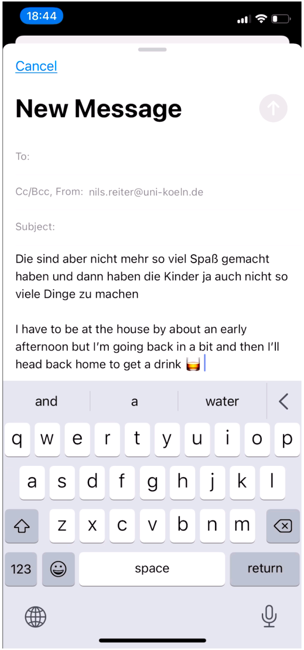
Language Modeling

VL Sprachliche Informationsverarbeitung

Nils Reiter
nils.reiter@uni-koeln.de

November 3, 2022
Winter term 2022/23

INSTITUT FÜR
DIGITAL HUMANITIES
UNIVERSITÄT ZU KÖLN


18:44
Cancel
New Message
To:
Cc/Bcc, From: nils.reiter@uni-koeln.de
Subject:
Die sind aber nicht mehr so viel Spaß gemacht haben und dann haben die Kinder ja auch nicht so viele Dinge zu machen
I have to be at the house by about an early afternoon but I'm going back in a bit and then I'll head back home to get a drink 🥃
and
a
water
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
return

Introduction

- One of the oldest NLP tasks
  - Long before predictive typing on smart phones became a thing
- Language model (LM) predicts the next word, given previous words (history)
  - Bidirectional LM: Previous and following words (context)
- Formally: $p(\text{word}|\text{history})$

Introduction

- One of the oldest NLP tasks
  - Long before predictive typing on smart phones became a thing
- Language model (LM) predicts the next word, given previous words (history)
  - Bidirectional LM: Previous and following words (context)
- Formally: $p(\text{word}|\text{history})$

Example

Sue swallowed the large green ___

Introduction

- One of the oldest NLP tasks
  - Long before predictive typing on smart phones became a thing
- Language model (LM) predicts the next word, given previous words (history)
  - Bidirectional LM: Previous and following words (context)
- Formally: $p(\text{word}|\text{history})$

Example

Sue swallowed the large green ___

Reading

Christopher D. Manning/Hinrich Schütze (1999). *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts and London, England: MIT Press, Ch. 6.1–6.2. Ilias

History

- Not all textual histories can be treated individually
  - We couldn't predict anything on completely new histories
  - Chance of a text re-appearing is astronomically slim
- Predicting the next word on unseen sentences requires *generalization*

History

- Not all textual histories can be treated individually
  - We couldn't predict anything on completely new histories
  - Chance of a text re-appearing is astronomically slim
- Predicting the next word on unseen sentences requires *generalization*
- Instances of textual histories need to be grouped together
  - Manning/Schütze (MS99, 192): "Equivalence Classes"

History

- Not all textual histories can be treated individually
  - We couldn't predict anything on completely new histories
  - Chance of a text re-appearing is astronomically slim
- Predicting the next word on unseen sentences requires *generalization*
- Instances of textual histories need to be grouped together
  - Manning/Schütze (MS99, 192): "Equivalence Classes"

$$\text{More generalization} \xleftarrow{\text{less}} \text{Equivalence classes} \xrightarrow{\text{more}} \text{More discrimination}$$

Figure: Compromise between generalization and discrimination

Forming Equivalence Classes

Different strategies

- Stemming/lemmatization: Don't look at word forms, look at lemmas or stems
  - E.e.: $p(\text{bark}|\text{the dog})$ instead of $p(\text{barks}|\text{The dog})$

Forming Equivalence Classes

Different strategies

- Stemming/lemmatization: Don't look at word forms, look at lemmas or stems
  - E.e.: $p(\text{bark}|\text{the dog})$ instead of $p(\text{barks}|\text{The dog})$
- Selected history: Only look at selected word classes
  - Content words like nouns, verbs, adjectives and adverbs
  - E.g., $p(\text{barks}|\text{dog})$ instead of $p(\text{barks}|\text{The dog})$

