

# Natural Language Processing 1

## Lecture 2: Language models and part-of-speech tagging

Katia Shutova

ILLC  
University of Amsterdam

28 October 2020

# Outline.

Probabilistic language modelling

Part-of-speech (POS) tagging

Q&A and Discussion

## Modelling word sequences

- ▶ We have seen the bag-of-words technique
- ▶ where each word is treated as independent from its context
- ▶ In reality, word likelihood depends on context
- ▶ This lecture introduces shallow **syntax:**  
**language modelling**, i.e. modelling word sequences using statistical techniques

## Statistical techniques: NLP and linguistics

*But it must be recognized that the notion ‘probability of a sentence’ is an entirely useless one, under any known interpretation of this term. (Chomsky 1969)*

*Whenever I fire a linguist our system performance improves. (Jelinek, 1988: reported)*

## Statistical techniques: NLP and linguistics

*But it must be recognized that the notion ‘probability of a sentence’ is an entirely useless one, under any known interpretation of this term. (Chomsky 1969)*

*Whenever I fire a linguist our system performance improves. (Jelinek, 1988: reported)*

# Corpora

- ▶ **corpus**: text that has been collected for some purpose.
- ▶ **balanced corpus**: texts representing different genres  
**genre** is a type of text (vs domain)
- ▶ **tagged corpus**: a corpus annotated with e.g. POS tags
- ▶ **treebank**: a corpus annotated with parse trees
- ▶ specialist corpora — e.g., collected to train or evaluate particular applications
  - ▶ Movie reviews for sentiment classification
  - ▶ Data collected from simulation of a dialogue system

# Language modelling and word prediction

Guess the missing word:

Wright tells her story with great \_\_\_\_\_.

# Language modelling and word prediction

Guess the missing word:

Wright tells her story with great professionalism.



## Uses of language modelling

- ▶ speech recognition to disambiguate results from signal processing:
  - ▶ *have an ice Dave*
  - ▶ *heaven ice day*
  - ▶ *have a nice day*
- ▶ word prediction for communication aids:  
e.g., to help enter text that's input to a synthesiser
- ▶ text entry on mobile devices
- ▶ spelling correction
- ▶ ...

## Uses of language modelling

- ▶ speech recognition to disambiguate results from signal processing:
  - ▶ *have an ice Dave*
  - ▶ *heaven ice day*
  - ▶ *have a nice day*
- ▶ word prediction for communication aids:  
e.g., to help enter text that's input to a synthesiser
- ▶ text entry on mobile devices
- ▶ spelling correction
- ▶ ...

## Uses of language modelling

- ▶ speech recognition to disambiguate results from signal processing:
  - ▶ *have an ice Dave*
  - ▶ *heaven ice day*
  - ▶ *have a nice day*
- ▶ word prediction for communication aids:  
e.g., to help enter text that's input to a synthesiser
- ▶ text entry on mobile devices
- ▶ spelling correction
- ▶ ...

## n-grams

**Bigram**: n-gram with  $N=2$

- ▶ A probability is assigned to a word **based on the previous word**:

$$P(w_n | w_{n-1})$$

where  $w_n$  is the  $n$ th word in a sentence.

- ▶ Probability of a sequence of words (assuming **independence**):

$$P(W_1^n) \approx \prod_{k=1}^n P(w_k | w_{k-1})$$

## n-grams

**Trigram**: n-gram with  $N=3$

- ▶ A probability is assigned to a word **based on two previous words**:

$$P(w_n | w_{n-1} w_{n-2})$$

where  $w_n$  is the  $n$ th word in a sentence.

- ▶ Probability of a sequence of words (assuming **independence**):

$$P(W_1^n) \approx \prod_{k=1}^n P(w_k | w_{k-1} w_{k-2})$$

## bigrams: probability estimation

**Maximum likelihood** estimation:

Probability is estimated from counts in a training corpus:

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{\sum_w C(w_{n-1} w)} \approx \frac{C(w_{n-1} w_n)}{C(w_{n-1})}$$

i.e. count of a particular bigram in the corpus divided by the count of all bigrams starting with the prior word.

