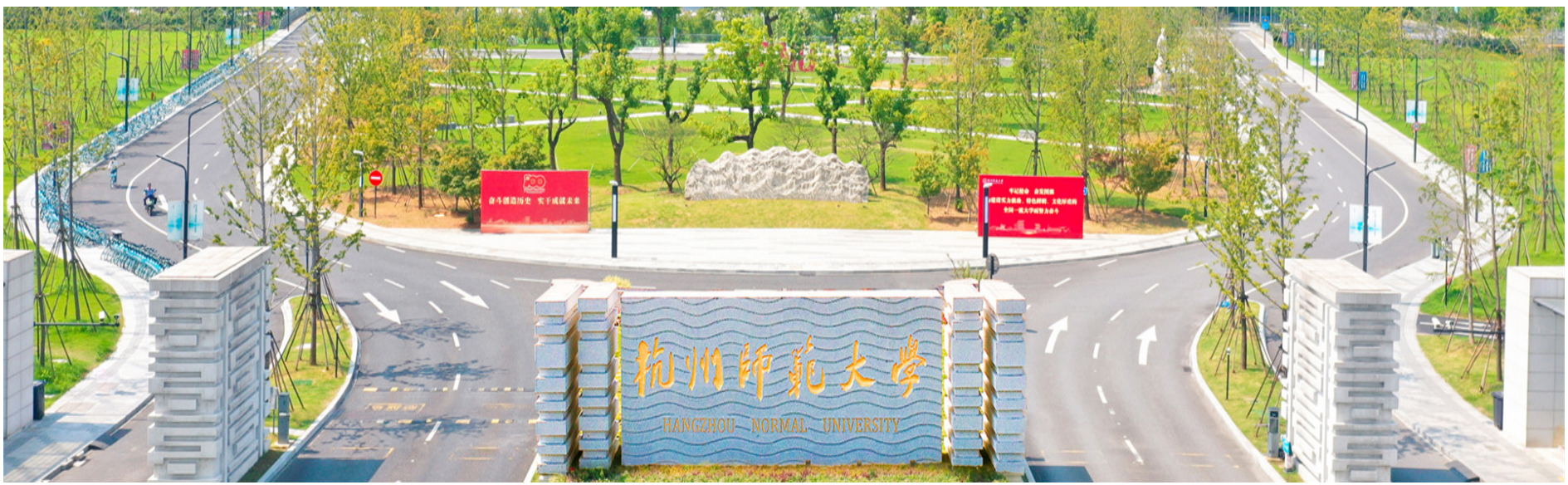
第三讲 - NLP基础算法：N-gram语言模型

张建章

阿里巴巴商学院
杭州师范大学

2025-02-01



目录

语言模型（Language Model，LM）：是一种机器学习模型，其主要任务是根据已有的上下文来**预测**接下来的单词。该模型为每个可能的下一个**单词**分配一个概率，进而构成一个完整的概率分布；同时，语言模型也能够为整个句子计算联合概率，从而反映出不同词序排列的合理性。

实际功能与应用场景：

- **文本生成与纠错：**通过比较不同词序的概率，语言模型能够帮助纠正语法或拼写错误，如将“Their are”纠正为“There are”，或修正诸如“has improve”之类的不规范表达；
- **语音识别与辅助沟通：**在语音识别系统中，语言模型帮助区分发音相似但意义截然不同的词序；同时，在辅助交流系统（AAC）中，预测功能可以为用户提供上下文相关的候选词，提升输入效率。

单词预测不仅是解决特定任务（如拼写检查、语音识别）的手段，更是训练**大语言模型**的核心任务。当前许多基于神经网络的大语言模型，其训练目标正是通过大量预测任务来学习丰富的语言结构与语义知识。

本讲将以最简单的**n-gram语言模型**为例，详细阐述如何利用固定窗口内的上下文（即n-1个词）来近似计算下一个词的概率。尽管n-gram模型本身在表达能力上较为局限，但其明确的数学形式和直观的统计方法，为理解训练集、测试集、困惑度（Perplexity）、采样以及后续更复杂的插值方法等**大语言模型核心概念**提供了坚实基础。

