基于 Focal loss 的 Bi-LSTM 和 TF-IDF＋K-means 的智能政务系统

摘要

随着科技的创新与发展，深度学习与自然语言技术逐渐应用于各个行业领域。近年来，将政务系统机械化、智能化，已成为提升政府管理水平与施政效率的社会治理创新发展的新趋势。借此机会，本文通过采用创新的思路，将常规算法进行结合与改进，构建基于 Focal loss 的 Bi-LSTM 的留言分类模型、$TF-IDF$ 结合 $K-means$ 的热点挖掘模型和 DSSM-LSTM 的答复评价模型。

在模型的建立前，首先将留言文本进行预处理，在完成数据清洗排除空留言、重复留言后，将文本进行中文分词和去停用词，最后将结果独热表示，得到计算机能够识别的数字形式。由于 One-Hot 表示后，词向量组合成的句子维度太大，会导致后续计算易出错且运行效率低下，因此在预处理后我们常通过分布式表示对句子的词向量矩阵进行降维处理，并获得包含语义信息的最终句子向量化表达形式。

针对留言分类模型，把降维后的词向量放入 Bi-LSTM，通过学习训练集文本中词向量的权重，语义相近的句子会有相近的权值。在模型中我们在两处设置了 Dropout 来控制训练集的过拟合，再通过 Flatten 函数将多维向量转为单行向量，然后采用 Dense 将 Bi-LSTM 的输出层全连接。并使用 Focal loss 损失函数，处理数据的不平衡，构建最终的留言分类模型。多次试验验算证明，转为 Focal loss 损失函数后，模型的 F1-Score 提升了 6%左右，精确度提升到 4%，且难分类样本的评价指标皆有所上升。

针对热点归类模型，在将文本预处理后，我们利用余弦相似度计算 $TF-IDF$ 训练好的词向量在空间上的距离，并利用 $K-means$ 实现文本聚类。我们设置 k=350 个起始质心，用欧氏距离计算其与剩余元素的距离，并就近归类。最后通过算术平均值迭代计算，确定 K 个簇的最终质心。将多种 k 值下的实验结果进行对比，当 k=350 时聚类效果最佳。

针对答复评价模型，由于已给出问答内容，我们将评价指标转为对两段文本的相似度计算。且考虑上下文语义，我们选用 DSSM-LSTM 这一深度语义匹配模型。将通过 Embedding 后的结果带入 LSTM 神经网络，得到语义特征，并用 Cosine 计算问题 Q 和答复 A、问题 Q 和无关答复 A 的相似度，求和后通过 softmax 函数归一化，得到条件概率分布。

并通过对实验的评估可得上述三个模型的可行性与有效性。

关键字：Bi-LSTM　Focal-loss　智能分类　$TF-IDF$　$K-means$ 聚类　热点挖掘　DSSM-LSTM　答复评价

Abstract

With the innovation and development of science and technology, deep learning and natural language technology are gradually applied to various industries.In recent years, the mechanization and intelligence of the government affairs system has become a new trend of social governance innovation and development to improve government management and governance efficiency.. Taking this opportunity, this article uses innovative ideas to combine and improve conventional algorithms to build a message classification model based on Bi-LSTM under Focal loss, a hot spot mining model based on $TF-IDF$ and $K-means$, and a response evaluation model based on DSSM-LSTM.

Before the establishment of the model, the text of the message is pre-processed. After the data cleaning is completed to exclude empty messages and repeated messages, the text is segmented in Chinese and the stop words are removed. Finally, the hot words are expressed and the numbers recognized by the computer are obtained. form. Because the dimension of the combined sentence after One-Hot representation is too large, it will cause subsequent calculations to be error-prone and inefficient in operation. Therefore, after preprocessing, we often perform dimensionality reduction on the word vector matrix of the sentence through distributed representation and obtain the inclusion Vectorized expression of the final sentence of semantic information.

For the hot spot classification model, after preprocessing the text, we use the cosine similarity to calculate the spatial distance of the $TF-IDF$ trained word vectors, and use $K-means$ to achieve text clustering. We set k = 350 starting centroids, use the Euclidean distance to calculate the distance to the remaining elements, and classify the nearest. Finally, iteratively calculates by arithmetic mean to determine the final centroid of K clusters. By comparing the experimental results under multiple k values, the clustering effect is best when k = 350.

For the response evaluation model, since the question and answer content has been given, we will turn the evaluation index into the calculation of the similarity between the two texts. And considering the context semantics, we choose DSSM-LSTM, a deep semantic matching model.

Embedding the results into the LSTM neural network to obtain semantic features, and use Cosine to calculate the similarity of question Q and answer A, question Q and irrelevant answer A, and then normalized by softmax function after summing to obtain the conditional probability distribution.

And through the evaluation of the experiment, the feasibility and effectiveness of the above three models can be obtained.

Keywords: Bi-LSTM Focal-loss intelligent-classification clustering hot-spot $K-means$ DSSM-LSTM reply-evaluation

目录

一、挖掘背景与目标

1.1 挖掘背景

当今社会，人类已经被淹没在知识信息的海洋中。在这海量的信息中，如何从中挖掘出有价值的数据，成为当下的研究热点。尤其是对于各种各样文本数据的处理，如各大新闻门户网站的头条推送等，都与文本数据挖掘的目标紧密联系。而随着文本数据挖掘的发展与创新，自然语言处理技术并不再局限于大众普遍认知的这类娱乐性系统。将文本处理技术应用到智慧政务系统，已成为目前社会治理创新发展的新趋势。微信、微博、市长信箱、阳光热线等网络问政平台逐步成为政府和人民群众进行沟通交流的重要渠道，各类社情民意相关的文本数据量也在不断攀升。对以往来言，主要是依靠人工来进行留言划分和热点整理，然而这种方法对相关部门来说工作量大、效率低，且差错率高。因此，我们希望政务系统能够通过自然语言处理技术变得更加机器化、智能化、精确化。

自然语言处理技术的关键在于让计算机“理解”自然语言，所以自然语言处理又叫做自然语言理解(NLU，NaturalLanguage Understanding)，也称为计算语言学(Computational Linguistics)。一方面它是语言信息处理的一个分支，另一方面它是人工智能(AI, Artificial Intelligence)的核心课题之一。NLU 因拥有直观、简易的优点，成为机器实现文本数据分类的核心应用。其中，神经网络系统因其独特的构造与性质，一直备受关注着，而一些基于统计的聚类算法，特征提取算法也随着创新逐渐适用于许多领域。基于这些成熟的自然语言处理技术，在有关文本的处理上，分类问题、聚类问题等都制定了很好的解决方案。

1.2 挖掘目标

1．根据附件 2 的留言分类数据分析同类数据中的共同或相似特征，即有相同或相近词意表达的词向量，挖掘出能代表数据特征的统计量构成留言分类的划分依据，作为关键词特征，再按照附件 1 的划分体系建立关于留言内容的一级标签分类模型。

2．通过特征词算法计算出附件 3 中留言的特征值权重，并计算距离，为相似度计算提供条件，最后使用聚类算法将留言归类，此时因每句留言的权重不同，所以归类的去向也会不同。并结合归类结果和附件 3 中的有关信息，构建热度评价指标。

3．用相似度算法挖掘出附件 4 中留言内容和答复意见之间的相似性，放入神经网络进行训练，得到可以作为相似评价的特征，且语句间的相似度即作为答复意见的评价指标。

二、文本预处理

2.1 预处理流程图

本文构造的三个模型都针对于中文文本，且每个模型的建立都基于文本预处理后的结果，因此我们将文本预处理作为一个大类提出，避免后文的重复说明。为了更贴近模型我们给出一个实例，表示从数据清洗后到词嵌入前，每条留言的预处理步骤，如下图所示：

