LLM 101

一起入门大语言模型

https://llm101.top

哔哩哔哩@一万篇论文笔记

一万篇论文笔记

微信扫描二维码，关注我的公众号

2024.12.04

课程简介

- 一门用于自学LLM的视频课程，难度系数 🌕🌕🌕🌗🌑
- 本课程面向**低年级研究生**、**高年级本科生**以及**对LLM感兴趣的朋友**
- 必备条件：会编程，能看懂Python代码
- 加分项：有一定的NLP/深度学习/机器学习背景，能看懂PyTorch代码
- 课程优点：
  - 理论知识 + 阅读论文**/开源代码** + **配套的编程实践**
  - 较为完善的LLM知识体系，尽量覆盖主流的LLM研究方向
- 课程不足：
  - 不涉及分布式、GPU和LLM System的相关知识
  - 可能存在错误

ChatGPT

CURSOR

编程实践的设计

- 个人的学习/项目经验
- 优秀的开源代码
- <u>国外相关课程的作业/项目</u>

什么是语言模型?

- 关于NLP的一些基础知识
  - NLP简介、常见的NLP任务、NLP历史、词向量、预训练-微调词向量
- 回顾语言模型的发展历史
  - N-gram LM、FFNN LM、RNN LM
- 编程实践(以Embedding为主线)
  - 词向量可视化、SiliconFlow Embedding API 句子向量相似度、基于transformers BERT fine-tuning的中文文本分类、基于arXiv论文数据 + SiliconFlow API + faiss + streamlit 构建论文搜索引擎demo
  - 数学：斯坦福CS224N 作业2中Understanding word2vec、普林斯顿 COS 484 作业1中LM和ppl理解

NLP的一些基础知识

- 自然语言处理 (Natural Language Processing, NLP) 是人工智能的子领域，研究的是**如何让计算机处理人类语言**

NLP的一些基础知识

- 自然语言处理 (Natural Language Processing, NLP) 是人工智能的子领域，研究的是**如何让计算机处理人类语言**
- 语言的2个特点：
  - 语言是序列(sequence)数据。“我们一起来学习大语言模型”
  - 语言是人类智能的体现，理解语言是通往AGI的必要条件

NLP的一些基础知识

- 自然语言处理 (Natural Language Processing, NLP) 是人工智能的子领域，研究的是**如何让计算机处理人类语言**
- 语言的2个特点：
  - 语言是序列(sequence)数据。“我们一起来学习大语言模型”
  - 语言是人类智能的体现，理解语言是通往AGI的必要条件
- 处理：
  - 自然语言理解 (Natural Language Understanding, NLU)
  - 自然语言生成 (Natural Language Generation, NLG)

NLP的一些基础知识

- 自然语言理解 (Natural Language **Understanding**, NLU)
  - 编码(Encoding)/表示学习(Representation Learning)

Embedding

before

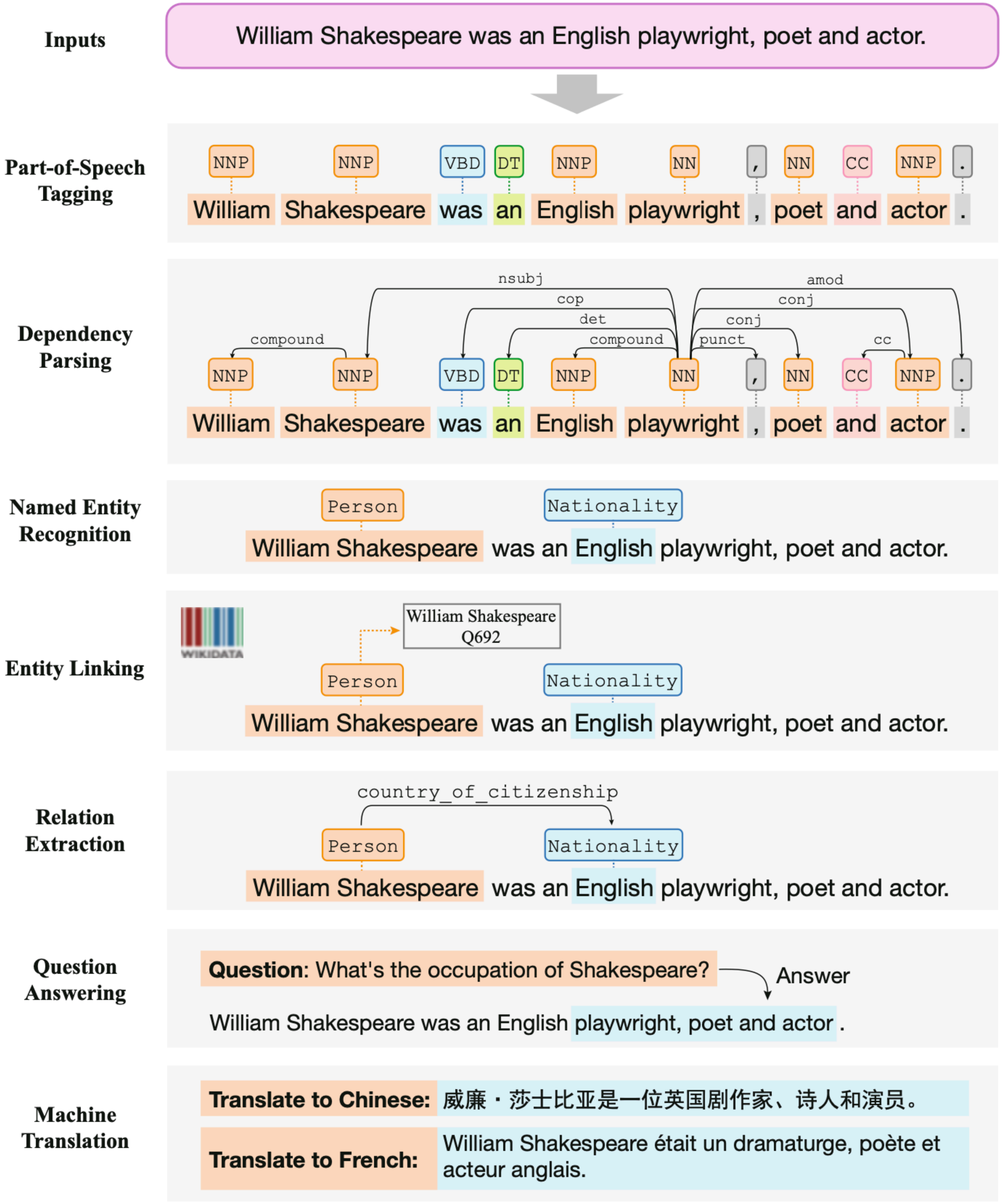
LLM时代

after

- 自然语言生成 (Natural Language **Generation**, NLG)
  - 解码(Decoding)/采样(Sampling)

NLU和NLG任务示例

ChatGPT



图片来自 Liu Z, Lin Y, Sun M.

