assignment_1_B

April 6, 2025

1 ΕΠΕΞΕΡΓΑΣΙΑ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ - Ε 1

1.1 B. N-gram Language Models

Μ : Ε Φ Γ

Σ : Ι Κ

Ε : 2025-03-22 | v.0.0.1

1.1.1 Β 1: Φ Δ Corpus

Χ corpus `treebank` NLTK, 199 Wall Street Journal.
Δ :

- **170** (training set).
- **29** (test set).

Τ corpus tokens .

[3]:
```
import nltk
from nltk.corpus import treebank
from collections import Counter

#  Κ     corpus treebank
nltk.download('treebank')

#  Λ       199
files = treebank.fileids()

#  Δ        train   test
train_files = files[:170]
test_files = files[170:]

#  Ε
train_sents = [sent for file in train_files for sent in treebank.sents(file)]
test_sents = [sent for file in test_files for sent in treebank.sents(file)]

print(f"Π               : {len(train_sents)}")
print(f"Π               : {len(test_sents)}")
```

```
[nltk_data] Downloading package treebank to
[nltk_data]     /Users/ioanniskoutsoukis/nltk_data…
[nltk_data]   Unzipping corpora/treebank.zip.
```

```
Π               : 3509
Π               : 405
```

1.1.2 B 2: Δ Λ A <UNK>

ϒ tokens **L**:

- T tokens **3** <UNK>.
- T **L**.

[4]:
```
#  Σ            tokens
train_tokens = [token for sent in train_sents for token in sent]

#  M
freqs = Counter(train_tokens)

#  K          :      ,            3
vocab = {word for word, count in freqs.items() if count >= 3}

#  Σ             <UNK>
def replace_with_unk(sent, vocab):
    return [token if token in vocab else "<UNK>" for token in sent]
```

1.1.3 B 3: Π Π <BOS> <EOS>

K tokens:

- <BOS> .
- <EOS> .

A - . H <UNK> tokens.

[5]:
```
def prepare_sentences(sents, vocab):
    result = []
    for sent in sents:
        replaced = replace_with_unk(sent, vocab)
        result.append(['<BOS>'] + replaced + ['<EOS>'])
    return result

#  T                      n-grams
train_prepared = prepare_sentences(train_sents, vocab)
test_prepared = prepare_sentences(test_sents, vocab)
```

1.1.4 B 4: E E

[11]:
```
print(train_sents[0])
print(train_prepared[0])

print("Pierre      :", freqs["Pierre"])
print("Vinken      :", freqs["Vinken"])
```

```
['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'will', 'join', 'the',
'board', 'as', 'a', 'nonexecutive', 'director', 'Nov.', '29', '.']
['<BOS>', '<UNK>', '<UNK>', ',', '61', 'years', 'old', ',', 'will', 'join',
'the', 'board', 'as', 'a', 'nonexecutive', 'director', 'Nov.', '29', '.',
'<EOS>']
Pierre      : 1
Vinken      : 2
```

