



温州大學
WENZHOU UNIVERSITY

深度学习-自然语言处理和词嵌入

黄海广 副教授

2023年05月

本章目录

2

01 词汇表征和文本数据处理

02 词嵌入

03 Word2Vec

04 GloVe

05 GPT

1.词汇表征

3

01 词汇表征和文本数据处理

02 词嵌入

03 **Word2Vec**

04 **GloVe**

05 **GPT**

1. 词汇表征和文本数据处理

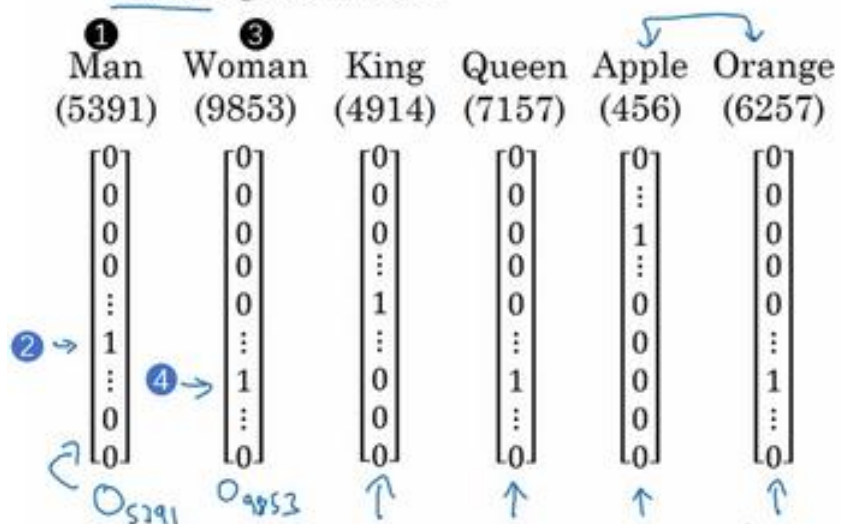
4

Word representation

$V = [a, aaron, \dots, zulu, <UNK>]$

$|V| = 10,000$

1-hot representation



I want a glass of orange juice.

I want a glass of apple _____.

Andrew Ng

1. 词汇表征和文本数据处理

5

Analogy

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.70	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97

① e_{5391}
 e_{man}

② e_{woman}

$\text{Man} \rightarrow \text{Woman} \quad \Leftrightarrow \quad \text{King} \rightarrow ? \text{ Queen}$

$e_{\text{man}} - e_{\text{woman}} \approx e_{\text{king}} - e_{?}$

$e_{\text{man}} - e_{\text{woman}} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

$e_{\text{king}} - e_{\text{queen}} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

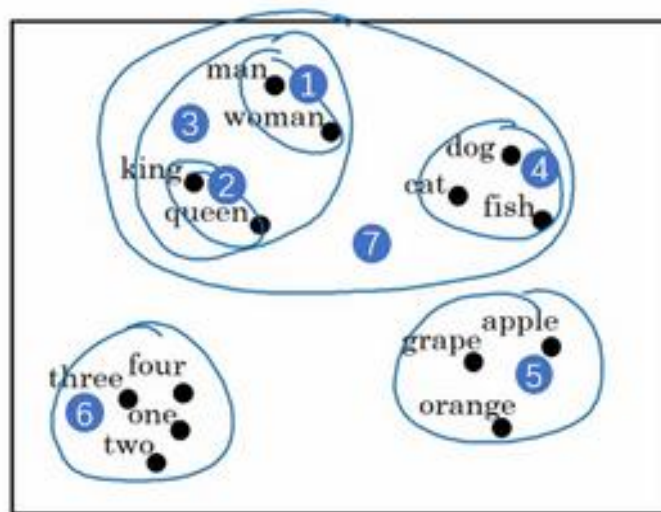
[Mikolov et. al., 2013, Linguistic regularities in continuous space word representations]

Andrew Ng

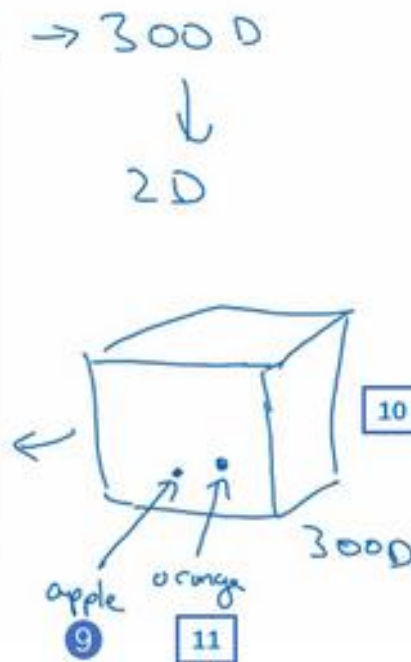
1.词汇表征和文本数据处理

6

Visualizing word embeddings

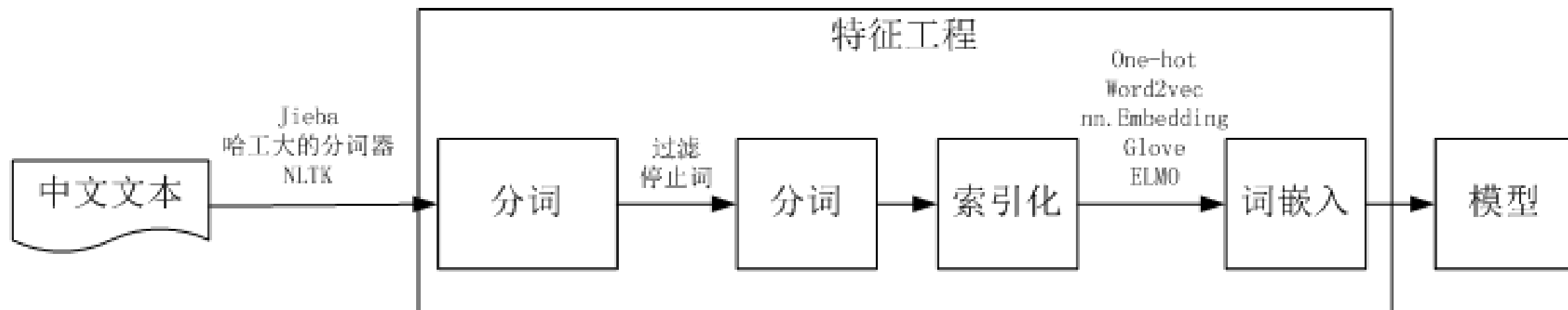


t-SNE



1.词汇表征和文本数据处理

7



2.词嵌入

8

01 词汇表征和文本数据处理

02 词嵌入

03 Word2Vec

04 GloVe

05 GPT

9

“Robert Lin is an apple farmer.”

1 1

0 0 0 0 0 0

Sally Johnson is an orange farmer

Robert Lin is an a

BRNN

1B words - 100B words

look words

Andrew Ng

2.词嵌入

10

如何用词嵌入做迁移学习的步骤。

第一步，先从大量的文本集中学习词嵌入。

第二步，你可以用这些词嵌入模型把它迁移到你的新的只有少量标注训练集的任务中，比如说用这个300维的词嵌入来表示你的单词。这样做的一个好处就是你可以用更低维度的特征向量代替原来的10000维的**one-hot**向量，现在你可以用一个300维更加紧凑的向量。

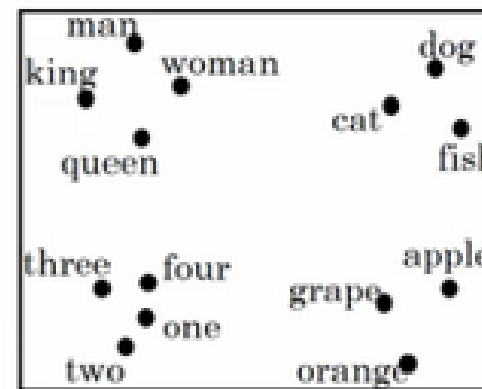
第三步，当你在你新的任务上训练模型时，在你的命名实体识别任务上，只有少量的标记数据集上，你可以自己选择要不要继续微调，用新的数据调整词嵌入。

2.词嵌入

11

Analogies

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.70	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97



$$e_{\text{man}} - e_{\text{woman}} = \begin{bmatrix} -1 \\ 0.01 \\ 0.03 \\ 0.09 \end{bmatrix} - \begin{bmatrix} 1 \\ 0.02 \\ 0.02 \\ 0.01 \end{bmatrix} = \begin{bmatrix} -2 \\ -0.01 \\ 0.01 \\ 0.08 \end{bmatrix} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$e_{\text{king}} - e_{\text{queen}} = \begin{bmatrix} -0.95 \\ 0.93 \\ 0.70 \\ 0.02 \end{bmatrix} - \begin{bmatrix} 0.97 \\ 0.95 \\ 0.69 \\ 0.01 \end{bmatrix} = \begin{bmatrix} -1.92 \\ -0.02 \\ 0.01 \\ 0.01 \end{bmatrix} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

