# 基于FastText和Bert的新闻标题文本分类

## 背景与方法调研

**课题目标：**

文本分类是借助计算机对文本集(或其他实体或物件)按照一定的分类体系或标准进行自动分类标记。新闻标题文本分类即根据提供的新闻标题文本和类别标签训练一个新闻分类模型，然后对测试集的新闻标题文本进行分类，评价指标上使用Accuracy = 分类正确数量 / 需要分类总数量。

**课题重要性：**

新闻，是人们获取信息，了解时事热点的重要途径，尤其是近年来，新闻行业数字化发展迅猛，新闻网络平台的普及，极大地满足了人们“足不出户而知天下事”的心愿。网络平台上新闻报道、新闻评论、网友发声等文本数据快速增加。将这些文本数据正确归类，可以更好地组织、利用这些信息，因此快速、准确地完成新闻分类任务具有十分重要的意义。

面对规模巨大且不断增长的文本信息，依靠人工将海量的文本信息分类是不现实的。近些年来，借助机器学习技术完成分类任务已成为主流，计算机可以通过不断学习获得经验技能，对未知的问题可以给出一个正确的分类标签。因此，通过机器学习，可以对新闻平台上的大量数据进行自动化分类，帮助用户提高检索效率，提升用户阅读体验，同时可以在分类的基础上分析与挖掘有用的信息，协助网站运营人员了解用户需求，让信息更有效的被利用，这也是该作业的研究意义所在。

**研究发展历程：**

文本分类是自然语言处理领域一个很经典的问题，相关研究最早可以追溯到专家规则进行分类，但这样费时费力，覆盖的范围和准确率都很有限。后来伴随着统计学习方法的发展，特别是互联网在线文本数量增长和机器学习学科的兴起，逐渐形成了人工特征工程+浅层分类建模流程。随后深度学习在图像和语音取得巨大成功，也相应的推动了深度学习中自然语言处理上的发展，使得深度学习的模型在文本分类上也取得了不错的效果。

传统的机器学习分类方法将整个文本分类问题拆分为特征工程和分类器两个部分，特征工程分为文本预处理、特征提取、文本表示三个部分，其主要目的是将文本转换成计算机可以理解的格式，并封装足够用于分类的信息。并将分类信息使用机器学习模型进行处理，常用的模型有朴素贝叶斯、KNN方法、决策树、支持向量机等。

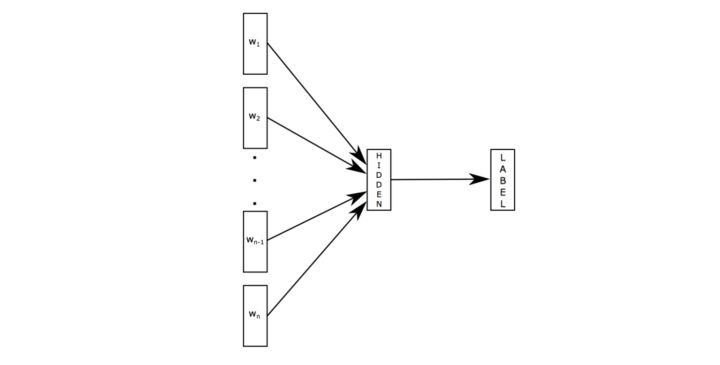
传统做法的主要问题是文本表示高纬度高稀疏，特征表达能力很弱，而且神经网络很不擅长对此类数据的处理。应用深度学习解决大规模文本分类问题最重要的是解决文本表示，再利用CNN/RNN等网络结构自动获取特征表达能力，去掉繁杂的人工特征工程，端到端的解决问题。