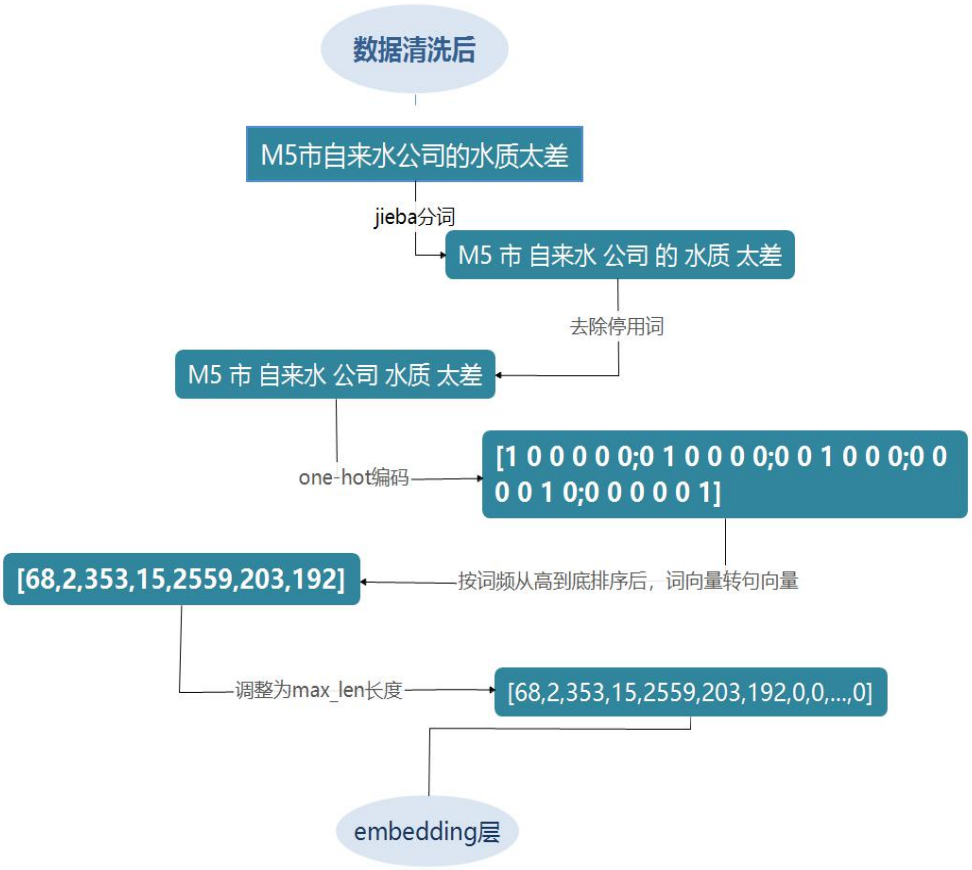


图 1：预处理流程

2.2 流程详解

2.2.1 数据清洗[1]

在众多留言中，可能会因各种原因出现相同的留言、空留言，以及将电话号码、门牌号码等数字符号转化为的 x 序列，为避免上述情况对后续计算热度指标等步骤造成影响，我们将对文本数据进行初步清理。

2.2.2 jieba 分词[2]

中文文本的无界性，使得从中提取词语具有难度，且随着越来越多新词的诞

生，对词典固定化和需要考虑语义的中文文本分词来说是一大难题。为了更好的解决上述问题，我们采用Python中jieba分词的算法——隐马尔可夫模型(HMM)，对数据清洗后的每一句留言进行中文分词。

HMM模型基于隐马尔科夫链，首先用Viterbi算法对每一个字符进行状态标注，然后以中间字为基准对它相邻的两个字间进行动态规划最优路径，按照最优状态序列，找出基于词频的最大切分组合进行分词。

2.2.3 去停用词

在一段文本中，一些出现频率较高，但是本身并没有什么实际的含义并且对找到结果毫无帮助、必须过滤掉的词，被称作停用词，如中文文本中的“你、是、了、的、吗”以及无意义符号“？、！、*、%、“”、”等。停用词会对后面文本数据的处理有着影响，因此在预处理阶段，我们就要除掉这些无关紧要的字词。本文所采用的停用词表，结合四川大学机器智能实验室停用词表[4]与哈工大停用词表[3]。

2.2.4 语言数字化表示

(1) 独热表示[5]（One-Hot Representation）

对于中文文本来说，不像英文字母，可以利用连续的数字对ASCLL码序列进行表示，为了解决这一问题，我们可以采用独热编码形式（One-Hot），将自然语言表示成计算机能够理解的数字形式，如图2所示：

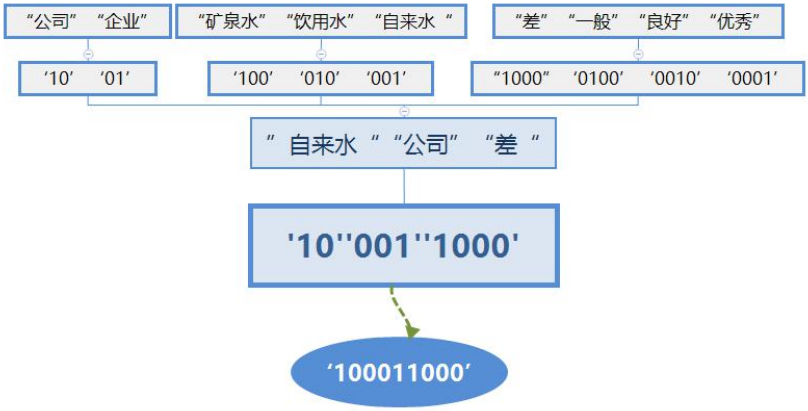


图2：特征数字化

其中，独热编码是将每个词用0和1构成向量来进行表示，这个向量的维度就是词表大小，其中绝大多数元素都为0，只有一个元素为1,而这个维度就代表了当前词。独热编码虽然方便易懂，但它却拥有两个“致命”的缺点：一是One-Hot编码的维数由词表的大小决定，过于稀疏，当词表长度增加时会造成**“维度灾难”**；另一个则是在One-Hot编码下的任意两个词之间是孤立的，不能表达出中文文本词与词之间的语义关系，形成**“词汇鸿沟”**。

(2) 分布式表示（Distributed Representation）

分布式向量或词嵌入向量基本上遵循分布式假设，即具有相似语义的词倾向于具有相似的上下文，因此这些词向量能尝试捕获邻近词的特征。分布式词向量

的主要优点在于它们能捕获单词之间的相似性，并且两个词语语义相近，它们在向量空间的位置也相近。

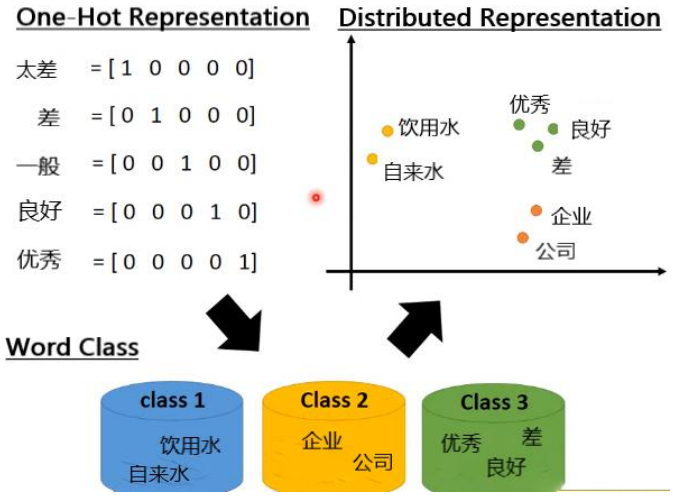


图 3：语言数字化表示

分布式表示中典型的代表是词嵌入（Word Embedding），Embedding[6]可以看作是数学上的一个空间映射，对应到 Word Embedding 概念中可以理解为寻找一个函数或映射，生成新的空间上的表达，把单词 One-Hot 所表达的 X 空间信息映射到 Y 的多维空间向量，且 dim(Y)<dim(X)，达到降维的目的。因此，我们采用这种分布式表示的方法来解决留言文本中单词嵌入的问题，避免“维度灾难”和“词汇鸿沟”的情况发生。后文将通过神经网络降维技术，对独热编码后的结果降维处理。

