

Lab2信息抽取

成员：

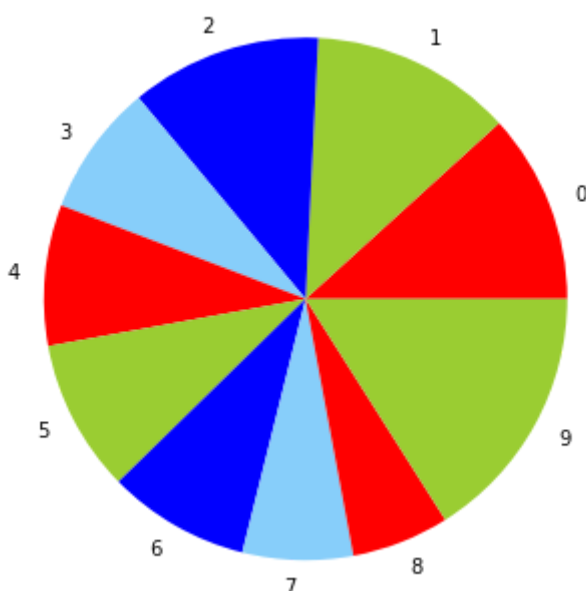
- 周恩帅 PB17111561
- 张文岚 PB17111587

本实验实现了关系抽取。

实验思路

由于时间关系，我们只完成了关系的抽取，即输入一个句子，输出句子中最有可能包含的关系。因此，可以将此问题建模成一个多分类任务，输入是一句话，输出是十类标签中的某一个。综合考虑，我们使用深度学习模型去实现这个多分类任务。

实验之前我们先统计各句子标签的比例：



可以看出各标签的比例还算均衡。

算法概述

最终采用的模型是由两部分组成：使用预训练的Albert网络获得词的向量化表示，将得到的词向量输入双向LSTM网络，LSTM后接一个全连接层，使用softmax激活，得到每个关系类别的概率，概率最大的标签即为句子的标签。这个模型的效果是最好的，但是在得到最好的结果之前，我们也探索了很多其他的方法。

在词嵌入、特征提取和LSTM后接的子网络上有很多选择。以下是我们探索的过程。

词嵌入

为了使一个字符串类型的句子可以输入进神经网络里运算，我们需要生成这个句子的向量化表示，句子的最小语义单元是单词，因此如何恰当地用一个数值向量来表征一个单词是我们首要考虑的问题。





独热码或整数编码

独热码和整数编码都可以理解为给每个单词分配一个序号，这种方式虽然简单，但是没有办法反映出词与词之间的语义相似关系，效果显然是不好的。因此，我们抛弃了这种办法。

Word2Vec、Glove、fasttext

Word2Vec、Glove、fasttext都是基于统计的一类算法，大多基于CBOW和Skip-gram模型，通过这些算法得到的词向量的维度较低，语意相似的词在向量空间上也会比较相近，而且通用性很强，可以用在不同的任务中。

由于实验所给的数据集很小，训练语料不足，因此我们使用了一些预训练的模型，如google的word2vec，斯坦福的glove，facebook的fasttext等预训练词向量，这些模型的预训练语料一般来源于新闻或者wiki等网站，通用性比较强。这些模型还提供了许多选择，可以根据实际情况选择不同长度的词向量。

 glove.6B.50d.txt	2014/8/5 4:15	Text 源文件	167,335 KB
 glove.6B.100d.txt	2014/8/5 4:14	Text 源文件	338,982 KB
 glove.6B.200d.txt	2014/8/5 4:14	Text 源文件	677,181 KB
 glove.6B.300d.txt	2014/8/28 3:19	Text 源文件	1,013,636 KB

