福州大学

《机器学习》
科研实训报告

题　　目：基于弱标签信息的深度学习情感分类

学生姓名（学号）：林鑫（031802319）

学生姓名（学号）：苏炜斌（031802326）

指　导　教　师：于元隆、朱丹红

目录

基于弱标签信息的深度学习情感分类

摘要：

情感分类的目的是解决文本情感倾向自动判断的问题。在在线评论的情感分类任务中，传统的深度学习情感分 类模型侧重于算法优化以提高模型的分类性能，但是当手动标记情感倾向的样本数据不足时，模型的分类性能将很差。

一方面，基于弱标签信息的深度学习情感分类模型将弱标签信息引入模型的训练过程中，以减少手动标签数据的使用。另一方面，标记信息薄弱可以在一定程度上代表评论的情绪倾向，但同时也包含杂音，该模型减少了弱标签信息中噪声的负面影响，从而提高了情感分类模型的分类性能。实验结果表明，在大众点评十大热门糖水店的评论的情感分类任务中，基于弱标签信息的深度学习情感分类模型在不增加人工成本的情况下具有优于传统深度模型的分类性能。

一、引言

互联网的发展为用户提供了一个自由发布在线评论的平台。在线评论包括产品评论，电影评论和服务评论。这些评论包含人们想要表达的情感信息。在线评论中的情感信息可以帮助公司改善产品，帮助政府监控公众舆论并为其他用户提供参考。从评论文本中挖掘情感信息需要情感分析技术，而情感分析技术的基本任务之一就是情感分类。自动区分文本的情感倾向的过程是情感分类。

传统的深度学习模型专注于算法优化，以提高模型的分类性能并在情感分类任务中表现良好。但是，基于深度学习的传统情感分类模型的训练过程需要手动缩放情感趋势样本，这将消耗大量的人力资源。

我们将以下将人工标记情感倾向的样本数据定义为标记数据。面对大量评论数据，很难手动标记每个评论样本的情绪倾向。

许多平台的在线评论不仅包含评论文本，还包含其他一些信息，例如乐谱，表情符号，标签等。这些信息可以视为用户对其自己的评论文本的标记。将此信息引入深度学习模型的模型训练可以减少标记数据的使用。

但是，由于某些评论是任意的，因此用户发布评论没有统一的标准。诸如评

分之类的信息标记的情感趋势与用户评论文本（例如具有高分的负面评论文本）中表达的情感不一致。

例如：用户在电子商务平台上购买的 iPhone 11 上发布了否定评论文字，但给出 4 颗星的积极评分这就是所谓的噪音。

由于乐谱，表情符号，标签和其他信息中存在杂音，我们将此信息定义为弱标记信息。包含弱标记信息的样本数据被定义为弱标记数据。将评论文字所表示的弱标签信息与情感倾向不一致的样本定义为噪声样本，并将评论文字所表示的弱标签信息与情绪倾向不一致的样本定义为正确样本。弱标记数据和标记数据之间的区别在于：弱标记数据包含噪声样本，而标记数据不包含噪声样本。

传统研究人员通常将使用弱标记数据等同于使用标记数据，即仅使用弱标记数据来训练情绪分类模型。这解决了标签数据不足的问题，但是由于弱标签数据中存在噪声样本，因此噪声样本将在模型 训练期间对模型产生负面影响，并降低模型的分类性能。因此，要减少噪声样本对模型的负面影响，同时将弱标签信息引入模型训练中，是很难实现的。

从所有样本中随机提取少量的样本，以手动标记情感倾向，并获得由剩余样本组成的少量标记数据和大量弱标记数据。提出了下面这个方法来训练使用标签数据和弱标签数据的情感分类模型，以减少弱标签数据中的噪声样本对模型的负面影响，从而提高模型的分类性能

方法思路： 情感分类模型的训练分为两个阶段。在训练的第一阶段， 使用大量的弱标记数据来训练模型，然后在第二阶段使用一些标记数据来继续训练模型以微调模型。

在引入弱标签信息参与模型训练的过程中提出的两种方法可以有效减少弱标

签信息中包含的噪声对情感分类模型的负面影响，从而提高模型的情感分类性能。

二、相关工作

目前，情感分类主要分为两个研究方向：基于字典的情感分类和基于机器学习的情感分类[1-3]。参考文献[4-7]各自提出了一个新的情感词典。实验表明，情感分类任务的分类性能优于传统的情感词典，基于字典的情感分类方法的分类性能优异。但是，情感词典是取决于域的。一旦情感分类任务的应用领域发生变化，基于情感词典的分类模型的分类性能就会下降。参考文献[8]提出使用多域数据创建情感字典来解决情感字典的域依赖问题。尽管情感词典的域依存性有所减弱，但基于词典的情感分类模型最大的问题是情感词典的构建需要大量的人为参与，而且随着网络数据的爆炸性增长，难以解决未知问题。通过手动在情感词典中的单词。

参考文献[9]提出使用机器学习技术来完成情感分类的任务。实验表明，基于机器学习的情感分类模型具有很好的分类性能。参考文献[10-12]各自提出了一种新的机器学习算法来完成情感分类任务并实现了较高的分类性能。与基于字典的情感分类模型相比，基于机器学习的情感分类模型避免了单词未知的问题，但是传统的机器学习算法的特征工程仍然需要较高的人工成本。作为机器学习的一个分支，深度学习在以下领域得到了迅速发展。