三、基于 Focal loss 下 Bi-LSTM 的留言分类模型

3.1 构思步骤

基于词组间具有较强的上下文联系，常规的分类算法不能满足对上下文语义的考虑，因此我们选用循环神经网络，同时考虑上下文语义的完整性，我们最终选用 Bi-LSTM 这一模型。且考察具体分类样本的数量特征，选取 Focal loss[12]这一损失函数对交叉熵损失进行改进。

3.1.1 RNN、LSTM 及 Bi-LSTM

(1) RNN[7]

循环神经网络（Recurrent Neural Networks,RNN），是一个拥有对时间序列显示建模能力的神经网络。RNN 相对于传统前馈神经网络的“循环”之处具体表现为 RNN 网络会对之前输入的信息进行记忆归纳，并把这份“记忆”应用于当前的计算。理论上来说，RNN 非常适用于处理序列数据、并且可以支持对任意长度的序列处理，能够较好地表征上下文的语义。

举一个实例来帮助理解，例如对“M 市水质真差”一句文本进行识别时，传统的神经网络在对“差”进行识别时不会参考“M 市”或者“水质”的语义信息，只能对当前时刻下的词向量信号“差”进行处理分析；而对于循环神经网络而言，它会根据历史数据信息结合“差”词向量所处的位置，参考改时间位置前面的词向量信号“水质”，更加容易的识别出语义为“真”的词向量信号。下图所示为 RNN 的链式形式：

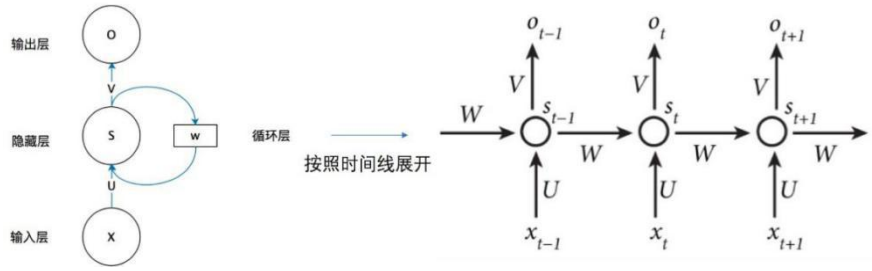


图 4：简单 RNN 网络

其中，将上述语句具象表示，即 x_{t-3} 代表“M 市”、x_{t-2} 代表“水质”、x_{t-1} 代表“太”、x_t 代表“差”，在循环神经网络里当前时刻 t 的隐层状态 s_t 与前一时刻 $t-1$ 的隐藏层传递下来的信息 s_{t-1} 和当前时刻的输入 x_t 相关。

虽然 RNN 具有记忆功能，但随着时间间隔的增大，相关信息与当前时刻信息的距离逐渐增大时，RNN 的记忆里达到极限，为了继续接收下一时刻的输入信息，会遗忘掉一部分时间间隔过长的初始信息，最终导致语义丢失。同时考虑到 RNN 有**过拟合**和**梯度消失、梯度爆炸**的缺点，为了避免这些问题对模型造成影响，我们选用 LSTM 这一 RNN 变种来替代。

(2) LSTM[8]

1997 年由 Hochreiter 和 Schmidhuber 提出了一种称作长短时间记忆单元(Long Short-Term Memory，LSTM)的特殊结构循环神经网络，最终 Graves 对该结构进行了进一步改良和推广，获得了巨大成功。在循环神经网络的结构上，LSTM 加入了输入门层、输出门层和遗忘门层，主要功能如下：

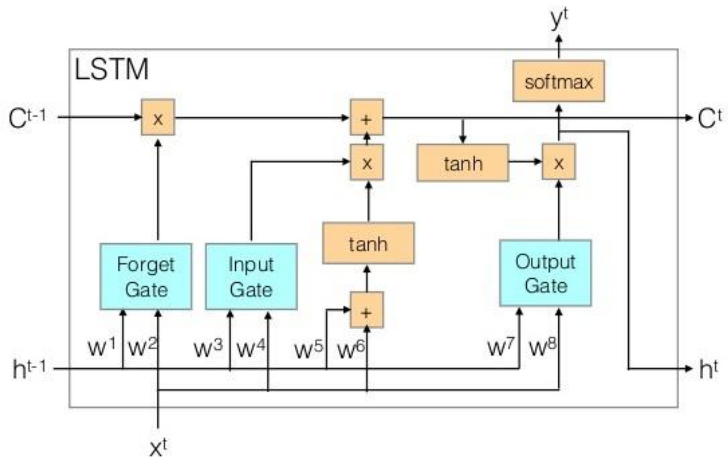


图 5：LSTM 细胞结构

（在 t 时刻 LSTM 细胞内，各结构表示：遗忘门（f_t）、输入门（i_t）、输出门（o_t），$t-1$时刻输入向量（h_{t-1}）、隐层状态（c_{t-1}），以及偏置项（b））

遗忘门层（Forget Gate）控制神经元是否摒弃当前节点所保存的历史时刻信息，该门通过一种称为“peephole”的连接方式和内部神经元相连这种连接方式可以提高 LSTM 在时间精度及计算内部状态相关研究的学习能力，若门取值为‘1’则保留以往的历史信息，相反若门取值为‘0’，则清除当前节点所保存的历史信息。其公式为：

$$f_t = \delta(W_f \cdot [h_{t-1}, x_t] + b_f)$$

输入门层（Input Gate）控制是否允许输入信息输入到当前网络隐层节点中，如果门取值为‘1’则输入门打开允许输入数据，相反若门取值为‘0’，则输入门关闭不允许数据输入。其公式为：

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\bar{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

$$c_t = f_t * c_{t-1} + i_t * \bar{c}_t$$

输出门层（Output Gate）控制经过当前节点的数据是否传递给下一个节点，如果门取值为‘1’，则当前节点的数据会传递给下一个节点，相反若门取值为‘0’，则不传递该节点信息。其公式为：

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(c_t)$$

通过训练后阈值的确定三扇门，可将远距离时间步长的有效数据信息保存下

来，在此模型中代表具有分类属性的特征词向量，而无效的数据信息则被摒弃，即权重低不能直接决定分类类别的一般词向量，且其对应的参数也无需保存，最终前面提及的梯度消失问题得到遏制。

(3) Bi-LSTM[10]

LSTM 相对于传统 RNN 在分类性能上有很好的改善，但由于 LSTM 只能根据时间步长**单向流动**，当前时刻t的输出只能由前面$t-1$时刻来决定，而在文本分类模型中，词语的语义关乎于上下文，因此 LSTM 仍然**遗失未来的信息**，语义处理上仍然有缺陷。

双向 LSTM 是传统 LSTM 的扩展，可以改善序列分类问题上的模型性能。在输入序列的所有时间步均可用的问题中，双向 LSTM 在输入序列上训练两个 LSTM，第一个按原样，第二个按输入序列的反向副本，可以为神经网络提供其他上下文，并导致对问题的更快，甚至更充分的学习。对比 LSTM 而言，Bi-LSTM 完成了上下文的语意连贯问题，可以对句子的深层语义进行编码，且避免丢失句尾的语义信息。且结构示意图如下：

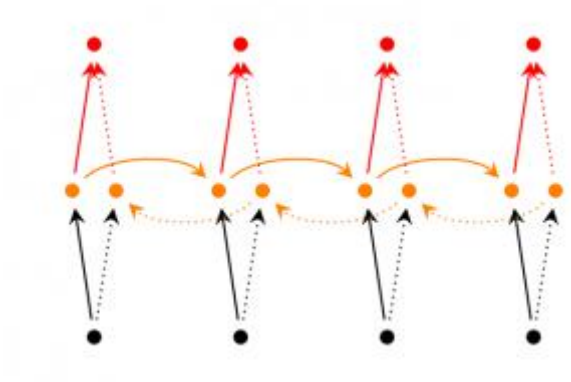


图 6：Bi-LSTM 网络结构

3.1.2 Focal loss 损失函数[11]

在用于文本分类的原始数据中，会存在部分类别的数据数量很多，而部分类别的数据数量很少的情况，这种类别数量不均衡的问题对文本总数小模型来说易造成两个后果：

1) **模型训练效率低，因为大多数训练样本都是容易产生负面影响，即负样本，导致模型进行无效学习。**
2) **负样本数量太大，占总的 loss 函数输入参数的大部分，而且多是容易分类的，因此使得模型的优化方向（即 loss 函数的梯度下降方向）并不是我们所希望的那样。**

