汉语分词的定义

什么是汉语分词？

汉语分词就是将连续的汉字序列按照一定的规范重新组合成词序列的过程。

也就是使用算法把中文的汉字序列切分成有意义的词。

比如

输入：　　什么是汉语分词

可能输出：　　什么/是/汉语/分词

不同的算法会输出不同的分词情况。

汉语分词的原因

为什么要进行汉语分词？

英文是以词为单位的，词和词之间是靠空格隔开，而中文是以字为单位，句子中所有的字连起来才能描述一个意思。

I am a student. → 我是一个学生。

计算机可以很简单通过空格知道student是一个单词，但是却不能很容易地明白“学”和“生”两个字放一起才表示一个词。

所以，对汉字序列进行分词是非常有必要的。

汉语分词的原因

为什么要进行汉语分词？

对于同一个汉字序列，进行不同的分割，可能会有不同的意思。对于“女朋友很重要吗”这个汉字序列进行不同的划分：

女朋友/很/重要/吗

女朋友/很重/要吗

所以，词的正确切分是进行中文文本处理的必要条件。

汉语分词的难点

歧义无处不在

- 组合型切分歧义(在不同的语境下切分不同）

  我画了一/直线

  我/一直/在写呢

- 真歧义（多种切分情况都可以）

  南京市/长江/大桥

  南京/市长/江大桥

- 交叉歧义（多种歧义交织在一起）

汉语分词的难点

新词层出不穷

- **未登录词识别困难**

（1）未登录词没有明确边界，缺少英语中的分隔符、大小写、词的形态、冠词等语法信息

例：拖颖而出

（2）许多未登录词的构成单元本身可以独立成词

例：人名： 王聪明

- **通常每一类未登录词都要构造专门的识别算法**
- **识别依据**

内容构成规律（用字规律）
外部环境（上下文）

汉语分词的地位

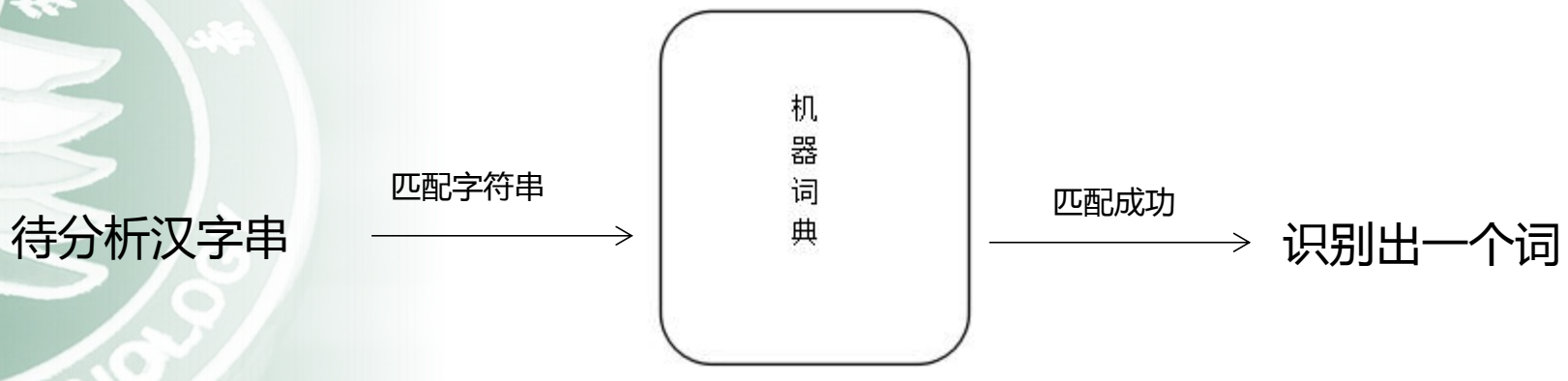
为什么要进行汉语分词？

互联网绝大部分应用都要用到汉语分词

- 信息检索：百度、搜狐、图书馆的检索系统...
- 内容分析：机器翻译...
- 文本校对：知网查重...
- 语音处理：语音识别...