近年来，在诸如在线评论之类的大型文本的情感分类任务中表现出 色。与机器学习技术相比，深度学习技术极大地降低了人工成本。Word2vec，ELMO [13]，BERT [14]促进了深度学习在情感分类任务中的应用。参考文献[15-17]为情感分类提出了创新的深度学习模型 。实验表明，分类性能优于传统的深度学习模型。但是，这些模型使用的大多数数据集都是用于人工标记情感倾向的标准数据集，

而缺乏用于深度学习模型的监督训练的大规模标记数据是深度学习模型的应用瓶颈。

三、弱标记数据

在深度学习模型的训练过程中引入弱标签数据可以解决训练数据不足的问题。目前，关于弱标记数据的研究相对较少。

在 Twitter 挖掘中，参考文献[18]使用半监督分类对症状监测中的相关性进行过滤，并在任务中引入了带有弱标记数据的数据，以完成分类任务。参考文献[19]已经标记了 160 万个 Twitter 文本，并建立了表情符号情感词典。实验验证了表情符号（即弱标记信息）在情感分类任务中的有效性。参考文献[20]提出了两个大数据系统来使用表情符号来完成 Twitter 文本的情感分类任务。实验证明，所提出的两个系统的准确性和鲁棒性都非常好。参考文献[21]在深度学习模型的训练中引入了弱标记数据，并在情感分类任务中实现了出色的分类性能。在情感分类任务中，这些研究人员都使用弱标签数据等效于标签数据，但忽略了弱标签数据中包含的噪声特征。

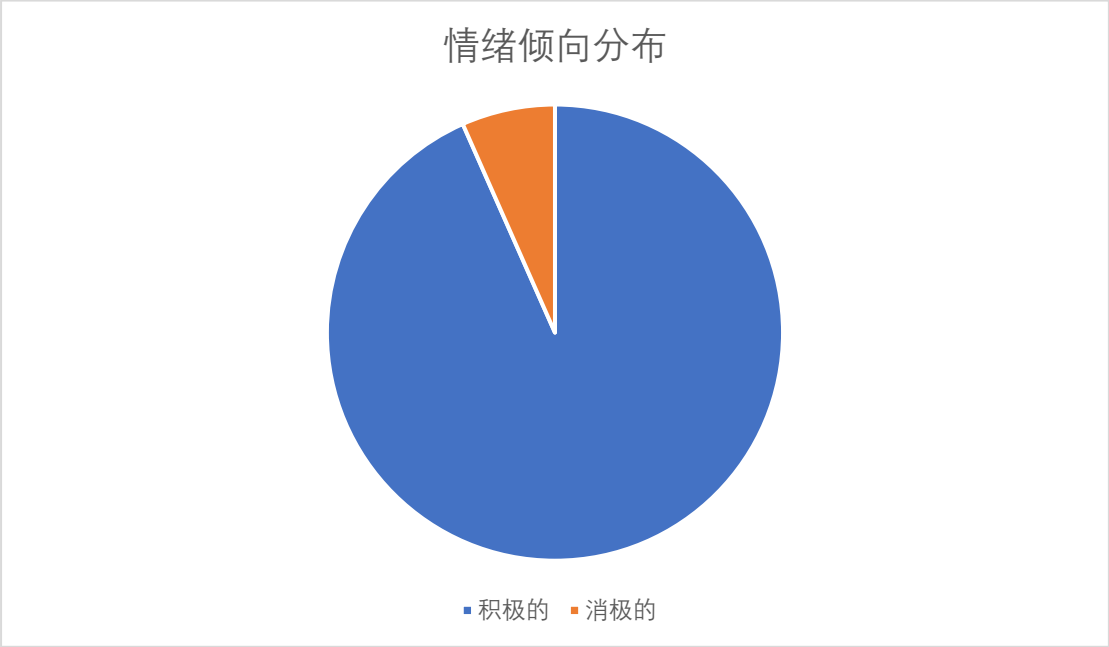
因此，在使用弱标记数据的同时，要减少弱标记数据中包含的噪声样本对情感分类模型的负面影响，从而提高情感分类模型的分类性能。

四、实验内容

4.1. 数据清理

本文中使用的示例数据是大众点评十大热门糖水店的评论，爬取网页后从 html 页面中把需要的字段信息（顾客 id、评论时间、评分、评论内容、口味、环境、服务、店铺 ID）提取出来并存储到 data.csv （原始数据集）中，使用的弱标记信息是用户在发布评论时给店铺的分数（1-5 分），根据评分系统，每条评论的

总分为 5。得分高于 3 分的评论被归 为积极情绪倾向，得分低于 3 分的评论被归为负面情绪倾向。总共抓取了 32484 个在线评论，根据弱标记信息划分所有数据后的情感趋势分布如图



评论数据通常包含一些信息，这些信息对后续的情感分类任务没有帮助。使用与数据清理相关的技术可以提高数据质量，从而提高后续情感分类模型的分类性能。具体的数据清理步骤如下：