因此针对类别不均衡问题，一种新的损失函数：focal loss 问世。这个损失函数是在标准交叉熵损失基础上修改而来的，我们先给出**二分类交叉熵损失**：

$$L=-y\log y'-(1-y)\log(1-y')=\begin{cases}-\log y', y=1\\-\log(1-y'), y=0\end{cases}$$

其中 y 是真实样本的标签（1 正 0 负），y' 是经过 *sigmoid* 激活函数的预测输出（数值在 0-1 之间）。可见普通的交叉熵对于正样本而言，输出概率越大损失越小。对于负样本而言，输出概率越小则损失越小。此时的损失函数在大量简单样本的迭代过程中比较缓慢且可能无法优化至最优。其次对于 **Focal loss 函数**：

$$L_{fl}=\begin{cases}-(1-y')^{\gamma}\log y', y=1\\-(1-\alpha)y'^{\gamma}\log(1-y'), y=0\end{cases}$$

首先在原有的基础上加了一个因子，其中 Gamma>0 使得减少易分类样本的损失，使得模型更关注于困难的、错分的样本。在此基础上，再引入一个平衡因子 Alpha，用来平衡正负样本本身的数量比例不均（即类别不均衡）。通过 Alpha 和 Gamma 两个参数的调节，可使得模型的效果达到最优。

3.2 具体流程

在上述对循环神经网络的阐述中，我们推出 Bi-LSTM（双向长短时记忆模型）具有良好的性能，以及在文本分类上拥有的优势。故此，我们建立基于 Focal loss 的 Bi-LSTM 智能分类模型，模型架构如图 7 所示：

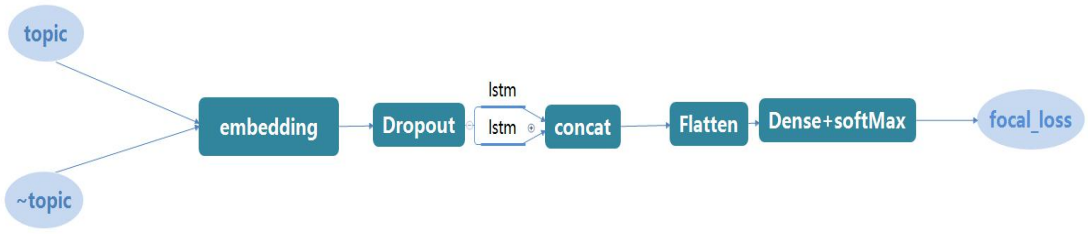


图 7：留言分类模型流程图

Step1：Topic/~Topic——预处理后代表示留言主题的固定长度序列，Topic 为正向，~Topic 为逆向；

Step2：Embedding 层——将序列进行词嵌入，通过对神经网络的多次训练，更新每个嵌入的向量，将词与词之间的相似性用向量表示出来，使得词语之间的关系可视化；

Step3：Dropout——输入向量进入 LSTM 前，对数据随机置零，让模型用少量的特征来学习，且每次被学习的特征不同，使模型的预测面更广，避免侧重于部分特征，导致过拟合；

Step4：Bi-LSTM 层——通过正向 LSTM 与逆向 LSTM 同时训练，对 7 类 cat-id 下的留言进行学习，更新词向量的权重，得到各类别下的特征词；

Step5：Concat——将正向 cell 和逆向 cell 输出的两个 output 进行拼接；

Step6:Dropout——从 LSTM 输出后，再次进行同③的操作，进一步降低训练集的过拟合；

Step7：Flatten——将多维向量转成一维向量，对数据进行降维，为后续的全连接而准备；

Step8：Dense+softmax——将单个神经元的输出向量链接起来，并用 softmax 函数激活；

Step9：Focal loss——通过减少易分类样本的权重，使得模型在训练时更专注于难分类的样本。

该模型为 n vs 1 分类模型，即只保留最后一次的输出值，输出结果为预测的分类结果。然后画出该模型分类后的混淆矩阵，即可得到测试集的可视化结果。

四、基于 TF-IDF＋K-Means 的热点挖掘模型

4.1 构思步骤

由于附件 3 中的留言数据没有附带任何的标签，因此我们将这种特殊的归类问题转化为无监督学习的文本聚类模型来解决。常见的聚类算法有 $K-means$ 、$BIRCH$ 、GMM 、$GAAC$ 等，而 $K-means$ 算法时间复杂度低且实现的效果较好，最终我们采用 $TF-IDF+K-means$ 这一模型。基本思想是 $TF-IDF$ 处理词向量将其转换为数值矩阵的形式，并根据余弦相似度计算留言在空间上的余弦距离的，再使用 $K-means$ 算法实现文本聚类，最后将聚类结果进行数量排序，利用热度评价公式得到最终的热点问题明细表。

4.1.1 TF-IDF 权重策略[13]

为了便于后续的 $K\text{-}means$ 聚类实现，首先我们需要计算出文本之间的相似度，即两个留言的相似程度。由于计算机无法直接计算自然语言之间的相似程度，此时，需要将文档相似度问题转换为数值向量矩阵问题，可以通过 VSM 向量空间模型来存储每个文档的词频和权重，特征抽取完后，因为每个词语对实体的贡献度不同，所以需要对这些词语赋予不同的权重。计算词频在向量中的权重方法——$TF-IDF$ 。

TF（词频）：Term-frequency，即关键词词频，表示词条（关键词）在文本中出现的频率。公式为：

$$TF=\frac{\text{某一类中词条}w\text{出现的次数}}{\text{该类中所有词条的数目}}$$

IDF（逆向文本频率）：Inverse document frequency，即逆向文本频率，用于衡量关键词权重的指数，公式为：

$$IDF=\log(\frac{|D|}{|w\in d|})$$

$|D|$表示所有文档的数目，$|w\in d|$表示包含词条w的文档数目。

$TF-IDF$:它表示TF（词频）和IDF（逆向文档频率）的乘积

$$TF-IDF=TF\times IDF$$

4.1.2 余弦相似度[14]

余弦相似度就是将一个向量空间中两个向量夹角的余弦值作为衡量两个个体之间差异的大小。把 1 设为相同，0 设为不同，那么相似度的值就是在 0~1 之间。余弦相似度的特点是余弦值接近 1，夹角趋于 0，表明两个向量在空间上的余弦距离越相近，即两个留言所表述的问题越相近。将$TF-IDF$处理后的词向量带入公式则有：

$$\cos(\theta)=\frac{\sum_{i=1}^{n}(x_i\times y_i)}{\sqrt{\sum_{i=1}^{n}(x_i)^2}\times\sqrt{\sum_{i=1}^{n}(y_i)^2}}$$

4.1.3 $K-means$ 聚类算法[15]

$K-means$是一种无监督的文本聚类算法，文本聚类即是将有着各自特征的文本进行分类，并利用余弦相似度将具有相同或相似属性的文本聚类在一起。其中k表示类别数，$means$表示均值，它通过均值来对数据点进行聚类。$K-means$算法通过预先设定的k值及每个类别的初始质心对相似的数据点进行划分。并通过划分后的均值迭代优化获得最优的聚类结果。

假设簇划分为（$C_1,C_2,...C_k$），则我们的目标是最小化平方误差J从而得到最好的聚类效果：

$$J=\sum_{i-1}^{k}\sum_{x\in C_i}(x_i-\mu_i)$$

其中μ_i是簇C_i的均值向量，有时也成为质心：

$$\mu_i = \frac{1}{|C_i|}\sum_{x \in C_i} x$$

4.2 具体流程

基于上述算法的阐述我们使用 $TF-IDF$ 权重策略处理词向量，将处理后的词向量放入 $K-means$ 模型中，基于 $TF-IDF+K-means$ 的热点挖掘问题，模型构架如图 8 所示：