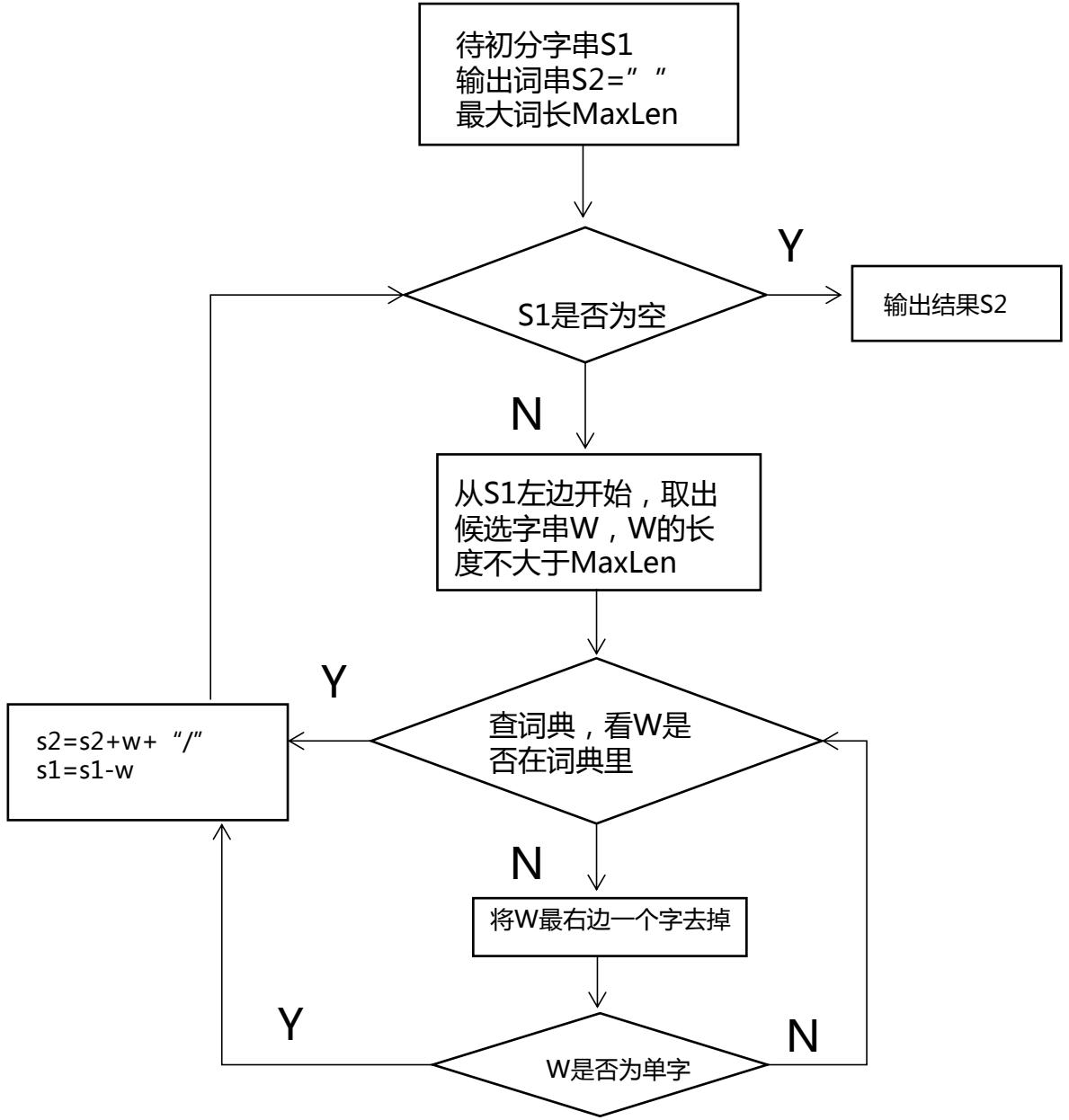
1 机械匹配算法介绍

机械分词方法又叫基于字符串匹配的分词方法，它是按照一定的策略将待分析的汉字串与一个“充分大的”机器词典中的词条进行区配，若在词典中找到某个字符串，则匹配成功（识别出一个词）。其主要原理都是切分出单字串，然后和词库进行比对，如果是一个词就记录下来，否则通过增加或者减少一个单字，继续比较，一直还剩下一个单字则终止，如果该单字串无法切分，则作为未登录处理。



1 机械匹配算法介绍

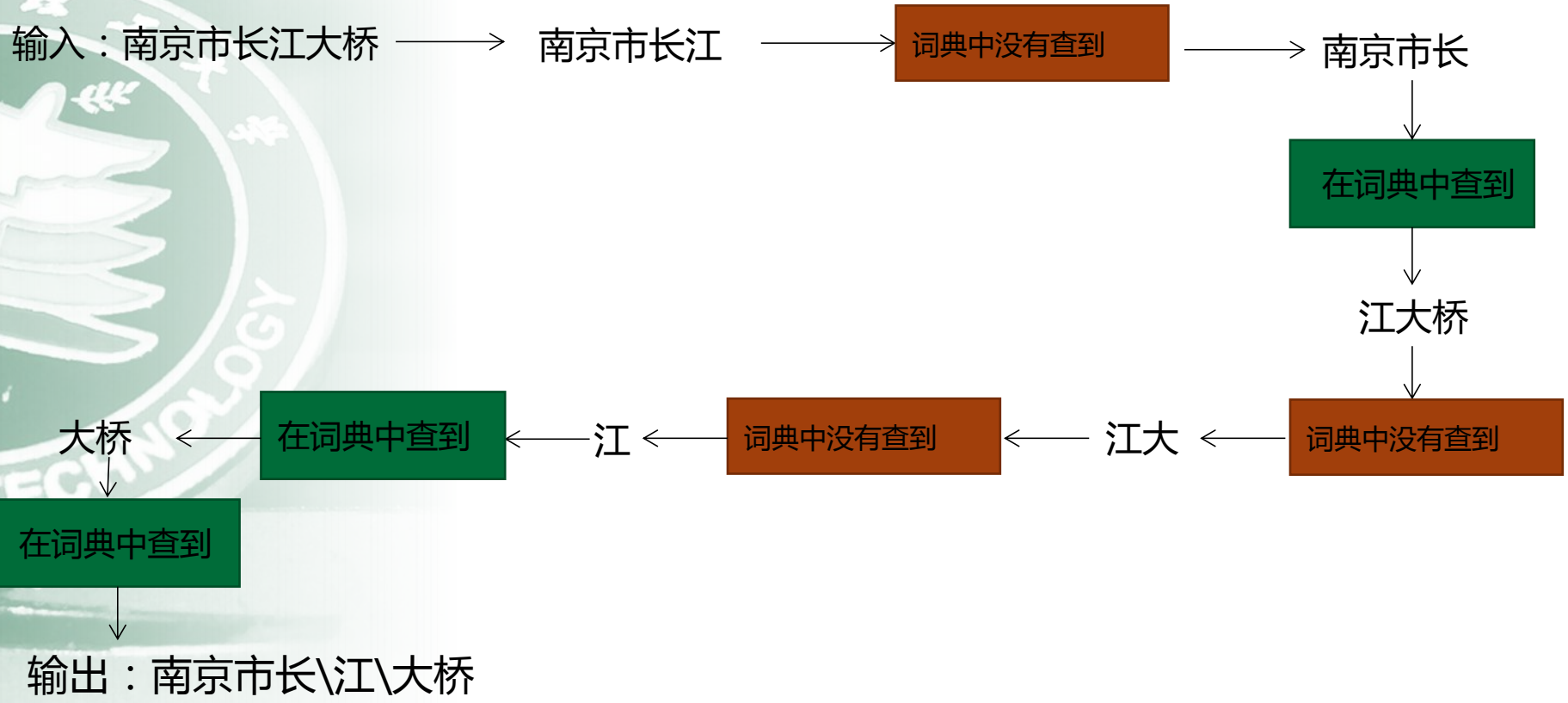
以最大正向匹配算法为例，输入待初分字串S1，并设定最大词长；判断S1是否为空，如果不为空，则从S1左边开始，取出候选字符串W,W的长度不大于最大词长；下一步将候选字符串w与词典中的字符串进行匹配，如果匹配成功，则从S1中去掉w,将剩下的字串继续进行匹配，如果没有匹配成功，则将w最右边的一个字去掉，并判断w是否为单字，如果是单字，则直接从S1中去掉，如果不是单字，则继续到词典中进行比较匹配，以此类推，直到s1中所有的词都匹配完成即s1为空。



