中国科学院大学
University of Chinese Academy of Sciences

课程作业报告

基于 TextCNN 的电影评论情感分类

作者姓名：陈子轩　2019E8015061013

学科专业：软件工程

所在单位：中国科学院大学计算机科学与技术学院

2020 年 5 月

基于 TextCNN 的电影评论情感分类

陈子轩

摘　　要

本项目属于自然语言处理任务中的情感二分类。通过输入一段电影评论，要求模型输出该段文字的情感极性（正向或负向）。对比了不同的 word2Vec 模型、是否使用停用词表在训练结果中的作用。本项目使用 textCNN 模型，在测试集上成功达到了 84.28% 的准确率。

1　介绍

(1) 本实验要解决的问题为电影评论情感分析，具体过程为，输入一段文字，要求模型输出该段文字的情感极性（正向或负向）。

(2) 训练集：包含 2 万条左右中文电影评论，其中正负向评论各 1 万条左右。
验证集：包含 6 千条左右中文电影评论，其中正负向评论各 3 千条左右。
测试集：包含 360 条左右中文电影评论，其中正负向评论各 180 条左右。

(3) 尝试使用 TextCNN 实现对中文电影评论的情感分析，目标为测试集准确率在 83% 以上。

2　实验过程

2.1　深度学习平台

TensorFlow == 2.0.0。

2.2　语料库分析

训练集：包含 2 万条左右中文电影评论，其中正负向评论各 1 万条左右。
验证集：包含 6 千条左右中文电影评论，其中正负向评论各 3 千条左右。
测试集：包含 360 条左右中文电影评论，其中正负向评论各 180 条左右。

对语料库中文本长度进行统计，如**图 1** 所示，可见有 99.61% 的文本长度都在 150 以下。于是在训练时设定统一输入文本长度 sequenceLength = 150，超出将被剪断，不足则补 <pad> 标志。

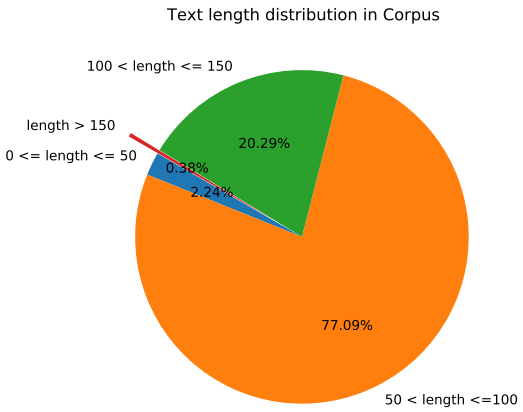

图 1: 语料库中文本长度分布

2.3 停用词表

停用词是指在信息检索中，为节省存储空间和提高搜索效率，在处理自然语言数据（或文本）之前或之后会自动过滤掉某些字或词，这些字或词即被称为 Stop Words（停用词）。本实验中的停用词表为来自https://github.com/goto456/stopwords的停用词表 cn_stopwords.txt 和空的停用词表 empty_stopwords.txt。

2.4 预训练词向量

本项目中没有使用预训练模型 wiki_word2vec_50.bin。而使用 word2vec 模型 [2]，一般分为 CBOW(Continuous Bag-of-Words) 与 Skip-Gram 两种模型，如**图 2** 所示。CBOW 模型的训练输入是某一个特征词的上下文相关的词对应的词向量，而输出就是这特定的一个词的词向量。Skip-Gram 模型和 CBOW 模型的思路是相反的，即输入是特定的一个词的词向量，而输出是特定词对应的上下文词向量。CBOW 对小型数据库比较合适，而 Skip-Gram 在大型语料中表现更好。

在 gensim.models.word2vec.Word2Vec 函数中，有参数 sg：用于设置训练算法，默认为 0，对应 CBOW 算法；sg = 1 则采用 Skip-Gram 算法。其他参数设置为 size = 300, min_count = 1, window = 10, workers = multiprocessing.cpu_count(), iter = 20。预训练 word2vec 模型如**表 1** 所示。

