神经概率语言模型
Neural Probabilistic Language Model

王小捷
智能科学与技术中心
北京邮电大学

大纲

- 引言
- 词表示
- 神经概率语言模型 (NPLM)
- 总结

引言

- N-gram语言模型中： $P(w_1^T) = \prod_{t=1}^{T} P(w_t \mid w_{t-n+1}^{t-1})$
- 两个问题
- 问题1
  - 数据稀疏问题
    - 理论上，模型阶数越高越好， 但由于数据稀疏，N-gram模型中n达到一定值后，n越大性能反而越差(<6)，有没有可以算高阶的模型？
    - 同样由于数据稀疏问题，平滑很重要，有没有不需要平滑就可以直接用的？

- LM的一个缺陷：建模时没有考虑词的相似性
  - XX cat is XX 的训练样本对计算p(is/dog)无贡献
    - 但是，由于cat和dog的相似性，p(is/dog)很可能与P(is/cat)相似
  - 改进：基于类的LM：$p(v/w) \approx p(v/c)p(c/w)$
    - 此时：如果w1和w2同属于类c，则$p(v/w1) \approx p(v/w2)$

- 更一般地，希望有如下性质：
  - 如果x,y是两个很相似的词，则f(x)接近f(y)。
  - 如果：f(x)是连续函数，x是连续变量，则这是一个常见的性质
    - f(x)是连续函数： f(x)是概率，可以构建成连续函数
    - x是连续变量：x是词，词的连续表示？

大纲

- 引言
- 词表示
- 神经概率语言模型 (NPLM)
- 总结

- 词表示
  - 符号表示
    - cat、dog、table…
  - 数值表示：一个词用一个多维向量表示
    - One-hot：向量维数=词表大小
      - (1,0,0,….), (0,1,0,……)…
      - 依靠词自身来表示词，和符号表示没有本质差异

- 词表示
  - 符号表示
    - cat、dog、table…
  - 数值表示：一个词用一个多维向量表示
    - One-hot：向量维数=词表大小
      - (1,0,0,….), (0,1,0,……)…
    - Distributional ： 向量维数=词表大小
      - 布尔式、频率式 ➔

- Distributional 词表示
  - 语料C，词表W={w1,w2,…WN}，窗口k
  - w∈W的词表示：用C中所有w的(窗口为k的)上下文中的词来构造w的表示：
    - 布尔式：(I_{w1}, I_{w2}, …… I_{wN})
    - 频率式：(f_{w1}, f_{w2}, …… f_{wN})
  - 例：C={I am here, I am fine.}， W={I, am, here, fine}, k=1
    - I
      - 布尔式：(0,1,0,0)
      - 频率式：(0,2,0,0)
    - am
      - 布尔式：(1,0,1,1)
      - 频率式：(2,0,1,1)
    - ……

One-hot

I	(1,0,0,0)
am	(0,1,0,0)
here	(0,0,1,0)
fine	(0,0,0,1)

- 无论是one-hot 还是distributional
  - 离散表示
  - 维度高，词表维度，每一个维度表示一个词，
    - one-hot ： 该词本身
    - distributional ： 该词上下文中的词
- 低维连续表示
  - 基于one-hot 或distributional 的降维表示
    - SVD分解等，后续LDA模型时再提
  - 词向量(distributed)

- 词表示
  - 符号表示
    - cat、dog、table…
  - 分布表示：一个词用一个多维向量表示
    - One-hot：向量维数=词表大小
      - (1,0,0,….), (0,1,0,……)…
    - Distributional ： 向量维数=词表大小
      - 布尔式、频率式
    - Distributed ： 向量维数=指定大小
      - 连续值：从PNLM等模型训练获得

- Distributed 词表示
  - 词表W={w1,w2,…WN}
  - m维词表示
    - c(W1)=(x_{11}, x_{12},…x_{1m})
    - c(W2)= (x_{21}, x_{22},…x_{2m})
    - …
    - c(WN)= (x_{N1}, x_{N2},…x_{Nm})
  - 表示容量没问题：即使x_{ij}为{0,1}，m也可以远远小于N
  - 实际上x_{ij}为实数

- Distributed 词表示
  - 设C(w)=(x_1, x_2,…x_m)是词w的Distributed表示
  - 则每个词是m维空间中的一个点

  - 希望：相似的词具有距离相近的词向量
    - 例如：牛、马、网：对于机器没有距离差别
    - 如果：
    - c(牛)=(0.5,0.4,0.20)
    - c(马)=(0.5,0.4,0.09)
    - c(网)=(0.2,0.7,0.18)
    - 则很容易计算出不同词之间的相近程度不同