1.1.5 B 5: Π Σ Π (E 1)

[15]:
```
from collections import Counter
import math
import re

# O                          n-grams
def extract_ngrams(sentences, n):
    """
    Δ              (                    tokens)
                n-grams                    .
    """
    ngrams = []
    for sent in sentences:
        ngrams.extend([tuple(sent[i:i+n]) for i in range(len(sent)-n+1)])
    return ngrams


# Lowercase
def to_lowercase(sents):
    return [[token.lower() for token in sent] for sent in sents]

# Abstract digits
def abstract_digits(sents):
    def abstract_token(token):
        return re.sub(r'\d', '#', token)
    return [[abstract_token(token) for token in sent] for sent in sents]


# E      n-gram        add-k smoothing
class NgramModel:
    def __init__(self, n, k, train_data):
```

```
        """
        n:         n-grams (2    bigram, 3    trigram)
        k:               (add-k)
        train_data:              tokens (          <UNK>, <BOS>, <EOS>)
        """
        self.n = n
        self.k = k
        self.train_data = train_data

        # Σ    n-grams    context ( . . bigram: (w1, w2), context: w1)
        self.ngrams = Counter(extract_ngrams(train_data, n))
        self.context = Counter(extract_ngrams(train_data, n - 1))

        # Λ      V (            smoothing)
        self.vocab = set(token for sent in train_data for token in sent)

    def prob(self, ngram):
        """
        ϒ               add-k smoothing
        """
        prefix = ngram[:-1]
        return (self.ngrams[ngram] + self.k) / (self.context[prefix] + self.k *␣
↪len(self.vocab))

    def perplexity(self, test_data):
        """
        ϒ      perplexity
        """
        ngrams_test = extract_ngrams(test_data, self.n)
        N = len(ngrams_test)
        log_sum = 0
        for ngram in ngrams_test:
            p = self.prob(ngram)
            log_sum += math.log(p)
        return math.exp(-log_sum / N)

# Σ                        (original, lowercase, digits)
def evaluate_all_versions(train_raw, test_raw):
    versions = {
        "Original text": (train_raw, test_raw),
        "Lowercase": (to_lowercase(train_raw), to_lowercase(test_raw)),
        "Abstract digits": (abstract_digits(train_raw),␣
↪abstract_digits(test_raw))
    }

    settings = [(2, 1), (2, 0.01), (3, 1), (3, 0.01)]
    results = {v: {} for v in versions}
```

```
    for version, (train, test) in versions.items():
        for n, k in settings:
            model = NgramModel(n, k, train)
            ppl = model.perplexity(test)
            results[version][f'{n}-gram (k={k})'] = round(ppl, 2)

    return results

evaluate_all_versions(train_prepared, test_prepared)
```

[15]:
```
{'Original text': {'2-gram (k=1)': 383.74,
  '2-gram (k=0.01)': 137.84,
  '3-gram (k=1)': 1505.81,
  '3-gram (k=0.01)': 464.06},
 'Lowercase': {'2-gram (k=1)': 349.46,
  '2-gram (k=0.01)': 130.43,
  '3-gram (k=1)': 1374.44,
  '3-gram (k=0.01)': 427.07},
 'Abstract digits': {'2-gram (k=1)': 336.26,
  '2-gram (k=0.01)': 122.12,
  '3-gram (k=1)': 1319.79,
  '3-gram (k=0.01)': 386.26}}
```

1.1.6 Π & Σ Perplexity

Language model	Original text	Lowercase	Abstract digits
Bigrams (k = 1)	383.74	349.46	336.26
Bigrams (k = 0.01)	137.84	130.43	122.12
Trigrams (k = 1)	1505.81	1374.44	1319.79
Trigrams (k = 0.01)	464.06	427.07	386.26

- **Π** : T **bigrams** (2-grams) **k = 0.01** **perplexity** .
- **T** **k**: H **k = 0.01** k = 1, n-grams, perplexity.
- **E** **lowercase**: H (lowercase) . A , (. . `The` `the`), .
- **E** **abstract digits**: H (`#`) **perplexity**. O (hapax legomena), .

