NLP期末总结 7-11

author：王若琪

7-9 语言模型

语言模型分为统计语言模型和神经语言模型

统计语言模型

基本概念

• 大规模语料库的出现为自然语言统计处理方法的实现提供了可能， 统计方法的成功使用推动了语料库语言学的发展。
• 基于大规模语料库和统计方法， 可以
o 发现语言使用的普遍规律
o 进行机器学习、 自动获取语言知识
o 对未知语言现象进行推测

语句 $s = w_1\ w_2\ \ldots\ w_m$ 的先验概率：

$$p(s) = p(w_1) \times p(w_2|w_1) \times p(w_3|w_1w_2) \times \ldots \times p(w_m|w_1 \ldots w_{m\text{-}1})$$

$$= \prod_{i=1}^{m} p(w_i \mid w_1 \cdots w_{i-1})$$

当 i=1 时， $p(w_1|w_0) = p(w_1)$。

语言模型！！

wi的概率由 w1, ..., wi-1 决定，由特定的一组w1, ...,wi-1 构成的一个序列， 称为 wi 的历史(history)。

问题： 随着历史基元数量的增加，不同的“历史”(路径)按指数级增长。

n 元文法(n-gram)模型

❑ 问题解决方法

设法减少历史基元的个数，将 $w_1\ w_2\ \ldots\ w_{i\text{-}1}$ 映射到等价类 $S(w_1\ w_2\ \ldots\ w_{i\text{-}1})$，使等价类的数目远远小于原来不同历史基元的数目。则有：

$$p(w_i \mid w_1, \ldots, w_{i\text{-}1}) = p(w_i \mid S(w_1, \ldots, w_{i\text{-}1}))$$

o 将两个历史映射到同一个等价类， 当且仅当这两个历史中的最近 n-1 个基元相同，即：

H_1: $w_1\, w_2 \ldots \ldots\ w_{i-n+1}\, w_{i-n+2} \ldots w_{i-1}\, w_i \ldots\ldots$

(n-1)

H_2: $v_1\, v_2 \ldots \ldots\ v_{k-n+1}\, v_{k-n+2} \ldots v_{k-1}\, v_k \ldots\ldots$

$$S(w_1, w_2, \cdots, w_i) = S(v_1, v_2 \cdots, v_k)$$
$$iff\ \ H_1:(w_{i-n+1}, \cdots, w_i) = H_2:(v_{k-n+1}, \cdots, v_k)$$

通常地，

- 当 n=1 时，即出现在第 i 位上的基元 w_i 独立于历史。一元文法也被写为 uni-gram 或 monogram;
- 当 n=2 时, 2-gram (bi-gram) 被称为1阶马尔可夫链;
- 当 n=3 时, 3-gram(tri-gram)被称为2阶马尔可夫链，依次类推。

为了保证条件概率在 i=1 时有意义，同时为了保证句子内所有字符串的概率和为 1，即 $\sum_s p(s) = 1$，可以在句子首尾两端增加两个标志: <BOS> $w_1\, w_2 \ldots w_m$ <EOS>。不失一般性，对于n>2 的 n-gram，$p(s)$ 可以分解为：

$$p(s) = \prod_{i=1}^{m+1} p(w_i \mid w_{i-n+1}^{i-1})$$

其中，w_i^j 表示词序列 $w_i \ldots w_j$，w_{i-n+1} 从 w_0 开始，w_0 为 <BOS>，w_{m+1} 为 <EOS>。

<BOS> John read a book <EOS>

基于2元文法的概率为：

$$p(\text{John read a book}) = p(\text{John}|\text{<BOS>}) \times p(\text{read}|\text{John}) \times p(\text{a}|\text{read}) \times p(\text{book}|\text{a}) \times p(\text{<EOS>}|\text{book})$$

应用：音字转换问题、汉语分词问题

如果汉字的总数为：N

- 一元语法：1）样本空间为 N
- 2元语法：1）样本空间为 N^2
  2）效果比一元语法明显提高
- 估计对汉字而言四元语法效果会好一些
- 智能狂拼、微软拼音输入法基于 n-gram.