Representation learning for natural language processing

The History of NLP

规则/专家系统　　(统计)机器学习　　词向量/DL/预训练-微调范式　　LLM

图片链接

The History of NLP

data iku

图片链接

词向量/DL/预训练-微调

- word2vec系列
    - word2vec 开启了NN for NLP的大航海时代

Tomas Mikolov

FOLLOWING

Senior Researcher, CIIRC CTU
Verified email at cvut.cz

Artificial Intelligence Machine Learning Language Modeling Natural Language Processing

TITLE	CITED BY	YEAR
Distributed representations of words and phrases and their compositionality T Mikolov, I Sutskever, K Chen, GS Corrado, J Dean Neural information processing systems	45705	2013
Efficient estimation of word representations in vector space T Mikolov arXiv preprint arXiv:1301.3781 3781	44835	2013
Distributed representations of sentences and documents Q Le, T Mikolov International conference on machine learning, 1188-1196	13156	2014

Google Scholar url

- GloVe: Global Vectors for Word Representation

参考 重读NLP经典论文系列

word2vec

- “一起学习大语言模型”

计算机内部如何存储字符串

- 字符编码 char-level

参考 重读NLP经典论文系列

word2vec

- “一起学习大语言模型”

计算机内部如何存储字符串

一	起	学	习	大	语	言	模	型

- 字符编码 char-level，字符序列(char sequence)

参考 重读NLP经典论文系列

word2vec

- “一起学习大语言模型”
- 分词得到word sequence

中文分词

word2vec

- “一起学习大语言模型”
- 分词得到word sequence

中文分词

```
>>> import pkuseg
>>> seg = pkuseg.pkuseg()
>>> text = seg.cut("一起学习大语言模型")
>>> print(text)
['一起', '学习', '大', '语言', '模型']
```

pkuseg

一起 / 学习 / 大 / 语言 / 模型

word2vec

- 创建词典 (Vocabulary) : [“一起”、“学习”、“大”、“语言”、“模型”]
- 每个词赋予唯一的ID，用one-hot形式表示
    - “一起” —> [1, 0, 0, 0, 0]
    - “学习” —> [0, 1, 0, 0, 0]
    - ……

Before word2vec

- 句子(word seq) “一起大模型”表示为 [1, 0, 1, 0, 1]
    - 用词频或者TFIDF作为单词权重
- 训练模型

分布式表示(distributed representations)

- 分布式假设(Distributional Hypothesis)
  - *A word is characterized by the company it keeps.*
  - 一个词的含义是由其上下文决定的。

苹果手机　　　　红彤彤的大**苹果**

- 分布式表示(Distributed Representations)
  - 分布式表示是分布式假设的具体实现，通过**模型**建模上下文关系
  - 用低维稠密向量表示单词
- 词向量(word embedding)历史悠久

CHAPTER 3

Distributed Representations

G. E. HINTON, J. L. McCLELLAND, and D. E. RUMELHART

Given a network of simple computing elements and some entities to be represented, the most straightforward scheme is to use one computing element for each entity. This is called a *local* representation. It is easy to understand and easy to implement because the structure of the physical network mirrors the structure of the knowledge it contains. The naturalness and simplicity of this relationship between the knowledge and the hardware that implements it have led many people to simply assume that local representations are the best way to use parallel hardware. There are, of course, a wide variety of more complicated implementations in which there is no one-to-one correspondence between concepts and hardware units, but these implementations are only worth considering if they lead to increased efficiency or to interesting emergent properties that cannot be conveniently achieved using local representations.

Hinton 1984

分布式表示(distributed representations)

- 分布式假设(Distributional Hypothesis)
  - *A word is characterized by the company it keeps.*
  - 一个词的含义是由其上下文决定的。

    苹果手机　　　　红彤彤的大苹果

- 分布式表示(Distributed Representations)
  - 分布式表示是分布式假设的具体实现，通过**模型**建模上下文关系
  - 用低维稠密向量表示单词
- 词向量(word embedding)历史悠久

CHAPTER 3

Distributed Representations

G. E. HINTON, J. L. McCLELLAND, and D. E. RUMELHART

Given a network of simple computing elements and some entities to be represented, the most straightforward scheme is to use one computing element for each entity. This is called a *local* representation. It is easy to understand and easy to implement because the structure of the physical network mirrors the structure of the knowledge it contains. The naturalness and simplicity of this relationship between the knowledge and the hardware that implements it have led many people to simply assume that local representations are the best way to use parallel hardware. There are, of course, a wide variety of more complicated implementations in which there is no one-to-one correspondence between concepts and hardware units, but these implementations are only worth considering if they lead to increased efficiency or to interesting emergent properties that cannot be conveniently achieved using local representations.

Hinton 1984

分布式表示(distributed representations)

- 分布式假设(Distributional Hypothesis)
  - *A word is characterized by the company it keeps.*
  - 一个词的含义是由其上下文决定的。

苹果手机　　　　红彤彤的大苹果

- 分布式表示(Distributed Representations)
  - 分布式表示是分布式假设的具体实现，通过**模型**建模上下文关系
  - 用低维稠密向量表示单词
- 词向量(word embedding)历史悠久

CS224N Assignment 1: Exploring Word Vectors

CHAPTER 3

Distributed Representations

G. E. HINTON, J. L. McCLELLAND, and D. E. RUMELHART

Given a network of simple computing elements and some entities to be represented, the most straightforward scheme is to use one computing element for each entity. This is called a *local* representation. It is easy to understand and easy to implement because the structure of the physical network mirrors the structure of the knowledge it contains. The naturalness and simplicity of this relationship between the knowledge and the hardware that implements it have led many people to simply assume that local representations are the best way to use parallel hardware. There are, of course, a wide variety of more complicated implementations in which there is no one-to-one correspondence between concepts and hardware units, but these implementations are only worth considering if they lead to increased efficiency or to interesting emergent properties that cannot be conveniently achieved using local representations.

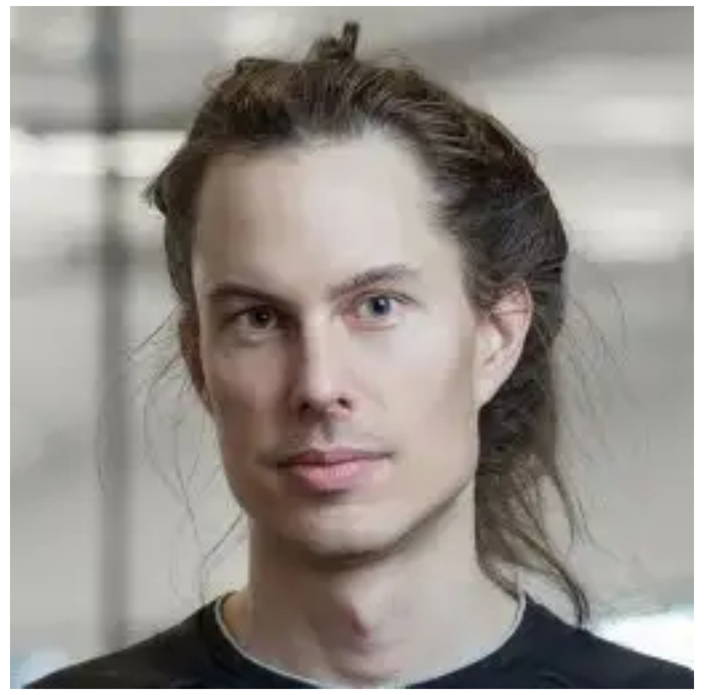
Hinton 1984

Why word2vec?