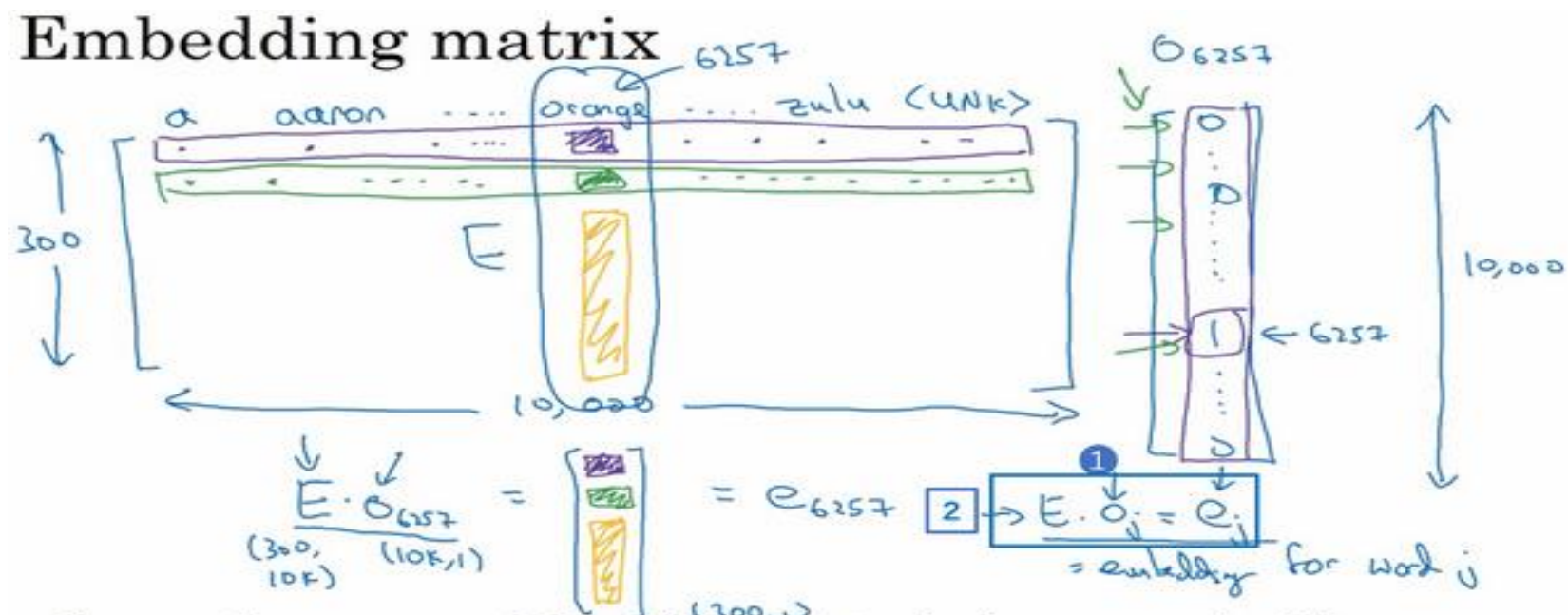
类似的，假如你用 e_{king} 减去 e_{queen} ，最后也会得到一样的结果

这个结果表示，**man**和**woman**主要的差异是**gender**（性别）上的差异

2.词嵌入

12

嵌入矩阵



In practice, use specialized function to look up an embedding.
 $\rightarrow \text{Embedding}$

