**机 器 学 习 理 论**

**Joint Embedding of Words and Labels for Text Classification**

**阅读报告**

学号： 18212010048

姓名： 曾捷航

**一、问题背景**

本章主要讲述文本分类问题的问题背景以及针对此问题的研究现状。

1. **文本分类**

分类问题是机器学习领域非常重要的一个组成部分。分类问题的目标是根据已知样本的某些特征，通过计算选择特征参数，判断一个新的样本属于哪种已知的样本类。分类问题是常见的监督式学习。因为训练集的每个样本既包含数据，又包含该数据所属于的类别(标签，label)，所以称为监督式学习。

文本分类是分类问题在自然语言处理领域的一个分支。在文本分类问题下，每个样本的输入是一段文本，可以是非常长的序列如文章，也可是非常短的序列如一个短语；输出是这段文本所属的类别(label)。文本分类问题旨在提供有限训练集样本的情况下，学习一个映射函数F,使得F(X)可以接近于样本的类别。文本分类问题的数学定义如下所示：

给定一个每个元素是一对(输入，标签)对的训练集 ,其中是一个长度为L的文本序列，包含L个单词，是相应的标签。在单标签问题下(每个输入只能分类到一种类别)，是一个one hot向量(向量中只有一个元素是1，其他全为0)；在多标签问题下(每个输入可以分类到多种类别)，是一个二进制向量(向量只包含0,1)。文本分类问题的目标是学习一个函数,通过最小化以下的损失函数：

其中是损失函数，用于衡量函数的预测结果与真实标签的区别。常见的损失函数包括平方误差损失 [[1]](#footnote-1)，交叉熵损失 [[2]](#footnote-2) 等。本论文使用的是交叉熵损失，具体公式会在以下模块介绍。

1. **研究现状**

**① 传统方法**

1. 频次法

频次法记录每篇文章每个词的次数分布，然后将分布输入到机器学习模型，训练一个合适的分类模型，对这类数据进行分类。在统计次数分布时，可合理提出假设，频次比较小的词对文章分类的影响比较少，因此可以合理地设置阙值，滤除频次小于阙值的词，减少特征空间维度。

1. TF-IDF

TF-IDF认为词出现的次数能从一定程序反应文章的特点，即TF。如果一个词在某个类别上出现的次数多，而在全部文本上出现的次数相对比较少，这个词就有强大的文档区分能力，也即反文档频率。TF-IDF综合考虑频次和反文档频率两个因素。

1. 互信息方法

互信息方法是一种基于统计的方法，计算文档中出现词和文档类别的相关程序，即互信息。

1. N-Gram

基于N-Gram的方法就是把文章序列，通过大小为N的窗口，然后对这些Group，然后对这些Group做统计，滤除出现频次较低的Group，把这些Group组成特征空间，传入分类器进行分类。

**② 深度学习方法**

深度学习方法的基础是向量建模，神经网络的输入一般是连续值的向量，因此在将深度学习方法应用到文本分类时，需要先把一个文本转换成向量。常见的方法是使用预训练好的词向量word embedding，对于词典中的每个词都有一个相应的向量embedding，随后将文本数据集里的每个词都表征成向量表示。常见的训练词向量word embedding的方法有：word2vec[1]、fasttext[2]等。

1. CNN

首次将CNN(卷积神经网络)应用到文本分类是Kim在2014年发的文章[3],文章首先通过预训练好的word embedding将文本转换成向量，然后使用不同filter的cnn网络，加入maxpool层，最后再加入全连接层得到最终的输出结果。整个过程的图示如下图1所示：