一个**n-gram**就是一个包含n个单词的序列，n常见的取值为1 (unigram)、2 (bigram)、3 (trigram)，如下图所示：

This is Big Data AI Book

Uni-Gram	This	Is	Big	Data	AI	Book
Bi-Gram	This is	Is Big	Big Data	Data AI	AI Book	
Tri-Gram	This is Big	Is Big Data	Big Data AI	Data AI Book		

目录

在n-gram模型中，**目标是**通过已知的历史词汇（即前面的n-1个单词），**预测下一个单词**。具体来说，n-gram模型估计$P(w|h)$的条件概率，其中w是下一个单词，而h是n-1个之前的单词。

1. 概率估计的直接方法

为了估计$P(w|h)$，一种直观的方法是通过**计算相对频率**：统计在一个大规模语料库中，历史序列h出现的次数和h后跟w出现的次数，并通过这些计数来估算概率：

$$P(w|h) = \frac{C(h,w)}{C(h)}$$

其中，$C(h,w)$表示序列h后接单词w的次数，C(h)表示序列h出现的总次数。

2. 链式法则（Chain Rule）

为了估算整个词序列的联合概率$P(w_1^n) = P(w_1, w_2, ..., w_n)$，可以使用链式法则，将长序列的联合概率转化为一系列条件概率的乘积

$$P(w_1, w_2, ..., w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)...P(w_n|w_1, ..., w_{n-1}) \quad (1)$$

$$= \prod_{k=1}^{n} P(w_k|w_{1:k-1}) \quad (2)$$

这种方法虽然直观，但在实际应用中，由于语言的创造性，某些长词序列的计数可能并不存在，因此需要进行更精巧的概率估计方法。

马尔科夫假设

1. 马尔科夫假设

为了简化问题，n-gram模型引入了**马尔科夫假设**，即假设下一个单词的概率仅依赖于前n-1个单词。例如，在bigram模型中 (一阶马尔科夫假设)，预测下一个单词w的概率为：

$$P(w|h) \approx P(w|w_{n-1}) \tag{3}$$

这意味着，n-gram模型通过限制上下文的长度，减少了计算复杂度，使得模型更加高效。

2. 扩展到更高阶的n-gram

当n增大时，模型能够利用更多的上下文信息，在N-gram模型中 (N-1阶马尔科夫假设)，预测下一个单词w的概率为：

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-N+1:n-1}) \tag{4}$$

3. N-gram模型

将 (4)式表示的N-gram模型带入 (2)式表示的词序列联合概率，可得如下近似：

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k | w_{k-N+1:k-1})$$

N=2时，应用bigram模型，词序列的联合概率可近似为：

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k | w_{k-1})$$

目录

最大似然估计（MLE）

n-gram模型的参数（即各个单词序列的概率）可以通过最大似然估计（MLE）来计算。这种方法通过计算语料库中n-gram出现的频率，并对频率进行归一化处理来得到概率值。具体而言，大体步骤如下：

1. 频率计算

对于每一个n-gram，计算其在语料库中的出现次数。

2. 概率归一化

为了将频率转换为概率，需要将每个n-gram的频率除以其前一个单词序列的总频率，概率估计公式为：

$$P(w_n|w_{n-N+1:n-1}) = \frac{C(w_{n-N+1:n-1}, w_n)}{C(w_{n-N+1:n-1})}$$

通过这种方法，n-gram模型根据统计数据估计词汇序列的概率。

Bigram模型参数最大似然估计示例

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Figure 3.1 Bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Zero counts are in gray. Each cell shows the count of the column label word following the row label word. Thus the cell in row **i** and column **want** means that **want** followed **i** 827 times in the corpus.