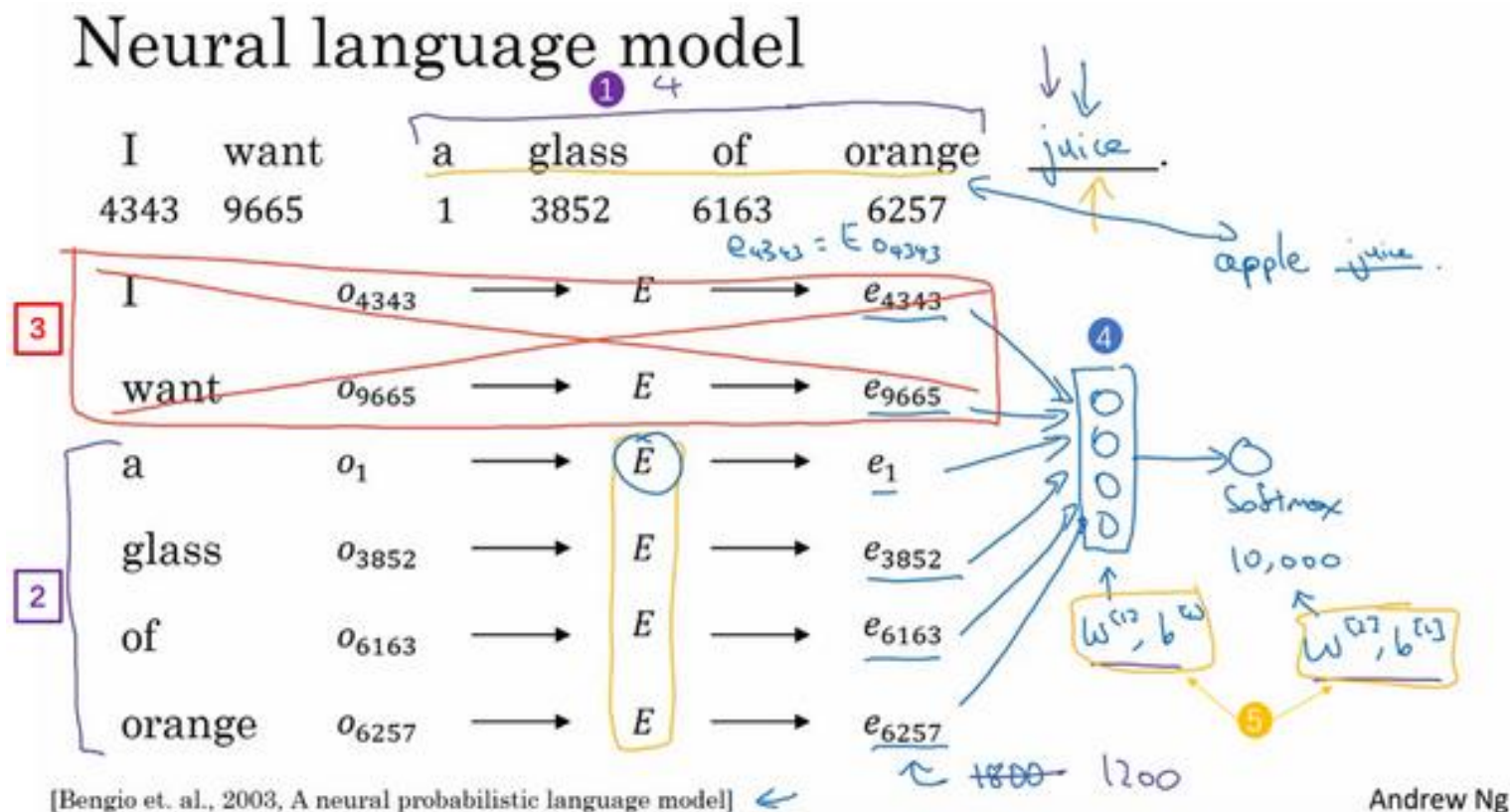
Andrew Ng

e_w = embedding for word w

2.词嵌入

13

嵌入矩阵



3.Word2Vec

14

01 词汇表征和文本数据处理

02 词嵌入

03 Word2Vec

04 GloVe

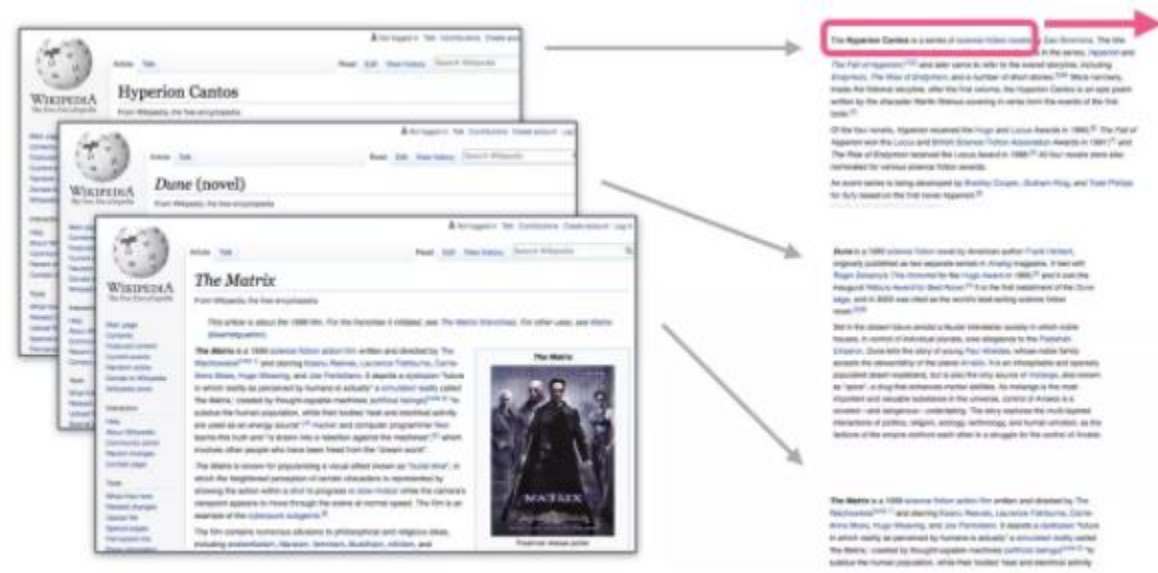
05 GPT

3.Word2Vec

15

语言模型的训练机制就是这样

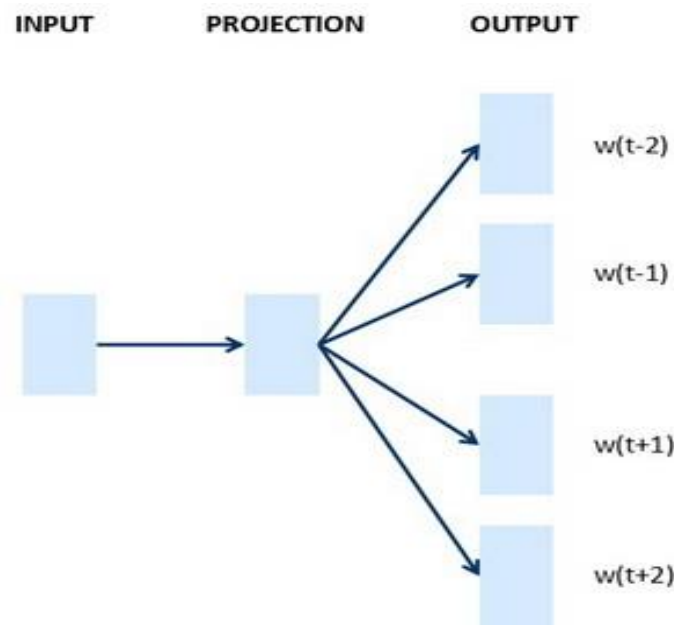
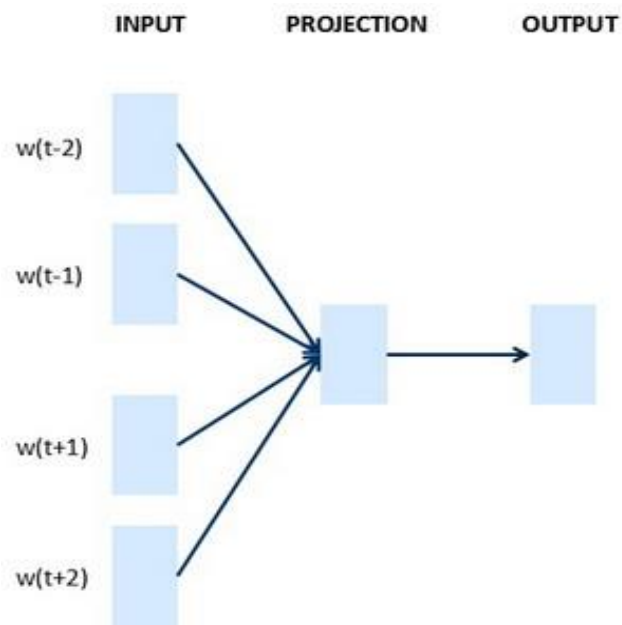
- 1.我们获得了大量文本数据（例如，所有维基百科文章）。然后
- 2.我们有一个窗口（比如说三个单词），我们会对所有文本进行滑动。
- 3.滑动窗口为我们的模型生成训练样本



3.Word2Vec

16

(下图左边为CBOW，右边为Skip-Gram)



CBOW对小型数据库比较合适，而**Skip-Gram**在大型语料中表现更好。

3.Word2Vec

17

Jay was hit by a _____ bus in...

by	a	red	bus	in
----	---	-----	-----	----

我们实际构建和训练模型的数据集将如下所示：

input 1	input 2	input 3	input 4	output
by	a	bus	in	red

这被称为连续词袋结构，并在word2vec论文 one of the word2vec papers 中进行过描述。

3.Word2Vec

18

负采样

计算的角度来看，SkipGram非常消耗资源：尤其是我们将在数据集中为每个训练样本做一次（很可能数千万次）。我们需要做一些事情来提高效率。

一种方法是将目标分成两个步骤：

- 1.生成高质量的单词嵌入（不要担心下一个单词预测）。
- 2.使用这些高质量的嵌入来训练语言模型（进行下一个单词预测）。

3.Word2Vec

19

负采样

并不是每次迭代都训练全部10,000个，我们只训练其中的5个，我们要训练对应真正目标词那一个分类器，再训练4个随机选取的负样本，这就是 $K = 4$ 的情况。所以不使用一个巨大的10,000维度的**softmax**，因为计算成本很高，而是把它转变为10,000个二分类问题，每个都很容易计算，每次迭代我们要做的只是训练它们其中的5个，一般而言就是 $K + 1$ 个，其中 K 个负样本和1个正样本。这也是为什么这个算法计算成本更低，因为只需更新 $K + 1$ 个逻辑单元， $K + 1$ 个二分类问题，相对而言每次迭代的成本比更新10,000维的**softmax**分类器成本低。

Selecting negative examples

context	word	target?
orange	juice	1
orange	king	0
orange	book	0
orange	the	0
orange	of	0

the, of, and, ...

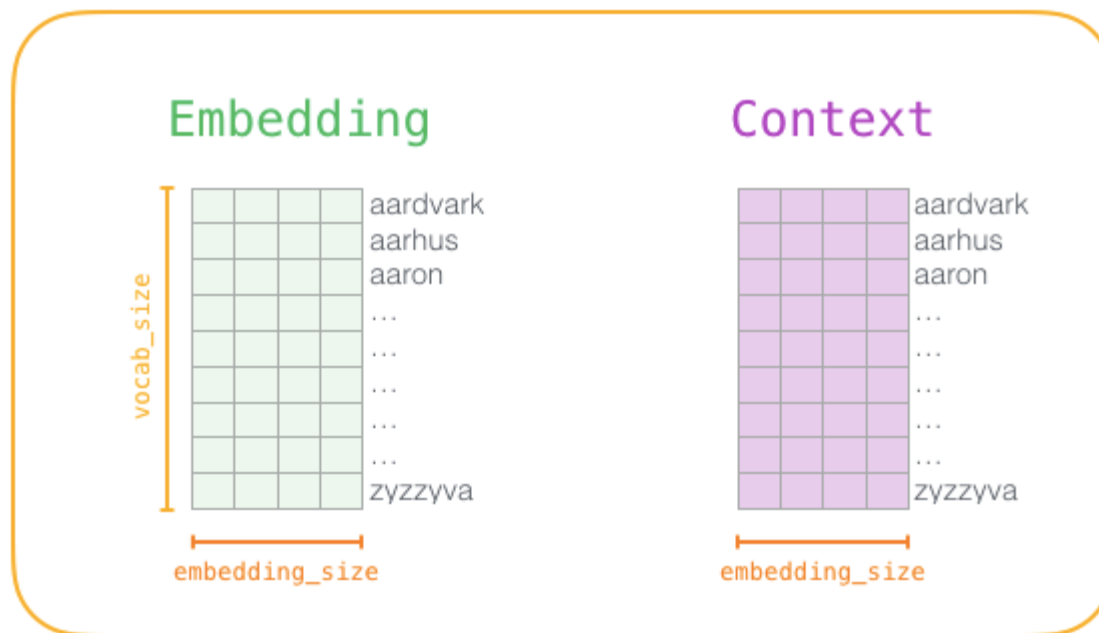
$$P(w_i) = \frac{f(w_i)^{\frac{3}{4}}}{\sum_{j=1}^{10,000} f(w_j)^{\frac{3}{4}}}$$

3.Word2Vec

20

训练流程

在训练过程开始之前，我们预先处理我们正在训练模型的文本。在这一步中，我们确定词汇量的大小（我们称之为`vocab_size`，比如说，将其视为10,000）以及哪些词属于它。在训练阶段的开始，我们创建两个矩阵 - `Embedding`矩阵和`Context`矩阵。这两个矩阵在我们的词汇表中嵌入了每个单词（这`vocab_size`是他们的维度之一）。第二个维度是我们希望每次嵌入的时间长度（`embedding_size`- 300是一个常见值）。



3.Word2Vec

21

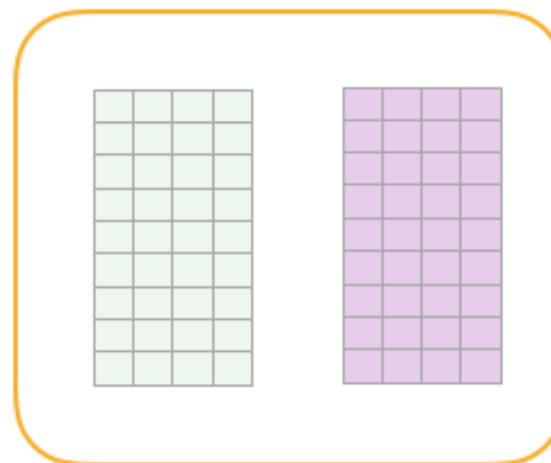
训练流程

在训练过程开始时，我们用随机值初始化这些矩阵。然后我们开始训练过程。在每个训练步骤中，我们采取一个正样本及其相关的负样本。我们来看看我们的第一组：

dataset

input word	output word	target
not	thou	1
not	aaron	0
not	taco	0
not	shalt	1
not	mango	0
not	finglonger	0
not	make	1
not	plumbus	0
...