2 最大正向匹配分词

S1 = 南京市长江大桥

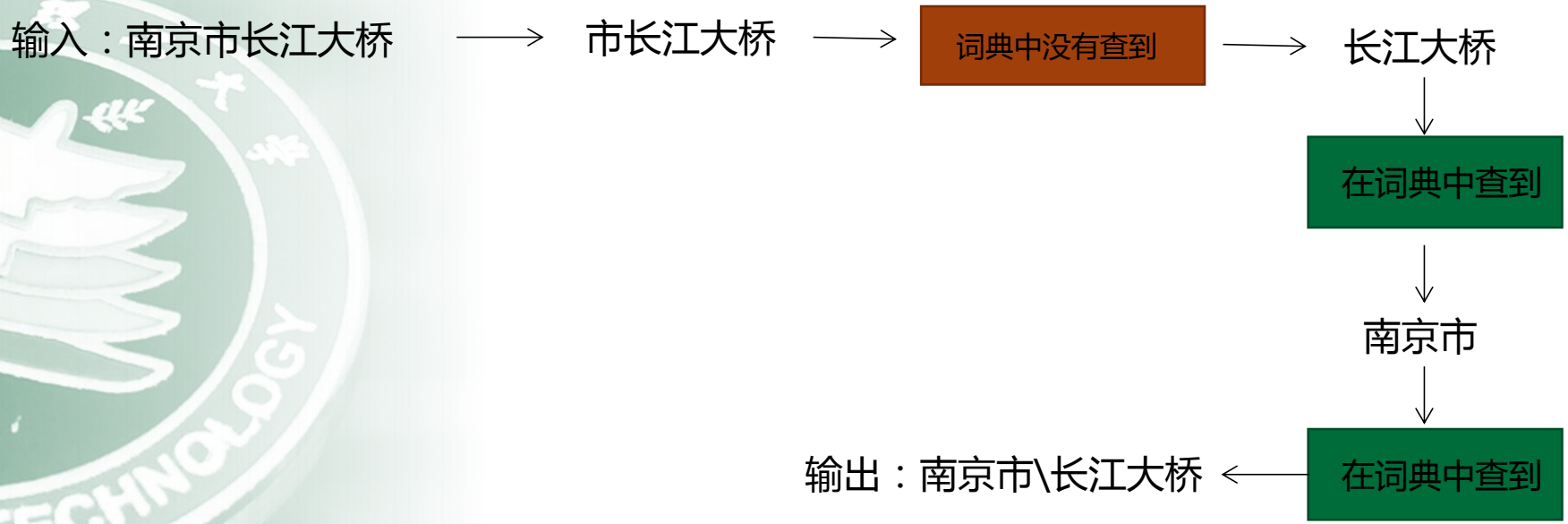
MaxLen=5



3 最大逆向匹配分词

S1 = 南京市长江大桥

MaxLen=5

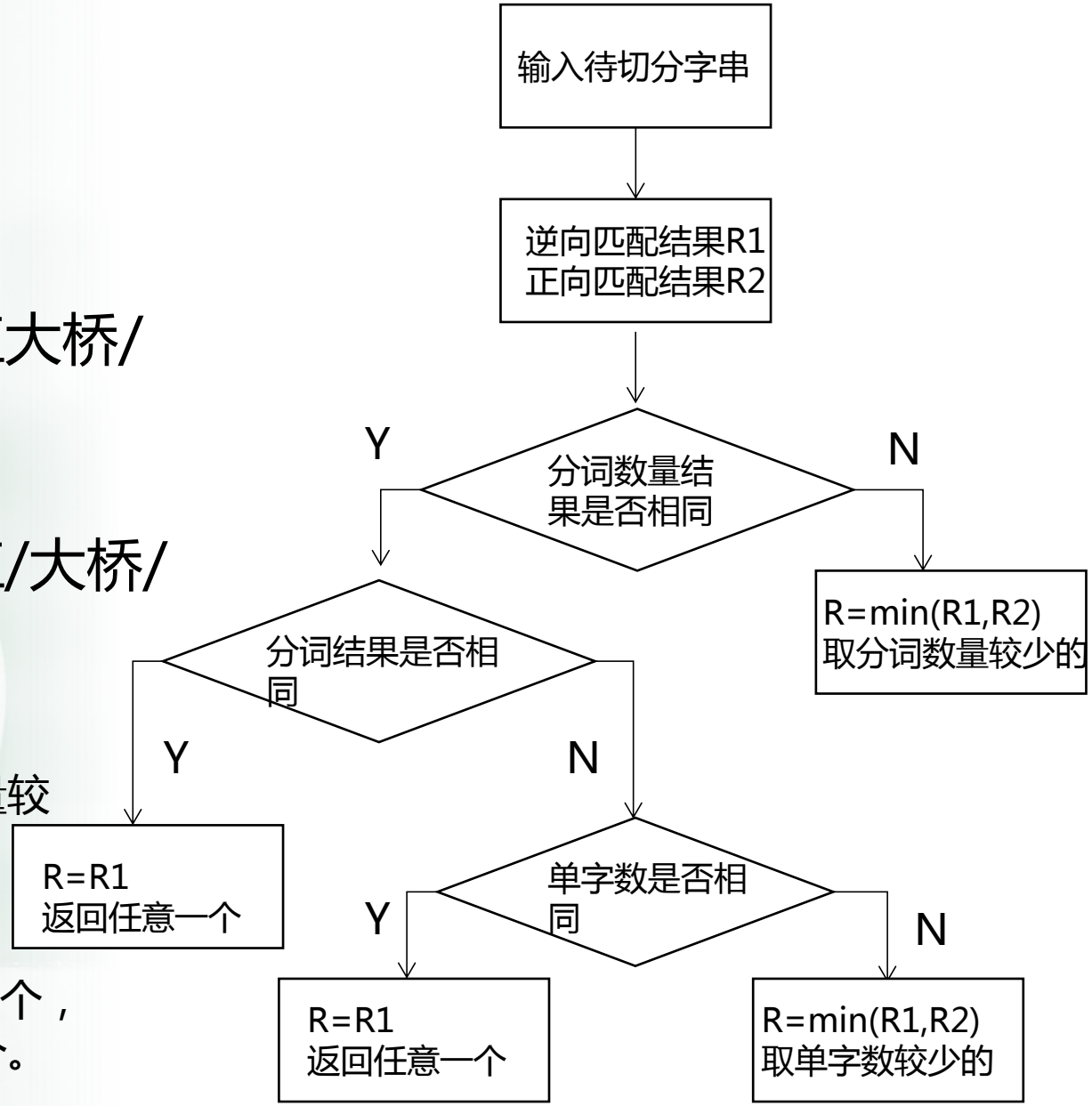


4 双向匹配

逆向匹配(R1)：南京市/长江大桥/

正向匹配(R2)：南京市长/江/大桥/

比较正向最大匹配和逆向最大匹配结果。
如果分词数量结果不同，那么取分词数量较少的那个。
如果分词数量结果相同：
1.分词结果相同，可以返回任何一个。
2.分词结果不同，返回单字数比较少的那个，如果单字数个数也相同，则任意返回一个。