（1）删除特殊标记：由于用户在网站上发布自己的评论时并没有统一的规则，因此他们经常在文本评论中包含一些与样本的情感倾向无关的特殊标记。

（2）分词：用户发布的中文在线评论以单词序列的形式出现。分词是指将每个在线评论的单词序列分为多个单独的单词。

（3）删除停用词：停用词是指对整个在线评论没有语义意义的词，例如感叹词，代词等。删除停用词和删除特殊符号具有相同的效果，这有助于随后的情感分类模型可以更好地捕捉句子的主要语义。

（4）删除低频词：由于互联网用户数量众多，每个人都有各种各样的词来表达

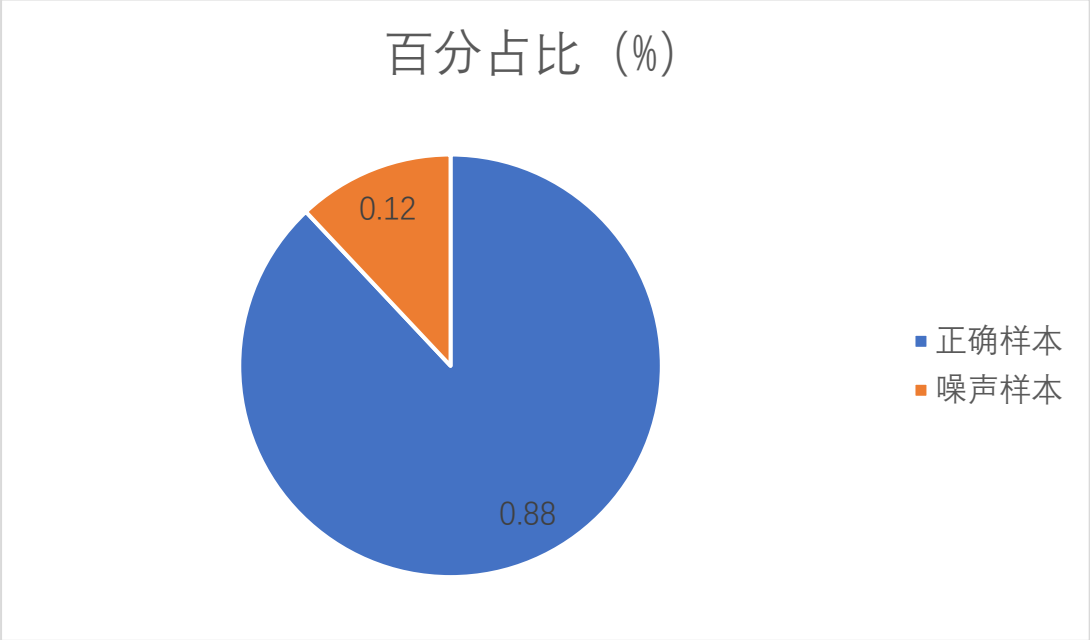
自己的情感。尽管有些词可以代表一定的情感倾向，但它们的频率太低。对于整个情感分类模型而言，学习低频词的情感倾向弊大于利。

（5）删除空白评论。

最后，将数据存储以方便后续情感分类模型的使用。

4.2 标记数据

在实验中，首先从所有样本中随机提取少量样本，以手动标记情感倾向。下图显示了在手动标记样本的情感趋势之前，噪声样本和正确样本的分布。



实验中总共对 200 个样本进行了标记，其中 24 个噪声样本占 12.0%。弱标签数据中的噪声样本比例不是很大，因此在大多数情况下，弱标签数据可以代表用户的真实情感倾向。但是，这部分噪声样本仍然会对情感分类模型产生很大的负面影响。因此，如果要进一步提高情感分类模型的分类性能，就不能忽略噪声样本的存在。

