第3章　语言模型

炼辞得奇句，炼意得余味。

——北宋哲学家、易学家　邵雍《论诗吟》

学习目标

- ❑ 了解语言模型的发展脉络。
- ❑ 掌握语言模型的参数估计和评价方法。
- ❑ 学习神经网络语言模型的前沿方法。

在自然语言处理中，一个重要的问题是如何判断文字序列是否是人们在日常生活中会表达出来的可理解的句子。例如，文字序列“研究表明，汉字顺序并不影响阅读”是一个正常的句子；而文字序列“研表究明，汉字的序顺并不定一能影阅响读”无论怎样理解都难以讲通，更无法准确解读出其中的意思，进而进行有效的沟通了。理论上，人类的语言是无穷无尽的，而语言模型的本质是在回答一个问题：“出现的语句是否合理?”或“如何度量一个语句的合理性?”本章我们将学习如何从数学上度量自然语言句子的合理程度。

3.1　语言模型概述

语言模型（language model）根据语言客观事实对语言进行抽象数学建模。语言模型在诸如语音识别（speech recognition）、机器翻译（Machine Translation，MT）等任务中起到关键的作用。其发展经历了专家语法规则模型、统计语言模型、神经网络语言模型。

（1）专家语法规则模型。该模型主要流行于 1980 年以前，是领域专家在计算机初始阶段随着编程语言的发展而归纳出的针对自然语言的语法规则。但是自然语言本身的多样性、口语化，在时间、空间上的演化，导致语法规则急剧膨胀而不可持续。

（2）统计语言模型。该模型在 1980—2000 年被广泛应用于自然语言处理中，是一种用来描述自然语言中词序列的概率模型。随着大规模语料库的出现，统计语言模型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，我们可以发现语言使用的普遍规律，进行语言知识的机器学习并对未知语言现象进行推测。

计算机借助统计语言模型的概率参数，可以估计出自然语言中每个句子出现的可能性，而不是简单地判断该句子是否符合文法。统计语言模型把语言序列看作一个随机事件，并赋予相应的概率来描述其属于某种语言集合的可能性。也就是，给定一个词汇集合 V，对于一个由 V 中的词构成的序列 $S=\{w_1;w_2;\cdots;w_m\}$，统计语言模型赋予这个序列一个概率 $P(S)$，来衡量 S 符合自然语言的语法和语义规则的置信度。常用的统计语言模型主要是 N 元语法模型（N-gram Model）。

（3）神经网络语言模型（Neural Network Language Model，NNLM）。该模型于 2003 年提出，它通过学习一个词的分布式表征来克服统计方法的维度灾难现象，且性能要大幅优于传统 N-gram 统计语言模型。随后，米科洛夫等于 2010 年提出了循环神经网络语言模型（Recurrent Neural Network Based Language Model，RNNLM），可实现对任意长度序列的语言表示。自此，神经网络语言模型逐渐成为自然语言处理领域主流的语言模型，并迅速发展。神经网络语言模型的主要思想是，通过提出词向量的概念，采用较低维度的实值连续向量来进行词和文本序列的分布式表示，代替 N-gram 模型使用高维离散的向量表示。

3.2 统计语言模型

3.2.1 模型概述

统计语言模型是自然语言处理的基础模型，是从概率统计角度出发，建模自然语言上下文相关特性的数学模型。统计语言模型的本质就是计算一个句子 $S=\{w_1;w_2;\cdots;w_m\}$ 在文本中出现的概率。

为计算一个句子的概率值，直觉上可以采用两种方法：① 直接以整个句子为单位统计相对频率 $P(w_1w_2\cdots w_m)$，但整个句子在语料库中的相对频率太低，不具有统计意义。② 根据句子构成单位的概率计算联合概率，即 $P(w_1)\times P(w_2)\times\cdots\times P(w_m)$，但这种方法假设句子中的词与词完全独立，损失了大量的序列信息。因此，需要一种更好的方法来计算句子的概率值，同时能尽可能保留句子中的序列信息。

定义 3.1 统计语言模型

对于一个句子 $S=\{w_1;w_2;\cdots;w_m\}$，统计语言模型定义其先验概率为

$$\begin{aligned}P(S)&=P(w_1)\times P(w_2|w_1)\times P(w_3|w_1w_2)\times\cdots\times P(w_m|w_1\cdots w_{m-1})\\&=\prod_{i=1}^{m}P(w_i|w_1\cdots w_{i-1})\end{aligned}\tag{3-1}$$

