https://www.bbc.co.uk/programmes/b00n7sf5

And now, for something...

N-GRAMS

Luísa Coheur

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

LEARNING OBJECTIVES

LEARNING OBJECTIVES

- After this class, students should be able to:
  - Explain what a N-gram is
  - Understand how to model language, with N-grams
  - Apply N-grams to
    - Predict the next word of a given sentence, and
    - Calculate the probability of a sentence
  - Explain the concept of smoothing
  - Apply Laplace-smoothing

TOPICS

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

LANGUAGE MODELS

- Models at the basis of Natural Language Generation
- Language models (notice that I am not using the word large) learn the probability distribution of words, that is, how words can be organized to create meaningful and grammatically correct sentences
- With Language Models we can:
  - Predict the next word within a text; and
  - Find how likely (probable) is a sequence of words

N-GRAMS

- Example 1: complete the following sentences:
  - And now for something…
  - Once upon a…
  - Spoiler…
  - Stranger…

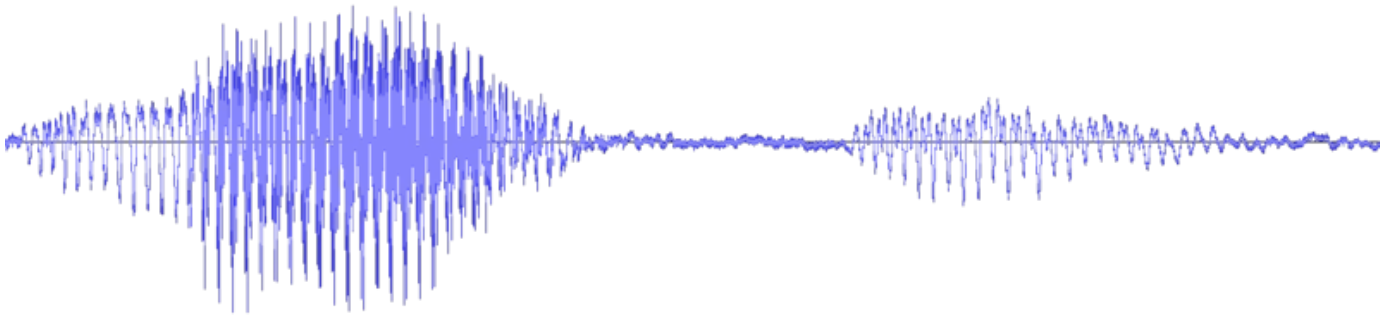
N-GRAMS

- Example 1: complete the following sentences:
  - And now for something completely different
  - Once upon a time
  - Spoiler alert
  - Stranger things

With N-grams we can make word prediction!!!!

N-GRAMS

- Example 2: consider the possible outputs of an Automatic Speech Recognizer (ASR):



https://commons.wikimedia.org/wiki/File:Signal-speech-martin-de.png

- olá edgar
- ou lá apagar
- ó lá edgar
- ...

Which sentence is the most likely?

N-GRAMS

- Example 3: consider the possible outputs of a Machine Translation System:
  - Input: It is raining cats and dogs
  - Possible translations:
    - Chovem cães e gatos
    - Chove a potes
    - Chovem potes
    - ...

Which sentence is the most likely?

Ok, but what is a N-gram?

N-GRAMS

- N-gram = sequence of N tokens
  - N = 1 => unigrams
  - N = 2 => bigrams
  - N = 3 => trigrams
  - ...
- A token can be
  - a word (ola, Maria, hello, ...)
  - a character (o, l, M, a, r, ...)
  - a set of sequences of characters (ol, la, Mar, ari, ...)

N-GRAMS

- Let us see now how to apply them to
  - Make word prediction
  - Calculate a sentence probability

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

WORD PREDICTION

- Input: H (or history $H = W_1 \ldots W_{N-1}$)
- Task: find what is the probability of W (= W_N) being the next word, that is, we want to find:
  - $P(W \mid H)$
- Notation:
  - $W_1^{N-1} = W_1 \ldots W_{N-1}$

Example:
H = Once upon a
W = time
?? P(time | Once upon a)??

WORD PREDICTION

- Hypothesis 1:

  P(W | H) = $count(HW)/\sum_k count(HK)$

  $= count(HW)/count(H)$
- Example:
  - H = Once upon a
  - W = time
  - P(time | Once upon a) =

    $= \text{count(once upon a time)}/\text{count(once upon a)}$
- Problem:
  - Some sequences were never seen, thus, you might not have all these values

WORD PREDICTION

- Markov Assumption:
  - It is possible to calculate the probability of a future event without having to look to the entire history

- Let's do some approximations!!

