自然语言处理的前世今生

CONTENT

历史

语言模型

隐马尔可夫过程

深度学习

历史

什么是自然语言处理？

“人工智能领域中帮助计算机理解，处理以及运用人类自然语言的分支学科”

[wiki_自然语言处理]

Natural Language Understanding

自然语言理解

01

Part-of-Speech Tagging
词性标注

02

Name Entity Recognition
命名实体识别

03

Sentimental Analysis
情绪分析

Natural Language Generation

自然语言生成

01

Machine Translation
机器翻译

Speech Recognition
语音识别

02

03

Question Answering
问答系统

自然语言发展历史

- 1948: Shannon把离散马尔可夫过程的概率模型引入描述语言的自动机
- 1956: Kleene提出正则表达式
- 1956: Chomsky提出上下文无关语法
- 1957-1970:
  - 基于规则方法的符号派: 形式语言理论和生成句法，形式逻辑系统研究
  - 统计学派: 基于贝叶斯方法的统计学研究方法
- 1960: 隐马尔可夫过程
- 1980: 循环神经网络
- 2011: Collobert证明深度学习的有效性
- 2013: Word2Vec

自然语言处理的主要困难—消除歧义

词法分析歧义

“北京大学生前来应聘”

- 北京/大学生/前来/应聘
- 北京大学/生前/来/应聘

语法分析歧义

“咬死了猎人的狗”

语义分析歧义

“开刀的是他父亲”

NLP应用中的歧义

语音识别： wǔ yí

- 武夷
- 五姨

为什么自然语言处理如此困难？

口语、成语、俚语

“戴高帽子”

分词问题

新词产生

“吃鸡”

“打call”

基本常识和上下文知识

“爸爸背着我和弟弟”

实体词

“捉妖记”

自然语言处理的应用

- 信息提取
- 语音识别
- 文档摘要
- 聊天机器人
- 垃圾邮件过滤
- 文档归类
- 机器翻译

语言模型

Language Model

语言模型

词汇表:

V = {我, 人, 二, ...}

字符串:

V^*

我<S>

我是<S>

我是中<S>

我是中国<S>

Language Model

语言模型

学习一个概率分布：P,其中P满足

$$\sum_{x \in V^*} p(x) = 1, p(x) \geq 0 \ \forall x \epsilon V^*$$

p(我<S>)=10^{-12}
p(我是<S>)=10^{-8}
p(我是中<S>)=2×10^{-8}
p(我是中国<S>)=10^{-15}

Language Model

语言模型

- N: 训练集句子的数量
- 对于任意一个句子 $x_1 \dots x_n$, $c(x_1 \dots x_n)$表示在训练集中该句子出现的次数
- Naïve estimate:

$$\boldsymbol{p(x_1 \dots x_n) = \frac{c(x_1 \dots x_n)}{N}}$$

Markov Process

https://hackernoon.com/from-what-is-a-markov-model-to-here-is-how-markov-models-work-1ac5f4629b71

马科夫过程

- 给定一个离散随机变量序列 $X_1, X_2, \dots X_n$，每个随机变量可以取集合V中的任意值
- 建模
- 链式法则

$$P(X_1 = x_1, \dots, X_n = x_n)$$

$$P(X_1 = x_1, \dots, X_n = x_n) = P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \dots, X_{i-1} = x_{i-1})$$

Probability of "ALL elements appear"= P(1st element appear)*P(2nd|1st)*P(3rd|1st & 2nd)*P(4th| 1st & 2nd & 3rd)*...;

其中P(A|B) 为 A happens at condition of B happening = P (A and B)/P(B)

Markov Process

一阶马科夫过程

1st order Markov Process belongs to bi-gram language model; n-th order markov process belongs to (n+1)-gram language model

建假设当前时刻随机变量取值只与上一时刻有关:

$$P(X_i = x_i | X_1 = x_1, \dots, X_{i-1} = x_{i-1}) = P(X_i = x_i | X_{i-1} = x_{i-1})$$

$$P(X_1 = x_1, \dots, X_n = x_n) = P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})$$