- 词向量用于语言模型的好处：
  - 将语言模型构建为关于连续词表示的连续函数，则可以在语言模型中充分利用词的相似性获得更好的语言模型，即相似的词(元组)得到相似的概率：
  - 例如：以bigram 为例：
  - 1、将p(w|x)建模为连续函数
  - 2、如果x、y两个词较接近，则p(w|x)与p(w|y)较接近。这是在x、y是符号表示时是难以建模到的！

大纲

- 引言
- 词表示
- 神经概率语言模型 (NPLM)
- 总结

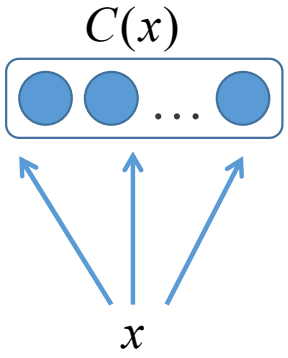
- 神经概率语言模型(NPLM)
  - Bengio 2001，2003模型
    - A Neural Probabilistic Language Model，JMLR，2003
- Bengio 2003 模型贡献
  - 得到基于分布式词表示的语言模型
    - 高阶(6), 无需平滑
  - 得到分布式词表示
  - 实验表明比基于符号的语言模型更好
    - PP值评测

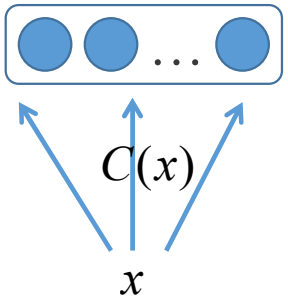
- 符号约定
  - x表变量，w_i表具体词
  - v’表转置，A_j表A的第j行

- NPLM：两部分
  - 词表示
    - 牛➔ $C(牛)=(x_{11},x_{12},\ldots,x_{1m})$
    - 马➔ $C(马)=(x_{21},x_{22},\ldots,x_{2m})$
    - 网➔ $C(网)=(x_{31},x_{32},\ldots,x_{3m})$
    - …
  - 语言模型
    - $p(w|w_1, w_2,\ldots w_n)$ ➔ $p(c(w)|c(w1),c(w2),\ldots c(wn))$

- 1.词表示：
  - 目标：词表V 中的词($w_1,\ldots w_{|V|}$) 得到其m维向量表示
  - 实现方式：词表映射
    - 查表映射C：将任意词映射为一个m维向量
    - 或者说是一个2层的神经网络

$$x \xrightarrow{C} (x_1,x_2,..x_m)$$





- 2.语言模型 $p(c(x_t)|c(x_{t-1}),\ldots c(x_{t-(n-1)}))$
- 建模连续函数:
  - $c(x_{t-1}), \ldots c(x_{t-(n-1)}) \rightarrow p(c(x_t)|c(x_{t-1}),\ldots c(x_{t-(n-1)}))$
- 神经网络模型
  - 训练数据
  - 损失函数

■神经网络模型

■模型目标：训练一个映射 g 建模n元语言模型：

$$g(C(x_t), C(x_{t-1}), .., C(x_{t-n+1}); \omega) = P(C(x_t) \mid C(x_{t-1}), .., C(x_{t-n+1}))$$