Figure 3.2 shows the bigram probabilities after normalization (dividing each cell in Fig. 3.1 by the appropriate unigram for its row, taken from the following set of unigram counts):

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Figure 3.2 Bigram probabilities for eight words in the Berkeley Restaurant Project corpus of 9332 sentences. Zero probabilities are in gray.

Here are a few other useful probabilities:

$$P(\texttt{i}\,|\,\texttt{<s>}) = 0.25 \qquad P(\texttt{english}\,|\,\texttt{want}) = 0.0011$$
$$P(\texttt{food}\,|\,\texttt{english}) = 0.5 \qquad P(\texttt{</s>}\,|\,\texttt{food}) = 0.68$$

Now we can compute the probability of sentences like *I want English food* or *I want Chinese food* by simply multiplying the appropriate bigram probabilities together, as follows:

$$\begin{aligned} P(\texttt{<s> i want english food </s>}) &= P(\texttt{i}\,|\,\texttt{<s>})P(\texttt{want}\,|\,\texttt{i})P(\texttt{english}\,|\,\texttt{want}) \\ &\qquad P(\texttt{food}\,|\,\texttt{english})P(\texttt{</s>}\,|\,\texttt{food}) \\ &= 0.25 \times 0.33 \times 0.0011 \times 0.5 \times 0.68 \\ &= 0.000031 \end{aligned}$$

n-gram模型的应用与挑战

1. 应用场景

n-gram语言模型广泛应用于各种自然语言处理任务，如机器翻译、语音识别、文本生成等。通过在大量文本数据上训练，n-gram模型能够有效地捕捉语言的统计特性，为后续的应用提供支持。

2. 挑战

- **数据稀疏问题**：n-gram模型的一个主要问题是，当语料库中不存在某个n-gram时，模型会面临“零概率”问题，即某些词序列的概率为零。为了解决这个问题，常使用**平滑技术**（如Laplace平滑）来调整概率估计。此外，为避免概率乘积数值下溢，语言模型的概率均在计算和存储过程中使用**对数概率**。

- **高阶模型的计算复杂度**：随着n的增加，n-gram模型的参数数量急剧增加，导致计算资源和存储需求的上升，n通常不超过3。

目录

评估语言模型的基本目标

1. 评价目标

评估语言模型的最终目的是判断其对未知数据的预测能力，即模型在面对未见过的测试数据时，能否较好地“拟合”数据。较优的模型会为测试数据分配较高的概率，从而体现出更好的泛化性能。

2. 内在与外在评价

外在评价（Extrinsic Evaluation）：将语言模型**嵌入到实际应用中**，评估其在具体任务中的表现。这种评价方式直接反映了模型在特定应用场景中的有效性，但通常需要高昂的计算成本和大量的实际应用数据。

例如，假设我们有两个不同的语言模型，一个使用的是bigram模型，另一个使用的是trigram模型。在语音识别任务中，我们将这两个模型分别应用到语音转录系统中，比较其转录准确率。

内在评价（Intrinsic Evaluation）：指在不依赖于具体应用的情况下，通过一些**统计指标**或数学方法直接评估语言模型本身的性能。这类评价通常使用一些标准的评估指标，如困惑度（Perplexity）等。

训练集、测试集、开发集

训练集（Training Set）：用于学习模型参数的语料库。例如，在n-gram语言模型中，使用最大似然估计，利用训练集中的n-gram计数，经过归一化处理得到模型的概率分布。

测试集（Test Set）：一组与训练集不重叠的、独立的语料，用于评估模型在新数据上的泛化能力。如果一个模型在训练集上表现极佳，但在测试集上效果很差，则说明模型可能过拟合训练数据，无法适用于未知数据。

开发集（Dev Set）：模型在训练过程中即使不直接将测试数据用于训练，多次反复在测试集上调试模型，也可能使模型隐式地适应测试集分布。因此，通常建议在模型开发阶段使用一个独立的开发集，待模型调试完毕后再**在测试集上进行一次性评估**。

