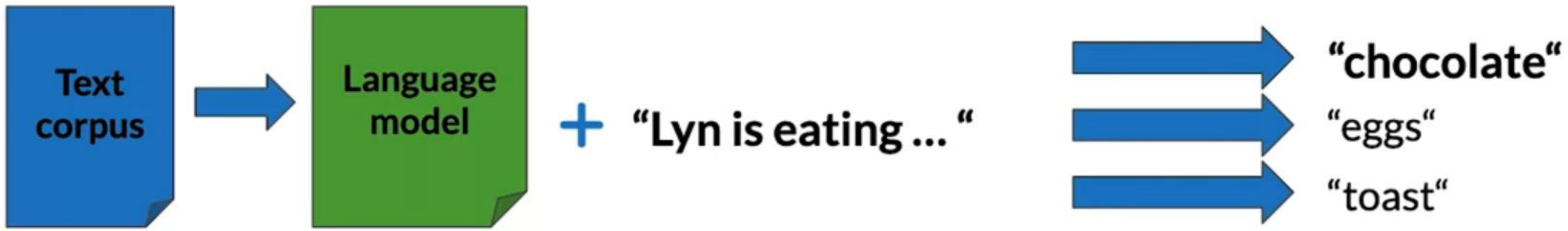
What you'll be able to do!

- Create **language model (LM)** from text corpus to
  - Estimate probability of word sequences
  - Estimate probability of a word following a sequence of words
- Apply this concept to **autocomplete a sentence** with most likely suggestions



Other Applications

Speech recognition

P(I saw a van) > P(eyes awe of an)

Spelling correction

"He entered the ship to buy some groceries" - "ship" a dictionary word

- P(entered the shop to buy) > P(entered the ship to buy)

Augmentative communication

Predict most likely word from menu for people unable to physically talk or sign.
(Newell et al., 1998)

Learning objectives

- Process text corpus to N-gram language model
- Out of vocabulary words
- Smoothing for previously unseen N-grams
- Language model evaluation

→ Sentence auto-complete

Outline

- What are N-grams?
- N-grams and conditional probability from corpus

N-gram

An N-gram is a sequence of N words

Corpus: I am happy because I am learning

Unigrams: { I , am , happy , because , learning }

Bigrams: { I am , am happy , happy because ... } ✗ I happy

Trigrams: { I am happy , am happy because, ... }

Sequence notation

Corpus: This is great ... teacher drinks tea. $m = 500$

w_1 w_2 w_3 ... w_{498} w_{499} w_{500}

$$w_1^m = w_1 \ w_2 \dots w_m$$

$$w_1^3 = w_1 \ w_2 \ w_3$$

$$w_{m-2}^m = w_{m-2} \ w_{m-1} \ w_m$$

Unigram probability

Corpus: I am happy because I am learning

Size of corpus m = 7

$$P(I) = \frac{2}{7}$$

$$P(happy) = \frac{1}{7}$$

Probability of unigram:

$$P(w) = \frac{C(w)}{m}$$

Bigram probability

Corpus: I am happy because I am learning

$$P(am|I) = \frac{C(I\ am)}{C(I)} = \frac{2}{2} = 1$$

$$P(happy|I) = \frac{C(I\ happy)}{C(I)} = \frac{0}{2} = 0$$ ✖ I happy

$$P(learning|am) = \frac{C(am\ learning)}{C(am)} = \frac{1}{2}$$

Probability of a bigram: $P(y|x) = \frac{C(x\ y)}{\sum_w C(x\ w)} = \frac{C(x\ y)}{C(x)}$

Trigram Probability

Corpus: I am happy because I am learning

$$P(happy|I\ am) = \frac{C(I\ am\ happy)}{C(I\ am)} = \frac{1}{2}$$

Probability of a trigram: $$P(w_3|w_1^2) = \frac{C(w_1^2\ w_3)}{C(w_1^2)}$$

$$C(w_1^2\ w_3) = C(w_1\ w_2\ w_3) = C(w_1^3)$$

N-gram probability

Probability of N-gram: $P(w_N|w_1^{N-1}) = \frac{C(w_1^{N-1}\, w_N)}{C(w_1^{N-1})}$

$$C(w_1^{N-1}\, w_N) = C(w_1^N)$$

Probability of a sequence

- Given a sentence, what is its probability?

$$P(the\ teacher\ drinks\ tea) = ?$$

- Conditional probability and chain rule reminder

$$P(B|A) = \frac{P(A,B)}{P(A)} \implies P(A,B) = P(A)P(B|A)$$

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

Probability of a sequence

$$P(the\ teacher\ drinks\ tea) =$$

$$P(the)P(teacher|the)P(drinks|the\ teacher)$$
$$P(tea|the\ teacher\ drinks)$$

Sentence not in corpus

- Problem: Corpus almost never contains the exact sentence we're interested in or even its longer subsequences!

