Computer version

Natural Language Processing_ Language Models

自然语言处理（NLP）与语言模型

处理自然语言的能力

1. **人与其他物种的区别**：

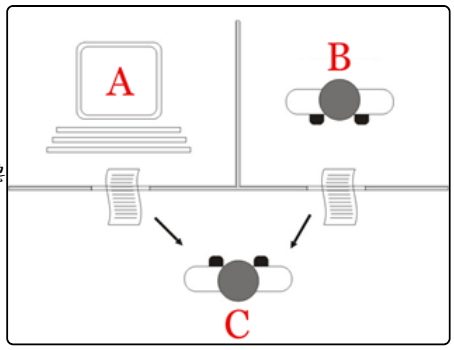
   - 人类能够通过自然语言进行交流和获取信息，这是区分人类与其他物种的重要特征。

2. **与图灵测试的关系**：

   - 自然语言处理是**图灵测试**的基础，图灵测试评估机器是否能够表现出与人类智能行为相当的能力。

3. **为什么需要这种能力？**

   - 机器需要



   - **获取信息**：例如，从文章中提取有用的数据。
   - **进行交流**：例如，聊天机器人用自然对话回答你的问题。

通过NLP实现知识获取

1. **理解含糊的语言**：

   - 自然语言经常模糊且依赖上下文，机器需要像人类一样理解这些复杂性，才能准确处理语言。

2. **信息检索任务**：

   - **文本分类**：比如将电子邮件分类为垃圾邮件或正常邮件。
   - **信息检索**：搜索引擎根据你的问题找到最相关的网页。
   - **信息抽取**：从新闻中提取关键信息，例如人名、地点和日期。

语言模型的作用

1. **什么是语言模型？**

   - 语言模型预测句子或上下文中**单词的概率分布**。
   - 例子：在句子 “The cat sat on the ___” 中，语言模型可能预测下一个单词是 “mat”，并赋予其较高的概率。

2. **语言模型的用途**：

它们在许多任务中起着辅助的自动补全功能）：

- **生成连贯的文本**
- **理解上下文**（如翻译或文本摘要）。

> 🌐 语言模型的实际生活例子
>
> 想象你在手机上输入：
>
> - 当你打出“我感觉”后，手机可能会建议“开心”或“兴奋”。
> - 这些建议来源于语言模型，它基于成千上万句子中学到的语言模式。

语言的特性

1. **无限的可能性**：
   - 语言中可以构造出无限多的句子（字符串），因此无法穷举所有可能。
   - 这些可能性由规则定义，即**语法（grammar）**。
2. **自然语言的挑战**：
   - **不确定性（Uncertainty）**：
     - 某些句子是否属于语法规则的范围是有争议的。例如，句子 "Not to be invited is sad" 是否符合英语语法，可能存在不同意见。
   - **歧义性（Ambiguity）**：
     - 自然语言句子常常有多种含义。例如，句子 "He saw a man with a telescope" 可以有两种解释：
       1. 他用望远镜看到了一个人。
       2. 他看到了一个带望远镜的人。

语言模型的概率分布

由于自然语言具有不确定性和歧义性，我们使用**概率分布（probability distributions）**来处理这些问题。

1. **处理不确定性**：
   - 对于句子 "Not to be invited is sad"，语言模型会计算它在英语中的出现概率，以此评估它的合理性。
2. **处理歧义性**：
   - 对于句子 "He saw a man with a telescope"，语言模型会为每种可能的含义分配概率。

自然语言的复杂性

1. **规模大**：
   - 自然语言词汇量庞大，句子结构复杂，覆盖了大量的上下文和语境。
2. **动态变化**：
   - 自然语言是不断演化的，新词、新表达层出不穷。

因此，语言模型只能对语言做**近似（approximation）**处理。

🧩 假设你输入一段文字：

- 输入："The weather is"

- 模型可能预测：
  - "nice"（概率 60%）
  - "bad"（概率 30%）
  - "cold"（概率 10%）

语言模型根据大量文本数据学习的概率分布，选择最可能的输出。这种机制使得语言模型可以完成从自动补全到翻译、生成文章等多种任务。

N-gram Character Models

什么是N-gram模型？

N-gram是指一段连续的字符或单词序列，用于预测自然语言中出现字符或单词的概率分布。N-gram模型常被用来简化语言建模问题。

1. **核心概念**：
   - N-gram 是一个长度为 n 的字符或单词序列。例如：
     - 1-gram（unigram）：单个字符或单词。
     - 2-gram（bigram）：两个连续的字符或单词。
     - 3-gram（trigram）：三个连续的字符或单词。
   - 模型目标：基于统计分布预测某个字符或单词在一段上下文中的出现概率。
2. **符号说明**：
   - $P(C_{1:N})$ 表示一段长度为 N 的字符序列的概率分布。例如：
     - $P(\text{'t', 'h', 'e'}) = 0.027$ 表示字符序列 "the" 出现的概率为 0.027。
     - $P(\text{'z', 'g', 'q'}) = 0.00000002$ 表示字符序列 "zgq" 几乎不可能出现。

N-gram模型的运作机制

1. **基于Markov假设**：
   - Markov链假设：序列中某个字符的概率只依赖于其前 $n-1$ 个字符。
   - 例如，对于三元模型（trigram），假设当前字符 c_i 的概率只与前两个字符 c_{i-2} 和 c_{i-1} 有关：

$$P(c_i \mid c_{1:i-1}) \approx P(c_i \mid c_{i-2}, c_{i-1})$$

2. **概率估计**：
   - 利用大量文本数据（称为语料库，corpus）统计不同字符序列的出现次数来估计概率。
   - 例如，计算字符 $'e'$ 在 $'t' \rightarrow 'h'$ 后出现的概率：

$$P(\text{'e'} \mid \text{'th'}) = \frac{\text{Count('the')}}{\text{Count('th')}}$$

-

例子：N-gram在实际中的应用

假设语料库中有以下文本：`"the quick brown fox jumps over the lazy dog"`，我们可以：

1. 使用2-gram（bigram）模型生成字符序列：

- "th" 的概率是 $\frac{\text{Count('th')}}{\text{Count('t')}}$.
- "he" 的概率是 $\frac{\text{Count('he')}}{\text{Count('h')}}$.

