古代文本断代模型

断代是指判断古代文本大致年代的过程，本部分使用Bi-LSTM神经网络分析输入文本的深层语言结构，从而实现古代文本断代的目的。本部分将详细介绍断代模型的实验数据来源、各部分结构组成，并做实验进行性能分析。

### 数据来源及预处理

实验所用的数据集为从网络的开放数据库下载的不同年代的古籍。根据古籍所处具体时期的不同，我们从各个时期中选择了部分书籍进行实验。将其分为成了不连续的几个时间段：春秋战国时期、后汉时期、南北朝时期、宋朝时期及明清时期五个时间段并分别使用T1、T2、T3、T4以及T5表示，对应关系如下表所示。

**表3-1 时间标签与年代对应表**

|  |  |  |
| --- | --- | --- |
| 时间标签 | 年份 | 时期 |
| T1 | 公元前770-公元前221 | 春秋战国 |
| T2 | 公元25年-公元220年 | 后汉 |
| T3 | 公元420年-公元589年 | 南北朝 |
| T4 | 公元960-公元1279年 | 宋 |
| T5 | 公元1368年-公元1912年 | 明清 |

在上述的每个时期中，我们分别选择著成于当前时代的古籍作为训练集，各个时期具体书目如下表所示，总计348万字。

**表3-2 各时期书目表**

|  |  |  |
| --- | --- | --- |
| 时间标签 | 书名 | 字数（万字） |
| T1 | 《春秋公羊传》《论语》  《春秋左传》《国语》  《道德经》《尚书》  《吕氏春秋》 | 90 |
| T2 | 《孔雀东南飞》《太平经》  《申鉴》《神农本草经》  《东观汉记》《论衡》  《素问》《白虎通义》 | 58 |
| T3 | 《洛阳伽蓝记》《颜氏家训》  《齐民要术》《世说新语》 | 43 |
| T4 | 《大唐三藏取经诗话》  《河南程氐遗书》  《进大慧禅师语录奏劄》  《东京梦华录》《梦溪笔谈》 | 57 |
| T5 | 《红楼梦》《聊斋志异》  《欹枕集》《庚巳编》  《清平山堂话本》 | 100 |

### 模型结构

本部分要解决古代文本时间判定的问题，首先是获取待判定的一部书籍或者书籍中的一段文本，其字符序列表示为S，将此文本送入模型M，模型进行计算并输出一个年代标签T。



(3-1)



(3-2)



**图3-1 古代文本断代模型结构框图**

这个过程中有以下几点关键技术需要注意，文本序列的向量化表示[68] 、模型总体结构和以及长短期记忆神经网络记忆单元结构。古代文本断代模型的结构框图如图3-1所示。

古籍断代模型接收一段古代文本作为输入送入模型，文本首先被送入嵌入层使模型获得该段文本的向量化表示。然后字符的嵌入向量被逐字送入双层LSTM神经网络层中分别计算正向和反向的隐藏向量。最终将两个隐藏向量串联送入输出层计算出最终的预测结果。模型的细节如图3-2所示：

**图3-2 古籍断代模型结构图**

模型主要分为三层，嵌入层、LSTM神经网络层和输出层。其中嵌入层使用的是word2vec中的CBOW模型，即通过文本上下文预测中间字的方式实现中间字的向量表示；神经网络层使用双层反向的LSTM作为主体，第一层接受文本的正向输入，输出一个隐藏层向量，第二层接受文本的逆向输入，输出一个隐藏层向量，然后将两个向量串联，由于LSTM具有记忆上文信息的能力，因此两个隐藏向量相当于将全文信息编码成一个向量表示；输出层为一个全连接网络，用于将上层得到的向量进行解码、计算，输出相应的预测年代。下面本文将对模型的三层结构进行详细的介绍。

#### 嵌入层

在式1中，文本字符序列仍是人类所能阅读的文字形式，等为一系列汉字的有序组合，式2中映射的作用是将人类所能理解的文字序列形式的转化成计算机所能理解的向量化表示：



(3-3)

其中分别为对应的每个字的向量化表示，这个将文字映射为向量表示的过程就叫做字嵌入。

目前字嵌入方式有两种，一种是对所有字符进行one-hot编码，但one-hot编码有编码过长，不同字之间向量相互垂直，没有语义联系，不能表示位置信息等缺点，因此我们不使用one-hot编码；另一种是稠密的低维向量表示(Dristributed representation)，它的思路是通过训练，将每个词都映射到一个较短的词向量上来，一方面可以解决One-hot 编码过长的问题，另一方面向量也可携带一定的语义信息。Google在2013年提出的word2vec[69] 是一种利用神经网络进行字嵌入训练的一个语言模型。他假设字向量是服从分布式假设的，如果两个词的上下文时相似的，那么他们语义也是相似的。word2vec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，如图所示。训练完毕后，输入层的每个单词与矩阵W相乘得到的向量的就是我们想要的词向量(word embedding)，这个矩阵也叫做查询表(look-up table)。



**图3-3 word2vec网络结构示意图**

Word2vec模型是非监督的，资料获取不需要很大的成本，我们可以通过word2vec模型在大量的未标注的语料上学习，就可以学习到比较好的向量表示，可以学习到词语之间的一些关系。比如男性和女性的关系距离，时态的关系，学到这种关系之后我们就可以把它作为特征用于后续的任务，从而提高模型的泛化能力。

同文字的向量化道理相同，时间标签同样需要考虑人类标签和机器理解的问题，本文使用的时间标签为年代标签，如汉代、唐代中期，首先将此各种时间标签按时间顺序排序，由于年代数相对较少，因此可以将朝代标签进一步表示为one-hot编码，方便模型通过Softmax评估出某一朝代。

#### 长短期记忆神经网络层

LSTM神经网络通过输入门、输出门和遗忘门这些门结构来控制长期记忆的遗忘与保存，LSTM记忆单元在时间维度上展开的结构图如图所示。



**图3-4 LSTM记忆单元在时间维度展开结构图**

LSTM记忆神经元中控制信息的门结构公式如下：



(3-4)



(3-5)



(3-6)



(3-7)



(3-8)

公式中，为输入门，是sigmoid函数，sigmoid函数的值域为[0，1]，起到将输入门向量中的每个元素限制在0到1之间。因此，输入门向量和另一个向量的哈达玛乘积就是对向量的每个维度进行一定的缩放，即可理解为保存或者遗忘向量中的某些信息。是tanh函数，用来将向量的每个元素映射到[-1，1]之间。是长期信息，用来保留距离当前时刻较远处的有用信息，是遗忘门，二者相乘可以用来控制长期信息中的某些信息是否继续保存到下一时刻，结合当前时刻的输入可以得到新的，此外还有输出门，用来控制输出信息，从而影响到当前时刻的输出以及下一时刻的输入。正是这些门结构使得LSTM可以有效的保留前文有用信息、遗忘前文中的无用信息，从而能做出更好预测。

#### 输出层

输出层是一个全连接网络，结构如图所示：



**图3-5 输出层网络结构图**

他接受一个上层传入的维度的向量，其中为LSTM神经网络输出的隐藏层向量维度。输出层包括一个维度的权重矩阵以及一个维度的偏置矩阵，其中为所有分类的类别数，输出层的输出为，是一个维的向量，最后通过Softmax函数对其进行输出归一化，归一化后的某一维度的数值即可视作该维度所表示年代的预测概率。Softmax公式如下。



(3-9)

### 实验

模型有两个评估指标：一个是单句分类正确率Ps；另一个是书籍分类正确率Pb：

Ps=N时间判定正确的句子条数/总句子条数

(3-10)

Pb=时间判定正确的书籍数/总书籍数

(3-11)

其中一本书籍时间判定的决策取决于书籍内部所有句子决策的投票结果，统计所有句子判定结果后，投票数最多的时间分类即判定为该书的时间段。在实现方面，模型通过Python的TensorFlow-GPU框架实现，所用的硬件配置为8G-cpu 1080Ti显卡。

训练流程图如下图所示：

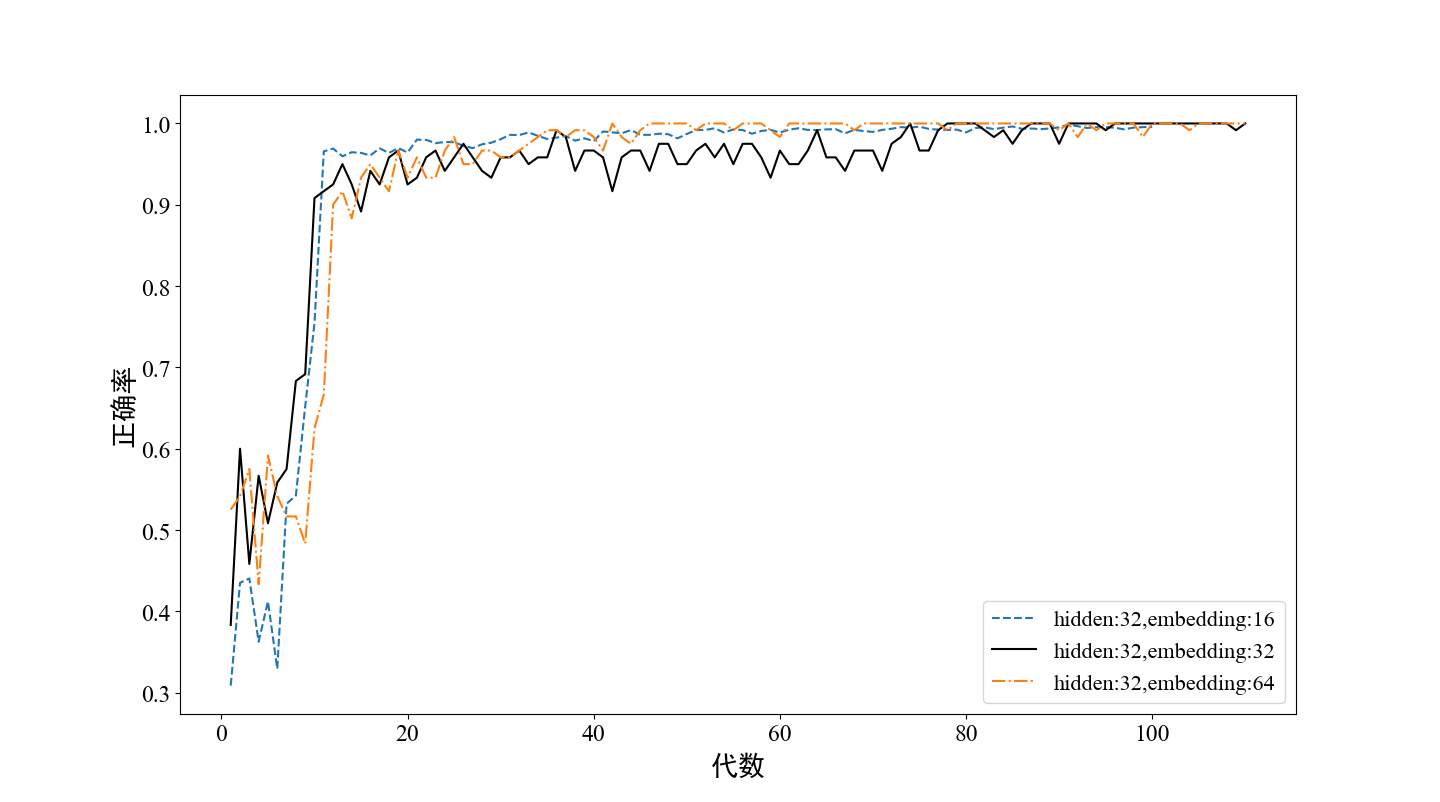


**图3-6 断代模型训练流程图**

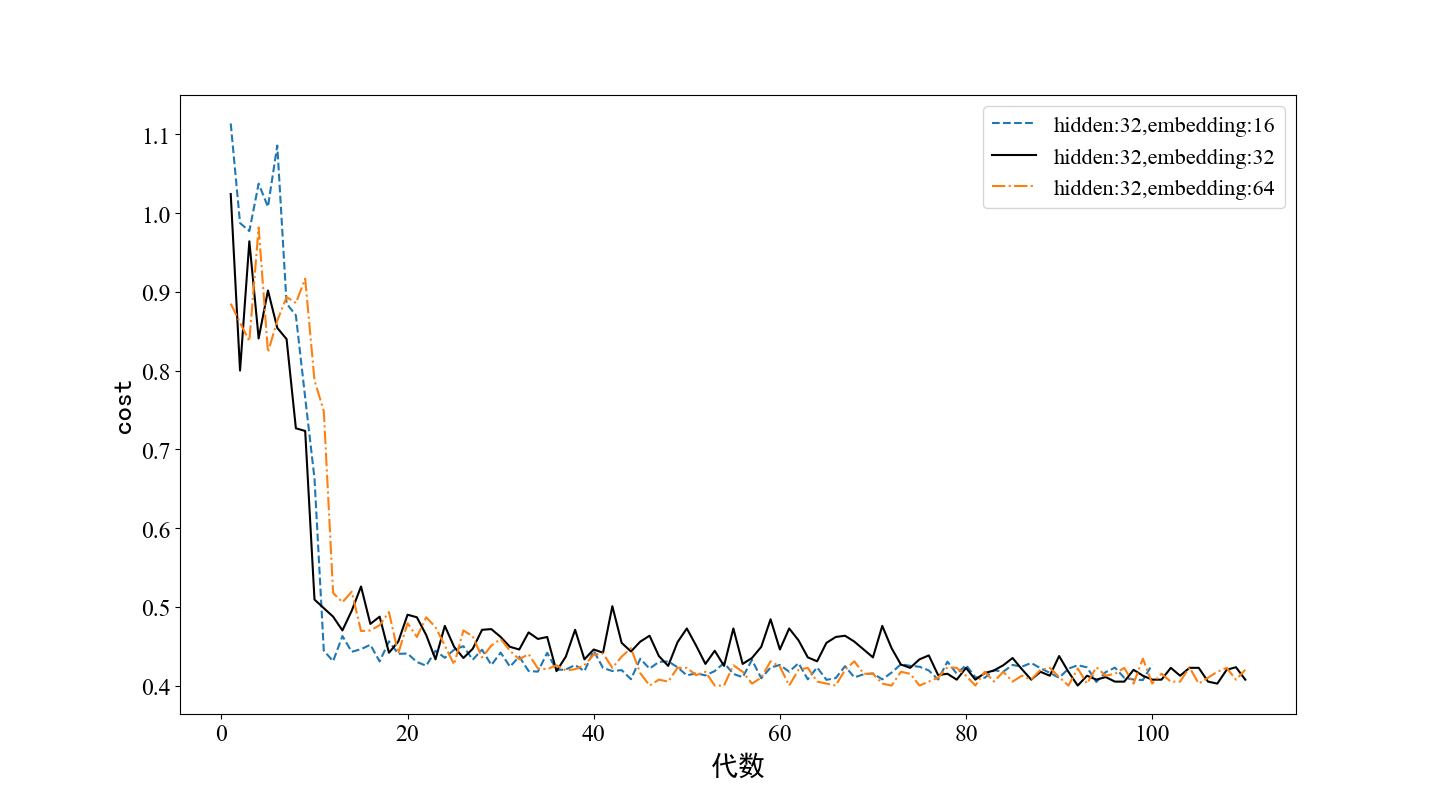
#### 参数选择

模型涉及到一些超参数的选择，我们分别在训练集和测试集上进行了实验，来比较不同超参数情况下模型的训练过程。本文中我们选择对不同维度的隐藏层(Hidden Layer Dimension, HLD）和单词嵌入向量(Embedding Layer Dimension, ELD)进行比较。

