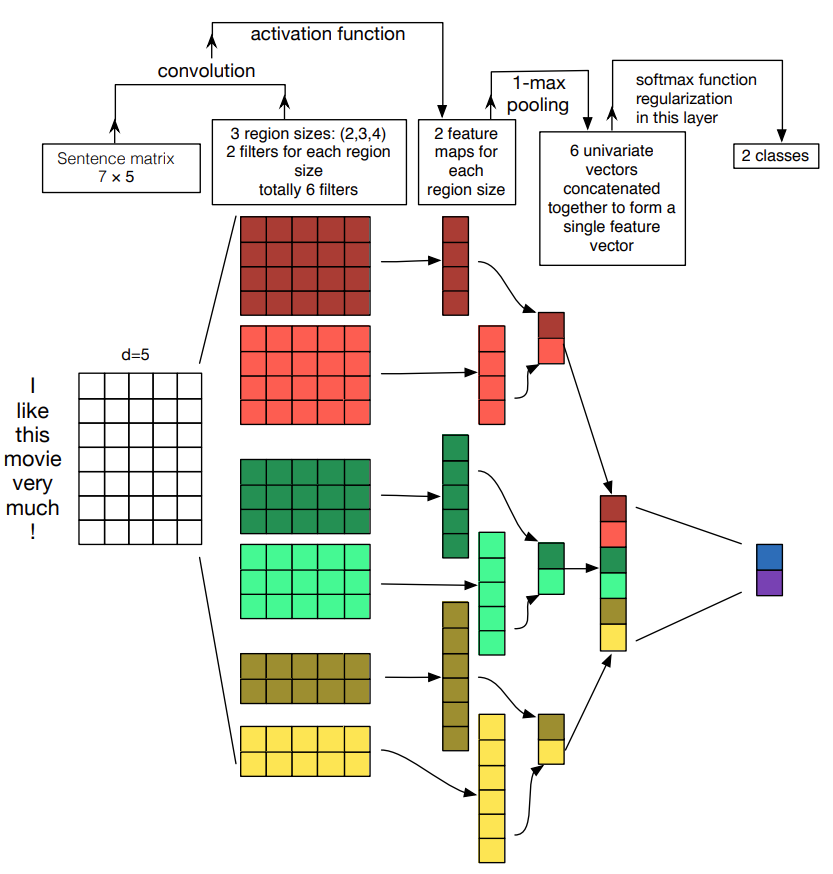


图5 TextCNN模型示意图

1. BiLSTM：BiLSTM 是一种基于循环神经网络（RNN）的模型，在处理时序数据上表现出色。BiLSTM 能够通过正向和反向传递信息，充分利用文本的上下文信息，并且具有优秀的建模能力。在情感分类任务中，BiLSTM 能够更好地处理长文本序列，并且对上下文信息的利用更加精细和全面，因此可以取得更好的分类效果。BiLSTM的网络架构示意图如下：