WORD PREDICTION

- Hypothesis 2 (based on Markov assumption)
  - To calculate $P(W \mid H) = P(W_N \mid W_1 \ldots W_{N-1})$:
    - $P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-1})$ (use bigrams)
    - $P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-2}\, W_{N-1})$ (use trigrams)

ACTIVE LEARNING MOMENT

EXERCISE

- Corpus (<s> for beginning of the sentence and </s> for the end):
  - <s>Eu adoro a Maria</s> (I adore Maria)
  - <s>A Maria eu adoro</s> (Maria I adore)
  - <s>Adoro bolachas Maria</s> ((I) adore cookies (named) Maria)

If I say "eu adoro" (I adore), what is the most probable next word: eu, a, Maria, adoro, bolachas or </s>?

- Use:
  - $P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-1})$ (use bigrams)
  - $P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-2}\, W_{N-1})$ (use trigrams)

EXERCISE: BIGRAMS

- First, some pre-processing:
  - <s>eu adoro a maria</s>
  - <s>a maria eu adoro</s>
  - <s>adoro bolachas maria</s>

What you need to know:

$P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-1})$

$P(W \mid H) = count(HW)/count(H)$

EXERCISE: BIGRAMS

What you need to know:

$P(W_N \mid W_1 \dots W_{N-1}) \cong P(W_N \mid W_{N-1})$

$P(W \mid H) = count(HW)/count(H)$

- First, some pre-processing:
  - <s>eu adoro a maria</s>
  - <s>a maria eu adoro</s>
  - <s>adoro bolachas maria</s>

- Using bigrams: $P(W_N \mid W_1 \dots W_{N-1}) \cong P(W_N \mid W_{N-1})$
  - P(eu | adoro) = count(adoro eu)/count(adoro) = 0
  - P(a | adoro) = count(adoro a)/count(adoro) = 1/3
  - P(Maria | adoro) = count(adoro Maria)/count(adoro) = 0
  - P(adoro | adoro) = count(adoro adoro)/count(adoro) = 0
  - P(bolachas | adoro) = count(adoro bolachas)/count(adoro) = 1/3
  - P(</s> | adoro) = count(adoro </s>)/count(adoro) = 1/3

EXERCISE: TRIGRAMS

- First, some pre-processing:
  - <s>eu adoro a maria</s>
  - <s>a maria eu adoro</s>
  - <s>adoro bolachas maria</s>

What you need to know:

$P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-2}\, W_{N-1})$

$P(W \mid H) = count(HW)/count(H)$

EXERCISE: TRIGRAMS

What you need to know:

$P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-2}\, W_{N-1})$

$P(W \mid H) = count(HW)/count(H)$

- First, some pre-processing:
  - <s>eu adoro a maria</s>
  - <s>a maria eu adoro</s>
  - <s>adoro bolachas maria</s>

- Using trigrams: $P(W_N \mid W_1 \ldots W_{N-1}) \cong P(W_N \mid W_{N-2}\, W_{N-1})$
  - P(eu | eu adoro) = count(eu adoro eu)/count(eu adoro) = 0
  - P(a | eu adoro) = count(eu adoro a)/count(eu adoro) = 1/2
  - P(Maria | eu adoro) = count(eu adoro Maria)/count(eu adoro) = 0
  - P(adoro | eu adoro) = count(eu adoro adoro)/count(eu adoro) = 0
  - P(bolachas | eu adoro) = count(eu adoro bolachas)/count(eu adoro) = 0
  - P(</s> | eu adoro) = count(eu adoro </s>)/count(eu adoro) = 1/2

Generated by DALL-E

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

SENTENCE PROBABILITY

- Consider now that you want to know how probable is a sentence $W_1 \ldots W_{N-1}\, W_N$

SENTENCE PROBABILITY

- We can use the chain rule (of probability):

$$P(w_1^N) = P(w_1|<s>)*P(w_2\,|<s>\, w_1)*\ldots * P(w_N\,|\; w_1^{N-1})$$
$$= \prod_{k=1}^{N} P(w_k|w_1^{k-1})$$

- We will have the same problem as before => some sequences were never seen. So, once again let us use the Markov assumption:

$$P(w_1^N) \cong \prod_{k=1}^{N} P(w_k|w_{k-1}) \text{ (use bigrams)}$$
$$P(w_1^N) \cong \prod_{k=1}^{N} P(w_k|w_{k-2}w_{k-1}) \text{ (use trigrams)}$$

ACTIVE LEARNING MOMENT

Taken from http://spring2015.cs-114.org/wp-content/uploads/2016/01/NgramModels.pdf

	I	Want	To	Eat	Chinese	Food	lunch
I	8	1087	0	13	0	0	0
Want	3	0	786	0	6	8	6
To	3	0	10	860	3	0	12
Eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
Food	19	0	17	0	0	0	0
Lunch	4	0	0	0	0	1	0

