Natural Language Processing I

10. 子词嵌入

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

2022/04/22

子词

分词策略

之前：手工指定规则

- 空格、标点分隔；汉字、词组

分词策略

之前：手工指定规则

- 空格、标点分隔；汉字、词组

机器学习：从文本数据训练，分析、提取规则

- 统计经常出现的子字符串：看成**固定组合**
  - 将所有固定组合添加至词汇表

分词策略

之前：手工指定规则

- 空格、标点分隔；汉字、词组

机器学习：从文本数据训练，分析、提取规则

- 统计经常出现的子字符串：看成**固定组合**
  - 将所有固定组合添加至词汇表
- 可以解决测试文本含有**未知单词**的问题

分词策略

之前：手工指定规则

- 空格、标点分隔；汉字、词组

机器学习：从文本数据训练，分析、提取规则

- 统计经常出现的子字符串：看成**固定组合**
  - 将所有固定组合添加至词汇表
- 可以解决测试文本含有**未知单词**的问题

例如：

```
训练文本：low, new, newer
测试文本：lower
```

子词分解

现代分词器把单词分解成更小的**子词 subwords**

- 子词可以是任意子字符串，或是有意义的词素
- **词素 morpheme**：最小意义单元
  - 例如：unlikeliest -> un-, likely, -est

子词分解

现代分词器把单词分解成更小的**子词 subwords**

- 子词可以是任意子字符串，或是有意义的词素
- **词素 morpheme**：最小意义单元
  - 例如：unlikeliest -> un-, likely, -est

现代分词机制包含两个部分：

- **学习器 learner**：从训练文本中提取词汇表，即字典
- **分词器 segmenter**：根据词汇表完成分词，如双向最长匹配等

Byte-Pair 编码器

[Sennrich 2016] **Byte-Pair 编码器**：相对简单的贪心算法

1. 词汇表初始化为所有出现过的单独字母；
2. 重复 k 次：
   1. 找出频率最高的相邻单元 $\alpha\beta$；
   2. 将 $\alpha\beta$ 组合成 π，并加入到词汇表中；
   3. 将训练文本中所有出现的 $\alpha\beta$ 替换成 π；

`BYTE-PAIR`(C, k)

1. $V = \{A, B, .., Z, a, b, .., z\} \cap C$;
2. FOR $i = 1..k$:
   1. $\pi = CAT(\arg\max_{\alpha\beta \in C} FREQ(\alpha\beta, C))$;
   2. $V.JOIN(\pi)$;
   3. $C.SUBS(\alpha\beta, \pi)$;

文本预处理

通常子词分解算法用于**空格分隔**的语言

- 首先将语料库分隔成单词
- 在每个单词之后添加词**结束符** _

文本预处理

通常子词分解算法用于**空格分隔**的语言

- 首先将语料库分隔成单词
- 在每个单词之后添加词**结束符** _

例如语料库：

```
low low low low low lowest lowest
newer newer newer newer newer newer
wider wider wider new new
```

文本预处理

通常子词分解算法用于**空格分隔**的语言

- 首先将语料库分隔成单词
- 在每个单词之后添加词**结束符** _

例如语料库：

```
low low low low low lowest lowest
newer newer newer newer newer newer
wider wider wider new new
```

统计词频，并构建词汇表：

```
5   l o w _
2   l o w e s t _
6   n e w e r _
3   w i d e r _
2   n e w _
```

```
_, d, e, i, l, n, o, r, s, t, w
```

BPE 学习器 I

```
5   l o w _
2   l o w e s t _
6   n e w e r _
3   w i d e r _
2   n e w _
```

```
_, d, e, i, l, n, o, r, s, t, w
```

- 将 e r 合并为 er

```
5   l o w _
2   l o w e s t _
6   n e w er _
3   w i d er _
2   n e w _
```

```
..., er
```

BPE 学习器 II

```
5   l o w _
2   l o w e s t _
6   n e w er _
3   w i d er _
2   n e w _
```

```
..., er
```

- 将 er _ 合并为 er_

```
5   l o w _
2   l o w e s t _
6   n e w er_
3   w i d er_
2   n e w _
```

```
..., er, er_
```

BPE 学习器 III

```
5   l o w _
2   l o w e s t _
6   n e w er_
3   w i d er_
2   n e w _
```

```
..., er, er_
```

- 将 n e 合并为 ne

```
5   l o w _
2   l o w e s t _
6   ne w er_
3   w i d er_
2   ne w _
```

```
..., er, er_, ne
```

BPE 学习器 IV

以此类推

```
(ne, w)
(l, o)
(lo, w)
(new, er_)
(low, _)
```

```
..., er, er_, ne, new
..., er, er_, ne, new, lo
..., er, er_, ne, new, lo, low
..., er, er_, ne, new, lo, low,
newer_
..., er, er_, ne, new, lo, low,
newer_, low_
```

- 将最后一步构造出的词汇表输出

BPE 分词器

首先将测试文本分句，然后：

1. 每个句子**拆分为单独的字符**
2. 按照词汇表的顺序**合并字符串**：词频反映置信度

当前词汇表：`..., er, er_, ne, new, lo, low, newer_, low_`

- `n e w e r _ -> newer_`
- `l o w e r _ -> low er_`

BPE 分词器

首先将测试文本分句，然后：

1. 每个句子**拆分为单独的字符**
2. 按照词汇表的顺序**合并字符串**：词频反映置信度

当前词汇表：`..., er, er_, ne, new, lo, low, newer_, low_`

- `n e w e r _` -> `newer_`
- `l o w e r _` -> `low er_`

因此，测试文本中的词频不重要

实验：BPE 分词器