- Tomas Mikolov

Language Modeling and AI

- AI should be mostly unsupervised, learn by observations
- Language is the core of human intelligence
- Progress in language modeling should get us closer to general AI! (me, 2006)



[dreamBIG with Tomáš Mikolov]

[如何评价2023年NeurIPS时间检验奖颁给word2vec?]

Why word2vec?

- 在训练RNN LM模型时，用到了Embedding层，即将one-hot线性映射为词向量
- 模型训练目标是Language Model，词向量是副产物
- 专门为了高效训练词向量，设计了word2vec，用**比LM更简单的模型和任务**训练词向量

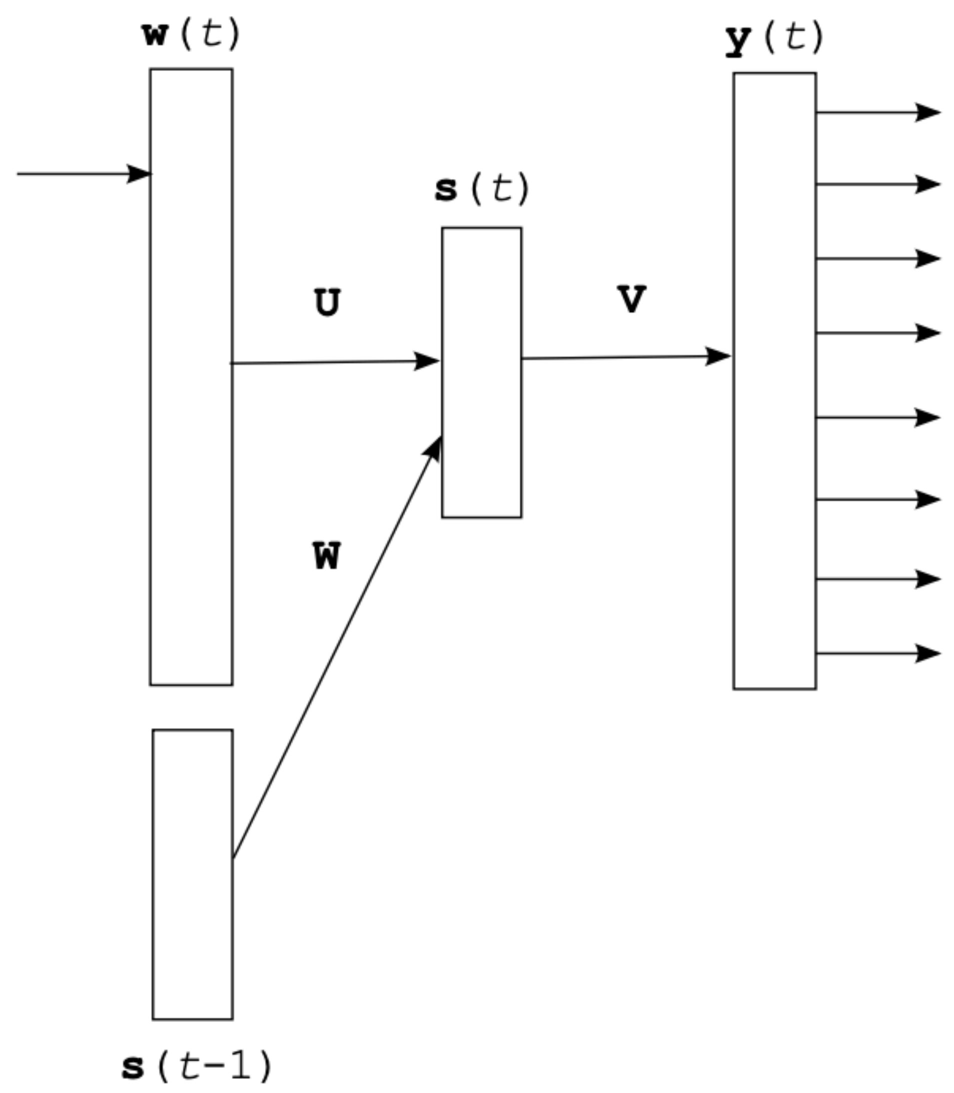


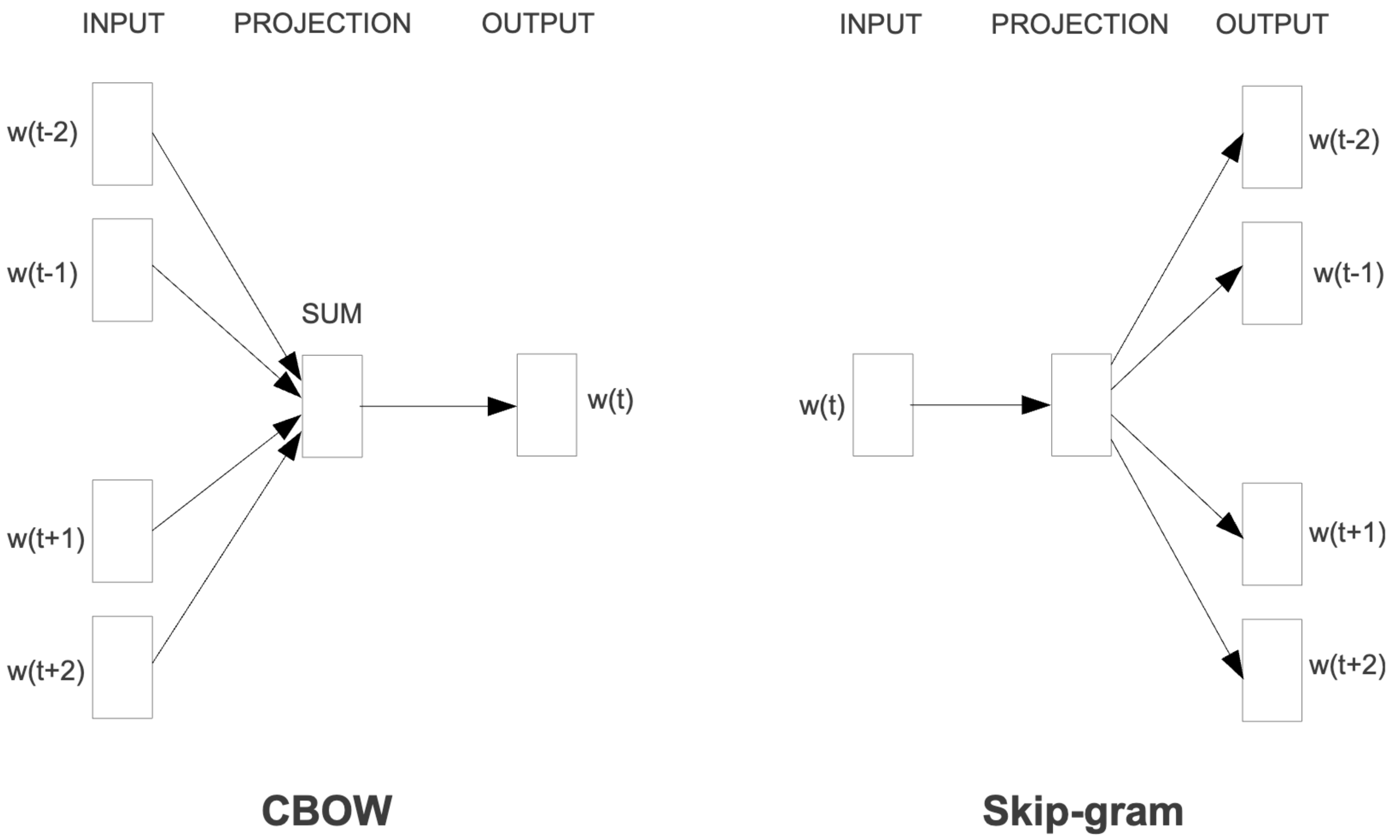
Figure 3.1: Simple recurrent neural network.