Markov Process

二阶马科夫过程

建假设当前时刻随机变量取值只与之前两个时刻有关:

$$P(X_i = x_i | X_1 = x_1, \dots, X_{i-1} = x_{i-1}) = P(X_i = x_i | X_{i-1} = x_{i-1}, X_{i-2} = x_{i-2})$$

$$P(X_1 = x_1, \dots, X_n = x_n) = P(X_1 = x_1) P(X_2 = x_2, X_1 = x_1) \prod_{i=3}^{n} P(X_i = x_i | X_{i-1} = x_{i-1}, X_{i-2} = x_{i-2})$$

Markov Process

可变长序列

- 对于自然语言来说, 随机过程序列的长度n也是一个随机变量
- 定义: $X_n =< S >$, 其中<S>不在词汇表V中
- 我们可以使用马尔可夫过程来描述句子：

$$P(X_1 = x_1, \dots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i | X_{i-1} = x_{i-1}, X_{i-2} = x_{i-2})$$

为了简化起见, 我们引入两个额外的变量$x_0 = x_1 =*$

Trigram Language Model

三元语言模型 二阶markov process belongs to trigram language model

NOTE: Bigram=二元 model (inc'd: 一阶markov process)

- 词汇表: V
- 对每一个三元组u, v, w, 我们有一个参数$q(w|u, v)$. 这里$w \in V \cup \{<S>\}, u, v \in V \cup \{*\}$

how to estimate q for each segment w in sentence? -- use Max Likelihood Estimate (see notes afterwards)

- 对于任何一个句子$x_1, x_2, \ldots, x_n$, 其中$x_i \in V$ $i = 1 \ldots (n-1)$, $x_n = <S>$. 句子的概率由如下三元语言模型描述:

$$P(x_1, x_2 \ldots, x_n) = \prod_{i=1}^{n} q(x_i|x_{i-1}, x_{i-2})$$

仍然使用$x_0 = x_1 = *$

Trigram Language Model

三元语言模型

句子

我是中国人<s>

模型

$$p(\text{我是中国人} < S >) \\ = q(\text{我}| *,*)q(\text{是}|\text{我},*)q(\text{中}|\text{我},\text{是})q(\text{国}|\text{是},\text{中})q(\text{人}|\text{中},\text{国})q(< S > |\text{国},\text{人})$$

Maximum Likelihood Estimate

极大似然估计

- 参数

$$q(w|u,v)$$

- MLE

$$q(w|u,v) = \frac{Count(w,u,v)}{Count(u,v)}$$

$$\mathrm{q}(人|中,国) = \frac{Count(中,国,人)}{Count(中,国)}$$

- $N = |V|$, 词汇表词的数量, 三元模型参数数量为

$$N^3$$

Maximum Likelihood Estimate: An Example

极大似然估计：例子

- *我是中国人<S>
- *你在吗<s>
- *我今天在听课<s>

$q(\text{我}|*) = \frac{2}{3} = .67$

$q(\text{是}|\text{我}) = \frac{1}{2} = .5$

$q(< S > |\text{中国人}) = \frac{1}{1} = 1.0$

$q(\text{在}|\text{你}) = \frac{1}{1} = 1.0$

$q(\text{今天}|\text{我}) = \frac{1}{2} = .5$

$q(\text{听课}|\text{在}) = \frac{1}{2} = .5$

Perplexity

To evaluate model:

类比:
SVM use loss function to evaluate model;

语言模型评价：迷惑度

m segments(句子) in TEST set, if prob of appearing in your model for each segment is larger, it means model is more accurate -->
Then we define a comprehensive value called Perplexity, it can take prob of each segment into consideration(see equations below):
when prob increases, perplexity decrease--> so using it we can tell model is more accurate when perplexity is small

- 测试集: m个句

$$s_1, s_2, \dots, s_m$$

- 计算log概率

$$log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log(p(s_i))$$

- Perplexity

$$Perplexity = 2^{-l}$$

$$l = \frac{1}{M} \sum_{i=1}^{m} \log(p(s_i))$$, M是测试集总词量

Perplexity

语言模型评价

- $N = |V|$

$$q(w|u,v) = \frac{1}{N}$$

- Perplexity

$$l = log\frac{1}{N}$$

$$Perplexity = 2^{-l} = N$$

迷惑度越小，句子概率越大，语言模型越好

Perplexity

语言模型评价

《华尔街日报》训练集数据: 38 million词汇, 测试集规模为1.5 million词汇, 结果如下:

N-gram	Unigram	Bigram	Trigram
Perplexity	962	170	109

Data Sparse Problem

Using traditional Markov Process model, we have the following problem called Data Sparse Problem:
when one segment in test set did not appear in training set, all sentences related to that segment will have appearance probability=0.
(Because Markov process 是一个连乘model, 一个为0, then总体值(e.g. trigram model value=0)

稀疏性问题

How to fix?
-- see notes afterwards:
1. Linear Interpolation Smoothing
2. Laplace Smoothing

- 大规模数据统计方法与有限的训练~~预料~~之间必然产生数据稀疏问题

" 《记忆碎片》是一部完美倒置的电影，
非其故事结构如何的复杂，
而是它的叙述手法一味倒置得惨绝人寰， 闻所未闻。"

- IBM, Brown: 366M训练trigram, 在测试语料中, 有14.7%的trigram和2.2%的bigram在训练集中未出现

Method 1:

Linear Interpolation Smoothing

线性插值平滑

- 当高元n-gram模型没有足够数据时使用低元n-gram模型进行概率估计

hyper parameters

$$\hat{q}(w|u,v) = \lambda_3 q(w|u,v) + \lambda_2 q(w|u) + \lambda_1 q(w)$$

$$\sum_i \lambda_i = 1$$

- 从训练集中分出部分数据作为hold-out数据

hold-out data is for validation purpose

- $c'(w,u,v)$: trigram (w, u, v)在hold-out数据集中出现的次数

- 选择$\lambda_1, \lambda_2, \lambda_3$使得

All lambda>0

$$L(\lambda_1, \lambda_2, \lambda_3) = \sum_{w,u,v} c'(w,u,v) \log(q(w|\, u,v))$$

likelihood maximization in hold-out data set

最大化

Method 2:

Laplace Smoothing

拉普拉斯平滑

Equation for Bigram:

$$q(w|u) = \frac{c(w,u) + 1}{c(w) + L}$$

--> If w not appeared, c(w)=0, then q(w|u)=1/L

L是所有bigram的个数

Equation for Trigram,
q(w|u,v)=[c(w,u,v)+1] / [c(w)+L], where L is # trigram

隐马尔可夫模型

Hidden Markov process(model) --> This is used to solve Sequence-To-Sequence problem

Sequence-to-sequence Problem

序列到序列的问题

Sequence to Sequence Problem definition:
input = a sequence (e.g. a sentence xxx/xxx/xxxx)
output= another kind of sequence (e.g. 1 词性 sequence) (e.g. 2 实体类别sequence)

词性标注

输入 我是中国人

输出 我/n是/v中国人/n

命名实体识别

输入 大摩维持阿里巴巴增持评级 目标股价维持在210美元

输出 大摩/[Company]维持阿里巴巴/[Company]增持评级 目标股价维持在210美元

Sequence-to-sequence Problem

序列到序列的问题

- 输入 $x_1, x_2, \dots, x_n$

  输出 $y_1, y_2, \dots, y_n$

- 判别式模型(Discriminative Model) Condition Random Field
  - 从训练数据中学习$p(y_1, y_2, \dots, y_n | x_1, x_2, \dots, x_n)$
  - 测试集上, 输出$argmax_y p(y|x)$

- 生成式模型(Generative Model) HMM: Hidden Markov Model
  - 从训练数据中学习$p(y, x)$
  - 测试集上, 输出$argmax_y p(y|x) = argmax_y p(x|y) p(y)$

Hidden Markov Model

隐马尔可夫模型

- 输入句子: $x_1, x_2, \dots, x_n, x_i \in V$
- 输出标注序列: $y_1, y_2, \dots, y_{n+1}, y_i \in S, \mathrm{i} = 1 \dots \mathrm{n}, y_{n+1} = \langle S \rangle$
- ~~Trigram~~ HMM:

Bigram

transition probability

emission probability

$$p(x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_{n+1}) = \prod_{i=1}^{n+1} q(y_i \mid y_{i-1}) \prod_{i=1}^{n} e(x_i \mid y_i)$$

- 模型参数:

$$q(s|u) \quad s \in S \cup \{\langle S \rangle\}, u \in S \cup \{*\}$$

$$e(x|s) \; s \in S, x \in V$$

#para for q = N^2
para for e = M*N

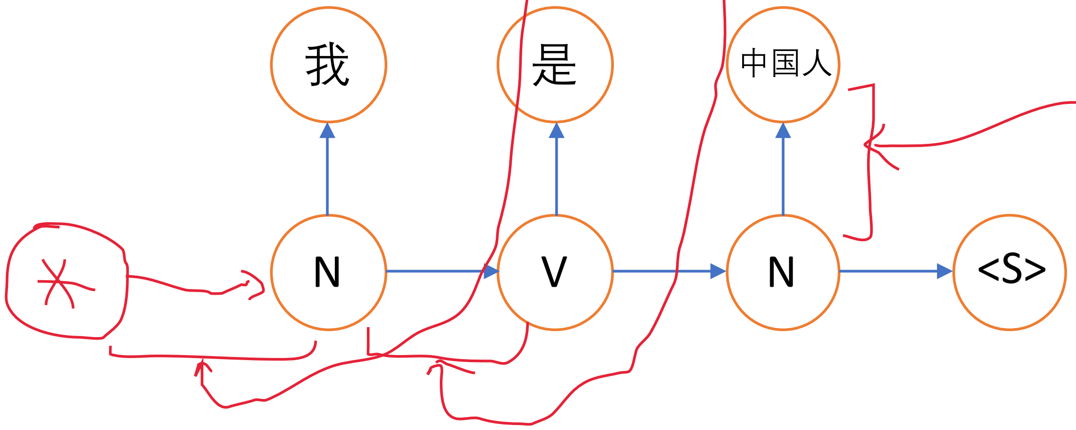
NOTE:
compare HMM and MM(Markov Model)::
e.g. Bigram HMM (i.e.一阶MM) 比 Bigram MM (i.e.一阶MM) 多了一个emission prob part

Hidden Markov Model

隐马尔可夫模型

- 输入句子: 我/是/中国人
- 输出标注序列:
  N/V/N/<S>
- Bigram HMM:

$$p(x_1,x_2,\dots,x_n,y_1,y_2,\dots,y_{n+1})$$
$$= q(N|*)q(V|N)q(N|V)q(<S>|N)e(我|N)e(是|V)e(中国人|N)$$



Parameter Estimate

参数估计

极大似然估计：

transition probability $$q(u|v)=\frac{c(u,v)}{c(v)}$$

emission probability $$e(x|s)=\frac{c(x,s)}{c(s)}$$

,

Data Sparse Problem

稀疏性问题

- 没有在训练集出现的词导致极大似然估计为零

  ”《记忆碎片》是一部完美倒置的电影,
  非其故事结构如何的复杂,
  而是它的叙述手法一味倒置得惨绝人寰，闻所未闻。”

- 解决办法：

  - ***Step 1:*** *将词汇分成两份，高频词=词频≥5, 低频词=所有其他词* "unknown"
  - *Step 2: 将所有低频词映射到一个固定的集合.*

Decoding Problem

解码问题

- 给定输入$x_1, x_2, \dots, x_n$, 找到

find y series to let p is max
(The final goal)