■其中 ω 为神经网络参数

■训练的目标是使得该n元模型对于测试词序列 $x_1, x_2, \ldots x_T$（x_i均为词表V中的词）具有最小PP值。即极小化：

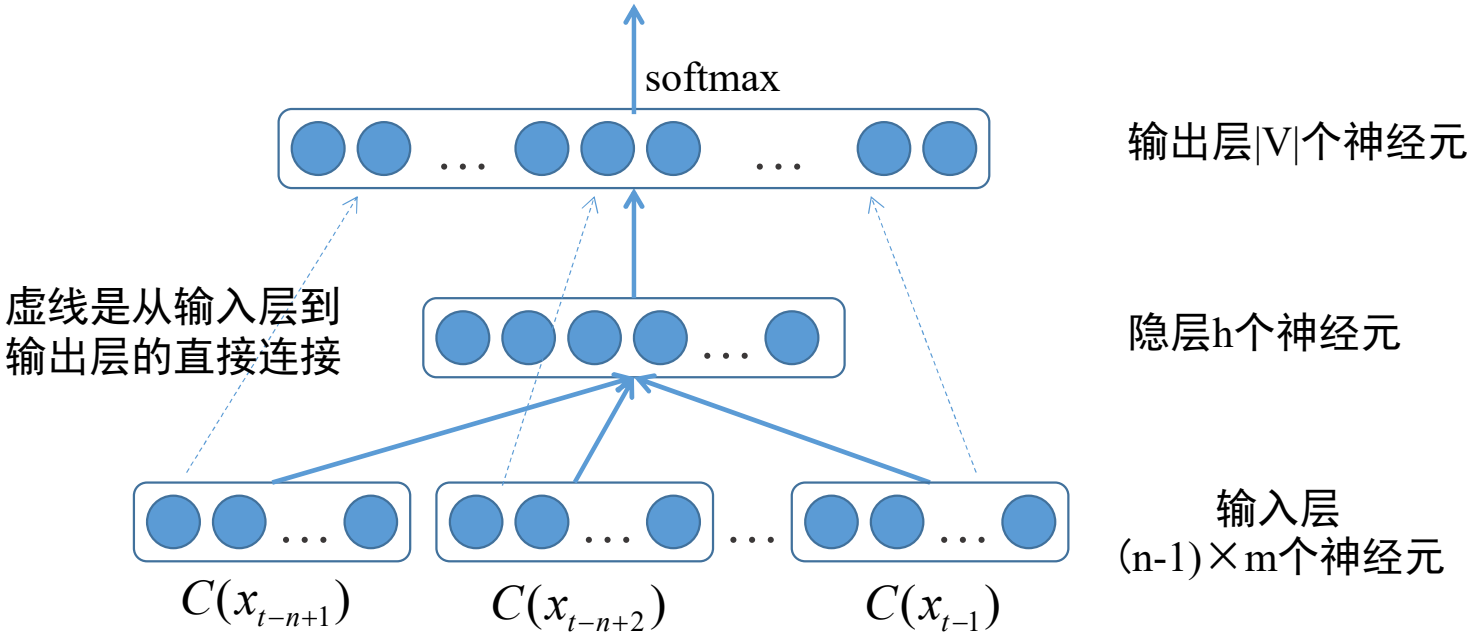
$$PP(C(x_1), \ldots, C(x_T)) = P(C(x_1), \ldots, C(x_T))^{-\frac{1}{T}} = \left(\prod_{t=1}^{T} P(C(x_t) \mid C(x_{t-1}), \ldots, C(x_{t-n+1}))\right)^{-\frac{1}{T}}$$

■即极大化：

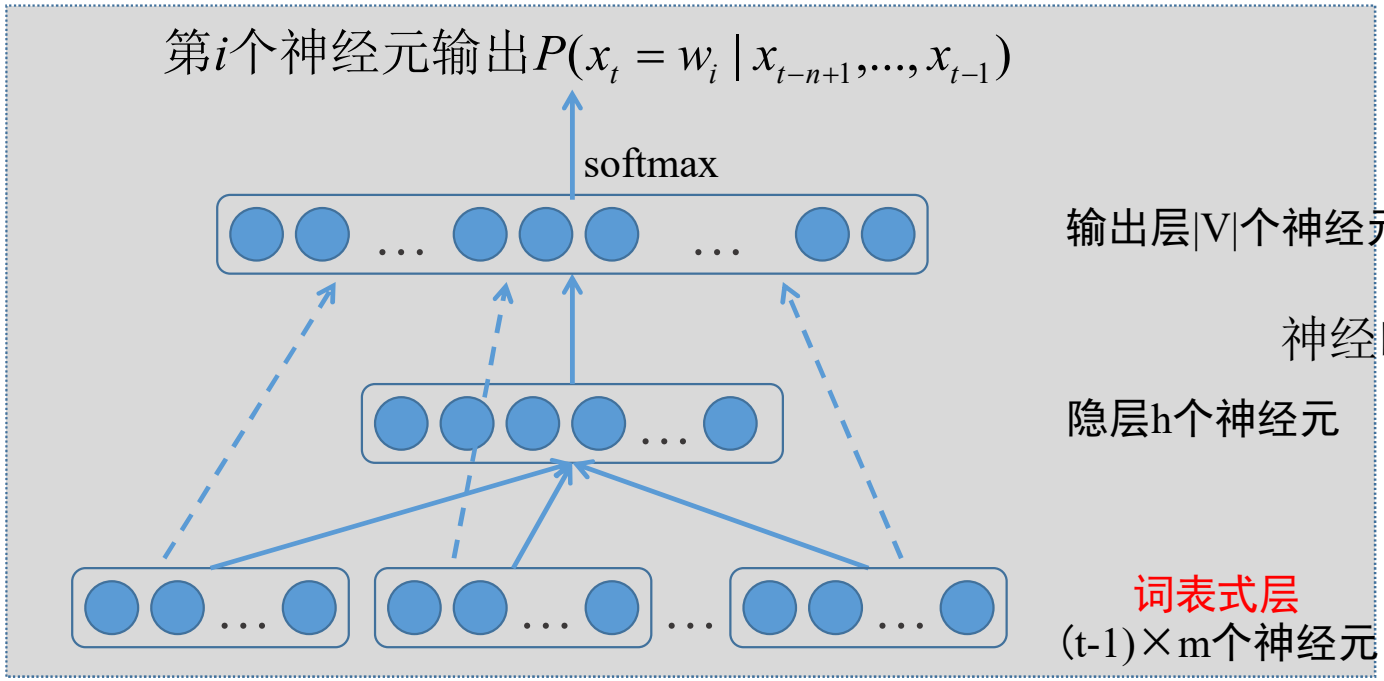
$$L = \frac{1}{T}\sum_{t=1}^{T} \log P(C(x_t) \mid C(x_{t-1}), \ldots, C(x_{t-n+1})) = \frac{1}{T}\sum_{t=1}^{T} \log g(C(x_t), C(x_{t-1}), \ldots, C(x_{t-n+1}); \omega)$$