This 1-of-V orthogonal representation of words is projected linearly to a lower dimensional space, using a shared matrix P, called also a projection matrix. The matrix P is shared among words at different positions in the history, thus the matrix is the same when

来自Tomas Mikolov 博士论文

word2vec: 高效训练大规模词向量

- Continuous Bag-of-Words (CBOW)
  - 用滑动窗口构建上下文(context)
  - 滑动窗口中的两边的词预测中心词
- Continuous Skip-Gram
  - 用滑动窗口构建上下文(context)
  - 滑动窗口的中心词预测两边的词



词类别 vector("King") - vector("Man") + vector("Woman") ~ vector("Queen")

论文第5章中有更多示例

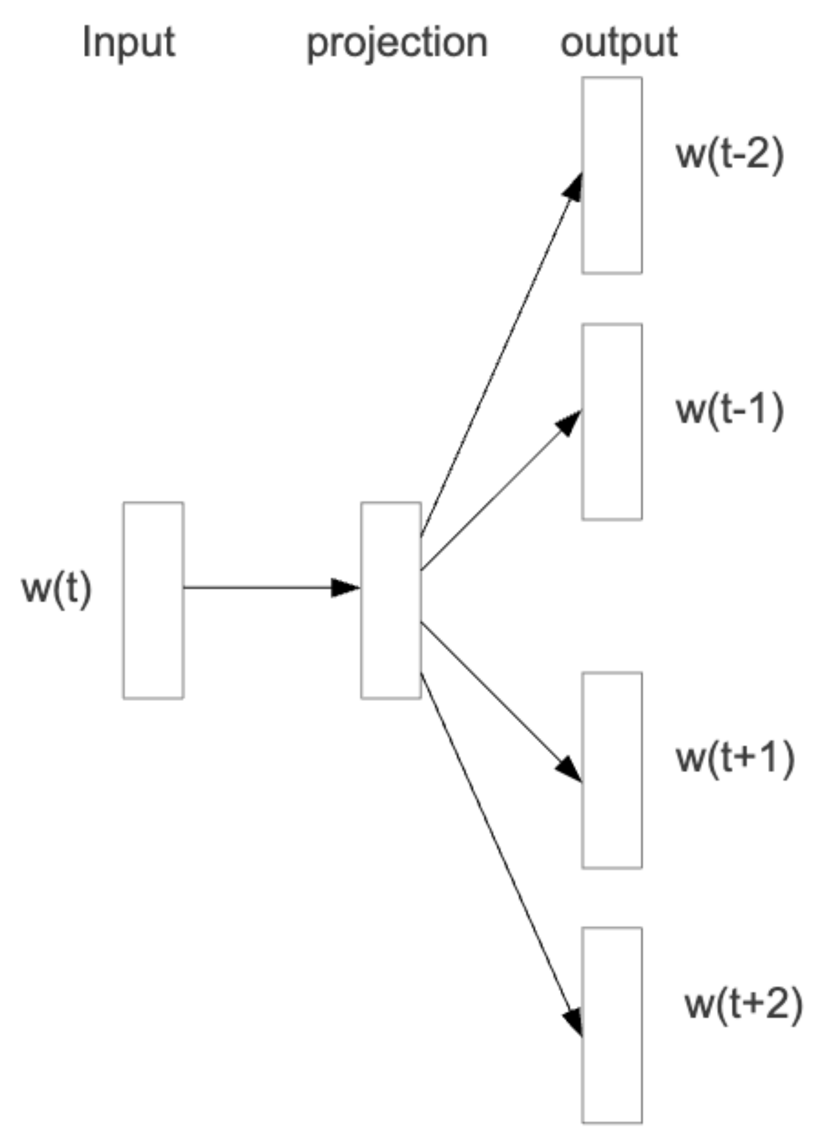
Efficient Estimation of Word Representations in Vector Space