**可行算法：**

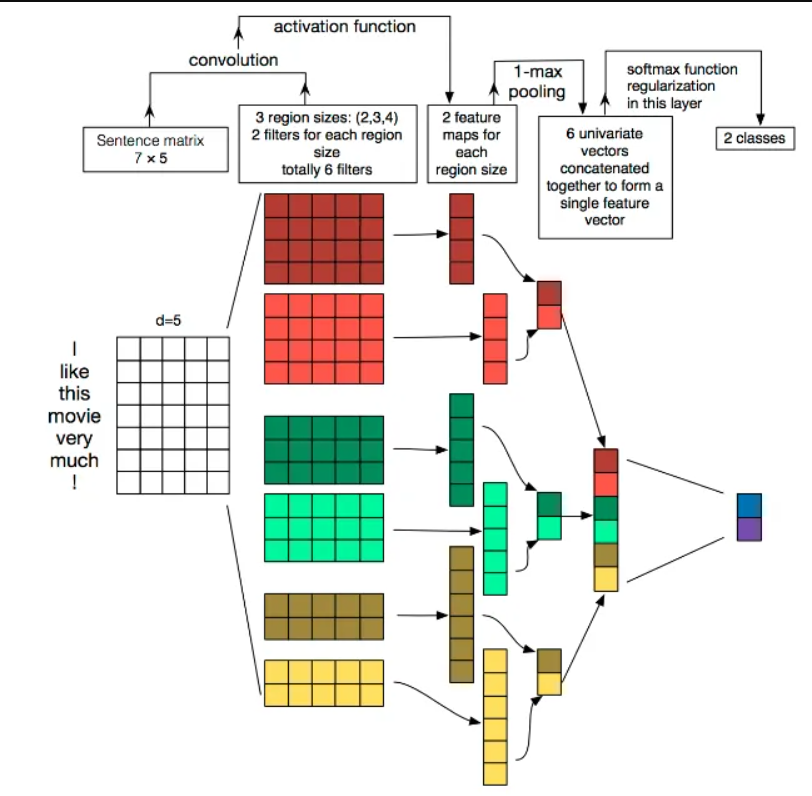
本次作业主要使用的就是深度学习的方法去对新闻标题进行分类，可行算法包括：FastText、TextCNN、TextRNN,、RCNN、Bert等，主流方法从对文本表示的方法可以分为词袋法和基于Transformer两类，本作业分别选取了两类算法中比较有特征的各一个算法：FastText和Bert进行了实验与测试。

## 方法或算法间比较与分析

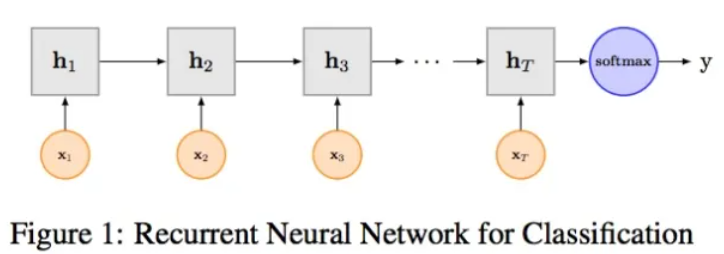
**FastText[1]**，在输入层将词袋向量化，并结合了n-grams的思想进行文本表示；在中间层将第一步中输入的向量相加求平均，得到一个新的向量w，然后将这个向量输入到输出层；输出层采用了层次softmax的方法，将一个全局多分类问题转化成了若干个二元分类问题，根据label的频次建立哈夫曼树，最终预测的结果落到某个label对应的节点上。



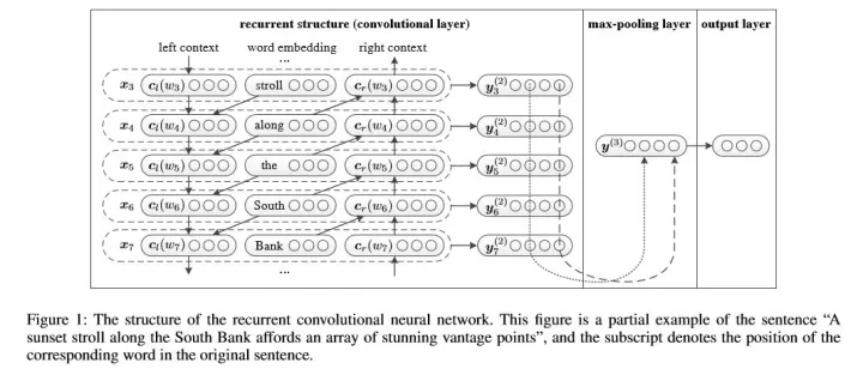
**TextCNN**，第一层为Embedding，每行是一个词向量，维度为5；第二层为Convolution，经过kernel\_sizes=(2,3,4)的卷积，每个kernel\_size有两个输出channel；第三层是一个max pooling层，这样不同长度句子经过pooling之后都能变成定长的表示；最后接一层全连接的softmax层，输出每个类别的概率。