表 1: 预训练 word2vec 模型

模型名	其他模型参数	
	sg	sentences
word2VecModel_1	0	train.txt
word2VecModel_2	0	train.txt + validation.txt
word2VecModel_3	1	train.txt
word2VecModel_4	1	train.txt + validation.txt

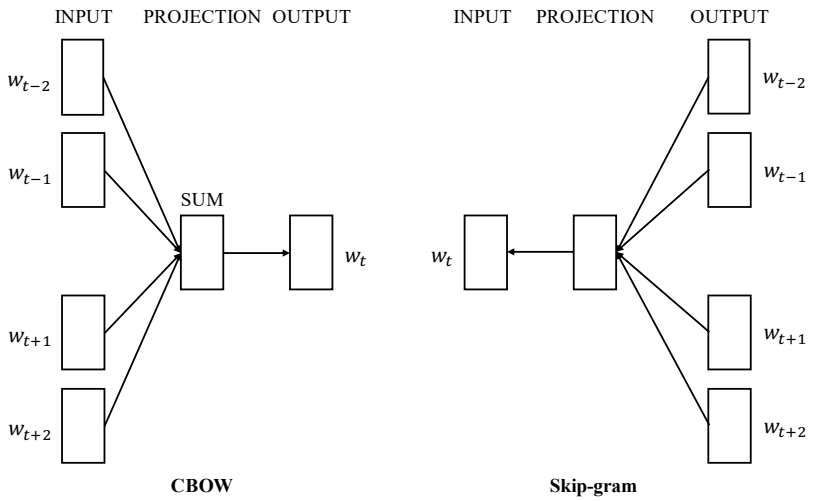

图 2: CBOW(Continuous Bag-of-Words) 与 Skip-Gram 两种模型

2.5 TextCNN

Yoon Kim 在论文 [1] 中提出 TextCNN。将卷积神经网络 CNN 应用到文本分类任务，利用多个不同尺寸的卷积核来提取句子中的关键信息（类似于多窗口大小的 n-gram），从而能够更好地捕捉局部相关性。

TextCNN 和传统的 CNN 结构类似，具有词嵌入层、卷积层、池化层和全连接层的四层结构，论文 [1] 中给出的 TextCNN 结构如**图 3** 所示。

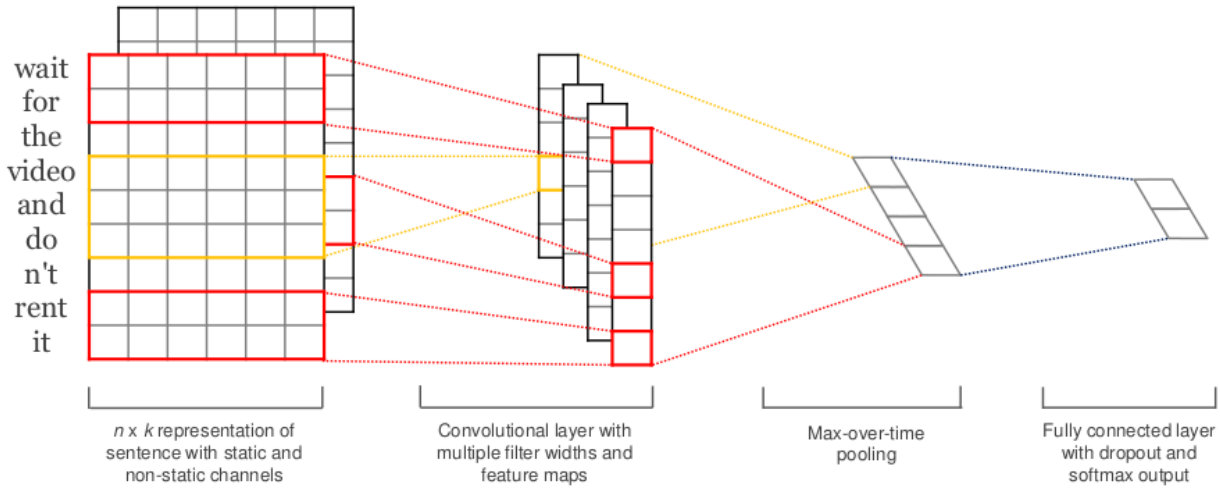

图 3: Model architecture with two channels for an example sentence.