数据集划分的原则

领域选择：测试集应能真实反映目标应用领域的语言特征。例如：若模型用于语音识别化学讲座，其测试集应来自化学讲座文本。若构建通用语言模型，则测试集应涵盖多样化的文本来源，避免单一作者或单一文档对评价结果的偏倚。

数据量的权衡：理想情况下，测试集应足够大，以确保统计上的代表性和评价的稳定性；但又不能过大以至于严重影响训练数据的量。

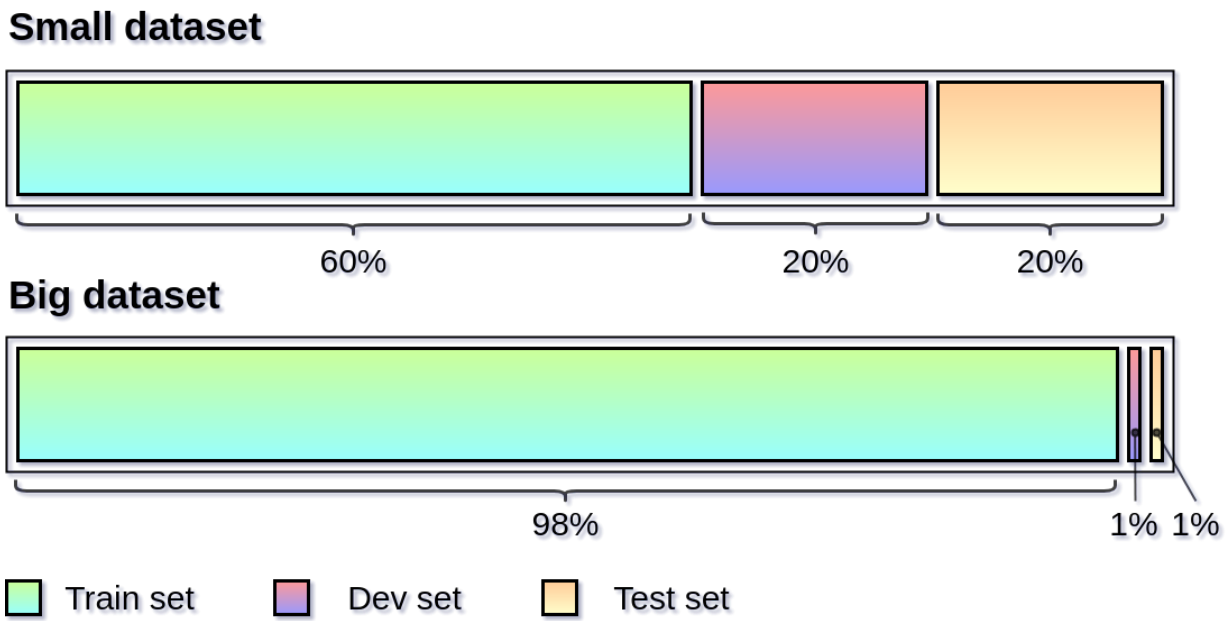


图 1: 数据集的划分比例

内部评估指标：困惑度

1. 困惑度的评价动机

语言模型的基本任务是为给定的测试文本分配概率，模型对测试集预测得越好，即为测试集赋予越高的概率，其性能就越优。然而，直接比较测试集的联合概率存在一个问题：**文本越长，概率值越小**，不便于跨文本或跨模型比较。

为了消除文本长度对联合概率的影响，引入**“困惑度”（Perplexity）**这一评价指标，通过对测试集的联合概率进行归一化（通常是取N次根），实现了每个单词的平均预测效果度量。

对于一个包含 N 个单词的测试集 $W = w_1 w_2 \dots w_N$，困惑度定义为：

$$\text{Perplexity}(W) = P(w_1 w_2 \dots w_N)^{-1/N}$$

通过链式法则展开为：

$$\text{Perplexity}(W) = \left(\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 \dots w_{i-1})} \right)^{1/N}$$