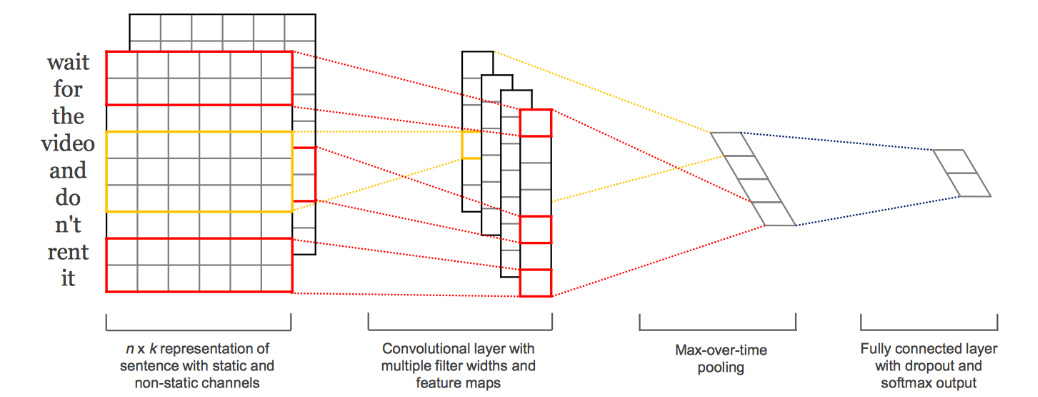


图1：CNN for文本分类

后续基于cnn的文本分类方法基本上文献[3]的改进和增强，如基于字符级别的cnn(charCNN[4])等。

1. RNN(LSTM)

循环神经网络可以很方便的建模序列数据，而文本就是一种常见的序列数据，因此使用循环神经网络(一种常用变种LSTM)来进行文本分类是非常常见的操作。在每一个时刻输入一个单词，将最后得到的隐层特征输入一个softmax层进行分类，图2展示了这种结构：