1.1.7 B 6: П П (E 2)

[29]:
```
import random

def generate_sentence(model, max_length=30, verbose=False):
    """
    Δ                          n-gram.
    E                               .
    - model:        NgramModel
    - max_length:
    - verbose:       True,
    """
    sentence = ['<BOS>']

    while len(sentence) < max_length:
        context = tuple(sentence[-(model.n - 1):]) if model.n > 1 else tuple()

        # Г          tokens          (    <UNK>)
        candidates = [w for w in model.vocab if w != '<UNK>']

        # Г
        probs = []
        for word in candidates:
            ngram = context + (word,)
            prob = model.prob(ngram)
            probs.append(prob)

        # K
        total = sum(probs)
        probs = [p / total for p in probs]

        # T
        next_word = random.choices(candidates, weights=probs, k=1)[0]

        if verbose:
            print(f"Context: {context} -> E    : '{next_word}'␣
 ↪(Prob={round(max(probs), 4)})")

        sentence.append(next_word)

        if next_word == '<EOS>':
            break

    return ' '.join(sentence[1:-1])  #       <BOS>, <EOS>

#   Σ
combinations = [
```

```
    (2, 1),      # Bigram, k=1
    (2, 0.01),   # Bigram, k=0.01
    (3, 1),      # Trigram, k=1
    (3, 0.01)    # Trigram, k=0.01
]

#  П
for n, k in combinations:
    print(f"\n M    {n}-gram   k={k}:\n" + "-"*50)
    model = NgramModel(n, k, train_prepared)
    for i in range(1, 4):  # 3
        sentence = generate_sentence(model)
        print(f"* П    {i}: <BOS> {''.join(sentence)} <EOS>")
```

```
 M    2-gram   k=1:
--------------------------------------------------
* П    1: <BOS> heart stopped medical Carla Carnival Old total start On globe
Co plastic foreign number On Says Entertainment Sen. surprisingly Canada
compound nearly Mercantile complex directly 51 equipment chemicals <EOS>
* П    2: <BOS> 8.50 N.J eager where remove plunged employees national gives
Two suffer gains seeing Several Markey basic roof-crush sense Price disciplinary
acne providing century as looming 13 conference few <EOS>
* П    3: <BOS> `` agreement widget people flat section 41 retired 30 So
bridge covered multi-crystal turns In wait estimated household won standard
expire Reupke becoming Money Co large front Computer <EOS>

 M    2-gram   k=0.01:
--------------------------------------------------
* П    1: <BOS> In her , a share . <EOS>
* П    2: <BOS> By earned fueled Trudeau refund card business appears US$
margins approach it to help *-1 always held media withdrawal year ending
resolution of two funded premium 9 1\/2 <EOS>
* П    3: <BOS> And 47 employer looming received letters A. commodity markets
Dorrance 2029 200 represent quarterly courts underwriters General 22\/32
Freeport-McMoRan Lake working Fairless mature fell agency live unlike stand
<EOS>

 M    3-gram   k=1:
--------------------------------------------------
* П    1: <BOS> 70 court where expressed federal 40 figures * Klauser Law %
made Group editorial Morgan simply Ms. court ship financings Although ones
himself 12 St. 33 French During <EOS>
* П    2: <BOS> great produce understand Profit Poor pretax as run scheduled
ballplayers suggests effective lesson index-arbitrage effects classroom family
bulk compliance remained U.S.A. prevent Johnson rules trying men Nasdaq able
<EOS>
* П    3: <BOS> lucrative Senate generally markets required respond
```

```
bankruptcy market respond themselves chairman Guinea damage staff treatment not
asks Index elected Moody industry Illuminating Singapore defense promotion *-3
fall impose <EOS>

 M    3-gram   k=0.01:
-------------------------------------------------
* Π    1: <BOS> accommodate hard Dodge race step parents less 47 compliance
disgorge Chevrolet profitable smoking activity smoking designers Many took
crisis Terms referred Richard branch larger According techniques Hahn opinion
<EOS>
* Π    2: <BOS> 30-year clients credit plastic senior district bells risks
local three decided early Article *-3 squeeze Smith publicly knew Chicago harder
Instead church liquidity disciplinary Acquisition effect big specified <EOS>
* Π    3: <BOS> nine dismissed statute rating heads debts total collected
commodity required one-third beaten political widespread large Donald 1 noted
hours 2009 hundred entered strategy quick N.C York-based charge France <EOS>
```

1.1.8 Π & ϒ Π Π Π

K n-gram (Bigram/Trigram k=1 k=0.01), :

- O .
- Π , .
- Σ , . . “smoking activity smoking designers”, .

Π :

- I n-gram (. .)
  .
- E , <BOS>
  .
- H **k=1** , corpus, n-grams.
- A **trigram** , , training corpus.

Σ , n-gram , , .