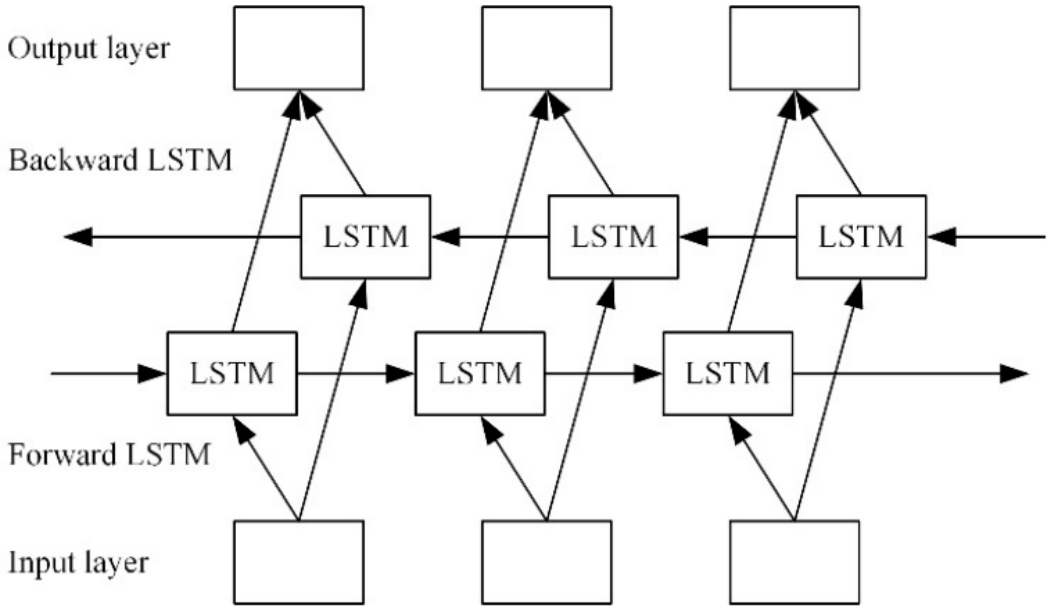
4.3 基于 BiLSTM 的情感分类模型

长短期记忆（LSTM）[22]神经网络是一种引入“门”机制的递归神经网络。LSTM

神经网络可以捕获更长距离的语义依赖性，避免了由于序列长而导致的传统递归神经网络梯度消失的问题。统计数据显示，大多数评论样本包含较少的单词，并且 50 个单词以内的样本数量占总数的 96.5%。因此，LSTM 适用于大众点评十大热门糖水店的评论的情感分类任务。

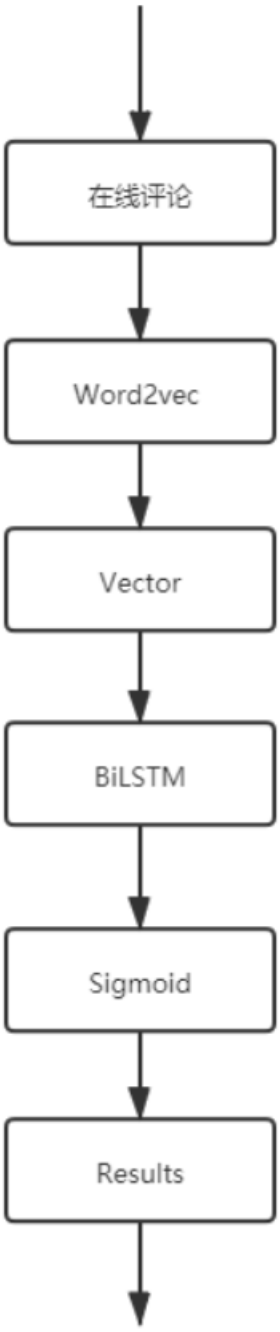
LSTM 可以从整体上理解文本语义，并且在情感分类任务中表现良好[23]。但是，LSTM 只能捕获单个方向的语义依赖性。

双向长期短时记忆（BiLSTM）[24]神经网络是一种改进的 LSTM 神经网络，可以捕获双向长距离语义依赖性。因此，在实验中，BiLSTM 神经网络主要用作深度学习情绪分类模型的基本组成部分，以获得优异的分类性能。下图显示了 BiLSTM 神经网络的结构：



4.4 Word2vec

Word2vec 用于将审阅数据的文本信息转换为矢量表示，然后将其输入到 BiLSTM 中。最终的样本情感分类结果是通过将 BiLSTM 传递到 Sigmoid 层获得的。具体的情感分类模型如下图所示：



4.5 基于两阶段训练的 BilSTM 情感分类模型

基于两阶段训练的两阶段训练的 BiLSTM 情感分类模型是第一个基于弱标记信息的深度学习情感分类模型。实验中使用的情感分类模型的结构与基于 BiLSTM（A_BiLSTM）的情感分类模型一致，并且仅在模型训练方法上进行了创新。模型的训练分为两个阶段，两阶段训练方法如下:

首先，在第一阶段，使用大量的弱标记数据来训练 A_BiLSTM （模型 0)。弱标签数据中的噪声样本所占的比例很小。因此，在使用大量弱标记数据训练模型 0 之后,我们可以得到情感分类模型(模型 1),该模型可以很好地捕获评论语义。

之后，将模型 1 的参数作为第二阶段模型（模型 2）训练的初始参数，并将一些标签数据输入模型进行训练，以达到对模型参数的微调目的。减少模型训练第一阶段输入的噪声样本对情感分类模型的负面影响，提高模型的分类性能，并获得最终的情感分类模型（模型 3)。

基于两阶段训练的 BiLSTM 情感分类模型训练过程
第一阶段训练的 A_BiLSTM： **输入： 数据清理后的大量弱标记数据** **开始：** **（1）A_BiLSTM 中的 Word2vec 将评论文本转换为矢量表示形式** **（2）在 A_BiLSTM 模型收敛后保存模型参数。** **输出 ： 保存已完成训练的第一阶段的 A_BiLSTM 模型（模型 1）**
第二阶段训练的 A_BiLSTM **输入： 数据清理后的一些标记数据** **开始：** **（1）加载模型 1 的参数以得到模型 2。** **（2）SC_BiLSTM 中的 Word2vec 将评论文本转换为矢量表示形式。** **（3）在 SC_BiLSTMmodel 再次收敛后保存模型参数。** **输出 ： 保存已完成训练的第二阶段的 A_BiLSTM 模型（模型 3）**

在实验中我们取使用欠采样方法消除正负样本不平衡后的数据集中的 8789 条弱标记数据（包含噪声样本）和标记数据（不包含噪声样本）作为实验数据，经过数据处理等步骤后训练模型，得到最终的 A_BiLSTM 情感分类模型。