■神经网络模型结构

第i个神经元输出为 $P(C(x_t)=C(w_i)\,|\,C(x_{t-n+1}),..,C(x_{t-1}))$



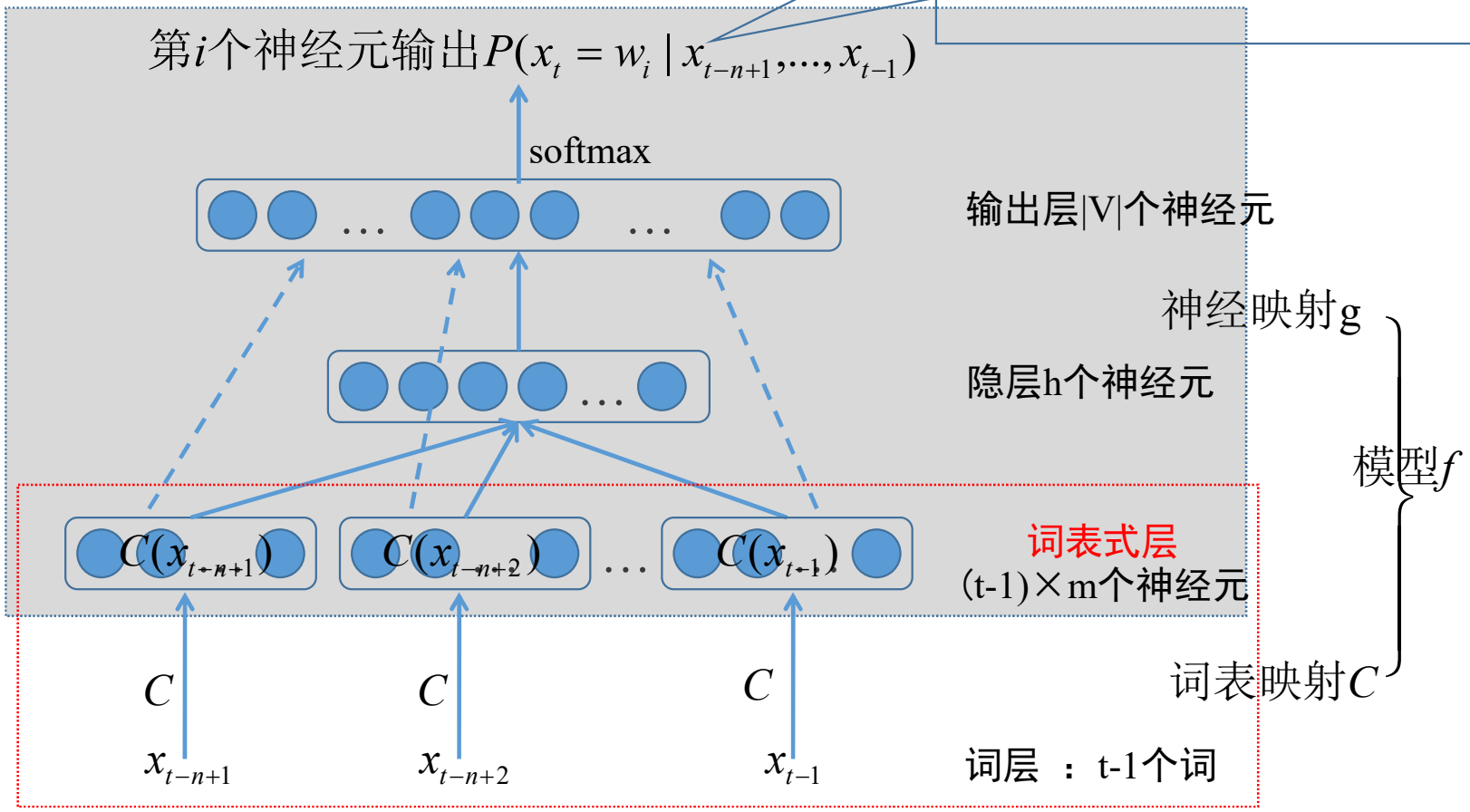
■合并词表映射与神经网络模型-1



■合并词表映射与神经网络模型-2

给定C时有：

$$P(C(x_t)\,|\,C(x_{t-n+1}),..,C(x_{t-1})) = P(x_t\,|\,x_{t-n+1},..,x_{t-1})$$



■整体模型的训练目标中把C也纳入，则为极大化：

$$L = \frac{1}{T}\sum_{t=1}^{T}\log g(C(x_t), C(x_{t-1}), .., C(x_{t-n+1}); \omega)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\log f(x_t, x_{t-1}, .., x_{t-n+1}; C, \omega)$$

■加上正则化项，则为：

$$L = \frac{1}{T}\sum_{t=1}^{T}\log f(x_t, x_{t-1}, .., x_{t-n+1}; C, \omega) + R(C, \omega)$$

- 模型参数
  - 各层
    - 词层n-1个节点，n元语法的n-1个历史词
    - 词表示层(n-1)×m个节点，每个词用m维向量表示
    - 隐层h个节点，阈值为d，h维
    - 输出层|V|个节点，阈值为b，|V|维
  - 层间
    - 词层到表示层：每一个词都有表示，C=|V|×m矩阵
    - 表示层到隐层：权重H，(n-1)m×h矩阵
    - 表示层到输出层：权重W，(n-1)m×|V|矩阵
    - 隐层到输出层：权值U，h×|V|矩阵
- 总参数个数
  - |V|* (1+mn+h)+h*(1+ (n-1)m)

- 模型计算：对每一个输入的n元串
  - 前向计算
    - 隐层输入为：y=b+W*C(x)+Utanh(d+HC(x))
    - 隐层输出为：

$$P(x_t \mid x_{t-1},\ldots,w_{t-n+1}) = \frac{e^{y_{x_t}}}{\sum_i e^{y_{x_i}}}$$