Forming Equivalence Classes

Different strategies

- Stemming/lemmatization: Don't look at word forms, look at lemmas or stems
  - E.e.: $p(\text{bark}|\text{the dog})$ instead of $p(\text{barks}|\text{The dog})$
- Selected history: Only look at selected word classes
  - Content words like nouns, verbs, adjectives and adverbs
  - E.g., $p(\text{barks}|\text{dog})$ instead of $p(\text{barks}|\text{The dog})$
- Both require linguistic pre-analysis of the text
  - Time-consuming and error-prone (on a large scale)

Forming Equivalence Classes

Different strategies

- Stemming/lemmatization: Don't look at word forms, look at lemmas or stems
  - E.e.: $p(\text{bark}|\text{the dog})$ instead of $p(\text{barks}|\text{The dog})$
- Selected history: Only look at selected word classes
  - Content words like nouns, verbs, adjectives and adverbs
  - E.g., $p(\text{barks}|\text{dog})$ instead of $p(\text{barks}|\text{The dog})$
- Both require linguistic pre-analysis of the text
  - Time-consuming and error-prone (on a large scale)
- Limit history: Only look at the last n words

Markov Assumption

- Assumption: Only the local context influences the next word W Markov property

Markov Assumption

- Assumption: Only the local context influences the next word (W Markov property)
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1\rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2\rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3\rangle)$

Markov Assumption

- Assumption: Only the local context influences the next word — W Markov property
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1\rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2\rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3\rangle)$

Example

Bigram model: "green ___"

Markov Assumption

- Assumption: Only the local context influences the next word — Markov property
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1\rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2\rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3\rangle)$

Example

Trigram model: "large green ___"

Markov Assumption

- Assumption: Only the local context influences the next word — Markov property
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1\rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2\rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3\rangle)$

Example

4-gram model: “the large green ___”

Markov Assumption

- Assumption: Only the local context influences the next word WMarkov property
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1\rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2\rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3\rangle)$

Example

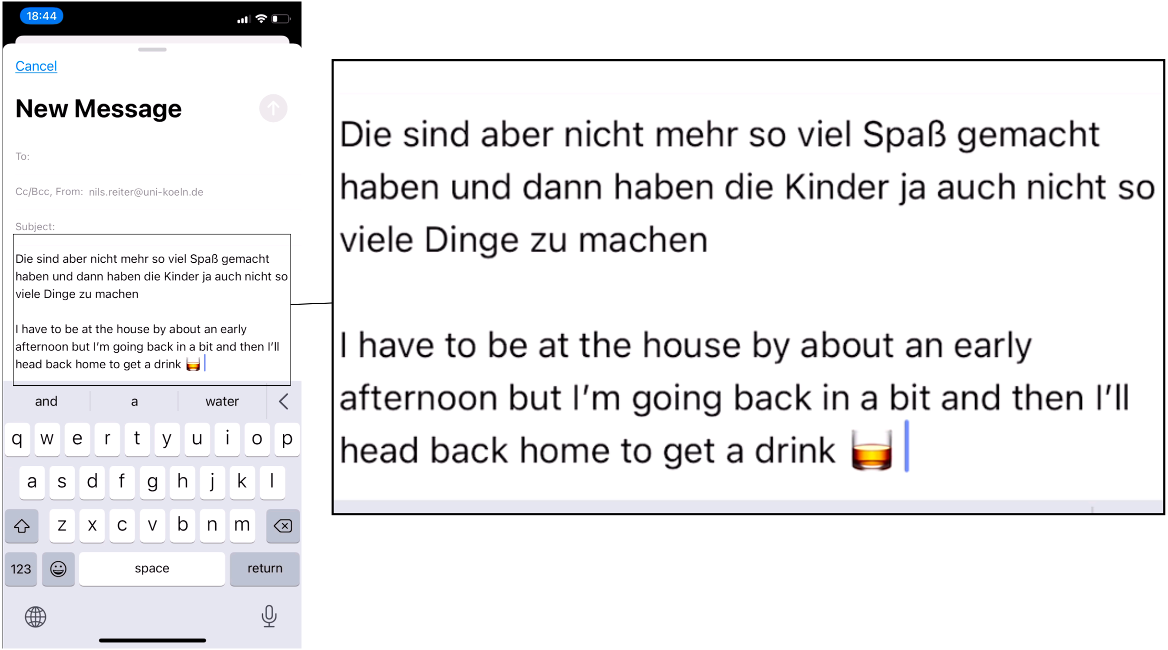
5-gram model: “swallowed the large green ___”