五、实验结果

5.1 分词结果

第一阶段

	stars	cus_comment
0	0	超级 恶心 打包 了 三份 回家 吃 到 第二份 的 时候 直接 吃 出 不 知道 是 什么...
1	0	服务态度 特别 差 尤其 是 那个 貌似 是 经理 的 女 的 赶客 态度 差到 不得了
2	0	南信 是 老牌 糖水 店 出品 虽然 无功 无过 但是 服务 实在 是 欠佳 点单 一碗 姜...
3	0	差 服务 差 以后 都 不来 吃 所以 的 管理员 不可一世
4	0	很多年 前 就 吃 过 南信 的 不少 出品 以前 一直 对南信 也 算是 百般 认可 都 ...
...	...	...
8783	1	这家 店 生意 也 太好了 吧 前后 去 了 两次 第一次 是 饭点 没 位子 人满为患 而...
8784	1	双皮奶 很 不错 牛 三星 我 不 知道 是 我 没点 好 还是 没 做好 心理 落差 极大
8785	1	点 了 招牌 的 双皮奶 等 了 大概 有 半个 小时 味道 很 好吃 价格 还 行 就是 ...
8786	1	上下 九 简直 就系 广州 老 西关 饮食 一条街 不够 而家 都 系 外地人 过来 帮衬 ...
8787	1	双皮奶 细比文字 的 强太多 甜品 都 不错 不过 红豆 的 有些 过甜 了 艇 仔粥 也 ...

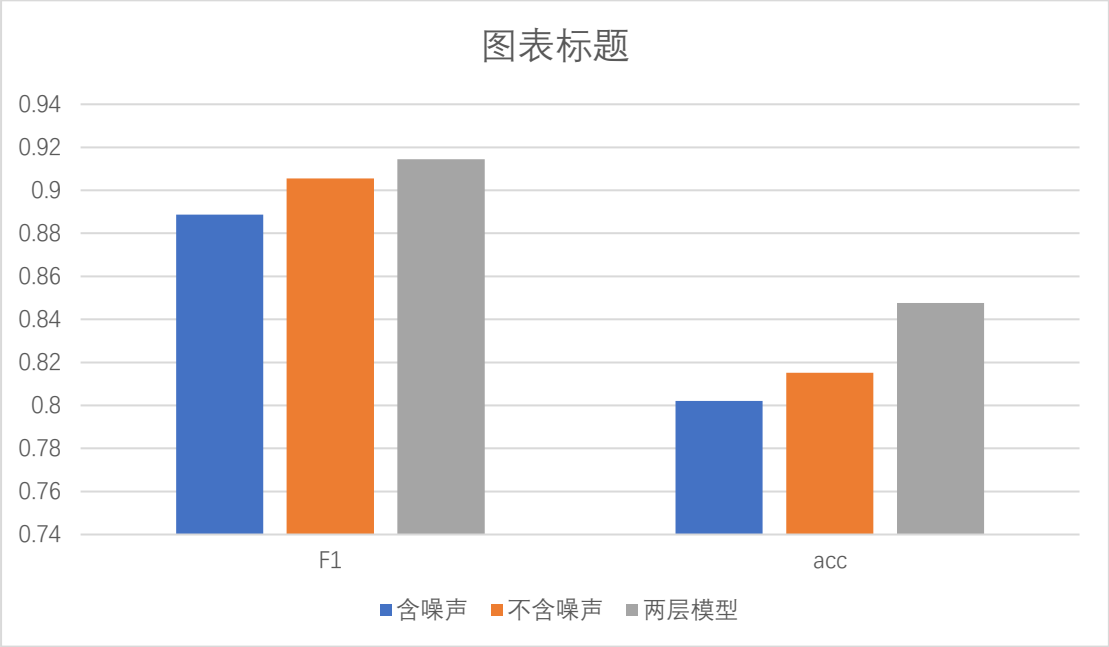
第二阶段

	label	cus_comment
0	0	很久没 见 过 只 收 现金 的 店 我 问 他 没有 现金 怎么办 店员 就 一副 爱 吃...
1	0	规矩 贴满 店铺 看 都 看 不 过来 什么 不能 外带 椰子 盅 不能 随意 搭配 配料 ...
2	0	点 了 一碗 芒果 酱 奶糊酱 是 便宜 酱 没有 芒果 奶糊加 了 十斤 糖 吃 完后 深...
3	0	味道 一般 但是 价格 真的 算贵 的 了 我 觉得 比该 有 的 价格 高 一倍 了 的 ...
4	0	写 评论 之前 翻看 了 百花 的 所有 点评 恶评 如流 证明 它 的 服务质量 同 口味...
...	...	...
8314	1	这家 店 生意 也 太好了 吧 前后 去 了 两次 第一次 是 饭点 没 位子 人满为患 而...
8315	1	双皮奶 很 不错 牛 三星 我 不 知道 是 我 没点 好 还是 没 做好 心理 落差 极大
8316	1	点 了 招牌 的 双皮奶 等 了 大概 有 半个 小时 味道 很 好吃 价格 还 行 就是 ...
8317	1	上下 九 简直 就系 广州 老 西关 饮食 一条街 不够 而家 都 系 外地人 过来 帮衬 ...
8318	1	双皮奶 细比文字 的 强太多 甜品 都 不错 不过 红豆 的 有些 过甜 了 艇 仔粥 也 ...