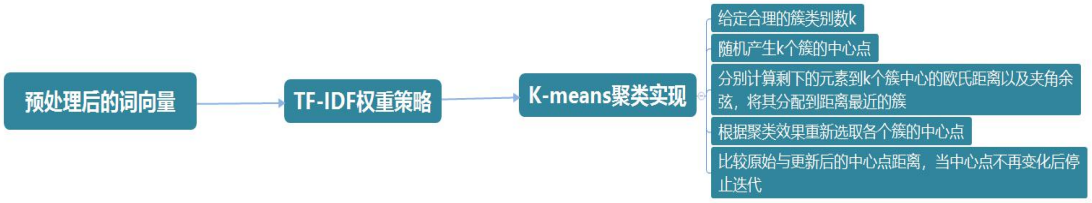


图 8：热点挖掘模型流程图

Step1：首先根据对数据的先验经验或交叉验证选择一个合适的 k 值（即 n_cluster 类别数）；

Step2：随机选择 k 个点作为 k 个簇的起始质心；

Step3：分别计算剩下的元素到 k 个簇中心的欧氏距离，将这些元素分别划归到距离最近的簇；

Step4：根据聚类结果，重新计算 k 个簇各自的中心点，计算方法是取簇中所有元素各自维度的算术平均值；

Step5：将 k 个簇的新中心点距离与原有中心点距离进行比较，如果中心点不再变化或聚类结果不再变化或迭代计算轮次达到最大值，则输出聚类结果，否则回到第 3 步继续迭代计算。

4.3 热度指标定义

及时发现热点问题，有助于相关部门进行有针对性地处理，提升服务效率。在本文中我们根据特定地点和人群的聚类结果以及某一类留言总的点赞数与反对数和作为主要的热度评价指标，得到如下的热度评价公式：

$$\begin{cases} Hot_i = 1000 * p_i * w_{hot_i}, p_i = \dfrac{s_i}{\sum_i^k s_i}, s = \text{点赞数} + \text{反对数} \\ w_{hot_i} = \dfrac{M_i}{\sum_1^k M_i}，\ \text{其中} M \text{表示热度排名为} i \text{的留言数} \end{cases}$$

上述公式中热点问题的评价指标考虑到了相似留言的数目以及点赞、反对

量，能够较好地对实际生活中的问题进行热度评价。

五、基于 DSSM-LSTM 的答复评价模型

5.1 构思步骤

此模型的构思是基于常规问答模型——“提出问题找出与其对应的回答”的逆向思考，因此我们将其归类为相似度计算的评价模型，即评价问题与回答的相关性。我们采用基于 DSSM[17]这一深度语义匹配模型作为模型基础，同时考虑语义问题，我们最终选择 LSTM 来构造 DSSM 模型，完成语句相似度的评价。

5.1.1 DSSM

DSSM 是在检索场景下使用点击数据来训练语义层次的匹配，简单的来说，传统检索场景下的匹配主要有:TF-IDF、BM25 等字面匹配以及使用 LSA 类模型进行的语义匹配，但是效果都不佳。图 9 是 DSSM 的训练框架图：

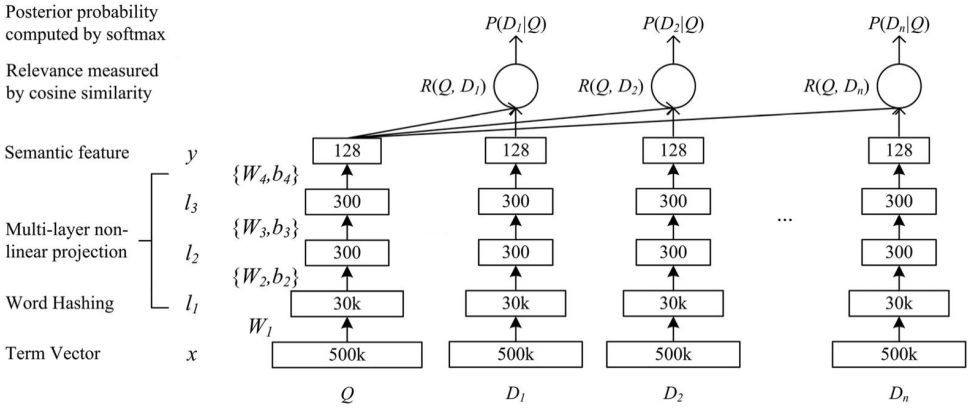


图 9：DSSM 训练框架图

输入问题 Q 以及与其相关的会回答 Di 的 One-Hot 表示，再放入 Word Hashing 层降维，降维后的词向量进入神经网络去训练，得到更深层次包含语义特征的词向量，最后使用 cosine 计算相似度，其中和表示 DNN 输出层问句 Q 和文档 Di 的低维语义向量，然后通过 softmax 得到条件概率分布。

5.1.2 Word2vec[16]

考虑为了记录输入句子的上下文，LSTM 这个模型则更为合适。其中，LSTM 中的 Embedding 层采用了 word2vec 训练好的词向量库，对独热表示后的向量进行分布式表示。

word2vec 是 Google 在 2013 年开源的一款将词表征为实数值向量的高效工具，是典型的分布式表示，包含了 CBOW(连续词袋模型)和 Skip-Gram 两种模型。我们采用 Skip-Gram 来训练我们的词向量库，且语料库采用的维基百科简体中文进行的模型训练。

5.2 具体流程

基于上述的简述，我们建立 LSTM 下的 DSSM 模型，将训练集问题、答案和无关答案带入模型中得到相似特征，将相似度的结果作为答复意见的评价指标。流程图如下图所示：

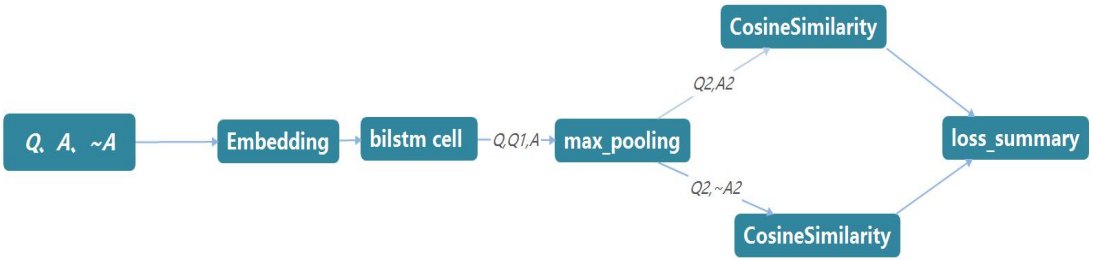


图 10：答复评价模型流程图

Step1：Q、A、~A——预处理后用独热编码表示的问题 Q、正确答案 A、无关答案 ~A;

Step2:Embedding——通过 Word2vec 训练后的词向量库进行降维处理，进行词嵌入，解决 One-Hot 输入词表太大和 oov 问题；

Step3：biLSTMcell——进入神经网络进行训练，进一步得到和语义有关的词向量 Q1、A1、~A1；

Step4：max_pooling＋tanh——最大池化在处理，对数据进一步浓缩降维，且将结果用激活函数 tanh 进行激活得到进一步词向量表达 Q2、A2、~A2；

Step5：CosineSimilarity——计算 Q2 和 A2 以及 Q2 和~A2 的相似度；

Step6：loss_summary——对 step5 的结果分别计算损失，并求和。

最终得到的相似度则为问题 Q 与正确答案 A 的相似度，即作为留言提问与留言回答相关与否的评价指标。

六、实验评估

本节中我们将通过模型之间的性能对比，证明我们建立的模型的有效性和优化性，同时给出本实验中部分所取参数值与附近值的比较结果，并以图表的方式进行展示。

6.1 实验需求

6.1.1 实验平台

构建模型的实验环境如表 1 所示：

表 1：实验环境

操作系统	DirectX 12
内存容量	8GB
固态硬盘容量	128GB
GPU	GXT 1050
CPU	Inter Core i5-8300H
Python	3.6.7
Tensorflow（模型一）	2.2.0rc4
Tensorflow（模型二）	1.11.0

