Language Modelling with N-grams

MMAI 5400 – lecture 3
Fall 2024

Contents

Why is NLP difficult?

- Challenges with natural language

Intro to language models

- N-gram language models
  - Limitations
  - Fixes
  - Evaluation

Book chapter 3 – N-gram Language Models

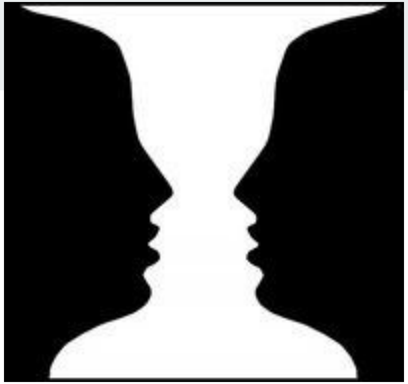
Read

- Sections: 3 - 3.7 & 3.9

Don't read (although, I'm not going to stop you)

- Section: 3.8 *Advanced: Perplexity's Relation to Entropy*

NLP challenges – part 1



Context

- The meaning of natural language is often context dependent

Ambiguity (multiple meanings)

- Crash blossoms (crash blossom - Wiktionary, the free dictionary), examples:

  "Violinist Linked to JAL Crash Blossoms"

  "Red Tape Holds Up New Bridges"

  "Local High School Dropouts Cut in Half"

Natural languages → ≈infinite number of possible sentences

- Listing all sentence–meaning pairs is not possible

NLP challenges – part 2

Non-standard language

“U taught us 2 @neversaynever!”

“Were SOO PROUD of what youve accomplished!”

Ambiguous segmentation

The New York-New Haven railroad vs The New York-New Haven railroad

NLP challenges – part 3

Idioms

“Get cold feet”

“lose face”

“raining cats & dogs”

Neologisms

“Unfriend”

“Retweet”

“bromance”

Difficult entity names

*“Where is **A Bug’s Life** playing?”*

*“when was **Let it be** recorded?”*

*“... a mutation in the **for** gene...”*

Two ways of using text in machine learning

Bag-of-Words

- Based on word (token) counts
- Documents are represented by their word frequencies
- Word order is ignored
- Good for long documents

Sequence models

- Mainly neural network-based
- Documents are represented as a sequence of tokens (word/sub-word/character)
- Word order is important
- Works sentence level

Two ways of using text in machine learning

Bag-of-Words

- Based on word (token) counts
- Documents are represented by their word frequencies
- Word order is ignored
- Good for long documents

Class 4

Sequence models

- Mainly neural network-based
- Documents are represented as a sequence of tokens (word/sub-word/character)
- Word order is important
- Works sentence level

Classes 8 to 12

Language models & N-grams

Language models (LMs)

Models that assign probabilities to sequences of words (e.g. sentences)

$$P(w_1, w_2, ..., w_t)$$

Simple LM

- N-grams

Neural LM

- Autoregressively trained RNNs & Transformers

Language Models: linguistic prophets

Which is the more natural-sounding sentence?

Whoever is happy will make others happy too.

Whoever make is will happy happy too others.

Language Models: linguistic prophets

Question: What word would you use to complete the following sentence?

What have you been up to ___

- *beehive*
- *lately*
- *N-gram*
- *complete*

Language Models: linguistic prophets

Question: What word would you use to complete the following sentence?

Most likely
I.e. among the options
$$P(\texttt{what}, \texttt{have}, \texttt{you}, \texttt{been}, \texttt{up}, \texttt{to}, \texttt{lately})$$
is highest

What have you been up to ___

- *What have you been up to beehive*
- *What have you been up to lately*
- *What have you been up to N-gram*
- *What have you been up to complete*

Why are LMs useful?

Question: What do these two examples have in common?

“Their are two finals.”