Input: the teacher drinks tea

$$P(the\ teacher\ drinks\ tea) = P(the)P(teacher|the)P(drinks|the\ teacher)P(tea|the\ teacher\ drinks)$$

$$P(tea|the\ teacher\ drinks) = \frac{C(the\ teacher\ drinks\ tea)}{C(the\ teacher\ drinks)}$$

← Both likely 0

Approximation of sequence probability

the teacher drinks tea

$P(tea|the\ teacher\ drinks) \approx P(tea|drinks)$

$P(teacher|the)$
$P(drinks|teacher)$
$P(tea|drinks)$

$$P(the\ teacher\ drinks\ tea) =$$

$$P(the)P(teacher|the)P(drinks|the\ teacher)\boxed{P(tea|the\ teacher\ drinks)}$$

$$\Downarrow$$

$$P(the)P(teacher|the)P(drinks|teacher)\boxed{P(tea|drinks)}$$

Approximation of sequence probability

- Markov assumption: only last N words matter

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

- Entire sentence modeled with bigram $P(w_1^n) \approx \prod_{i=1}^{n} P(w_i|w_{i-1})$

$$P(w_1^n) \approx \boxed{P(w_1)} P(w_2|w_1) ... P(w_n|w_{n-1})$$

Outline

- Start of sentence symbols <s>

- End of sentence symbol </s>

Start of sentence token <s>

the teacher drinks tea

$$P(the\ teacher\ drinks\ tea) \approx P(the)P(teacher|the)P(drinks|teacher)P(tea|drinks)$$

<s> the teacher drinks tea

$$P(<s>\ the\ teacher\ drinks\ tea) \approx P(the|<s>)P(teacher|the)P(drinks|teacher)P(tea|drinks)$$

Start of sentence token <s> for N-grams

- Trigram:

$$P(the\ teacher\ drinks\ tea) \approx$$
$$P(the)P(teacher|the)P(drinks|the\ teacher)P(tea|teacher\ drinks)$$

the teacher drinks tea => <s> <s> the teacher drinks tea

$$P(w_1^n) \approx P(w_1|<s>\ <s>)P(w_2|<s>\ w_1)...P(w_n|w_{n-2}\ w_{n-1})$$

- N-gram model: add N-1 start tokens <s>

End of sentence token </s> - motivation

$$P(y|x) = \frac{C(x\ y)}{\sum_w C(x\ w)} = \frac{C(x\ y)}{C(x)}$$

Corpus:

<s> Lyn drinks chocolate

<s> John drinks

$$\sum_w C(drinks\ w) = 1$$

$$C(drinks) = 2$$

End of sentence token </s> - motivation

Corpus

<s> yes no

<s> yes yes

<s> no no

Sentences of length 2:

<s> yes yes

<s> yes no

<s> no no

<s> no yes

$$P(<s>\ \text{yes yes}) =$$

$$P(\text{yes} \mid <s>) \times P(\text{yes} \mid \text{yes}) =$$

$$\frac{C(<s>\ \text{yes})}{\sum_w C(<s>\ w)} \times \frac{C(\text{yes yes})}{\sum_w C(\text{yes}\ w)} =$$

$$\frac{2}{3} \times \frac{1}{2} = \frac{1}{3}$$

End of sentence token </s> - motivation

<u>Corpus</u>

<s> yes no

<s> yes yes

<s> no no

<u>Sentences of length 2:</u>

<s> yes yes

<s> yes no

<s> no no

<s> no yes

$$P(< s > \ \text{yes yes}) = \frac{1}{3}$$

$$P(< s > \ \text{yes no}) = \frac{1}{3}$$

$$P(< s > \ \text{no no}) = \frac{1}{3}$$

$$P(< s > \ \text{no yes}) = 0$$

$$\sum_{\text{2 word}} P(\cdots) = 1$$

End of sentence token </s> - motivation

<u>Corpus</u>

<s> yes no

<s> yes yes

<s> no no

<u>Sentences of length 3:</u>

<s> yes yes yes

<s> yes yes no

...