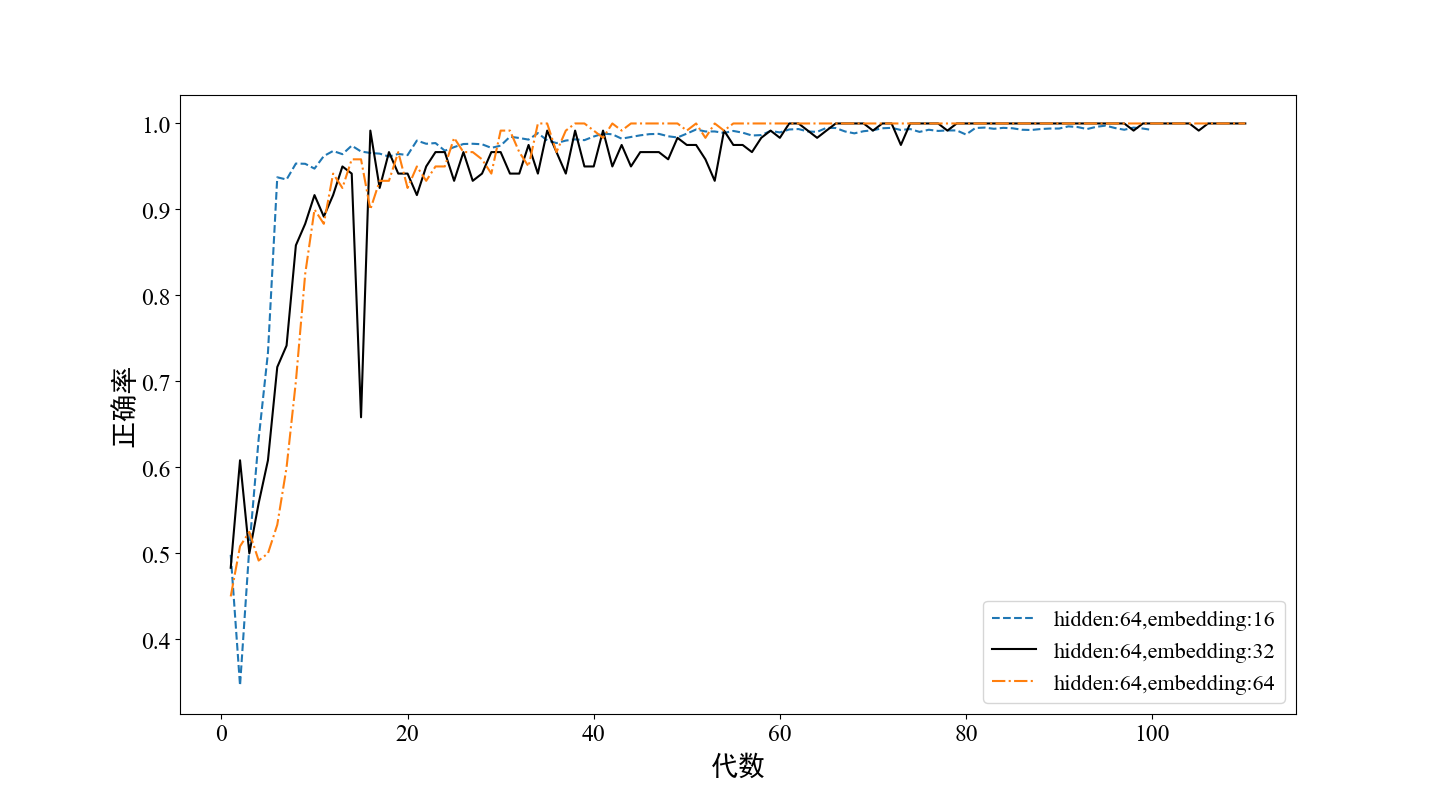
下面四副曲线图为在不同超参情况下训练过程的训练精度曲线。可见，当ELD为64（图中黄色点划线）时，模型在两组实验中上均能在训练50代左右处开始稳定在较高水平。此外，每组图中我们保持ELD不变，利用两组HLD进行比较，比较发现在每个实验图中，ELD=16的曲线(蓝色虚线)都比较平滑，有稳定上升的趋势。因此从收敛速度的角度来看，我们应选择HLD为64的参数进行实验，从模型的稳定程度来说，应选择ELD=16的模型参数。但是HLD=64时，模型的复杂度较高，虽然模型可以收敛到准确率很高的水平，但是却有过拟合的风险，下面我们将在训练集上进行实验来进一步进行参数选择。



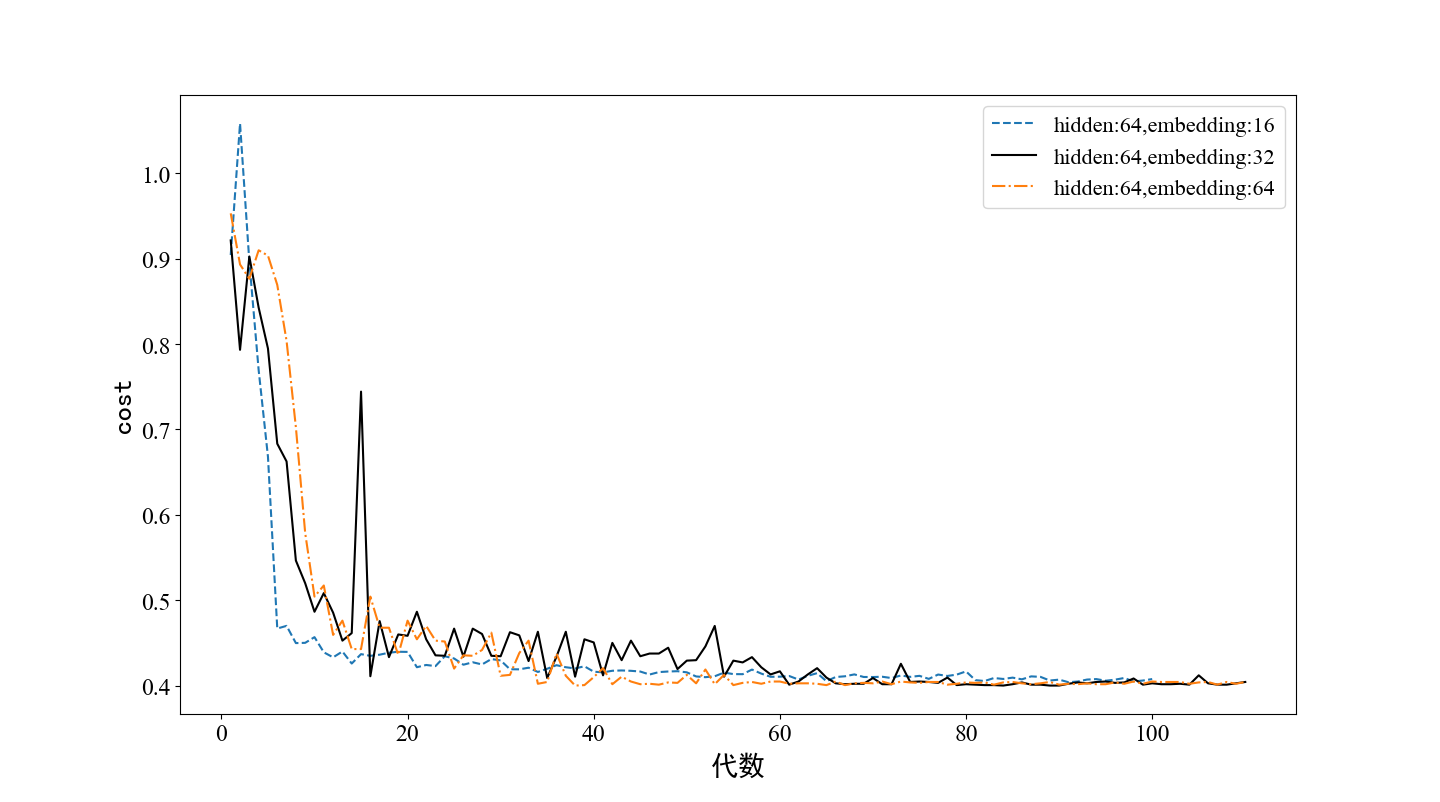
**图3-7 HLD=32不同ELD情况下训练精度曲线**



**图3-8 HLD=32不同ELD情况下训练代价曲线**



**图 3-9 HLD=64不同ELD情况下训练精度曲线**

****

**图 3-10 HLD=64不同ELD情况下训练代价曲线**

下面我们比较不同参数时模型在测试集上的表现，实验结果如表3-3所示。表中可见，虽然HLD为64，ELD为64的参数在训练集中收敛速度表现较好，但是在测试集中其表现却不够优秀，而HLD为64，ELD为16时在训练集上虽然在收敛速度上略逊于HLD=64，ELD=64的参数设置的模型，但其在测试集上的正确率高于HLD=64，ELD=64的模型。这种情况证明了模型在HLD=64，ELD=64的时候训练过程中产生了过拟合的现象，模型对训练集拟合程度过高，而对于其他数据集其表现将有所欠缺。因此，我们后面的实验将统一使用在训练集上表现更好的HDL=64、ELD=16的模型参数，一方面兼顾收敛速度，一方面ELD=16可以简化网络结构，防止过拟合。

**表3-3不同超参情况下模型在部分测试集上的表现情况**

|  |  |  |
| --- | --- | --- |
| 隐藏层向量维度(HLD) | 字嵌入向量维度(ELD) | 断代  正确率 |
| 32 | 16 | 0.921 |
| 32 | 32 | 0.937 |
| 32 | 64 | 0.910 |
| 64 | 16 | 0.935 |
| 64 | 32 | 0.912 |
| 64 | 64 | 0.925 |

#### 断代实验

在第一个实验中，各个时期的每本书中都选出70%的章节作为训练集，保留每本书的其余30%章节不参与训练作为最后的测试集，以此来保证某个时期内的所有书籍均有部分语句用于训练。如图所示：



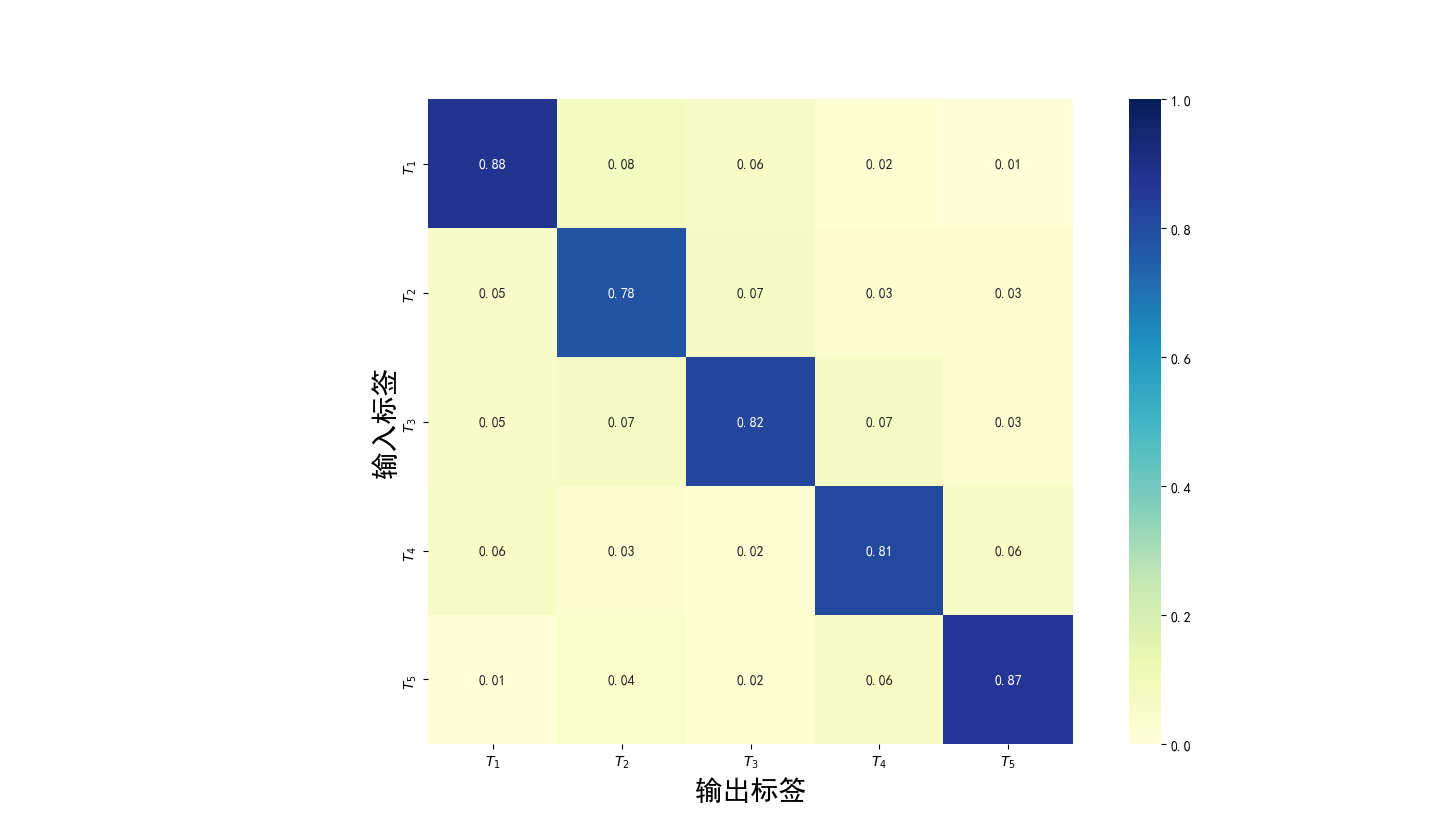
**图3-11 实验一中训练集和测试集的获取方法示意图**

然后我们将测试集中的文本送入模型，来预测他们的年代。下表为实验结果。表中的每一行表示输入为某时代的古籍文本时，模型的预测为不同时代的结果的句子条数。同时我们还将结果以热力图的形式呈现，如图3-12所示。从图中非常清楚的可以看到热力图的对角线有一条颜色很深的连线，这说明模型将输入样本做出正确判断的概率非常大，模型的输出和正确结果之间有着很强的相关性。这说明当所有书籍均有部分语句用于训练时，判断一个句子为正确时期的概率很大。证明了某一时代内同一本书中的句法语法结构基本一致，模型学习了部分章节的结构信息后，可以较好的适用于同本书的其他章节中。