    - 其中：C(x)是词x的向量表示
    - 参数集：$\theta = (b, W, C, U, d, H)$
  - 反向随机梯度下降

$$\theta \leftarrow \theta + \varepsilon \frac{\partial \log P(x_t \mid x_{t-1},\ldots,w_{t-n+1})}{\partial \theta}$$

    - ε 为学习率
    - 不在输入窗口中的词向量值不需要调整

- 混合模型
  - 神经网络模型+插值trigram
- 跨层连接
- 算法的并行执行
- …

■实验语料

	Brown语料	AP新闻
训练语料规模	1,181,041词的前800000词	13,994,528词
发展语料规模(模型选择、权重衰减、early stopping)	随后的200,000词	963,138词
测试语料规模	其余181,041词	963,071词
语料实际含的不同词	47,578(含标点、大小写不同、分割段落与文本的标记符)	148,721词
使用的词，即|V|	16,383去除频率小于等于3的	17,964词(进行一些合并)
学习率	初始$\varepsilon_0=10^{-3}$，之后衰减，按$\varepsilon_t=\varepsilon_0/(1+rt)$	
权重衰减惩罚	10^{-4}	10^{-5}
Early stopping	采用	没有
收敛	10-20epochs后	5epochs后

- 对比模型
  - Benchmark n-gram models
    - interpolated or smoothed trigram model (Jelinek and Mercer, 1980)
  - state-of-the-art n-gram models
    - back-off n-gram models with theModified Kneser-Ney algorithm (Kneser and Ney, 1995, Chen and Goodman., 1999)
    - class-based n-gram models (Brown et al., 1992, Ney and Kneser, 1993, Niesler et al., 1998).