但是这些方法也有一定的缺陷，通过它们得到的词向量是静态的，即一个词对应的词向量在任何文本中都是固定不变的，这显然无法解决一词多义的问题。

实际情况也与我们的分析相一致：我们尝试了多种维度的词向量，最好的准确率也只有48%。

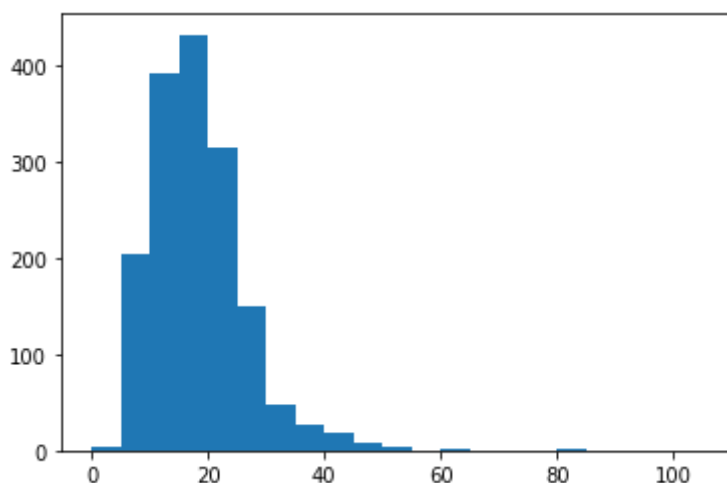
ALBERT

ALBERT的全称是A Lite BERT。BERT是近年来很火的开源nlp模型，但是它的网络参数太多，使用起来太慢。而ALBERT基于这一点进行了改进，相较于BERT，它缩小了整体的参数量，加快了训练速度，增加了模型效果。