**表3-4 实验一结果**

输出

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入 | T1 | T2 | T3 | T4 | T5 |
| T1 | 15672  (88.3%) | 536  (7.5%) | 524  (5.7%) | 359  (2.5%) | 186  (0.9%) |
| T2 | 821  (4.6%) | 5563  (78.1%) | 675  (7.3%) | 416  (2.9%) | 669  (3.2%) |
| T3 | 897  (5.0%) | 485  (6.8%) | 7531  (82.2%) | 1069  (7.4%) | 561  (2.7%) |
| T4 | 111  (0.6%) | 239  (3.3%) | 231  (2.5%) | 11569  (80.6%) | 1239  (6.0%) |
| T5 | 239  (1.3%) | 298  (4.1%) | 196  (2.1%) | 924  (6.4%) | 17880  (87.1%) |
| 共计 | 17740  (100%) | 7121  (100%) | 9157  (100%) | 14337  (100%) | 20535  (100%) |



**图 3-12 断代模型输入输出关系热力图**

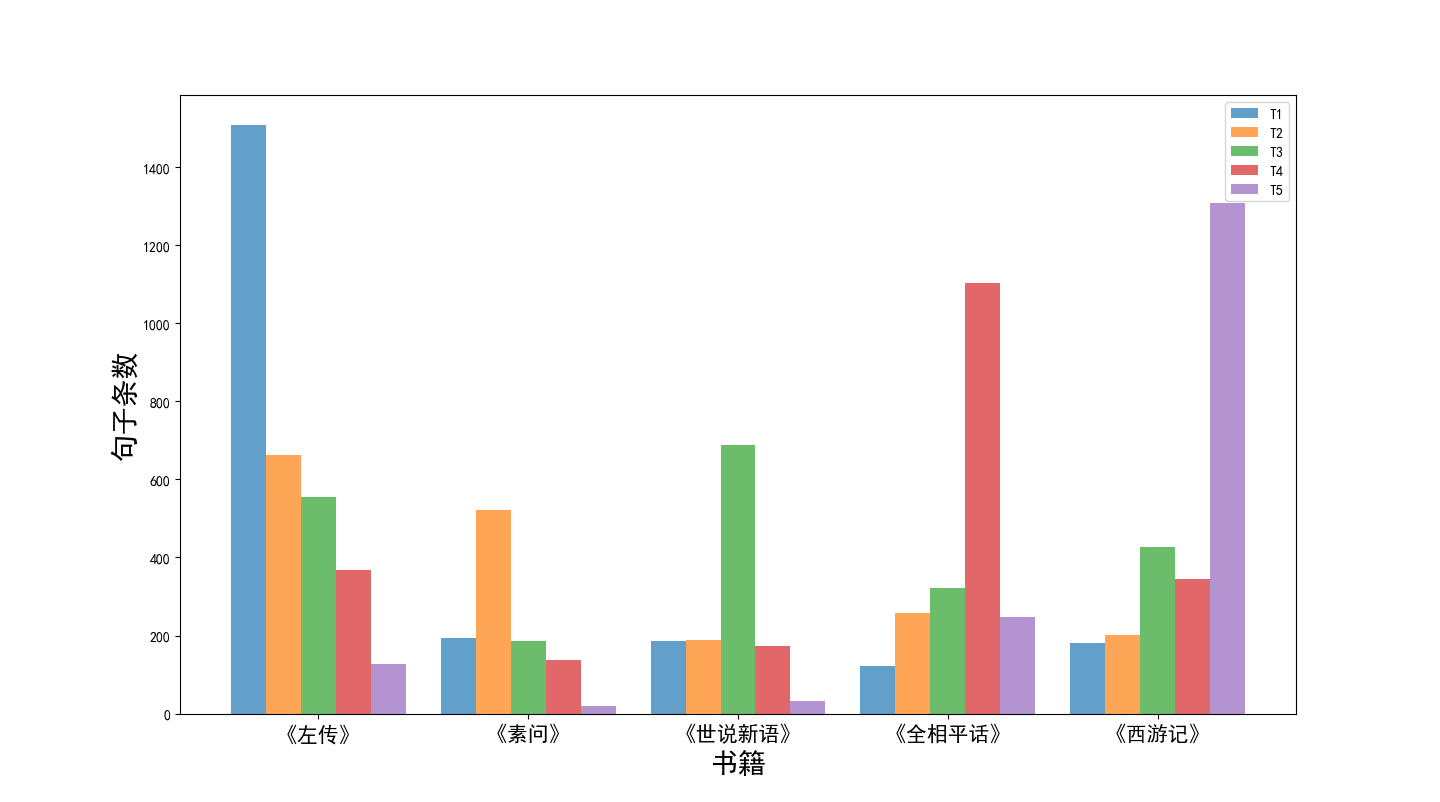
我们也做了另外一个实验，在该实验中我们在每个时期中都挑选约30%的整本书作为测试集不参与训练，用该时期剩下70%的书籍文本训练模型，如图3-13，去观察模型对训练集之外的书籍的断代效果。



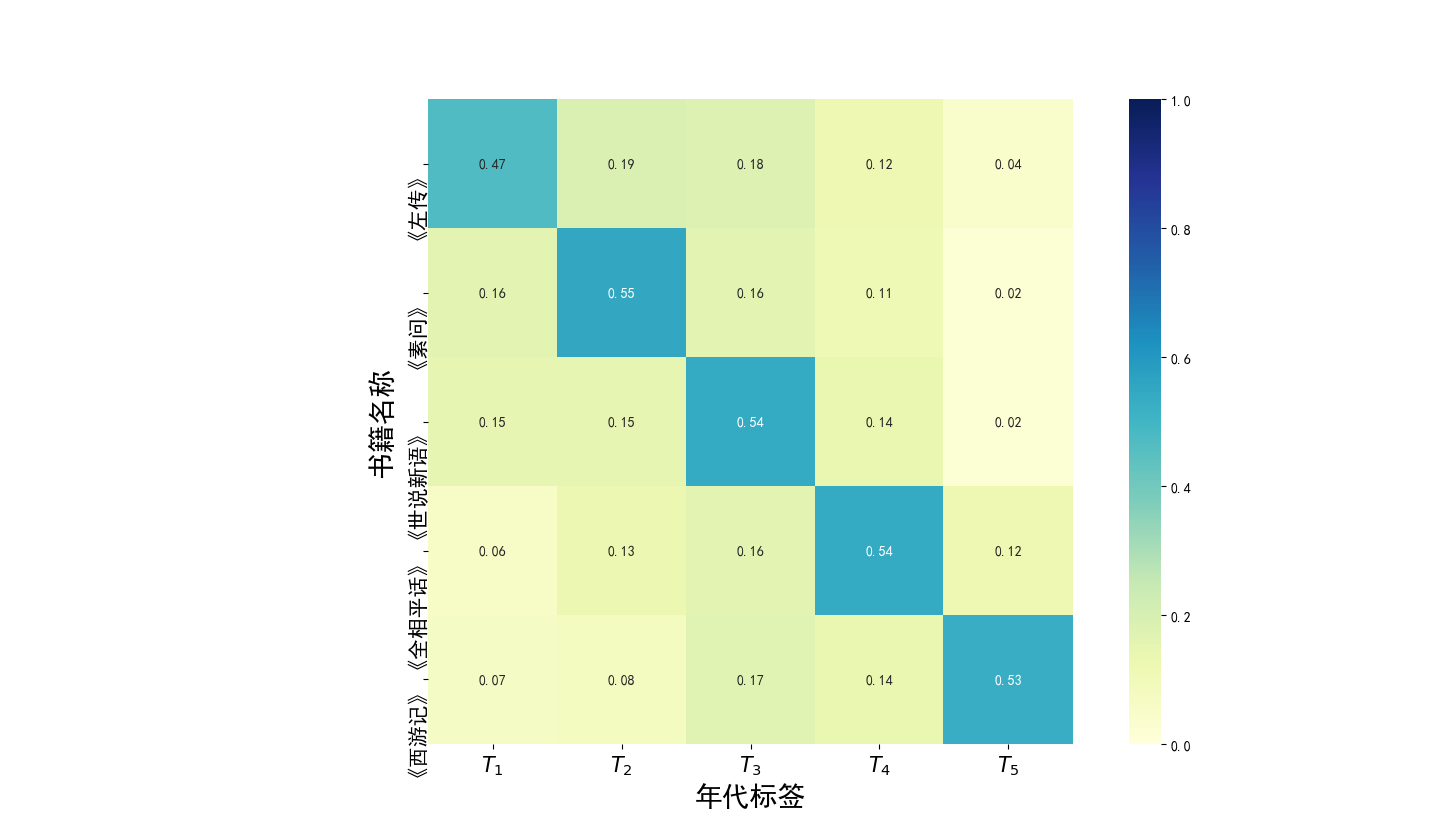
**图3-13 实验二中训练集和测试集的获取方法示意图**

我们在春秋战国、后汉、南北朝、宋、明清时期分别选择《左传》、《素问》、《世说新语》、《全相平话》和《西游记》共5本书为测试集进行了实验。在训练集没有输入该五本书任何文本的情况下，我们输入这五本书的文本，让模型预测其年代。模型的预测结果如图3-14。以《左传》为例，在所有《左传》的句子中，模型将其中1508句预测为春秋战国时期，将663句判断为后汉时期，556句判断为南北朝时期，369句判断为宋朝时期，126句判断为明清时期，其他年代书籍也有近似的分布规律。在训练集中不包含测试集书籍的情况下，模型对该书的单句断代正确率(47%左右)相比实验一降低了不少，可见一本书籍在词法、句法上有一定的独立性，在模型学习了同一时期的其他书籍之后，不能十分精确的将该时期另外书籍的文本正确分类。而从整体上看，我们利用一本书中的所有句子断代结果为该本书进行投票，投票占比最多的时代即判断为该书的时代。在这种情况下，图中可见，《左传》预测结果最多的标签为T1标签，即模型最终判断《左传》是春秋战国时期的书籍，此结果与中国古籍研究领域目前的共识是一致的，也印证了我们模式的正确性。该实验证明了同一时期内的某一本书虽然同其他书籍在词法句法上有一定的独立性，但是总体而言同一时期内的书籍之间也是有潜在联系的，也有一定的统一性，因此模型可通过学习同一时期内一定量的书籍来判断其他书籍的信息。

此外，我们展示了模型对这5本书预测结果的热力图，如图3-15，图中横坐标为年代标签，纵坐标为某书籍，图中交叉块的颜色深度为模型预测某本书为某年代的概率是多少，颜色越深，概率越大。我们发现图像的整体分布为左上角到右下角的对角线颜色最深，同时向两侧逐渐减淡。我们观察模型的错误输出，错误最多的是将书籍判断为书籍所在时代的临近时代，并随着距离正确时代越远，判断错误的情况越少。这是主要因为中国古汉语演变至今不是突变导致的，而是实验展示的那样逐步一点一点演变的，古汉语在时代和时代之间是有相似性的，这种相似性随着时代之间相隔年份的增加而减小，而这种相似性会使我们的模型产生判断错误。这也从侧面说明了我们的模型学习到了古代文本中的潜在的语法、语义信息。



**图3-14 模型对各年代书籍的输出结果条形图**

****

**图3-15 模型对各年代书籍的输出结果热力图**

实验结果表明，如果选取某一时期所有古籍的部分文本作为训练集，其余文本作为测试集（实验一），该模型的实验正确率可以达到90%。但是，若测试集的文本所属的书籍未参与模型的训练（实验二），古籍句子的正确率会降低，但综合某本古籍的所有句子来看，仍然可以通过投票原则正确判断古籍的年代。以上两个实验可以看出，同一部古籍的词汇和语法规则相对统一，同一历史时期不同古籍之间既互相独立，又一定程度上统一、互相联系。我们的模型成功学习到了不同时代古籍之间的内在联系和区别。

### 本章小结

我们在本部分提出了一个针对古籍断代的有效并可以投入实际使用模型，本模型可用于辅助古籍工作人员的进行断代工作。本部分使用了Bi-LSTM网络实现了古籍断代的任务。本部分通过实验展示了不同情况下模型的预测结果对断代任务有一定的参考价值，证明了Bi-LSTM在文献量较少的古汉语领域也能训练出正确率不错的断代模型。