<s> no no no

$$P(<s> \text{ yes yes yes}) = \ldots$$

$$P(<s> \text{ yes yes no}) = \ldots$$

$$\ldots = \ldots$$

$$P(<s> \text{ no no no}) = \ldots$$

$$\sum_{\text{3 word}} P(\cdots) = 1$$

End of sentence token </s> - motivation

Corpus

<s> yes no

<s> yes yes

<s> no no

$$\sum_{2\ \text{word}} P(\cdots) + \sum_{3\ \text{word}} P(\cdots) + \ldots$$

End of sentence token </s> - solution

- Bigram

<s> the teacher drinks tea => <s> the teacher drinks tea </s>

$$P(the|<s>)P(teacher|the)P(drinks|teacher)P(tea|drinks)P(</s>|tea)$$

Corpus:
<s> Lyn drinks chocolate </s>
<s> John drinks </s>

$$\sum_{w} C(drinks\ w) = 2$$

$$C(drinks) = 2$$

End of sentence token </s> for N-grams

- N-gram => just one </s>

  E.g. Trigram:
  the teacher drinks tea => <s> <s> the teacher drinks tea </s>

Example - bigram

Corpus

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

$$P(sentence) = \frac{2}{3} * \frac{1}{2} * \frac{1}{2} * \frac{2}{2} = \frac{1}{6}$$

$$P(John|<s>) = \frac{1}{3}$$

$$P(</s>|tea) = \frac{1}{1}$$

$$P(chocolate|eats) = \frac{1}{2}$$

$$P(Lyn|<s>) =? = \frac{2}{3}$$

Example - bigram

Corpus

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

$$P(sentence) = \frac{2}{3} * \frac{1}{2} * \frac{1}{2} * \frac{2}{2} = \frac{1}{6}$$

$$P(John|<s>) = \frac{1}{3}$$

$$P(</s>|tea) = \frac{1}{1}$$

$$P(chocolate|eats) = \frac{1}{2}$$

$$P(Lyn|<s>) =? = \frac{2}{3}$$

Example - bigram

Corpus

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

$$P(sentence) = \frac{2}{3} * \frac{1}{2} * \frac{1}{2} * \frac{2}{2} = \frac{1}{6}$$

$$P(John|<s>) = \frac{1}{3}$$

$$P(</s>|tea) = \frac{1}{1}$$

$$P(chocolate|eats) = \frac{1}{2}$$

$$P(Lyn|<s>) =? = \frac{2}{3}$$

Outline

- Count matrix
- Probability matrix
- Language model
- Log probability to avoid underflow
- Generative language model

Count matrix

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}, w_n)}{C(w_{n-N+1}^{n-1})}$$

- Rows: unique corpus (N-1)-grams
- Columns: unique corpus words

Corpus: <s>I study I learn</s>

- Bigram count matrix

"study I" bigram →

	<s>	</s>	I	study	learn
<s>	0	0	1	0	0
</s>	0	0	0	0	0
I	0	0	0	1	1
study	0	0	1	0	0
learn	0	1	0	0	0

Probability matrix

$$P(w_n | w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}, w_n)}{C(w_{n-N+1}^{n-1})}$$

- Divide each cell by its row sum

$$sum(row) = \sum_{w \in V} C(w_{n-N+1}^{n-1}, w) = C(w_{n-N+1}^{n-1})$$

Corpus: <s>I study I learn</s>

Count matrix (bigram)

	<s>	</s>	I	study	learn	sum
<s>	0	0	1	0	0	1
</s>	0	0	0	0	0	0
I	0	0	0	1	1	2
study	0	0	1	0	0	1
learn	0	1	0	0	0	1

Probability matrix

	<s>	</s>	I	study	learn
<s>	0	0	1	0	0
</s>	0	0	0	0	0
I	0	0	0	0.5	0.5
study	0	0	1	0	0
learn	0	1	0	0	0

Language model

- probability matrix => language model
  - Sentence probability
  - Next word prediction

	<s>	</s>	I	study	learn
<s>	0	0	1	0	0
</s>	0	0	0	0	0
I	0	0	0	0.5	0.5
study	0	0	1	0	0
learn	0	1	0	0	0

Sentence probability:

<s> I learn </s>

$$P(sentence) =$$
$$P(I|<s>)P(learn|I)P(</s>|learn) =$$
$$1 \times 0.5 \times 1 =$$
$$0.5$$

Log probability

$$P(w_1^n) \approx \prod_{i=1}^{n} P(w_i|w_{i-1})$$

- All probabilities in calculation <=1 and multiplying them brings risk of underflow
- Logarithm properties reminder $$log(a * b) = log(a) + log(b)$$

Generative Language model

Corpus:

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

1. (<s>, Lyn) or (<s>, John)?
2. (Lyn,eats) or (Lyn,drinks) ?
3. (drinks,tea) or (drinks,chocolate)?
4. (tea,</s>) - always

Algorithm:

1. Choose sentence start
2. Choose next bigram starting with previous word
3. Continue until </s> is picked

Outline

- Train/Validation/Test split
- Perplexity

Test data

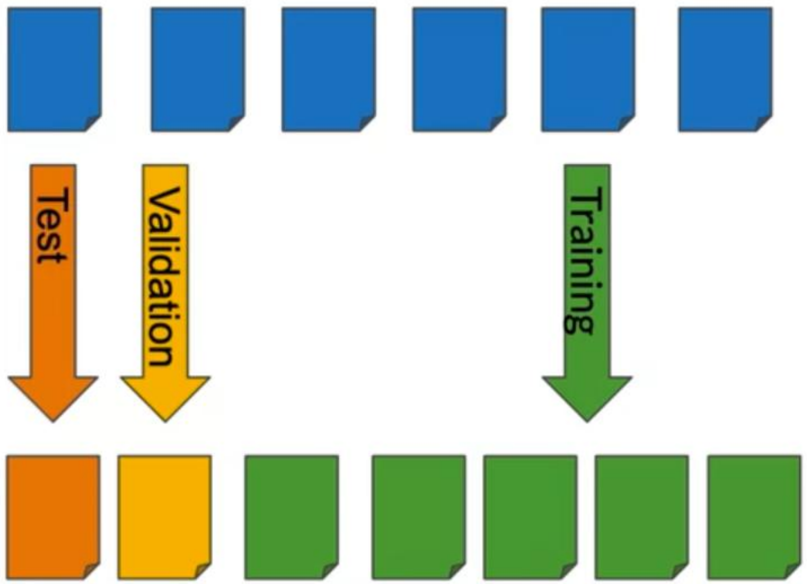
- Split corpus to Train/Validation/Test

✖ Evaluate on Training dataset

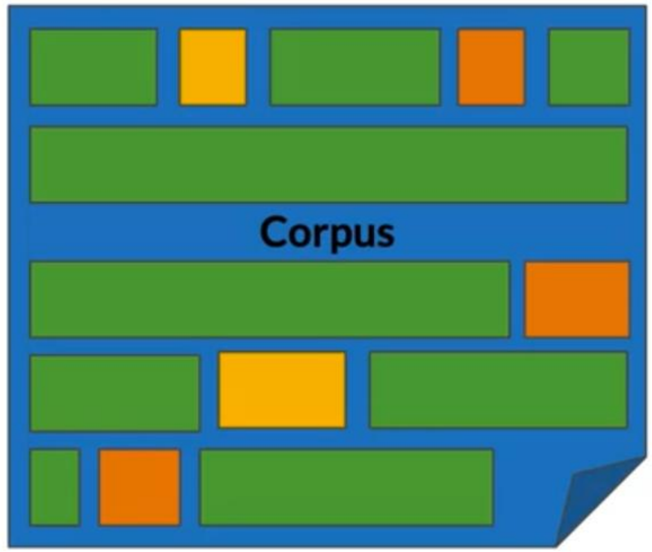
- For smaller corpora
  - 80% Train
  - 10% Validation
  - 10% Test
- For large corpora (typical for text)
  - 98% Train
  - 1% Validation
  - 1% Test

Test data - split method

- Continuous text



- Random short sequences



Perplexity

$$PP(W) = P(s_1, s_2, ..., s_m)^{-\frac{1}{m}}$$

W → test set containing **m** sentences **s**

s_i → i-th sentence in the test set, each ending with </s>

m → number of all words in entire test set W including </s> but not including <s>

