3 综述 —— N元语言模型

第三章内容概述

语言模型建模是提高机器语言智能的主要方法[1]。一般来说，语言模型对语言序列的生成可能性进行建模，从而能够预测词元进而预测语言序列的概率。通过“下一个词元预测”，语言模型可以非常自然地应用在文本生成任务上，并能以这种方式统一所有自然语言处理任务。因此，不断增强语言模型的对序列概率的建模能力，似乎是一条提高模型任务表现性能、最终实现通用人工智能的可能途径。而这也正是当今大语言模型时代所选择的路径。无论是当下的大语言模型阶段还是上一个预训练语言模型阶段，其本质都是语言模型，而本章的主要内容则关注于统计语言模型阶段的N元语言模型。N元语言模型的基本思想是假设一个单词的出现概率仅依赖于其前面的N-1个单词，即马尔可夫假设。语言模型仍然是一种模型，同样会有训练、预测、预测、评估等模型该有的“生命周期”。原始的N元语言模型存在数据稀疏这一重大问题，因此针对其提出了平滑技术进行改进，包括拉普拉斯平滑和古德-图灵估计等。本章对困惑度进行了详细的阐述，除了其作为语言模型性能的评估的简单定义，还对其与熵的可能关系做了初步分析。

语言模型

语言模型在机器翻译、语音识别、文本生成和情感分析等各类自然语言处理任务中扮演着关键角色。通过分析和理解人类语言的结构和用法，能够处理和生成上下文恰当、连贯的文本。自然语言处理的历史进程从统计模型发展到神经语言模型，然后从预训练语言模型发展到大语言模型。

语言模型的主要目的是捕捉自然语言的统计特性。通过学习单词序列的概率分布，语言模型可以预测给定词元在一系列词元之后的可能性。这种预测能力对于需要理解文本上下文和意义的任务至关重要。例如，在文本生成中，语言模型可以通过逐次预测序列中的下一个单词来生成合理且上下文相关的内容。在机器翻译中，语言模型通过理解并生成目标语言中的语法正确的句子，来帮助将文本从一种语言翻译成另一种语言。

假设词汇表为 V ，词 $w_i \in V$ ，词序列或句子 $W = (w_1, w_2, ..., w_n)$ ，则语言模型的一般形式定义为如下分布：

$$P(W) = P(w_1, w_2, \ldots, w_n)$$

根据概率链式法则，词序列的联合概率可以分解为：

$$P(W) = P(w_1, w_2, \ldots, w_n) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times \cdots \times P(w_n|w_1, w_2, ..., w_{n-1})$$

N 元语言模型

直觉理解与形式定义

直觉上，N元语言模型的核心思想是通过观察文本中词语的**共现**情况，预测下一个词或字符。

直接计算或估计 $P(W) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times \cdots \times P(w_n|w_1, w_2, ..., w_{n-1})$ 是非常困难的，这既是由于语言的复杂性意味着每个词的出现可能依赖于文本中前面多个词，那么考虑整个上下文会导致组合数量爆炸，计算将变得不可行。同时，在大多数真实世界的语言数据中，许多词的组合是非常稀疏的，导致模型无法有效估计其出现的概率。

因此， N元语言模型引入马尔科夫假设来简化上述概率分布的计算。

马尔可夫假设源自马尔可夫过程，其核心思想是系统的未来状态仅与当前状态相关，而与过去的状态无关。这一假设在统计学和机器学习中有广泛的应用，尤其在自然语言处理和时间序列分析中。在N元语言模型中，假设第t个词的出现仅依赖于前N-1个词：

$$P(w_t|w_1, w_2, ..., w_{t-1}) \approx P(w_t|w_{t-N+1}, ..., w_{t-1})$$

上述N即N元模型中的“N”，表示在给定的上下文中考虑的词或字符的数量。常见的有unigram（1-gram，只考虑单个词）、bigram（2-gram，考虑前一个词来预测当前词）以及trigram（3-gram，考虑前两个词来预测当前词）。

模型训练

N元语言模型的训练包括以下步骤：

1. N元频次统计

首先对收集的文本或已经准备好的语料库进行数据预处理，确保文本的干净和规范，并在进行分词后形成词元。

然后提取N元并进行频次统计，对于N元模型，需要提取单词元频数、二元组频率一直到N元组频率。

2. 概率表达式定义

对于下一个预测的词元，N元语言模型条件概率的一般表达式为：

$$P(w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-N+1}) = \frac{C(w_{t-N+1}, w_{t-N+2}, \ldots, w_t)}{C(w_{t-N+1}, w_{t-N+2}, \ldots, w_{t-N})}$$

其中 w_t 是当前词， $w_{t-1}, w_{t-2}, \ldots, w_{t-N+1}$ 是当前词之前的 $N-1$ 个词，
$C(w_{t-N+1}, w_{t-N+2}, \ldots, w_t)$ 表示这组N元在训练语料中出现的频次，
$C(w_{t-N+1}, w_{t-N+2}, \ldots, w_{t-N})$ 表示前 $N-1$ 个词的联合频次。