古代汉语断句模型

古代书籍浩如烟海，因此提出一种自动断句的模型十分必要。本部分利用机器学习方法，设计一种针对古汉语文本的断句模型，通过实验对参数进行调优，获得最佳模型来辅助古汉语工作者完成大量的古籍的句读工作。

### 数据来源及预处理

本部分可使用的数据来源较为开放灵活，为针对后面部分所使用的上古语料，本部分也从互联网的开放数据集上下载带有标点的上古汉语文本。文本包括《尚書》、《詩經》、《周易》、《儀禮》、《周禮》、《禮記》、《春秋公羊傳》、《左傳》、《國語》、《論語》、《孟子》、《莊子》、《呂氏春秋》、《老子》、《孝經》、《史記》、《春秋繁露》等在内的30余本著作，共计300万字。

本部分的重点是对无标点的古代汉语文本进行断句。因此在所有的数据集的基础上，我们对文本进行以下预处理：编写Python脚本将文本中的所有标点均使用“/”符号代替，对文字打上断句标签。预处理过程如示意图4-1所示：



**图 4-1 文本预处理过程示意图**

表4-1是对文本进行预处理过程的一个实例展示：

**表4-1利用原文标点对文本进行标签化预处理实例**

|  |  |
| --- | --- |
| 原句 | 子曰：“参乎！吾道一以贯之。”曾子曰：“唯。”子出，门人问曰：“何谓也？”曾子曰：“夫子之道，忠恕而已矣。” |
| 标签化后 | 子曰/参乎/吾道一以贯之/曾子曰/唯/子出/门人问曰/何谓也/曾子曰/夫子之道/忠恕而已矣/ |

### 模型构建

本部分使用的主要结构是Bi-LSTM神经网络，不同于古代文本断代模型接受完整句输入后才输出年代预测结果，断句模型对每个字都有一个输出断句标签概率，我们首先定义一个阈值(D)，当模型输出概率大于该阈值时，则模型输出“s”标签，表示在此字之后需要断句；若小于阈值，则输出“n”标签，表示此处无句读。如图4-2所示：



**图4-2 模型处理断句任务流程图**

模型对每个字的输出和正确答案比较求出当前输出的代价C，并利用反向传播算法调整模型当中的各种参数。该模型的核心是嵌入层和Bi-LSTM神经网络，其网络结构如图4-3所示：



**图 4-3 断代模型嵌入层和神经网络层结构示意图**

### 实验及效果展示

#### 实验配置

模型是使用Python TensorFlow-GPU搭建，所使用的机器的配置为：

**表4-2 机器的配置表**

|  |  |
| --- | --- |
| 名称 | 配置 |
| 处理器 | Intel Xeon(R) CPU ES-2620 v4@2.10GHz×16 |
| 内存 | 62.8GiB |
| 显卡 | GeForce GTX 1080 Ti/PCIe/SSE2×2 |
| 显卡内存 | 11264MB×2 |
| 系统 | Ubuntu 16.04 64-bit |

#### 网络搭建伪代码

模型的搭建全部使用Python中TensorFlow框架实现，下面为使用TensorFlow框架搭建Bi-LSTM神经网络的伪代码。

|  |
| --- |
| Bi-LSTM网络搭建 |
| 输入： X\_inputs,Y\_inputs,layer\_num,batch\_size,timestep\_size,hidden\_size,class\_num |
| 1：cell←调用TensorFlow的rnn类中的LSTMCell方法初始化一个LSTM cell |
| 2：inputs←对X\_inputs调用查询表查询，查询各字的对应嵌入向量，得到嵌入向量列表 |
| 3：while layer<layer\_num do |
| 4： cell\_fw，cell\_bw←调用TensorFlow包中rnn类的MultiRNNCell方法,初始化正向LSTM层和反向LSTM层网络对象 |
| 5：end while. |
| 6：initial\_state\_fw,initial\_stat\_bw←根据batch\_size初始化正向和反向LSTM网络的初始参数 |
| 7：output\_fw←[] |
| 8：state\_fw←initial\_state\_fw |
| 9：for timestep←1,timestep\_size do  10： output\_fw,state\_fw←将inputs输入到cell\_fw中，返回前向网络输出和当前参数状态 |
| 11：end for |
| 12：inputs←reverse(inputs) 将原input前后取反作为后向网络的输入 |
| 13：for timestep←1,timestep\_size do |
| 14： output\_bw,state\_bw←将inputs输入到cell\_bw中，返回后向网络输出和当前参数状态 |
| 15：end for |
| 16：Hidden\_vector←concatenate(ouput\_fw,output\_bw)将前向输出向量和后向输出向量串联 |
| 17：全连接神经网络: |
| 18：bilstm\_output←Hidden\_vector |
| 19：W←根据hidden\_size和class\_num初始化全连接层的权重矩阵 |
| 20：b←根据class\_num初始化全连接层的偏置矩阵 |
| 21：y\_pred←W×bilstm\_output+b将Bi-LSTM层的输出送入全连接层，返回y\_pred |
| 22：cost←调用TensorFlow的reduce\_mean方法利用corss entropy计算模型输出的误差代价 |
| 23：optimizer←初始化一个Adam优化器 |
| 24：调用Adam优化器优化所有网络参数 |

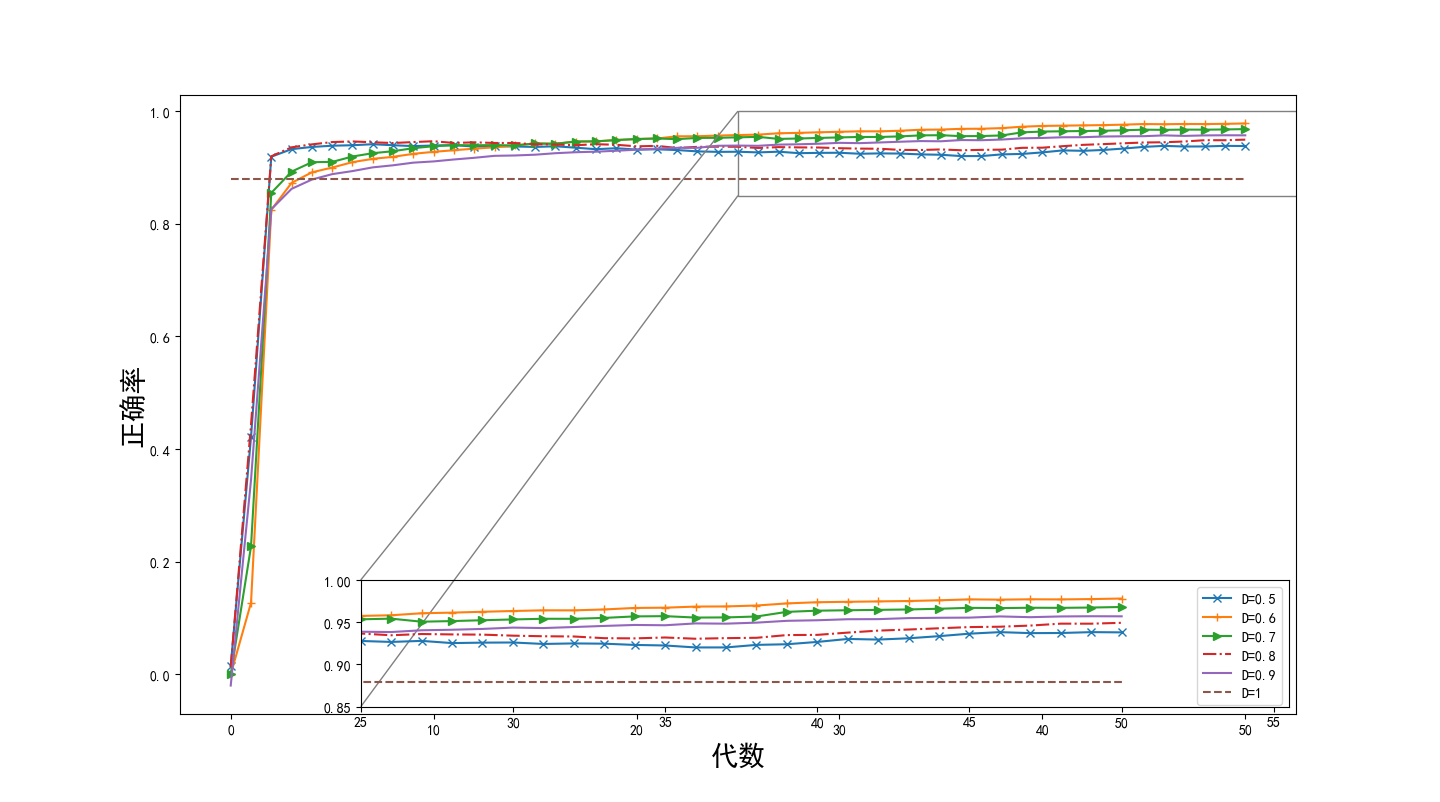
#### 实验及分析

我们使用上古古籍的文本对模型进行训练，模型的输入是最大长度为100的无标点文本，模型输出每个字的断句标记。模型的正确率R定义如下，其中表示模型预测正确的字数，表示输入文本的总字数。

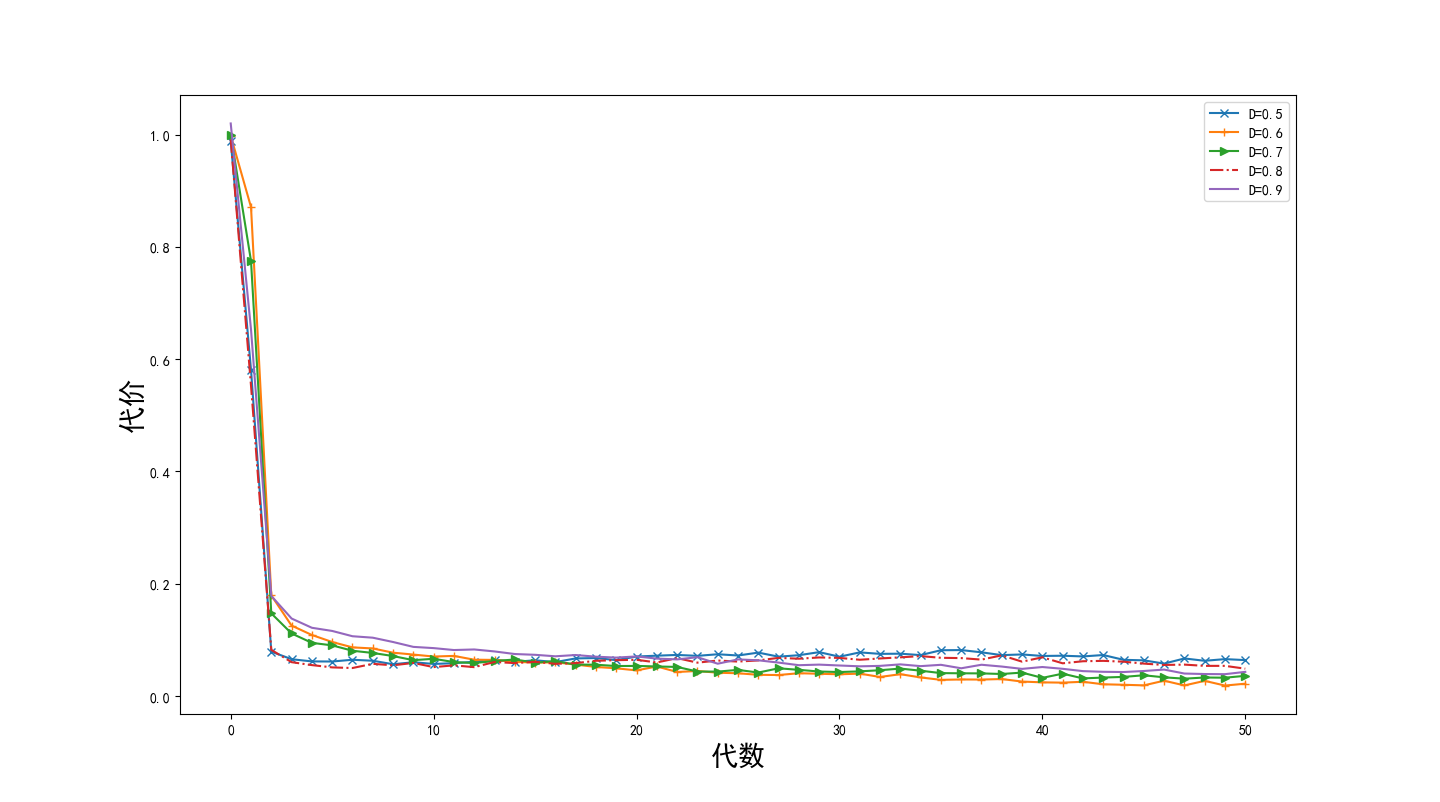


(4-1)