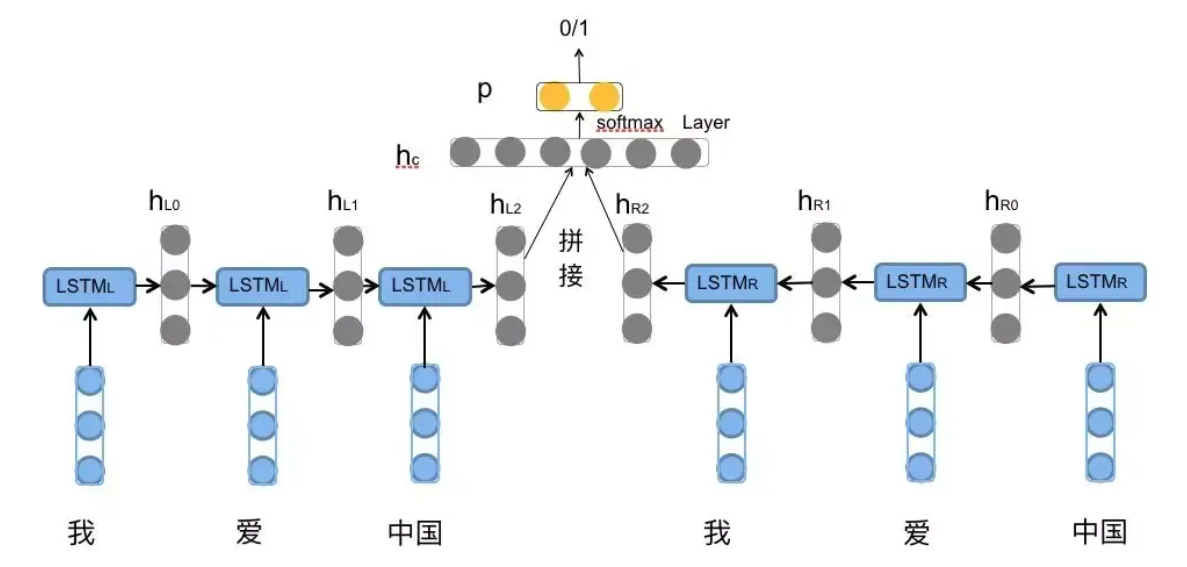


图6 BiLSTM模型示意图

TextCNN通过预训练的词向量模型得到文本数据的词向量矩阵，在本文中包含中文和英文两种语言的数据集，分别采用“tencent-ailab-embedding-zh-d200”和“glove.42B.300d”作为预训练词向量模型。

* + 1. 预训练模型

预训练模型是指在大规模无标签语料库上进行预训练，学习到通用的语言知识和表征能力，然后针对具体的下游任务（如文本分类、命名实体识别等），进行有监督的微调。

Bert 是一种基于 Transformer 架构的预训练语言模型，被广泛应用于自然语言处理任务中。在情感分类任务中，Bert 可以利用大规模语料库的上下文信息和语义特征，并且能够针对不同类型的情感分类任务进行微调，从而取得较好的分类效果。具体来说，Bert 首先使用无监督的方式进行预训练，通过掩码语言模型和下一句预测等任务，学习到了丰富的语言知识和表征能力。在情感分类任务中，Bert 能够充分利用文本的上下文和语义信息，对于长文本和复杂情感分类任务有很好的应用效果。此外，还可以通过多层抽取和特征融合等技术，进一步提高分类效果。Bert的网络架构示意图如下：

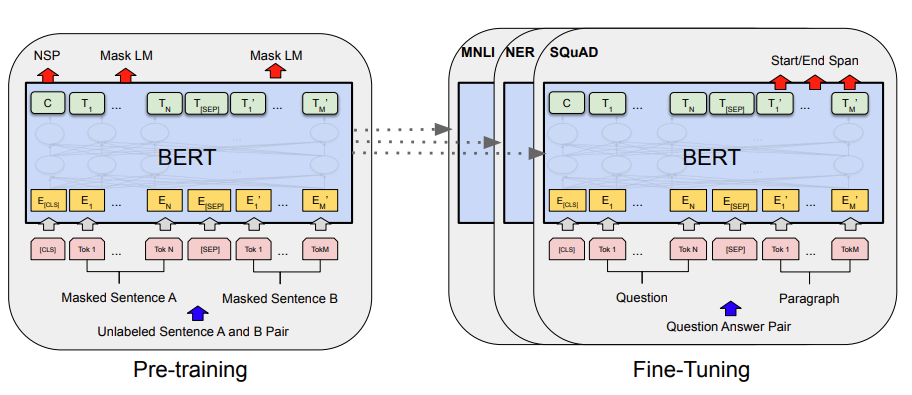


图7 Bert模型示意图

在 Bert 中，每一个文本序列都会被添加特殊标记 [CLS] 和 [SEP]，其中 [CLS] 标记表示整个文本序列的“类别”。在句子级别的情感分类任务中，可以将文本序列作为输入，同时在序列的开头添加 [CLS] 标记，并将该位置的输出向量作为整个句子的表示向量。在本文实验中，我们尝试对不同规模的Bert进行微调，在多个数据集上测试其效果。

