Natural Language Processing with Deep Learning

CS224N/Ling284

Christopher Manning

Lecture 1: Introduction and Word Vectors

系列内容 Awesome AI Courses Notes Cheatsheets

中英字幕视频　**课件动态注释**　官方note翻译　作业代码解析

I：课程简介与词向量入门

本门课程**全部资料和信息**已整理发布，扫描下方的任意二维码，均可获取！！

微信公众号·全套资料
回复CS224n/底部**菜单栏**

Bilibili·课程视频
视频**简介**/置顶**评论**

GitHub·项目代码
阅读**ReadMe**/点击**超链接**

系列内容 Awesome AI Courses Notes Cheatsheets

是 ShowMeAI 资料库的分支系列，覆盖最具知名度的TOP20+门AI课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

有任何建议和反馈，也欢迎通过右侧渠道和我们联络 (* 3)~

机器学习	Machine Learning	Stanford · CS229
深度学习	Deep Learning	Stanford · CS230
自然语言处理	**Natural Language Processing**	**Stanford · CS224n**
计算机视觉	Computer Vision	Stanford · CS231n
深度强化学习	Deep Reinforcement Learning	UCBerkeley · CS285
知识图谱	Knowledge Graphs	Stanford · CS520
图机器学习	Machine Learning with Graphs	Stanford · CS224W
自动驾驶	DL for Self-Driving Cars	MIT · 6.S094
...	...	...

斯坦福大学(Stanford) *Natural Language Processing with Deep Learning (CS224n)* 课程，是本系列的第三门产出。

课程版本为2019 Winter，核心深度内容(transformer、bert、问答、摘要、文本生成等)在当前(2021年)工业界和研究界依旧是前沿的方法。最新版课程的笔记生产已在规划中，也敬请期待。