这表明，困惑度是所有**条件概率倒数的几何平均值**，其值**与模型对测试集的预测准确性成反比**。

2. 困惑度的直观解释

一个好的语言模型能够对测试集中的每个单词赋予较高的条件概率，从而使得整个测试集的联合概率较高，其倒数取N次根后的困惑度较低。在实际评价中，**困惑度是衡量语言模型性能的重要指标**，困惑度低表示模型对下一个单词的预测更为准确和确定。

困惑度在模型评价中的应用

1. 比较不同模型

模型的困惑度**越低越好**。困惑度计算对词汇表敏感，只有在**不同模型使用相同的词汇表**时，困惑度值才具有可比性。

2. 困惑度与实际任务表现

困惑度作为一种内在评价指标，不仅方便计算（无需嵌入实际应用），而且在统计上能较好地反映模型对语言序列的拟合程度。尽管困惑度通常与实际应用（如语音识别、机器翻译）的表现呈负相关，但**困惑度的改善不一定能直接转化为实际任务效果的提升**。

3. 数据划分要求

为确保评价结果的有效性，**测试集必须与训练集严格分离**；同时，如果在模型调整过程中频繁使用测试集进行评估，可能导致隐性调参，使困惑度偏低，评价结果失真。

目录

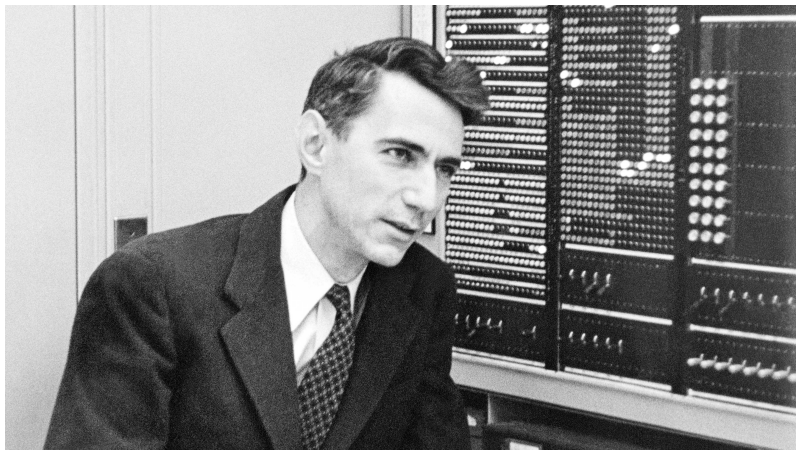
采样 (sampling)的基本思想

1. 定义

采样即根据概率分布随机选择数据点。语言模型定义了一个关于句子（或词序列）的概率分布，采样过程就是按照各个句子的概率高低随机生成句子：高概率的句子更容易被生成，低概率的则较少出现。

2. 目的

通过从模型定义的概率分布中随机抽取样本，可以“生成”出一系列句子，**直观观察**模型认为哪些句子更有可能出现，从而了解**模型对语言结构和词序的把握情况**，检测模型生成文本的连贯性和合理性。这种利用采样来可视化语言模型的方法，最早由 Shannon（1948）等人提出。



采样句子的具体过程

1. 以 Unigram 模型为例

- 设想整个英语词汇沿着区间 [0, 1] 分布，每个单词所占的区间长度与其在训练语料中的频率成正比；
- 通过随机选取 [0, 1] 内的一个数，找到其落在某个单词对应区间内，则生成该单词；
- 持续进行这一过程，直到随机生成特殊的句末符号</s>，从而构成一个完整的句子。