当 $i=1$ 时，记 $P(w_1|w_0)=P(w_1)$。

w_i 可以是字、词、短语或词类等，称为基元或统计基元，通常以“词”代之。由 $w_1\cdots w_{i-1}$ 构成的一个序列，称为 w_i 的历史，w_i 的概率由其历史决定。 ♣

给定句子：“友好的哥谭市民”可以有两种语义，即

① 友好的 || 哥谭 || 市民

② 友好 || 的哥 || 谭市民

我们可以通过统计语言模型计算这两种语义表达的概率，判断哪种语义更可能是正确的句子。

$$\begin{aligned} P(\text{友好的}||\text{哥谭}||\text{市民}) = {} & P(\text{友好的}|\text{<BOS>}) \times P(\text{哥谭}|\text{<BOS>}, \text{友好的}) \times \\ & P(\text{市民}|\text{<BOS>}, \text{友好的}, \text{哥谭}) \times \\ & P(\text{<EOS>}|\text{<BOS>}, \text{友好的}, \text{哥谭}, \text{市民}) \end{aligned} \tag{3-2}$$

$$\begin{aligned} P(\text{友好}||\text{的哥}||\text{谭市民}) = {} & P(\text{友好}|\text{<BOS>}) \times P(\text{的哥}|\text{<BOS>}, \text{友好}) \times \\ & P(\text{谭市民}|\text{<BOS>}, \text{友好}, \text{的哥}) \times \\ & P(\text{<EOS>}|\text{<BOS>}, \text{友好}, \text{的哥}, \text{谭市民}) \end{aligned} \tag{3-3}$$

其中，<BOS>/<EOS> 为增加的开始/结束标记，用于保证模型计算条件概率在 i=1 时有意义，且保证句子内所有字符串的概率之和为 1。可以通过大规模语料库统计各个条件概率值，并相乘得到不同语义的概率值。在不考虑上下文环境的情况下，自然语言处理模型通常将更高概率的语义作为该句的真实语义。

统计语言模型有一个关键的问题：随着历史基元数量的增加，不同的历史（路径）按指数级增长。对于第 i（$i > 1$）个统计基元，历史基元的个数为 $i-1$，如果共有 L 个不同的基元（词汇表大小为 L），理论上每一个词都有可能出现在 $1 \sim i-1$ 的每一个位置上，那么，第 i 个基元就有 L^{i-1} 种不同的历史情况。我们必须考虑在所有的 L^{i-1} 种不同历史情况下产生第 i 个基元的概率。那么，对于长度为 m 的句子，模型中有 L^m 个自由参数 $P(w_m|w_1 \cdots w_{m-1})$。假设 $L = 5000$，$m = 3$，自由参数的数目约为 1250 亿。

因此，统计语言模型需要设法减少历史基元的个数，将 $w_1 w_2 \cdots w_{i-1}$ 映射到等价类 $S(w_1 w_2 \cdots w_{i-1})$，使等价类的数目远远小于原来不同历史基元的数目，则有

$$P(w_i|w_1 w_2 \cdots w_{i-1}) = P(w_i|S(w_1 w_2 \cdots w_{i-1})) \tag{3-4}$$

在自然语言处理中，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。这种情况下的语言模型称为 n 元语法（N-gram）模型。