5 总结

一般说来，逆向匹配的切分精度略高于正向匹配，遇到的歧义现象也较少。统计结果表明，单纯使用正向最大匹配的错误率为1/169， 单纯使用逆向最大匹配的错误率为1/245。但这种精度还远远不能满足实际的需要。实际使用的分词系统，都是把机械分词作为一种初分手段，还需通过利用各 种其它的语言信息来进一步提高切分的准确率。

统计分词方法介绍

基于统计的分词方法逐渐取代基于规则或词典的方法成为主流方法，其中有两个主要原因：

- **大规模语料库的建立**

  大规模语料集指的是包含大规模（一般最低在十万数量级以上）的中文句子的文档。

- **统计机器学习方法的研究和发展**
  - N-gram 模型
  - HMM 模型
  - ME 模型
  - CRF 模型

什么是语料库？

- 语料库中存放的是在语言的实际使用中真实出现过的语言材料，因此例句库通常不应算作语料库
- 语料库是承载语言知识的基础资源，但并不等于语言知识
- 真实语料需要经过加工（分析和处理），才能成为有用的资源。

统计分词方法的主要思想：

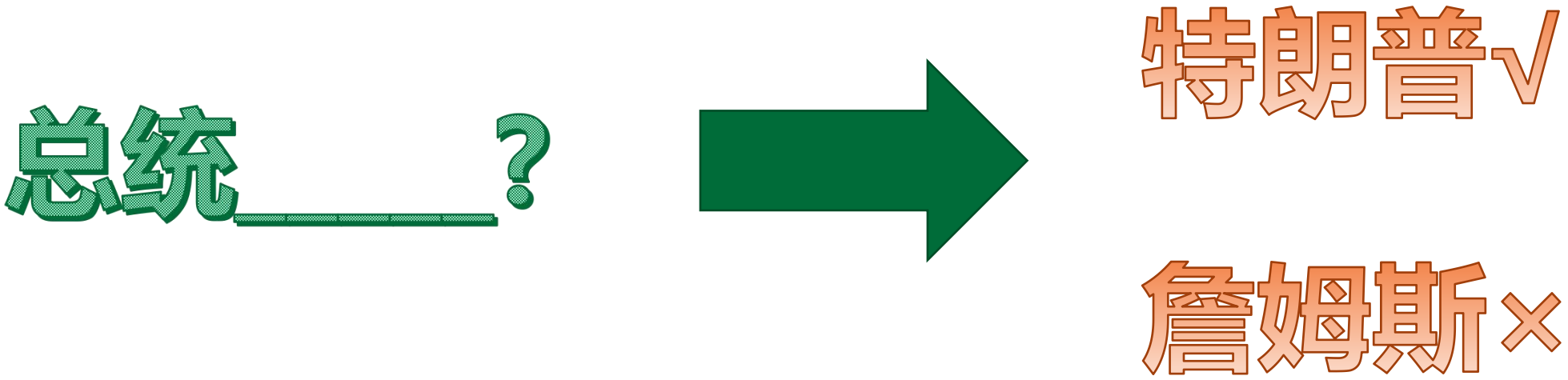
1. 把每个词看做是由词的最小单位各个字总成的，如果相连的字在不同的文本中出现的次数越多，就证明这相连的字很可能就是一个词。
2. 利用字与字相邻出现的频率来反应成词的可靠度，统计语料中相邻共现的各个字的组合的频度，当组合频度高于某一个临界值时，我们便可认为此字组可能会构成一个词语。

N-gram模型

N-gram模型主要思想：

假设第n个词的出现只与前面N-1个词相关，而与其它任何词都不相关，整句的概率就是各个词出现概率的乘积。

即我们给定一个词，然后去猜测下一个词是什么



N-gram模型

对于一个句子T，我们怎么算它出现的概率呢？

假设T是由词序列W1,W2,W3,…Wn组成的，那么这个句子产生的概率为

$$P(T)=P(W1W2W3…Wn)=P(W1)P(W2|W1)P(W3|W1W2)…P(Wn|W1W2…Wn-1)$$

但是这种方法存在两个致命的缺陷：

1. 参数空间过大，不可能实用化
2. 数据稀疏严重。

N-gram模型

为了解决这个问题，我们引入了**马尔科夫假设**：

一个词的出现仅仅依赖于它前面出现的有限的一个或者几个词。

如果一个词的出现仅依赖于它前面出现的一个词，那么我们就称之为bigram。

即：P(T) =P(W1W2W3…Wn)=P(W1)P(W2|W1)P(W3|W1W2)…P(Wn|W1W2…Wn-1)
≈P(W1)P(W2|W1)P(W3|W2)…P(Wn|Wn-1)

如果一个词的出现仅依赖于它前面出现的两个词，那么我们就称之为trigram。
以此类推，N元模型就是假设当前词的出现概率只同它前面的N-1个词有关

在实践中用的最多的就是bigram和trigram，高于四元的用的很少，因为训练它需要更庞大的语料，而且数据稀疏严重，时间复杂度高，精度却提高的不多。

最大熵(ME)模型

熵是对一个随机事件的不确定性的度量

最大熵原理：对一个随机事件的概率分布进行预测时，预测应当满足全部已知的约束，而对未知的情况不要做任何主观假设。在这种情况下，概率分布最均匀，预测的风险最小，因此得到的概率分布的熵是最大。