model

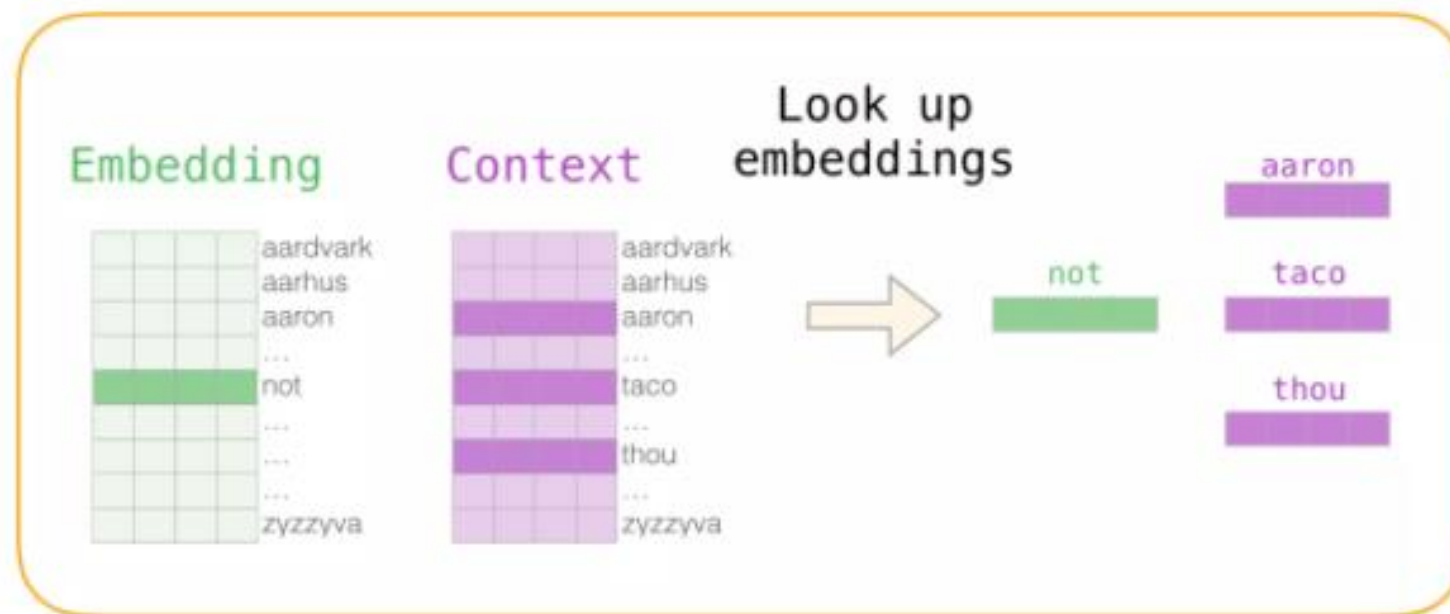


3.Word2Vec

22

训练流程







现在我们有四个单词：输入单词`not`和输出/上下文单词:(`thou`实际邻居),`aaron`, 和`taco` (负样本)。我们继续查找它们的嵌入 - 对于输入词, 我们查看`Embedding`矩阵。对于上下文单词, 我们查看`Context`矩阵 (即使两个矩阵都在我们的词汇表中嵌入了每个单词)。









3.Word2Vec

23

训练流程 然后，我们计算输入嵌入与每个上下文嵌入的点积。在每种情况下，会产生一个数字，该数字表示输入和上下文嵌入的相似性。

input word	output word	target	input • output
not 	thou 	1	0.2
not 	aaron 	0	-1.11
not 	taco 	0	0.74

现在我们需要一种方法将这些分数转化为看起来像概率的东西：
使用sigmoid函数把概率转换为0和1。







input word	output word	target	input • output	sigmoid()
not 	thou 	1	0.2	0.55
not 	aaron 	0	-1.11	0.25
not 	taco 	0	0.74	0.68

3.Word2Vec

24

训练流程

现在我们可以将sigmoid操作的输出视为这些样本的模型输出。您可以看到taco得分最高aaron，并且在sigmoid操作之前和之后仍然具有最低分。既然未经训练的模型已做出预测，并且看到我们有一个实际的目标标签要比较，那么让我们计算模型预测中的误差。为此，我们只从目标标签中减去sigmoid分数

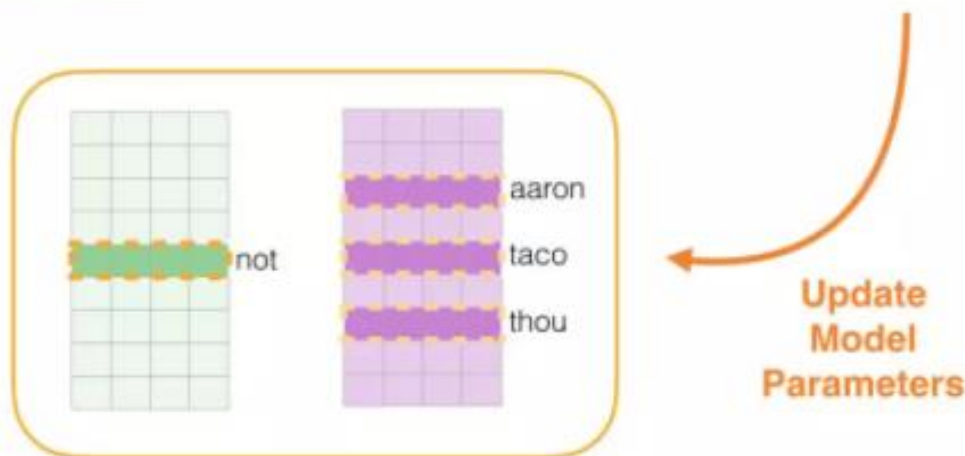
input word	output word	target	input • output	sigmoid()	Error
not 	thou 	1	0.2	0.55	0.45
not 	aaron 	0	-1.11	0.25	-0.25
not 	taco 	0	0.74	0.68	-0.68

3.Word2Vec

25

训练流程 这是“机器学习”的“学习”部分。现在，我们可以利用这个误差分数调整`not`，`thou`，`aaron`和`taco`的嵌入,使下一次我们做出这一计算，结果会更接近目标分数。

input word	output word	target	input • output	sigmoid()	Error
not	thou	1	0.2	0.55	0.45
not	aaron	0	-1.11	0.25	-0.25
not	taco	0	0.74	0.68	-0.68



3.Word2Vec

26

训练流程 训练步骤到此结束。我们从这一步骤中得到稍微好一点的嵌入（`not`，`thou`，`aaron`和`taco`）。我们现在进行下一步（下一个正样本及其相关的负样本），并再次执行相同的过程。

dataset			model	
input word	output word	target		
not	thou	1		
not	aaron	0		
not	taco	0		
not	shalt	1		
not	mango	0		
not	finglonger	0		
not	make	1		
not	plumbus	0		
...		

当我们循环遍历整个数据集多次时，嵌入继续得到改进。然后我们可以停止训练过程，丢弃`Context`矩阵，并使用`Embeddings`矩阵作为下一个任务的预训练嵌入。

4.GloVe

27

01 词汇表征和文本数据处理

02 词嵌入

03 Word2Vec

04 GloVe

05 GPT

4.GloVe

28

GloVe代表用词表示的全局变量 (**global vectors for word representation**)。

对于**GloVe**算法，我们可以定义上下文和目标词为任意两个位置相近的单词，假设是左右各10词的距离，那么 X_{ij} 就是一个能够获取单词 i 和单词 j 出现位置相近时或是彼此接近的频率的计数器。

GloVe模型做的就是进行优化，我们将他们之间的差距进行最小化处理：

$$\text{minimize } \sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f(X_{ij}) (\theta_i^T e_j + b_i + b'_j - \log X_{ij})^2$$