笔记内容经由深度加工整合，以**5**个部分构建起完整的“CS224n内容世界”，并依托GitHub创建了汇总页。快扫描二维码，跳转进入吧！

中英字幕 **视频**

课件 动态注释

官方Note **翻译**

作业 代码解析

结业 项目参考

微信公众号

扫码回复”**CS224n**”，下载最新全套资料

回复“**添砖加瓦**”，成为AI内容创作者

授课计划

Lecture Plan

Lecture 1: Introduction and Word Vectors

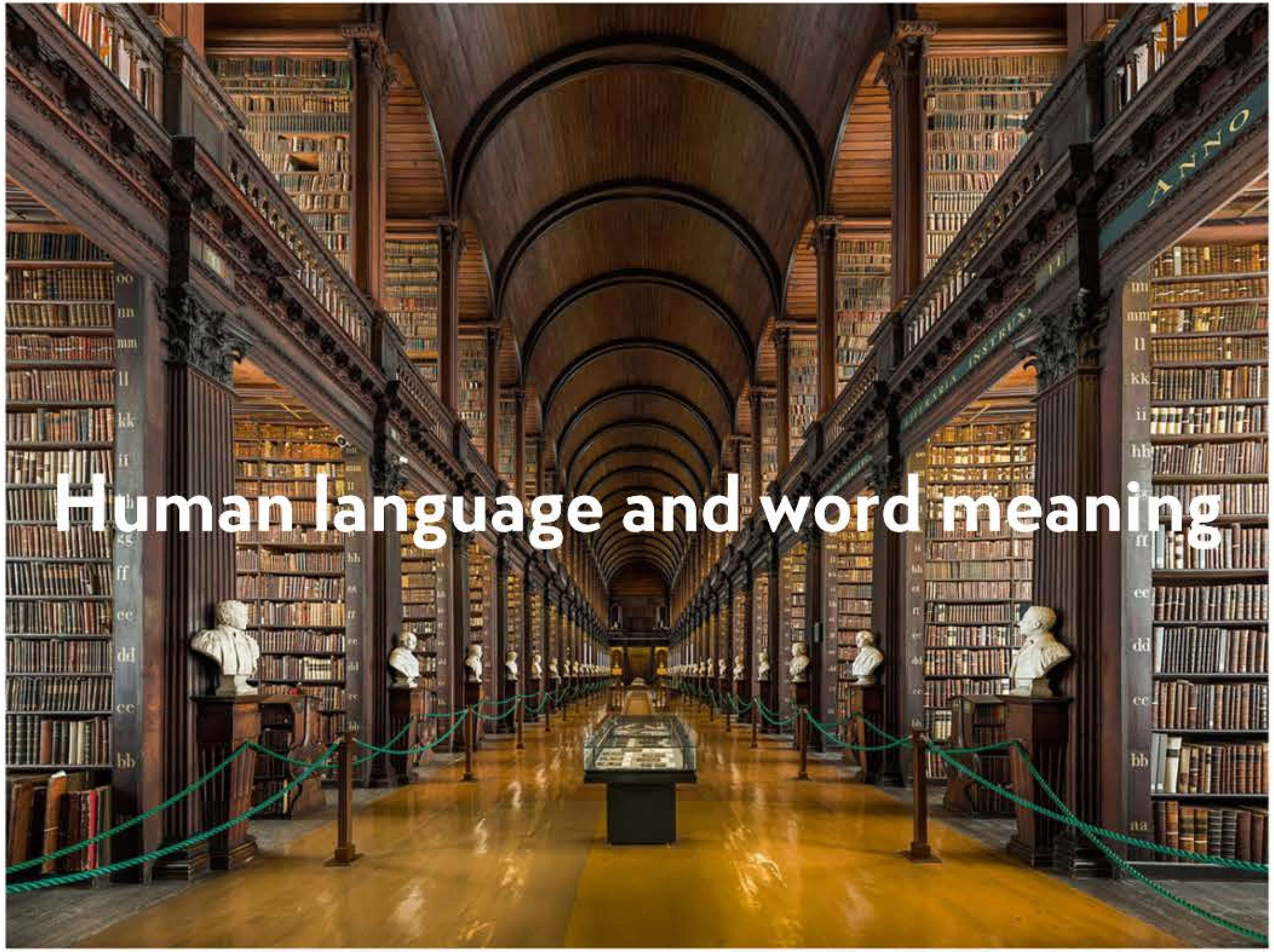
1. The course (10 mins)
2. Human language and word meaning (15 mins)
3. Word2vec introduction (15 mins)
4. Word2vec objective function gradients (25 mins)
5. Optimization basics (5 mins)
6. Looking at word vectors (10 mins or less)

ShowMeAI

CS224n – Lecture 1

- 中英字幕视频：82分钟
- 课件动态注释：课件22页，34个注释块
- 配合视频学习，约需1小时

扫码，学习本节课程视频 | B站·中英双语字幕



人类的语言与词汇含义

ShowMeAI

- 人类之所以比类人猿更“聪明”，是因为我们有语言，因此是一个人机网络，其中人类语言作为网络语言。人类语言具有信息功能和社会功能。
- 据估计，人类语言只有大约5000年的短暂历史。语言和写作是让人类变得强大的原因之一。它使知识能够在空间上传送到世界各地，并在时间上传送。
- 但是，相较于如今的互联网的传播速度而言，人类语言是一种缓慢的语言。然而，只需人类语言形式的几百位信息，就可以构建整个视觉场景。这就是自然语言如此迷人的原因。

扫码，学习本节课程视频 | B站·中英双语字幕

1. How do we represent the meaning of a word?

Definition: **meaning** (Webster dictionary)

- the idea that is represented by a word, phrase, etc.
- the idea that a person wants to express by using words, signs, etc.
- the idea that is expressed in a work of writing, art, etc.

Commonest linguistic way of thinking of meaning:

signifier (symbol) ⟺ signified (idea or thing)

= denotational semantics

我们如何表达一个词的意思?

ShowMeAI

- 用一个词、词组等表示的概念。
- 一个人想用语言、符号等来表达的想法。
- 表达在作品、艺术等方面的思想。

ShowMeAI

- 理解意义的最普遍的语言方式(linguistic way):
- 语言符号与语言符号的意义的转化
  - denotational semantics: 语义

扫码，学习本节课程视频 | B站·中英双语字幕

How do we have usable meaning in a computer?