本实验主要编程软件为 Python，Excel 中的函数库做辅助。其中使用到的 Python 库有 13 个：

tensorflow、jieba、numpy、keras、pandas、codes、TfidfTransformer、CountVectorizer、xlrd、seaborn、sklearn、matplot

6.1.2 评价指标

本模型主要采用的评价指标有精确率（Precision）、召回率（Recall）、F1-Score、准确率（Accuracy），根据本文模型的特征有具象定义如下：

- 精确率（Precision）：同类别下，分配正确的留言与分配到此类别下的全部留言的比值，即考察有无分配错误的留言。
- 召回率（Recall）：同类别下，分配正确的留言与属于该类别的所有留言的比值，即考察分类中有无遗漏的留言。
- F1-Score：为精确率和召回率的调和平均值，适用于对留言数据在不同类别下数据不平衡的分类模型进行性能评估。
- 准确率（Accuracy）：所有类别下，分配正确的留言总和与所有参与分类的留言的比值，即一定程度上可表示模型的准确率，而对于数据不平衡的分类模型则不太适合。

6.1.3 相关参数设置

在模型训练时候需要对部分参数进行调整以确定最佳值，实验中模型一、二、三参数统计表如下：

表 2：模型参数设置统计表

参数	参数值
Batch_size	128
Dense	7
Epochs	8
Max_len	500
Alpha	0.25
Gamma	7
Input embedding_dim	5001
Output embedding dim	128
n_cluster	60

6.2 实验结果与分析

6.2.1 留言分类模型

1.实验结果

我们给出在该模型下测试集结果的混淆矩阵：

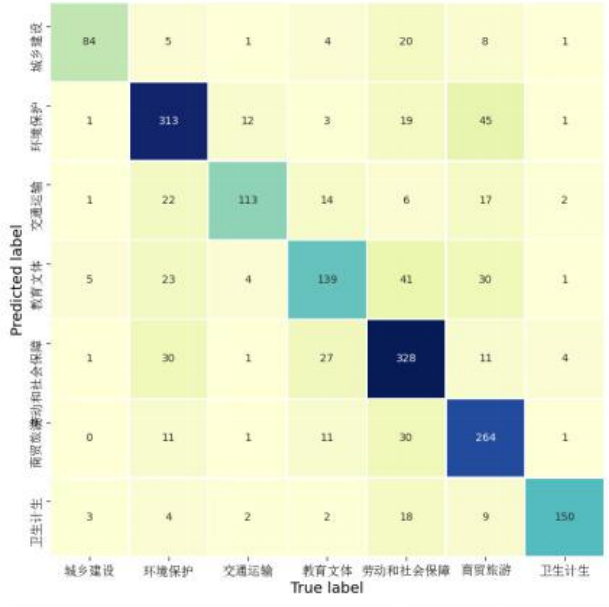


图 11：模型一测试集结果的混淆矩阵

其中横坐标是留言真实标签，纵坐标是模型预测下对该留言进行分类的标签，并且颜色越深代表对应两个指标下留言的数量越多。在上述的混淆矩阵中，主对角线上是分类正确的留言数量，易得该对角线上的文本数量总和占总文本数量的大比例，因此可证明本模型的有效性。

2.结果分析

- 比较 LSTM 与 Bi-LSTM 在 Focal loss 取值 Alpha=0.25、Gamma=7 下模型的性能

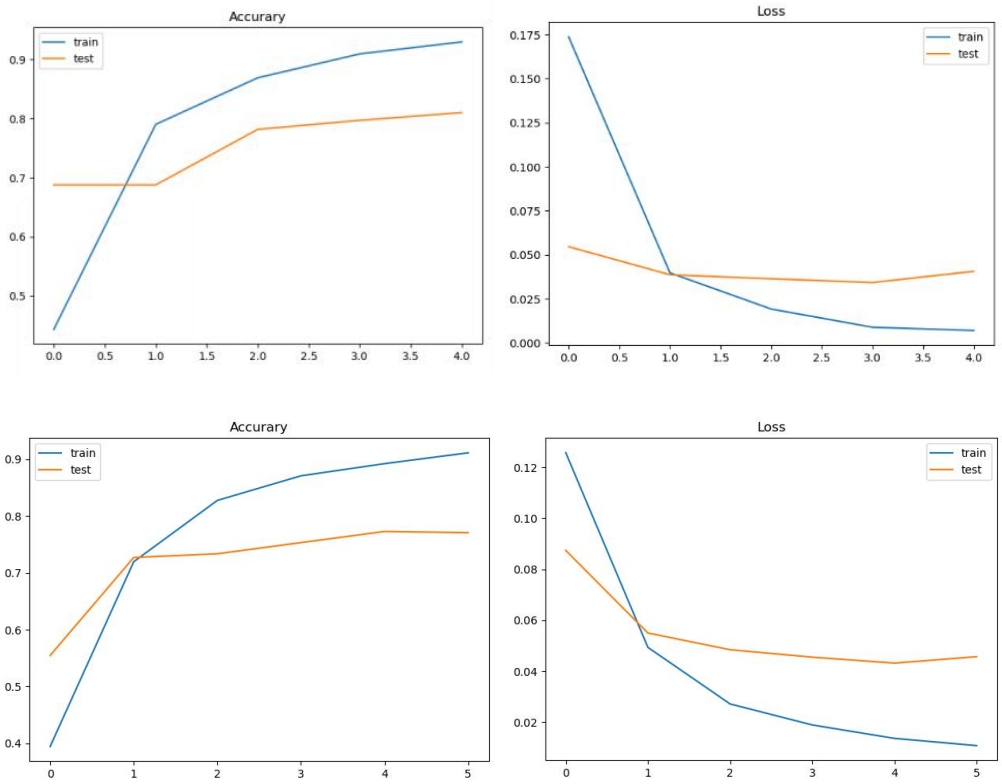


图 12：Bi-LSTM（上）与 LSTM（下）准确率与损失率

在两种模型下训练集的指标相似，故我们以下针对测试集进行考察。两者准确率相近，最终稳定值都在 0.7 到 0.8 之间，Bi-LSTM 相比 LSTM 整体浮动性较小；两者损失率也大致相近，LSTM 最终稳定值在 0.04 到 0.06 之间，Bi-LSTM 稳定在 0.025 到 0.05 之间，而 Bi-LSTM 整体仍比 LSTM 更加稳定。

且 7 个类别下评价指标的平均值如表 3 所示：

表 3：Bi-LSTM 与 LSTM 评价指标

	precision	recall	f1-score	accuracy
Bi-LSTM	0.81	0.83	0.82	0.82
LSTM	0.75	0.78	0.75	0.78

Bi-LSTM 下的精确率、召回率、f1-score、准确率均比 LSTM 高，且 Bi-LSTM 平均每个指标都在 0.8 以上，而 LSTM 在 0.75 附近。

因此结合上述两项评价可见我们的模型在 Bi-LSTM 中的性能优于 LSTM，且在基于语义的分类模型下，即不考虑基于统计的传统分类模型，可以证明本模型

的优化性。

■ **在 Bi-LSTM 下损失函数设为 Focal loss 和交叉熵时模型的性能**

在该论文建立的模型中我们采用 Alpha=0.25 和 Gamma=7，本模型下交叉熵和 Focal loss 的损失值的图像如下：

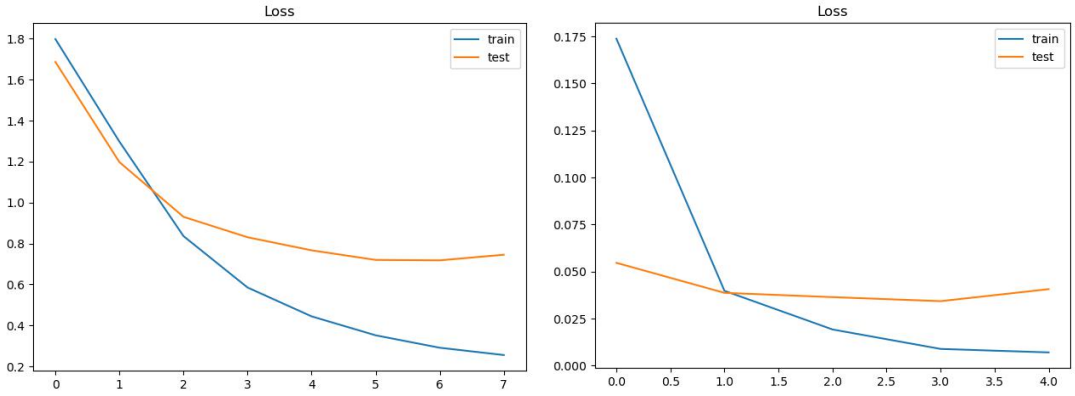


图 13：交叉熵（左）和 Focal loss（右）的损失值图像

上图横坐标代表迭代次数，纵坐标代表损失值，明显可见测试集在 Focal loss 下的损失值在 0.025 到 0.075 之间，且曲线整体趋于稳定；而在交叉熵函数下测试集的损失值在 0.6 到 0.8 之间，曲线在前 3 次迭代下相对陡峭，之后趋于稳定。可见得，focal loss 在降低测试集的损失程度上展现了它优异的性能，优化了模型的分类效果。