最大熵模型主要思想：

在学习概率模型时，所有可能的模型中熵最大的模型是最好的模型；若概率模型需要满足一些约束，则最大熵原理就是在满足已知约束的条件集合中选择熵最大模型。

最大熵模型：

给定数据集 $\{(x_i, y_i)\}_{i=1}^{N}$ ，特征函数 fi(x,y)， i=1,2…,n，根据经验分布得到满足约束集的模型集合 C ：

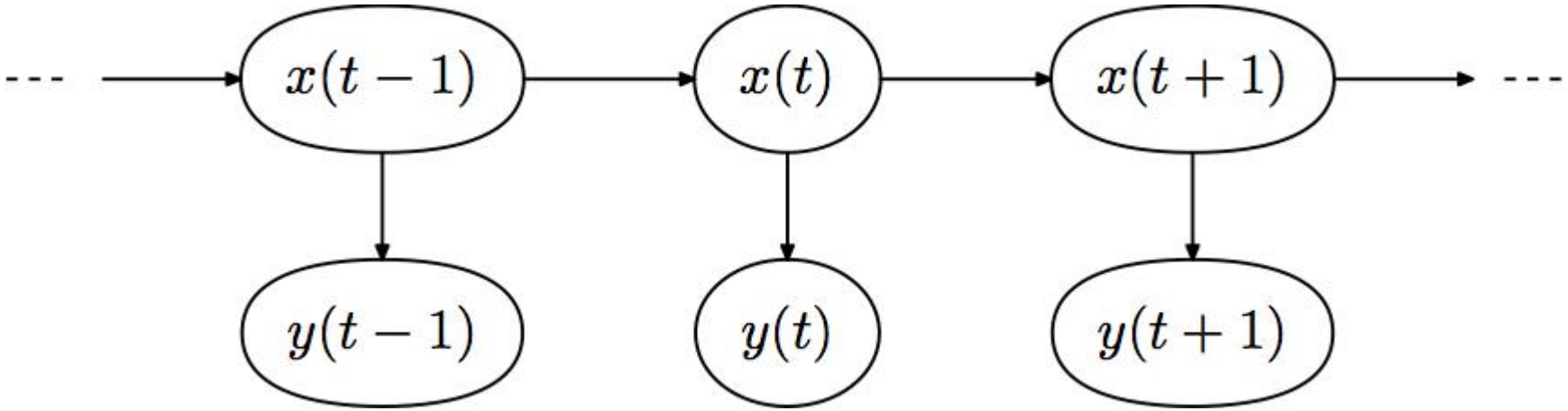
$$\min_{P \in C} \quad \sum_{x,y} \tilde{P}(x) P(y|x) log P(y|x)$$

$$s.t. \quad E_p(f_i) = E_{\tilde{p}}(f_i)$$

$$\sum_{y} P(y|x) = 1$$

隐含马尔可夫(HMM)模型

隐马尔可夫模型（Hidden Markov Model，HMM）是统计模型，它用来描述一个含有隐含未知参数的马尔可夫过程。



X为状态， y为观察值，假设观察到的结果为 Y=y(0),y(1),…,y(L-1)，隐藏条件为X=x(0),x(1),…,x(L-1)，长度为 L，则马可夫模型的概率可以表达为：

$$P(Y) = \sum_{X} P(Y \mid X)P(X)$$

隐含马尔可夫(HMM)模型

HMM模型是一个五元组:

- StatusSet: 状态值集合（隐状态）
- ObservedSet: 观察值集合（输出文字集合）
- TransProbMatrix: 转移概率矩阵（隐状态）
- EmitProbMatrix: 发射概率矩阵（隐状态表现为显状态的概率）
- InitStatus: 初始状态概率（隐状态）

HMM模型能解决三种问题：

- 参数(StatusSet, TransProbMatrix, EmitRobMatrix, InitStatus)已知的情况下，求解观察值序列。(Forward-backward算法)
- 参数(ObservedSet, TransProbMatrix, EmitRobMatrix, InitStatus)已知的情况下，求解状态值序列。(Viterbi算法)
- 参数(ObservedSet)已知的情况下，求解(TransProbMatrix, EmitRobMatrix, InitStatus)。(Baum-Welch算法)

其中，第二种问题是Viterbi算法求解状态值序列最常用，语音识别、中文分词、新词发现、词性标注都有它的一席之地。

隐含马尔可夫(HMM)模型

StatusSet & ObservedSet
状态值集合为(B, M, E, S): {B:begin, M:middle, E:end, S:single}
分别代表每个状态代表的是该字在词语中的位置，B代表该字是词语中的起始字，M代表是词语中的中间字，E代表是词语中的结束字，S则代表是单字成词。

小明硕士毕业于中国科学院计算所
输出的状态序列为： *BEBEBMEBEBMEBES*
根据这个状态序列我们可以进行切词: *BE/BE/BME/BE/BME/BE/S*
所以切词结果如下: 小明/硕士/毕业于/中国/科学院/计算/所

不难发现，B后面只可能接(M or E)，不可能接(B or S)。而M后面也只可能接(M or E)，不可能接(B, S)。

隐含马尔可夫(HMM)模型

InitStatus为初始状态概率分布，一般是对概率值取对数之后的结果(可以让概率相乘的计算变成对数相加)

```
#B -0.26268660809250016
#E -3.14e+100
#M -3.14e+100
#S -1.465263339853767 8
```

此数值是对概率值取对数之后的结果(可以让概率相乘的计算变成对数相加)，其中-3.14e+100作为负无穷，也就是对应的概率值是0。其实就是句子的第一个字属于{B,E,M,S}这四种状态的概率，E和M的概率都是0，这和实际相符合，开头的第一个字只可能是词语的首字(B)，或者是单字成词(S)。

隐含马尔可夫(HMM)模型

TransProbMatrix：转移概率矩阵

马尔科夫链最大的特点就是当前T=i时刻的状态Status(i) 只和T=i时刻之前的n个状态有关，也就是和{Status(i-1), Status(i-2), Status(i-3), ... Status(i - n)}有关。

HMM模型有三个基本假设作为模型的前提，其中有个“有限历史性假设”，也就是马尔科夫链的n=1。即Status(i)只和Status(i-1)相关，这个假设能大大简化问题。

TransProbMatrix其实就是一个4x4(4就是状态值集合的大小)的二维矩阵，示例如下：
矩阵的横坐标和纵坐标顺序是BEMS x BEMS(数值是概率求对数后的值）

-3.14e+100	-0.510825623765990	-0.916290731874155	-3.14e+100
-0.589714973685451 3	-3.14e+100	-3.14e+100	-0.808525047466993 7
-3.14e+100	-0.333448568119485 14	-1.260362382026822 6	-3.14e+100
-0.721196565466984 1	-3.14e+100	-3.14e+100	-0.665863144879821 2

比如TransProbMatrix[0][0]代表的含义就是从状态B转移到状态B的概率，由TransProbMatrix[0][0] = -3.14e+100可知，这个转移概率是0，这符合常理。由状态各自的含义可知，状态B的下一个状态只可能是ME，不可能是BS，所以不可能的转移对应的概率都是0，也就是对数值负无穷，在此记为-3.14e+100。