$\langle s \rangle$  good morning  $\langle /s \rangle$   $\langle s \rangle$  good afternoon  $\langle /s \rangle$   $\langle s \rangle$  good  
 afternoon  $\langle /s \rangle$   $\langle s \rangle$  it is very good  $\langle /s \rangle$   $\langle s \rangle$  it is good  $\langle /s \rangle$

sequence	count	bigram probability
$\langle s \rangle$	5	
$\langle s \rangle$ good	3	.6
$\langle s \rangle$ it	2	.4
good	5	
good morning	1	.2
good afternoon	2	.4
good $\langle /s \rangle$	2	.4

...

## Sentence probabilities

⟨s⟩ good morning ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩ ⟨s⟩ good  
afternoon ⟨/s⟩ ⟨s⟩ it is very good ⟨/s⟩ ⟨s⟩ it is good ⟨/s⟩

Probability of ⟨s⟩ it is good afternoon ⟨/s⟩ is estimated as:

$$P(\text{it}|\langle s \rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle /s \rangle|\text{afternoon})$$
$$= .4 \times 1 \times .5 \times .4 \times 1 = .08$$

What about the probability of ⟨s⟩ very good ⟨/s⟩ ?

$$P(\text{very}|\langle s \rangle)?$$



## Sentence probabilities

⟨s⟩ good morning ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩ ⟨s⟩ good  
afternoon ⟨/s⟩ ⟨s⟩ it is very good ⟨/s⟩ ⟨s⟩ it is good ⟨/s⟩

Probability of ⟨s⟩ it is good afternoon ⟨/s⟩ is estimated as:

$$P(\text{it}|\langle s \rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle /s \rangle|\text{afternoon})$$
$$= .4 \times 1 \times .5 \times .4 \times 1 = .08$$

What about the probability of ⟨s⟩ very good ⟨/s⟩ ?

$$P(\text{very}|\langle s \rangle)?$$

## Sentence probabilities

⟨s⟩ good morning ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩  
⟨s⟩ it is very good ⟨/s⟩ ⟨s⟩ it is good ⟨/s⟩

Probability of ⟨s⟩ it is good afternoon ⟨/s⟩ is estimated as:

$$P(\text{it}|\langle s \rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle /s \rangle|\text{afternoon}) \\ = .4 \times 1 \times .5 \times .4 \times 1 = .08$$

What about the probability of ⟨s⟩ very good ⟨/s⟩ ?

$$P(\text{very}|\langle s \rangle)?$$

## Sentence probabilities

Problems because of **sparse data**:

- ▶ **smoothing**: distribute 'extra' probability between rare and unseen events
- ▶ **backoff and interpolation**: approximate unseen probabilities by a more general probability, e.g. unigrams

cf Chomsky: *Colorless green ideas sleep furiously*  
smoothing means unseen phrases have a non-zero probability estimate.

## Sentence probabilities

Problems because of **sparse data**:

- ▶ **smoothing**: distribute 'extra' probability between rare and unseen events
- ▶ **backoff and interpolation**: approximate unseen probabilities by a more general probability, e.g. unigrams

cf Chomsky: *Colorless green ideas sleep furiously*  
smoothing means unseen phrases have a non-zero probability estimate.

## Laplace (add 1) smoothing

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

- ▶ simple to implement, **BUT**
- ▶ only suitable for problems with few unseen events
- ▶ we have a lot of unseen n-grams

But add-1 is used to smooth other NLP models:

- ▶ e.g. for text classification
- ▶ in domains where the number of zeros isn't so huge

# Backoff and Interpolation

- ▶ Sometimes it helps to use **less context**
  - ▶ Condition on less context for contexts you haven't learned much about
- ▶ **Backoff**
  - ▶ use trigram if you have good evidence,
  - ▶ otherwise bigram, otherwise unigram
- ▶ **Interpolation**
  - ▶ mix unigram, bigram, trigram
  - ▶ Interpolation works better

## Linear interpolation

- ▶ Combine **different order n-grams**
- ▶ by linearly interpolating all the models:

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

such that  $\sum_i \lambda_i = 1$

- ▶  $\lambda$ s are learned from a held-out corpus

## More options

Advanced smoothing methods:

- ▶ Absolute discounting
- ▶ Good Turing smoothing
- ▶ Kneser-Ney smoothing
- ▶ ...