word2vec之Skip-gram

- Skip-gram
  - 用滑动窗口构建上下文(context)
  - 滑动窗口的中心词预测两边的词

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^\top v_{w_I}\right)}$$



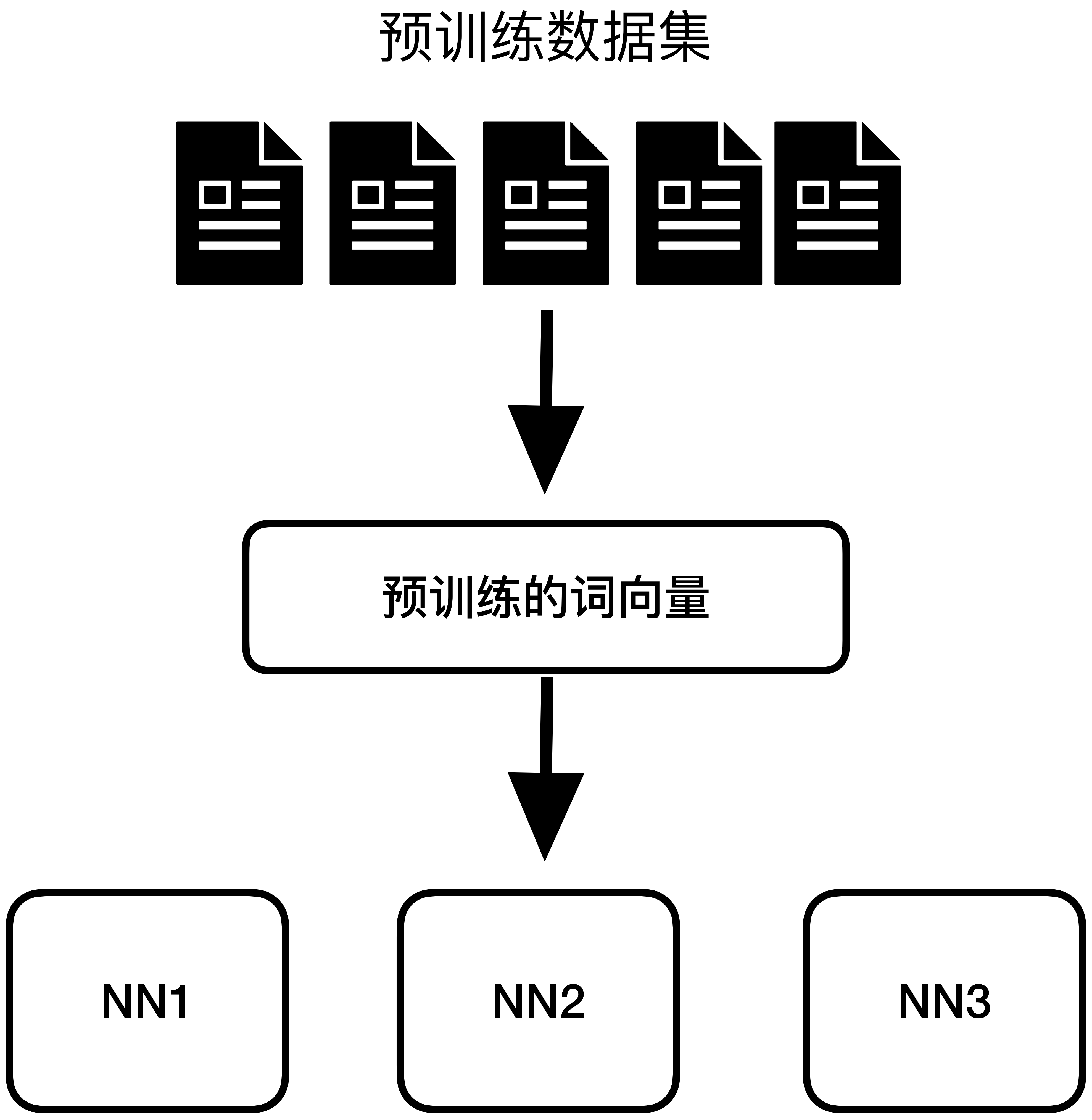
  - 负采样(negative sampling)

$$\log\sigma({v'_{w_O}}^\top v_{w_I}) + \sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\sigma(-{v'_{w_i}}^\top v_{w_I})\right]$$

Distributed Representations of Words and Phrases and their Compositionality

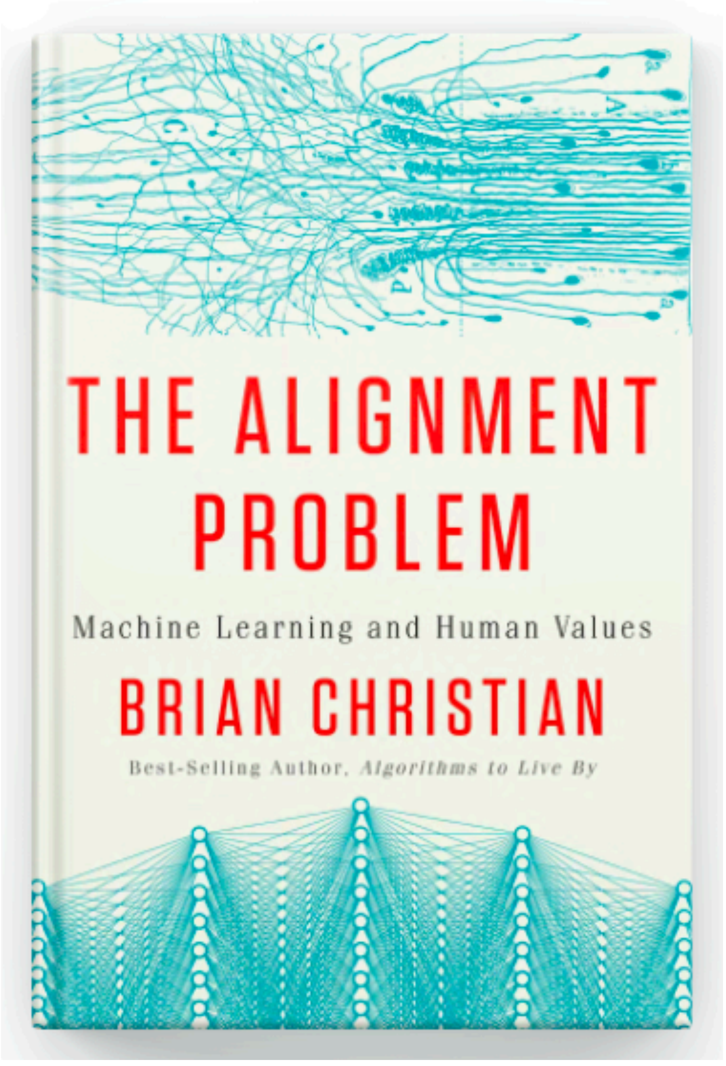
after word2vec

- 预训练-微调 词向量
- 在大规模语料(数据集)训练词向量模型，得到预训练的词向量
  - 或者用开源的已训练好的词向量
- 根据NLP任务设计神经网络(NN)模型
- 用预训练词向量初始化NN模型的Embedding层
- 训练NN模型过程中对Embedding层微调



AI伦理/安全

- doctor – man + woman ==> nurse
- shopkeeper – man + woman ==> housewife
- computer programmer – man + woman ==> homemaker
- 性别/种族/文化等的bias



The Alignment Problem

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

什么是语言模型?