■Brown语料结果

模型阶数　词类数　隐单元数　词表示维数　输入到输出的连接　是否与插值trigram混合（权值均为0.5）　PP

	n	c	h	m	direct	mix	train.	valid.	test.
MLP1	5		50	60	yes	no	182	284	268
MLP2	5		50	60	yes	yes		275	257
MLP3	5		0	60	yes	no	201	327	310
MLP4	5		0	60	yes	yes		286	272
MLP5	5		50	30	yes	no	209	296	279
MLP6	5		50	30	yes	yes		273	259
MLP7	3		50	30	yes	no	210	309	293
MLP8	3		50	30	yes	yes		284	270
MLP9	5		100	30	no	no	175	280	276
MLP10	5		100	30	no	yes		265	**252**
Del. Int.	3						31	352	336
Kneser-Ney back-off	3							334	323
Kneser-Ney back-off	4							332	321
Kneser-Ney back-off	5							332	321
class-based back-off	3	150						348	334
class-based back-off	3	200						354	340
class-based back-off	3	500						326	**312**
class-based back-off	3	1000						335	319
class-based back-off	3	2000						343	326
class-based back-off	4	500						327	312
class-based back-off	5	500						327	312

- 从上图得到的一些结论：
  - 更多上下文时(高阶语言模型)神经模型性能改善，而原有LM没有太多受益
  - NN模型的隐单元(有无以及数量变化)是有影响的
  - NN模型与原有LM的混合是有帮助的
  - 从输入到输出的直接连接是否有用(图中看不出，但是作者有如下)：
    - 小语料时提供更好的泛化能力(很有限)，大语料时直接连接提供更快的收敛速度(2倍)

■AP语料结果

	n	h	m	direct	mix	train.	valid.	test.
MLP10	6	60	100	yes	yes		104	**109**
Del. Int.	3						126	132
Back-off KN	3						121	127
Back-off KN	4						113	119
Back-off KN	5						112	**117**

■由于语料规模大，所以只执行了5epochs迭代，结论和前面相似。

■ M&H 2007： Log-bilinear Language (LBL) model

Model type	Context size	Model test score	Mixture test score
Log-bilinear	5	117.0	97.3
Log-bilinear	10	107.8	92.1
Back-off KN3	2	129.8	
Back-off KN5	4	123.2	
Back-off KN6	5	123.5	
Back-off KN9	8	124.6	

■Mnih, A., & Teh, Y. W. (2012)