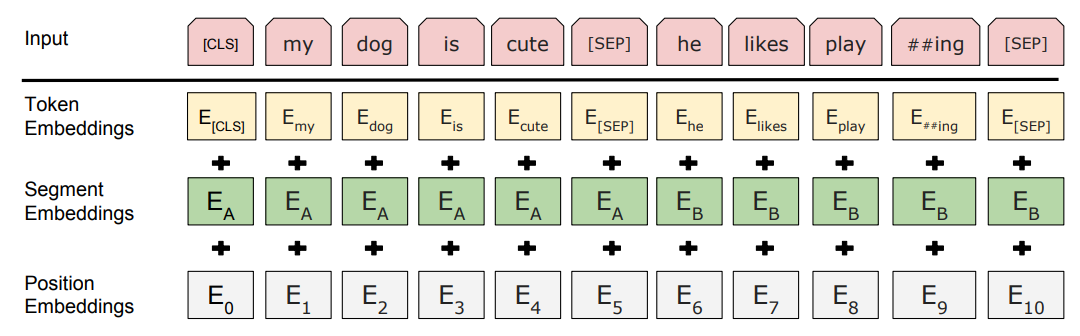
**TextRNN**，RNN网络可以处理时间序列，通过前后时刻的输出链接保证了“记忆”的留存，但其循环机制过于简单，使用了简单的f=activate(ws+b)，在梯度反向传播时出现了时间上的连乘操作，导致了梯度消失和梯度爆炸的问题。



**RCNN**，首先使用双向LSTM学习word的上下文，之后再接和TextCNN相同的卷积层，然后在seq\_length维度进行max pooling，最后接全连接层进行分类。



**Bert**，是一种基于Transformer的预训练语言表示的方法，将NLP模型的建立分为了两个阶段：pre-training和fine-training，pre-training是在大量语料上训练了一个通用的“语言理解”模型，fine-training则是在具体的NLP任务上进行相应的微调学习。对于分类层的输入，取[CLS]位置对应的向量即可。



**可行性分析：**

针对上述各个算法，TextRNN和RCNN循环神经网络具有训练慢的缺点，因为无法进行并行计算；Bert使用了位置编码加快了模型的运算速度，尽管Bert的准确率在测试集上稍低于FastText，但能用于更多来源的文本，而不是仅仅与数据集来源相同的文本，泛用性较强。而FastText基于词袋法，实现起来较Bert更为简单，是词袋法表示文本的算法的一个代表，但模型的泛用性较差。本项目的要求是训练时间不要过长，对泛用性和准确率有较高的要求，因此分别使用FastText和Bert进行实现和测试。

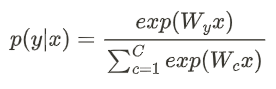
## 拟使用算法

**FastText算法：**

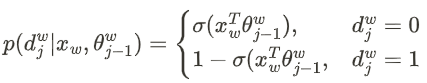
FastText的文本表示方法是词向量法，即使用独热码的方式，将每个词表示成一个向量，输出到中间层；中间层对每个向量进行相加求平均，然后输出到输出层；输出层使用层次softmax的方法，将问题看成是一个全局多分类问题，转化成若干个二分类问题。该算法主要使用了n-grams和层次softmax两个思想。

n-grams可以理解为一个窗口，在句子上以此滑动来选词，如“我 喜欢 学习”这句话，如果选择n=2，则词语组合为(“我”，“喜欢”，“学习”，“我喜欢”，“喜欢学习””)，而如果不使用n-grams，则词语组合为(“我”，“喜欢”，“学习”)，这与“学习 喜欢 我”的词语组合相同，这样表达语句的意思不够准确，因此使用n-grams来增强文本表示中特征的表达能力。

层次softmax算法是对多分类问题的一个优化解法，最朴素的方法是直接使用Softmax，



而层次softmax则是一个哈夫曼树的结构，以label出现的频次建立哈夫曼树，在每个节点进行选择时，根据如下公式计算得到的概率选择概率较大的一个方向去移动，直到叶子结点，对应的即为输出到label。使用层次softmax算法可以将计算复杂度从O(V)降到O(logV)，加快了模型训练、预测的速度。