参数估计

两个重要概念：
o 训练语料(training data)，用于建立模型确定模型参数的已知语料
o 最大似然估计(MLE)，用相对频率计算概率的方法

参数问题：

数据匮乏(稀疏) (Sparse Data) 引起零概率问题， 如何解决?

数据平滑(data smoothing) 。

数据平滑

❑ 数据平滑的基本思想：
调整最大似然估计的概率值, 使零概率增值， 使非零概率下调， 消除零概率， 改进模型的整体正确率
❑ 基本目标：
测试样本的语言模型困惑度(Perplexity)越小越好

❑ 基本约束： $\sum_{w_i} p(w_i | w_1, w_2, \cdots, w_{i-1}) = 1$

困惑度（perplexity）的基本思想是：**给测试集的句子赋予较高概率值的语言模型较好,当语言模型训练完之后，测试集中的句子都是正常的句子，那么训练好的模型就是在测试集上的概率越高越好，**公式如下：

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N}}$$

由公式可知，**句子概率越大，语言模型越好，迷惑度越小。**

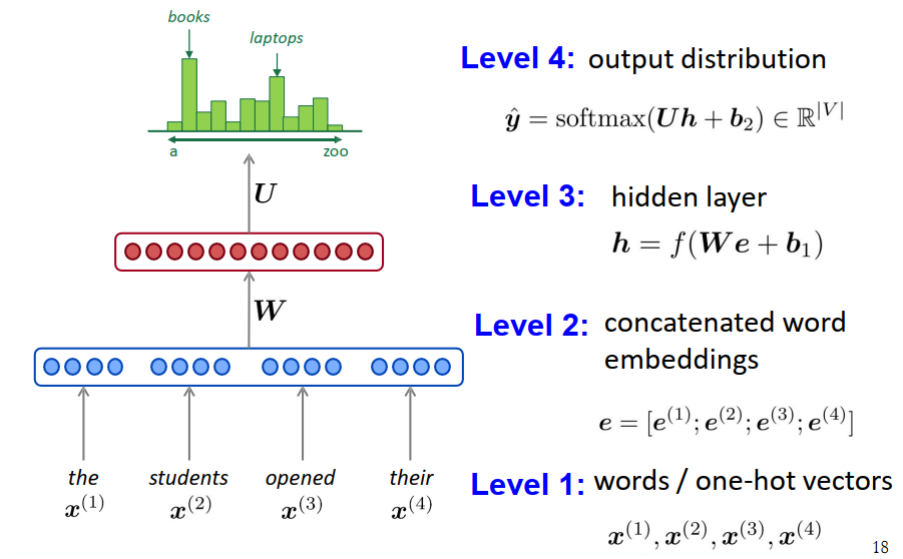
数据平滑方法：

1）加一法：每一种情况出现的次数加1

2）减值法/折扣法(Discounting)
基本思想： 修改训练样本中事件的实际计数， 使样本中(实际出现的)不同事件的概率之和小于1，剩余的概率量分配给未见概率 。绝对减值法：基本思想：从每个计数r 中减去同样的量，剩余的概率量由未见事件均分。