模型训练过程中有一个非常重要的步骤，就是“阈值(D)”的选择，我们分别对不同阈值做了多组实验，训练过程的正确率及代价曲线图如下图所示：

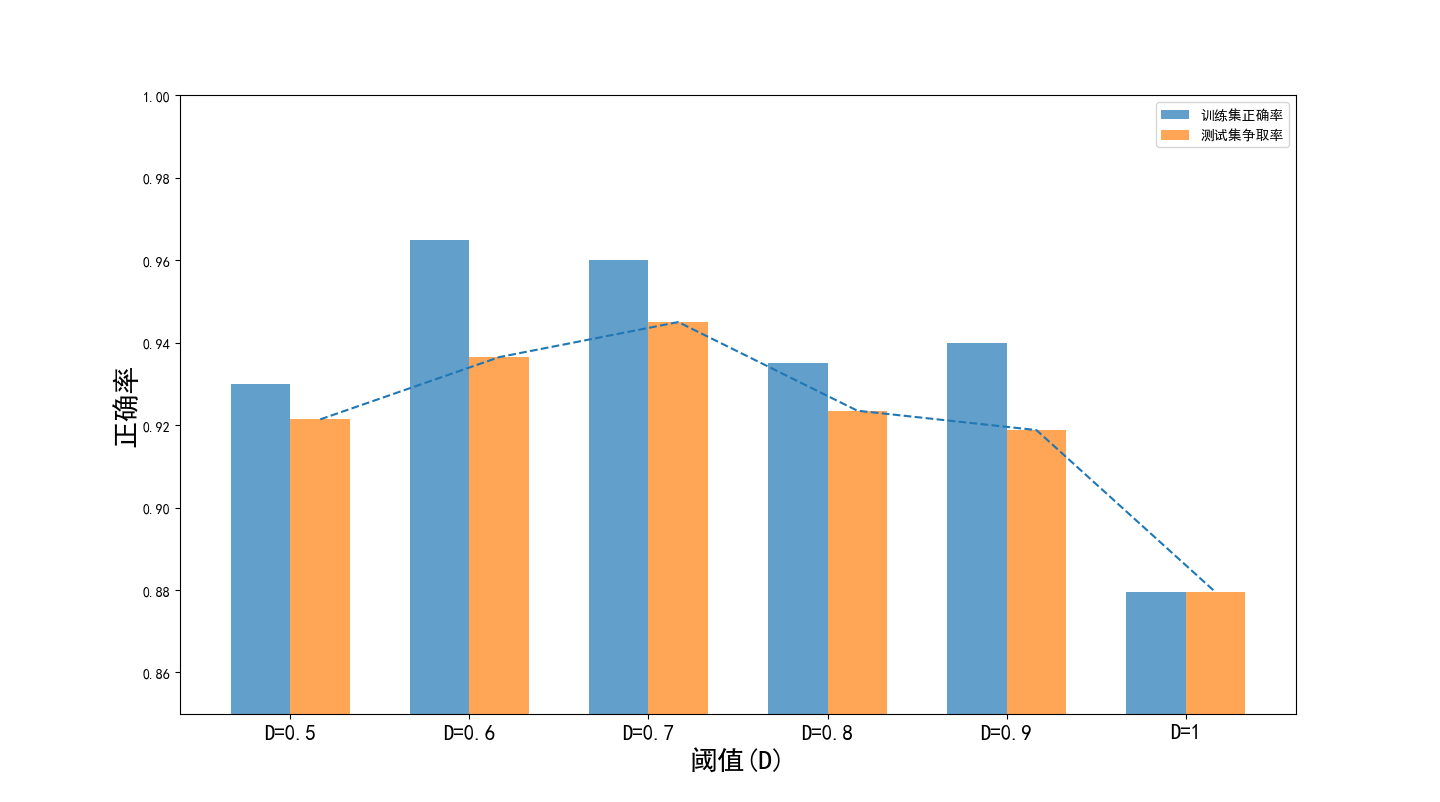


**图4-4 训练过程的断句正确率曲线图**

****

**图4-5 训练过程的断句代价曲线图**

图4-4展示了阈值分别等于0.5/0.6/0.7/0.8/0.9以及1的情况下模型训练过程正确率的变化曲线，其中当阈值等于1时，模型所有输出标签概率均达不到阈值，则所有标签均标记为“n”即不断句，在这个情况下单字标签的正确率是0.87左右，反映了整个训练集中，“n”标签占所有标签的87%，则0.87这条线可以作为衡量模型是否有效的基准线。

****

**图4-6 不同阈值模型在训练集和测试集的表现**

图中可见在训练集上，模型的标注正确率可以达到92%以上，高于87%的基准线，说明模型在不同阈值下均能训练得到一套比较有效的模型参数。同时我们发现，当阈值从0.5变化到1的过程中，模型在测试集上的争取率由低变高，到D=0.7处达到最高，之后又随着阈值的增加而减小。因此通过这个实验我们可以得到训练过程的最佳阈值，在我们的训练条件下，最佳阈值为0.7。

利用D=0.7的阈值对模型进行训练完成之后，我们在部分测试集样本进行了实验并分析。下面我们将部分实验结果进行展示如下，其中(-)表示此处应有断句，而模型模型输出有所遗漏。(+)此处不应有断句，而模型输出误将此处进行了断句。

实例一：

正确断句：

從者曰 長子近 且城厚完 襄子曰 民罷力以完之 又斃死以守之 其誰與我 從者曰 邯鄲之倉庫實 襄子曰 浚民之膏澤以實之 又因而殺之 其誰與我

模型输出：

從者曰 長子近(-)且城厚完 襄子曰 民罷力以完之 又斃死以守之 其誰與我(-)從者 (+)曰 邯鄲之倉庫實(-)襄子曰 浚民之膏澤以實之 又因而殺之 其誰與我

误分(+)：1处 遗漏(-)：3处

实例二：

正确断句：

故君子操權一政以立術 立官貴爵以稱之 論勞舉功以任之 則是上下之稱平 上下之稱平 則臣得盡其力 而主得專其柄 天地設 而民生之 當此之時也 民知其母而不知其父 其道親親而愛私

模型输出：

故君子操權一政以立術 立官貴爵 (+)以稱之 論勞舉功以任之 則是上下之稱平(-)上下之稱平 則臣得盡其力(-)而主得專其柄 天地設(-)而民生之 當此之時也 民知其母而不知其父 其道親親而愛私

误分(+)：1处 遗漏(-)：3处

实例三：

正确断句：

駔琮五寸 宗后以為權 大琮十有二寸 射四寸 厚寸 是謂內鎮 宗后守之 駔琮七寸 鼻寸有半寸 天子以為權 兩圭五寸 有邸 以祀地 以旅四望

模型输出：

駔琮五寸 宗后以為權 大琮十有二寸 射四寸 厚寸 是謂內鎮(-)宗后守之 駔琮七寸 鼻寸有半寸 天子以為權 兩圭五寸 有邸(-)以祀地 以旅四望

误分(+)：0处 遗漏(-)：2处

实例四：

正确断句：

君仁莫不仁 君義莫不義 君正莫不正 一正君而國定矣 孟子曰 有不虞之譽 有求全之毀 孟子曰 人之易其言也 無責耳矣

模型输出：

君仁莫不仁 君義莫不義 君正莫不正 一正君而國定矣 孟子曰 有不虞之譽(-)有求全之毀(-)孟子曰 人之易其言也 無責耳矣

误分(+)：0处 遗漏(-)：2处

实例五：

正确断句：

陽也者稹理而堅 陰也者疏理而柔 是故以火養其陰而齊諸其陽 則轂雖敝不藃 轂小而長則柞 大而短則摯 是故六分其輪崇 以其一為之牙圍 參分其牙圍而漆其二

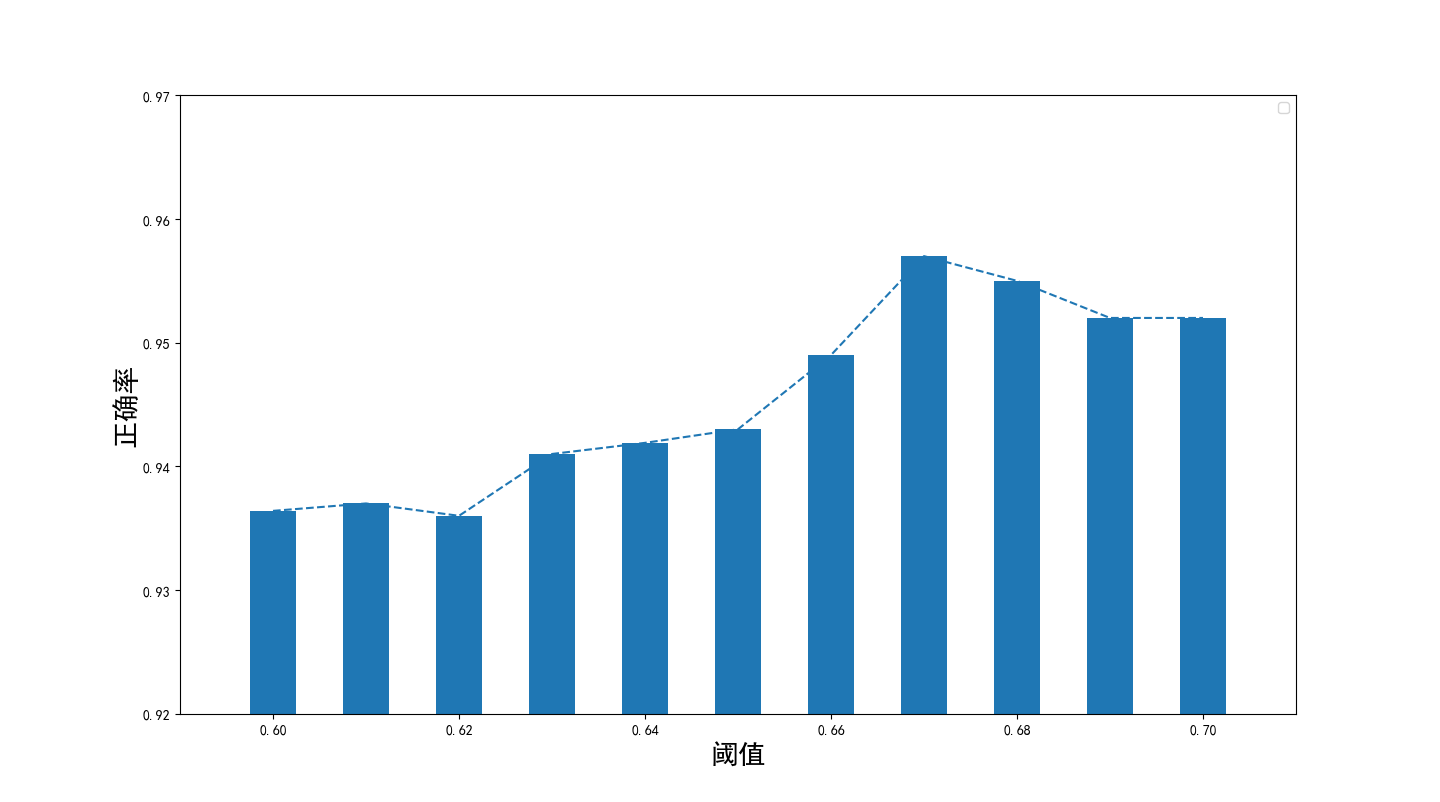
模型输出：

陽也者稹理而堅(-)陰也者 (+)疏理而柔 是故以火養其陰而齊諸其陽 則轂雖敝不藃 轂小而長則柞 大而短則摯 是故六分其輪崇 以其一為之牙圍 參分其牙圍而漆其二

误分(+)：1处 遗漏(-)：1处

共计：303字 标记正确：289字 误分(+)：3处 遗漏(-)：11处

由上面的实例表明，模型做出的断句判断整体正确，整体正确率达95.4%，也有遗漏和误分等情况的发生，错误主要发生的情况是遗漏问题，即某处本应有句读，而模型未在此处断句。模型参数阈值D=0.7在当前测试集中略偏大，导致部分断句标签概率输出较小的地方被误分为不断句。由于我们的模型不是静止不变的，我们是在不断添加数据库的过程中不断的调整模型参数，这时我们可以在上面实验基础上，继续微调阈值，类似于在现有模型的基础上进行fine-tune。



**图4-7 微调阈值后模型在新语料库中的表现**

上面实验表明，在测试集上，也可视作在新增加的语料库里，模型的最佳参数(D=0.67)略小于原始模型的阈值(0.70)，我们对模型参数略作修改，模型即可适应新增加的语料。同理，在后续继续扩充语料库的过程中，我们可以循环进行训练模型、更新数据库、微调模型参数这三项步骤来同时迭代更新模型和语料库。

### 本章小结

本部分通过使用Bi-LSTM对文本进行逐字标签化的方式，实现了一个古代汉语断句模型。并进行了一定量的实验选择最佳阈值，实验表明，模型在阈值设置为0.7的情况下正确率最高，标记正确率可达95%左右，可以分担古汉语工作者部分部分断句的任务。

古代汉语分词、标注系统及数据库建设

计算机在进行中文自然语言处理时一般是以词为最小单位的。中文不同于英文词与词之间有空格分割，中文是以‘字’为基本书写单位，词语之间没有一个形式上分界符。所以中文分词是汉语自然文本处理的基础问题之一。本部分将通过机器学习搭建自动分词及词性标注一体化系统，并做实验分析模型性能。模型输出的带有分词及词性标记的文本则可存储到数据库实现古带汉语的信息化和结构化管理。

### 数据来源及预处理

由于当前尚无开放的已完成分词标注的古汉语语料库，因此，本部分使用的数据来是源于网上下载的少量开放数据集，并在后期在实验室的同学们帮助下，参考中央研究院的分词标准对部分语料进行手工标记所得。数据内容包括《尚书》、《礼记》、《论语》等多本上古古籍的部分章节。调查数据包含超过10,000条语句，平均每条语句包含9个字。由于本部分所使用的语料库较小，无法对模型进行充分的训练，因此本部分的重点为提出一种可同时用于分词及词性标注的模型并证明该模型具有一定的实用性，后序继续可以使用该模型对预料进行大致的分词及词性标注，所产生的带有分词标签及词性标签的语料在经过人工确认后又可继续用来训练模型以提高模型的对上古汉语分词及词性标注任务的拟合度。

#### 标签编码

首先，我们对语料标签进行标签转换，将语料的分词标记和词性标记转成统一的二元标签结构，如(5-1)-(5-3)式所示，为词性标签列表，其中元素、等分别表示不同的词性标记。为分词标签列表，其中的、等表示不同的分词标记，携带不同的分词信息。而则表示二元标签组，其中的每一个标签是由的转置点乘得到的矩阵，则中的每一个标签都分别携带了词性标记信息和分词标记信息式。



(5-1)

****

(5-2)

****

(5-3)

****

(5-4)

预处理过程如图5-1所示，首先将词****拆分成单字****，并且单字****也携带该字隶属词的词性标记信息****，例如将“瀑布/N”转换为‘瀑/N’和‘布/N’，然后使用分词标签为每个字打上分词标记，即将中文分词当成分类任务。最后将词性标签作为第一维标签，分词信息标签作为第二维标签组合成二元标签组的结构。



**图5-1 文本预处理过程示意图**

本模型的输出不同于分词或词性标记单任务模型的单标签输出，本模型输出的是经过编码的二元标签组，标签的两个维度分别表示词性标记和分词信息，可以使网络更加充分考虑字、词性和分词之间的内在关联，相较先分词后进行词性标注这样的 “两步走”方法增加了词性和分词之间的内在关联，使一体化模型的精确度更高。