5.情感分类

29

01 词汇表征和文本数据处理

02 词嵌入

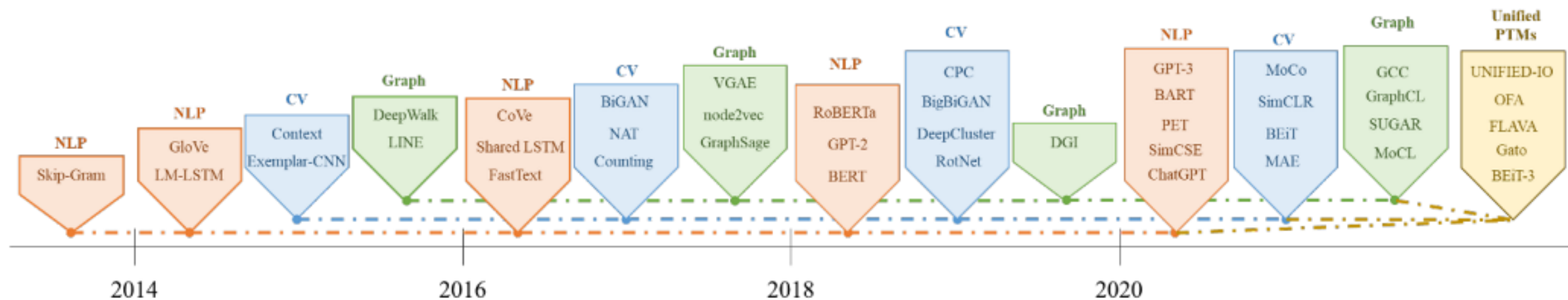
03 Word2Vec

04 GloVe

05 GPT

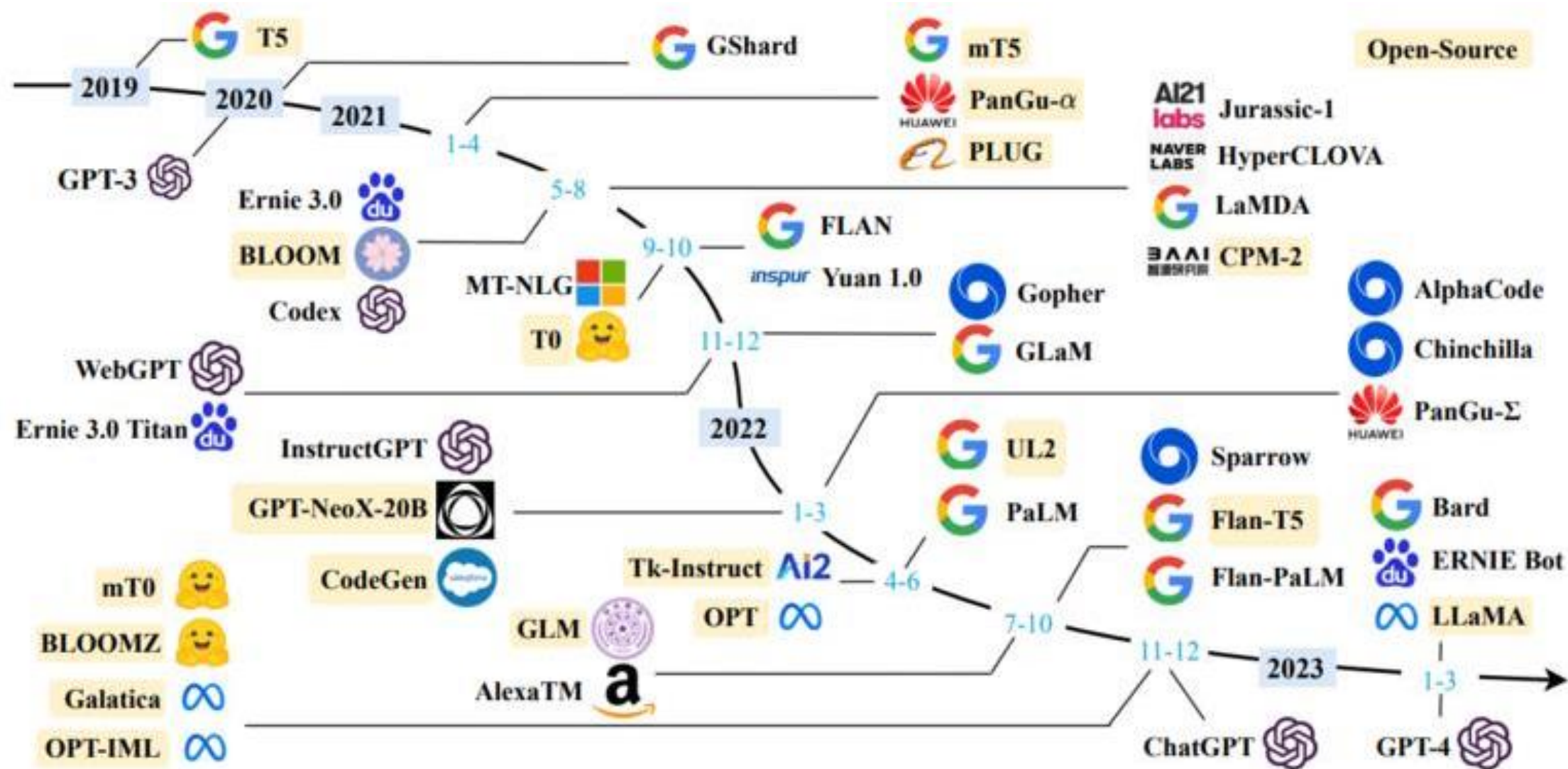
预训练模型的发展

30



预训练模型的发展

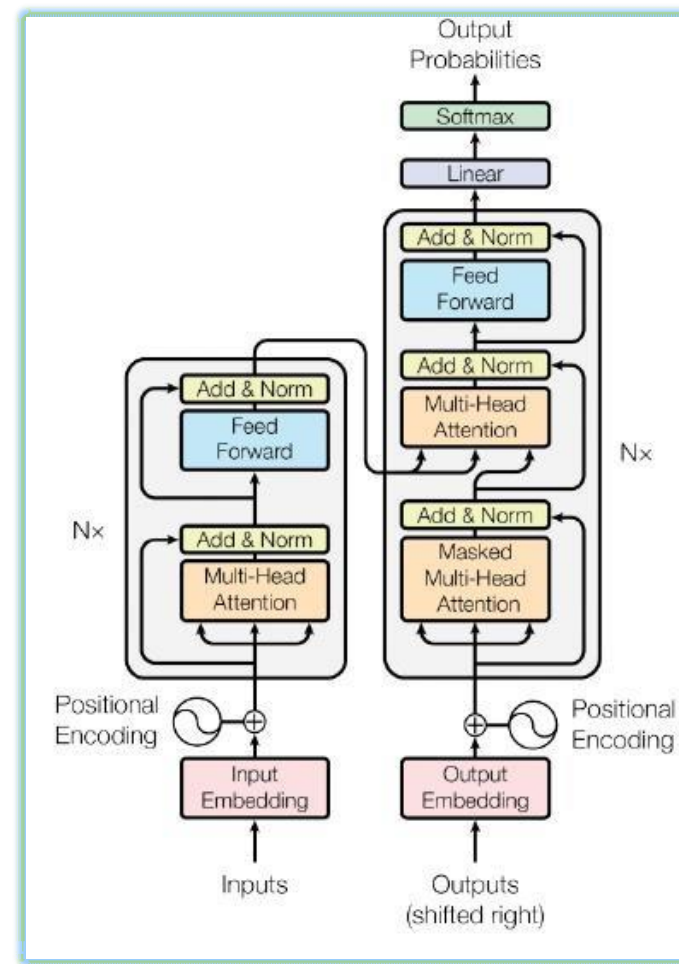
31



Transformer

32

- Transformer摆脱了人工标注数据集的缺陷，模型在质量上更优、更易于并行化，所需训练时间明显更少
- Transformer通过成功地将其应用于具有大量和有限训练数据的分析，可以很好地推广到其他任务
- ✓ 2017年，在Ashish Vaswani et.al 的论文《Attention Is All You Need》中，考虑到主导序列转导模型基于编码器-解码器配置中的复杂递归或卷积神经网络，性能最好的模型被证明还是通过注意力机制（attention mechanism）连接编码器和解码器，因而《Attention Is All You Need》中提出了一种新的简单架构——Transformer，它完全基于注意力机制，完全不用重复和卷积，因而这些模型在质量上更优，同时更易于并行化，并且需要的训练时间明显更少。
- ✓ Transformer出现以后，迅速取代了RNN系列变种，跻身主流模型架构基础。（RNN缺陷正在于流水线式的顺序计算）



图：Transformer模型架构

资料来源：《Attention Is All You Need》, Ashish Vaswani et.al 2017

Transformer

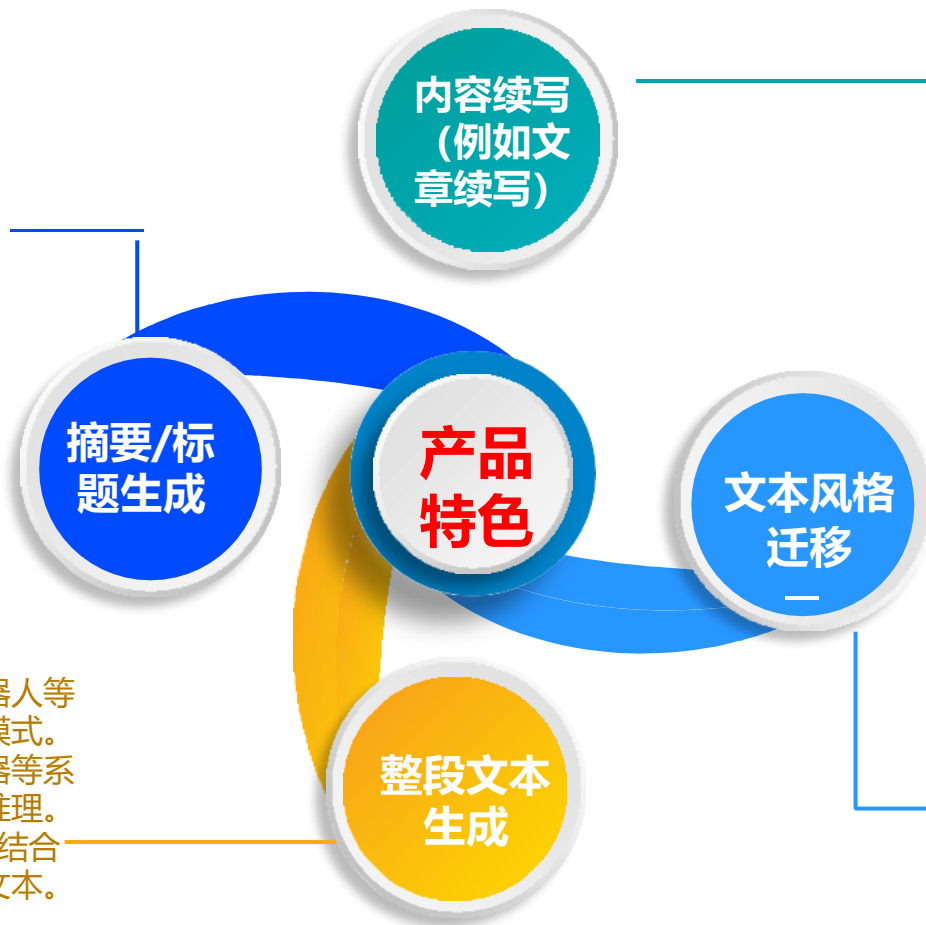
33

- Transformer架构可分为自回归系列（例如GPT-3，偏好生成性任务）、双向Transformer+Mask的自编码系列（例如BERT，偏好自然语言理解）、Encoder-decoder架构（例如T5，使用双向/单向attention，偏好条件文本生成）