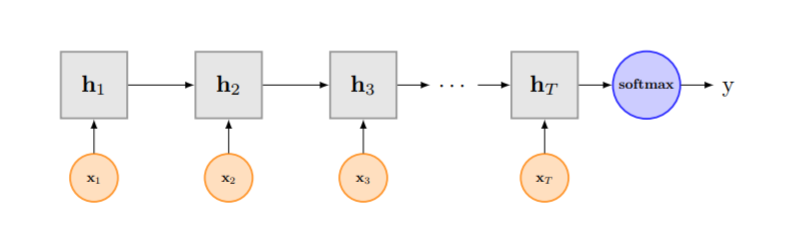


图2：LSTM for文本分类

C-LSTM[5]便是基于这种方法。

1. RNN + CNN: RCNN

传统CNN模型经常忽略文本的上下文信息和词序，无法捕捉到词义。文献[6]结合了RNN和CNN，一定程度上解决了这种问题。图3展示了RCNN模型：

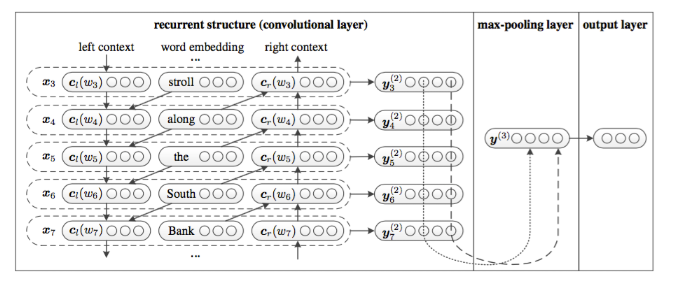


图3：RCNN

RCNN考虑了当前句子的上下文句子context，每个LSTM模块都是一个CNN网络，通过这种结合获得了比较好的分类效果。

1. Attention机制

**二、论文解决方案**

**1 研究思路**

图5展示了当前文本分类模型的典型结构：

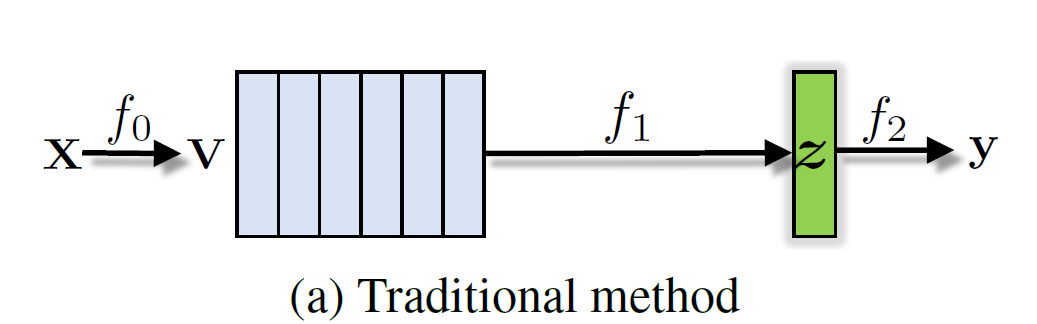


图5：RCNN

当前文本分类模型的典型结构可以分成三个部分：。

，通过预训练的词向量word embedding，将文本序列表征成连续的词向量，其中V的维度为 P \* L. P表示词向量的维度，L为文本X的长度。

，将变长的V转换成固定长度的词向量z，如上述CNN方法使用了不同filter的卷积网络，RNN方法输入文本序列得到最后一个时刻的隐层特征。

,一个分类器，将z进行分类。通常使用全连接层，非线性函数使用softmax(多分类问题)或者sigmoid(单分类问题)等。

最后将得到的y输出与样本的真实类别(标签，label)进行比较。通过上述过程可以发现，样本的标签label信息只出现在最后一步，即只直接影响到函数f2的学习过程，标签label作用在函数f0和函数f1的影响上是不直接的，比如对函数f1的参数，需要由f2反向传播计算。

因此，论文首次提出了一种新的模型，能够将标签label的信息作用到以上的每一个函数，即f0、f1、f2，充分利用标签label所提供的信息。基于与类别标签label相关性较强的词会更多地出现属于这一类别的文本种这一假设，论文使用label embedding对输入的文本向量进行矩阵乘——用于衡量当前标签label与输入的word embedding之间的“契合度”(相似度)，并将输出的相似度作为attention的权重，对输入文本的word embedding进行加权求和运算，得到一个比较好的向量z用于最后一步的分类。

这种方法相较于传统的文本分类模型，可以充分利用label embedding提供的信息，同时降低模型复杂度，缩小模型参数。因为在输入阶段同时运用了词和标签的embedding，因此论文命名为joint embedding of words and labels。

更详细的模型将在下一部分进行讲述。

**2 解决方案**

图6展示了论文提出的模型：

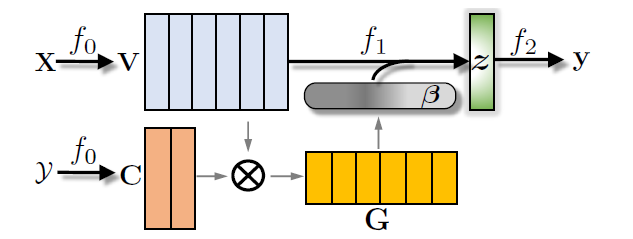
****

图6：joint embedding of words and labels

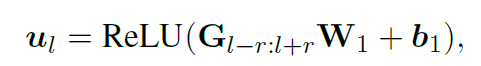
其中X表示文本输入， ,L为文本的长度，y为分类的标签，总共有K个标签。与上述模型一样，首先将X、y转换成embedding向量形式，这一部直接通过查找相应词的embedding即可实现，如果遇到OOV(out of vocabulary)词，即词不在词典中，则通过随机初始化相应的embedding。

V、C分别为X、y查找到embedding后的表示， ,其中P为word embedding的维度，论文通过向量的cosine相似度衡量了label embedding与文本中每一个词的embedding之间的“契合度”:



其中每个元素的计算方式是，符号表示矩阵元素除。更简单地说，矩阵G的每个元素是相应位置向量的cosine相似度，如衡量了第k个标签与文本第l个词的embedding的cosine距离，因为从上述公式可以看到：