隐含马尔可夫(HMM)模型

EmitProbMatrix：发射概率矩阵

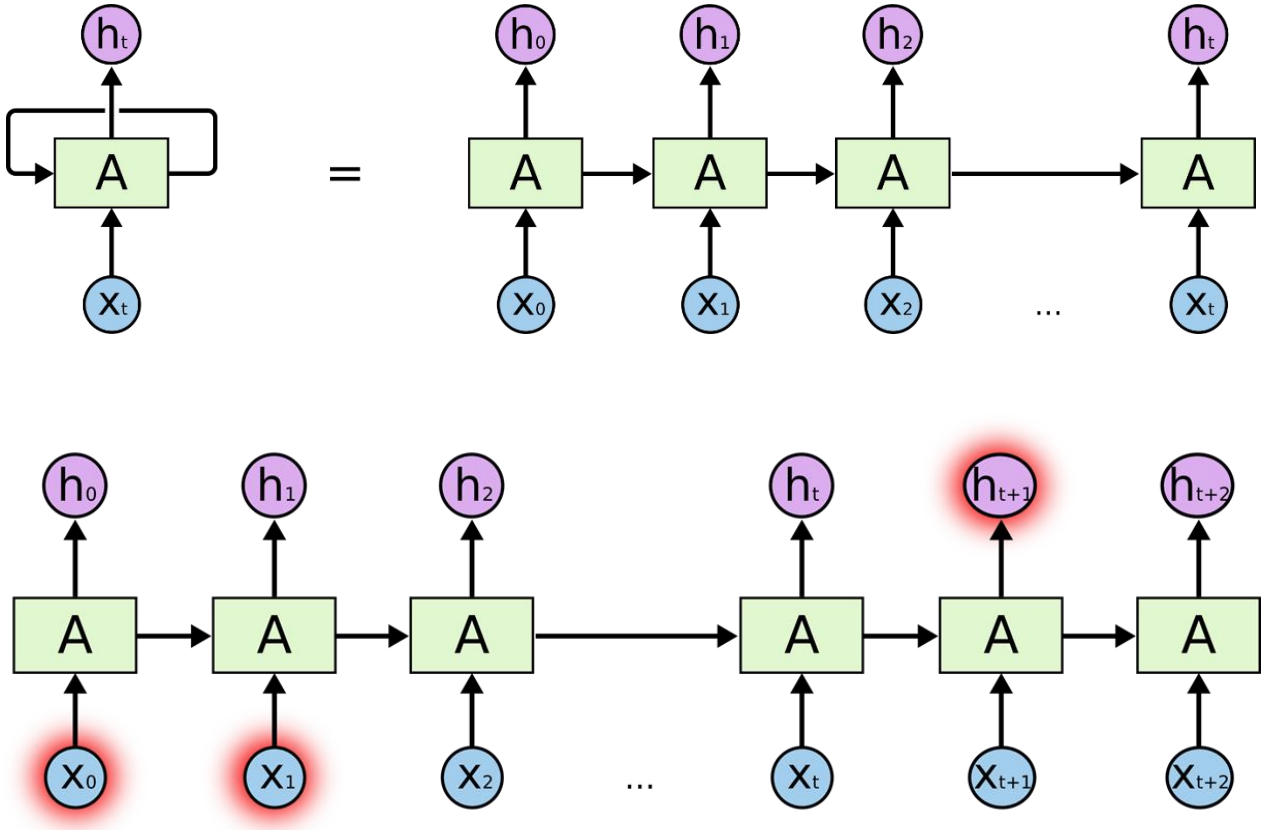
发射概率是一个条件概率，根据HMM模型三个基本假设里的“观察值独立性假设”，观察值只取决于当前状态值，也就是:P(Observed[i], Status[j]) = P(Status[j]) * P(Observed[i]|Status[j])，其中P(Observed[i]|Status[j])这个值就是从EmitProbMatrix中获取。

这五元的关系是通过一个叫Viterbi的算法串接起来， ObservedSet序列值是Viterbi的输入， 而StatusSet序列值是Viterbi的输出， 输入和输出之间Viterbi算法还需要借助三个模型参数， 分别是InitStatus, TransProbMatrix, EmitProbMatrix

Viterbi算法是一种动态规划算法，用于最可能产生观测时间序列的- Viterbi路径-隐含状态序列，特别是在马尔可夫信息源上下文和隐马尔科夫模型中 “Viterbi路径”和“Viterbi算法”也被用于寻找观察结果最有可能解释的相关动态规划算法，例如在统计句法分析中动态规划可以被用于发现最有可能的上下文无关的派生的词汇

RNN

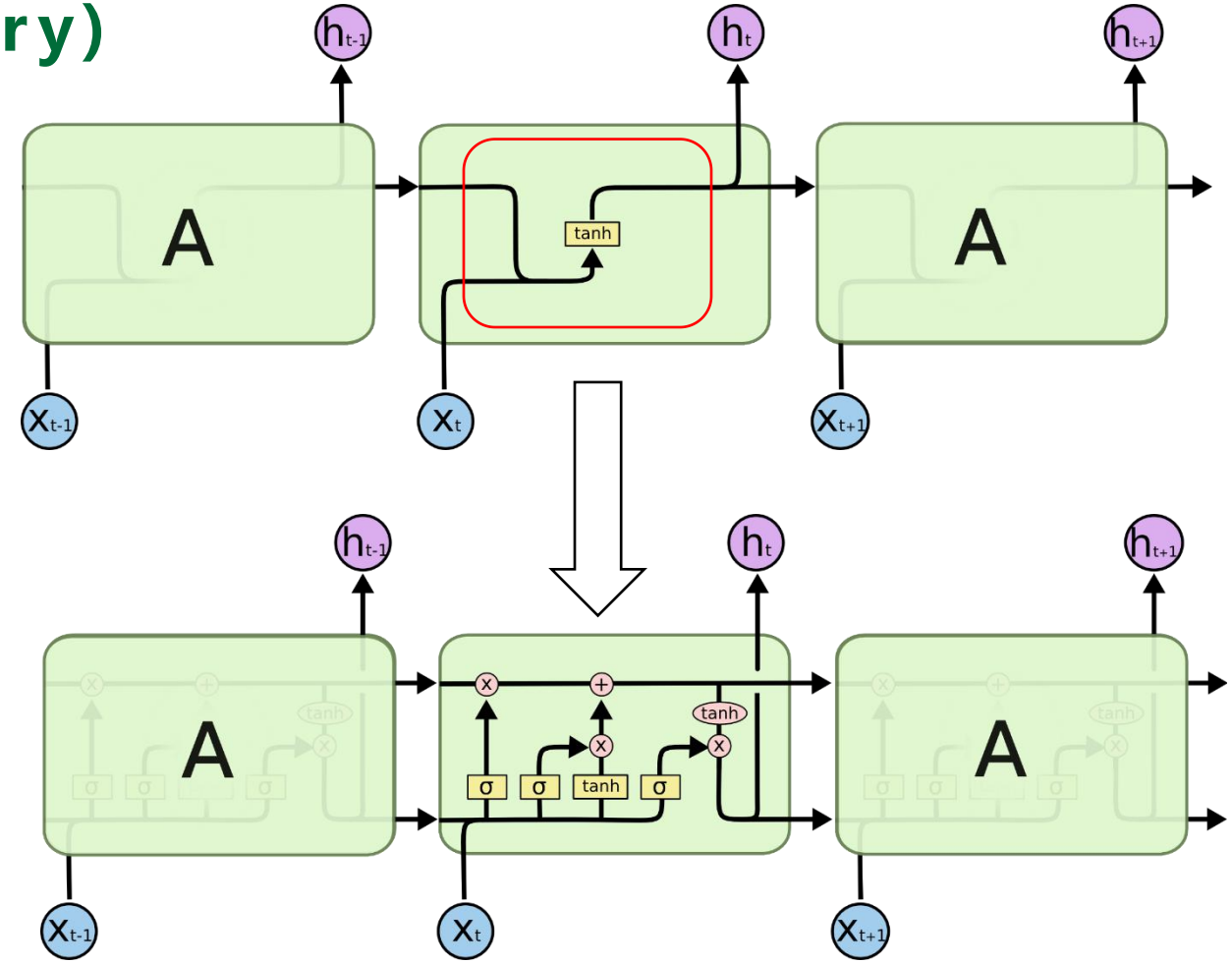
- 一个包含循环的网络，允许信息的持久性
- 处理序列数据
- 缺陷：长期依赖问题