5.2 词向量转换结果

	0	1	2	3	4	5	6	7	...	93	94	95	96	97	98	99	
0	0	0.987618	0.427896	0.055643	0.095133	0.058742	0.856806	0.077551	...	0.067299	0.017914	0.119739	0.123184	0.030294	0.117672	0.988299	0.87
1	0	0.006949	0.495237	0.981364	0.984653	0.106757	0.833881	0.066228	...	0.126630	0.170799	0.173342	0.934567	0.132481	0.027565	0.996146	0.88
2	0	0.988173	0.415809	0.117598	0.082517	0.061089	0.829436	0.027887	...	0.046439	0.057602	0.153045	0.097469	0.082987	0.093078	0.985457	0.87
3	0	0.044487	0.366637	0.968532	0.999444	0.143709	0.846701	0.967755	...	0.164023	0.195173	0.199092	0.985414	0.122097	0.032093	0.994510	0.84
4	0	0.990575	0.296077	0.079143	0.155601	0.994550	0.821766	0.145198	...	0.020867	0.151583	0.247809	0.073126	0.041233	0.944938	0.979004	0.84
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
8781	1	0.960821	0.385861	0.098360	0.111935	0.029488	0.828995	0.082584	...	0.046520	0.108050	0.179778	0.081009	0.055907	0.991266	0.984135	0.86
8782	1	0.994799	0.321537	0.122288	0.146125	0.003789	0.805220	0.211507	...	0.085758	0.117638	0.199060	0.985903	0.146978	0.970676	0.981352	0.88
8783	1	0.961120	0.471055	0.144736	0.180701	0.004739	0.841746	0.115017	...	0.111474	0.224394	0.180368	0.121844	0.022034	0.120428	0.943413	0.87
8784	1	0.958725	0.286335	0.998698	0.965299	0.992660	0.772663	0.071164	...	0.980055	0.079114	0.237013	0.099458	0.211309	0.990090	0.926617	0.91
8785	1	0.955631	0.299363	0.104877	0.160735	0.003388	0.810455	0.263855	...	0.020150	0.122029	0.264383	0.093372	0.019431	0.926621	0.990883	0.85

5.3 训练结果



含噪声：指用弱标记数据（包含噪声样本）BilSTM 模型上训练

不含噪声：指用标记数据（不包含噪声样本）BilSTM 模型上训练

由此可见本实验设计的两阶段训练的方法，能够有效 BilSTM 情感分类模型的准确率

参考文献

[1] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao, "Targetdependent Twitter sentiment Classification", Proc. 49th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol, vol. 1, pp. 151-160, Jun. 2011.
[2] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, "NRC-Canada-2014: Detecting aspects and sentiment in customer reviews", Proc. 8th Int. Workshop Semantic Eval. (SemEval), pp. 437-442, 2014.
[3] D. T. Vo, Y. Zhang, "Target-dependent twitter sentiment classification with rich automatic features", Proc. IJCAI, pp. 1347-1353, 2015.
[4] C. S. Khoo, S. B. Johnkhan, "Lexicon-based sentiment analysis: Comparative evaluation of six sentiment lexicons," Journal of Information Science, vol. 44, no. 4, pp. 491-511, 2018.
[5] N. Rizun, Y. Taranenko, and W. Waloszek, "Improving the accuracy in sentiment classification in the light of modelling the latent semantic relations," Information, vol. 9, no. 12, pp. 307, 2018.
[6] MZ. Asghar, A. Khan, S. Ahmad, M. Qasim, IA. Khan “Lexicon-enhanced sentiment analysis framework using rulebased classification scheme,” PLoS ONE, vol. 12, no. 2, pp. e0171649, 2017.
[7] A. Al-Saffar, S. Awang, H. Tao, N. Omar, W. Al-Saiagh, M. Al-bared, “Malay sentiment analysis based on combined classification approaches and Senti-lexicon algorithm,” PLoS ONE, vol. 13, no. 4, pp. e0194852, 2018.
[8] Jha V, Savitha R, Shenoy P D, et al. A novel sentiment aware dictionary for multi-domain sentiment classification[J]. Computers & Electrical Engineering, 2018, 69: 585-597.
[9] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment Classification using Machine Learning Techniques," Empirical Methods in Natural Language Processing, vol. 10, pp. 79-86, 2002.
[10] Y. Wang, “Iteration-based naive Bayes sentiment classification of microblog multimedia posts considering emoticon attributes,” Multimedia Tools and Applications, pp. 1-16, 2020.
[11] S. N. Alyami and Olatunji S O, “Application of Support Vector Machine for Arabic Sentiment Classification Using Twitter-Based Dataset,” Journal of Information & Knowledge Management, vol. 19, no. 01, pp. 2040018, 2020.
[12] F. Xu, Z. Pan and R. Xia, “E-commerce product review sentiment classification based on a naïve Bayes continuous learning framework,” Information Processing & Management, pp. 102221, 2020.
[13] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” arXiv preprint arXiv. pp. 1802.05365, 2018.
[14] J. Devlin, MW. Chang, K. Lee, and K. Toutanova, "Bert: Pretraining of deep bidirectional transformers for language understanding," arXiv preprint arXiv, pp. 1810.04805, 2018.
[15] R. Fei, Q. Yao and Y. Zhu, “Deep Learning Structure for Cross-Domain Sentiment Classification Based on Improved Cross Entropy and Weight,” Scientific Programming, 2020.

