IN5550: Neural Methods in Natural Language Processing
Sub-lecture 4.3
Pre-neural language models

Andrey Kutuzov

University of Oslo

10 February 2022



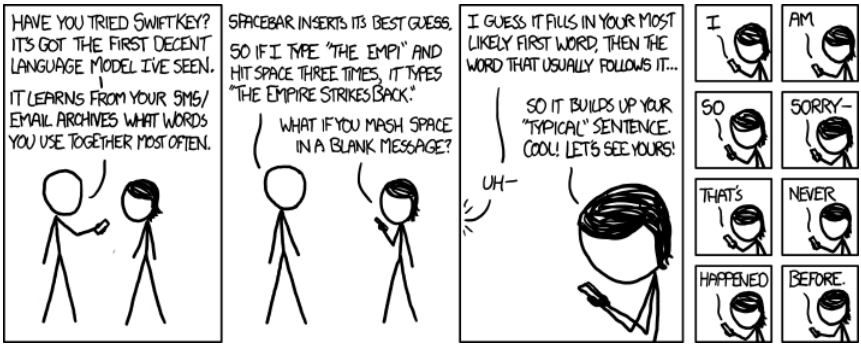
Contents



1. Language modeling
   - Language modeling task definition
   - Traditional approach to LM

Language modeling

Predicting the next word in the text given the previous words:



(XKCD)

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:

Language modeling task definition



Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'
  - 'What is the probability of *The quick brown fox jumps over the lazy dog*?'

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'
  - 'What is the probability of *The quick brown fox jumps over the lazy dog*?'
  - 'What is the probability of *green colorless ideas sleep furiously*?'

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'
  - 'What is the probability of *The quick brown fox jumps over the lazy dog*?'
  - 'What is the probability of *green colorless ideas sleep furiously*?'
- Task 2: to assign a probability for the likelihood of a word a to follow a word sequence S of length n:

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'
  - 'What is the probability of *The quick brown fox jumps over the lazy dog*?'
  - 'What is the probability of *green colorless ideas sleep furiously*?'
- Task 2: to assign a probability for the likelihood of a word a to follow a word sequence S of length n:
  - 'What is the probability of seeing *jumps* after *The quick brown fox*?'

Language modeling task definition

Modeling linguistic sequences

- Task 1: to assign probabilities to natural language sequences:
  - 'What is the probability of *lazy dog*?'
  - 'What is the probability of *The quick brown fox jumps over the lazy dog*?'
  - 'What is the probability of *green colorless ideas sleep furiously*?'
- Task 2: to assign a probability for the likelihood of a word a to follow a word sequence S of length n:
  - 'What is the probability of seeing *jumps* after *The quick brown fox*?'
- These two tasks are mathematically equivalent.

$$P(w_{1:n}) = P(w_1)P(w_2|w_1)P(w_3|w_{1:2})P(w_4|w_{1:3})...P(w_n|w_{1:n-1}) \quad (1)$$

Language modeling task definition



Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.

Language modeling task definition



Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.
- Hence, the Markov assumption a.k.a. Markov property:
  - **future is independent of the past given the present**.

Language modeling task definition



Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.
- Hence, the Markov assumption a.k.a. Markov property:
  - **future is independent of the past given the present**.
- In LM context: we can look at only the last k words.

Language modeling task definition



Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.
- Hence, the Markov assumption a.k.a. Markov property:
  - **future is independent of the past given the present**.
- In LM context: we can look at only the last k words.
- It is a simplification, but it produces good results anyway.

Language modeling task definition



Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.
- Hence, the Markov assumption a.k.a. Markov property:
  - **future is independent of the past given the present**.
- In LM context: we can look at only the last k words.
- It is a simplification, but it produces good results anyway.

- Language modeling is widely used: text messaging, machine translation, chat-bots, summarization...., representation learning!