当 $n = 1$ 时，即出现在第 i 位上的基元 w_i 独立于历史。一元语法也写为 Unigram。

当 $n = 2$ 时，2-gram（Bigram）称为一阶马尔可夫链[12]；

当 $n = 3$ 时，3-gram（Trigram）称为二阶马尔可夫链，以此类推。

对于 $n > 2$ 的 N-gram，长为 m 的句子 S 的概率 $P(S)$ 可分解为

$$P(S) = \prod_{i=1}^{m+1} P(w_i|w_{i-n+1}^{i-1}) \tag{3-5}$$

[12] 马尔可夫链是数学中的一个随机过程，它描述了一个系统随时间变化的状态。m 阶马尔可夫链系统未来的状态仅与当前给定的 m 个状态有关，而与更早的状态无关。

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

仍以“友好的哥谭市民”为例，假设其为第一种语义“友好的 || 哥谭 || 市民”，根据 N-gram 语言模型计算其概率值。

1-gram：<BOS>，友好的，哥谭，市民，<EOS>。

2-gram：(<BOS>，友好的)，(友好的，哥谭)，(哥谭，市民)，(市民，<EOS>)。

3-gram：(<BOS>，友好的，哥谭)，(友好的，哥谭，市民)，(哥谭，市民，<EOS>)。

计算其二元语法概率值为

$$\begin{aligned} P(S) =& P(\text{友好的}|\text{<BOS>}) \times P(\text{哥谭}|\text{友好的}) \times \\ & P(\text{市民}|\text{哥谭}) \times P(\text{<EOS>}|\text{市民}) \end{aligned} \tag{3-6}$$

相较于式(3-2)，N-gram 语言模型可以极大减少模型历史基元的个数。

注意 假设词的总数为 m，N-gram 语言模型的样本空间大小为 V^m。对一元语法而言，本质上假设所有的词之间是相互独立的，因此会丢失大量序列信息。另外，即使采用 $N=4$、$N=5$ 也不可能覆盖所有词与词之间的相关性，需要采取其他一些长程的依赖性（long distance dependency）来解决。因此，在实际自然语言处理任务中，通常采用 N=2 或 N=3 进行处理。

3.2.2 参数估计

在明晰统计语言模型的基本定义后，需要根据语料库对模型的各项参数进行估计，即通过最大似然估计（maximum likelihood）方法，在训练语料上，用相对频率计算条件概率，如式(3-7)所示。

$$P(w_i|w_{i-n+1}^{i-1}) = f(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{\sum\limits_{w_i} c(w_{i-n+1}^{i})} \tag{3-7}$$

其中，$\sum\limits_{w_i} c(w_{i-n+1}^{i})$ 是历史串 w_{i-n+1}^{i-1} 在给定语料中出现的次数，$f(w_i|w_{i-n+1}^{i-1})$ 是在给定历史 w_{i-n+1}^{i-1} 条件下 w_i 出现的相对频度。

例题 3.1 假设有一个简单的英语语料库，其中包含以下句子

The cat sat on the mat.

The dog barked at the moon.

The sun set in the west.

根据二元语法求句子“The cat sat on the moon.”的概率时，首先将句子首、尾分别加入标识符 <BOS>、<EOS>，并统一转换成小写，概率计算过程如式(3-8)所示。

$$\begin{aligned} P(\text{the cat sat on moon}) = & P(\text{the}|\text{<BOS>}) \times P(\text{cat}|\text{the}) \times P(\text{sat}|\text{cat}) \\ & \times P(\text{on}|\text{sat}) \times P(\text{the}|\text{on}) \\ & \times P(\text{moon}|\text{the}) \times P(\text{<EOS>}|\text{moon}) \end{aligned} \tag{3-8}$$

$$= 1 \times \frac{1}{6} \times 1 \times 1 \times 1 \times \frac{1}{6} \times \frac{1}{3}$$
$$\approx 0.0093$$

一般而言，用于构建语言模型的训练语料需要上百万的词汇，规模越大，则采用最大似然估计得到的相对频率越接近语言现象的真实概率。但由于语言的开放性特点，无论多大的语料规模仍然不能覆盖所有的语言现象。当某个语言片段（词汇搭配）在语料库中没有出现时，语言模型的计算会出现零概率问题，即在计算实例的概率时，如果某个词在训练集中没有出现过，会导致整个语言片段的概率结果是 0。例如，如果想计算句子“The bird sat on the mat.”的概率，由于“The bird”没有在训练集中出现，其概率为 0。

显然，“The bird sat on the mat.”是一句完全正确的语句，仅仅因为训练语料的数据稀疏（sparse data）导致其出现零概率问题是不合理的。对此，需要对数据进行一定的处理，即数据平滑。

3.2.3　参数平滑

参数平滑的基本思想是，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，通过消除零概率，改进模型的整体正确率。

小知识　在介绍平滑方法之前，先给出一个简单的数据预处理的方法。对于未登录词（Out Of Vocabulary，OOV），即不在训练集中的词语，如果训练数据 V 含有大量词汇，由于词语分布属于长尾分布[13]，一般不会对所有的词汇全部进行训练，而是需要预先做下面的处理后再进行数据的平滑和训练。