针对一体化模型中的输入、输出层，我们提出两种二元标签结构的编码方式，如图5-2和5-3所示，若二元标签组包括m种词性标记，用表示，n种分词标记，用表示，则编码方式一为：对种不同词性标记和分词标记的自由组合结果进行编号，即每一个不同的二元标签有自己固定的从0至的某一编号，然后对其进行One-hot编码，即一串位的0/1序列，其中除某一位为1以外，其他均为0，为1的位对应的编号即为对应的二元标签，如图5-2所示。编码方式二：对种二元标签进行二维编码，第一维有m列，表示词性标注信息，第二维有n列，表示分词信息，则二元标签组的编码为一串位的0/1序列，其中前m位中仅有一位为1，对应词性标签的One-hot编码，后n位有一位为1，对应分词标签的One-hot编码。如图5-3所示。这样二元标签组就表示成了一串计算机能处理的0/1序列。



**图5-2 编码方式一示意图**



**图5-3 编码方式二示意图**

### 分类模型的评估标准

准确率和召回率是广泛用于信息检索和统计学分类领域的两个度量值，用来评价结果的质量。其中精度是检索出相关文档数与检索出的文档总数的比率，衡量的是检索系统的查准率；召回率是指检索出的相关文档数和文档库中所有的相关文档数的比率，衡量的是检索系统的查全率。一般来说，Precision就是检索出来的条目（比如：文档、网页等）有多少是准确的，Recall就是所有准确的条目有多少被检索出来了。



(5-5)



(5-6)

其中为系统判断为正确的数量， 为样本中所有正样本的数量，为样本中所有负样本的数量。准确率和召回率是互相影响的，理想情况下肯定是做到两者都高，但是一般情况下准确率高、召回率就低，召回率低、准确率高，当然如果两者都低，那是什么地方出问题了。通常，我们希望准确率和召回率均越高越好，但事实上这两者在某些情况下是矛盾的。

比如我们只搜出了一个结果，此结果是正确的，求得precisin等于1。但是由于只搜出一个结果，recall值反而很低，接近于0。所以需要综合考量，因此便引入了F-measure。F-measure又称F-score，其公式为：

****

(5-5)

其中，当β=2时，更加注重召回率；β=0.5时，更加重视准确率；当β=1时，就是F1-score：

****

(5-6)

F-measure综合了precision和recall，其值越高，通常表示算法性能越好，下表展示了召回率、准确率和F-score之间的关系。



**图5-4 两类分类的混淆矩阵示意图**

### 模型架构

本部分将介绍基于Bi-LSTM的古汉语自动分词及词性标注一体化模型的相关细节，图5-5为模型的训练流程图。其中，数据预处理是指将原预料库中即词/词性标记形式的内容处理为模型所需要的即单字/二元标签组形式的内容。编码指将计算机不能理解的二元标签组转变成计算机可以处理的数字序列，以便送入神经网络进行计算。本节将包括语料库数据的预处理，网络结构和标签编码三个部分。



**图5-5 模型的训练流程图**

#### 网络结构

本模型利用Bi-LSTM网络进行训练。图5-6是基于Bi-LSTM的自动分词及词性标注一体化模型总体示意图。



**图5-6 基于Bi-LSTM的自动分词及词性标注模型结构示意图**

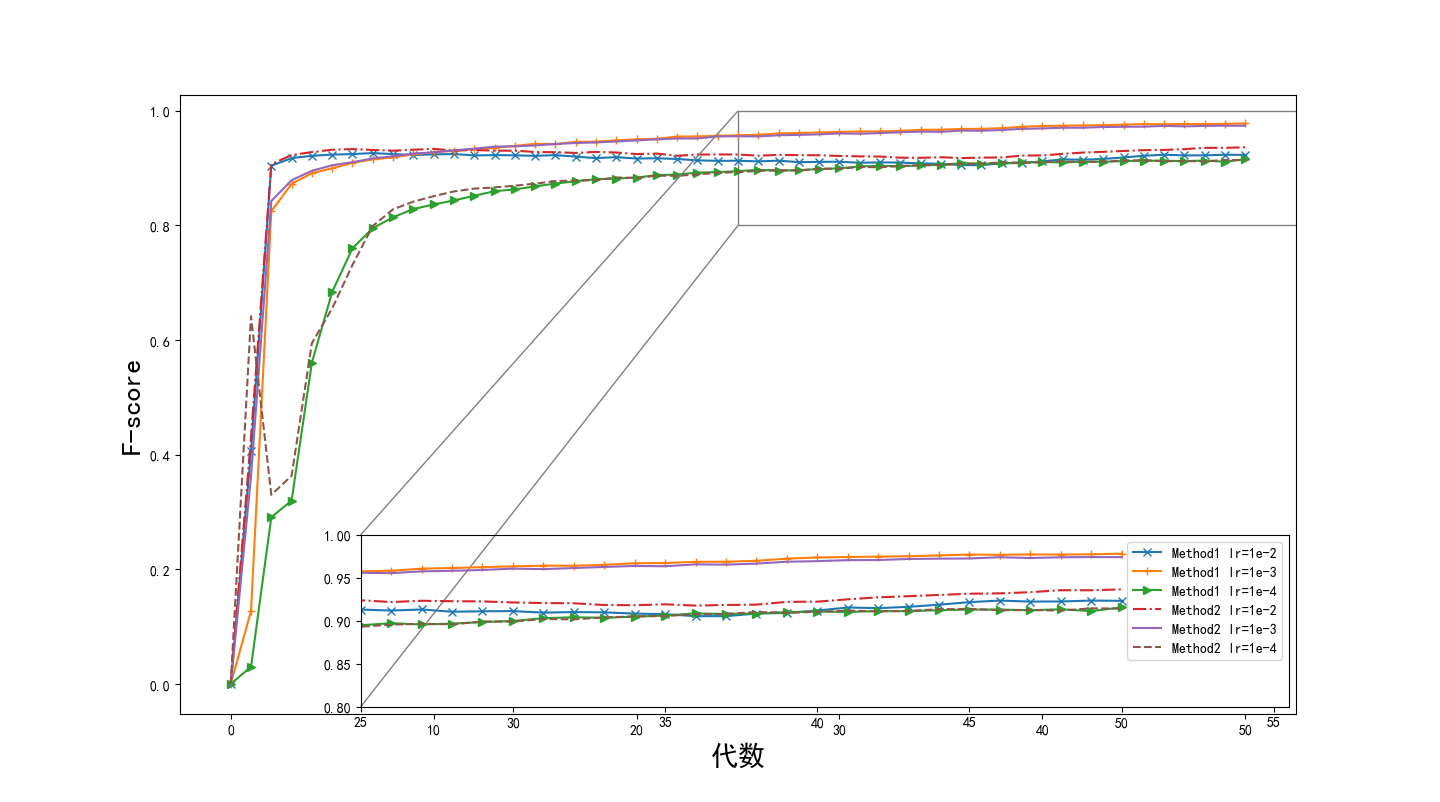
（1）嵌入层解决的是古汉语自然语言的数字化表示并嵌入到模型的问题。古汉语是一种自然语言无法被计算机直接处理，因此古汉语语句在通过神经网络模型进行处理之前必须首先将其数字化。该模型是将字表示为分布式向量，也称为嵌入向量。前面部分已经介绍过，在中文自然语言处理过程中有一个从训练集中提取出的大小为的字符字典，每个字符表示为实值向量（字符嵌入）其中d是向量空间的维数。然后将所有字符嵌入堆叠成嵌入矩阵。对于字符，查找表C检索相应的字符嵌入。查找表和嵌入层之间可以被视为简单的投影层，每个字符嵌入通过其查找表索引到相应的列操作来实现，然后将字符向量送入到相应的神经网络中。

（2）模型神经网络层是古汉语分词与词性标注一体化模型的核心，是采用深度学习方法进行古汉语分词特征学习与古汉语词性标注特征学习的重要的中间层，本部分模型网络层采用Bi-LSTM神经网络，其最主要的作用是针对输入的字向量数据，在训练过程通过不断调整神经网络参数来进行高度抽象的古汉语特征的学习，使之自动提取出字、分词、词性标注和上下文之间的内在关联。

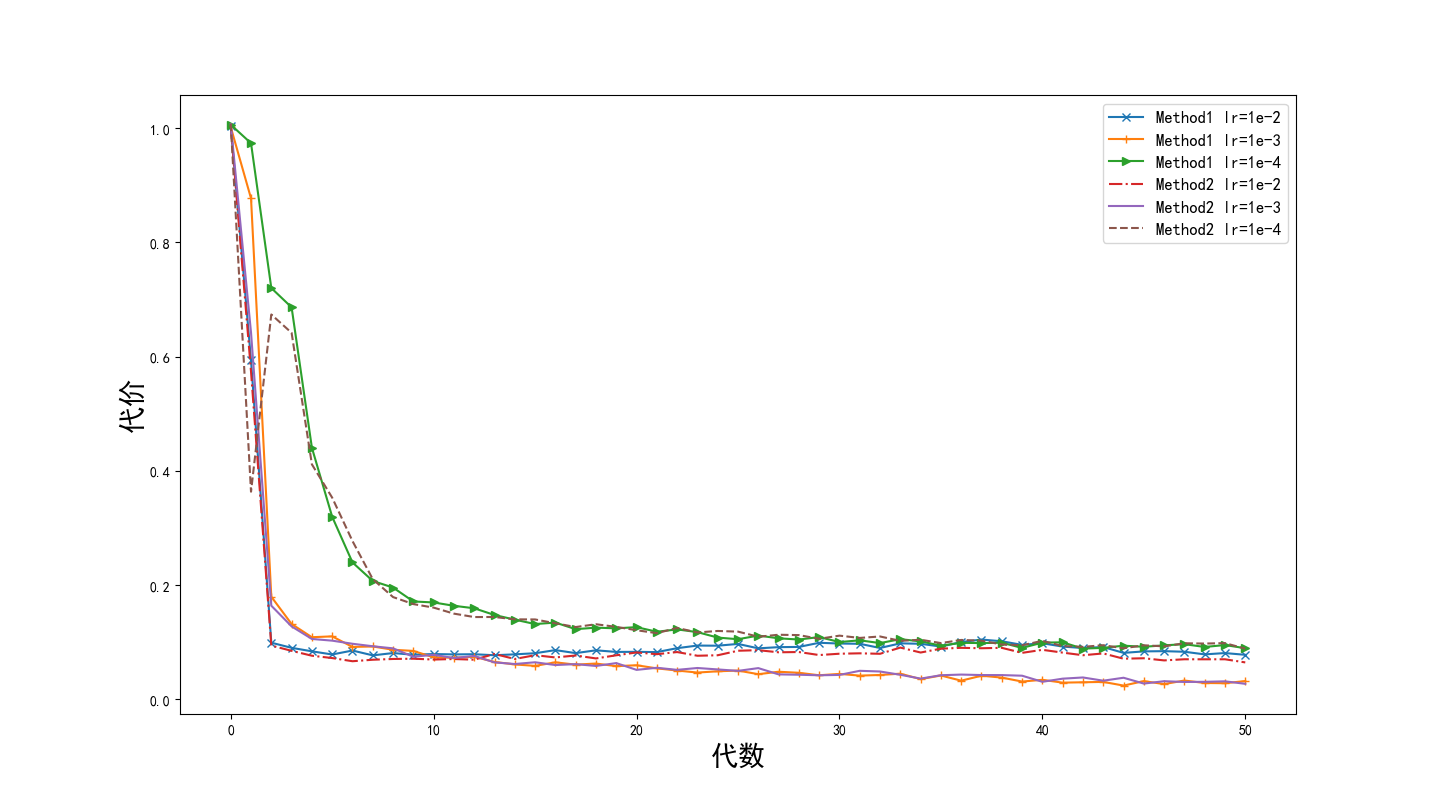
（3）标注推理层本质上是一个隐马尔科夫过程，它可以在句子层级上利用上下文标注信息推断当前字对应的词位标记或当前词对应的词性标记。古汉语所具有的语法关系会影响标记，标注推理层通过标记之间前后的互相影响计算一个句子的最优标注分数。

### 实验及性能分析

本模型通过Python下的TensorFlow框架搭建，使用的数据包括《尚书》、《礼记》、《诗经》、《论语》等含有分词和词性标记的上古书籍文本。该语料库中使用普及化标记标记上古汉语文本，且所有语料文本均使用Unicode编码。实验选取上古语料中75%文本作为训练集对模型进行训练，25%作为测试集进行后期实验性能分析和比较。训练过程中利用Adam(Adaptive Moment Estimation)方法对参数进行调节，该方法能分别计算每个参数的自适应学习率，可使模型参数达到较好的效果。图5-8、图5-9是训练过程中二元标签组的两种编码结构在不同步长情况下的精确度及代价随代数变化的比较图。图5-8中可见，第一种编码方式的情况下，学习率为1e-2的时候上升速度最快，但精确度最高稳定在92.7%；1e-3的学习率曲线上升速度虽然不如1e-2快，但能达到更高的精确度97.7%；1e-4学习率情况下由于学习率过小，正确率曲线在300批次之后上升十分缓慢，最终只可达到91.8%。与此同时，编码方式二的情况下，正确率曲线在不同步长参数下的变化趋势基本相同，但在三种步长下结构二所能达到的最优正确率仅为1e-3情况下的96.8%，达不到方式一中1e-3时的97.7%正确率。造成这种情况的原因，一方面可能因为编码方式二是分前后两段的，则分词和词性标注的过程独立性相较方式一更高，不能充分的利用到分词标志和词性标志之间的内在联系，另一方面可能是因为参数未设置到方式二的最佳状态。在下面的实验中，我们均选择1e-3步长下利用方式一编码训练得到的模型参数进行实验，以获得最好的实验效果。