Common solution: Use e.g. WordNet, a thesaurus containing lists of **synonym sets** and **hypernyms** ("is a" relationships).

e.g. synonym sets containing "good":

```
from nltk.corpus import wordnet as wn
poses = { 'n':'noun', 'v':'verb', 's':'adj (s)', 'a':'adj', 'r':'adv'}
for synset in wn.synsets("good"):
    print("{}: {}".format(poses[synset.pos()],
          ", ".join([l.name() for l in synset.lemmas()])))
```

```
noun: good
noun: good, goodness
noun: good, goodness
noun: commodity, trade_good, good
adj: good
adj (sat): full, good
adj: good
adj (sat): estimable, good, honorable, respectable
adj (sat): beneficial, good
adj (sat): good
adj (sat): good, just, upright
…
adverb: well, good
adverb: thoroughly, soundly, good
```

e.g. hypernyms of "panda":

```
from nltk.corpus import wordnet as wn
panda = wn.synset("panda.n.01")
hyper = lambda s: s.hypernyms()
list(panda.closure(hyper))
```

```
[Synset('procyonid.n.01'),
Synset('carnivore.n.01'),
Synset('placental.n.01'),
Synset('mammal.n.01'),
Synset('vertebrate.n.01'),
Synset('chordate.n.01'),
Synset('animal.n.01'),
Synset('organism.n.01'),
Synset('living_thing.n.01'),
Synset('whole.n.02'),
Synset('object.n.01'),
Synset('physical_entity.n.01'),
Synset('entity.n.01')]
```

在电脑里，如何有可用的意义？

ShowMeAI

WordNet:

- 一个包含同义词集和上位词("is a" 关系) 的列表的辞典。

扫码，学习本节课程视频 | B站·中英双语字幕

Problems with resources like WordNet

- Great as a resource but missing nuance
  - e.g. "proficient" is listed as a synonym for "good". This is only correct in some contexts.
- Missing new meanings of words
  - e.g., wicked, badass, nifty, wizard, genius, ninja, bombest
  - Impossible to keep up-to-date!
- Subjective
- Requires human labor to create and adapt
- Can't compute accurate word similarity →

WordNet的问题

ShowMeAI

- 作为一个资源很好，但忽略了细微差别
  - 例如 "proficient" 被列为 "good" 的同义词。这只在某些上下文中是正确的。

ShowMeAI

- 缺少单词的新含义
  - 难以持续更新！
  - 例如 wicked, badass, nifty, wizard, genius, ninja, bombast

ShowMeAI

- 主观的
- 需要人类劳动来创造和调整
- 无法计算单词相似度

Representing words as discrete symbols

In traditional NLP, we regard words as discrete symbols:
hotel, conference, motel – a localist representation

Means one 1, the rest 0s

Words can be represented by one-hot vectors:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

Vector dimension = number of words in vocabulary (e.g., 500,000)

对词做离散表征

ShowMeAI

- 在传统的自然语言处理中，我们把词语看作离散的符号：hotel, conference, motel – a localist representation

ShowMeAI

- 单词可以通过独热向量(one-hot vectors)
  - 独热向量：只有一个1，其余均为0的稀疏向量

ShowMeAI

- 向量维度 = 词汇量(如500,000)

扫码，学习本节课程视频 | B站·中英双语字幕

Problem with words as discrete symbols

Example: in web search, if user searches for "Seattle motel", we would like to match documents containing "Seattle hotel".

But:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

These two vectors are orthogonal.

There is no natural notion of **similarity** for one-hot vectors!

Solution:

- Could try to rely on WordNet's list of synonyms to get similarity?
  - But it is well-known to fail badly: incompleteness, etc.
- **Instead: learn to encode similarity in the vectors themselves**

离散表征的弱点

ShowMeAI

- 所有向量是正交的。对于独热向量，没有关于相似性概念，并且向量维度过大。

ShowMeAI

Solution:

- 使用类似 WordNet 的工具中的列表，获得相似度，但会因不够完整而失败。
- 学习在向量本身中编码相似性。

扫码，学习本节课程视频 | B站·中英双语字幕

Representing words by their context

- Distributional semantics: **A word's meaning is given by the words that frequently appear close-by**
  - *"You shall know a word by the company it keeps"* (J. R. Firth 1957: 11)
  - One of the most successful ideas of modern statistical NLP!
- When a word w appears in a text, its context is the set of words that appear nearby (within a fixed-size window).
- Use the many contexts of w to build up a representation of w