假设训练数据集中出现 N 个不同的词汇，可根据词频对这些词汇进行排序，选择词频最高的 m 个词汇作为词汇集合 V，这样在训练集和测试集中，将不属于 V 的词汇都替换成特殊的词汇 UNK（unknown），这样可大幅减少计算量，也可提高计算的精度。

根据平滑技术是否组合使用不同信息分为两类：简单平滑和组合平滑。简单平滑通常包括加法平滑、留存平滑和古德-图灵（Good Turing）平滑等。组合平滑主要包括插值平滑和回退平滑等。下面简要介绍几种常用的简单平滑方法，更详细的介绍和比较可参阅有关文献。

1. 加法平滑

加法平滑也叫加 1 平滑、拉普拉斯（Laplace）平滑，是最简单的平滑算法，由法国数学家拉普拉斯最早提出。该方法的核心思想是在统计元素的次数时，计数器的初始值不要设成 0，而是设成 1。这样，即使该元素没有在训练集中出现，其出现次数统计值至少也是 1。

例如，对于二元语法模型，采用加法平滑后，条件概率变为

$$P(w_i|w_{i-1}) = \frac{1 + c(w_{i-1}w_i)}{\sum\limits_{w_i}[1 + c(w_{i-1}w_i)]} = \frac{1 + c(w_{i-1}w_i)}{|V| + \sum\limits_{w_i}[c(w_{i-1}w_i)]} \tag{3-9}$$

[13] 只有少部分词语是高频常用词，大部分词语出现次数很少。

其中，$|V|$ 为语料库中词的总个数。假设语料库中共有 w_1、w_2、w_3 三个词，概率分别为 1/3、0、2/3，加法平滑后概率变为 2/6、1/6、3/6。

加法平滑简单易懂，但存在两个严重问题：① 对非零项转移了太多的概率；② 对于没有出现过的项均给予一样的概率值。

2. 留存平滑

留存平滑的基本思想是通过把训练语料分成两个部分，即训练语料和留存语料，通过留存语料内的数据分布来改善训练语料估计出来的值。

对于每一个 n 元组 $w_1w_2\cdots w_n$ 计算其在训练语料中出现的频率 $c_{\text{tr}}(w_1w_2\cdots w_n)$ 和在留存数据中出现的频率 $c_{\text{ho}}(w_1w_2\cdots w_n)$。有可能存在 n 元组不在训练语料中出现但是在留存语料中出现。令

r：某个 n 元组在训练语料中出现的频率。

N_r：训练语料中出现 r 次的不同 n 元组的个数。

T_r：训练语料中出现 r 次的 n 元组在留存语料中的频率之和。

T：留存语料中 n 元组的数目。

则有

$$T_r = \sum_{w_1w_2\cdots w_n|c_{\text{tr}}(w_1w_2\cdots w_n)=r} c_{\text{ho}}(w_1w_2\cdots w_n) \tag{3-10}$$

$$p(w_1w_2\cdots w_n) = \frac{T_r}{T} \times \frac{1}{N_r}, r = c_{\text{tr}}(w_1w_2\cdots w_n) \tag{3-11}$$

其中，$\frac{T_r}{T}$ 是在训练语料中出现 r 次的 n 元组在留存语料中出现的概率。$\frac{1}{N_r}$ 是前面这个概率均分给出现 r 次的 n 元组。

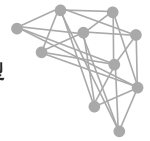
3. 古德-图灵平滑

古德-图灵估计法是很多平滑技术的核心，于 1953 年由古德引用图灵的方法而提出来的。其基本思想是：对于没有看见的事件，我们不能认为它发生的概率就是零，因此我们从概率的总量（probability mass）中，分配一个很小的比例给这些没有看见的事件。这样一来，看得见的事件概率总和就要小于 1。因此，需要将所有看见的事件概率调小一点。至于小多少，要根据“越是不可信的统计折扣越多”的方法进行。

以统计词典中的每个词的概率为例，假设语料库中出现 r 次的词有 n_r 个，未出现的词数量为 n_0，语料库的大小为 N。那么很显然：有

$$N = \sum_{r=1}^{\infty} rn_r \tag{3-12}$$

出现 r 次的词在整个语料库上的相对频度则是 r/N，如果不做任何处理，就以这个相对频度作为这些词的概率估计。



现在假设当 r 比较小时，它的估计可能不可靠，因此在计算出现 r 次的词的概率时，要使用一个更小的系数 r^*（而不是直接使用 r）。古德-图灵估计按照式(3-13)计算 r^*，即