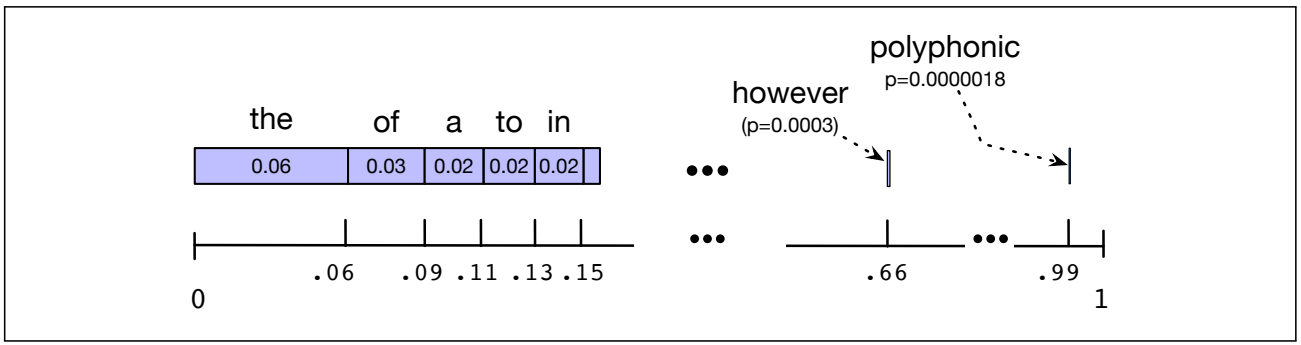


Figure 3.3 A visualization of the sampling distribution for sampling sentences by repeatedly sampling unigrams. The blue bar represents the relative frequency of each word (we've ordered them from most frequent to least frequent, but the choice of order is arbitrary). The number line shows the cumulative probabilities. If we choose a random number between 0 and 1, it will fall in an interval corresponding to some word. The expectation for the random number to fall in the larger intervals of one of the frequent words (*the*, *of*, *a*) is much higher than in the smaller interval of one of the rare words (*polyphonic*).

2. 扩展到 Bigram 模型

假设有一个词汇集合 (包括句首符号<s>和句末符号</s>)和经过训练得到的条件概率分布，且所有条件概率均满足对每个前驱词归一化，例如，对于单词"I"，$P(am|I) = 0.8$，$P(like|I) = 0.2$。

- 首先，从标记为句首的特殊符号 <s> 开始，根据 <s> 后各单词的 bigram 概率采样生成第二个单词；
- 接下来，将刚生成的单词作为条件，再从以该单词为前缀的 bigram 分布中采样生成下一个单词；
- 重复此过程，直到生成句末符号 </s> 为止，从而形成完整的句子。

目录

1. 泛化能力的体现

一个优秀的语言模型不仅在训练数据上拟合得好，更重要的是能在未见过的测试数据上做出准确预测，即具备较强的**泛化能力**。模型对测试数据赋予的概率越高，说明其预测能力越强。

2. 模型过拟合训练数据

n-gram 模型是基于训练语料统计得到的概率分布，其参数直接来源于语料中的 n-gram 计数。因此，模型不仅捕捉到一般的语言规律，还**不可避免地编码了训练语料中出现的具体事实和特定模式**。

随着 n-gram 阶数的增高，即n变大，模型能利用更长的上下文捕捉细节，**记住**更多具体的词序信息，从而在训练集上表现出更高的预测准确性，但同时也容易陷入**过拟合**，即**模型过分依赖于训练语料中的具体模式**，导致在测试集或新数据上表现不佳。

低阶模型（如 unigram、bigram）：生成的句子往往缺乏连贯性，无法捕捉长期依赖；

高阶模型（如 trigram、4-gram）：生成的句子在局部结构上较为连贯，但当模型阶数过高时，生成的句子可能直接“复制”训练语料中的片段，显示出明显的过拟合现象。

3. 缓解过拟合

为了提高泛化能力，应选择**与实际应用场景相似**的训练语料，并注意**数据的多样性**。例如，构建用于社交媒体文本处理的语言模型时，则应采集多个不同社交媒体平台的数据。

除了选择合适的语料外，还应严格保证测试集与训练集不重叠，防止模型“见过”测试数据，从而造成**过拟合现象的低估**。

目录

“零概率”问题

由于训练语料有限，某些在测试集中可能合理出现的 n-gram 在训练中未曾出现，其 MLE 估计为零；若某个词在特定上下文中概率为 0，则整个句子的概率也为 0，导致困惑度等评价指标失效。为应对训练数据中“零概率”问题而设计的3种常用技术：