…government debt problems turning into	*banking*	*crises as happened in 2009…*
…saying that Europe needs unified	*banking*	*regulation to replace the hodgepodge…*
…India has just given its	*banking*	*system a shot in the arm…*

These context words will represent ***banking***

基于上下文的词汇表征

ShowMeAI

- Distributional semantics：一个单词的意思可以由经常出现在它附近的单词给出。
  - 现代统计NLP最成功的理念之一
  - 有点物以类聚，人以群分的感觉

ShowMeAI

- 当一个单词w出现在文本中时，它的上下文是出现在其附近的一组单词(在一个固定大小的窗口中)。

ShowMeAI

- 使用w的许多上下文来构建w的表示

Word vectors

We will build a dense vector for each word, chosen so that it is similar to vectors of words that appear in similar contexts

$$
\textit{banking} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}
$$

Note: word vectors are sometimes called word embeddings or word representations. They are a distributed representation.

词向量表示

ShowMeAI

- 我们为每个单词构建一个稠密的向量，使其与出现在相似上下文中的单词向量相似。

ShowMeAI

Note:

- 词向量(word vectors)有时被称为词嵌入(word embeddings)或词表示(word representations)。
- 它们是分布式表示(distributed representation)。

扫码，学习本节课程视频 | B站·中英双语字幕

3. Word2vec: Overview

Word2vec (Mikolov et al. 2013) is a framework for learning word vectors

Idea:

- We have a large corpus of text
- Every word in a fixed vocabulary is represented by a vector
- Go through each position *t* in the text, which has a center word *c* and context ("outside") words *o*
- Use the similarity of the word vectors for *c* and *o* to calculate the probability of *o* given *c* (or vice versa)
- Keep adjusting the word vectors to maximize this probability

Word2vec原理介绍

ShowMeAI

- Word2vec (Mikolov et al. 2013)是一个学习单词向量的框架.

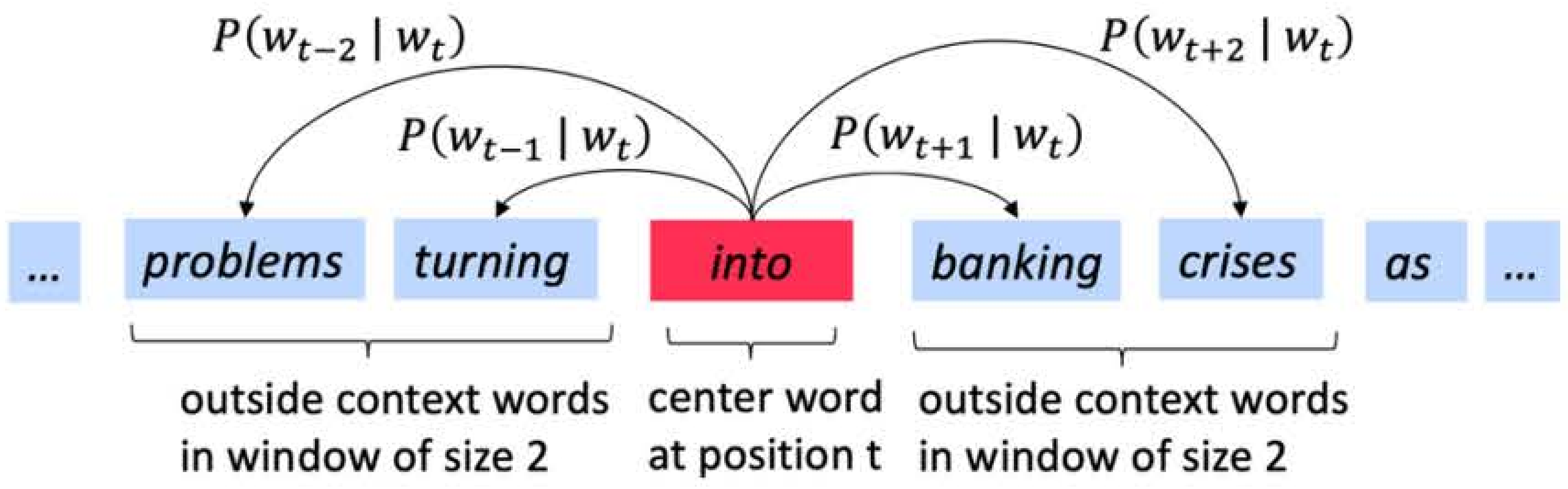
ShowMeAI

Idea / 思路:

- 我们有大量的文本
- 固定词汇表中的每个单词都由一个向量表示
- 文本中的每个位置t，其中有一个中心词 c 和上下文("外部")单词 o
- 使用 c 和 o 的词向量的相似性来计算给定c的o的概率(反之亦然)
- 不断调整词向量来最大化这个概率

Word2Vec Overview

- Example windows and process for computing $P(w_{t+j} \mid w_t)$



Word2vec原理介绍

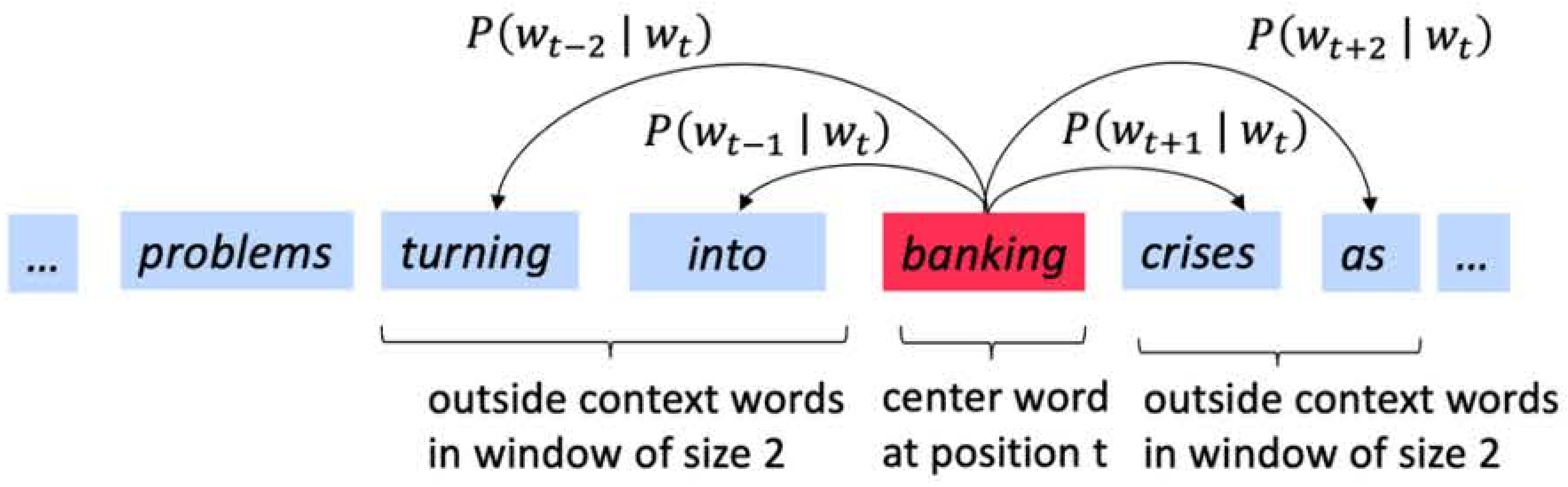
ShowMeAI

- 下图为窗口大小 $j=2$ 时的 $P(w_{t+j}|w_t)$
- 计算过程，center word 为 into

扫码，学习本节课程视频 | B站·中英双语字幕

Word2Vec Overview

- Example windows and process for computing $P(w_{t+j} \mid w_t)$



Word2vec原理介绍

ShowMeAI

- 下图为窗口大小 $j=2$ 时的 $P(w_{t+j}|w_t)$
- 计算过程，center word 为 banking

扫码，学习本节课程视频 | B站·中英双语字幕

Word2vec: objective function

For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_j.