LSTM(长短期记忆)

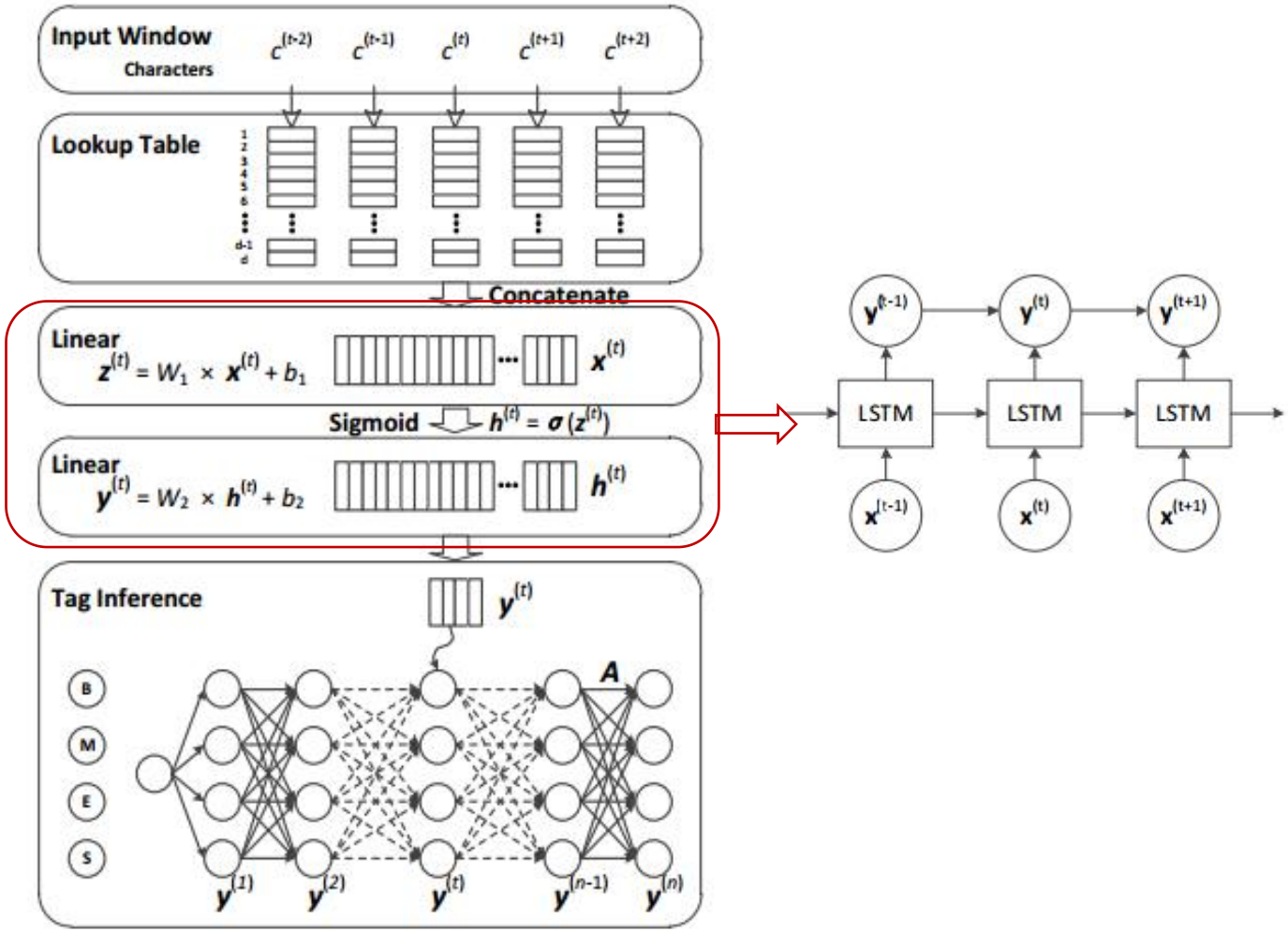
LSTM(Long Short-Term Memory)

- 是一种特殊的RNN
- 核心思想：细胞状态
- 引入“门”
  - 遗忘门：丢弃老细胞状态的信息
  - 输入门：更新细胞状态的信息
  - 输出门：输出细胞状态的信息
- 解决长期依赖的问题