*See Chapter 3 in Jurafsky & Martin (3 edition) for more details*

- ▶ Neural language models (later in the course)



## Handling unknown words

- ▶ Most tasks in NLP are open vocabulary tasks
- ▶ Test data will contain **out of vocabulary (OOV)** words
- ▶ Create an unknown word token <UNK>
- ▶ Train <UNK> probabilities
  - ▶ Create a fixed lexicon  $L$  of size  $V$
  - ▶ in the corpus, replace all words not in  $L$  with <UNK>
  - ▶ train its probabilities like a normal word
  - ▶ use UNK probabilities for any OOV word

## Using n-grams to generate sequences

### *Some Shakespeare...*

2  
gram

–Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.  
–What means, sir. I confess she? then all sorts, he is trim, captain.

3  
gram

–Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.  
–This shall forbid it should be branded, if renown made it empty.

4  
gram

–King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;  
–It cannot be but so.

## Using n-grams to generate sequences

### *Wall Street Journal*

2  
gram

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

3  
gram

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

## Limitations of n-gram models

- ▶ In general this is an insufficient model of language
- ▶ because language has **long-distance dependencies**:

*The computer which I had just put into the machine room on the fifth floor is crashing.*

- ▶ But we can often get away with N-gram models

## Limitations of n-gram models

- ▶ In general this is an insufficient model of language
- ▶ because language has **long-distance dependencies**:

*The computer which I had just put into the machine room on the fifth floor is crashing.*

- ▶ But we can often get away with N-gram models

# Evaluation of language models

## 1. Intrinsic evaluation

- ▶ evaluate directly on a test set designed for the task at hand
- ▶ using some metric
- ▶ for LMs — perplexity

## 2. Extrinsic evaluation

- ▶ evaluate in the context of some external task
- ▶ e.g. speech recognition, machine translation

# Perplexity

**Intuition:** The best language model is one that best predicts an unseen test set (i.e. with the highest probability)

- ▶ **Perplexity** is the inverse probability of the test set, normalized by the number of words:

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

- ▶ For bigrams:

$$PP(W) = \sqrt[N]{\frac{1}{\prod_{k=1}^n P(w_k | w_{k-1})}}$$

- ▶ **Minimize** perplexity

## Lower perplexity = better model

- ▶ Wall Street Journal corpus
- ▶ Train on 38 million words
- ▶ test on 1.5 million words

<b>N-gram Order</b>	<b>Unigram</b>	<b>Bigram</b>	<b>Trigram</b>
Perplexity	962	170	109



## Problem with intrinsic evaluation of LMs

- ▶ depends on how different the test and training set are
- ▶ not comparable across datasets
- ▶ but useful for pilot experimentation

So extrinsic evaluation is better, but time-consuming

# Outline.

Probabilistic language modelling

Part-of-speech (POS) tagging

Q&A and Discussion

## Part of speech tagging

They can fish.

- ▶ They\_pronoun can\_modal fish\_verb.  
(‘can’ meaning ‘are able to’)
- ▶ They\_pronoun can\_verb fish\_plural-noun.  
(‘can’ meaning ‘put into cans’)

### Ambiguity

*can*: modal verb, verb, singular noun

*fish*: verb, singular noun, plural noun

## Tagset (CLAWS 5)

**tagset**: standardized codes for fine-grained parts of speech.  
CLAWS 5: over 60 tags, including:

NN1	singular noun	NN2	plural noun
PNP	personal pronoun	VM0	modal auxiliary verb
VVB	base form of verb	VVI	infinitive form of verb

- ▶ They\_PNP can\_VM0 fish\_VVI .\_PUN
- ▶ They\_PNP can\_VVB fish\_NN2 .\_PUN

## POS tagging: Why do we care?

- ▶ First step towards syntactic analysis (which in turn, is often useful for semantic analysis).
- ▶ Simpler models and often faster than full syntactic parsing, but sometimes enough to be useful
  - ▶ POS tags can be useful features in e.g. text classification, authorship identification, etc.
  - ▶ Useful for applications such as text to speech synthesis: “it is time to wind the clock up” versus “the wind was strong”

## Extent of POS Ambiguity

The Brown corpus (1,000,000 word tokens) has 39,440 different word types.

- ▶ 35340 have only 1 POS tag anywhere in corpus (89.6%)
- ▶ 4100 (10.4%) have 2 to 7 POS tags

So why does just 10.4% POS-tag ambiguity by word type lead to difficulty?

## Extent of POS Ambiguity

The Brown corpus (1,000,000 word tokens) has 39,440 different word types.

- ▶ 35340 have only 1 POS tag anywhere in corpus (89.6%)
- ▶ 4100 (10.4%) have 2 to 7 POS tags

So why does just 10.4% POS-tag ambiguity by word type lead to difficulty?