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} \mid w_t; \theta)$$

θ is all variables to be optimized

sometimes called *cost* or *loss* function

The objective function $J(\theta)$ is the (average) negative log likelihood:

$$J(\theta) = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m \le j \le m \\ j \ne 0}} \log P(w_{t+j} \mid w_t; \theta)$$

Minimizing objective function ⟺ Maximizing predictive accuracy

Word2vec目标函数

ShowMeAI

- 对于每个位置$t = 1, \ldots, T$，在大小为m的固定窗口内预测上下文单词。给定中心词 w_j。
  - θ 为所有需要优化的变量。

ShowMeAI

- 目标函数 $J(\theta)$ 是(平均)负对数似然：
  - 有时被称为代价函数或损失函数
  - 最小化目标函数 ⇔ 最大化预测精度

ShowMeAI

补充解读

- 其中 log 形式是方便将连乘转化为求和，负号是希望将极大化似然率转化为极小化损失函数的等价问题
- 在连乘之前使用log转化为求和非常有效，特别是做优化时

Word2vec: objective function

- We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m\\ j\ne 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$

- Question: How to calculate $P\left(w_{t+j} \mid w_t; \theta\right)$?
- Answer: We will *use two* vectors per word *w*:
  - v_w when *w* is a center word
  - u_w when *w* is a context word
- Then for a center word *c* and a context word *o*:

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}$$

Word2vec目标函数

ShowMeAI

- 我们希望最小化目标函数

ShowMeAI

- 问题：如何计算 $P(w_{t+j}|w_t;\theta)$?
- 回答：对于每个单词都是用两个向量
  - v_w 当 w 是中心词时
  - u_w 当 w 是上下文词时

ShowMeAI

- 于是对于一个中心词 c 和一个上下文词 o

Word2vec: objective function

- We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m \\ j\ne 0}} \log P(w_{t+j} \mid w_t; \theta)$$

- Question: How to calculate $P(w_{t+j} \mid w_t; \theta)$?
- Answer: We will *use two* vectors per word *w*:
  - v_w when *w* is a center word
  - u_w when *w* is a context word
- Then for a center word *c* and a context word *o*:

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}$$

Word2vec目标函数

ShowMeAI

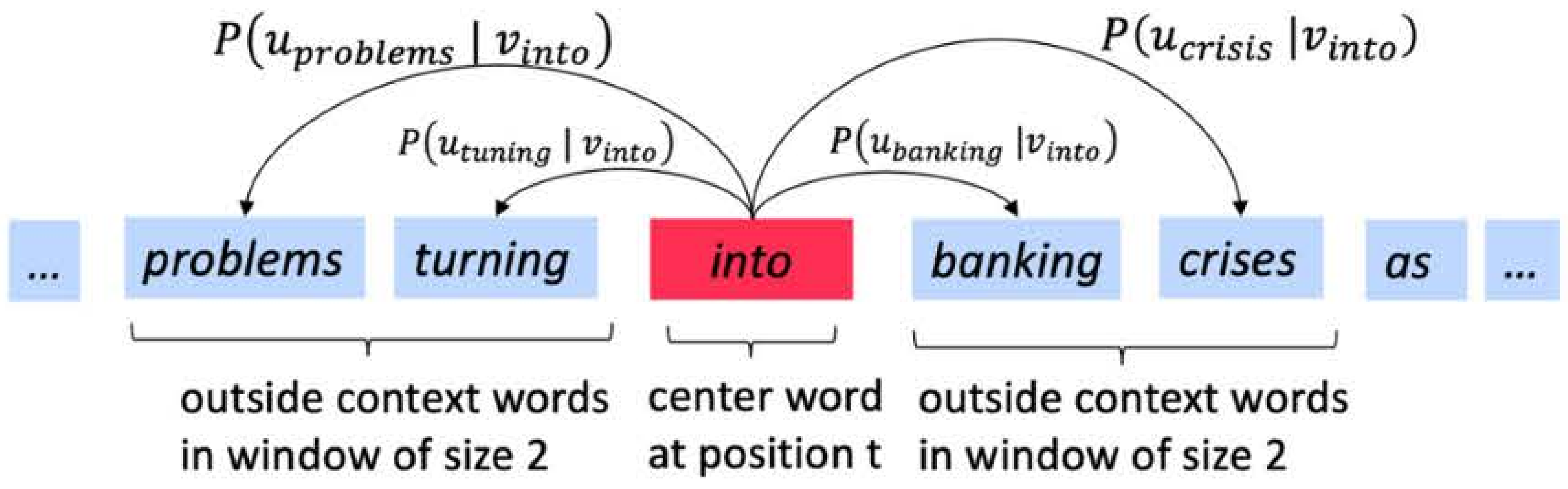
补充解读

- 公式中，向量 u_o 和向量 v_c 进行点乘。
- 向量之间越相似，点乘结果越大，从而归一化后得到的概率值也越大。
- 模型的训练正是为了使得具有相似上下文的单词，具有相似的向量。
- 点积是计算相似性的一种简单方法，在注意力机制中常使用点积计算得分score。

扫码，学习本节课程视频 | B站·中英双语字幕

Word2Vec Overview with Vectors

- Example windows and process for computing $P(w_{t+j} \mid w_t)$
- $P(u_{problems} \mid v_{into})$ short for $P(problems \mid into\ ; u_{problems}, v_{into}, \theta)$



从向量角度回顾Word2vec

ShowMeAI

- 计算 $P(w_{t+j}|w_t)$ 的示例
- 这里把 $P(problems|into; u_{problems}, v_{into}, \theta)$ 简写为 $P(u_{problems}|v_{into})$

ShowMeAI

- 窗口大小2的上下文分布
- 左右2个单词 + 一个中心词

扫码，学习本节课程视频 | B站·中英双语字幕

Word2vec: prediction function

Exponentiation makes anything positive

$$P(o|c) = \frac{\exp(\boxed{u_o^T v_c})}{\boxed{\sum_{w \in V} \exp(u_w^T v_c)}}$$

Dot product compares similarity of o and c.
$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$
Larger dot product = larger probability

Normalize over entire vocabulary to give probability distribution

- This is an example of the softmax function $\mathbb{R}^n \to \mathbb{R}^n$

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)} = p_i$$

- The softmax function maps arbitrary values x_i to a probability distribution p_i
  - "max" because amplifies probability of largest x_i
  - "soft" because still assigns some probability to smaller x_i
  - Frequently used in Deep Learning

Word2vec预测函数

ShowMeAI

- 取幂使任何数都为正
- 点积比较 o 和 c 的相似性
  $u^T v = u.v = \sum_{i=1}^{n} u_i v_i$，点积越大，概率越大
- 对整个词汇表进行标准化，从而给出概率分布

ShowMeAI

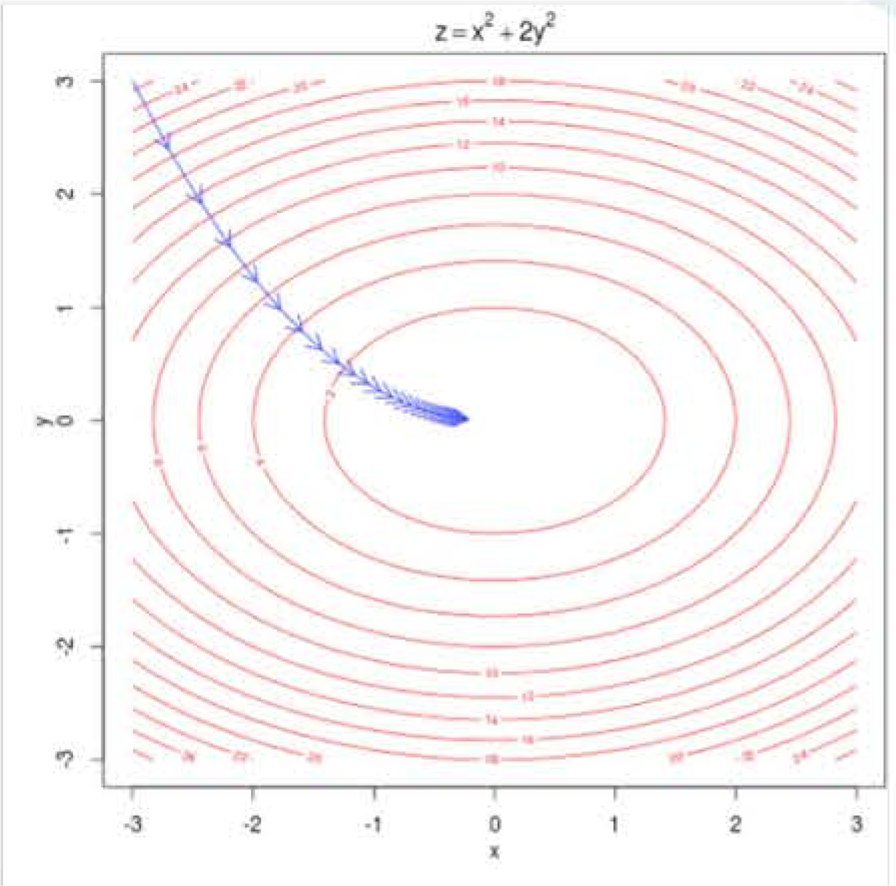
- softmax function $\mathbb{R}^n \in \mathbb{R}^n$示例
- 将任意值 x_i 映射到概率分布 p_i
  - max: 因为放大了最大的概率
  - soft: 因为仍然为较小的 x_i 赋予了一定概率
  - 深度学习中常用