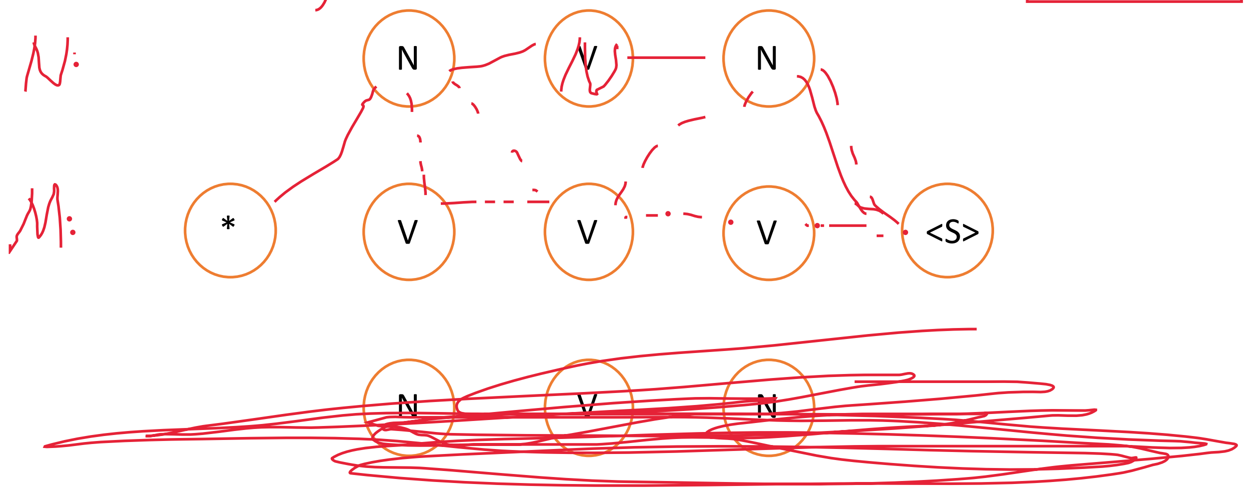
$$\text{argargmax}_{y_1,y_2,\dots,y_{n+1}} p(x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_{n+1})$$

- 暴力解：

1 $O(M^n)$ → 2

Dynamic
Programming:
(see page after)

$O(M^2)$



Viterbi 算法

Dynamic Programming

- n为句子长度
- $S_k, k = 0 \dots n$为k处所有可能标注的集合

y

$$S_0 = \{*\} \quad S_{n+1} = \{<s>\}$$

$$S_k = S \; k \in \{1 \dots n\}$$

- 定义:

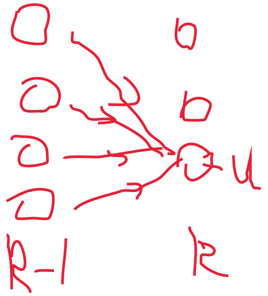
HMM

$$r(y_0, y_1, \dots, y_k) = \prod_{i=1}^{k} q(y_i \mid y_{i-1}) \prod_{i=1}^{k} e(x_i \mid y_i)$$

- 定义动态规划表:

see last slide

$$\pi(k,u) = max_{<y_0,y_1,\dots,y_k>: y_k = u} r(y_0, y_1, \dots, y_k)$$



i.e. pi = max r when yk=u; To explain: see graph, from state (k-1) to y=u@state k, among these several pathways, we need to find the max probability pathway, i.e. r(max)

- 定义Back Pointer:

After finding pi (i.e. r(max)), we use backpointer (i.e. dynamic Programming tech) to find which pathway was selected(i.e. which pt in state k-1 this pathway was from).

$$bp(k,u) = argmax_{w \in S_{k-1}} (\pi(k-1, w) q(u \mid w) e(x_k \mid u))$$

Viterbi 算法演示

- 输入: 句子$x_1, x_2, \ldots, x_n$, 参数$q(s|u), e(x|s)$
- 初始化: $\pi(0,*) = 1$
- 定义: $S_0 =*, S_k = S \;\; k = 1 \ldots n$
- 算法:
  - For $k = 1 \ldots n$
    - For $u \in S_k$