3. 最大似然估计

似然函数是用来描述在给定参数下，观察到的数据的概率。而我们希望构建的N元模型能够在训练的语料库上表现的很好，因此需要最大化给定参数下观测到该语料库中语言序列的概率。可定义为：

$$L(\theta) = \prod_{i=1}^{M} P(w_i|w_{i-1}, w_{i-2}, \ldots, w_{i-N+1})$$

于bigram模型，似然函数可以表示为：

$$L(\theta) = \prod_{i=1}^{N} P(w_i|w_{i-1})$$

然后对上述似然函数进行最大似然估计，其是一种统计方法，用于估计模型参数，使得在给定观测数据的情况下，模型生成这些数据的概率最大。往往会先对上述似然函数进行取log对数化，然后对模型参数求导，得到似然函数的极大值。

4. 概率表生成

将计算得到的条件概率汇总到一个概率表中，该表包含所有N元组合及其对应的概率值。概率表可以使用字典、哈希表或数据库来存储，以便于后续的检索和使用。

模型预测

模型训练完成后我们将得到一张概率表。当提供N-1个词作为上下文时，根据上下文在概率表中查找所有可能的下一个词及其对应概率，然后再根据概率选择下一个词，通常可以采用最大概率选择或随机采样（加权）。

模型评估——困惑度

N元语言模型的性能评估指标通常采用困惑度，用于衡量模型对给定文本序列的预测能力。其基本思想是，困惑度越低，模型的预测能力越强，意味着它能够更好地理解和生成语言。

困惑度是对语言模型在给定测试集上的不确定性度量。数学上定义为：

$$\text{Perplexity}(P) = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 P(w_i)}$$

其中N 是测试集中的词汇总数，是 $P(w_i)$ 模型对第 iii 个词的预测概率。

直观上看，“困惑度可以视为模型在每一步预测中面临的选择数量的指数值”。较低的困惑度意味着模型在预测中有较高的确定性，反之则表示更多的选择和更大的不确定性。该评估指标的取值范围通常在1到无穷大之间。理想情况下，困惑度越接近1，表示模型越好。

从另一个角度，“困惑度实际上也可以被视为语言的加权平均分支因子”[2]。分支因子指的是在生成序列时，模型在每一步选择下一个词时可能的选项数量。在计算困惑度时，实际上是在考虑每个词的预测概率。通过对这些概率取对数并求平均，我们得到了一个反映模型在所有可能选择中平均不确定性的值。因此，困惑度在某种程度上是对这些选项的加权平均，权重由每个词的概率决定。

还可以从“困惑度与熵的关系”来看，熵是用来衡量随机变量不确定性的度量。在语言模型中，熵可以描述模型生成下一个词的平均不确定性。而困惑度则是熵的指数化形式，反映了模型在生成文本时的选择复杂性。下面的公式从左到右分别表示熵和困惑度，可以清楚地看到他们之间的指数关系。

$$H(X) = -\sum_{i} P(w_i)\log_2 P(w_i) \quad \text{——>} \quad \text{Perplexity}(P) = 2^{H(X)}$$

模型改进

原始N元语言模型在实际应用中面临的首要局限性是数据稀疏性问题。在训练数据中，尤其是在较大的词汇表中，许多N元组合可能未出现。这意味着在预测新词时，模型无法给出准确的概率，导致生成文本时频繁出现未见过的组合，影响模型的整体效果。为了缓解这一问题，平滑技术应运而生。通过引入平滑技术，比如拉普拉斯平滑或Kneser-Ney平滑[3]，分别采用了不同的分配策略为未出现的N元组合赋予一定的概率，使得模型能够更合理地处理未知词汇。

另外，N元模型在上下文处理上也存在一定不足。模型仅考虑固定数量的前文（即N-1个词），无法捕捉更长的上下文信息。这意味着当句子结构复杂或依赖于长距离信息时，模型的预测能力会显著下降。例如，在长句子中，前面几个词可能无法有效指引后续词的选择。为了解决这一问题，已经有研究者提出了层次化N元模型和缓存机制等方法，增强模型对长距离依赖的捕捉能力，通过考虑更长的上下文，提升整体的语义理解。

原始N元模型不考虑词与词之间的语法关系和短语结构，这使得生成的文本在语法上可能不够自然，容易导致生成内容的流畅性差。可以考虑结合语法分析器或使用结构化N元模型的方法，引入句法信息，使得模型不仅关注词的顺序，还能理解词汇间的关系，从而提高生成文本的质量。

随着N值的增加，模型需要存储和计算的参数数量迅速膨胀，导致存储和计算开销大。这使得在资源受限的环境中应用N元模型变得更加困难。为了解决这个问题，研究提出低秩近似和参数剪枝技术，通过减少不必要的参数，优化模型的复杂度，降低存储需求，从而使得N元模型能够在较小的设备上高效运行。

本章问题

无论是N元语言模型还是后续发展的预训练语言模型乃至如今的大语言模型，其本质仍然是语言模型，即一种基于联合概率分布建模的概率模型。我们通过语料库来对这些统计模型进行训练，模型似乎可以很好的预测写一个词元（从目前大语言模型的效果来看）。我很好奇这种基于统计的方法得到的模型有没有掌握我们所提供的语料库中的知识，还是说其只是学习到了语料库中文本表面的表达形式，并没有对其内部知识进行理解？

总结

本章首先介绍了语言模型，并以其中的N元语言模型为代表进行了详细的介绍，包括其训练、预测评估和一些采用平滑方法的改进。N元语言模型简单，但非常具有代表性，其一般思想体现了当今统计语言模型的基本思路。

参考文献

[1] https://arxiv.org/abs/2303.18223v12

[2] https://web.stanford.edu/~jurafsky/slp3/3.pdf

[3] https://arxiv.org/pdf/1807.03583