English Word Suggestion Based on Part of Speech Ngram

Hamana Laboratory, Gunma University

Borann Chanrathnak

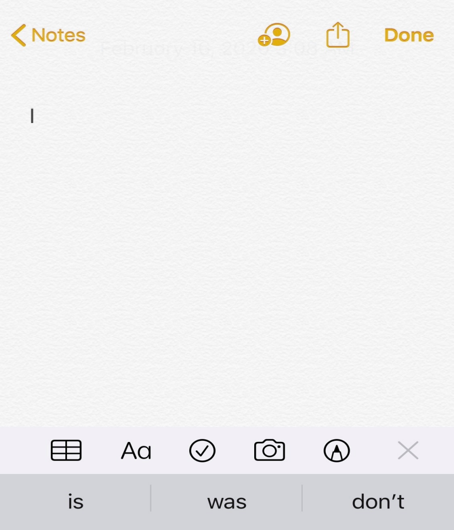
February 20, 2020

Table of Contents

Motivation

Motivation

- Word suggestion for note-taking app on mobile devices

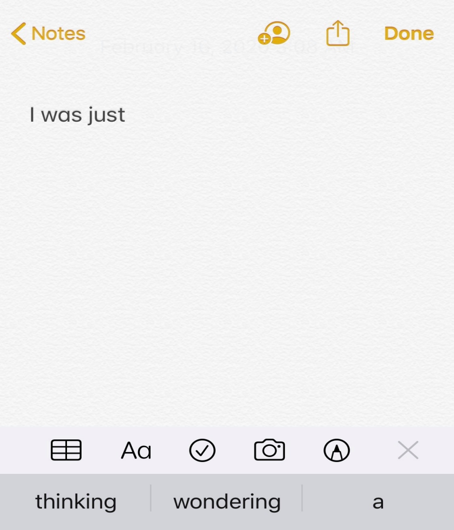


Motivation

- Word suggestion for note-taking app on mobile devices

Motivation

- Word suggestion for note-taking app on mobile devices



Part Of Speech Ngram (POS-Ngram)

What does POS-Ngram mean?

N-gram is a contiguous sequence of n items from a given sample of text or speech

Ex: I study english

- **unigram (1-gram)** : (I,), (study,), (english,)
- **bigram (2-gram)** : (I, study), (study, english)
- **trigram (3-gram)** : (I, study, english)

What does POS-Ngram mean?

N-gram is a contiguous sequence of n items from a given sample of text or speech

Ex: I study english

- **unigram (1-gram)** : (I,), (study,), (english,)
- **bigram (2-gram)** : (I, study), (study, english)
- **trigram (3-gram)** : (I, study, english)

N-gram model is one of statistical langauge models for predicting the next item (word) based on Markov assumption, and usually abbreviated as N-gram.

What does POS-Ngram mean?

N-gram is a contiguous sequence of n items from a given sample of text or speech

Ex: I study english

- **unigram (1-gram)** : (I,), (study,), (english,)
- **bigram (2-gram)** : (I, study), (study, english)
- **trigram (3-gram)** : (I, study, english)

N-gram model is one of statistical langauge models for predicting the next item (word) based on Markov assumption, and usually abbreviated as N-gram.

POS-Ngram is an improved model using part of speech as a class indicator.

How To Compute Probability of a Sentence?

How can we compute the joint probability of words in a sentence?
Ex: "an apple is on the table"

$$P(\textit{an apple is on the table}) = ?$$

Notation

Notation

Notation

Notation

- To represent the probability of a particular random variable X_i taking on the value "the", or $P(X_i = "the")$ we will use simplification $P(the)$.

Notation

Notation

- To represent the probability of a particular random variable X_i taking on the value "the", or $P(X_i = "the")$ we will use simplification $P(the)$.
- We represent a sequence of N words either as $w_1 \cdots w_n$ or w_1^n

Notation

Notation

- To represent the probability of a particular random variable X_i taking on the value "the", or $P(X_i = "the")$ we will use simplification $P(the)$.
- We represent a sequence of N words either as $w_1 \cdots w_n$ or w_1^n
- **Ex:** "an apple is on the table"
  $\rightarrow w_1 = "an", w_2 = "apple", w_3 = "is", \cdots$

Notation

Notation

- To represent the probability of a particular random variable X_i taking on the value "the", or $P(X_i = "the")$ we will use simplification $P(the)$.
- We represent a sequence of N words either as $w_1 \cdots w_n$ or w_1^n
- **Ex:** "an apple is on the table"
  $\rightarrow w_1 = "an", w_2 = "apple", w_3 = "is", \cdots$
  $\rightarrow w_1^4 = "an\ apple\ is\ on"$

Chain Rule Of Probability

In General

$$P(X_1 \cdots X_n) = P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_1^2) \cdots P(X_n \mid X_1^{n-1})$$