$$\pi(k,u) = max_{w \in S_{k-1}}(\pi(k-1,w)q(u|w)e(x_k|u)$$

$$bp(k,u) = argmax_{w \in S_{k-1}}(\pi(k-1,w)q(u|w)e(x_k|u)$$

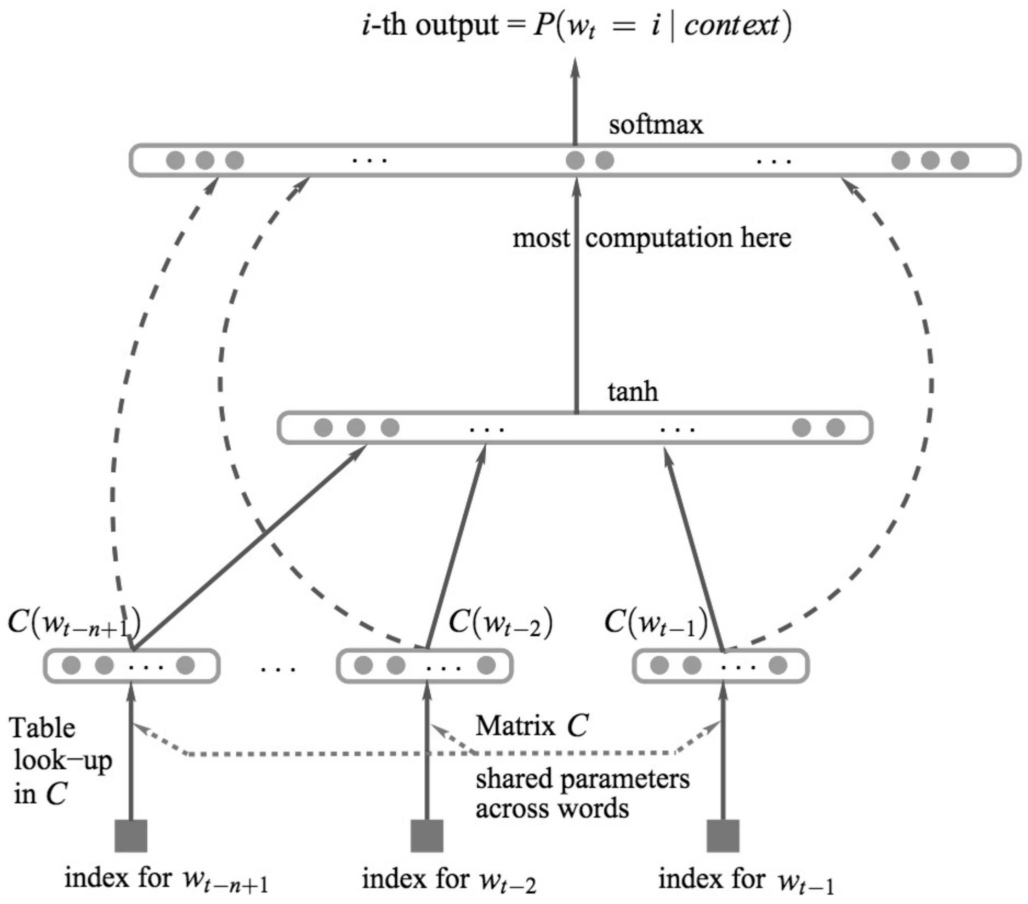
    - 令$y_n = argmax_u(\pi(n,u)q(< S > |u))$
    - For $k = (n-1) \ldots 1, y_k = bp(k+2, y_{k+1})$
    - 返回标注序列$y_1 \ldots y_n$