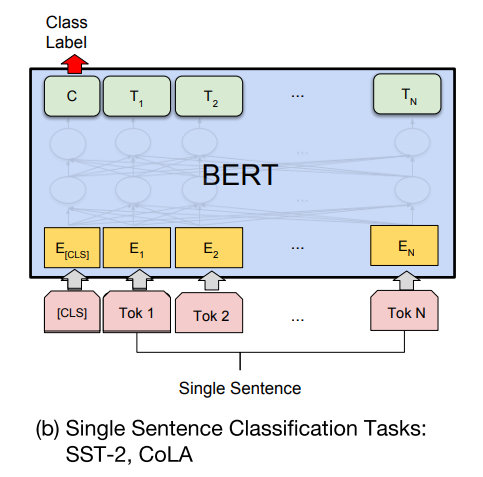
**Bert算法：**

Bert[2]是基于Transformer模型的一个自然语言处理模型，使用了谷歌预先训练的中文预训练模型，在原始的分类模型后面加上一个分类层即可进行文本分类问题的解决。对于文本表示方面，我们只需要构造原始文本对应的Token序列，并在首位分别加上一个[CLS]符和[SEP]符作为输入，由于使用了Transformer的自注意力机制，会更好的反应特征，构造Token序列即为Encoder的部分。

常用的中文汉字有3500个，这些字组成词汇，中文词汇数量高达50万个，假设词向量的维度是512，那么语言模型的参数数量至少是512\*50万=256M。模型参数数量这么大，必然需要海量的训练语料，而每一篇文章天生都是训练语料，并不需要标注，可以进行无监督学习。文章、说话都有一个特点：“说话不要颠三倒四，要通顺，要连贯”，这说明了上下文的词汇应该具有语义的连贯性。基于自然语言的连贯性，语言模型根据前文的词，预测下一个将出现的词，如果语言模型的参数正确，而且每个词的词向量设置正确，那么语言模型的预测就应该比较准确。通过这种方式实现了无监督的训练，而训练数据也是数不胜数。

Bert模型的论文名称叫《BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding》，分析其中的几个关键词，pre-training表明，作者认为，确实存在通用的语言模型，先用文章预训练通用模型，然后再根据具体应用，用有监督的方式训练数据；bidirectional，表明使用的是双向预测，从前往后预测或者从后往前预测都是单向预测，不能完整的理解整个句子的语义；而deep bidirectional意思是，双向预测依然不能完全理解句子的语义，更好的办法是用上下文全向来预测[mask]；transformer则指明了使用transformer模型。

用于文本分类的Bert模型，使用Bert预训练模型提取出的Class Label，放入全连接层中进行预测，根据数据集的类别标签进行有监督学习。



**实现难度：**

FastText模型构造较为简单，使用fasttext库封装的train\_supervised函数即可完成模型的训练，需要进行的修改为：在输入时需要对数据集进行分词、停用词处理、添加格式标签。

Bert模型由于使用了预训练模型，故只需要完成下游的文本分类任务，在bert模型后接dropout层和全连接层，使用dropout防止过拟合，使用全连接层进行分类任务，得到分类标签。需要进行的修改：添加用于文本分类的全连接层，使用全连接和dropout层训练得到分类标签。

## 主要步骤介绍

**主要步骤：**

1. 数据集的准备，其要求为新闻标题，且带有标注好的分类标签
2. 将数据集进行处理，转换为模型需要的数据格式，其中FastText需要进行分词、删除停用词、添加\_\_label\_\_<label>标签，而Bert则不需要对数据集进行预处理，提供文本和标签即可
3. 配置模型需要环境，并进行第一次训练
4. 根据每次训练结果调整模型超参，尽量提升模型文本分类的准确率
5. 开发模型的GUI使用界面，使用调参得到的最优模型参数进行预测

**详细说明：**

1. 参照一个开源仓库提供的中文自然语言处理的常用数据集[3]，先后尝试了THUCNews和toutiao\_cat\_text两个数据集，THUCNews是根据新浪新闻RSS订阅频道2005~2011年间的历史数据筛选过滤生成，包含74万篇新闻文档，不仅有新闻的标题，还包含了新闻的正文；toutiao\_cat\_text数据2018年5月采集自今日头条新闻客户端，共382688条新闻数据，作者根据领域对数据进行了分类标注，分类相较于THUCNews也更加细致，在试验之后选择了今日头条的数据集进行训练
2. 对数据集处理主要是针对FastText的训练，首先使用jieba对文本进行分词，分词之后根据停用词去删除分词中出现的对应停用词，之后再添加标签，在FastText上划分0.8为训练集，0.2为测试集；在Bert上划分0.7为训练集，0.2为验证集，0.1为测试集
3. 配置模型需要环境，主要是python库的配置，第一次训练FastText的训练速度很快，在测试集上的表现也很好，但是进行数据集之外的新闻标题预测时表现并不好；Bert由于使用了预训练模型，虽然训练较慢，但是很快（300Batch）达到了较好的效果，准确率在0.9左右
4. 在训练FastText时由于训练速度较快调整了较多次，在测试集上都能达到90%的准确率，但调整超参对于准确率的提升并不明显；在训练Bert时，发现最高准确率能达到98%，但损失函数在低处来回抖动，无法停下，是学习率过大，然后调低了学习率，并进行了8个epoch的训练
5. 开发模型的GUI使用界面，使用PyQT5库，先使用QTDesigner进行界面的设计，信号槽的预设，然后使用PyUIC工具将.ui文件转为对应的.py文件；然后再将项目重构，以mainDialog.py作为顶层模块，使用.ui导出的.py文件作为样式文件以及相关的函数调用模板，被调用函数的实现在mainDialog中，主要包括了对输入信息的读取、模型的选择、预测结果的输出，此外提供了一项随机选择新闻标题的功能，从测试集中随机选出一条新闻标题