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \tag{3-13}$$

显然有

$$\sum_r r^* n_r = N \tag{3-14}$$

在古德-图灵估计中，样本中出现 r 次的事件的概率为

$$p_r = \frac{r^*}{N} \tag{3-15}$$

通常出现一次的词的数量比出现两次的多，出现两次的比三次的多，r 越大，词的数量 n_r 越小，即 $n_{r+1} < n_r$。这种规律称为 Zipf 定律。一般情况下 $r^* < r$，而 $r_0^* > 0$，且有

$$\sum_{r>0} n_r \times p_r = 1 - n_0 \times p_0 = 1 - n_0 \times \frac{r_0^*}{N} = 1 - n_0 \times \frac{(r_0+1)\,n_1}{N n_0} = 1 - \frac{n_1}{N} \tag{3-16}$$

这样就有 $\frac{n_1}{N}$ 的剩余概率量可以均分给所有未见的事件（$r=0$），即给未出现的词赋予一个很小的非零值，从而解决了零概率的问题，同时下调了出现概率很低的词的概率。在实际的自然语言处理中，一般会设置一个阈值 T，仅对出现次数小于 T 的词做上述调整。

除此之外，常见的平滑方法还有：① Kneser-Ney 平滑，它是一种改进的加法平滑方法，它考虑了 n 元组的稀有程度。对于稀有的 n 元组，它使用不同的平滑因子，通常是一个递减的函数，这样稀有的 n 元组的平滑会比常见的 n 元组更小。② 逆文档频率（Inverse Document Frequency，IDF）平滑，一种结合了文本分析和信息检索的方法。它通过提高在语料库中出现次数较少的词的权重来改善模型。这种方法可以提高模型对文档中稀有词汇的敏感性。③ 矩阵分解平滑方法，如奇异值分解（Singular Value Decomposition，SVD）方法，通过将 n 元组概率矩阵分解为更小的矩阵来提高模型的参数效率和预测能力。每种平滑方法都有其优点和局限性，选择哪种方法取决于特定应用的需求和可用资源的考虑。在实际应用中，可能需要尝试多种平滑技术，以找到最适合特定任务的模型。

3.2.4　语言模型评价

在得到不同的语言模型时，如何判断一个语言模型的好坏，一般可分为内部评价和外部评价两类方法。外部评价是将其应用到具体的问题中，如机器翻译、语言识别、自动校正等，然后看这个语言模型在这些任务中的表现。但是，这种方法一方面难以操作，另一方面可能非常耗时。内部评价是根据语言模型自身的一些特性，来设计一种简单高效的评测指标，即困惑度。

困惑度的基本思想是：给测试集的句子赋予较高概率值的语言模型较好。当语言模型训练完后，测试集中的句子都是正常的句子，那么模型在测试集上的计算得到的句子概率越高越好，即

$$PP(W) = P(w_1 w_2 \cdots w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \cdots w_N)}} \tag{3-17}$$

由式(3-17)可知，句子概率越大，语言模型越好，困惑度越小。另一种更直观的理解是，困惑度表示的是平均分支系数，即平均来说，预测下一个词时有多少种选择。例如，某一个语言模型的困惑度是 70 时，可以理解为：平均情况下这个语言模型预测下一个词时，有 70 个词等可能地可以作为下一个词的合理选择。

3.2.5 统计语言模型存在的问题

统计语言模型虽然在建模句子的统计概率方面效果显著，但也存在一些问题，具体如下。

1. 稀疏性

因为 N-gram 模型只能对文本中出现的词或词组进行建模，当新的文本中出现意义相近但是没有在训练文本中出现的词或词组时，传统离散统计模型无法正确计算这些训练样本中未出现词的应有概率，它们都会被赋予零概率预测值。这是非常不符合语言规律的事情。传统的统计语言模型花费大量的精力来处理这个问题，包括平滑、插值、回退等方法，但效果不佳。

2. 泛化能力差

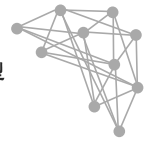
除对未出现的词本身进行预测非常困难外，离散模型还依赖固定的词组合，无法建模多个相似词的关系，需要完全的模式匹配，否则也无法正确输出词组出现的概率。一个好的语言模型应该能够识别出同义句应该有近似的概率分布，但是离散模型是无法达到这个要求的。这就使得此类模型的泛化能力不足。