Many **high-frequency** words have more than one POS tag.  
In fact, around 50% of the word tokens are ambiguous.

## Word Frequencies in Different languages

Ambiguity by part-of-speech tags:

Language	Type-ambiguity	Token-ambiguity
English	13.2%	56.2%
Greek	<1%	19.14%
Japanese	7.6%	50.2%
Czech	<1%	14.5%
Turkish	2.5%	35.2%



## Some tagging strategies

- ▶ One simple strategy: just assign to each word its most common tag. (Call this Uni-gram tagging)
- ▶ Surprisingly, even this crude approach typically gives around 90% accuracy. (State-of-the-art (English) is 97 - 98%).
- ▶ Can we do better?

## Some tagging strategies

- ▶ One simple strategy: just assign to each word its most common tag. (Call this Uni-gram tagging)
- ▶ Surprisingly, even this crude approach typically gives around 90% accuracy. (State-of-the-art (English) is 97 - 98%).
- ▶ Can we do better?

## Part of speech tagging using Hidden Markov Models (HMM)

1. Start with untagged text.
2. Assign all possible tags to each word in the text on the basis of a lexicon that associates words and tags.
3. Find the most probable sequence (or n-best sequences) of tags, based on probabilities from the training data.
  - ▶ lexical probability: e.g., is *can* most likely to be VM0, VVB, VVI or NN1?
  - ▶ and tag sequence probabilities: e.g., is VM0 or NN1 more likely after PNP?

## Assigning probabilities

Estimate tag sequence:  $n$  tags with the maximum probability, given  $n$  words:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(t_1^n | w_1^n)$$

By Bayes theorem:

$$P(t_1^n | w_1^n) = \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)}$$

but  $P(w_1^n)$  is constant:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} P(w_1^n | t_1^n) P(t_1^n)$$

## Bigrams

Bigram assumption: probability of a tag depends on previous tag, hence product of bigrams:

$$P(t_1^n) \approx \prod_{i=1}^n P(t_i | t_{i-1})$$

Probability of word estimated on basis of its tag alone:

$$P(w_1^n | t_1^n) \approx \prod_{i=1}^n P(w_i | t_i)$$

Hence:

$$\hat{t}_1^n = \operatorname{argmax}_{t_1^n} \prod_{i=1}^n P(w_i | t_i) P(t_i | t_{i-1})$$

## Example

Tagging: *they fish* (ignoring punctuation)

Assume PNP is the only tag for *they*, and that *fish* could be NN2 or VVB.

Then the estimate for PNP NN2 will be:

$$P(\text{they}|\text{PNP}) P(\text{NN2}|\text{PNP}) P(\text{fish}|\text{NN2})$$

and for PNP VVB:

$$P(\text{they}|\text{PNP}) P(\text{VVB}|\text{PNP}) P(\text{fish}|\text{VVB})$$

## Training the POS tagger

They\_PNP used\_VVD to\_T00 can\_VVI fish\_NN2 in\_PRP  
those\_DT0 towns\_NN2 .\_PUN But\_CJC now\_AV0 few\_DT0  
people\_NN2 fish\_VVB in\_PRP these\_DT0 areas\_NN2  
.\_PUN

sequence	count	bigram probability
NN2	4	
NN2 PRP	1	0.25
NN2 PUN	2	0.5
NN2 VVB	1	0.25

Also lexicon: fish NN2 VVB

## Training the POS tagger

They\_PNP used\_VVD to\_T00 can\_VVI fish\_NN2 in\_PRP  
those\_DT0 towns\_NN2 .\_PUN But\_CJC now\_AV0 few\_DT0  
people\_NN2 fish\_VVB in\_PRP these\_DT0 areas\_NN2  
.\_PUN

sequence	count	bigram probability
NN2	4	
NN2 PRP	1	0.25
NN2 PUN	2	0.5
NN2 VVB	1	0.25

Also lexicon: fish NN2 VVB



## Training the POS tagger

They\_PNP used\_VVD to\_T00 can\_VVI fish\_NN2 in\_PRP  
those\_DT0 towns\_NN2 .\_PUN But\_CJC now\_AV0 few\_DT0  
people\_NN2 fish\_VVB in\_PRP these\_DT0 areas\_NN2  
.\_PUN

sequence	count	bigram probability
NN2	4	
NN2 PRP	1	0.25
NN2 PUN	2	0.5
NN2 VVB	1	0.25

Also lexicon: fish NN2 VVB

## Applying in practice

- ▶ Maximise the overall tag sequence probability
- ▶ Actual systems use trigrams — smoothing and backoff are critical.
- ▶ Unseen words: these are not in the lexicon, so use all possible **open class** tags, possibly restricted by morphology.