图：Transformer典型技术场景下的原理介绍如下所述

首先通过词嵌入(Word Embedding)将字、词、句进行区分，然后基于特征评分、序列标注、分类模型等提取内容特征计算相关文本单元权重其次选择相应的文本单元子集组成摘要候选集，完成内容选择，最后针对字数要求等限定条件，对候选集的内容进行整理形成最终摘要，完成内容组织。其细分路径又包含生成式文本摘要(AATS)，即形成抽象认知并创造新词灵活概括，和抽取式文本摘要(EATS)，即直接抽取原始素材并拼接成简单概要

对话式文本生成适用于智能客服等任务型和闲聊型机器人等非任务型人机交互场景，可分类为管道模式及端对端模式。结构性的文本生成，首先通过注意力机制、多层感知器等系统进行语句内容预选，对数值、时间等类型数据进行推理。增强数据间的结构信息。其次通过Transformer等模式结合上下文进行推导，生成最终文本。



通过随机Mask(即遮挡)数据库文本中的词语或语段，让神经网络自主学习复原被遮挡部分，从而拥有“猜测”缺失内容的能力，产出预训练模型。再通过大规模预训练模型理解上文或给定条件，从概率层面推测最符合要求的输出结果。其本质是借助超大规模的训练参数猜测上下文的过程

主流思路是分离文本属性及文本内容

隐式方法即使用某类无监督学习或强化学习模式将文本属性及内容自动分离，常见的有生成对抗方式，即通过GAN实现目标属性和文本量性完全由不同的编码控制的状态。

GPT的发展

34

GPT-1：借助预训练，进行无监督训练和有监督微调

- GPT-1模型基于Transformer解除了顺序关联和依赖性的前提，采用生成式模型方式，重点考虑了从原始文本中有效学习的能力，这对于减轻自然语言处理（NLP）中对监督学习的依赖至关重要
- ✓ GPT（Generative Pre-training Transformer）于2018年6月由OpenAI首次提出。GPT模型考虑到在自然语言理解中有大量不同的任务，尽管大量的未标记文本语料库非常丰富，但用于学习这些特定任务的标记数据却很少，这使得经过区分训练的模型很难充分执行。同时，大多数深度学习方法需要大量手动标记的数据，这限制了它们在许多缺少注释资源的领域的适用性。
- ✓ 在考虑以上局限性的前提下，GPT论文中证明，通过对未标记文本的不同语料库进行语言模型的**生成性预训练**，然后对每个特定任务进行区分性微调，可以实现这些任务上的巨大收益。和之前方法不同，GPT在微调期间使用任务感知输入转换，以实现有效的传输，同时对模型架构的更改最小。

图：GPT-1模型的核心手段是预训练（Pre-training）

3.1 Unsupervised pre-training

Given an unsupervised corpus of tokens $\mathcal{U} = (u_1, \dots, u_n)$, we use a standard language modeling objective to maximize the following likelihood:

$$L_{\mathcal{U}}(\theta) = \sum_{i=1}^n \log P(u_i | u_{-i}; \theta) \quad (1)$$

where k is the size of the context window, and the conditional probability P is modeled using a neural network with parameters θ . These parameters are trained using stochastic gradient descent [51].

In our experiments, we use a multi-layer *Transformer decoder* [34] for the language model, which is a variant of the transformer [62]. This model applies a multi-headed self-attention operation over the input context tokens followed by position-wise feedforward layers to produce an output distribution over target tokens:

$$h_0 = U W_e + W_p \\ h_i = \text{transformer_block}(h_{i-1}) \quad \forall i \in [1, n] \quad (2)$$

$$P(u_i) = \text{softmax}(h_n W_v^T)$$

where $U = (u_{-k}, \dots, u_{-1})$ is the context vector of tokens, n is the number of layers, W_e is the token embedding matrix, and W_p is the position embedding matrix.

无监督预训练
(Unsupervised pre-training)

不需要标注数据集，即大规模自学阶段，在保证AI算力充足的条件下，根据attention机制进行自学



有监督微调
(Supervised fine-tuning)

微调，用来修正模型理解力。即小规模指导过程，让AI在小样本数据下进行调整

3.2 Supervised fine-tuning

After training the model with the objective in Eq. 1, we adapt the parameters to the supervised target task. We assume a labeled dataset \mathcal{V} , where each instance consists of a sequence of input tokens, x^1, \dots, x^n , along with a label y . The inputs are passed through our pre-trained model to obtain the final transformer block's activations h^n , which is then fed into an added linear output layer with parameters W_y to predict y :

$$P(y | x^1, \dots, x^n) = \text{softmax}(h^n W_y) \quad (3)$$

This gives us the following objective to maximize:

$$L_2(\theta) = \sum_{(x,y) \in \mathcal{V}} \log P(y | x^1, \dots, x^n) \quad (4)$$

We additionally found that including language modeling is an auxiliary objective to the fine-tuning helped learning by (a) improving generalization of the supervised model, and (b) accelerating convergence. This is in line with prior work [50, 43], who also observed improved performance with such an auxiliary objective. Specifically, we optimize the following objective (with weight λ):

$$L_{\mathcal{V}}(\theta) = L_2(\theta) + \lambda \cdot L_{\mathcal{U}}(\theta) \quad (5)$$

Overall, the only extra parameters we require during fine-tuning are W_y and embeddings for delimiter tokens (described below in Section 3.3).

结合形成了一种使用无监督预训练和有监督微调相结合的语言理解任务的“半监督方法”

GPT的发展

35

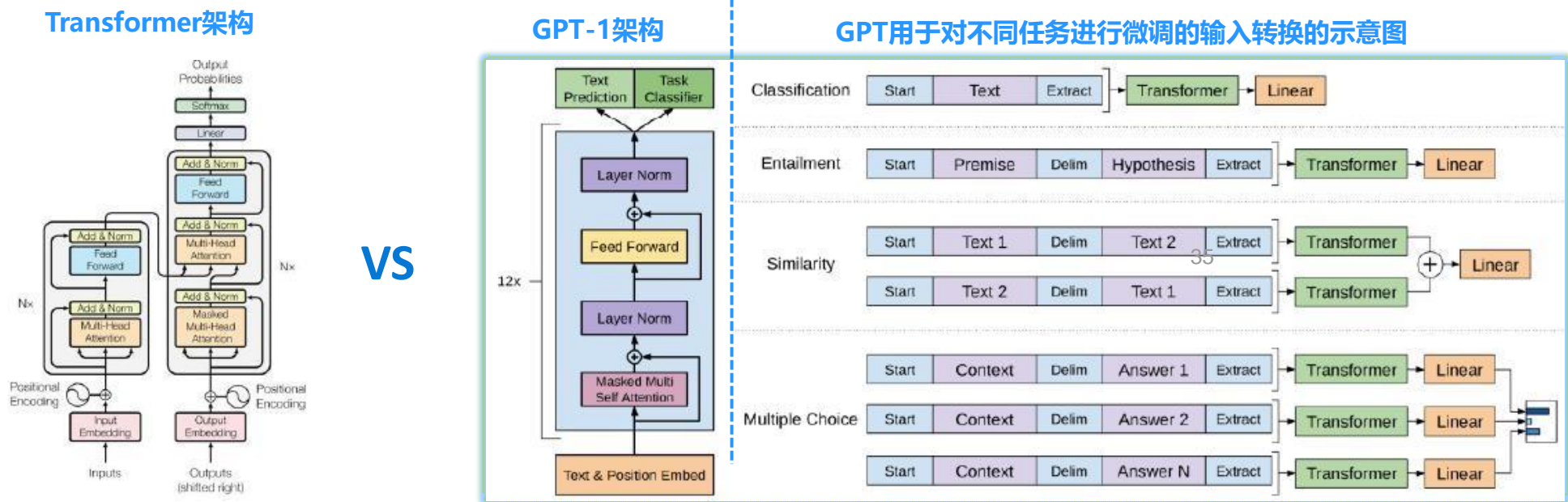
GPT-1: 模型更简化、计算加速, 更适合自然语言生成任务 (NLG)

■ GPT1相比于Transformer等模型进行了显著简化

- ✓ 相比于Transformer, GPT训练了一个12层**仅decoder的解码器** (原Transformer模型中包含Encoder和Decoder两部分)。
- ✓ 相比于Google的BERT(Bidirectional Encoder Representations from Transformers,双向编码生成Transformer), **GPT仅采用上文预测单词** (BERT采用了基于上下文双向的预测手段)。