3. 维度灾难

在统计语言模型中，实际应用最多的是的三元模型。IBM 曾进行过一次信息检索评测，发现二元语法模型需要数以亿计的词汇才能达到最优表现，而三元语法模型则需要数十亿级别的词汇，更高阶的模型很少有人使用，主要有两方面原因：一是阶数越高，计算量越大，潜在的计算量呈指数增长；二是高阶模型会面临严重的数据稀疏问题。当我们将更多词组合挑出来后才能更精准地预测特定词组出现的概率，但是这种组合的量是非常大的。假设词库有一万个独立词，对于一个包含 5 个词的词组模式，潜在的词组合多达 10^{20}。这给模型的表示和计算造成了非常大的困难。

4. 马尔可夫性过强（无法处理长距离依赖）

人在对文字进行处理时，能够将很长一段上下文纳入考虑，但是 N-gram 的离散模型统计特征只依赖该词汇前面的有限个词汇；一旦 N 给定，其依赖关系的窗口也就固定。在这种情形下，N-gram 模型对于超出窗口之外的依赖关系就无法建模。在自然语言中，上下文之间的相关度有可能跨度非常大，甚至可以从一个段落跨到另一个段落，远远超过模型的窗口。N-gram 模型在这种长距离依赖的情形下就变得无能为力，这也是马尔可夫假设的局限所在。



3.3　神经网络语言模型

传统语言模型的上述几个内在缺陷使得人们开始把目光转向神经网络模型，期望深度学习技术能够自动化地学习代表语法和语义的特征，解决稀疏性问题，并提高泛化能力。

本吉奥等较早提出了神经网络语言模型的概念，但由于神经网络训练困难，而且受当时硬件条件的限制，因此神经网络语言模型并未得到足够的应用。直到 2003 年，本吉奥发布其研究成果，该模型才引起学术界及工业界的兴趣。随后，米科洛夫等将 RNN 引入语言建模，使得语言模型的性能得到较大的提升。接着，RNN 的改进版本——LSTM 循环神经网络以及门限循环单元（Gated Recurrent Unit，GRU）神经网络，相继被用于进一步改善语言建模的性能。另外，卷积神经网络（Convolutional Neural Network，CNN）也在语言建模中表现优异，广泛地应用于文本表示和分类等各项任务中。

神经网络语言模型主要设计用来解决传统统计语言模型稀疏性问题，而 RNN 模型以及 CNN 模型对于统计语言模型的泛化能力和长上下文信息的处理能力有着极大的改善。本节将主要介绍神经网络语言模型，RNN 模型和 CNN 模型将在后续文本表示章节详细说明。

前面提到，语言模型的一个主要任务就是要基于给定的上下文信息，估计当前每一个词出现的概率。神经网络词向量表示技术通过神经网络技术对上下文及其与目标词之间的关系进行建模。神经网络较为灵活，其最大优势在于可以表示复杂的上下文。

本吉奥等提出的第一个神经网络语言模型利用一个三层（输入层、全连接层和输出层）的全连接神经网络模型来估计给定 $n-1$ 个上文的情况下，第 n 个词出现的概率。其架构如图3-1所示。

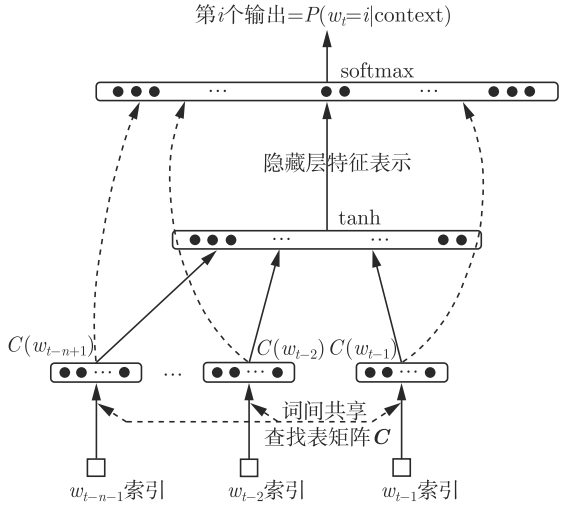


图 3-1　神经网络语言模型基本架构

给定一个文本语料库，其中所有的词构成一个词典 D，对语料中一段长为 n 的词窗口

中的词语序列 $w_{i-n+1}\cdots w_{i-1}w_i$，$N$-gram 语言模型需要最大化以下概率或似然函数，即

$$P(w_i|w_{i-n+1}\cdots w_{i-1}) \tag{3-18}$$

w_i 为需要通过语言模型预测的词 (目标词)。对于整个模型而言，输入为条件部分的整个词序列 $w_{i-n+1}\cdots w_{i-1}w_i$，输出为目标词的分布。模型通过如下方式进行学习。