Chain Rule Of Probability

In General

$$P(X_1 \cdots X_n) = P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_1^2) \cdots P(X_n \mid X_1^{n-1})$$

By applying chain rule to a sequence of words

$$P(w_1 \cdots w_n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2) \cdots P(w_n \mid w_1^{n-1})$$

Chain Rule Of Probability

In General

$$P(X_1 \cdots X_n) = P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_1^2) \cdots P(X_n \mid X_1^{n-1})$$

By applying chain rule to a sequence of words

$$P(w_1 \cdots w_n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2) \cdots P(w_n \mid w_1^{n-1})$$

Examples

- $P(an\ apple\ is) = P(an) \times P(apple \mid an) \times P(is \mid an\ apple)$

Let's Apply Chain Rule

By applying chain rule to the phrase "*an apple is on the*":

$$P(\text{"}an\ apple\ is\ on\ the\text{"}) = P(an) \times P(apple \mid an) \times P(is \mid an\ apple) \\ \times P(on \mid an\ apple\ is) \times P(the \mid an\ apple\ is\ on)$$

In practice chain rule does not help

- We don't know the way to compute the exact probability of a word given a long sequence of preceding words, $P(w_n \mid w_1^{n-1})$.

The Presence of N-gram

The general equation for this n-gram approximation to the conditional probability of the next word in a sequence is

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

The Presence of N-gram

The general equation for this n-gram approximation to the conditional probability of the next word in a sequence is

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

In case of Bigram (2-gram)

When we use bigram model to predict the conditional probability of the next word, we can approximate by

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-1})$$

The Presence of N-gram

The general equation for this n-gram approximation to the conditional probability of the next word in a sequence is

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

In case of Bigram (2-gram)

When we use bigram model to predict the conditional probability of the next word, we can approximate by

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-1})$$

Example

$$P(the \mid \ an\ apple\ is\ on) \approx P(the \mid \ on)$$

Bigram in Practice

By supposing that C is the counts of a sequence of words from a corpus.

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1} w_n)}{\sum_{w \in dict} C(w_{n-1} w)}$$

Bigram in Practice

By supposing that C is the counts of a sequence of words from a corpus.

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_{w \in dict} C(w_{n-1}w)}$$

Example

$$P(the \mid \ on) = \frac{C(on\ the)}{\sum_x C(on\ x)}$$

Bigram in Practice

By supposing that C is the counts of a sequence of words from a corpus.

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_{w \in dict} C(w_{n-1}w)}$$

Example

$$P(the \mid \ on) = \frac{C(on\ the)}{\sum_x C(on\ x)}$$

$(on\ x)$ means (on the), (on board), (on time), (on that), $\cdots$

The probability of a complete word sequence

Given the **Bigram** assumption for the probability of an individual word, we can compute the probability of complete word sequence by:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1})$$

The probability of a complete word sequence

Given the **Bigram** assumption for the probability of an individual word, we can compute the probability of complete word sequence by:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1})$$

Chain Rule

$$P(\text{"}an\ apple\ is\ on\ the\text{"}) = P(an) \times P(apple \mid an) \times P(is \mid an\ apple) \times P(on \mid an\ apple\ is) \times P(the \mid an\ apple\ is\ on)$$

The probability of a complete word sequence

Given the **Bigram** assumption for the probability of an individual word, we can compute the probability of complete word sequence by:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1})$$

Chain Rule

$$\begin{aligned} P(''an\ apple\ is\ on\ the'') = & P(an) \times P(apple \mid an) \times P(is \mid an\ apple) \\ & \times P(on \mid an\ apple\ is) \times P(the \mid an\ apple\ is\ on) \end{aligned}$$

Bigram Assumption

$$\begin{aligned} P(''an\ apple\ is\ on\ the'') = & P(apple \mid an) \times P(is \mid apple) \\ & \times P(on \mid is) \times P(the \mid on) \end{aligned}$$

OOV & Smoothing

OOV stands for **Out Of Vocabulary**

- Words appear only in a test set but not in the training set.
- OOV problem occurs even when we work on big data.

Types of smoothing

- Laplace smoothing (Add-one smoothing)
- Add-k smoothing
- Back-off
- Linear interpolation
- $\cdots$

Types of smoothing

- Laplace smoothing (Add-one smoothing)
- Add-k smoothing
- Back-off
- Linear interpolation
- $\cdots$

Linear Interpolation

We mix the probability estimates from all the n-gram estimators, weighing and combining the trigram, bigram, and unigram counts.

Linear Interpolation

We mix the probability estimates from all the n-gram estimators, weighing and combining the trigram, bigram, and unigram counts.

Bigram + Unigram

$$\hat{P}(table \mid the) = \lambda_1 P(table \mid the) + \lambda_2 P(table)$$

Linear Interpolation

We mix the probability estimates from all the n-gram estimators, weighing and combining the trigram, bigram, and unigram counts.