图 4 为本实验中设计的 TextCNN 结构，具体参数量如**表 2** 所示。在 CNN 子网络模型中，如**表 3** 所示，设计了不同的卷积核尺寸 [1, 2, 3, 4, 5]，实验中卷积核数 numFilters = 120。全连接层中神经元数量 units = 10，激活函数为 relu 函数，l2 正则项系数为 0.1。Dropout 层中 rate = 0.5。输出层使

用 sigmoid 函数进行二分类。使用优化器为 Adam(lr=1e-4) 及 metrics = ['accuracy']。

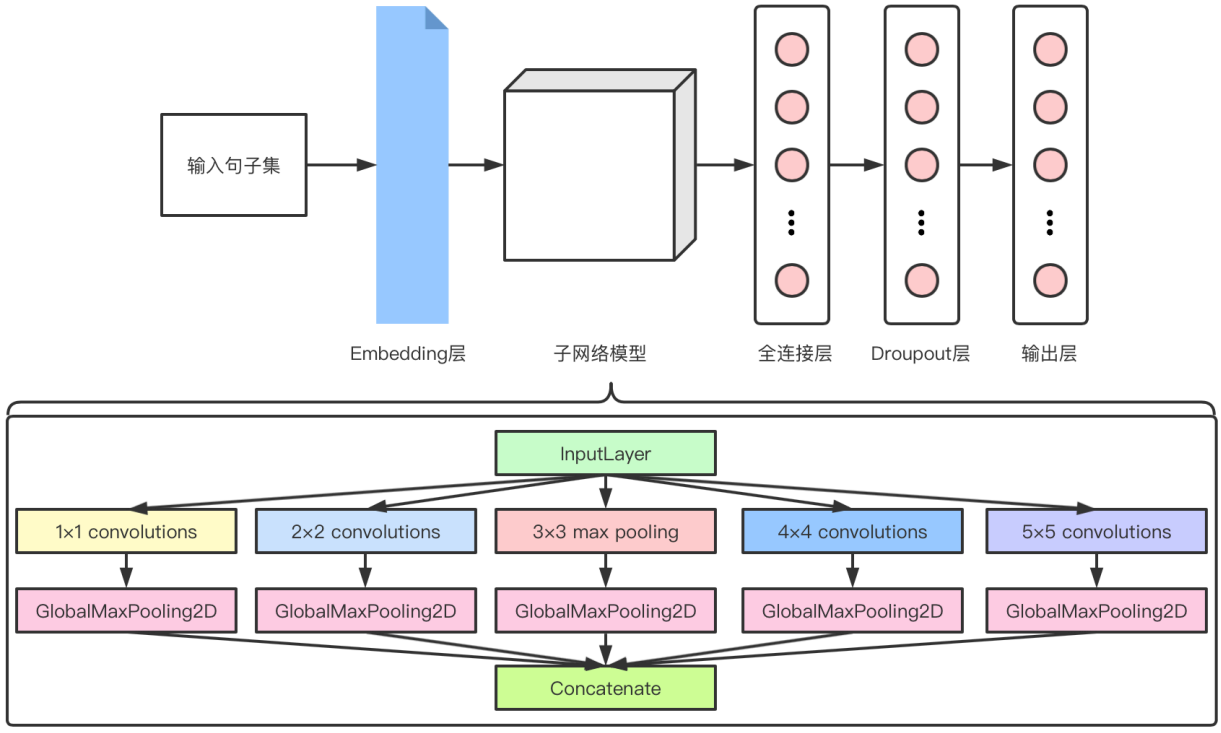

图 4: 本实验中设计的 TextCNN 结构

表 2: 本实验中的 TextCNN 结构 Model: "sequential"