I	Want	To	Eat	Chinese	Food	Lunch
3437	1215	3256	938	213	1506	459

Exercise:
What is the probability of the sentence "I eat Chinese food"

What you need to know:

$P(w_1^N) \cong \prod_{k=1}^{N} P(w_k | w_{k-1})$

$P(W | H) = count(HW)/count(H)$

Taken from http://spring2015.cs-114.org/wp-content/uploads/2016/01/NgramModels.pdf

	I	Want	To	Eat	Chinese	Food	lunch
I	8	1087	0	13	0	0	0
Want	3	0	786	0	6	8	6
To	3	0	10	860	3	0	12
Eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
Food	19	0	17	0	0	0	0
Lunch	4	0	0	0	0	1	0

Exercise:
What is the probability of the sentence "I eat Chinese food"

What you need to know:

$P(w_1^N) \cong \prod_{k=1}^{N} P(w_k|w_{k-1})$

P(W | H) = $count(HW)/count(H)$

I	Want	To	Eat	Chinese	Food	Lunch
3437	1215	3256	938	213	1506	459

P(I eat Chinese food) = P(I | <s>) * P(eat | I) * P(Chinese | eat) * P(food | Chinese) * P(</s> | food) Assumindo que não se sabe P(I | <s>) e * P(</s> | food), então = C(I eat)/C(I) * C(eat Chinese)/C(eat) * C(Chinese food)/count(Chinese) = 13/3437* 19/938* 120/213 = …

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

EVALUATION OF N-GRAMS

- Perplexity:
  - Still used
  - "Train" set T:
    - Calculate:
      - $Model_1$ = unigrams in T
      - $Model_2$ = bigrams in T
      - …

EVALUATION OF N-GRAMS

- Perplexity:
  - Test set: $W = w_1\ w_2\ \dots\ w_N$,
    - Calculate perplexity PP(W) (for instance – different formulas):

$$PP(W) = P(w_1 w_2 \dots w_n)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

    - There will be a different PP(W) for each model:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \dots w_{i-1})}}$$

  - Lower value of PP(W) => better model (less ”perplex”)

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

CHALLENGES

- N-gram models are biased to the training corpus
- N-grams are not appropriate to deal with long distance dependencies
  - Gollum loves in a very sick way his precious
- Data sparseness
  - Bigger N (N-grams) => sparse data
- How to deal with 0 counts?
  - Smoothing is the answer

Overview

- Learning objectives
- Topics
  - Concepts
    - Language Models
    - N-grams
  - Word prediction
  - Sentence probability
  - Evaluation of N-grams
  - Challenges
  - Smoothing
- Key takeaways
- Suggested readings

SMOOTHING

- Techniques that allow to deal with the fact that some sequences were never seen or have not been seen many times
- These techniques will change estimations/probability mass (and we need to guarantee that the counts still make sense => Robin Hood)

SMOOTHING

- Laplace or Add-one smoothing:
  - Add 1 to all the counts (and recalculate counts)

SMOOTHING

- Laplace or Add-one smoothing:
  - Example with bigrams:
    - Previously (Maximum Likelihood Estimation – MLE):
      - PMLE(W_N | W_{N-1}) = count(W_{N-1} W_N)/count(W_{N-1})

    - Now:
      - PLaplace(W_N | W_{N-1}) = (count(W_{N-1} W_N)+1)/(count(W_{N-1}) + | V |)
        - (| V | is the number of words in the vocabulary V)

SMOOTHING

- Laplace or Add-one smoothing:
  - Example:
    - |V| = 100.000 words
    - count(w_2) = 10, count($w_2 w_3$) = 9,
    - Previously:
      - PMLE(W_3 | W_2) = count($W_2 W_3$)/count(W_2) = 9/10 = 0.9
    - Now:
      - PLaplace(W_3 | W_2) = (count(W_2W_3)+1)/(count(W_2) + |V|) = 10/100.010

Problem:
If count(w_1) = 10, and count(w_1w_3) = 0,
Then:
$P_{MLE}(W_3 | W_1) = 0$, $P_{Laplace}(W_3 | W_1) = 1/100.010$

Too close

SMOOTHING

- There are many more smoothing techniques
  - Good-Turing Discounting
    - In order to estimate the probabilities of things that occur c times, it uses the counts of things that occurred (c+1) times (and then you will have to adjust everything again).
  - ...

KEY TAKEWAYS

KEY TAKEWAYS

- Understand concepts such as of N-grams, Markov assumptions and smoothing and Language Model
- Be able to apply N-grams to estimate the probability of a sentence or of a word, given a previous sequence of words

SUGGESTED READINGS

READINGS

- Sebenta: chapter about N-grams
- Jurafsky: 3.1, 3.3 and 3.6.1