## Evaluation of POS tagging

- ▶ percentage of correct tags, i.e. **accuracy**
- ▶ one tag per word (some systems give multiple tags when uncertain)
- ▶ accuracy over 97% for English (but note punctuation is unambiguous)
- ▶ **baseline** of taking the most common tag gives 90% accuracy

## Other tagging or sequence labelling tasks

- ▶ **Named entity recognition**: e.g., label words as belonging to persons, organizations, locations, or none of the above:

*Barack/PER Obama/PER spoke/NON from/NON  
the/NON White/LOC House/LOC today/NON ./NON*

- ▶ **Information field segmentation**: Given specific type of text (e.g. classified advert), identify which words belong to which fields (e.g. price/ size/ location)

*3BR/SIZE flat/TYPE in/NON Bruntsfield/LOC ./NON  
near/LOC main/LOC roads/LOC ./NON Bright/FEAT  
./NON well/FEAT maintained/FEAT ...*

Correct tags depend on the sequence of words.

# Outline.

Probabilistic language modelling

Part-of-speech (POS) tagging

Q&A and Discussion

## Question 1: Ambiguity

What kinds of ambiguity manifest themselves in the following newspaper headlines?

Iraqi head seeks arms

Stolen painting found by tree

Teacher strikes idle kids

## Question 2: Language modelling – perplexity

- ▶ Wall Street Journal corpus
- ▶ Train on 38 million words
- ▶ test on 1.5 million words

<b>N-gram Order</b>	<b>Unigram</b>	<b>Bigram</b>	<b>Trigram</b>
Perplexity	962	170	109

## Question 2: Language modelling – perplexity

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

The table shows a clear inverse correlation between  $n$ -gram order and perplexity.

- ▶ Why does perplexity decrease as  $n$  increases?
- ▶ Do you expect this decrease in perplexity to continue with higher values of  $n$ ? Why?



## Question(s) 3: Discussion

1. What sources of information can you use to PoS-tag unknown words?
2. How are n-gram language modelling and PoS tagging methods affected by the differences in genre? Which of the two tasks is more sensitive to this difference and why?
3. For the challenges and applications listed in lecture 1, which problems can n-gram language modelling and PoS tagging solve?

## Question 4: PoS tagging

NN1	singular noun	NN2	plural noun
PNP	personal pronoun	VM0	modal auxiliary verb
VVB	base form of verb	VVI	infinitive form of verb

Two possible analyses of *They can fish*.

- ▶ They\_PNP can\_VM0 fish\_VVI .\_PUN
- ▶ They\_PNP can\_VVB fish\_NN2 .\_PUN

List the probabilities involved in each of the analyses. Which of the analysis will be most likely chosen, and why? (Assuming the model was trained on a balanced corpus.)

## Question 5: Examples of PoS tagging errors

*Turkey will keep for several days in a fridge.*

Turkey\_**NP0** will\_VM0 keep\_VVI for\_PRP several\_DT0  
days\_NN2 in\_PRP a\_AT0 fridge\_NN1

*We have hope that the next year will be peaceful.*

We\_PNP have\_VHB hope\_**VVB** that\_CJT the\_AT0 next\_ORD  
year\_NN1 will\_VM0 be\_VBI peaceful\_AJ0

Why does the model make these errors?

## Question 5: Examples of PoS tagging errors

*Turkey will keep for several days in a fridge.*

Turkey\_**NP0** will\_VM0 keep\_VVI for\_PRP several\_DT0  
days\_NN2 in\_PRP a\_AT0 fridge\_NN1

*We have hope that the next year will be peaceful.*

We\_PNP have\_VHB hope\_**VVB** that\_CJT the\_AT0 next\_ORD  
year\_NN1 will\_VM0 be\_VBI peaceful\_AJ0

Why does the model make these errors?

# Acknowledgement

*Some slides were adapted from Ann Copestake, Dan Jurafsky  
and Tejaswini Deoskar*