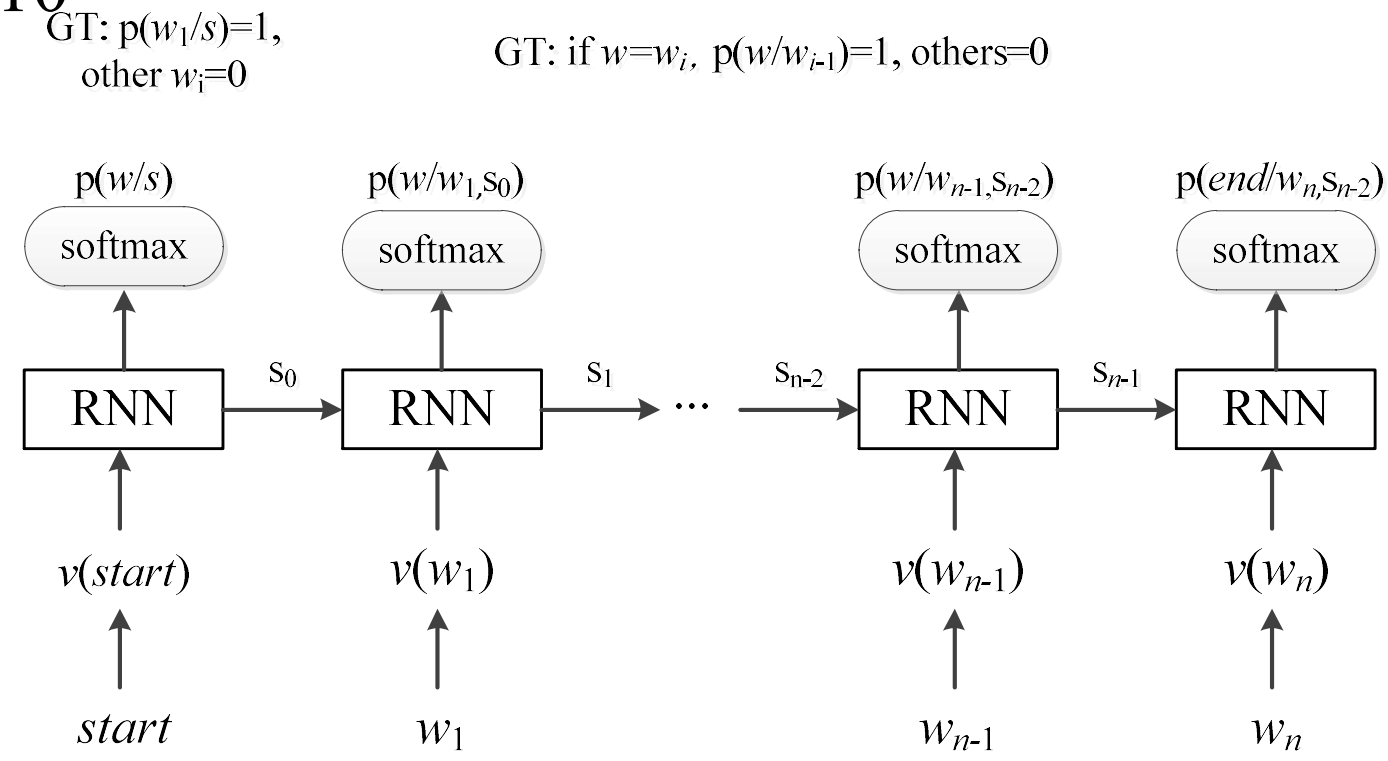
Model	Context size	Latent Dim	Test PP
3-Gram	2		130.8
4-Gram	3		122.1
5-Gram	4		121.5
6-Gram	5		121.7
LBL	2	100	145.5
LBL	3	100	135.6
LBL	5	100	129.8
LBL	10	100	124.0
LBL	10*2	100	38.6
LBL	5	200	123.6
LBL	10	200	117.7
LBL	10*2	200	33.6
LBL	10	300	116.4

- 神经概率语言模型(NPLM)
  - Bengio 2001，2003： NN-LM
    - A Neural Probabilistic Language Model，JMLR，2003
    - M&H 2007，2008: log-bilinear language (LBL) model
  - 基于NN的问题：固定输入长度
  - Mikolov 2010，2013：RNN-LM
    - Recurrent neural network based language model

RNN LM

- Mikolov, et al. Recurrent neural network based language model, Interspeech 2010

- 不固定输入
- 所以 N?