Perplexity

E.g. m=100

$$P(W) = 0.9 => PP(W) = 0.9^{-\frac{1}{100}} = 1.00105416$$

$$P(W) = 10^{-250} => PP(W) = (10^{-250})^{-\frac{1}{100}} \approx 316$$

- Smaller perplexity = better model

- Character level models PP < word-based models PP

Perplexity for bigram models

$$PP(W) = \sqrt[m]{\prod_{i=1}^{m} \prod_{j=1}^{|s_i|} \frac{1}{P(w_j^{(i)} | w_{j-1}^{(i)})}}$$

$w_j^{(i)}$ → j-th word in i-th sentence

- concatenate all sentences in W

$$PP(W) = \sqrt[m]{\prod_{i=1}^{m} \frac{1}{P(w_i | w_{i-1})}}$$

w_i → i-th word in test set

Log perplexity

$$PP(W) = \sqrt[m]{\prod_{i=1}^{m} \frac{1}{P(w_i|w_{i-1})}}$$

$$\Downarrow$$

$$logPP(W) = -\frac{1}{m}\sum_{i=1}^{m} log_2(P(w_i|w_{i-1}))$$

Examples

Training 38 million words, test 1.5 million words, WSJ corpus
Perplexity Unigram: 962 Bigram: 170 Trigram: 109

Unigram

Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

Bigram

Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

Trigram

They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

[Figure from *Speech and Language Processing* by Dan Jurafsky et. al]

Outline

- Unknown words
- Update corpus with <UNK>
- Choosing vocabulary

Out of vocabulary words

- Closed vs. Open vocabularies
- Unknown word = Out of vocabulary word (OOV)
- special tag <UNK> in corpus and in input

Using <UNK> in corpus

- Create vocabulary V
- Replace any word in corpus and not in V by <UNK>
- Count the probabilities with <UNK> as with any other word

Example

Corpus

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

Corpus

<s> Lyn drinks chocolate </s>

<s> <UNK> drinks <UNK> </s>

<s> Lyn <UNK> chocolate </s>

Min frequency f=2

Vocabulary

Lyn, drinks, chocolate

Input query

<s>Adam drinks chocolate</s>

<s><UNK> drinks chocolate</s>

How to create vocabulary V

- Criteria:
  - Min word frequency f
  - Max |V|, include words by frequency
- Use <UNK> sparingly
- Perplexity - only compare LMs with the same V

Outline

- Missing N-grams in corpus
- Smoothing
- Backoff and interpolation

Missing N-grams in training corpus

- Problem: N-grams made of known words still might be missing in the training corpus "John", "eats" in corpus ✖ "John eats"
- Their counts cannot be used for probability estimation

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}, w_n)}{C(w_{n-N+1}^{n-1})}$$

← Can be 0

Smoothing

- Advanced methods:
  Kneser-Ney smoothing
  Good-Turing smoothing

- Add-one smoothing (Laplacian smoothing)

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + 1}{\sum_{w \in V}(C(w_{n-1}, w) + 1)} = \frac{C(w_{n-1}, w_n) + 1}{C(w_{n-1}) + V}$$

- Add-k smoothing

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + k}{\sum_{w \in V}(C(w_{n-1}, w) + k)} = \frac{C(w_{n-1}, w_n) + k}{C(w_{n-1}) + k * V}$$

Backoff

- If N-gram missing => use (N-1)-gram, …
  - Probability discounting e.g. Katz backoff
  - "Stupid" backoff

Corpus

<s> Lyn drinks chocolate </s>

<s> John drinks tea </s>

<s> Lyn eats chocolate </s>

$$P(chocolate|John\ drinks) = ?$$

$$0.4 \times P(chocolate|drinks)$$

Interpolation

$$\hat{P}(chocolate|John\ drinks) = 0.7 \times P(chocolate|John\ drinks)$$
$$+0.2 \times P(chocolate|drinks) + 0.1 \times P(chocolate)$$

$$\hat{P}(w_n|w_{n-2}\ w_{n-1}) = \lambda_1 \times P(w_n|w_{n-2}\ w_{n-1})$$
$$+\lambda_2 \times P(w_n|w_{n-1}) + \lambda_3 \times P(w_n)$$

$$\sum_i \lambda_i = 1$$

Summary

- N-Grams and probabilities
- Approximate sentence probability from N-Grams
- Build language model from corpus
- Fix missing information
  - Out of vocabulary words with <UNK>
  - Missing N-Gram in corpus with smoothing, backoff and interpolation
- Evaluate language model with perplexity
- Coding assignment!