Training a model by optimizing parameters

To train a model, we adjust parameters to minimize a loss

E.g., below, for a simple convex function over two parameters

Contour lines show levels of objective function



通过优化参数训练模型

ShowMeAI

- 训练模型的过程，实际上是我们在调整参数最小化损失函数.
- 如下是一个包含2个参数的凸函数，我们绘制了目标函数的等高线.

扫码，学习本节课程视频 | B站·中英双语字幕

To train the model: Compute all vector gradients!

- Recall: θ represents all model parameters, in one long vector
- In our case with d-dimensional vectors and V-many words:

$$\theta = \begin{bmatrix} v_{aardvark} \\ v_a \\ \vdots \\ v_{zebra} \\ u_{aardvark} \\ u_a \\ \vdots \\ u_{zebra} \end{bmatrix} \in \mathbb{R}^{2dV}$$

- Remember: every word has two vectors
- We optimize these parameters by walking down the gradient

训练模型：计算所有向量梯度

ShowMeAI

- θ代表所有模型参数，写在一个长的参数向量里
- 在我们的场景汇总是d维向量空间的V个词汇

参考资料

- Stanford CS224n slides [点击]
- Stanford CS224n notes [点击]
- Stanford CS224n projects [点击]
- CS224n Assignments reference solution [点击]
- CS224n课程笔记 by @XuXiao [点击]
- CS224n 2017 中英video [点击]

扫码，学习本节课程视频 | B站·中英双语字幕

系列内容 Awesome AI Courses Notes Cheatsheets

是 ShowMeAI 资料库的分支系列，覆盖最具知名度的TOP20+门AI课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

有任何建议和反馈，也欢迎通过右侧渠道和我们联络 (* 3)~

机器学习	Machine Learning	Stanford · CS229
深度学习	Deep Learning	Stanford · CS230
自然语言处理	**Natural Language Processing**	**Stanford · CS224n**
计算机视觉	Computer Vision	Stanford · CS231n
深度强化学习	Deep Reinforcement Learning	UCBerkeley · CS285
知识图谱	Knowledge Graphs	Stanford · CS520
图机器学习	Machine Learning with Graphs	Stanford · CS224W
自动驾驶	DL for Self-Driving Cars	MIT · 6.S094
...	...	...

斯坦福大学(Stanford) *Natural Language Processing with Deep Learning (CS224n)* 课程，是本系列的第三门产出。

课程版本为2019 Winter，核心深度内容(transformer、bert、问答、摘要、文本生成等)在当前(2021年)工业界和研究界依旧是前沿的方法。最新版课程的笔记生产已在规划中，也敬请期待。

笔记内容经由深度加工整合，以**5**个部分构建起完整的“CS224n内容世界”，并依托GitHub创建了汇总页。快扫描二维码，跳转进入吧！

中英字幕 **视频**

课件 动态注释

官方Note **翻译**

作业 代码解析

结业 项目参考

微信公众号

扫码回复”**CS224n**”，下载最新全套资料

回复“**添砖加瓦**”，成为AI内容创作者