Markov Assumption

- Assumption: Only the local context influences the next word — Markov property
- n-gram model: Only the last $n-1$ words are looked at to predict the nth word
  - Bigram model: $p(w_2|\langle w_1 \rangle)$
  - Trigram model: $p(w_3|\langle w_1, w_2 \rangle)$
  - 4-gram model: $p(w_4|\langle w_1, w_2, w_3 \rangle)$

Example

6-gram model: "Sue swallowed the large green ___"


18:44
Cancel
New Message
To:
Cc/Bcc, From: nils.reiter@uni-koeln.de
Subject:
Die sind aber nicht mehr so viel Spaß gemacht haben und dann haben die Kinder ja auch nicht so viele Dinge zu machen
I have to be at the house by about an early afternoon but I'm going back in a bit and then I'll head back home to get a drink
and
a
water
space
return
123

Increasing n

- The higher n, the better?
- Storage and training time increases
  - Number of parameters: Number of numbers (frequencies/probabilities) we need to store separately

Increasing n

- The higher n, the better?
- Storage and training time increases
  - Number of parameters: Number of numbers (frequencies/probabilities) we need to store separately
- Assuming a vocabulary of $20\,000$ words ($=$ types)

Rechtschreibduden: $140\,000$

  - Bigram model: $20\,000^2 = 400\,000\,000$ parameters
  - Trigram model: $20\,000^3 = 8\,000\,000\,000\,000 = 8 \times 10^{12}$ parameters
  - 4-gram model: $20\,000^4 = 1.6 \times 10^{17}$ parameters

Increasing n

- The higher n, the better?
- Storage and training time increases
  - Number of parameters: Number of numbers (frequencies/probabilities) we need to store separately
- Assuming a vocabulary of $20\,000$ words (= types)

Rechtschreibduden: $140\,000$

  - Bigram model: $20\,000^2 = 400\,000\,000$ parameters (= ca. 50 MB)
  - Trigram model: $20\,000^3 = 8\,000\,000\,000\,000 = 8 \times 10^{12}$ parameters (= ca. 8 GB)
  - 4-gram model: $20\,000^4 = 1.6 \times 10^{17}$ parameters (= ca. 20 PB)

Again, a Compromise

- Longer n-grams would give better predictions
- Shorter n-grams would be easier/faster to train and use

Again, a Compromise

- Longer n-grams would give better predictions
- Shorter n-grams would be easier/faster to train and use
- Common: $n = 2$ or $n = 3$
  - Trigrams are surprisingly good at predicting the next word!

- Where do we actually get these probabilities from?
  - Corpora.

Training

- Where do we actually get these probabilities from?
  - Corpora.
- Training
  - Count frequences of features from data
  - Convert them into probabilities, maybe apply mathematical transformations

Training

- Where do we actually get these probabilities from?
  - Corpora.
- Training
  - Count frequences of features from data
  - Convert them into probabilities, maybe apply mathematical transformations
- Definition of conditional probabilities:

$$p(w_n|\langle w_1, \ldots, w_{n-1}\rangle) = \frac{p(\langle w_1, \ldots, w_n\rangle)}{p(\langle w_1, \ldots, w_{n-1}\rangle)}$$

Maximum Likelihood Estimation (MLE)

- Parameters that maximize probability on the training corpus
- I.e., use the relative frequency from the training corpus as probability

$$p(\langle w_1, \ldots, w_n \rangle) = \frac{c(\langle w_1, \ldots, w_n \rangle)}{N}$$

Maximum Likelihood Estimation (MLE)

- Parameters that maximize probability on the training corpus
- I.e., use the relative frequency from the training corpus as probability

$$
\begin{aligned}
p(\langle w_1, \ldots, w_n \rangle) &= \frac{c(\langle w_1, \ldots, w_n \rangle)}{N} \\
p(w_n | \langle w_1, \ldots w_{n-1}) &= \frac{p(\langle w_1, \ldots, w_n \rangle)}{p(\langle w_1, \ldots, w_{n-1} \rangle)}
\end{aligned}
$$

demo

Maximum Likelihood Estimation (MLE)