神经语言模型

Window-based neural model

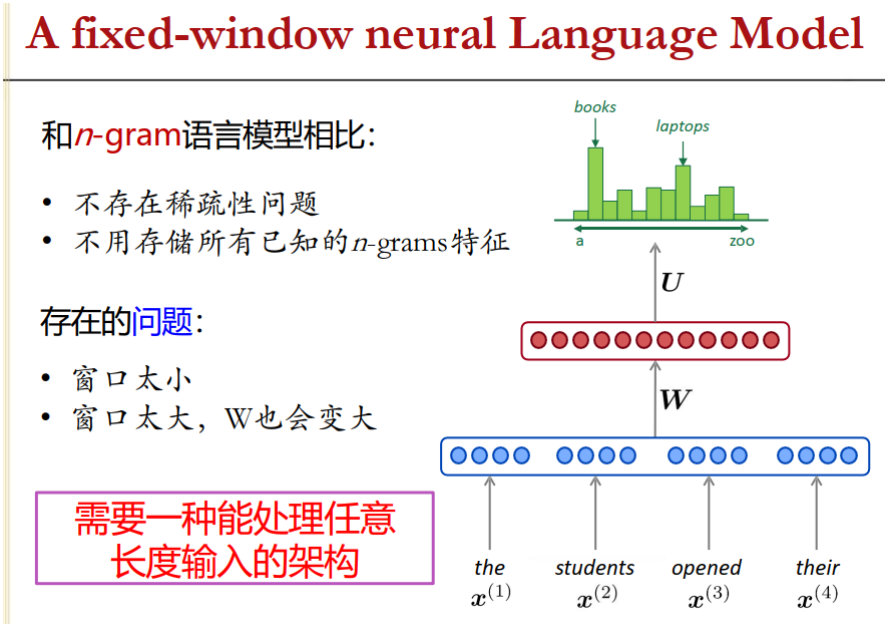


典型应用： Word2vec

百度百科：Word2vec，是一群用来产生词向量的相关模型。这些模型为浅而双层的神经网络，用来训练以重新建构语言学之词文本。训练完成之后，word2vec模型可用来映射每个词到一个向量，可用来表示词对词之间的关系，该向量为神经网络之隐藏层。

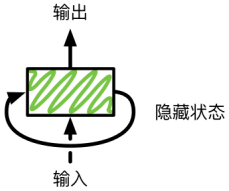
- 词向量给NLP问题提供一个全新的视角
- Word2vec通过一种无监督的方式获取词向量

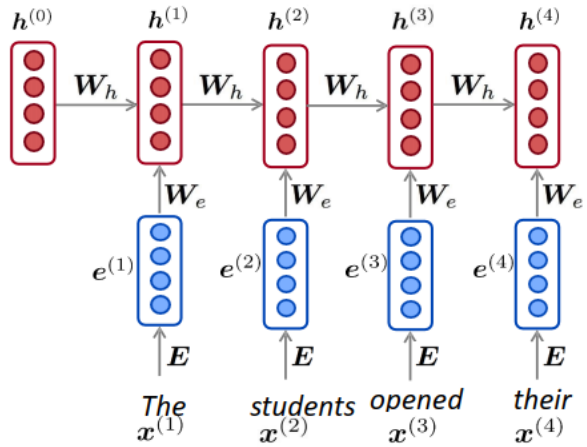
和n-gram对比分析：



循环神经网络

循环神经网络主要用于处理（变长）序列数据，不同的神经单元共享相同的参数





Level 4: output distribution

$$\hat{\boldsymbol{y}}^{(t)} = \text{softmax}\left(\boldsymbol{U}\boldsymbol{h}^{(t)} + \boldsymbol{b}_2\right) \in \mathbb{R}^{|V|}$$

Level 3: hidden states

$$\boldsymbol{h}^{(t)} = \sigma\left(\boldsymbol{W}_h\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_e\boldsymbol{e}^{(t)} + \boldsymbol{b}_1\right)$$

$\boldsymbol{h}^{(0)}$ is the initial hidden state

Level 2: word embeddings

$$\boldsymbol{e}^{(t)} = \boldsymbol{E}\boldsymbol{x}^{(t)}$$

Level 1: words / one-hot vectors

$$\boldsymbol{x}^{(t)} \in \mathbb{R}^{|V|}$$

RNN 的优点:

- 能够处理任意长度的输入；
- t时刻可以访问之前任意时刻的信息；
- 对于较长的输入，模型的大小不变；
- 权重共享

RNN 的不足:

- 循环计算过程较慢；
- 实际运用中，很难访问距当前时刻较远的信息；

RNN语言模型的训练

RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；
- Step 3: 用交叉熵(cross-entropy)计算每一时刻的损失(loss)
  - 方法：使用时刻t的预测分布 $\hat{\boldsymbol{y}}^{(t)}$ 和下一个真实词 $\boldsymbol{x}^{(t+1)}$ 的表示 $\boldsymbol{y}^{(t)}$：

$$J^{(t)}(\theta) = CE(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) = -\sum_{w \in V} \boldsymbol{y}_w^{(t)} \log \hat{\boldsymbol{y}}_w^{(t)} = -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

