北京航空航天大學

BEIHANG UNIVERSITY

自然语言处理实验三

基于 DNN 的新闻文本分类

院 系 名 称	人工智能研究院
任 课 老 师	丁嵘
学 生 姓 名	赵天一
学 生 学 号	19373006

2021 年 12 月 11 日

一. 赛题任务及数据集介绍

赛题数据为新闻文本，并按照字符级别进行匿名处理。整合划分出 14 个候选分类类别,分别是:

财经、彩票、房产、股票、家居、教育、科技、社会、时尚、时政、体育、星座、游戏、娱乐的文本数据。

数据标签和数字的对应如下:

{'科技': 0, '股票': 1, '体育': 2, '娱乐': 3, '时政': 4, '社会': 5, '教育': 6, '财经': 7, '家居': 8, '游戏': 9, '房产': 10, '时尚': 11, '彩票': 12, '星座': 13}

赛题数据由以下几个部分构成：训练集 20w 条样本，测试集 A 包括 5w 条样本，为了预防选手人工标注测试集的情况，我们将比赛数据的文本按照字符级别进行了匿名处理。处理后的赛题训练数据如下：

label	text
6	57 44 66 56 2 3 3 37 5 41 9 57 44 47 45 33 13 63 58 31 17 47 0 1 1 69 26 60 62 15 21 12 49 18 38 20 50 23 57 44 45 33 25 28 47 22 52 35 30 14 24 69 54 7 48 19 11 51 16 43 26 34 53 27 64 8 4 42 36 46 65 69 29 39 15 37 57 44 45 33 69 54 7 25 40 35 30 66 56 47 55 69 61 10 60 42 36 46 65 37 5 41 32 67 6 59 47 0 1 1 68

所以由于数据提供者为了防止作弊进行了字符级的匿名处理,显然不用进行分词任务了,可以直接当作输入进行训练

二. FastText 算法及网络结构介绍

1.fastText 简介

fastText 是一个快速文本分类算法，与基于神经网络的分类算法相比有两大优点：

(1)fastText 在保持高精度的情况下加快了训练速度和测试速度

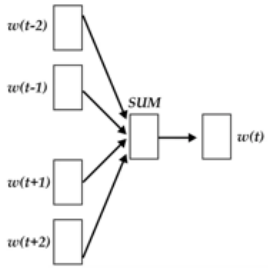
(2)fastText 不需要预训练好的词向量，fastText 会自己训练词向量

(3)fastText 两个重要的优化：Hierarchical Softmax、N-gram

2、fastText 模型架构

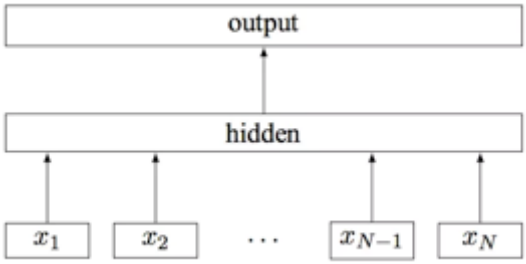
fastText 模型架构和 word2vec 中的 CBOW 很相似,不同之处是

fastText 预测标签而 CBOW 预测的是中间词，即模型架构类似但是模型的任务不同。下面我们先看一下 CBOW 的架构：



word2vec 将上下文关系转化为多分类任务，进而训练逻辑回归模型，这里的类别数量|V|词库大小。但是通常的文本数据中，词库的规模很大，在训练中直接训练多分类逻辑回归并不现实。

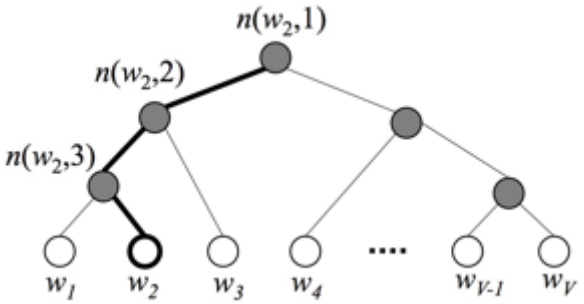
fastText 模型架构:其中 x1,x2,...,x_n表示一个文本中的 n-gram 向量，每个特征是词向量的平均值。这和前文中提到的 cbow 相似，cbow 用上下文去预测中心词，**而此处用全部的 n-gram 去预测指定类别**