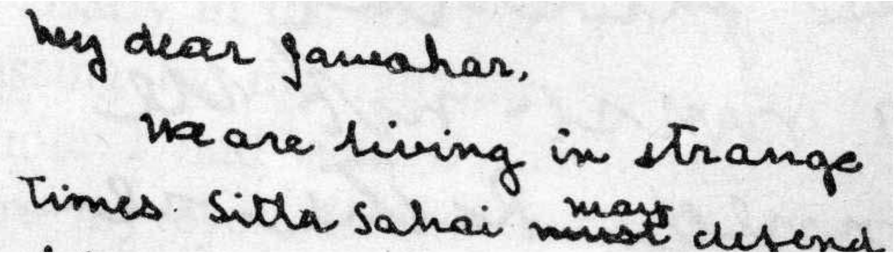

My dear Jawahar,
We are living in strange
Times. Sitla Sahai may
must depend

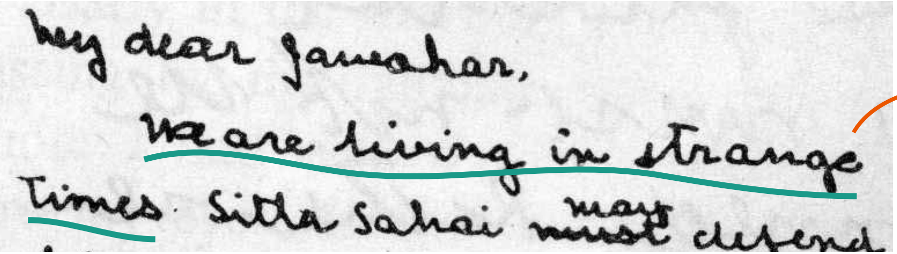
Image credit: Mohandas K. Gandhi - http://www.mkgandhi.org/images/lefthand.JPG, Public Domain, https://commons.wikimedia.org/w/index.php?curid=1455138

Why are LMs useful?

Question: What do these two examples have in common?

- We use our knowledge of likely sentences to correct & disambiguate

"Their are two finals." *"There are two finals."*



"We are living in strange times"

Image credit: Mohandas K. Gandhi - http://www.mkgandhi.org/images/lefthand.JPG, Public Domain, https://commons.wikimedia.org/w/index.php?curid=1455138

When are LMs useful?
- Examples

Human computer interfacing (lecture 12)

Optical character recognition (OCR)

Spelling & grammar correction

Chatbots

Machine translation (MMAI 5500 lecture 9)

Augmentative & alternative communication systems

Text generation (lectures 11 & 12)

Image credit: by NASA/Paul Alers - https://images.nasa.gov/details-200804210005HQ.html, Public Domain, https://commons.wikimedia.org/w/index.php?curid=16350790

Simple Language Models (LM)

- N-grams

N-gram: a sequence of *N* words

1-gram (unigram)

2-gram (bigram)

3-gram (trigram)

…

N-gram

1-gram examples

- “*wake*”, “*up*”

2-gram examples

- “*wake up*”, “*this morning*”, …

3-gram examples

- “*wake up this*”, “*up this morning*”

Question: Is the chain rule below related to the chain rule used to compute gradients?

N-gram LM

Probability of a sequence

Probability of an n word long sequence: $P(w_1\ w_2\ ...\ w_n)$

Count

The probability of a specific n word long sequence can be computed by:

$$P(w_1\ w_2\ ...\ w_n) = \frac{C(w_1\ w_2\ ...\ w_n)}{C(all\ n\ word\ long\ sequences)}$$

However, it is hard to count the number of all n word long sequences.

It is easier to use ***the chain-rule of probability***:

$$P(w_1\ w_2\ ...\ w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1,\ w_2)...\ P(w_n|w_1\ ...\ w_{n\text{-}1})$$

N-gram LM

The chain-rule of probability:

$$P(w_1\ w_2\ \ldots\ w_n) = \overbrace{P(w_1)P(w_2|w_1)P(w_3|w_1\ w_2)\ldots\ P(w_n|w_1\ \ldots\ w_{n\text{-}1})}^{\text{A product of conditional probabilities}}$$

Joint probability written more concisely:

$$P(w_1\ \ldots\ w_n) = P(w_{1:n})$$

A sequence from 1 to n

The chain-rule written more concisely:

$$P(w_{1:n}) = \prod_{t=1}^{n} P(w_t|w_{1:t-1})$$

N-gram example

Example (mirco) corpus:

<s> I am Sam </s>

<s> Sam I am </s>

<s> I do not like green eggs and ham </s>

Bigram: $P(w_t|w_{t\text{-}1})$

example:

$$P(\texttt{I}|\texttt{<s>}) = \frac{C(\texttt{<s> I})}{C(\texttt{<s>})} = \frac{2}{3}$$

<s> is the start token

</s> is the end token

Example (mirco) corpus:

<s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>

N-gram example

Bigram: $P(w_t|w_{t\text{-}1})$

examples:

$$P(\texttt{I}|\texttt{<s>}) = \frac{C(\texttt{<s> I})}{C(\texttt{<s>})} = \frac{2}{3}$$

$$P(\texttt{do}|\texttt{I}) = \frac{C(\texttt{I do})}{C(\texttt{I})} = \frac{1}{3}$$

$$P(\texttt{do}|\texttt{not}) = \frac{C(\texttt{do not})}{C(\texttt{do})} = \frac{1}{1}$$

$$P(\texttt{</s>}|\texttt{Sam}) = \frac{C(\texttt{Sam</s>})}{C(\texttt{Sam})} = \frac{1}{2}$$

Trigram: $P(w_t|w_{t\text{-}1}\ w_{t\text{-}2})$

examples:

$$P(\texttt{do}|\texttt{<s> I}) = \frac{C(\texttt{<s> I do})}{C(\texttt{<s> I})} = \frac{1}{2}$$

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

The probability of the sequence is approximated with the N-gram probabilities.

The LM – $P(w_{1:n})$

Example (mirco) corpus:

<s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>

Bigram LM

$$P(w_{1:n}) \approx \prod_{t=1}^{n} P(w_t | w_{t-1})$$

$$\begin{aligned}
&P(\texttt{<s> I do not}) \\
&\approx P(\texttt{<s> I})P(\texttt{I do})P(\texttt{do not}) \\
&= \frac{C(\texttt{<s> I})}{C(\texttt{<s>})}\frac{C(\texttt{I do})}{C(\texttt{I})}\frac{C(\texttt{do not})}{C(\texttt{do})} \\
&= \frac{2}{3} \times \frac{1}{3} \times \frac{1}{1} \\
&\approx 0.22
\end{aligned}$$

Trigram LM

$$P(w_{1:n}) \approx \prod_{t=1}^{n} P(w_t | w_{t-2:t-1})$$

$$\begin{aligned}
&P(\texttt{<s> I am Sam </s>}) \\
&\approx P(\texttt{<s> <s> I})P(\texttt{<s> I am})P(\texttt{I am Sam})P(\texttt{am Sam </s>}) \\
&= \frac{C(\texttt{<s> <s> I})}{C(\texttt{<s> <s>})}\frac{C(\texttt{<s> I am})}{C(\texttt{<s> I})}\frac{C(\texttt{I am Sam})}{C(\texttt{I am})}\frac{C(\texttt{am Sam </s>})}{C(\texttt{am Sam})} \\
&= \frac{2}{3} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{1} \\
&\approx 0.17
\end{aligned}$$

Generating sentences with language models

Bigram example

1. Start with a *start token*: `<s>`
2. Randomly draw a new word according to $P(w^i|\texttt{<s>})\forall i \in V$
3. Append the word w to the sequence
4. Randomly draw the next word according to $P(w_t^i|w_{t-1})\forall i \in V$
5. Repeat steps 3 & 4
6. End when the *end token*, `</s>`, is drawn

Start (`<s>`) and end (`</s>`) tokens are part of the vocabulary V

The vocabulary V is a list of unique/distinct words in the dataset.

Question: What is the most likely sequence, starting with <s>?

Generating a sequence

- example

Vocabulary

$$V = \{\texttt{go}, \texttt{</s>}, \texttt{here}, \texttt{we}, \texttt{<s>}\}$$

Bigram probabilities

w_t \ w_{t-1}	go	</s>	here	we	<s>
go	0.05	0	0.35	0.2	0.15
</s>	0.40	0	0.30	0.2	0.05
here	0.25	0	0.30	0.4	0.35
we	0.30	0	0.05	0.2	0.45
<s>	0	0	0	0	0

Limits of N-grams

Would a good **bi-gram** model assign the following "sentence" high or low probability?

It is it is it is it is it is it is it is it is.

Why?

Limits of N-grams

Would a good **tri-gram** model assign the following "sentence" high or low probability?

It is it is it is it is it is it is it is it is.

Why?

Limits of N-grams
- data requirement

Higher order N-grams capture longer range dependencies between words

But, the number of unique possibilities of 3, 4, & 5-word sequences increases exponentially

- An example of the combinatorial explosion

Thus, **exponentially more data is required to get non-zero counts** of all/nearly all sequences

Question: Which of the sentences below sound most likely?