Viterbi 算法演示

深度学习

Neural Language Model

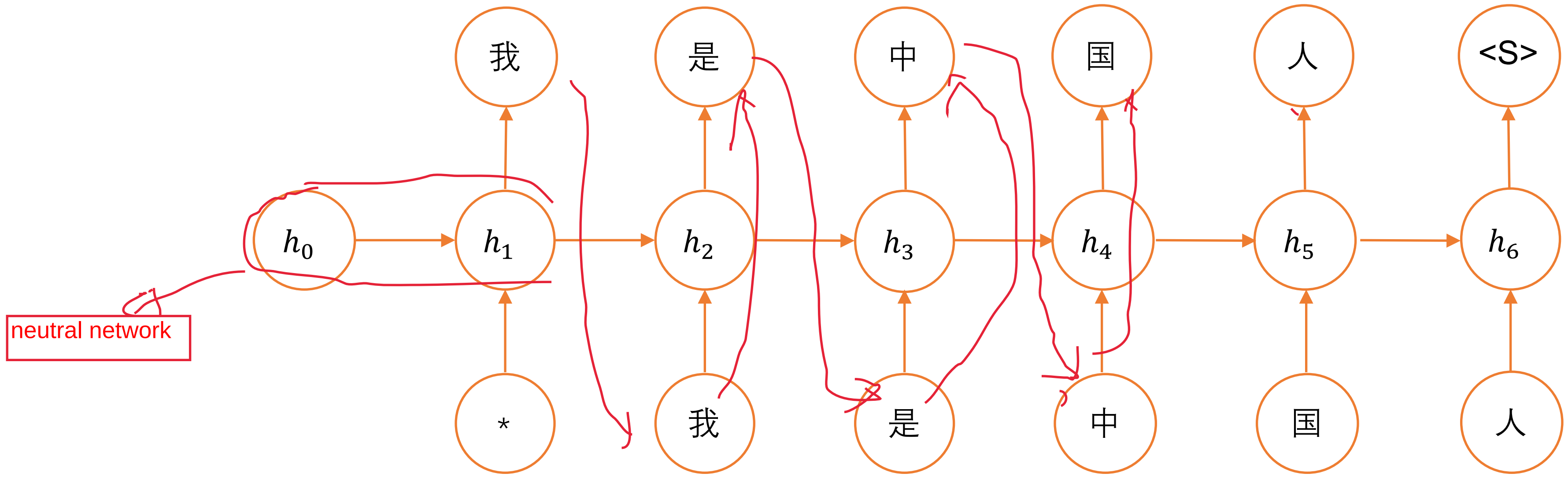
神经语言模型



A Neural Probabilistic Language Model, Yoshua Bengio, etc

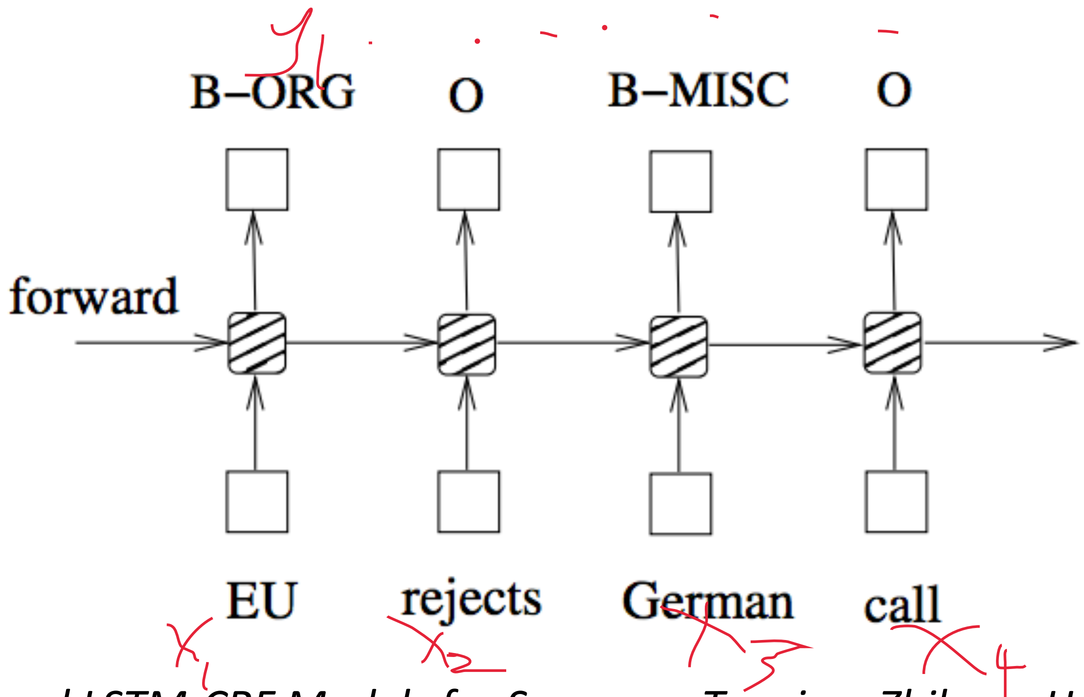
RNN

递归神经网络模型



POS

递归神经网络



Bidirectional LSTM-CRF Models for Sequence Tagging, Zhiheng Huang, etc

THANKS.