三. 层次 softmax

softmax 函数常在神经网络输出层充当激活函数，目的就是将输出层的值归一化到 0-1 区间，将神经元输出构造成概率分布，主要就是起到将神经元输出值进行归一化的作用 softmax 的公式很熟悉不在报告里加以赘述了。

在普通的 softmax 中，计算一个类别的 softmax 概率时，我们需要对所有类别概率做归一化，在这类别很大情况下非常耗时，因此提出了分层 softmax(Hierarchical Softmax),思想是根据类别的频率构造霍夫曼树来代替标准 softmax，通过分层 softmax 可以将复杂度从 N 降低到log_N,下图给出分层 softmax 示例：



在层次 softmax 模型中，叶子结点的词没有直接输出的向量，而非叶子节点都有响应的输在在模型的训练过程中，通过 Huffman 编

码，构造了一颗庞大的 Huffman 树，同时会给非叶子结点赋予向量。我们要计算的是目标词 w 的概率，这个概率的具体含义，是指从 root 结点开始随机走，走到目标词 w 的概率。

四、N-gram 特征

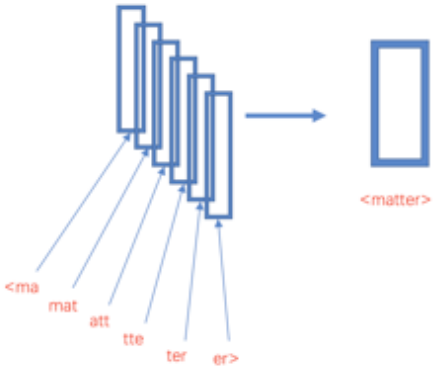
n-gram 是基于语言模型的算法，基本思想是将文本内容按照子节顺序进行大小为 N 的窗口滑动操作，最终形成窗口为 N 的字节片段序列。而且需要额外注意一点是 n-gram 可以根据粒度不同有不同的含义，有字粒度的 n-gram 和词粒度的 n-gram，下面分别给出了字粒度和词粒度的例子：

我来到达观数据参观

相应的 bigram 特征为：我来 来到 到达 达观 观数 数据 据参 参观

相应的 trigram 特征为：我来到 来到达 到达观 达观数 观数据 数据参 据参观

对于文本句子的 n-gram 来说，如上面所说可以是字粒度或者是词粒度，同时 n-gram 也可以在字符级别工作，例如对单个单词 matter 来说，假设采用 3-gram 特征，那么 matter 可以表示成图中五个 3-gram 特征，这五个特征都有各自的词向量，五个特征的词向量和即为 matter 这个词的向其中“<”和“>”是作为边界符号被添加，来将一个单词的 n-grams 与单词本身区分开来：



从上面来看，使用 n-gram 有如下优点

1、为罕见的单词生成更好的单词向量：根据上面的字符级别的 n-gram 来说，即是这个单词出现的次数很少，但是组成单词的字符和其他单词有共享的部分，因此这一点可以优化生成的单词向量

2、在词汇单词中，即使单词没有出现在训练语料库中，仍然可以从字符级 n-gram 中构造单词的词向量

3、n-gram 可以让模型学习到局部单词顺序的部分信息，如果不考虑 n-gram 则便是取每个单词，这样无法考虑到词序所包含的信息，即也可理解为上下文信息，因此通过 n-gram 的方式关联相邻的几个词，这样会让模型在训练的时候保持词序信息

三.实验结果

四.问题与思考

1.比赛成绩高于本地测试成绩

在本机测试的正确率是 0.825 左右，最后的实验结果如图所示：0.8499，和本地测试很接近也是类似的，所以可以推断出 test 和 train 的数据分布是一致的，才可能导致测试结果优于本地测试结果。

```
(mmdetection) practice@ubuntu18-Precision-7920-Tower:/data/course/ZTY/text_class
ification$ python fastText.py
Read 9M words
Number of words:  5341
Number of labels: 14
Progress: 100.0% words/sec/thread:  223216 lr:  0.000000 avg.loss:  0.153626 ETA:   0h 0m 0s
0.8252288532444629
```