* + 1. Prompt Tuning

Prompt 技术是一种基于语言模型的自然语言处理技术，它通过在输入文本中插入提示语（Prompt），来引导模型生成与指定任务相关的输出。Prompt 技术可以帮助解决少样本学习、零样本学习等问题，并且具有较好的可解释性和易用性。其中，Soft Prompt 是一种特殊的 Prompt 技术，它采用软提示方式对模型进行微调。与硬提示不同，Soft Prompt 不会强制指定任务的确切格式和内容，而是使用一组关键词或短语来引导模型生成相关的输出。具体来说，在训练时，Soft Prompt 可以将预定义的提示词或短语加入到输入序列中，作为模型学习的一部分。在推理时，模型将这些提示词或短语视为一种信号或信息，从而更好地完成特定的任务。Soft Prompt 的主要优点是其灵活性和可扩展性。相比硬提示，它能够更准确地匹配输入数据的多样性，从而提高模型的泛化能力。其中较为经典的Soft Prompt方法为P-Tuning v2。与之前的Soft Prompt方法相比，P-Tuning v2 引入了多层Prompt的机制，同时采用了强化学习算法来优化Prompt的生成，提高了模型的适应性和泛化能力。P-Tuning v2的模型示意图如下：

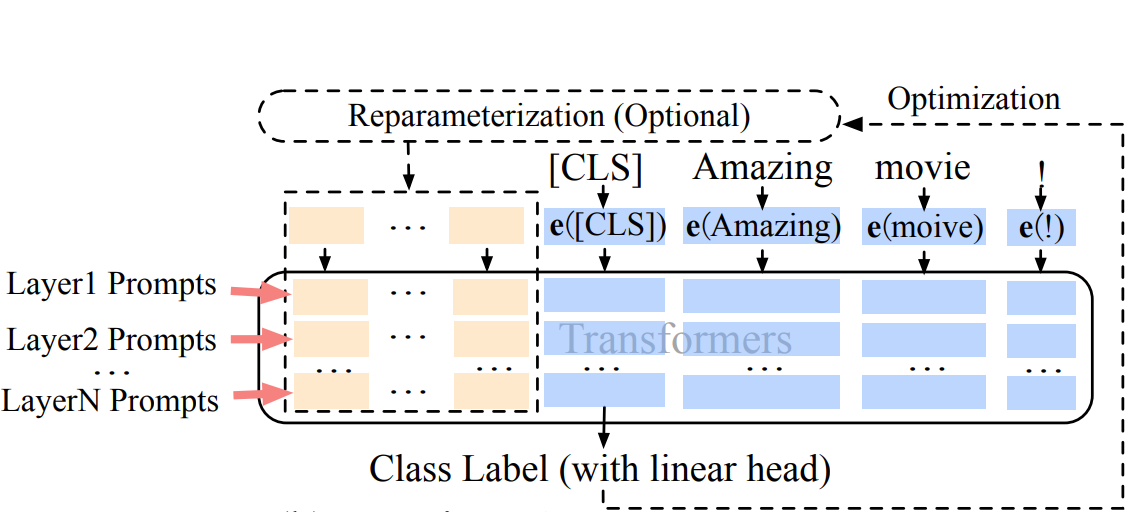


图8 P-Tuning v2的模型示意图

在文本实验中，我们设置Prompt token数为20，并且采用多层感知机（MLP）进行重参数化，提高模型效果。

* 1. 文本情感分析实验

通过传统方法进行文本模态的情感识别的结果如下：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | IMDB | | | | Climate | | | |
| Acc. | Pre. | Rec. | F1 | Acc | Pre. | Rec. | F1 |
| Dict | 0.501 | 0.501 | 0.988 | 0.665 | / | / | / | / |
| Linear | 0.8238 | 0.8246 | 0.8238 | 0.8237 | 0.5522 | 0.5451 | 0.3885 | 0.3813 |
| Logistic Regression | 0.8288 | 0.8292 | 0.8288 | 0.8287 | 0.6680 | 0.6253 | 0.5292 | 0.5549 |
| Naïve Bayes | 0.7910 | 0.7912 | 0.7910 | 0.7909 | 0.6351 | 0.5852 | 0.4831 | 0.5000 |
| Decision Tree | 0.6966 | 0.6967 | 0.6966 | 0.6966 | 0.5659 | 0.4769 | 0.4743 | 0.4754 |
| Random Forest | 0.7784 | 0.7797 | 0.7783 | 0.7781 | 0.6200 | 0.6611 | 0.4758 | 0.4859 |
| SVM | 0.8282 | 0.8286 | 0.8282 | 0.8281 | 0.6604 | 0.6223 | 0.5035 | 0.5256 |
| TextCNN | 0.9000 | 0.9000 | 0.9000 | 0.9000 | 0.7449 | 0.7001 | 0.6835 | 0.6903 |
| BiLSTM | 0.9006 | 0.9009 | 0.9006 | 0.9006 | 0.7454 | 0.7029 | 0.6937 | 0.6972 |
| bert-base | 0.9360 | 0.9373 | 0.9360 | 0.9360 | 0.7899 | 0.7540 | 0.7381 | 0.7446 |
| bert-large | 0.9412 | 0.9412 | 0.9412 | 0.9412 | 0.7909 | 0.7532 | 0.7426 | 0.7472 |