- **平滑**：通过对计数进行加常数处理（如加一或加k），为未见事件赋予非零概率，但可能过度平滑。；
- **插值**：结合不同阶数 n-gram 的信息，利用低阶模型补充高阶模型的不足，并通过保留集来学习合适的权重分布；
- **回退**：当高阶 n-gram 无法获得可靠统计时，直接回退到低阶模型，并乘以固定的衰减因子，以实现一种简单高效的概率估计。

这些方法各有优劣，但都旨在改善模型对未见事件的估计，从而提高语言模型的整体泛化能力。

平滑（Smoothing）

1. 基本思想

对计数进行调整，将一部分概率质量从已见事件转移到未见事件上。

Laplace（加一）平滑：对每个 n-gram 的计数加一，即将零计数变为1，已见计数由 c 变为 $c+1$。

对于 unigram：

$$P_{\text{Laplace}}(w_i) = \frac{c(w_i)+1}{N+V}$$

对于 bigram：

$$P_{\text{Laplace}}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

其中 V 为词汇表大小。

局限性：此方法简单直观，但往往会过度转移概率质量，导致已见事件的概率被显著降低，从而影响模型性能。

Add-k 平滑：与加一平滑类似，但不是加 1，而是加一个较小的常数 k（例如0.5或0.01），以减少对已见计数的过度修正。

$$P^*_{\text{Add-k}}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + k}{C(w_{n-1}) + kV}$$

调优要求：需要利用开发集 (Dev Set)来确定一个合适的 k 值。

插值（Interpolation）技术

1. 基本思想

当高阶 n-gram 的计数不足时，可以利用低阶 n-gram（如 unigram、bigram）的信息对其进行补充。插值方法通过线性组合不同阶数 n-gram 的概率，平滑地整合上下文信息，从而提高对未见事件的估计。

简单线性插值：对于 trigram 模型，利用 unigram、bigram 和 trigram 的概率进行组合，其形式为：

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n|w_{n-2}w_{n-1})$$

其中 $\lambda_1 + \lambda_2 + \lambda_3 = 1$。

上下文条件插值：为使插值更灵活，可令各个 λ 权重依赖于上下文，即：

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) = &\lambda_1(w_{n-2},w_{n-1})P(w_n)\\ &+\lambda_2(w_{n-2},w_{n-1})P(w_n|w_{n-1})\\ &+\lambda_3(w_{n-2},w_{n-1})P(w_n|w_{n-2}w_{n-1})\end{aligned}$$

简单插值和上下文插值的权重λ通过在独立的保留语料（held-out corpus）上最大化似然函数来学习，使更可靠的 n-gram 拥有更高的权重。

回退（Backoff）技术

1. 基本思想

当高阶 n-gram (例如 trigram)无法获得可靠统计时 (例如计数为零)，直接回退到低阶模型 (例如 bigram)，并乘以固定的衰减因子，以实现一种简单高效的概率估计。

2. Stupid Backoff 算法

如果某个 n-gram 的计数大于零，则使用其 MLE 估计；否则，将概率设置为较**低阶模型概率**乘以一个固定的衰减因子 λ (例如 0.4)，递归回退至 unigram。

$$S(w_i \mid w_{i-N+1:i-1}) = \begin{cases} \dfrac{\text{count}(w_{i-N+1:i})}{\text{count}(w_{i-N+1:i-1})}, & \text{if count}(w_{i-N+1:i}) > 0; \\ \lambda\, S(w_i \mid w_{i-N+2:i-1}), & \text{otherwise.} \end{cases}$$