**拟使用时间：**

1. 数据集的准备期间也是对文本分类方向各种算法的了解，挑选合适的数据集，并了解各种算法的优缺点，2天
2. 数据集预处理部分，主要是对FastText进行训练，1天
3. 配置模型需要环境部分，2h
4. 调整参数部分，由于Bert的训练较慢，调整周期较长，3天
5. 开发模型的GUI使用界面，2天

**可能存在难点：**

1. 对于文本分类的准确率提升较小，虽然能达到0.9左右，但是再向上提升较难
2. 新闻的分类有时候较为模糊，如军事与科技，教育和社会等

## 代码实现与解释性标注

**主要工作：**

1. ++ Bert/test/mytest.py
2. ++ FastText/testFastText.py
3. ++ mainDialog.py
4. ++ predict.py
5. ++ predict.ui
6. ++ Readme.md
7. ++ requirements.txt

**结果演示图片：**



## 项目总结评估及未来发展展望

**代码评估：**

FastText模型，在该问题上取得了很好的效果，具有较高的准确率和较快的训练速度，平均预测时间为1.1s，缺点是模型较大，训练准确率为90.6%。

Bert模型，在该问题上效果略差，其优点为泛用性较好，缺点为推理速度和训练速度较慢，训练准确率为89%，平均预测时间为1.7s。

**开发过程：**

学习了FastText，Bert等自然语言处理的模型原理和思想，学习了数据集的搜集和预处理，对两个模型分别进行了训练、调参优化，学习了使用PyQT库编写GUI操作界面，以及对严重异常的异常处理。

仍然存在的问题是模型的准确率相较于领域内较低，而且通过调整参数难以提升；由于训练时间原因，没有尝试应用更多的模型进行比较，如RCNN等使用了循环神经网络的模型。

**未来发展：**

在看了Bert和Transformer的论文后，感受到了人工智能的魅力，Transformer提出了自注意的机制，给文本表示提供了位置编码的新思路，提供了并行计算的可能；Bert则使用了无监督学习，从而大大减少了深度学习准备数据集需要的大量人力物力。

在未来随着GPU算力的进一步发展，模型也会越来越大，以这次训练的模型为例，Bert的模型为399M，而FastText的模型更是高达851M，自然语言处理领域词的组合数量是有限的，而模型越来越大，因此可以预料未来模型的性能也会越来越好。

目前主要的创新在于对文本表示的方式上进行更新，从原始的词袋法到现在的位置编码法，这样提升了文本的表达能力，是从模型输入的角度去进行提升；在分类方面，目前完全使用全连接的网络进行分类，在这一方面可以做出一些提升，比如提供更多的参数，或是使用其他的分类方法，或是使用新的算法提升分类速度等。

## 参考资料

[1] <https://zhuanlan.zhihu.com/p/76003775> 文本分类算法综述

[2] <https://zhuanlan.zhihu.com/p/51413773> NLP必读：十分钟读懂谷歌BERT模型

[3] <https://github.com/CLUEbenchmark/CLUEDatasetSearch#%E6%96%87%E6%9C%AC%E5%88%86%E7%B1%BB> 自然语言处理常用数据集

[4] <https://github.com/facebookresearch/fastText> 使用的FastText模型实现（库形式）

[5] <https://github.com/moon-hotel/BertWithPretrained> 参考的Bert模型实现

[6] <https://arxiv.org/pdf/1810.04805.pdf> Bert模型论文