[16] M. Wang, Z. H. Ning, and T. Li, “Information geometry enhanced fuzzy deep belief networks for sentiment classification,” International Journal of Machine Learning and Cybernetics, vol. 10, no. 11, pp. 3031-3042, 2019.
[17] W. Li, P. Liu, and Q. Zhang, “An Improved Approach for Text Sentiment Classification Based on a Deep Neural Network via a Sentiment Attention Mechanism,” Future Internet, vol. 11, no. 4, pp. 96, 2019.
[18] O. Edo-Osagie, G. Smith, I. Lake, O. Edeghere, La. De, B. Iglesia, “Twitter mining using semi-supervised classification for relevance filtering in syndromic surveillance,” PLoS ONE vol.14, no. 7, pp. e0210689, 2019.
[19] P. K. Novak, J. Smailović and B. Sluban, “Sentiment of emojis,” PloS one, vol. 10, no. 12, pp. e0144296, 2015.
[20] A. Kanavos, N. Nodarakis and S. Sioutas, “Large scale implementations for twitter sentiment classification,” Algorithms, vol. 10, no. 1, pp. 33, 2017.
[21] Z. Jianqiang, G. Xiaolin and Z.Xuejun, “Deep convolution neural networks for twitter sentiment analysis,” IEEE Access, vol.6, pp. 23253-23260, 2018.
[22] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997.
[23] X. Zhu, P. Sobihani and H. Guo, “Long short-term memory over recursive structures,” International Conference on Machine Learning, PMLR, pp. 1604-1612, 2015.
[24] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," in IEEE Transactions on Signal Processing,

附录（部分代码）

Setting.py

```
# 文件索引
file_nolabeled = './data/data_no.csv'
file_labeled = './data/data_la.csv'
model_address = 'data/w2v_model_50.pkl'
result_address = 'data/vector_data.csv'
# 目标列名
col_nolabeled = 'stars'
col_labeled = 'label'
col_comment = 'cus_comment'
```

w2vmodel.py

```
import pandas as pd
import numpy as np
from gensim.models.word2vec import Word2Vec
from gensim.models.word2vec import Text8Corpus
from WordVector.cut_sentence import cut_sentence_cn

from settings import file_nolabeled, col_nolabeled, col_comment , model_address

# 构建 word2vec 模型，词向量的训练与生成
def get_dataset_vec(dataset):
    data_list=[]
    for i in range(len(dataset)):
        data_list.append(str(dataset[i]).split())
    n_dim = 100
    w2v_model = Word2Vec(data_list, sg=1, vector_size=n_dim, min_count=5, hs=0)  # 初始化模型并训练
    w2v_model.save(model_address)  # 保存训练结果


if __name__ == '__main__':
    df = pd.read_csv(file_nolabeled)
    df = df[[col_comment, col_nolabeled]]

    cw=lambda x:cut_sentence_cn(x)

```

```
    # #查看分词结果
    df[col_comment].apply(cw).to_csv('./data/temp.csv')

    data = np.array(df[col_comment].apply(cw))

    get_dataset_vec(data)
```

cut_sentence.py

```
import jieba
import re

pattern = re.compile(r'[]，。！“”：,.!~@#*？?&;":(-+]()')

def cut_sentence_cn(document):
    return document
    # 删除特殊符号
    document = re.sub(pattern,'',document)

    # 分词
    document_cut = jieba.cut(document)
    stopwords = stopwordslist('./WordVector/stopword.txt')
    outstr = '' #设置一个空的字符串，用于储存结巴分词后的句子
    for word in document_cut: #遍历分词后的每一个单词
        if word not in stopwords: #如果这个单词不在停用表里面
            if word != '\t': #且这个单词不是制表符
                outstr += word #就将这个词添加到结果中
                outstr += "     " #但是这里为什么又要添加双引号中间带空格？
                                  #测试了一下，原来是为了让结巴分词后的词以空格间隔分割开
    if len(outstr)<=5:
        return ""
    return outstr


# 创建停用词 list 函数
def stopwordslist(filepath):
    stopwords = [line.strip() for line in open(filepath, 'r', encoding='utf-8').readlines()] #分别读取停用词表里的每一个词，

#因为停用词表里的布局是一个词一行
    return stopwords #返回一个列表，里面的元素是一个个的停用词
```

sentence_vector.py

```
import numpy as np
import math


#对每个句子的所有词向量取均值，来生成一个句子的 vector
def build_sentence_vector(sentence,size,w2v_model):
    sen_vec=np.zeros(size).reshape((1,size))
    count=0
    for word in sentence:
        try:
            sen_vec+=w2v_model.wv[word].reshape((1,size))
            count+=1
        except KeyError:
            continue
    if count!=0:
        sen_vec/=count
    return sen_vec
```