Limits of N-grams

- the generative property of language

Natural language is generative.

I.e. we can generate novel sentences that still make sense

Examples. Which of the sentences below are most likely?

- *"The arctic char is Lapland's most beautiful fish."* **vs** *"The beautiful most arctic Lapland's fish char is."*
- *"Is, angry Norway & the small suicidal adorable lemming."* **vs** *"The suicidal Norway lemming is small, angry & adorable."*

Limits of N-grams
- the generative property of language


Natural language is generative.

I.e. we can generate novel sentences that still make sense

Examples. Which of the sentences below are most likely?

- *"The arctic char is Lapland's most beautiful fish."* **vs** *"The beautiful most arctic Lapland's fish char is."*
- *"Is, angry Norway & the small suicidal adorable lemming."* **vs** *"The suicidal Norway lemming is small, angry & adorable."*

* example from the textbook section 3.4 based on the WSJ Treebank3 corpus

Slide 37

Limits of N-grams
- zero counts*

Training set (**bigram** *continuation*: count)

- **denied the** *allegations*:5
- **denied the** *speculation*: 2
- **denied the** *rumors*: 1
- **denied the** *report*: 1

Hypothetical test set (**bigram** *continuation*)

- **denied the** *offer*
- **denied the** *loan*

Two problems: 1) obviously sensible 3-grams were assigned 0 probability; 2) *perplexity* is undefined

What to do with sequences (e.g. trigrams) missing from the training set?

Dealing with unknown words

Closed vocabulary

- The test set can only contain words from a fixed size, known vocabulary

Open vocabulary

- Potentially unknown words can occur in the test set
- We allow for unknown/out-of-vocabulary words
- All unknown words are replaced by the `<UNK>` token

Unknown word

- Words that occur in the test set, but **NOT** in the training set
- Or (not xor) words that are very rare in training set

Two strategies

- Replace all words not in a fixed vocabulary with `<UNK>`
- Replace all words below some frequency with `<UNK>`

Dealing with zero counts
- smoothing

Add-1 smoothing aka Laplace smoothing

- Add 1 to all N-gram counts & normalize by the number of N-gram counts
- Results in a smoother probability distribution with no zeroes & lower max
- With many zeroes, too much probability mass is shifted resulting in bad estimates
- Not good performance & is not in common use

Dealing with zero counts
- smoothing

Backoff

- If counts for a higher-order N-gram is missing, then use the counts for a lower-order instead
- E.g. if the count for the tri-gram $w_{n\text{-}2}$, $w_{n\text{-}1}$, w_n is missing, then we can use the count for $w_{n\text{-}1}$, w_n instead.
- Example: if the sequence “*suicidal Norway lemming*” does not occur then we can back off to “*Norway lemming*”

Further improvement: we can use a weighted average over all N-grams (uni, bi, tri, ...) in place of a high-order N-gram (called interpolation in the textbook)

Google N-grams

Released 2006 (before deep learning NLP)