RNN LM

- 随训练数据规模增加时的情况
- 90：隐层节点数
- 2：合并低频词的词阈值

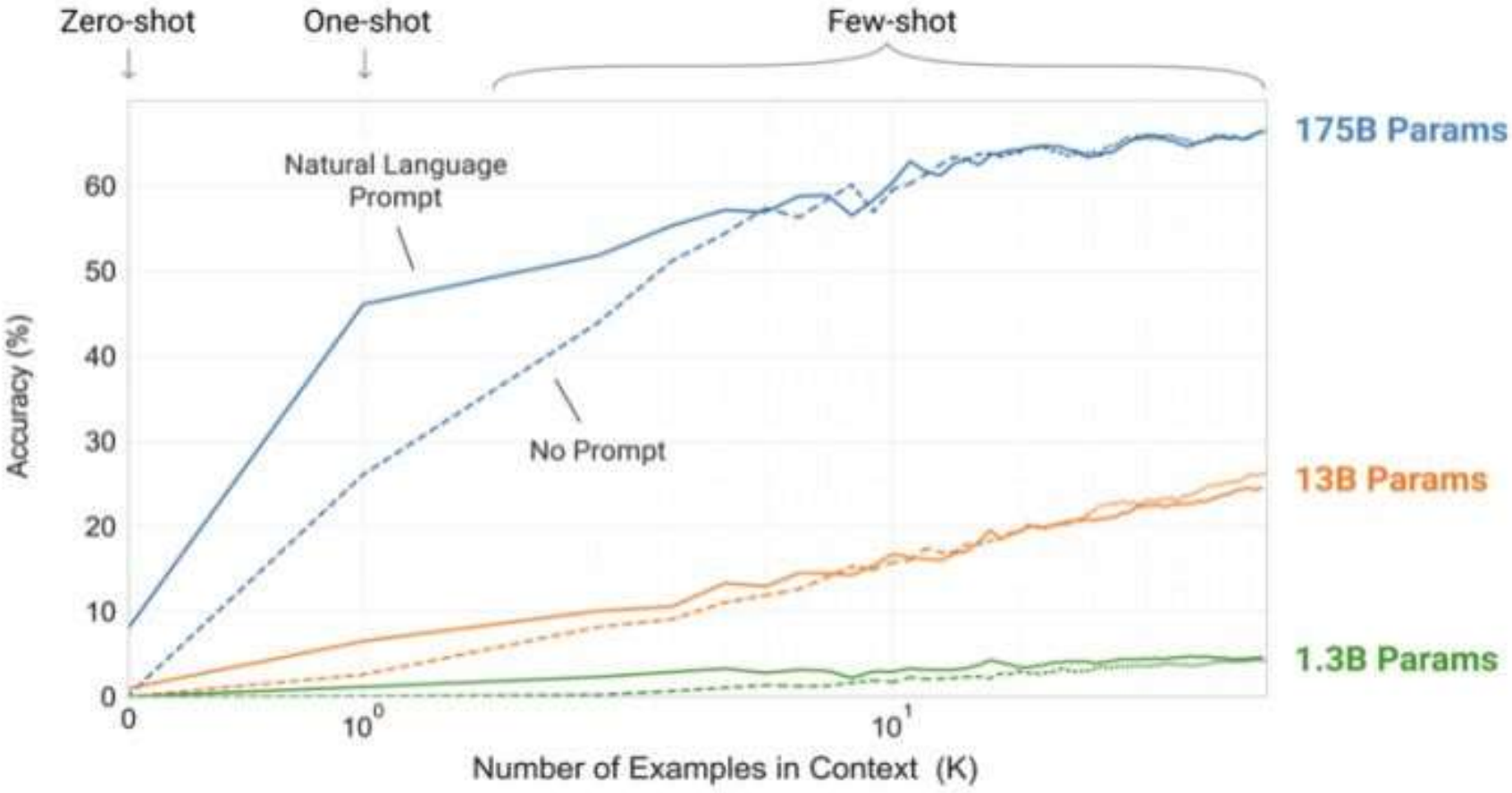
Model	#word	PPL
KN5 LM KN5 LM + RNN 90/2	200K	336 271
KN5 LM KN5 LM + RNN 90/2	1M (0.8 6hours)	287 225
KN5 LM KN5 LM + RNN 90/2	6.4M	221 156

- 神经语言模型的近期发展-1：

- 从基于词单元的LM到基于其他不同语言单元：
  - 英语：基于character间关系
  - 汉语：基于字间关系
  - 基于其他语言单元

- 神经语言模型的近期发展-2
  - 从RNN LM到 attention RNN LM
    - Mei2017,Salton 2017: attention RNN-LM
  - 从attention RNN LM到纯self-attention LM(Transformer)
    - GPT: Radford2018:improving language understanding by generative pre-training
    - GPT2：Radford 2019:Language Models are Unsupervised Multitask Learners (48 层 Transformer Decoder,15亿参数)
    - GPT3：2020 (96 层 Transformer Decoder, 1750亿参数(10^{11}))
      - https://beta.openai.com/

数据稀疏情况下


Zero-shot
One-shot
Few-shot
Natural Language Prompt
No Prompt
175B Params
13B Params
1.3B Params
Accuracy (%)
0
10
20
30
40
50
60
0
10^0
10^1
Number of Examples in Context (K)

大纲

- 引言
- 词表示
- 神经概率语言模型 (NPLM)
- 总结

- 总结
  - 从基于MLE的LM到基于神经网络的LM
  - 表示上：从符号词表示到向量词表示
    - 关键
  - 估计上：从基于统计的估计到基于预测的估计
    - 引入更多变化

思考与讨论

- N-gram语言模型建模了什么知识？
- 这种知识有什么用处？
- N-gram语言模型建模的知识有什么问题？

- 扩大N的其他途径：组块
- 灵活选择N的途径：递归

Thank You !