- Step 4: 计算所有文本的平均损失：

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta) = \frac{1}{T}\sum_{t=1}^{T} -\log \hat{\boldsymbol{y}}_{\boldsymbol{x}_{t+1}}^{(t)}$$

52

- 通常，使用随机梯度下降(Stochastic Gradient Descent) 计算一小部分随机数据(mini-batch)的损失；
- 根据得到的损失，计算梯度并更新参数；

对于重复出现的变量，它的梯度是每次出现的梯度之和。（多变量链式法则）

RNN 语言模型用于文本生成

和n-gram 语言模型一样， RNN语言模型可以通过循环采样生成文本

语言模型的评估

评估语言模型的标准方法是计算困惑度(perplexity)

注意

• 语言模型: 预测下一个词的系统
• RNN: 一种神经网络:
• 输入是任意长度的文本;
• 共享权重;
• 每一步都可以产生输出;
• Recurrent Neural Network ≠ Language Model
• RNN可以用来构建语言模型，但RNN还有更多用途;

• RNN可以用于做序列标注，如词性标注

• RNN可以用于文本分类 （基本方法：使用最后一个隐状态 。更好方法： 取所有隐状态的最大值或者平均值 ）

• RNN还可以用于问答系统和机器翻译等任务

• RNN还可以用于语音识别

• 常见的RNN模型
o LSTM
o GRU
o Bidirectional
o Multi-layer

最近火的技术

1. Word Embeddings

How do we have usable meaning in a computer?

普通方法：WordNet，一个包含同义词集和连词(“is a”关系)列表的同义词词典。

WordNet 的问题：

- 作为一种很好的资源，但是缺少了细微的差别
- 缺少词语的新含义，不可能保持最新!
- 主观的
- 需要人类劳动来创造和适应
- 无法计算准确的单词相似度

one hot 编码：无法表达单词间的相似性等联系。

可以尝试依赖WordNet的同义词列表来获得相似性吗?

•但众所周知，失败之处很严重:不完整，等等。

相反:学习在向量本身中编码相似性

word vectors ： 一种分布式表示。“分布式表示”是指两种表示类型(如概念和神经元)之间的多对多关系。每个概念都由许多神经元来表示，每个神经元都参与了许多概念的表达。

Word2vec (Mikolov et al. 2013)是一个学习词向量的框架。

2. Contextual Embeddings

最初，我们基本上只有一种词的表达方式:

•我们开始学过的词向量 Word2vec, GloVe, fastText

这有两个问题:

•对于单词类型总是相同的表示，而不管单词标记在什么上下文中出现，我们可能需要非常细粒度的词义消歧

•我们对一个词只有一种表示，但词有不同的方面，包括语义、句法行为和语域/内涵

我们一直都有解决这个问题的办法吗?

•在NLM中，我们立即通过LSTM层插入词向量(可能只在语料库上训练)

•这些LSTM层被训练来预测下一个单词

•但是这些语言模型会在每个位置产生上下文特定的单词表示!

ELMo:来自语言模型的嵌入

•使用长上下文学习单词标记向量，而不是上下文窗口(这里，整个句子可能更长)