- 关于NLP的一些基础知识
  - NLP简介、常见的NLP任务、NLP历史、词向量、预训练-微调词向量
- **回顾语言模型的发展历史**
  - N-gram LM、FFNN LM、RNN LM
- 编程实践(以Embedding为主线)
  - 词向量可视化、SiliconFlow Embedding API 句子向量相似度、基于transformers BERT fine-tuning的中文文本分类、基于arXiv论文数据 + SiliconFlow API + faiss + streamlit 构建论文搜索引擎demo
  - 数学：斯坦福CS224N 作业2中Understanding word2vec、普林斯顿 COS 484 作业1中LM和ppl理解

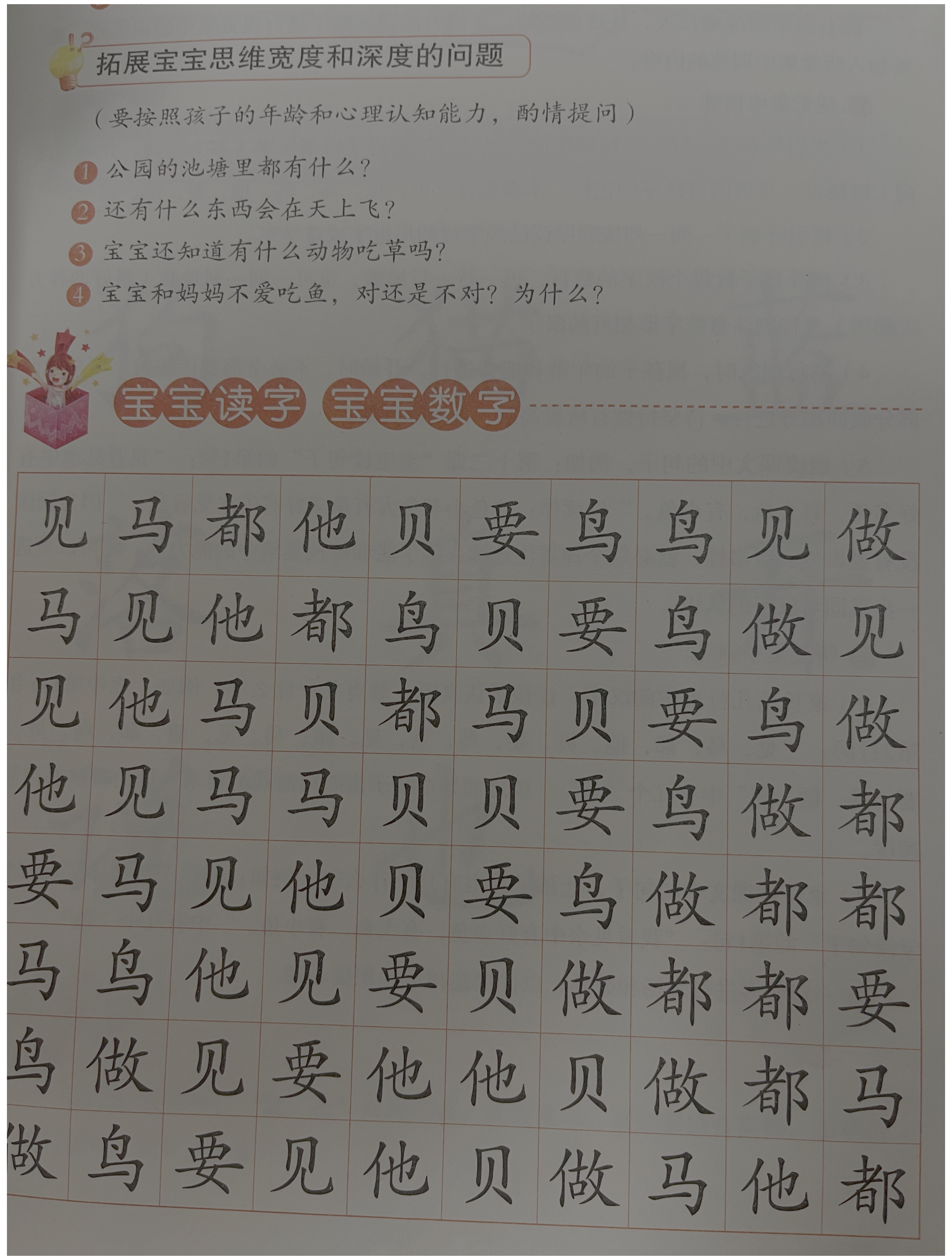
语言模型的发展历史

- 【定义1】 语言模型(Language Model, LM)：对于任意的token序列，它能够计算出这个序列是一句话的概率。

语音识别(Speech Recognition)

语言模型的发展历史

- 【定义1】 语言模型(Language Model, LM)：对于任意的token序列，它能够计算出这个序列是一句话的概率。



四五快读（第2册）

语言模型的定义

- $V=\{$“猫”, “狗”, “机器”, “语言”, “模型”, …$\}$ ， $w_i \in V$
- 【定义1】语言模型：给定V，能够计算出任意token序列 $w_1, w_2, \ldots, w_n$是一句话的概率$p(w_1, w_2, \ldots, w_n)$，其中$p \geq 0$。
- 如何计算概率p?

入门语言模型(Language Models)

语言模型的定义

- $V = \{$“猫”, “狗”, “机器”, “语言”, “模型”, …$\}$, $w_i \in V$
- 【定义1】语言模型：给定V，能够计算出任意token序列 $w_1, w_2, \ldots, w_n$是一句话的概率$p(w_1, w_2, \ldots, w_n)$，其中$p \geq 0$。
- 如何计算概率p?　**数数？！**

$$p(w_1, w_2, \ldots, w_n) = \frac{n}{N}$$

n → 在训练集中出现的次数

N → 训练集的句子数量

👎

入门语言模型(Language Models)

语言模型的定义

- $V=\{$“猫”, “狗”, “机器”, “语言”, “模型”, …$\}$ ，$w_i \in V$
- 【定义1】语言模型：给定V，能够计算出任意token序列 $w_1, w_2, ..., w_n$是一句话的概率$p(w_1, w_2, ..., w_n)$，其中$p \geq 0$。
- 如何计算概率p?　　**数数？！**

$$p(w_1, w_2, ..., w_n) = \frac{n}{N}$$

n - - - - - - - - → 在训练集中出现的次数

N - - - - - - - - → 训练集的句子数量

- 链式法则(chain rule)

$$p(w_1, w_2, ..., w_n) = p(w_1) \prod_{i=2}^{n} p(w_i | w_1, ..., w_{i-1})$$

入门语言模型(Language Models)

语言模型的定义

- 【定义1】语言模型：给定V，能够计算出任意token序列 $w_1, w_2, ..., w_n$是一句话的概率$p(w_1, w_2, ..., w_n)$，其中$p \geq 0$。
- 如何计算概率p？ **数数？！** $p(w_1, w_2, ..., w_n) = \frac{n}{N}$ 在训练集中出现的次数 训练集的句子数量
- 链式法则(chain rule) $p(w_1, w_2, ..., w_n) = p(w_1)\prod_{i=2}^{n} p(w_i | w_1, ..., w_{i-1})$
- 【定义2】语言模型：给定V，能够计算出任意条件概率 $p(w_i | w_1, w_2, ..., w_{i-1})$，其中$p \geq 0$。
- 【定义3】语言模型：给定V和任意token序列 $w_1, w_2, ..., w_n$，能够计算出V中每个token是w_{n+1}的概率

next-token prediction

马尔可夫假设(Markov Assumption)

- 一阶马尔可夫假设(first-order Markov assumption)：
  - 每一个词只依赖前一个词，$p(w_i | w_1, \ldots, w_{i-1}) \approx p(w_i | w_{i-1})$
- 那么，$p(w_1, w_2, \ldots, w_n) = p(w_1) \prod_{i=2}^{n} p(w_i | w_1, \ldots, w_{i-1})$

$$\approx p(w_1) \prod_{i=2}^{n} p(w_i | w_{i-1})$$

- 二阶马尔可夫假设：每个词只依赖前两个词，$p(w_i | w_1, \ldots, w_{i-1}) \approx p(w_i | w_{i-2}, w_{i-1})$

Michael Collins, Language Modeling

N-gram 语言模型: unigram、bigram、trigram、4-gram ...