Example

History	w_n	Count
Bier und	Wein	4
Bier und	Schnaps	3
Bier und	Bratwürsten	1
Bier und	Männerschweiß	1
Bier und	nichtalkoholischen	1
...	...	1
Bier	und	29

Maximum Likelihood Estimation (MLE)

Example

History	w_n	Count
Bier und	Wein	4
Bier und	Schnaps	3
Bier und	Bratwürsten	1
Bier und	Männerschweiß	1
Bier und	nichtalkoholischen	1
...	...	1
Bier	und	29

$$
\begin{aligned}
p(\text{Bier und}) &= \frac{22}{1880232} \\
p(\text{Wein}|\text{Bier und}) &= \frac{p(\text{Bier und Wein})}{p(\text{Bier und})} \\
&= \frac{\frac{4}{1880232}}{\frac{22}{1880232}} \\
&= \frac{4}{1880232} \times \frac{1880232}{22} \\
&= \frac{4}{22} = 0.1818
\end{aligned}
$$

Application

- Training corpus used for estimating probability
- Test/application corpus used for using probability
- Never use the same corpus for training and testing

Application

- Training corpus used for estimating probability
- Test/application corpus used for using probability
- Never use the same corpus for training and testing
- After having trained, we can check how probable a new document/corpus is (= test/application)

Example

$$
\begin{aligned}
p(\text{Ich trinke gerne Bier und Wein}) &= p(\text{Ich}|\text{SYM SYM}) \times p(\text{trinke}|\text{Ich SYM}) \\
&\times p(\text{gerne}|\text{Ich trinke}) \times p(\text{Bier}|\text{trinke gerne}) \\
&\times p(\text{und}|\text{gerne Bier}) \times p(\text{Wein}|\text{Bier und})
\end{aligned}
$$

Maximum Likelihood Estimation (MLE)

Drawbacks

- What happens with words not in the training corpus? Zero probability
  - ‘out of vocabulary’ (OOV)
- Because of multiplication, everything will be zero

Maximum Likelihood Estimation (MLE)

Drawbacks

- What happens with words not in the training corpus? Zero probability
  - 'out of vocabulary' (OOV)
- Because of multiplication, everything will be zero
- There will be OOV words – because Zipf
- MLE conceptually important, but rarely used in NLP

$\Rightarrow$ We need another estimator for the probability

Lidstone's Law

- Core problem: All probability mass is used on words in vocabulary
- Nothing left for OOV words in test/application
- OOV words need to receive a probability > 0

Lidstone's Law

- Core problem: All probability mass is used on words in vocabulary
- Nothing left for OOV words in test/application
- OOV words need to receive a probability > 0

$$p(\langle w_1, \dots w_n \rangle) = \frac{c(\langle w_1, \dots w_n \rangle) + \lambda}{N + B\lambda}$$

Lidstone's Law

- Core problem: All probability mass is used on words in vocabulary
- Nothing left for OOV words in test/application
- OOV words need to receive a probability > 0

$$p(\langle w_1, \ldots w_n\rangle) = \frac{c(\langle w_1, \ldots w_n\rangle) + \lambda}{N + B\lambda}$$

- B: Number of different n-grams (i.e., n-gram types)
- λ: Parameter set to control how much mass remains for OOV words
  - Typical setting: $\lambda = \frac{1}{2}$ (for reasons see MS99, 204)

Smoothing

- Lidstone's law is a 'smoothing' technique
- Goal
  - Prevent zero probabilities
  - Reserve some amount of probability mass for OOV words

Smoothing

- Lidstone's law is a 'smoothing' technique
- Goal
  - Prevent zero probabilities
  - Reserve some amount of probability mass for OOV words
- Different strategies
  - Often need for fine-tuning (e.g., what value to we use for λ?)

Neural Language Models

- Based on neural networks
- Efficient matrix handling
- Bidirectional
- Use of sub words
- Attention: Not all contextual words are equally important

Summary

- Language modeling
  - Given some history, predict the next word
  - Use cases: Smart phone, ...
  - Maximum Likelihood estimation: Easy, but problematic
  - Lidstone's Law: Smoothing
    - Other smoothing techniques exist
- Different data sets for different purposes
  - Cross validation