**图5-8 两种输出结构下不同学习率下正确率随批次增加的变化示意图**

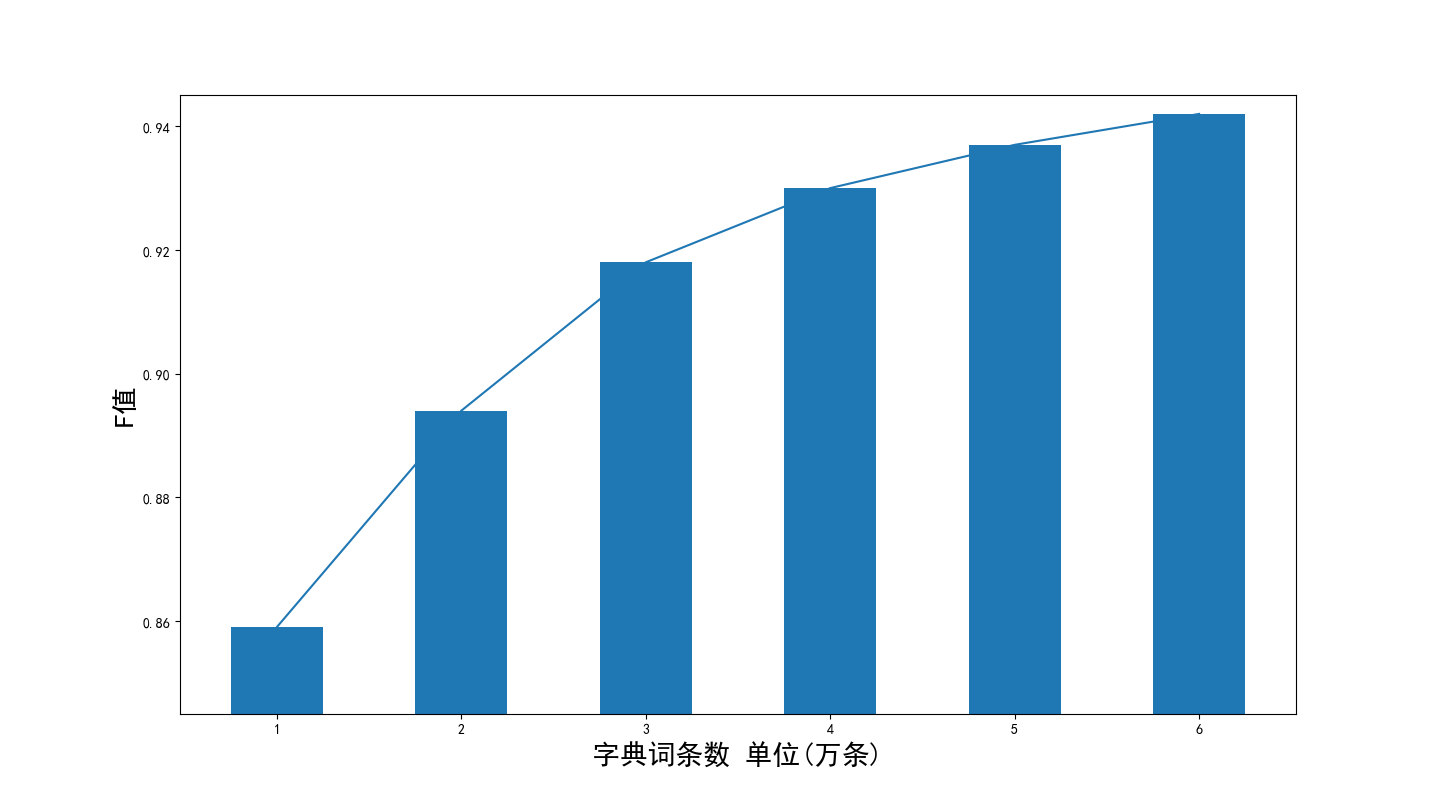


**图5-9 两种输出结构下不同学习率下正确率随批次增加的变化示意图**

本部分将本文提出的方法同基于字典的分词方法以及基于马尔科夫链的标注方法分别进行性能上的比较。该部分使用F值（F-measure）作为主要指标评估标准对不同方法的分词标注结果进行统一衡量。一个完美的分词器其F值为1，F值越大说明分词器分词标注效果越好。

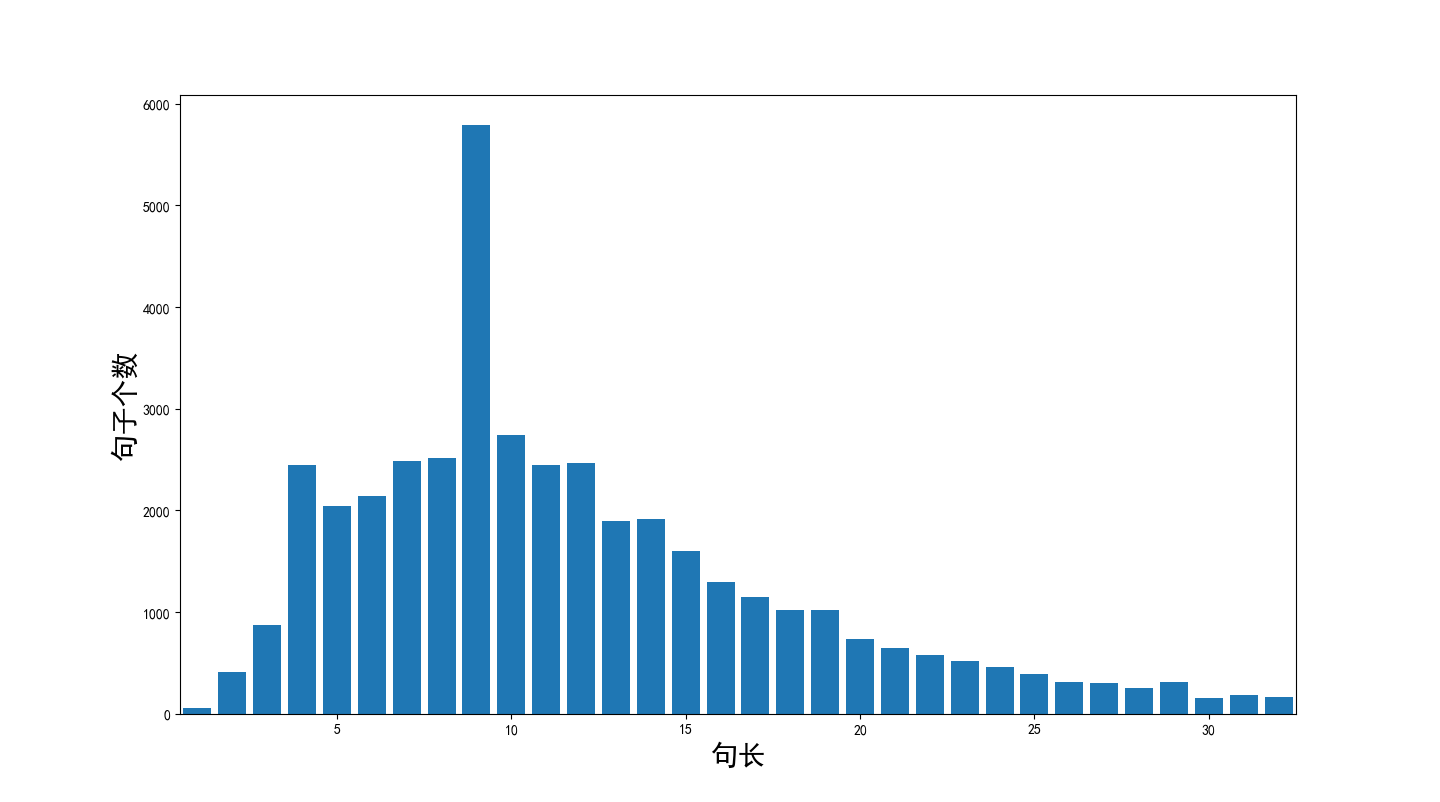
#### 分词

字典匹配法中使用的字典是包含上古语料库中提取出的包含所有词语的理想词典，即语料库中所有的词均包含在字典中，上古汉语中70%以上都是单字成词，对此本文针对性的使用最大长度为4的基于字典的分词方法。图5-10为在不同字典大小情况下字典法F值的比较图，可见字典的大小对分词精确度有直接影响，这里利用无未登录词的理想字典同本文模型进行比较，图中可见理想字典匹配法F值可达0.94。

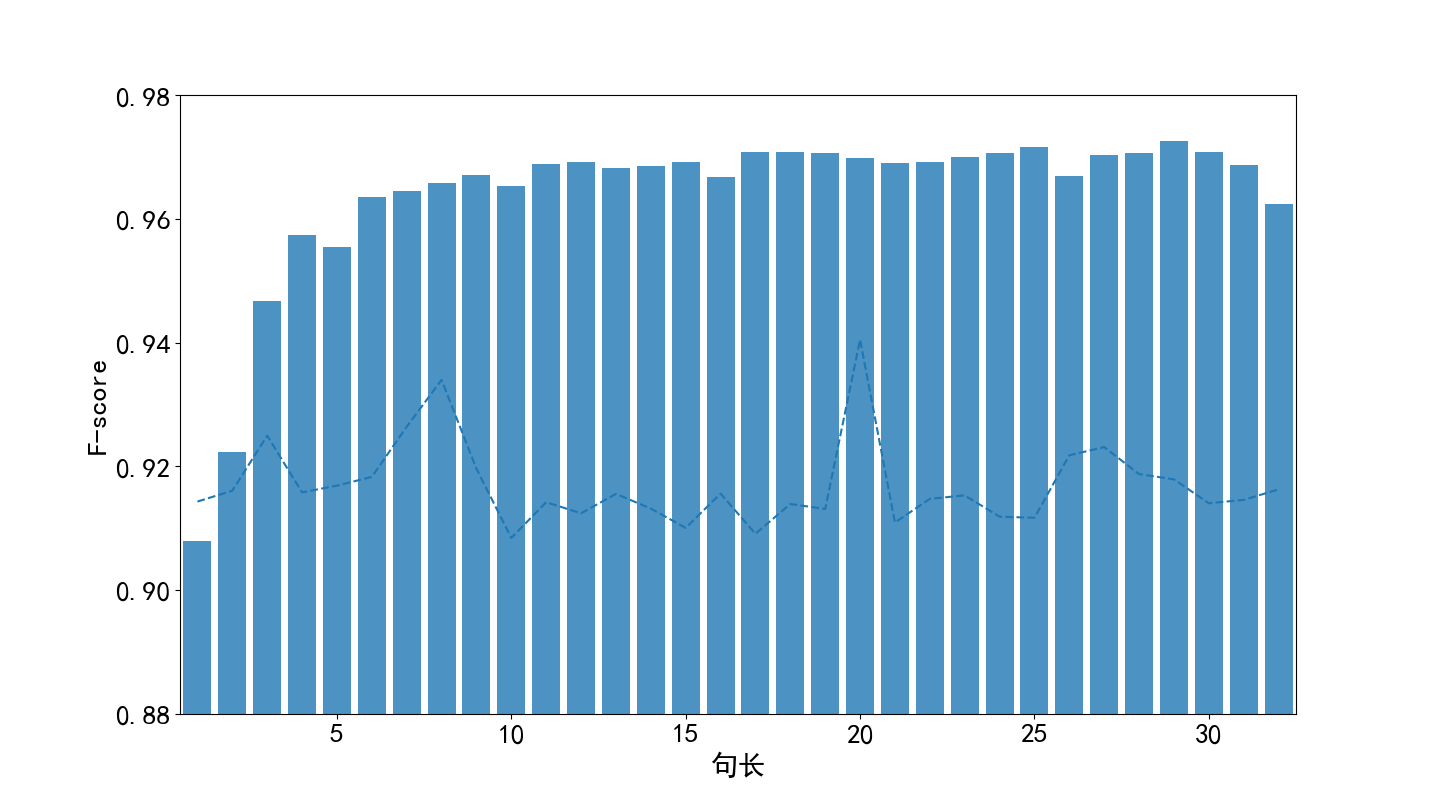


**图5-10 不同大小字典分词F值比较图**

图5-11是不同句长的句子个数分布图，图中某条形越高，说明该条形对应的句长在语料库中越多，图中可以看出，句子长度大多分布在4字到15字之间，其中含有九个字的句子最多。图5-12是利用上古预料库75%语料训练的Bi-LSTM一体化模型并利用25%作为测试集进行的测试结果条形图，其中虚线是利用理想字典的最大匹配算法的F值。实验中一体化模型的测试集不包含于训练集当中，但利用训练集训练得到的模型对测试集的分词及标签F值仍然可以达到0.95，效果优于理想字典匹配法的0.91的单任务自动分词F值。图5-11显示在句长小于5的情况下，标签的正确率稍低，原因是句长过短，LSTM网络从上下文中提取的信息有限，影响了标签正确率。虽然句长小于10的情况下分词F值略低，但图5-12所展示的句长分布可见包含五个字以下的句子在上古文本中所占比例并不大，所以对于短句对于正确率的影响也可控制在可接受的范围之内。