注: ChatGPT的表现更贴近人类意图, 部分因为一开始GPT是基于上文的预测, 这更贴近人类的话语模式, 因为人类言语无法基于将来的话来做分析。

图: GPT-1模型相比于Transformer模型有了显著简化



GPT的发展

36

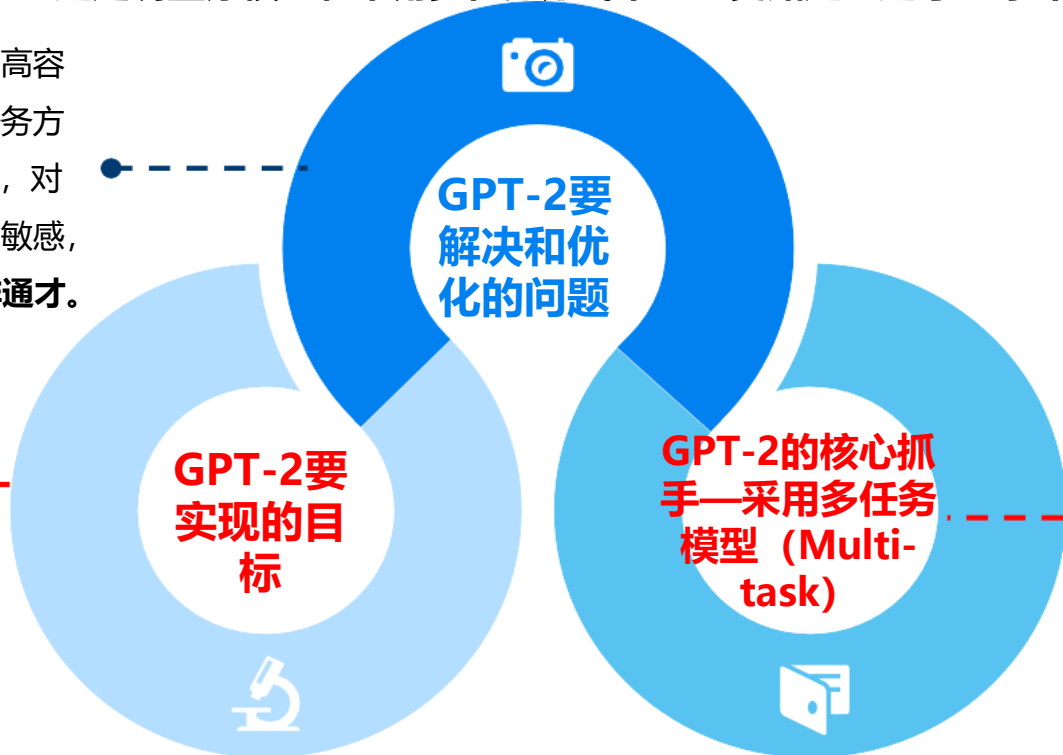
■ GPT-2 (2019.2) 在GPT-1的基础上进行诸多改进，实现执行任务多样性，开始学习在不需要明确监督的情况下执行数量惊人的任务

- ✓ 在GPT-2阶段，OpenAI去掉了GPT-1阶段的有监督微调（fine-tuning），成为**无监督模型**。
- ✓ 大模型GPT-2是一个1.5B参数的Transformer，在其相关论文中它在8个测试语言建模数据集中的7个数据集上实现了当时最先进的结果。模型中，Transformer堆叠至48层。GPT-2的数据集增加到8 million的网页、大小40GB的文本。

图：GPT-2通过调整原模型和采用多任务方式来让AI更贴近“通才”水平

- ✓ 机器学习系统通过使用大型数据集、高容量模型和监督学习的组合，在训练任务方面表现出色，然而这些系统较为脆弱，对数据分布和任务规范的轻微变化非常敏感，因而使得AI表现更像狭义专家，并非通才。

- ✓ 转向更通用的系统，使其可以执行许多任务，最终无需为每个任务手动创建和标记训练数据集。



- **GPT-2 调整优化的目的**是为了解决零次学习问题（zero-shot）（注：zero-shot问题，就是针对AI在面对不认识的事物时，也能进行推理）
- **多任务模型的特点**：跟传统ML需要专门的标注数据集不同（从而训练出专业AI），多任务模型不采用专门AI手段，而是在海量数据喂养训练的基础上，适配任何任务形式。

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

GPT的发展

37

■ GPT-2聚焦在无监督、zero-shot（零次学习）上，然而GPT-2训练结果也有不达预期之处，所存在的问题也亟待优化

- ✓ 在GPT-2阶段，尽管体系结构是任务无关的，但仍然需要任务特定的数据集和任务特定的微调：要在所需任务上实现强大的性能，通常需要对特定于该任务的数千到数十万个示例的数据集进行微调。

图：GPT-2尚未解决诸多瓶颈问题



GPT的发展

38

GPT-3（2020.5）取得突破性进展，任务结果难以与人类作品区分开来

■ GPT-3对GPT-2追求无监督与零次学习的特征进行了改进

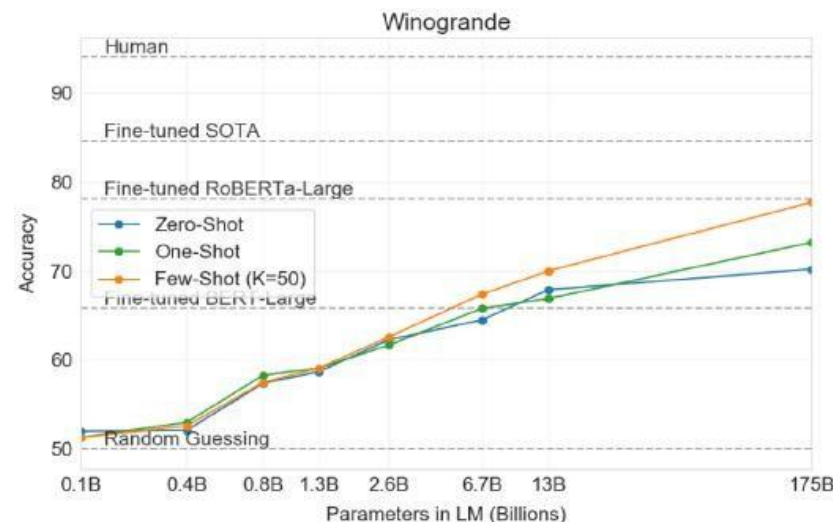
■ GPT-3利用了过滤前45TB的压缩文本，在诸多NLP数据集中实现了强大性能

- ✓ GPT-3是一个具有**1750亿个参数**的自回归语言模型，比之前的任何非稀疏语言模型多10倍。对于所有任务（在few-shot设置下测试其性能），GPT-3都是在没有任何梯度更新或微调的情况下应用的，仅通过与模型的文本交互来指定任务和few-shot演示。
- ✓ GPT-3在许多NLP数据集上都有很强的性能（包括翻译、问题解答和完形填空任务），以及一些需要动态推理或领域适应的任务（如解释单词、在句子中使用一个新单词或执行三位数算术）。GPT-3可以生成新闻文章样本（已很难将其与人类撰写的文章区分开来）。

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

图：GPT-3的模型参数在GPT-2的基础上增加110多倍

资料来源：《Language Models are Few-Shot Learners》



图：GPT-3相关研究显示，few-shot(少量样本)的综合表现是在无监督模式下最优的

GPT的发展

■ InstructGPT使用来自人类反馈的强化学习方案RLHF（reinforcement learning from human feedback），通过对大语言模型进行微调，从而能够在参数减少的情况下，实现优于GPT-3的功能

- ✓ InstructGPT提出的背景：使语言模型更大并不意味着它们能够更好地遵循用户的意图，例如大型语言模型可以生成不真实、有毒或对用户毫无帮助的输出，即这些模型与其用户不一致。另外，GPT-3虽然选择了少样本学习（few-shot）和继续坚持了GPT-2的无监督学习，但基于few-shot的效果，其稍逊于监督微调（fine-tuning）的方式。
- ✓ 基于以上背景，OpenAI在GPT-3基础上根据人类反馈的强化学习方案RHLF，训练出奖励模型（reward model）去训练学习模型（即用AI训练AI的思路）
- ✓ InstructGPT的训练步骤为：对GPT-3监督微调——训练奖励模型（reward model）——增强学习优化SFT（第二、第三步可以迭代循环多次）

SFT Data			RM Data			PPO Data		
split	source	size	split	source	size	split	source	size
train	labeler	11,295	train	labeler	6,623	train	customer	31,144
train	customer	1,430	train	customer	26,584	valid	customer	16,185
valid	labeler	1,550	valid	labeler	3,488			
valid	customer	103	valid	customer	14,399			

图：InstructGPT训练三步骤各自对应的数据集规模如下图所示（labeler指的是OpenAI的标注人员，customer指GPT-3 API的用户）

资料来源：《Training language models to follow instructions with human feedback》论文

ChatGPT核心技术优势

40

- InstructGPT与ChatGPT属于相同代际的模型，ChatGPT只是在InstructGPT的基础上增加了Chat属性，且开放了公众测试
- ChatGPT提升了理解人类思维的准确性的原因在于利用了基于人类反馈数据的系统进行模型训练

(注：根据官网介绍，GhatGPT也是基于InstructGPT构建，因而可以从InstructGPT来理解ChatGPT利用人类意图来增强模型效果)