- 假设N=2，bigram语言模型采用一阶马尔可夫假设 $p(w_i|w_1,\ldots,w_{i-1}) \approx p(w_i|w_{i-1})$
- 用**数数法**计算条件概率
  $$p(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$
  - count(*)表示*在训练集中的出现次数

```
 <s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>
```

Here are the calculations for some of the bigram probabilities from this corpus

$$P(\texttt{I}|\texttt{<s>}) = \frac{2}{3} = 0.67 \qquad P(\texttt{Sam}|\texttt{<s>}) = \frac{1}{3} = 0.33 \qquad P(\texttt{am}|\texttt{I}) = \frac{2}{3} = 0.67$$
$$P(\texttt{</s>}|\texttt{Sam}) = \frac{1}{2} = 0.5 \qquad P(\texttt{Sam}|\texttt{am}) = \frac{1}{2} = 0.5 \qquad P(\texttt{do}|\texttt{I}) = \frac{1}{3} = 0.33$$

Speech and Language Processing (3rd ed. draft)

N-gram 语言模型: unigram、bigram、trigram、4-gram ...

- 假设N=2，bigram语言模型采用一阶马尔可夫假设 $p(w_i|w_1,...,w_{i-1}) \approx p(w_i|w_{i-1})$
- 用数数法计算条件概率 $$p(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$
    - count(*)表示*在训练集中的出现次数
- 模型的参数量： $|V|^N$ 指数爆炸 $p(w_i|w_{i-N+1},...,w_{i-2},w_{i-1})$ 类比一个参数
    - 大量的条件概率是0，模型泛化(generalization)能力差，回退(backoff)、平滑(smoothing)技巧
- 长距离依赖(long dependency)问题

小明|出生|在|北京|他|目前|在|南京|上|大学|国庆|他|去|上海|旅游|请问|他|出生|在|? |

N-gram 语言模型: unigram、bigram、trigram、4-gram ...

- 假设N=3，trigram语言模型采用二阶马尔可夫假设 $p(w_i | w_1, ..., w_{i-1}) \approx p(w_i | w_{i-2}, w_{i-1})$
- 用数数法计算条件概率
  - count(*)表示*在训练集中的出现次数

$$p(w_i | w_{i-2}, w_{i-1}) = \frac{count(w_{i-2}, w_{i-1}, w_i)}{count(w_{i-2}, w_{i-1})}$$

- 模型的参数量： $|V|^N$ 指数爆炸 $p(w_i | w_{i-N+1}, ..., w_{i-2}, w_{i-1})$ 类比一个参数
  - 大量的条件概率是0，模型泛化(generalization)能力差，各种平滑(smoothing)技巧

“维度灾难”（Curse of Dimensionality）

- 长距离依赖(long dependency)问题

小明|出生|在|北京|他|目前|在|南京|上|大学|国庆|他|去|上海|旅游|请问|他|出生|在|? |

FFNN 语言模型

- 基于分布式表示：词向量
- C是Embedding层($|V| \times m$), g是**FFNN**/RNN/…
- 输入是前n-1个token组成的context/prefix，输出下一个token的概率

$$x = (C(w_{t-1}), C(w_{t-2}), \cdots, C(w_{t-n+1})).$$

$$y = b + Wx + U \tanh(d + Hx)$$

$$\hat{P}(w_t | w_{t-1}, \cdots w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}$$

$$f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t | w_1^{t-1})$$

$$L = \frac{1}{T} \sum_t \log f(w_t, w_{t-1}, \cdots, w_{t-n+1}; \theta) + R(\theta),$$