2. 利用这些概率生成新文本，或为现有文本分配概率.

N-gram的优点与局限性

1. **优点**:
   - 简单易懂，基于统计的方法实现语言建模.
   - 对短文本和结构化文本效果较好.
2. **局限性**:
   - **数据稀疏性问题**: 某些N-gram可能在语料库中从未出现，导致概率为零.
   - **上下文有限性**: 无法捕捉长距离的依赖关系.

N-gram Word Models

什么是N-gram Word模型？

- N-gram Word模型基于单词序列的概率分布进行语言建模.
- 核心思想是将上下文限制为某个固定数量的单词，从而简化计算复杂度.

1. **定义**:
   - **Bigram（2-gram）**：两个连续单词的序列，如 "please wait", "wait for".
   - **Trigram（3-gram）**：三个连续单词的序列，如 "please wait for", "wait for your".
2. **用途**:
   - 预测下一个单词的概率，给定上下文. 例如:
     - 给定 h = “Walden Pond's water is so transparent that”，模型预测下一个单词 w = “the”.

N-gram Word模型的概率计算

1. **条件概率的公式**:
   - 目标是计算下一个单词的概率:

$$P(\text{the} \mid \text{Walden Pond's water is so transparent that}) = \frac{\text{Count(Walden Pond's wat}}{\text{Count(Walden Pond's w}}$$

2. **问题**:
   - 长序列的上下文在语料库中可能极少出现，导致统计次数不足.
3. **解决方案**:
   - 使用Bigram模型，简化为计算前一个单词的条件概率:

$$P(w_n \mid w_{1:n-1}) \approx P(w_n \mid w_{n-1})$$

   - 例如:

$$P(\text{the} \mid \text{that}) = \frac{\text{Count}(\text{that the})}{\text{Count}(\text{that})}$$

实际案例分析

假设有一句话：`"I do not like green eggs and ham"`

1. **Bigram模型（$n = 2$）**：
   - 目标是计算 $P(\text{ham} \mid \text{and})$：

$$P(\text{ham} \mid \text{and}) = \frac{\text{Count}(\text{and ham})}{\text{Count}(\text{and})}$$

2. **Trigram模型（$n = 3$）**：
   - 目标是计算 $P(\text{ham} \mid \text{eggs and})$：

$$P(\text{ham} \mid \text{eggs and}) = \frac{\text{Count}(\text{eggs and ham})}{\text{Count}(\text{eggs and})}$$

N-gram Word模型的优缺点

1. **优点**：
   - 简单且易于实现。
   - 在小规模语料库中表现良好。
2. **缺点**：
   - **数据稀疏性**：某些N-gram组合可能在语料库中未出现，导致概率为零。
   - **长距离依赖问题**：无法捕获远距离单词之间的依赖关系。

N-gram Word Models: Vocabulary and Applications

N-gram Word模型的挑战

1. **词汇量（Vocabulary）**：
   - 与字符模型相比，单词模型需要处理的词汇量大得多：
     - **字符模型**：语言中通常只有大约 100 个左右的唯一字符。
     - **单词模型**：需要处理数万到数百万的单词。
   - **问题**：
     - 什么才算是一个单词？例如，"U.S." 是一个单词还是三个字符？
     - 不同语言或文本格式可能有不同的单词边界（word boundaries）。
2. **超出词汇表的单词（Out-of-Vocabulary Words, OOV）**：
   - 模型可能遇到训练语料库中未出现的新单词。
   - 解决方法：引入占位符 `<UNK>`，用来表示任何未知单词。

处理OOV单词的机制

1. **显式建模**：
   - 在训练语料库中，将第一次出现的未知单词替换为 `<UNK>`。
   - 对 `<UNK>` 进行统计，计算它的出现频率和与其他单词的关系。
2. **Fallback机制**：
   - 在测试语料库中，任何未见过的单词都会被映射到 `<UNK>`。
   - 这样，模型可以基于 `<UNK>` 的概率分布继续工作，而不会因为未见过的单词而失败。

N-gram模型的应用

N-gram模型的简单性和高效性使其被广泛应用于以下领域：

1. **语言识别（Language Identification）**：
   - 给定一段文本，识别它属于哪种自然语言（如英语、中文）。
   - 例如，通过分析字符或单词序列的分布来区分法语和西班牙语。
2. **拼写检查与纠正（Spelling Correction）**：
   - 根据上下文预测最可能的字符或单词序列。
   - 例如，将 "hte" 自动更正为 "the"。
3. **文体分类（Genre Classification）**：
   - 判断文本是新闻、科学论文还是推文。
   - 根据N-gram分布特征，区分不同的写作风格。
4. **命名实体识别（Named Entity Recognition, NER）**：

> a total of 41 chemical compounds, including 4 flavone-C-glycosides, 7 flavonoid-O-glycosides and 19 polymethoxyflavones were unambiguously identified or tentatively characterized in CRP. The occurrence of 1 flavone-C-glycoside and 3 cyclic peptides in particular has not yet been described.

   - 从文本中识别专有名词，如人名、地名或组织名。
   - 例如，从一段描述化学成分的文本中提取特定化学名称。