图：基于人类反馈强化的核心训练流程如下所示：

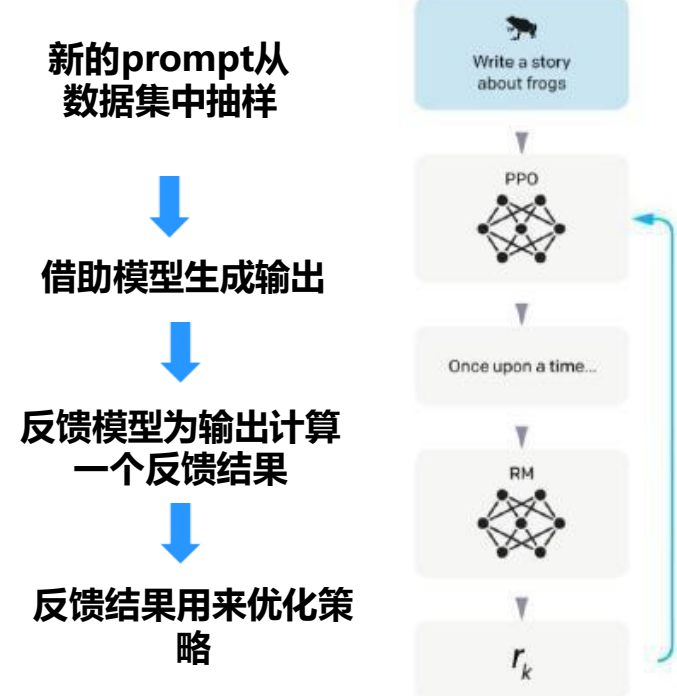
步骤1：搜集说明数据，训练监督策略



步骤2：搜集比较数据，训练一个奖励模型



步骤3：搜集说明数据，使用增强学习优化模型



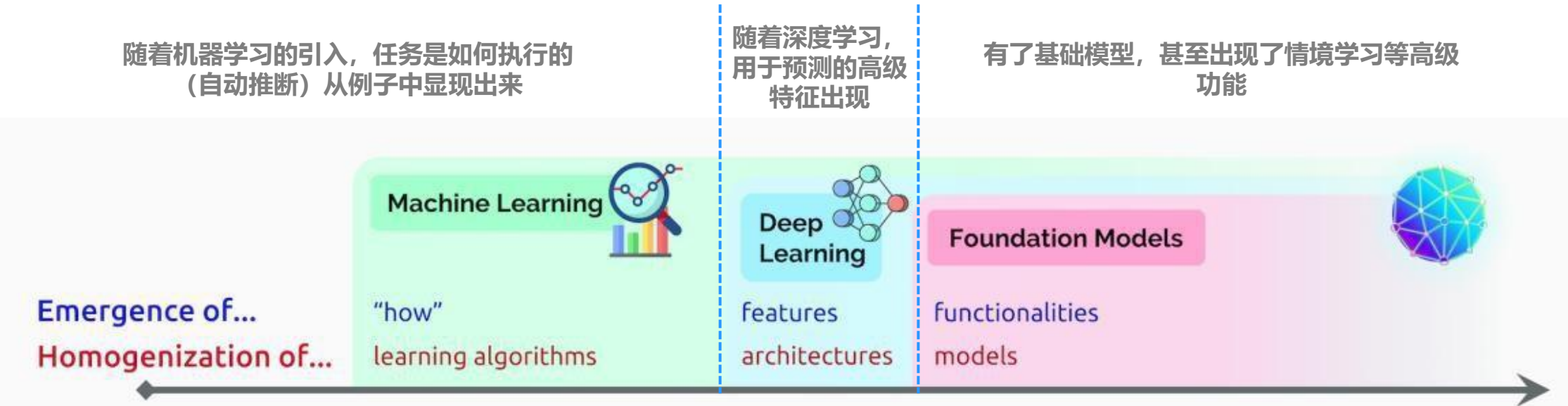
ChatGPT核心技术优势

41

ChatGPT得益于通用（基础）模型所构建 AI 系统的新范式

- **基础模型（Foundation Model）** 在广泛的应用中整合构建机器学习系统的方法，它为许多任务提供了强大的杠杆作用
- ✓ 基础模型是在深度神经网络和自我监督学习的基础上演化而来。基础模型基于广泛数据（通常使用大规模自我监督）训练的任何模型，可以适应（例如微调）广泛的下游任务，目前例子包括BERT（Devlin et al.）、GPT-3（Brown et al. 2020）和CLIP（Radford et al. 2021）。
- ✓ 机器学习使学习算法同质化（例如，逻辑回归），深度学习使模型架构同质化（如卷积神经网络），而**基础模型使模型本身同质化**（比如，**GPT-3**）。

图37：人工智能的发展呈现同质化的过程



资料来源：《On the Opportunities and Risks of Foundation Models》论文

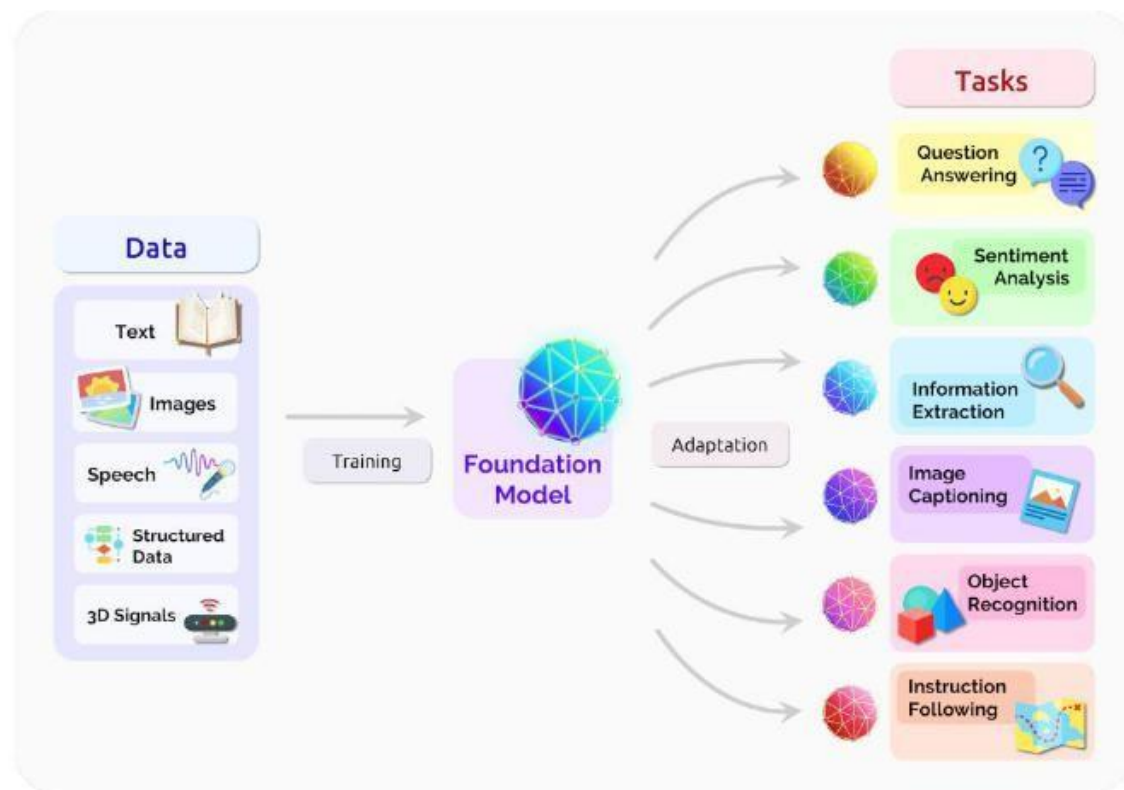
ChatGPT核心技术优势

42

ChatGPT以基础模型为杠杆，可适用多类下游任务

- ChatGPT采用了GPT3.5 (InstructGPT) 大规模预训练模型，在自然语言理解和作品生成上取得极大性能提升
- ✓ 鉴于传统NLP技术的局限问题，基于大语言模型（LLM）有助于充分利用海量无标注文本预训练，从而文本大模型在较小的数据集和零数据集场景下可以有较好的理解和生成能力。基于大模型的**无标准文本书收集**，ChatGPT得以在情感分析、信息钻取、理解阅读等文本场景中优势突出。
- ✓ 随着训练模型数据量的增加，数据种类逐步丰富，模型规模以及参数量的增加，会进一步促进模型语义理解能力以及抽象学习能力的极大提升，实现ChatGPT的**数据飞轮效应**（用更多数据可以训练出更好的模型，吸引更多用户，从而产生更多用户数据用于训练，形成良性循环）。
- ✓ 研究发现，每增加参数都带来了文本合成和/或下游NLP任务的改进，有证据表明，日志丢失与许多下游任务密切相关，随着规模的增长，日志丢失呈现平稳的改善趋势。

图：基础模型可以集中来自各种模态的所有数据的信息，然后这一模型可以适用于广泛的下游任务



资料来源：《On the Opportunities and Risks of Foundation Models》论文

1. IAN GOODFELLOW等, 《深度学习》, 人民邮电出版社, 2017
2. Andrew Ng, <http://www.deeplearning.ai>
3. <https://twitter.com/jalammar>

谢谢!