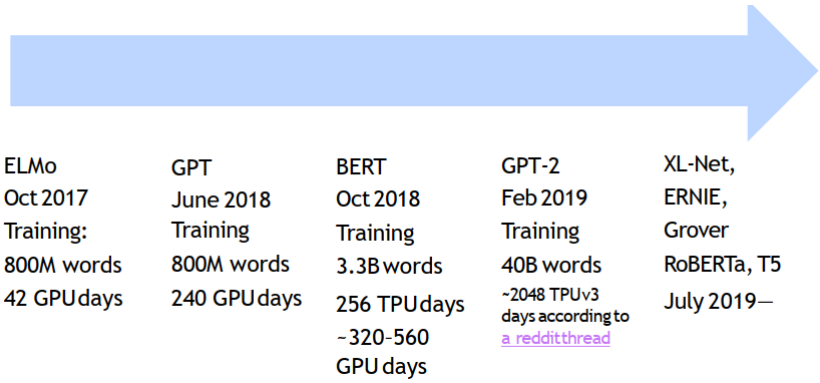
•学习一个深度的Bi-NLM，并在预测中使用它的所有层

两个biLSTM NLM层有不同的用法/含义

底层更适合底层语法，等等。词性标注、句法依赖、NER

更高的层次更适合更高层次的语义。情感、语义角色标注、问题回答、SNLI

这似乎很有趣，但更有趣的是看看在两层以上的网络中它是如何运作的



3. Transformer

RNN and Attention

在Encoder-Decoder结构中， Encoder把所有的输入序列都编码成一个统一的语义特征c再解码，因此，c中必须包含原始序列中的所有信息，句子的长度就成了限制模型性能的瓶颈。如机器翻译问题，当要翻译的句子较长时，一个c可能存不下那么多信息，就会造成翻译精度的下降。 Attention机制通过在每个时间输入不同的c来解决这个问题 。 每一个c会自动去选取与当前所要输出的y最合适的上下文信息。具体来说，我们用aij衡量Encoder中第j阶段的hj和解码时第i阶段的相关性，最终Decoder中第i阶段的输入的上下文信息ci 就来自于所有hj 对aij的加权和。

MultiHead机制 ，每个head可以学习不同的知识

每个区块有两个"子层"， Multihead attention 、2层前馈NNet(带ReLU)

4. BERT

BERT (Bidirectional Encoder Representations from Transformers):
Pre-training of Deep Bidirectional Transformers for LanguageUnderstanding, which is then fine-tuned for a task
Want: truly bidirectional information flow without leakage in a deep model

---

句法分析

句法分析是自然语言处理中的基础性工作，它分析句子的**句法结构**（主谓宾结构）和**词汇间的依存关系**（并列，从属等）；

句法分析可以为语义分析、情感倾向、观点抽取等NLP应用场景打下坚实的基础。

句法分析不是自然语言处理任务的最终目标，但它往往是实现最终目标的一个关键环节！

- ❑ 任务类型：
  - ❖ 短语结构分析(Phrase parsing)，也叫成分结构分析
    - 分析句子的主谓宾定状补的句法结构
      - 完全句法分析：以获取整个句子的句法结构为目的；
      - 局部句法分析：以获得局部成分为目的；
  - ❖ 依存句法分析(Dependency parsing)
    - 通过分析语言单位内成分之间的依存关系揭示其句法结构，如并列、从属、比较、递进等。

短语结构分析（上下文无关文法，线图分析法，概率上下文无关算法）：

表示方法：括号嵌套表示、树状表示（叶节点-词语，内部节点-短语类型）、

目标： 实现高正确率、高鲁棒性(robustness)、 高速度的自动句法分析过程；

困难： 自然语言中存在大量的复杂的结构歧义(structural ambiguity)；

❑ 基本方法和开源的句法分析器:

- 基于CFG规则的分析方法
  - CFG: Context-Free Grammar (上下文无关文法)
  - 代表: 线图分析法(chart parsing)

- 基于 PCFG 的分析方法
  - PCFG: Probabilistic Context-Free Grammar (概率上下文无关文法)

上下文无关文法

CFG由一系列规则组成，每条规则给出了语言中的某些符号可以被组织或排列在一起的方式 。

上述过程用树表示非常方便， terminals是叶子节点，而non-terminals是非叶子节点。

基于上下文无关文法（CFG）的句法分析是指基于预定义的语法，为输入语句生成恰当的句法树，要求该树：

✓ 符合给定语法；
✓ 叶子节点包含所有的词；

符合这样条件的树通常有很多！

线图分析法

三种策略
□ 自底向上 (Bottom-up)
□ 从上到下 (Top-down)
□ 从上到下和从下到上结合

概率上下文无关法

对于可能产生多种语法分析结果的问题，我们该如何应对呢?
□ 引入概率上下文无关文法（PCFG， Probabilistic context-free grammar）： 给每棵树计算一个概率

思考题

1. 什么是CFG/PCFG?
2. 简述依存句法与CFG/PCFG的区别?
3. 何为依存句法树的投射性?