1. 输入层（嵌入层）

在第一层，模型首先将词典 D 中的所有词映射到一个给定维度 k 的高维空间，这个映射就是图3-1中的参数矩阵 $\boldsymbol{C}\in\mathbb{R}^{|V|\times k}$，随机初始化生成，也称查找表。该矩阵行数等于词典中的词数量，列数等于给定的高维空间的维度 k，即将词 w_t 表示为一个 k 维行向量[14]，$\boldsymbol{w}_t\in\mathbb{R}^k$。词在高维空间的映射就是该词的词向量表达，将 $\boldsymbol{w}_{t-n+1},\cdots,\boldsymbol{w}_{t-1}$ 这 $n-1$ 个 k 维向量[15]首尾相接拼起来，形成一个 $(n-1)k$ 维的向量 $\boldsymbol{X}$ 作为目标词 w_t 的上文表示输入神经网络模型，即

$$\boldsymbol{X}=\boldsymbol{w}_{t-n+1}\oplus\cdots\oplus\boldsymbol{w}_{t-2}\oplus\boldsymbol{w}_{t-1} \tag{3-19}$$

2. 隐藏层

隐藏层是一个标准的全连接神经网络模型。具体使用线性函数和非线性激活函数进行计算，即

$$\boldsymbol{h}=\tanh(\boldsymbol{X}\boldsymbol{W}_h+\boldsymbol{b}_h) \tag{3-20}$$

其中，$\boldsymbol{b}_h\in\mathbb{R}^h$ 是一个偏置向量，$\boldsymbol{W}_h\in\mathbb{R}^{(n-1)k\times h}$ 是隐藏层权重向量。使用 $\tanh(\cdot)$ 作为激活函数[16]，将每个词对应的上下文映射到词典全部词对应的高维语义分布空间中。

3. 输出层

在输出层中对隐藏层的输出 $\boldsymbol{X}$ 进行一次线性变换，将高维语义分布空间重新映射到词的条件概率分布空间中，每个结点 $\boldsymbol{y}_i$ 表示下一个词为 w_i 的未归一化概率。使用 softmax 函数将输出值 $\boldsymbol{y}$ 归一化成概率。计算交叉熵损失函数值，以梯度下降方式进行反向传播，在反向传播过程中对参数 $\boldsymbol{\theta}=\{\boldsymbol{C},\boldsymbol{W}_h,\boldsymbol{b}_h,\boldsymbol{U},\boldsymbol{W}_r,\boldsymbol{b}_r\}$ 进行更新，即

$$\boldsymbol{y}=\tanh(\boldsymbol{X}\boldsymbol{W}_h+\boldsymbol{b}_h)\boldsymbol{U}+\boldsymbol{X}\boldsymbol{W}_r+\boldsymbol{b}_r \tag{3-21}$$

$$P(w_t=i|\text{context})=\text{softmax}(\boldsymbol{y}_i)=\frac{\mathrm{e}^{\boldsymbol{y}_i}}{\sum\limits_{j=1}^{|V|}\mathrm{e}^{\boldsymbol{y}_j}} \tag{3-22}$$

其中，$\boldsymbol{U}\in\mathbb{R}^{h\times|V|}$，$\boldsymbol{W}_r\in\mathbb{R}^{(n-1)k\times|V|}$，$\boldsymbol{b}_r\in\mathbb{R}^{|V|}$。

注意，矩阵 $\boldsymbol{W}_r$ 包含了从输入层到输出层的直连边 (图3-1中虚线)，其作用是实现从输入层直接到输出层的一个线性变换。

[14]在不同的文献中，词也可以表示为列向量。

[15]$\boldsymbol{w}_{t-n+1}$ 即为图3-1中的 $C(w_{t-n+1})$，统一为本书的向量表示形式。

[16]激活函数见 2.1.3 节。