Layer (type)	Output Shape	Param #
embedding(Embedding)	(None, 150, 300)	16542300
reshape (Reshape)	(None, 150, 300, 1)	0
model (Model)	(None, 600)	540600
flatten (Flatten)	(None, 600)	0
dense (Dense)	(None, 10)	6010
dropout (Dropout)	(None, 10)	0
dense_1 (Dense)	(None, 1)	11

Total params: 17,088,921

Trainable params: 17,088,921

Non-trainable params: 0

训练过程中使用 callbacks = [reduce_lr, early_stopping, model_checkpoint]，其中

```
reduce_lr = keras.callbacks.ReduceLROnPlateau(monitor = "val_loss", patience = 10,
    mode = "auto")
early_stopping = keras.callbacks.EarlyStopping(monitor="val_loss", patience=5)
```

```
model_checkpoint = keras.callbacks.ModelCheckpoint(config.modelCheckpoint, save_
    best_only=True, save_weights_only=True)
```

表 3: CNN 子网络模型 Model: "model"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 150, 300, 1)]	0	
conv2d (Conv2D)	(None, 150, 1, 120)	36120	input_1[0][0]
conv2d_1 (Conv2D)	(None, 149, 1, 120)	72120	input_1[0][0]
conv2d_2 (Conv2D)	(None, 148, 1, 120)	108120	input_1[0][0]
conv2d_3 (Conv2D)	(None, 147, 1, 120)	144120	input_1[0][0]
conv2d_4 (Conv2D)	(None, 146, 1, 120)	180120	input_1[0][0]
global_max_pooling2d (GlobalMaxPooling2D)	(None, 120)	0	conv2d[0][0]
global_max_pooling2d_1 (GlobalMaxPooling2D)	(None, 120)	0	conv2d_1[0][0]
global_max_pooling2d_2 (GlobalMaxPooling2D)	(None, 120)	0	conv2d_2[0][0]
global_max_pooling2d_3 (GlobalMaxPooling2D)	(None, 120)	0	conv2d_3[0][0]
global_max_pooling2d_4 (GlobalMaxPooling2D)	(None, 120)	0	conv2d_4[0][0]
			global_max_pooling2d[0][0]
			global_max_pooling2d_1[0][0]
concatenate(Concatenate)	(None, 600)	0	global_max_pooling2d_2[0][0]
			global_max_pooling2d_3[0][0]
			global_max_pooling2d_4[0][0]

Total params: 540,600

Trainable params: 540,600

Non-trainable params: 0

训练结果如**表 4** 所示。

表 4: 训练结果

TextCNN 模型	预训练模型	停用词表	训练集		验证集		测试集		过程图
			acc %	loss %	acc %	loss %	acc %	loss %	
textCNN_1	word2VecModel_1	empty_stopwords	95.08	22.10	81.79	47.92	79.95	48.61	图 **5**
textCNN_2		cn_stopwords	97.33	19.68	83.02	47.13	81.03	52.03	图 **6**
textCNN_3	word2VecModel_2	empty_stopwords	96.76	18.91	82.68	48.20	82.66	48.30	图 **7**
textCNN_4		cn_stopwords	93.58	22.15	82.84	46.66	82.11	47.21	图 **8**
textCNN_5	word2VecModel_3	empty_stopwords	91.75	20.49	83.44	43.01	84.28	42.28	图 **9**
textCNN_6		cn_stopwords	95.76	19.39	83.05	44.10	81.84	41.37	图 **10**
textCNN_7	word2VecModel_4	empty_stopwords	93.99	21.21	83.82	43.56	83.20	42.08	图 **11**
textCNN_8		cn_stopwords	94.60	24.02	84.06	42.96	83.47	40.03	图 **12**