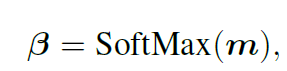
为更好地捕捉连续几个单词(比如短语)之间的相对空间信息，论文在“契合度”测量尚引入了非线性函数，对前一步计算得到的G矩阵进行进一步计算。具体来说，对于一个以单词l为中心，长度为2r+1的一个短语，矩阵G的局部块度量了标签label到第l个短语对的“契合度”，通过对这个“label-pharse”对的“契合度”建模可以学习到一个更高级别的契合度衡量向量：



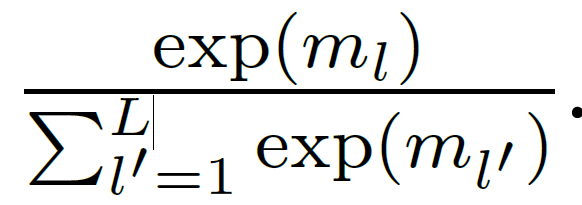
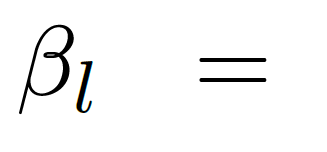
其中，计算得到的，第i个元素表示第i个标签与第l个短语的“契合度”的非线性衡量。“契合度”最高的标签通过求的最大值得到：

C:\Users\zjiehang\AppData\Local\Temp\1542378463(1).png

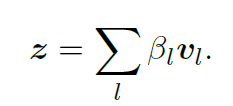
表示与第l个短语“契合度”最好的标签，通过将L(文本长度为L，短语长度只取以第l个词为中心的2r+1个词，如果是第1个词，只取往后的r+1个词，以此类推)个短语堆积在一起，得到一个长度为L的向量**，**使用向量计算attention的分数：



其中softmax函数的定义为：



最终的文本embedding是以上attention分数和原来的word embedding向量的加权和：

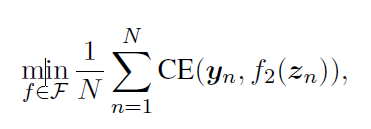


其中表示为文本第l个词的词向量。

最后将得到的向量**z** 输入一层带非线性激活函数的全连接层，将得到的输出结果作为样本的标签预测结果。

**3 损失函数**

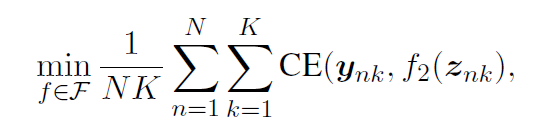
① 对于单标签问题(每个样本只能分类到K个类别中的一个)，损失函数如下所示：



其中CE(y,x)表示交叉熵损失，定义如下所示：

。

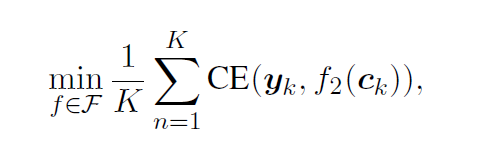
② 对于多标签问题(每个样本能够分类到K个类别中的多个)，损失函数如下所示：



其中

③ 正则化损失

针对与分类器函数，当输入的向量为每个标签的label embedding或者接近于label embedding时，分类器函数要能正确地分类成功，因此论文中增加了针对这一问题的正则化损失：



其中为第k个标签的embedding，以上公式表示将输入到分类器函数，模型能够正确地区别相应的label embedding。

整个模型的总损失等于以上两个损失的和(单标签和多标签损失针对具体问题选择其中一个)

**4 预测阶段**

预测阶段与训练阶段的过程基本一致，已知所有标签的embedding矩阵**C** 与输入的文本embedding矩阵**V**。按照第2部分的步骤输入C、V，便可以得到最终的输出。