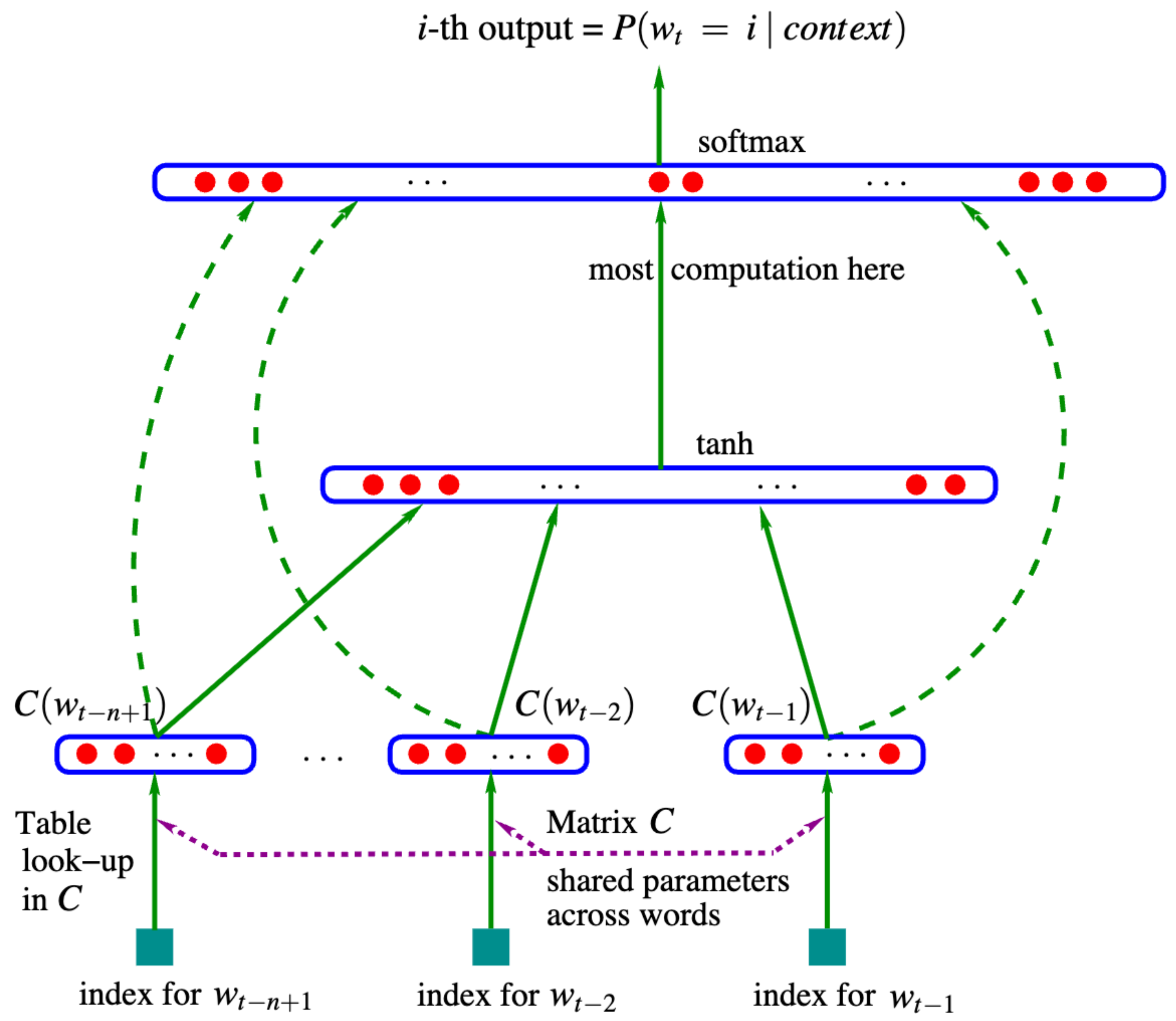


Figure 1: Neural architecture: $f(i, w_{t-1}, \cdots, w_{t-n+1}) = g(i, C(w_{t-1}), \cdots, C(w_{t-n+1}))$ where neural network and $C(i)$ is the i-th word feature vector.

Yoshua Bengio, 2003, A Neural Probabilistic Language Model

～14M Tokens

context: 5～10

RNN语言模型

$$x(t) = w(t) + s(t-1)$$

$$s_j(t) = f\left(\sum_i x_i(t)u_{ji}\right) \qquad f(z) = \frac{1}{1+e^{-z}}$$

$$y_k(t) = g\left(\sum_j s_j(t)v_{kj}\right) \qquad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}$$

- Extensions of the basic recurrent neural network language model:
  - Simple classes based on unigram frequency of words
  - Joint training of neural network and maximum entropy model
  - Adaptation of neural net language models by sorting the training data
  - Adaptation of neural net language models by training the model during processing of the test data

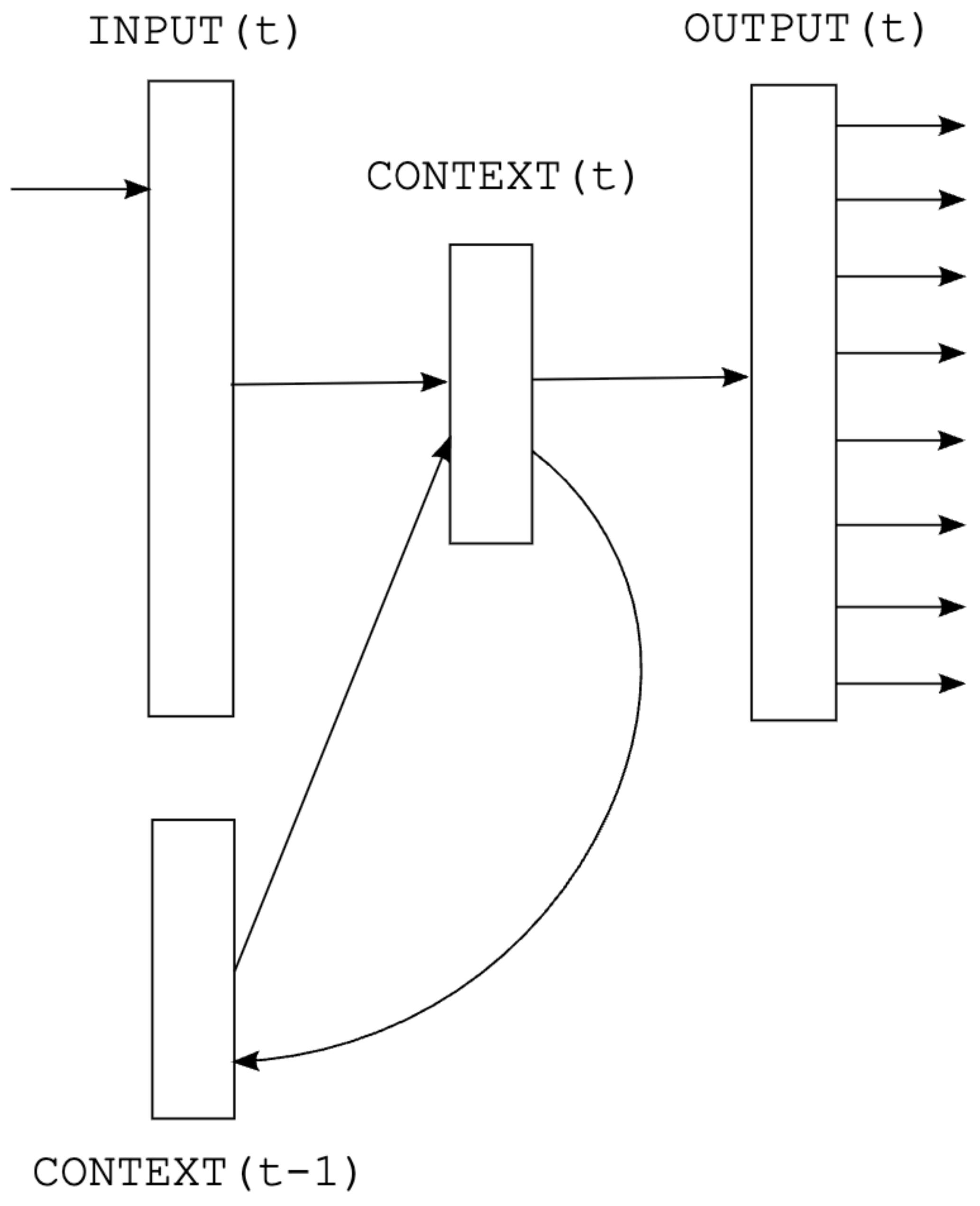


Figure 1: *Simple recurrent neural network.*

Recurrent neural network based language model

开源 RNNLM Toolkit

训练技巧

什么是语言模型?

- 关于NLP的一些基础知识
  - NLP简介、常见的NLP任务、NLP历史、词向量、预训练-微调词向量
- 回顾语言模型的发展历史
  - N-gram LM、FFNN LM、RNN LM
- **编程实践(以Embedding为主线)**
  - 词向量可视化、SiliconFlow Embedding API 句子向量相似度、基于transformers BERT fine-tuning的中文文本分类、基于arXiv论文数据 + SiliconFlow API + faiss + streamlit 构建论文搜索引擎demo
  - 数学：斯坦福CS224N 作业2中Understanding word2vec、普林斯顿 COS 484 作业1中LM和ppl理解

Perplexity of fixed-length models