1 to 5-grams

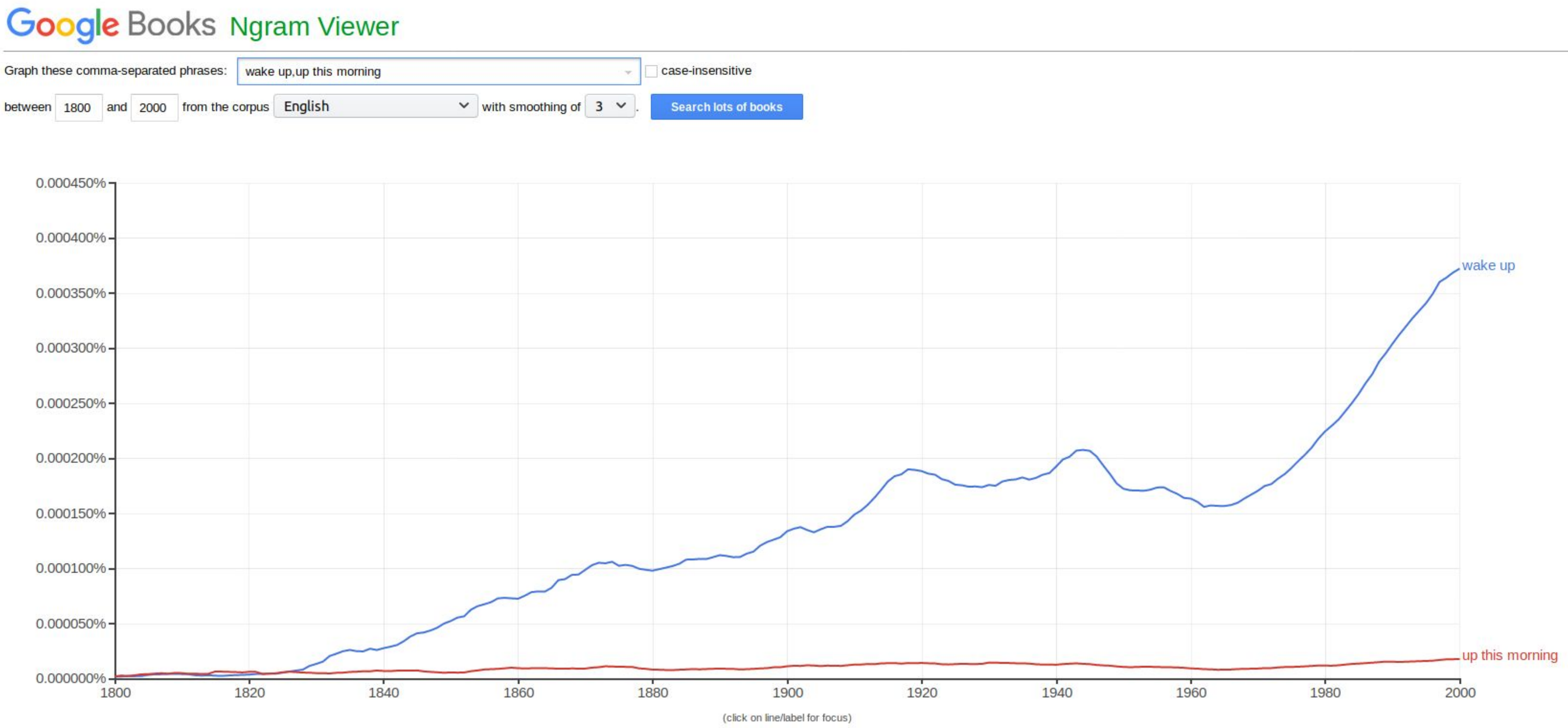
From 1,024,908,267,229 words of running text

Based on all 5-word sequences (1,176,470,663) that appear at least 40 times

For a data set of this size smoothing has to be fast, & not too sophisticated

Google N-gram viewer

Question: Why do you think "*wake up*" has a higher probability than "*up this morning*"?



Language models
- Evaluation

Extrinsic

- Deploy the LM in an application & see how much the application improves
- Only way of knowing if the LM really helps the task
- E.g.
  - Task: speech recognition
  - Two LMs
  - Run the task with each model & compare results

Intrinsic

- Use an evaluation metric that is independent of the application
- Evaluate on a test set
- Best model: the LM assigning the highest probability to the test set

Actually: lowest perplexity

Question: Why normalize by the number of words?

Language models

- Evaluation metric

Instead of raw probabilities we use **perplexity** (*PP*)

- *PP* is the inverse probability of the test set normalized by the number of words

$$test\ set\ W_{test} = w_1, w_2, ..., w_N$$

$$PP(W_{test}) = \sqrt[N]{\frac{1}{P(w_1, w_2, ..., w_N)}}$$

Language models

- Generalization

How to select training data?

- An LM will reflect the corpus (data set) it is trained on
- Trivial example: an LM trained on a corpus of *Swedish* texts will be useless for *English* sentence generation
- Better example: an LM trained on Shakespeare won't do good on tasks involving modern legal documents

Solutions

1. Train on a corpus similar to the future application
2. Train on a huge corpus containing all kinds of texts (e.g. GPT-3* was trained on 2 TB text)

* GPT-3 is a big *neural* LM aka an LLM

Terminology

Language model

Vocabulary

N-gram

- The N-gram vs the N-gram LM

Perplexity

Extrinsic evaluation

Intrinsic evaluation

Chain-rule of probability

LLM

Book exercises

Do the following exercises from the book:

- 3.1
- 3.2
- 3.3
- 3.4

You will be tested on similar problems on the midterm

Tutorial

`MMAI5400_class03_ngrams.ipynb`