在 7 个类别下的 F1-Score 值如表 4 所示：

表 4：交叉熵与 Focal loss 的 F1-Score 值

	0	1	2	3	4	5	6
交叉熵	0.73	0.84	0.68	0.66	0.82	0.84	0.85
Focal loss	0.77	0.84	0.81	0.70	0.79	0.91	0.91

其中 Focal loss 下每个类别的 F1-Score 值比交叉熵函数运行后的 F1-Score 值相对较高，且提高了模型对类别 2、类别 5 和类别 6 这些文本数量较少的类别分类性能，指标明显上升。

6.2.2 热点挖掘模型

1. 实验结果

在模型二下部分留言聚类后的结果展示，如下表所示：

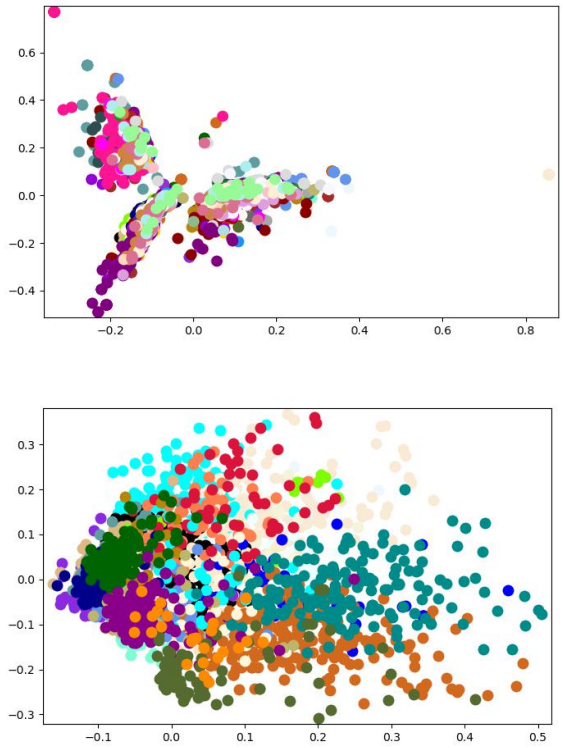
表 5：部分聚类结果

Label	留言主题
209	A7 县城泉塘街道圣力华苑小区业主想办理“两证合一”该走什么程序
209	请求解决 A7 县圣力华苑小区房屋两证合一的问题
209	A7 县圣力华苑房屋两证合一找什么部门可以解决
210	反映 A 市金毛湾配套入学的问题
210	请 A3 区协调解决旭辉御府业主子女入学问题
210	A 市旭辉御府业主子女的配套入学资格呢
232	A 市值赣州高铁对绿地海外滩二期小区的影响太大了
232	渝长厦高铁的长赣高铁征地路线对 A6 区周边小区影响巨大
232	A4 区绿地海外滩小区距长赣高铁最近只有 30 米不到，合适吗?

在上表的热点留言中，每一个类别下的留言的相关性较高，不同类别下留言内容的相关性较低。

2. 结果分析

为了更直观的观察聚类的结果，我们将聚类结果可视化：



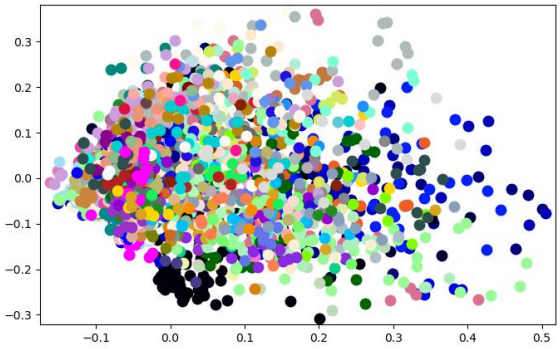


图 14：聚类结果

从上图 k=50（上）、k=100（中）、k=350（下）三个值下的聚类图可知：当 k=50 时候，聚类结果越紧密，此时会导致不相关留言被分到一类，聚类效果差；当 k=100 时，留言有明显的划分，但是仍会出现部分相关程度不高的留言被划分到同一类，如中间图所示，同种颜色的点在空间上虽然靠近，却占据了很大部分面积。与上述两种取值相比，当 k=350 时，分到一起的留言相关程度较高。故对 k 值结果的多次比较下，我们最终将 k 值设为 350，此时的分类效果较良。

七、模型评价及模型优化

7.1 模型优缺点

1.基于 Focal loss 下 Bi-LSTM 分类模型

- **优点：**

①基于双向长短时循环神经网络（Bi-LSTM）来构建模型，考虑了中文文本中的语义信息，并且 Bi-LSTM 能够保留句尾的信息，可以更好地表征完整的句子语义。

②采用改进的损失函数，提升了数量少少、难分类的样本的准确率。

- **缺点：**

①对于少数包含大量无意义语句的长文本而言，在对词语进行向量化处理及降维时特征值选取过多，导致进行特征值匹配时出现错误分类的情况。

②无法实现一个留言有同属于多种类别情况时的分类

2.基于 $TF-IDF+Kmeans$ 热点挖掘模型

- **优点：**

①使用 $TF-IDF+K-means$ 解决聚类问题的一种改进算法，实现简、快速。

②对处理大数据集，该算法保持可伸缩性和高效性。

③当簇接近高斯分布时，它的效果较好。且在代码实现中加入 PCA 对数据进

行处理，更好的表征文本的中心语义。

- **缺点：**

①*K-means* 聚类难以确定合适的n_cluster类别数且容易得到局部最优解。

②不能对地点、人群进行多样化识别（如：魅力之城小区、魅力之城、魅城）

3.基于 DSSM-LSTM 的答复评价模型

- **优点：**

①相似度算法与神经网络的结合，尽可能的保留了词与词之间的语义信息。

②其中包含 Word2vec，可以对高维向量进行降维。

③可以判断问答之间的相关性（答复意见的内容是否与问题内容有关）

④可以判断答复的可解释性（答复意见中有无内容的相关解释）

- **缺点：**

①不能判断问答之间的完整性（提出的问题是否都有相应的解释）

7.2 模型优化

1.模型一优化：添加 Attention 层

加入一个注意力层可以帮助神经网络更注意到主要起到分类作用的特征向量，将其他次要特征值的权重降低，更有助于对长文本的分类。

2.模型二优化：增加基于 HMM 命名实体识别算法

由于需根据特定地点和人群对留言进行分类，因此可以利用基于 HMM 的命名实体识别模型对留言序列进行实体标注，从而提取出特定地区、人群，并将其分为不同类别。另外，使用 word2vec 训练词向量，最后再使用 *K-means* 对留言进行聚类，增大模型对语义的层面的理解，是模型更智能化。

3.模型三优化：将 DSSM-LSTM 换成 CDSSM 模型

CDSSM 模型相比 DSSM-LSTM 模型来说加入了卷积层，能更好的对问答语句中的高维度特征向量进行降维，并通过 DNN 卷积神经网络的 FC 层计算语义向量，保留语句原有的语义信息。对问答类的文本进行语义的匹配工作是非常合适的选择。

八、参考文献

[1]https://github.com/mosu027/nlp-utils-ch

[2]https://github.com/fxsjy/jieba

[3]https://github.com/goto456/stopwords

[4]https://github.com/baipengyan/Chinese-StopWords

[5]https://github.com/asyncins/OnehotCode

[6]Bonelee.神经网络中 embedding 的作用.[EB/OL]
https://www.cnblogs.com/bonelee/p/7904495.html 2020.05.02

[7]fengbingchun.循环神经网络(RNN)简介.[EB/OL]
https://blog.csdn.net/fengbingchun/article/details/82288537
2020.05.01

[8]AI 深入浅出.LSTM 神经网络
[EB/OL]https://blog.csdn.net/m0epnwstyk4/article/details/79124800
2020.05.01.

