小作业1：语言模型PPL计算

PPL计算过程

首先给出PPL计算公式：

$$PPL = 10^{-\frac{1}{K}\sum_{k=1}^{K} log_{10} P(\omega_k|W_{k-n+1}^{k-1})}$$

其中，

$$\sum_{k=1}^{K} log_{10} P(\omega_k|W_{k-n+1}^{k-1}) = log_{10} P(\omega_1|\omega_0) + log_{10} P(\omega_2|\omega_0,\omega_1) + \cdots + log_{10} P(\omega_k|\omega_{k-1},\omega_{k-2})$$

如果模型中没有n-gram的记录，则要相应进行backoff

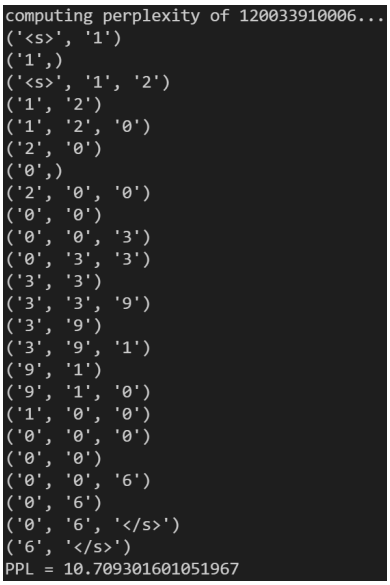
计算结果

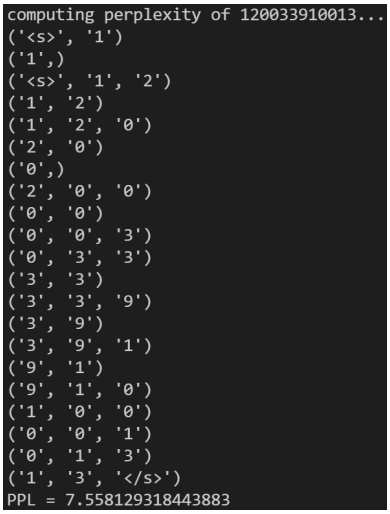
1. 021033210023：PPL=12.19325542238174
2. 019033910051：PPL=12.431021282846018
3. 120033910006：PPL=10.709301601051967
4. 120033910013：PPL=7.558129318443883

附录1 计算过程详细信息

```
computing perplexity of 021033210023...
('<s>', '0')
('0',)
('<s>', '0', '2')
('0', '2')
('0', '2', '1')
('2', '1', '0')
('1', '0')
('1', '0', '3')
('0', '3', '3')
('3', '3')
('3', '3', '2')
('3', '2')
('2',)
('3', '2', '1')
('2', '1')
('2', '1', '0')
('1', '0')
('1', '0', '0')
('0', '0', '2')
('0', '2')
('0', '2', '3')
('2', '3')
('3',)
('2', '3', '</s>')
('3', '</s>')
PPL = 12.19325542238174
```

```
computing perplexity of 019033910051...
('<s>', '0')
('0',)
('<s>', '0', '1')
('0', '1')
('0', '1', '9')
('1', '9')
('1', '9', '0')
('9', '0', '3')
('0', '3')
('0', '3', '3')
('3', '3')
('3', '3', '9')
('3', '9')
('3', '9', '1')
('9', '1')
('9', '1', '0')
('1', '0', '0')
('0', '0', '5')
('0', '5')
('0', '5', '1')
('5', '1')
('5', '1', '</s>')
('1', '</s>')
PPL = 12.431021282846018
```





附录2 计算过程使用的代码：

```
# compute PPL for pre-trained n-gram model
# For any question or problem, please feel free to contact:
# Email: douyiming@sjtu.edu.cn
# Wechat: 18017112986

# load model
model_path = './data/cs382_1.arpa'
data = {}
with open(model_path, 'r') as f:
    raw_data = f.readlines()
```

```
    data['uni'] = raw_data[7:19]
    data['bi'] = raw_data[21:98]
    data['tri'] = raw_data[100:142]
for k, v in data.items():
    data[k] = [s.split() for s in v]
seq2info = {}  # {seq: {"log_p": xxx, "backoff": xxx}}
for k, v in data.items():
    for l in v:
        if k == 'uni':
            seq = (l[1])
            info = {"log_p": float(l[0])}
            info["backoff"] = float(l[2]) if len(l) == 3 else 0
            seq2info[seq] = info
        elif k == 'bi':
            seq = (l[1], l[2])
            info = {"log_p": float(l[0])}
            info["backoff"] = float(l[3]) if len(l) == 4 else 0
            seq2info[seq] = info
        elif k == 'tri':
            seq = (l[1], l[2], l[3])
            info = {"log_p": float(l[0])}
            info["backoff"] = float(l[4]) if len(l) == 5 else 0
            seq2info[seq] = info


def p(seq):
    print(tuple(seq))
    l = len(seq)
    assert 1 <= l and l <= 3
    if l == 1:
        info = seq2info.get(seq)
    elif l == 2:
        info = seq2info.get(seq)
        if info == None:  # unseen
            return p(seq[1])
    else:
        info = seq2info.get(seq)
        if info == None:  # unseen
            return p(seq[1:])
    return info['log_p']+info['backoff']


def ppl(seq):
    # computes the perplexity of a sequence
    seq_ex = ['<s>']+[s for s in seq]+['</s>']
    ans = p(tuple(seq_ex[0:2]))
    for i in range(0, len(seq_ex)-2):
        ans += p(tuple(seq_ex[i:i+3]))
    ans = pow(10, -1/len(seq_ex)*ans)
    return ans


seq = ['021033210023', '019033910051', '120033910006', '120033910013']
ans = {}
```

```
for s in seq:
    print("computing perplexity of {}...".format(s))
    print("PPL = {}".format(ppl(s)))
```