**5 模型复杂度分析**

图7列出了不同模型的参数个数比较和复杂度分析：

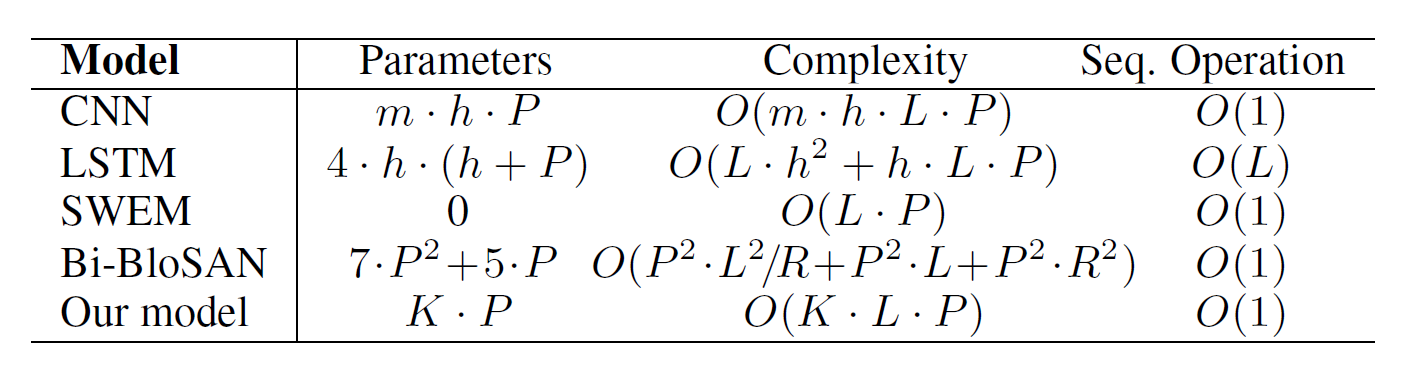


图7：不同模型的参数个数和复杂度比较

其中，CNN、LSTM的模型结构在第二章深度学习方法中介绍，SWEM、

Bi-BloSAN模型分别在文献[8]、[9]中。其中图例中K表示标签个数、m为卷积神经网络CNN中filter的个数、h为隐藏层权重维度。由于K << m、h, 论文中提出的模型参数个数远比CNN、LSTM小。在时间复杂度对比上，论文提出的模型与简单的模型SWEM在同一个数量级上，但是远比CNN、LSTM小。

**三、实验结果**

**1 测试结果**

论文在不同文本分类的数据集上做了测试，给出了比较结果，表1展示了数据集的详细信息：

表1：不同数据集及描述

|  |  |
| --- | --- |
| **数据集** | **描述** |
| AGNews | 网络新闻的主题分类数据集，包含世界、环境、体育、商业等主题 |
| Yelp Review Full | 情感分类数据集，分类标签从1-5，1、2为负面评价，3、为正面评价 |
| Yelp Review Polarity | 情感分类数据集，分类标签从1-5,1、2为负面评价，3、为正面评价 |
| DBPedia | 从DBpedia 2014（维基百科）中挑选的14个非重叠类的本体分类 |
| Yahoo！Answers Topic | 从Yahoo问答系统上挑选的主题分类数据集，包含问题内容、最好答案等十几个标签 |