| base-P-Tuning v2 | 0.8904 | 0.8905 | 0.8904 | 0.8904 | 0.7689 | 0.7346 | 0.6890 | 0.7067 |
| large-P-Tuning v2 | 0.9038 | 0.9039 | 0.9038 | 0.9038 | 0.7692 | 0.7272 | 0.7112 | 0.7172 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | WaiMai | | | | IEMOCAP | | | |
| Acc. | Pre. | Rec. | F1 | Acc. | Pre. | Rec. | F1 |
| Dict | 0.6546 | 0.4903 | 0.9083 | 0.6368 | / | / | / | / |
| Linear | 0.7787 | 0.8040 | 0.6875 | 0.7038 | 0.4512 | 0.5530 | 0.4231 | 0.4036 |
| Logistic Regression | 0.7741 | 0.8067 | 0.6778 | 0.6926 | 0.6282 | 0.6265 | 0.6368 | 0.6303 |
| Naïve Bayes | 0.7824 | 0.8161 | 0.6896 | 0.7066 | 0.5884 | 0.6042 | 0.5749 | 0.5861 |
| Decision Tree | 0.7713 | 0.7848 | 0.6819 | 0.6969 | 0.5144 | 0.5037 | 0.5189 | 0.5095 |
| Random Forest | 0.7731 | 0.7939 | 0.6813 | 0.6964 | 0.5090 | 0.5994 | 0.4825 | 0.4690 |
| SVM | 0.7750 | 0.8006 | 0.6819 | 0.6974 | 0.6119 | 0.6102 | 0.6201 | 0.6138 |
| TextCNN | 0.8685 | 0.8527 | 0.8507 | 0.8517 | 0.6264 | 0.6386 | 0.6312 | 0.6338 |
| BiLSTM | 0.8778 | 0.8650 | 0.8576 | 0.8611 | 0.6336 | 0.6322 | 0.6477 | 0.6389 |
| bert-base | 0.9065 | 0.8935 | 0.8972 | 0.8953 | 0.7058 | 0.7034 | 0.7251 | 0.7117 |
| bert-large | / | / | / | / | 0.7238 | 0.7144 | 0.7295 | 0.7207 |
| base-P-Tuning v2 | 0.9046 | 0.8966 | 0.8868 | 0.8914 | 0.6841 | 0.6727 | 0.7042 | 0.6843 |
| large-P-Tuning v2 | / | / | / | / | 0.7238 | 0.7099 | 0.7362 | 0.7203 |

从测试结果可以看出，传统机器学习方法中Logistic Regression和SVM比较适合，其中Logistic Regression对大规模数据有很好的扩展性，SVM的模型泛化能力强，两者在测试集上达到了0.62左右的性能。但深度学习方法可以达到更高的识别精度，最高的精度与机器学习方法相比要高出0.1左右，这是因为深度学习模型可以自动学习文本的语义特征表示，而不再依赖手工提取的特征。相比之下，传统机器方法需要人工选取特征，依赖于人工经验和专业知识。

但是，深度学习模型在其他方面也有不足之处，深度学习模型通常具有较高的模型复杂度，需要大数据集进行训练，计算资源要求也更高，而传统机器学习模型相对简单，计算资源需求较小，且具有更好的解释性。