[9]Paishen. 基于 LSTM 中文文本多分类实战
[EB/OL]https://blog.csdn.net/weixin_42608414/article/details/89856566?depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-2&utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-2 2020.05.02

[10]https://github.com/albertwy/BiLSTM

[11]Qinhaonan.解剖 focal loss 损失函数
[EB/OL].https://blog.csdn.net/Gentleman_Qin/article/details/87343004?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-7&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-7 2020.05.03

[12]Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár. Focal Loss for Dense Object Detection[J].2017

[13]https://github.com/mayank408/TFIDF

[14]京东云成都.余弦计算相似度度量
https://blog.csdn.net/u012160689/article/details/15341303.[EB/OL].2013.11.11
[15]memory of seven seconds.python 实现完整的 K-means 文本聚类算法.
https://blog.csdn.net/weixin_43718084/article/details/90231783
[EB/OL].2019.05.15
[16]https://dumps.wikimedia.org/
[17]https://github.com/makeplanetoheaven/NlpModel/tree/master/SimNet/TransformerDSSM/TrainData
[18]https://github.com/cjfcsjt/QA-matching

附录

（一）Focal loss 函数

Focal Loss 是 Kaiming He 和 RBG 在 2017 年的“Focal Loss for Dense Object Detection”论文中所提出的一种新的 Loss Function，Focal Loss 主要是为了解决样本类别不均衡问题，下图是 Focal loss 原文中的截图，其中是关于这个损失函数的中心思想：

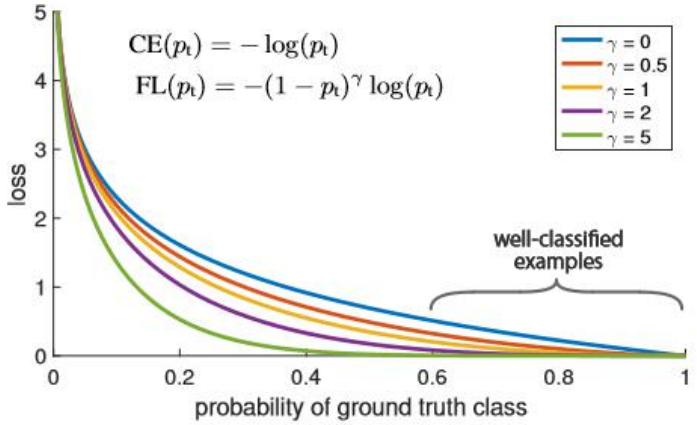


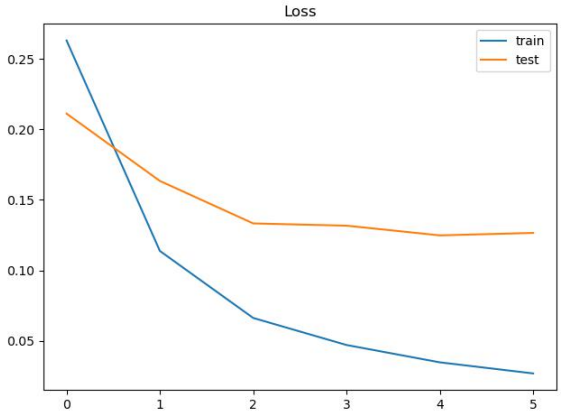
Figure 1. We propose a novel loss we term the *Focal Loss* that adds a factor $(1 - p_t)^{\gamma}$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_t > .5$), putting more focus on hard, misclassified examples. As our experiments will demonstrate, the proposed focal loss enables training highly accurate dense object detectors in the presence of vast numbers of easy background examples.

论文链接：https://arxiv.org/pdf/1708.02002.pdf

（二）Focal loss 的 Gamma 选值

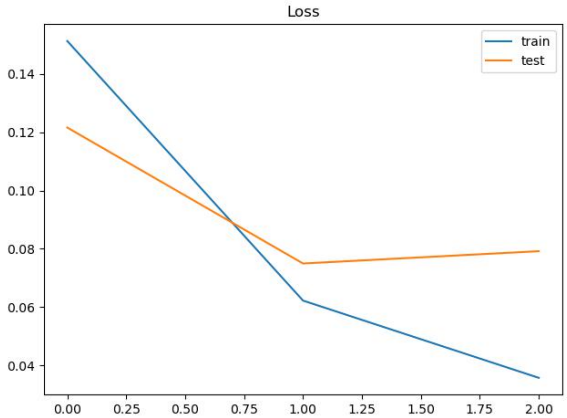
在模型一中我们使用的损失函数中的 Gamma 参数需要通过实践确认，由于 Alpha 用来平衡正负样本本身的数量比例不均，通常都取值都为 0.25，故我们比较当 Alpha=0.25 时 Gamma 分别取合理值 2,5,7 时模型的损失函数和模型的评价指标的值。

- Gamma=2



	precision	recall	f1-score	support
0	0.76	0.78	0.77	120
1	0.92	0.74	0.82	486
2	0.59	0.76	0.67	136
3	0.61	0.71	0.65	209
4	0.83	0.79	0.81	422
5	0.80	0.81	0.81	315
6	0.77	0.93	0.84	155
accuracy			0.78	1843
macro avg	0.75	0.79	0.77	1843
weighted avg	0.80	0.78	0.78	1843

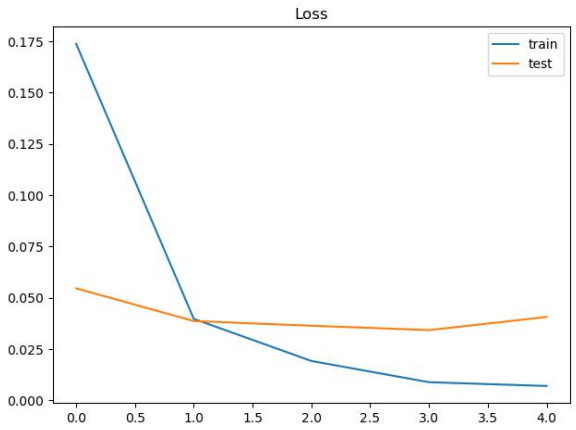
- Gamma=5



	precision	recall	f1-score	support
0	0.74	0.60	0.66	151
1	0.86	0.82	0.84	412
2	0.69	0.80	0.74	151
3	0.51	0.76	0.61	164
4	0.87	0.65	0.75	537
5	0.71	0.88	0.79	257
6	0.78	0.85	0.81	171
accuracy			0.76	1843

macro avg	0.74	0.77	0.74	1843
weighted avg	0.78	0.76	0.76	1843

- Gamma=7



	precision	recall	f1-score	support
0	0.73	0.82	0.77	110
1	0.92	0.77	0.84	468
2	0.85	0.77	0.81	192
3	0.69	0.71	0.70	236
4	0.75	0.83	0.79	366
5	0.88	0.94	0.91	297
6	0.87	0.94	0.91	174
accuracy			0.82	1843
macro avg	0.81	0.83	0.82	1843
weighted avg	0.83	0.82	0.82	1843

由上述的图表可见当 Gamma 取 7 时，模型的损失函数平均值最小为 0.05 左右，且曲线的走势相对平稳，代表模型性能稳定；在四个指标中也可见此时精确率、召回率、F1-Score 以及准确率都在 0.8 以上，相比 Gamma 取值为 2 和 5 时更高。因此，Gamma=7 为最优值。