Stupid Backoff 不试图形成一个完整的概率分布，其主要目的是作为一个简单、计算高效的启发式方法来处理稀疏性问题，λ通常为0.4。

目录

熵的定义及其在语言中的应用

1. 熵的定义

给定随机变量 X 及其概率分布 $p(x)$，熵定义为

$$H(X) = -\sum_{x \in \chi} p(x) \log_2 p(x)$$

熵衡量了随机变量的不确定性，直观上可以看作编码某一信息所需的最少平均比特数 (bits)。

- 当 X 表示单个词时，熵反映了对单词选择的不确定性；

- 当 X 表示一个词序列 $W = w_1, w_2, \ldots, w_n$ 时，熵反映了整个语言的统计特性。为了得到每个词的平均不确定性，可以定义**熵率**，即：

$$\frac{1}{n} H(w_1, w_2, \ldots, w_n)$$

对于语言L，需要考虑无限长序列，即$n \to \infty$，将语言L视为一个生成词语序列的随机过程，则语言的熵率可以定义为：

$$\begin{aligned} H(L) &= \lim_{n\to\infty} \frac{1}{n} H(w_1, w_2, \ldots, w_n) \\ &= -\lim_{n\to\infty} \frac{1}{n} \sum_{w_{1:n}\in L} p(w_1, w_2, \ldots, w_n) \log p(w_1, w_2, \ldots, w_n) \end{aligned}$$

2. Shannon-McMillan-Breiman 定理

如果一个随机过程（如语言L）满足一定的正则性条件，即，既平稳又遍历，则有

$$H(L) = \lim_{n\to\infty} -\frac{1}{n} \log p(w_1, w_2, \ldots, w_n)$$

这一定理说明，一个足够长的文本样本其平均对数概率可以近似于语言的真实熵率，从而为利用单个长序列估计熵提供了理论依据。

交叉熵与模型评价

1. 交叉熵的定义

当不知道真实分布 p 时，可以用模型 m 来近似。交叉熵定义为：

$$H(p,m) = \lim_{n\to\infty} -\frac{1}{n}\sum_{W\in L} p(w_1,\ldots,w_n)\log m(w_1,\ldots,w_n)$$

对于满足正则条件的随机过程，可用单个长序列来近似估计交叉熵：

$$H(p,m) = \lim_{n\to\infty} -\frac{1}{n}\log m(w_1 w_2 \ldots w_n)$$

2. 交叉熵与真实熵的关系

交叉熵 $H(p,m)$ 总是不低于真实熵 $H(p)$（即 $H(p) \le H(p,m)$），且两者之间的差值反映了模型 m 的准确程度。因此，在模型比较中，交叉熵较低的模型通常被认为对数据分布的近似更为准确。

困惑度与交叉熵的关系

使用一个固定长度 (N)的足够长的序列 W，即长度为N的测试集，来近似计算模型与真实分布之间的交叉熵，则交叉熵的估计表达式可以写为：

$$H(W) = -\frac{1}{N}\log P(w_1 w_2 \dots w_N)$$

困惑度是评价语言模型对测试集预测能力的指标，其定义为测试集联合概率的归一化倒数：

$$\text{Perplexity}(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

因此，困惑度可以表示为：

$$\text{Perplexity}(W) = 2^{H(W)}$$

困惑度与交叉熵之间的关系说明：模型对测试集赋予的平均对数概率越高，即**交叉熵越低**，其**困惑度就越低**，说明该模型的预测能力越强。

本小节的核心内容在于建立了困惑度与熵、交叉熵之间的理论联系，为使用困惑度作为评价语言模型性能的指标提供了信息论基础。具体地：

- 通过定义熵与熵率，说明了信息的不确定性；

- 利用 Shannon-McMillan-Breiman 定理，用长序列的平均对数概率估计语言的熵率；

- 定义了交叉熵，并指出交叉熵是对真实熵的上界，其差值反映了模型的精度；

- 建立交叉熵与困惑度的联系，即：

$$\text{Perplexity}(W) = 2^{H(W)}$$

这一系列理论构建解释了困惑度为何采用测试集概率的归一化倒数，因为要建立与交叉熵的关系，而交叉熵又是衡量模型准确程度的信息论标准。因此，**困惑度定量反映模型对语言的预测能力，是衡量语言模型好坏的重要内在指标**。

THE END