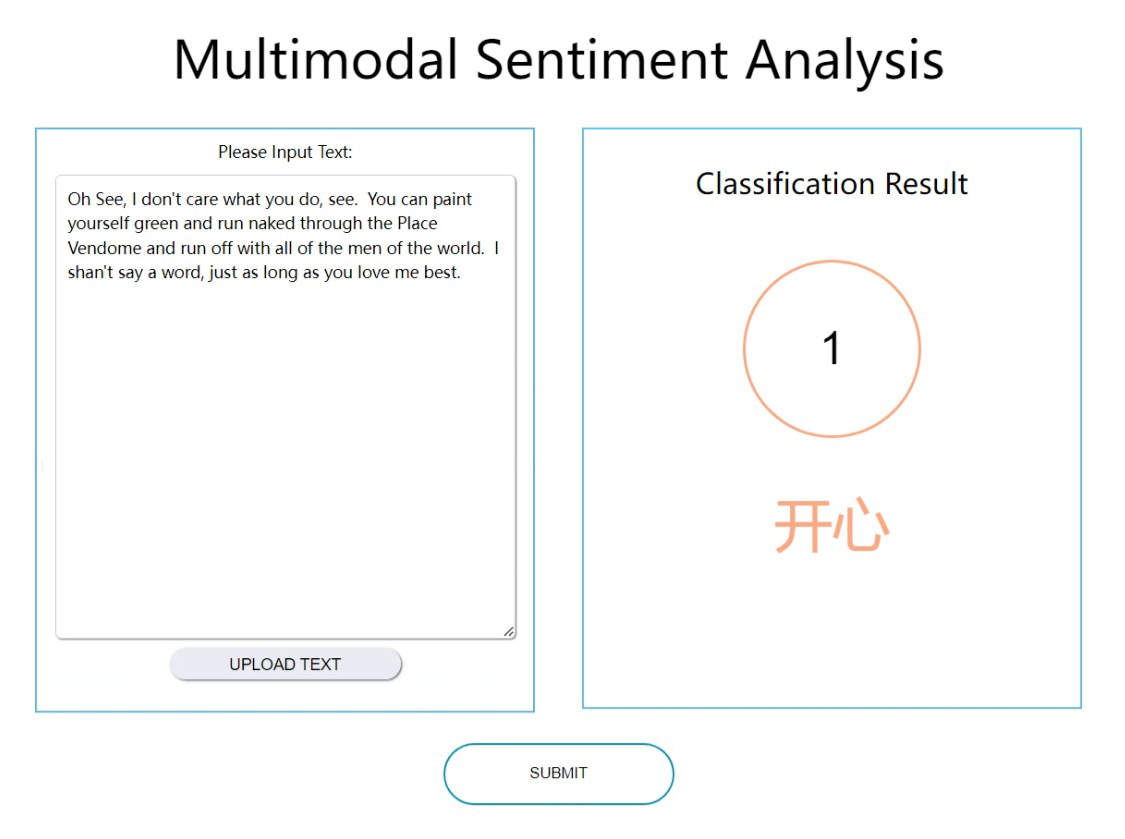
综上所述，深度学习方法在文本情感识别任务上具有更高的识别精度和更好的泛化能力等优势。但其模型较为复杂，需要大量数据和计算资源。相比之下，传统机器学习方法更加简单和可解释，但识别效果有限，且需要专业知识进行特征工程。

1. 文本情感识别系统

在提升算法性能的同时，为了方便查看端到端的情感识别效果，我们实现了一个简单的情感识别系统，支持用户在本地自行输入文本，然后在网页上实时查看情感识别的效果。系统的后端使用Python的Flask框架，前端使用简单的HTML+CSS+Javascript进行搭建。

在后端调用我们训练好的模型进行推理的时候，其中的文本模型我们使用Mindspore框架进行重新部署。Mindspore框架相对于Pytorch框架来说，对CPU推理的支持更好，部署简单且性能优异。部署流程使用Docker进行自动化控制，部署的服务器为华为云2核4G的云耀云服务器，硬盘大小为40G。

系统的效果展示如下：



左侧为文本的输入框，输入文字后点击上传文本，然后点击下方按钮即可以在右侧展示情感分类的结果。其中的粉色圆形内部是多模态推理的数值形式的标签，粉色文字是推理的实际标签。可以看出，对于输入的文本来说，推理效果正确。

1. 参考文献

[1] 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.

[2] Sohn K, Berthelot D, Carlini N, et al. Fixmatch: Simplifying semi-supervised learning with consistency and confidence[J]. Advances in neural information processing systems, 2020, 33: 596-608.

[3] Sun Y, Wang S, Feng S, et al. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation[J]. arXiv preprint arXiv:2107.02137, 2021.

[4] Miyato T, Dai A M, Goodfellow I. Adversarial training methods for semi-supervised text classification[J]. arXiv preprint arXiv:1605.07725, 2016.

[5] Wei C, Sohn K, Mellina C, et al. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 10857-10866.

[6] Wei J, Zou K. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks[C]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. 2019: 6382-6388.

[7] Alammar, J (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/

[8] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language?. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.