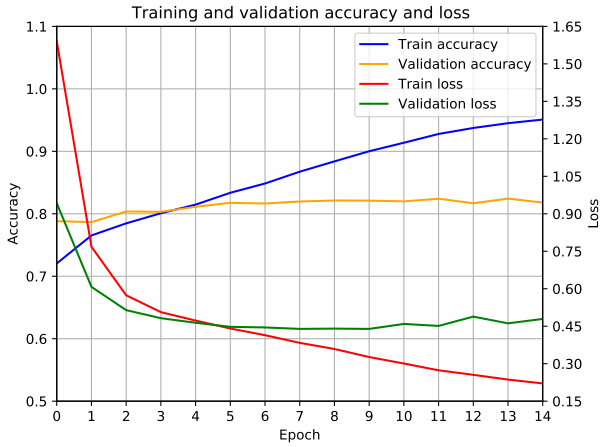

图 **5:** textCNN_1 训练过程图

图 **6:** textCNN_2 训练过程图

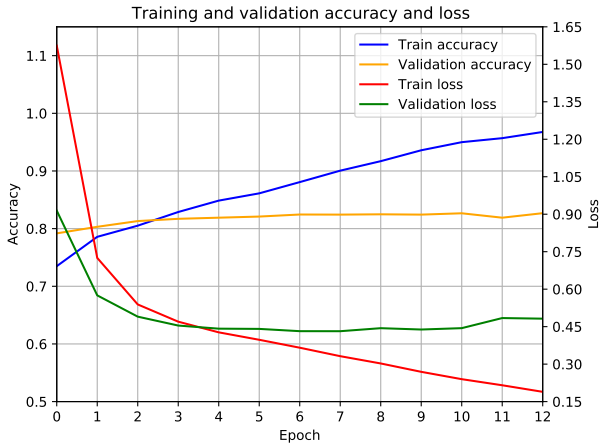

图 **7:** textCNN_3 训练过程图

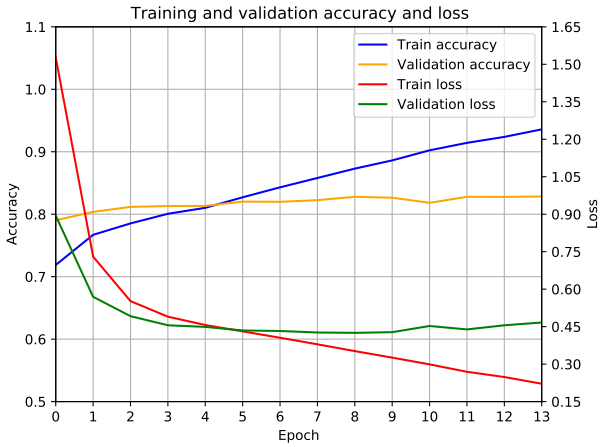

图 **8:** textCNN_4 训练过程图

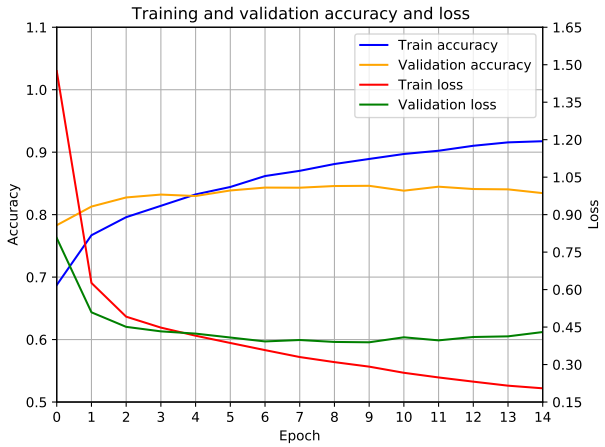

图 **9:** textCNN_5 训练过程图

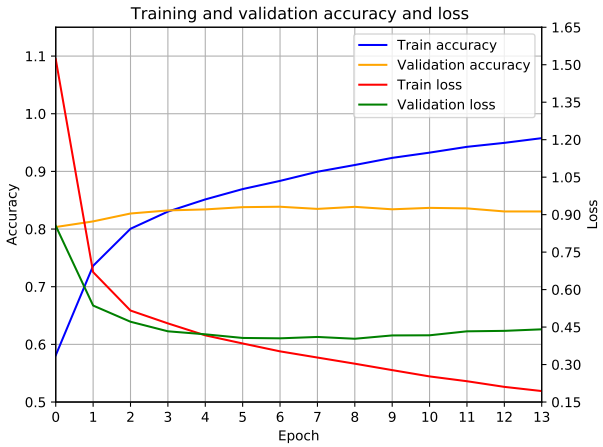

图 **10:** textCNN_6 训练过程图

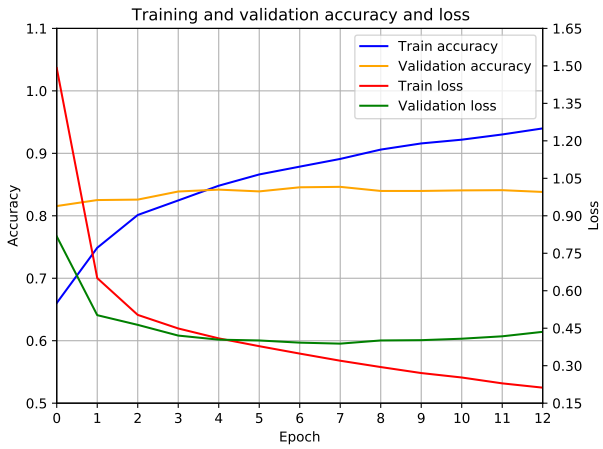

图 11: textCNN_7 训练过程图

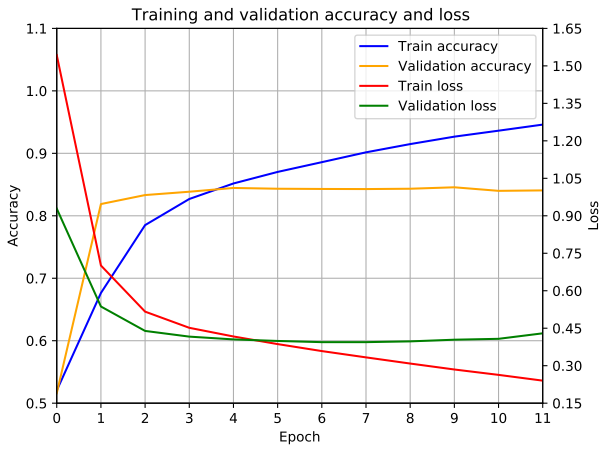

图 12: textCNN_8 训练过程图

3 结论

本项目属于自然语言处理任务中的情感二分类。总结上述 8 个模型各项数据，可以发现 textCNN_5 模型超过了 83%，达到了 84.28%。word2vec 模型中：一方面，word2Vec_3 比 word2Vec_1 训练效果好、word2Vec_4 比 word2Vec_2 训练效果好，说明了在本实验中 Skip-Gram 模型比 CBOW 模型效果好；另一方面，word2Vec_2 比 word2Vec_1 训练效果好、word2Vec_4 比 word2Vec_3 训练效果好，说明了在预训练词向量过程中，语料库越大，训练越精确。在是否使用停用词表方面，影响既有积极也有消极。我认为对于参数量如此巨大的模型，实验所用的语料库过小，很容易出现过拟合现象，由各训练图也可以看出，在第 3 轮训练后，验证集上 acc 值增长缓慢，同时虽然预设 epochs = 20，但均出现了早停，说明实验所用语料库确实太小。

参考文献

[1] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751, 2014.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.