Bigram + Unigram

$$\hat{P}(table \mid the) = \lambda_1 P(table \mid the) + \lambda_2 P(table)$$

Trigram + Bigram + Unigram

$$\begin{aligned}\hat{P}(table \mid on\ the) &= \lambda_1 P(table \mid on\ the) \\ &+ \lambda_2 P(table \mid the) \\ &+ \lambda_3 P(table)\end{aligned}$$

Linear Interpolation

We mix the probability estimates from all the n-gram estimators, weighing and combining the trigram, bigram, and unigram counts.

Bigram + Unigram

$$\hat{P}(table \mid the) = \lambda_1 P(table \mid the) + \lambda_2 P(table)$$

Trigram + Bigram + Unigram

$$\begin{aligned} \hat{P}(table \mid on\ the) &= \lambda_1 P(table \mid on\ the) \\ &+ \lambda_2 P(table \mid the) \\ &+ \lambda_3 P(table) \end{aligned}$$

where we choose λ_i such that $\sum_i \lambda_i = 1$

How are the λ values set?

- They can be learned from a held-out corpus
- Can be found by EM algorithm.
- For the purpose of this project, we assume without loss of generality that $\lambda_i > \lambda_j (\forall i < j)$

POS-Ngram

Formula

$$P(w_n \mid w_1^{n-1}) = \sum_{c_n} P(w_n \mid c_n) \times P(c_n \mid c_{n-N+1}^{n-1})$$

where c_n is a class of w_n

(In this context, c_n is considered a part of speech of the word w_n)

POS-Ngram

Formula

$$P(w_n \mid w_1^{n-1}) = \sum_{c_n} P(w_n \mid c_n) \times P(c_n \mid c_{n-N+1}^{n-1})$$

where c_n is a class of w_n

(In this context, c_n is considered a part of speech of the word w_n)

Example

cat, dog, thought, $\cdots$	c_1:Noun
go, speak, $\cdots$	c_2:Verb
play, $\cdots$	c_1:Noun, c_2:Verb

POS-Ngram

Formula

$$P(w_n \mid w_1^{n-1}) = \sum_{c_n} P(w_n \mid c_n) \times P(c_n \mid c_{n-N+1}^{n-1})$$

where c_n is a class of w_n

(In this context, c_n is considered a part of speech of the word w_n)

Example

cat, dog, thought, $\cdots$	c_1:Noun
go, speak, $\cdots$	c_2:Verb
play, $\cdots$	c_1:Noun, c_2:Verb

It means that to predict the next word w_n

1. $P(w_n \mid c_n) = \frac{C(w_n, c_n)}{C(c_n)}$
2. Compute $P(c_n \mid c_{n-N+1}^{n-1})$
3. w_n^* is determined by $\arg\max_{w_n} P(w_n \mid w_1^{n-1})$

POS-Ngram

Formula

$$P(w_n \mid w_1^{n-1}) = \sum_{c_n} P(w_n \mid c_n) \times P(c_n \mid c_{n-N+1}^{n-1})$$

where c_n is a class of w_n

(In this context, c_n is considered a part of speech of the word w_n)

Example

cat, dog, thought, $\cdots$	c_1:Noun
go, speak, $\cdots$	c_2:Verb
play, $\cdots$	c_1:Noun, c_2:Verb

It means that to predict the next word w_n

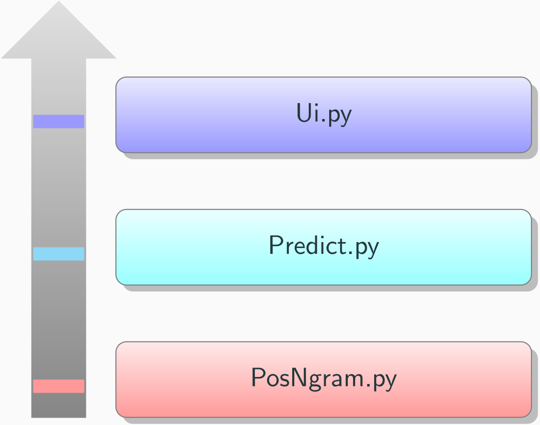
1. $P(w_n \mid c_n) = \frac{C(w_n, c_n)}{C(c_n)}$
2. Compute $P(c_n \mid c_{n-N+1}^{n-1}) \sim P(w_n \mid w_{n-N+1}^{n-1})$
3. w_n^* is determined by $\arg\max_{w_n} P(w_n \mid w_1^{n-1})$

Implementation

Implementation

- Programming Language : Python
- Ui : Tkinter
- NLTK (Natural Language ToolKit)

Programming Structure



References

- Speech and Language Processing (Chapter 3) (Daniel Jurafsky - Stanford University, 1999)
- N-gram Language Modeling Tutorial (Lecture notes courtesy of Prof. Mari Ostendorf, 2007-06-21)
- Probabilistic Language Model (Chapter 3) (Kenji KITA, University of Tokyo Press, 1999)

DEMO