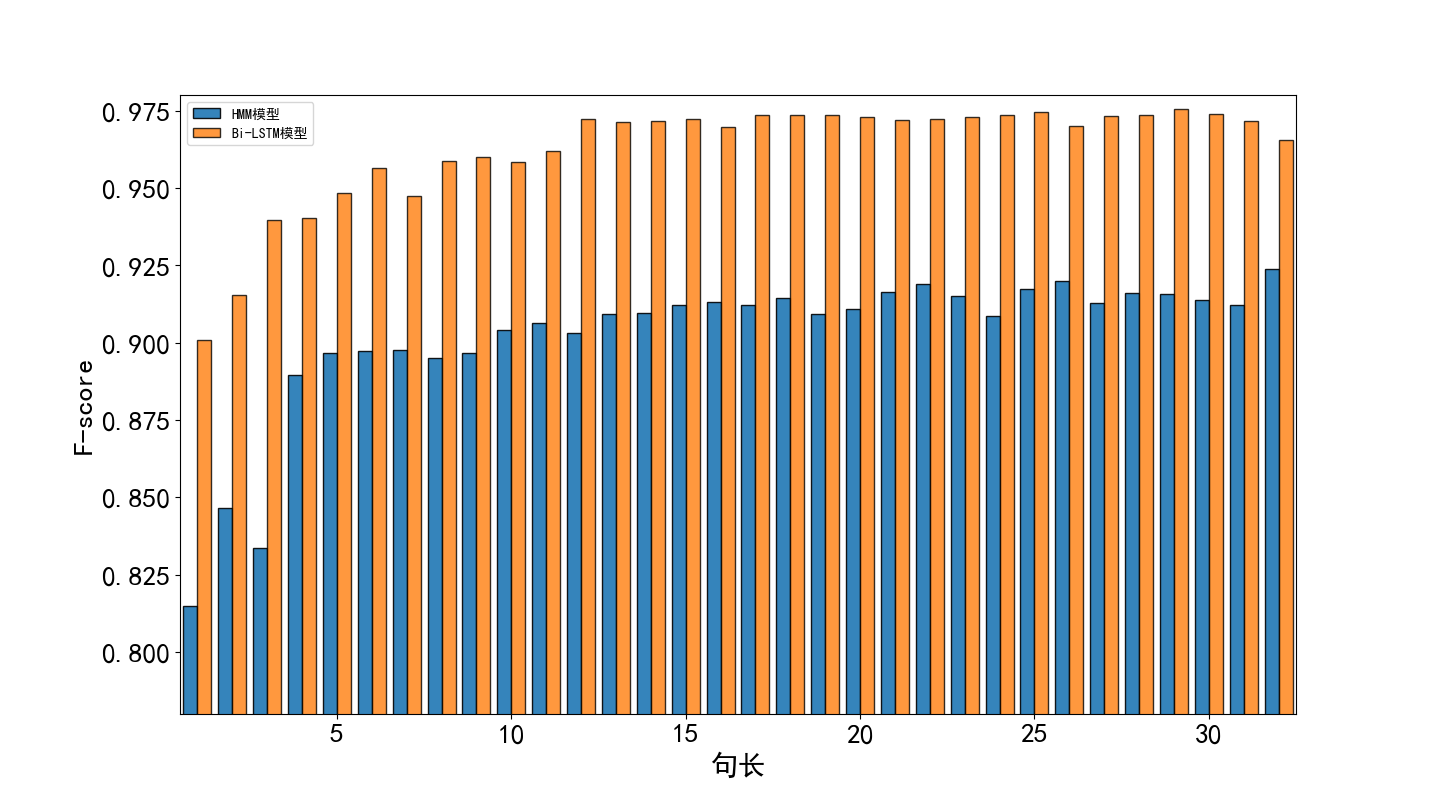
**图5-11 句长分布图**



**图5-12 Bi-LSTM一体化模型和字典法在不同句长下分词F值**

### 词性标注

本部分我们将一体化模型同隐马尔科夫链模型（Hidden Markov Model）进行词性标注上的比较，这里使用的HMM模型使用的参数是通过统计所有已有语料文本得到的。HMM模型在已经分好词的现代汉语上进行词性标记可达到95%的准确率，而应用到语料较少的上古汉语中则效果较差，如图5-13中深色条形图所示，在不同句子长度的情况下，其F值仅0.90左右，图中浅色条形图为本一体化模型在不同句长下的词性标注F值情况。可以看出由于古代汉语语料较少，而HMM是基于概率的模型，因此HMM方法的F值受到一定的影响。而本部分提出的一体化模型是基于上下文信息以及字词内部深层次的关联来输出标签，在语料相对较少的上古汉语语料集中也有较好的表现。



**图5-13 HMM方法与一体化模型在不同句长下的词性标注F值比较图**

通过上述实验可得，本文提出的一体化模型在分词效果上比基于理想字典的匹配法正确率高，在词性标注上效果优于利用所有语料进行参数设置的HMM法。查阅文献可知，当前对于古汉语来说，分词所能达到的最佳性能为文献[70] 中达到的97.7%正确率，词性标注目前能达到的最佳性能为文献[71] 中达到的97%正确率。如果将以上两种分词和词性标注单任务领域的最佳方法串联在一起，则其正确率将会变为二者正确率的乘积，又将有一定程度上的损失而无法达到97%。而本文的一体化模型从最开始就将分词和词性标注任务一同考虑，不存在错误扩散的问题，结果更加精确。

#### 实验结果分析

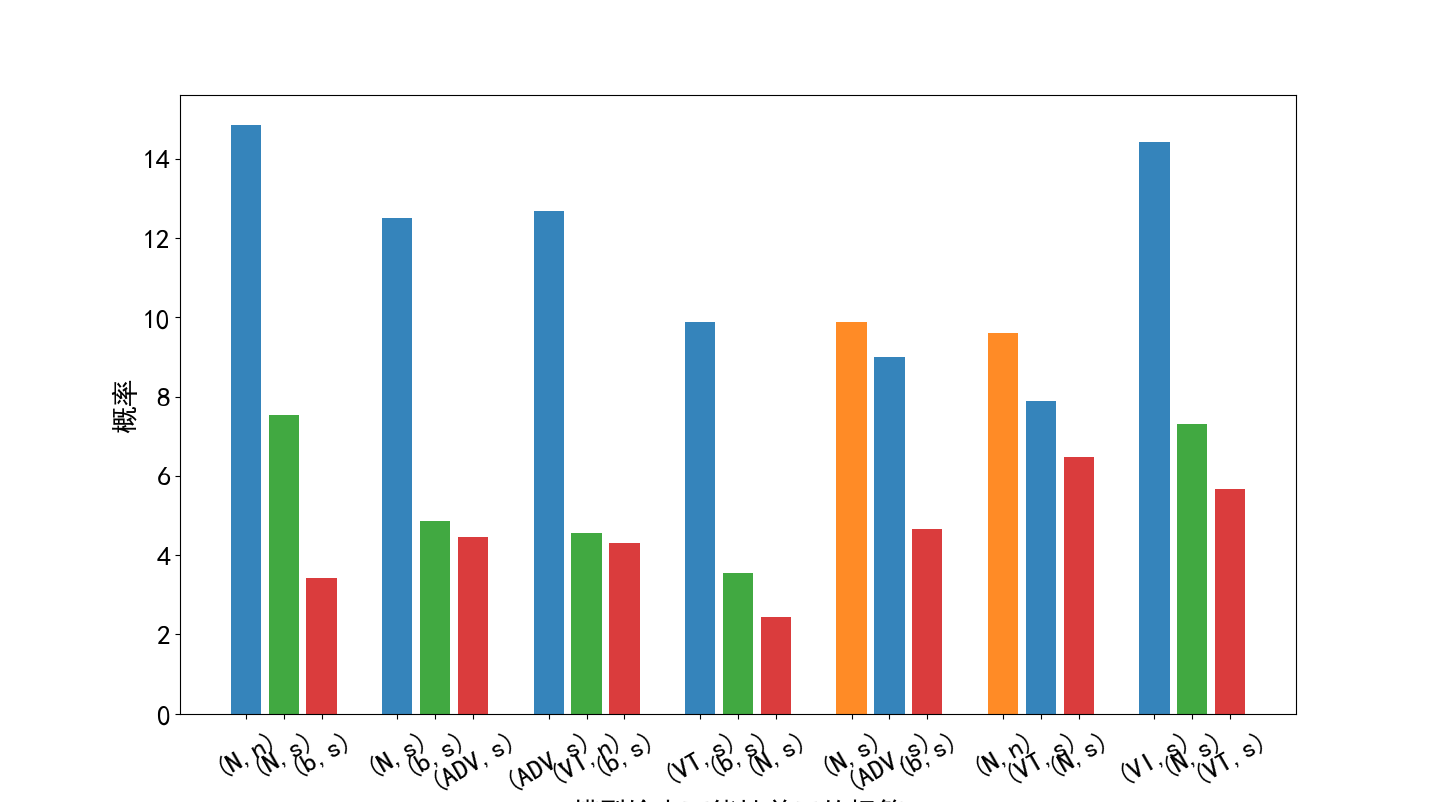
正如古汉语计算语言学家尉迟治平的呼吁：“我们期望能有可以用于汉语史电子文献自动分词、自动断句、自动标注的软件早日问世，专家只需对结果刊谬补缺，这将大大减轻属性式标注的劳动强度，加快工作进度。”

本模型不仅可输出可信度最大的二元标签，而且可输出第二大、第三大等标签的可信度作为辅助供使用者对结果刊谬补缺。表5-1是通过一体化模型对句子进行自动分词及词性标注的一个实例，其中第一行是输入句子的单字，第二行是输入汉字的正确二元标签，第三行为本模型的输出二元标签，最后一行表示模型的输出与正确标签是否一致，‘1’表示在该位置模型预测正确，‘0’则表示模型预测失败。

**表5-1 《睡虎地秦墓竹简·为吏之道》实例**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 输入 | 民 | 心 | 將 | 移 | 乃 | 難 | 親 |
| 正确标签 | (N,n) | (N,s) | (ADV,s) | (Vt,s) | (ADV,s) | (Vt,s) | (VI,s) |
| 预测标签 | (N,n) | (N,s) | (ADV,s) | (Vt,s) | (N,n) | (N,s) | (VI,s) |
| 是否相同 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |

如图5-14展示了每个字前三名标签的分数，其中深色条形对应的分数为正确标注，可以看出前4个字最大与次大标签评分差距较大，且评分最大标签均为正确标签。而第五、第六个字最大与次大标签评分差距较小，评分差距较小是程序对该字判断模糊的表现，该两字的正确答案均为模型评分第二大的标签。所以输出评分前三高的标签作为辅助可以覆盖更多的正确标签，共古汉语研究者进行手动选择，这一特性将为古汉语研究者接下来的编辑和校对提供便利。



**图5-14 实例中每个字的前三名评分条形图**

### 语料库建设

通过分词及词性标注一体化模型，我们就可以对我们的语料库做标注处理。首先我们获取大量未标记的语料，手动标注其中的部分语料作为训练集来训练分词及词性标注一体化模型，得到训练好的模型后，我们又可利用一体化模型对语料库的未标注部分做进一步的自动标注，经自动标注的文本需要经过人工校验方可入库，用以更进一步训练分词及词性标注一体化模型。过程流程图如图5-15：

通过流程图展示的流程，我们处理了部分上古汉语存档入库。并且，我们利用建立了一个基于Apache+Django+MySQL的数据管理网站方便对数据库的管理。网站可实现古籍按时代分类、原文检索、断句辅助以及释义检索四项基本功能。



**图5-15 数据库建设流程图**

#### 基本功能介绍

网站首页如图5-16所示，网站的左侧为导航栏，可以选择相应的功能。下面我们将分别介绍网站的各项基本功能。按时代检索：用户在输入栏选择相应的时间，则网站输出对应时代所有书籍；原文检索：用户输入古籍原文中某字，网站展示所有带有该字的古文原文；释义检索：用户输入某字或一段文言文，网站输出该文本的翻译内容；断句辅助：用户可输入一段无标点文本，网站自动为该文本添加标点并进行展示。



**图5-16 语料库网站首页**

### 本章小结

古汉语自动分词及词性标注是古汉语研究电子化的基础，在中文研究领域逐渐受到重视。由于古汉语与现代汉语的一些不同，直接使用现代汉语的相关技术有一定的困难。本文针对古汉语自动分词及词性标注技术：一、提出一种二元组标签；二、提出一种自动分词及词性标注的一体化设计方案，同时考虑分词和标注；三、将Bi-LSTM用到了古汉语分词标注领域，有一定的创新性。实验效果表明，本模型的自动分词及词性标注正确率可以达到97%，较现有的HMM和字典法有一定的提升，且分词标注一体化效果优于当前性能最佳的分词和词性标注方法串联后得到的处理结果。四、初步建立了一个上古汉语语料库。

总结与展望

### 总结

基于我国现阶段古代汉语研究和现代机器学习方法结合相对欠缺的现象，本文针对古代汉语提出了的三项基本目标：实现自动古籍断代、实现古代文本自动切分、实现古代文本自动标注以及实现古籍的结构化入库。为了实现古代汉语信息化以及古代汉语的现代化研究，本文将深度学习与古汉语研究的基本任务相结合，利用Bi-LSTM深度神经网络完成古汉语任务，并利用得到的模型处理古籍文本，最终将古籍文本进行切分及标注，并入库管理。

本文的贡献可总结为以下几个方面：

（1）为解决古代书籍断代的问题，本文提出使用双向长短期记忆神经网络作为主体构建古代文本断代模型。整理互联网上现有的已知年代的文本作为训练集对模型进行训练。利用word2vec模型将文本中的每一个字被转换成一串高维向量，然后将文本包含的所有文字的字向量送入模型分析它们之间的非线性关系。最终，模型会输出一个该段文本的年代类别标签。实验结果表明利用Bi-LSTM神经网络构造的模型能够很好的完成断代任务，断代的正确率能达到80%以上。本文的断代模型提供了一种高效且准确的古文断代方法，这将节省古文研究工作者在文本断代过程中的时间消耗。

（2）针对某些古代汉语书籍原著中缺少标点符号的问题，本文提出一个断句模型。本部分我们通过深度神经网络对大量经过断句的古汉语文本进行学习，使断句模型自动学习到某一时期、某种题材的断句规则，从而实现输入一段无断句的文字序列，机器自动为其添加断句的效果。

（3）针对古汉语分词及词性标注任务，我们需要解决训练集获取的问题，分词标注任务需要已经分好词、标注好词性的文本来做模型的训练集，但目前目前尚没有公开的具有分词和词性标注的古汉语语料库。因此我们通过手工标记部分语料的方法得到了少量的数据集对我们所设计的分词标注模型进行少量的实验，用以验证本文提出的分词标注模型可以较好的完成古汉语分词标注任务。

### 展望

从模型数据角度来说，现有的古代汉语预料主要是未进行分词及词性标注的，本文中所有使用到的分词及词性标注训练集均为人工标注的，因此数据量较小，训练所得的模型正确率也有限。下一步研究工作将考虑结合机器学习和人工纠错，不断扩充古汉语语料库，不断循环迭代，模型和语料库互相促进，提高模型性能以及语料库容量。

从模型算法角度来说，由于数据量较小的关系，现有模型均基于Bi-LSTM神经网络，下一步研究工作可以采用其它大规模自然语言处理模型进行建模，可考虑使用现代汉语模型加在古汉语数据集上的fine-tune的方法训练模型。