Language modeling task definition

Markov assumption

- Multiplying hundreds or thousands of probabilities can be cumbersome.
- Hence, the Markov assumption a.k.a. Markov property:
  - **future is independent of the past given the present**.
- In LM context: we can look at only the last k words.
- It is a simplification, but it produces good results anyway.

- Language modeling is widely used: text messaging, machine translation, chat-bots, summarization...., representation learning!
- LMs are measured by perplexity:
  - how surprised is the model by test word sequences, the lower the better.
- For a test corpus of n word tokens:

$$probs = \sum_{i=1}^{n} \log_2 LM\left(w_i | w_{1:i-1}\right) \tag{2}$$

$$perplexity = 2^{-\frac{1}{probs}}$$

Traditional approach to LM



Old way: extract probabilities from corpus counts

1. Take a large enough corpus;

Traditional approach to LM



Old way: extract probabilities from corpus counts

1. Take a large enough corpus;
2. count all sequences;

Traditional approach to LM



Old way: extract probabilities from corpus counts

1. Take a large enough corpus;
2. count all sequences;
3. use maximum likelihood estimate for each word m:
   $\hat{P}\left((w_{i+1} = m)\,|w_{i-k:i}\right) = \frac{\#(w_{i-k:i+1})}{\#(w_{i-k:i})}$
4. where $\#$ are corpus counts.

Traditional approach to LM



Old way: extract probabilities from corpus counts

1. Take a large enough corpus;
2. count all sequences;
3. use maximum likelihood estimate for each word m:
   $\hat{P}\left((w_{i+1} = m)\,|w_{i-k:i}\right) = \frac{\#(w_{i-k:i+1})}{\#(w_{i-k:i})}$
4. where $\#$ are corpus counts.
5. Et voila! You have probabilities for all seen words given previous sequences:
   $\hat{P}\left((w_4 = \mathit{jumps})\,|[\mathit{the}, \mathit{quick}, \mathit{brown}, \mathit{fox}]\right) = 0.5$

Traditional approach to LM

Old way: extract probabilities from corpus counts

1. Take a large enough corpus;
2. count all sequences;
3. use maximum likelihood estimate for each word m:
   $\hat{P}\left((w_{i+1} = m) \,|w_{i-k:i}\right) = \frac{\#(w_{i-k:i+1})}{\#(w_{i-k:i})}$
4. where $\#$ are corpus counts.
5. Et voila! You have probabilities for all seen words given previous sequences:
   $\hat{P}\left((w_4 = jumps) \,|[the, quick, brown, fox]\right) = 0.5$
   - .. because your corpus had 2 occurrences of '*the quick brown fox*', and in one case it was followed by '*jumps*', while in another by '*barks*'.

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.
- Number of parameters increases exponentially with increasing k.

Traditional approach to LM



Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.
- Number of parameters increases exponentially with increasing k.
- Words are discrete features:

Traditional approach to LM

Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.
- Number of parameters increases exponentially with increasing k.
- Words are discrete features:
  - Representation power not shared between similar words

Traditional approach to LM

Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.
- Number of parameters increases exponentially with increasing k.
- Words are discrete features:
  - Representation power not shared between similar words
  - If we saw '*fox eats*' and '*dog eats*' 1000 times each, but never saw '*wolf eats*', the probability of '*wolf eats*' will still be 0.

Traditional approach to LM

Many shortcomings

- Sequence not seen in the training data? $\hat{P} = 0$
- There are ways to deal with unseen events...
  - but they are tricky...
  - ...and do not scale well to larger n-grams.
- Unseen events become more frequent as one increases k;
- number of possible word combinations is $|V|^k$;
- for the vocabulary of 10 000 words and 5-grams: 10000^5.
- Number of parameters increases exponentially with increasing k.
- Words are discrete features:
  - Representation power not shared between similar words
  - If we saw '*fox eats*' and '*dog eats*' 1000 times each, but never saw '*wolf eats*', the probability of '*wolf eats*' will still be 0.

Deep learning comes to help in the next sub-lecture 4.4!