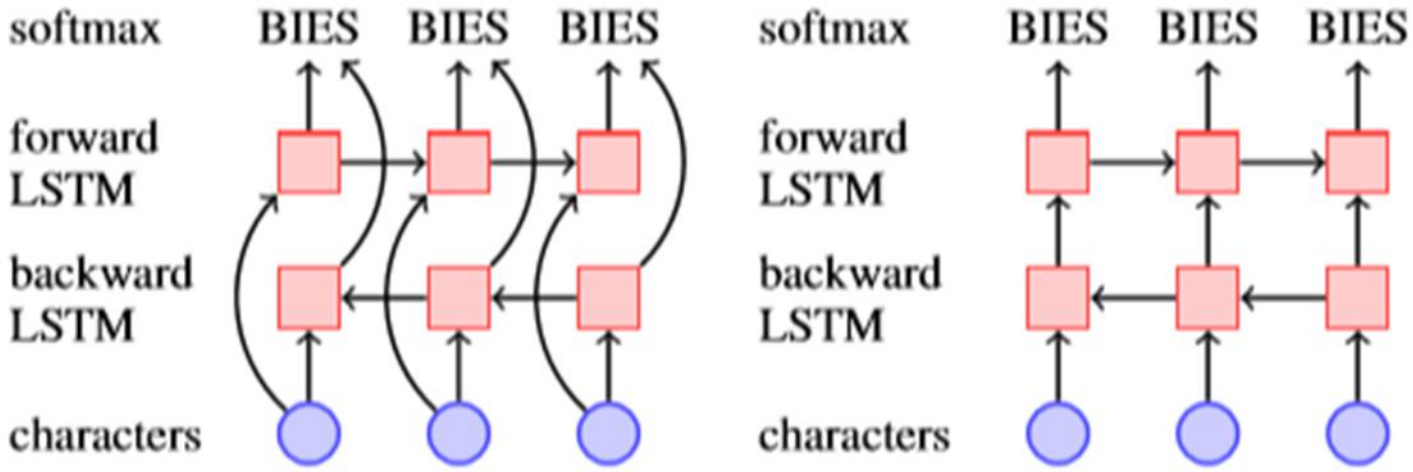
汉语分词

- 序列标注，标注标签为{B，M，E，S}
- 用于分词的神经网络
  - 用LSTM替换神经网络处理部分
  - 提出四种LSTM结构



Bi-LSTM

- 使用堆叠双向LSTM
- 网络简单
- 技巧
  - 预训练词向量
  - LSTM中加入dropout
  - 调整超参数



参考

[1]Chen X, Qiu X, Zhu C, et al. Long short-term memory neural networks for chinese word segmentation[C]//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015: 1197-1206.

[2] Ma J, Ganchev K, Weiss D. State-of-the-art Chinese word segmentation with bi-lstms[J]. arXiv preprint arXiv:1808.06511, 2018.

[3] https://colah.github.io/posts/2015-08-Understanding-LSTMs/

jieba

jieba分词是国内**使用人数最多**的中文分词工具。

- 支持三种分词模式：

  全模式、精确模式和搜索引擎模式

- 支持**繁体分词**
- 支持**自定义词典**

使用：

```
import jieba

#全模式
test1 = jieba.cut("我是一名北京理工大学的学生！", cut_all=True)
print("全模式: " + "| ".join(test1))
#精确模式
test2 = jieba.cut("我是一名北京理工大学的学生！", cut_all=False)
print("精确模式: " + "| ".join(test2))
#搜索引擎模式
test3= jieba.cut_for_search("我是一名北京理工大学的学生！")
print("搜索引擎模式:" + "| ".join(test3))
```

输出：

```
Loading model cost 1.192 seconds.
Prefix dict has been built succesfully.
全模式: 我| 是| 一名| 北京| 北京理工| 北京理工大学| 理工| 理工大| 理工大学| 工大| 大学| 的| 学生| |
精确模式: 我| 是| 一名| 北京理工大学| 的| 学生| ！
搜索引擎模式:我| 是| 一名| 北京| 理工| 工大| 大学| 理工大| 北京理工大学| 的| 学生| ！
```

THULAC

THULAC由清华大学自然语言处理与社会人文计算实验室研制推出的一套中文词法分析工具包，具有**中文分词**和**词性标注**功能。

- **能力强。**利用目前世界上规模最大的人工分词和词性标注中文语料库训练而成，**模型标注能力强大。**
- **准确率高。**
- **速度较快。**同时进行分词和词性标注速度为300KB/s，每秒可处理约15万字。只进行分词速度可达到1.3MB/s。

使用：

```
import thulac

#默认模式，分词的同时进行词性标注
test1 = thulac.thulac()
text1 = test1.cut("我是一名北京理工大学的学生！")
print(text1)

#只进行分词
test2 = thulac.thulac(seg_only=True)
text2 = test2.cut("我是一名北京理工大学的学生！")
print(text2)
```

输出：

```
[['我', 'r'], ['是', 'v'], ['一', 'm'], ['名', 'q'], ['北京', 'ns'], ['理工大学', 'n'], ['的', 'u'], ['学生', 'n'], ['！', 'w']]
Model loaded succeed
[['我', ''], ['是', ''], ['一', ''], ['名', ''], ['北京', ''], ['理工大学', ''], ['的', ''], ['学生', ''], ['！', '']]
```

pkuseg

pkuseg是由**北京大学语言计算与机器学习研究组**研制推出的一套全新的中文分词工具包。

- **高分词准确率**。相比于其他的分词工具包，该工具包在不同领域的数据上都大幅提高了分词的准确度。
- **多领域分词**。该分词包训练了多种不同领域的分词模型。根据待分词的领域特点，用户可以自由地选择不同的模型。
- **支持用户自训练模型**。支持用户使用全新的标注数据进行训练。

细领域分词在不同数据集上的对比结果：

MSRA	Precision	Recall	F-score
jieba	87.01	89.88	88.42
THULAC	95.60	95.91	95.71
pkuseg	96.94	96.81	**96.88**

WEIBO	Precision	Recall	F-score
jieba	87.79	87.54	87.66
THULAC	93.40	92.40	92.87
pkuseg	93.78	94.65	**94.21**

NLPIR

NLPIR分词系统是由**北京理工大学张华平博士**研发的中文分词系统，拥有丰富的功能和强大的性能。NLPIR是一整套对**原始文本集**进行处理和加工的软件，提供了中间件处理效果的可视化展示，也可以作为小规模数据的处理加工工具。主要功能包括：**中文分词**，**词性标注**，**命名实体识别**，**用户词典**、**新词发现**与**关键词提取**等功能。

```
import pynlpir

pynlpir.open()#打开分词器

#分词，默认打开分词和词性标注功能
test1 = pynlpir.segment("我是一名北京理工大学计算机学院的学生！")
print('默认分词模式:\n' + str(test1))

#将词性标注语言变更为汉语
test2 = pynlpir.segment("我是一名北京理工大学计算机学院的学生！",pos_english=False)
print('汉语标注模式:\n' + str(test2))

#关闭词性标注
test3 = pynlpir.segment("我是一名北京理工大学计算机学院的学生！",pos_tagging=False)
print('无词性标注模式:\n' + str(test3))
```

```
[('我', 'pronoun'), ('是', 'verb'), ('一', 'numeral'), ('名', 'classifier'), ('北京理工大学', 'noun'), ('计算机', 'noun'), ('
汉语标注模式:
[('我', '代词'), ('是', '动词'), ('一', '数词'), ('名', '量词'), ('北京理工大学', '名词'), ('计算机', '名词'), ('学院', '名词'
无词性标注模式:
['我', '是', '一', '名', '北京理工大学', '计算机', '学院', '的', '学生', '！']
```