doc_vector.py

```
from typing import Text
import pandas as pd
import numpy as np
from gensim.models.word2vec import Word2Vec
from WordVector.sentence_vector import build_sentence_vector
from WordVector.cut_sentence import cut_sentence_cn

from settings import model_address, result_address, col_comment,file_nolabeled

# 将文本数据转换为文本向量
def doc_vec(file_address, target_name):
    data = pd.read_csv(file_address)
    w2v_model = Word2Vec.load(model_address)      #加载训练好的 Word2Vec 模型

    #读取词权重字典
    #with open('data/key_words_importance', 'r') as f:
        #key_words_importance = eval(f.read())

    #数据集处理
    data = data[[target_name,col_comment]]
    data = data.dropna()
    data[target_name] = data[target_name].apply(lambda x:int(((float(x))/3)))
```

```
    data[col_comment] = data[col_comment].apply(lambda x:cut_sentence_cn(x))

    data = data.dropna()
    data = data [data[target_name]!=3]
    print(data)
    data_y = np.array(data[target_name])
    text_list = np.array(data[col_comment])

    #训练集转换为向量
    data_list=[]
    for i in range(len(text_list)):
            data_list.append(str(text_list[i]).split())


    data_x=np.concatenate([build_sentence_vector(sen,100,w2v_model) for sen in data_list])
    for i in np.nditer(data_x, op_flags=['readwrite']):
        for x in np.nditer(i, op_flags=['readwrite']):
            if x<0:
                x *= 0.5
                x += 1
    return data_x,data_y

if __name__ == '__main__':
    data_x,data_y = doc_vec(file_nolabeled,'stars')
    df = pd.concat([pd.DataFrame(data_y), pd.DataFrame(data_x)],axis=1,ignore_index=True)
    print(df)
    df.to_csv(result_address)
```

BILSTM.py

```
import numpy as np
from numpy import random
import pandas as pd
from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers import Dense,Dropout,Embedding,LSTM,Bidirectional
from keras.datasets import imdb
from sklearn.model_selection import train_test_split
from doc_vector import doc_vec
from F1 import G_mean,f1_socre,specificity,accuracy,recall,cal_base,precision
from imblearn.over_sampling import SMOTE
from keras import regularizers
my_metrics=[f1_socre,accuracy]
```

```
from settings import file_nolabeled, col_nolabeled, file_labeled, col_labeled

# 设置最大特征的数量，对于文本，就是处理的最大单词数量。若被设置为整数，则被限制为待处理数据集中最常见的 max_features 个单词
max_features=20000
# 设置每个文本序列的最大长度，当序列的长度小于 maxlen 时，将用 0 来进行填充，当序列的长度大于 maxlen 时，则进行截断
maxlen=100
# 设置训练的轮次
batch_size=32
import matplotlib.pyplot as plt
def show_train_history(train_history,train,validation):
    plt.plot(train_history.history[train])
    plt.plot(train_history.history[validation])
    plt.title('Train History')
    plt.ylabel(train)
    plt.xlabel('Epoch')
    plt.legend(['train', 'validation'], loc='upper left')
    plt.show()

def dealing_data(data_x, data_y, rate):
    train_x,test_x,train_y,test_y = train_test_split(data_x,data_y,test_size=rate, random_state=5)
    # 查看数据大小
    print(len(train_x),'train sequences')
    print(len(test_x),'test sequences')

    print('Pad sequences (samples x time)')
    # 将文本序列处理成长度相同的序列
    train_x = sequence.pad_sequences(train_x, maxlen=maxlen)
    test_x = sequence.pad_sequences(test_x, maxlen=maxlen)
    print('x_train shape:', train_x.shape)
    print('x_test shape:', test_x.shape)
    train_y = np.array(train_y)
    test_y = np.array(test_y)

    return train_x,test_x,train_y,test_y


# 创建网络结构
model=Sequential()
model.add(Embedding(max_features,100,input_length=maxlen))
model.add(Bidirectional(LSTM(64)))
# model.add(Bidirectional(LSTM(64,return_sequences=False)))

```

```
model.add(Dropout(0.5))
model.add(Dense(1,activation='sigmoid'))

# 编译模型
model.compile('adam','binary_crossentropy',metrics=my_metrics)

# 加载弱标记数据
print("loading weak labeled data ...")
data_x,data_y = doc_vec(file_nolabeled, col_nolabeled)
train_x,test_x,train_y,test_y = dealing_data(data_x, data_y, 0.3)

# 第一次训练模型
print('1st Train...')
train_history = model.fit(train_x, train_y,batch_size=batch_size,epochs=4,validation_split=0.3,)

# 加载标记数据
print("loading labeled data ...")
data_x ,data_y = doc_vec(file_labeled, col_labeled)
train_x,test_x,train_y,test_y = dealing_data(data_x, data_y, 0.3)
show_train_history(train_history,'accuracy','val_accuracy')   #查看精度变化
show_train_history(train_history,'f1_socre','val_f1_socre')   #查看精度变化

# 第二次训练模型
print('2nd Train...')
train_history = model.fit(train_x, train_y,batch_size=batch_size,epochs=4,validation_split=0.3,)
show_train_history(train_history,'accuracy','val_accuracy')   #查看精度变化
show_train_history(train_history,'f1_socre','val_f1_socre')   #查看精度变化
```