图8列出了不同模型在上述数据集上的对比效果：

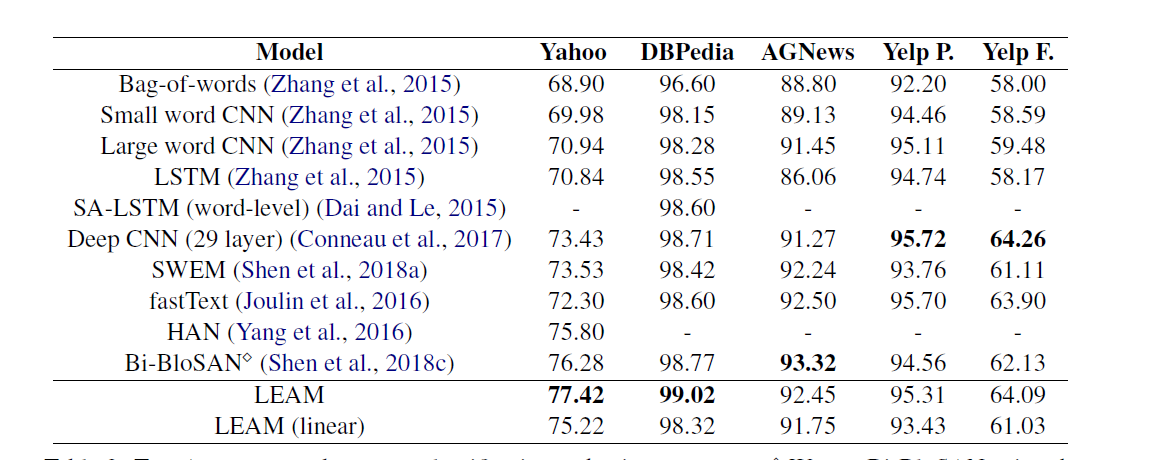


图7：不同模型的实验结果

在前两个数据集上，论文提出的模型达到了最好的效果；在其他数据集上，论文给出的模型也排到了第2、第3位，这可能出于两种原因：① 这些数据集的类别label个数比较少 ② 这些类别没有找到相应的label embedding，如后两个数据集都是情感分类数据集，其相应的标签为1—5，针对这些label没有显式对应的embedding，只能通过随机初始化embedding的方式。

同时可以从图7的最后两行注意到，非线性函数的引入更有利于分类效果。

从以上结果中可以分析得到，label embedding的引入确实能在一定程度上提高文本分类的效果，引入label embedding之后，即使是简单的模型，也可以达到与CNN、LSTM等复杂模型一样的效果。

**2 参数个数和时间对比**

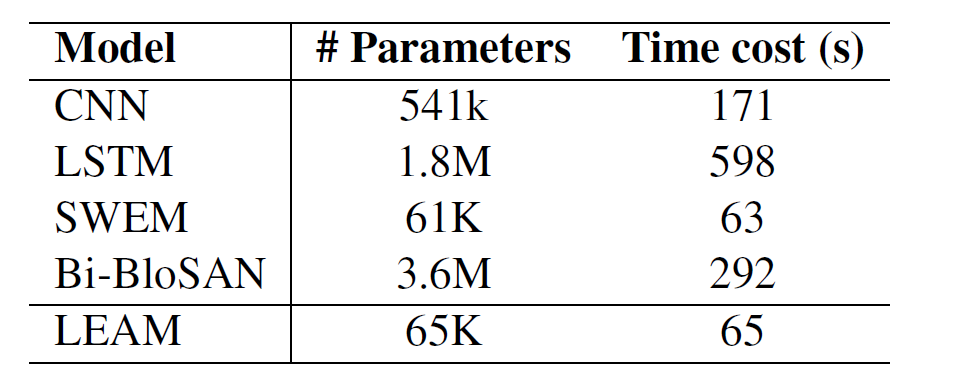


图8：不同模型的运行时间

相对于复杂的模型，我们的模型参数更少，运行时间更快，同时从上一小节可以看到，我们的模型也能取得非常好的结果(在有些数据集上也有可能达到最好效果)。只要模型结构适当，简单而浅层的模型也能达到与复杂模型一样的效果，甚至性能上超过复杂模型。

1. **收敛时间比较**
2. **超参对比(窗口大小)**
3. **词云分析**
4. **实用效果(医疗文本分类)**

**四、分析与总结**

**五、参考文献**

[1] 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, pages 3111–3119.

[2] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. EACL.

[3] Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[4] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015.Character-level convolutional networks for text classification. In NIPS.

[5] Chunting Zhou, Chonglin Sun; Zhiyuan Liu; Francis C. M. Lau. 2015. A C-LSTM Neural Network for Text Classification.. In: CoRR, abs/1511.08630.

[6] Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao. 2018. Recurrent convolutional neural networks for text classification. In AAAI.

[7] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

[8] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin, Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018a. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In ACL.

[9] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2018c. Bi-directional block selfattention for fast and memory-efficient sequence modeling. ICLR.

1. <https://zh.wikipedia.org/zh-hans/>均方误差 [↑](#footnote-ref-1)
2. <https://zh.wikipedia.org/wiki/>交叉熵 [↑](#footnote-ref-2)