同样由于语料太少，训练效果不好，我们直接使用tf hub上开源的ALBERT预训练模型

`albert_en_base`。然而在笔记本上网络推理的速度依然无法接受，内存也不够用。因此我们直接把它放到服务器上跑了。

ALBERT的词向量维度是768，除此之外这里存在一个超参，即句子的填充长度，原模型默认128，但显然过于长了，为此我们统计了下训练集的句子长度：



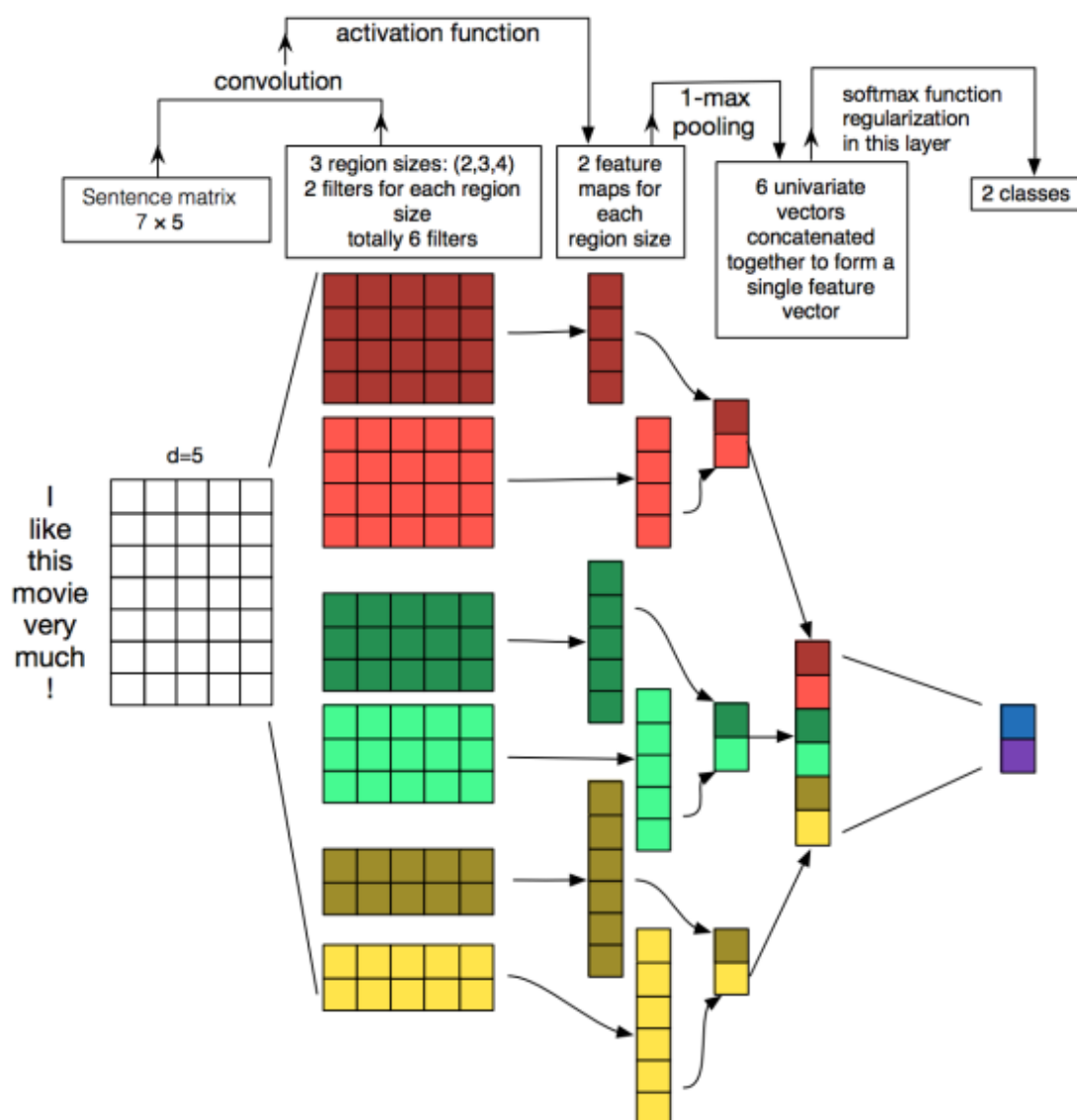
最终我们设置长度为64。效果还是不错的，最高达到了60%的准确率。

神经网络

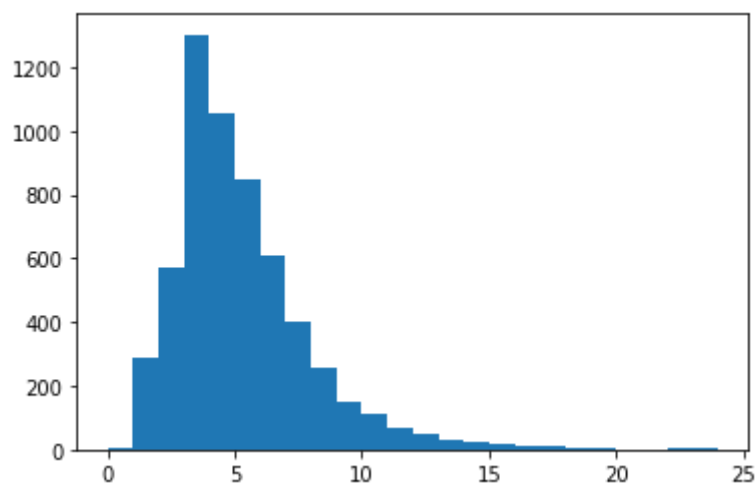
在选定词嵌入模型之后，我们需要思考如何实现这个分类网络。显然，句子中每个词前后都是有一定关系的，所以我们主要考虑卷积神经网络和循环神经网络。

CNN

CNN的主要思路就是通过不同大小的卷积核来提取不同距离的词之间的关系特征，如卷积核高为3，宽为词向量长度，表示的意思是一个词与其相近的3个词之间可能存在实体关系。在句子通过不同大小的卷积核之后，我们得到了相同数量的channel，每个channel都是一个列向量，在这之上我们进行最大池化操作，得到一个标量值。再将得到的所有标量融合进一个列向量，该列向量输入进一个softmax层，最终得到概率。网络的结构大致如下：



这里面有一个超参，即卷积核的尺寸，确切的说卷积核的高，反映在句子里面就是：我们认为距离为多少的两个词之间最有可能有实体关系。为此我们先统计了下训练集中具有实体关系的两个词的距离：



我们发现距离为[2, 6]的词对居多，因此我们使用了高为2、3、4、5、6的卷积核。

但令人遗憾的是模型很难收敛，准确率也十分的低，猜测可能是因为网络结构复杂导致梯度消失。

双向LSTM

既然CNN的效果不好，那么我们就考虑使用NLP领域常用的LSTM网络，这里的LSTM接受一个长为64的句子输入，输出一个列向量，即多对一的结构。开始我们使用的是单向LSTM，准确率大概有54%。之后我们又使用双向LSTM，这次效果最好，准确率达到了60%。

同样这里也有一个超参，即LSTM的结点个数，我们试了64，128，256，768，最终发现256和768效果基本相同且强于其他情形，因此我们将结点数设为256。

此外，我们还尝试了GRU，推理速度与LSTM基本相同，但是效果略差。

LSTM+CNN

一个大胆的想法，CNN和LSTM一起使用会怎么样？我们很快尝试了这个想法，遗憾的是，依然没法收敛，准确率到38%就不再变化了。所以，我们最终只使用了双向LSTM。

分类网络

在LSTM提取出句子特征后，我们需要将该特征映射到一个概率向量，以表征每个标签的可能性。最直接的方式就是后接一个全连接层，使用softmax激活。事实上，我们正是这样做的。

其实，我们也考虑再多加几个全连接层，但是后来发现效果并无提升。于是我们只使用了一层softmax。

算法实现

具体代码请参考src目录下的 `re-bert-lstm.py`，一些关键函数的说明请见 `README.md` 文件。

实验结果

一共提交了10次，准确率最好的一次是0.6，具体如下：

Web info 2020 Project2

Your student ID

Select file...

Browse ...

Success!

Filename	ACC-Relation	ACC-NER
周恩帅-PB17111561-9.txt	0.60375	0.0