Foundations of Natural Language Processing

Lecture 11: Language Models

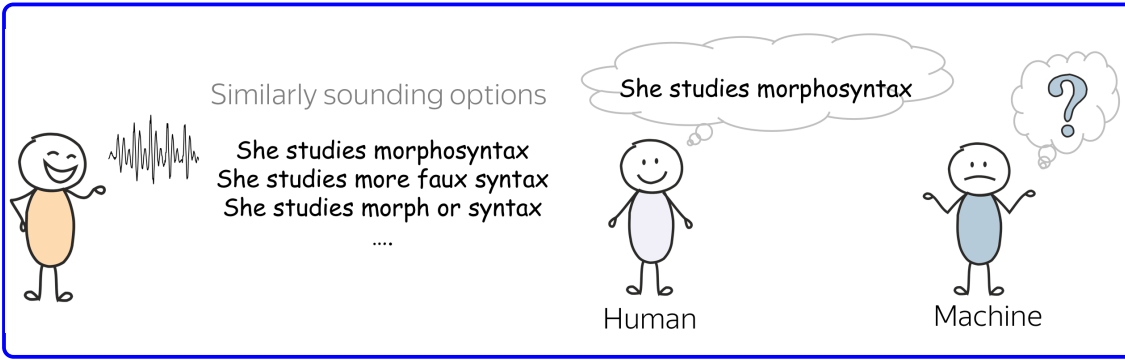
Mirella Lapata
School of Informatics
University of Edinburgh
mlap@inf.ed.ac.uk

Slides based on content from: Philipp Koehn, Alex Lascarides, Sharon Goldwater, Shay Cohen, Khalil Sima'an, Ivan Titov

Recap

- **Last time:** we talked about neural embeddings and how they can be learned from large collections of unannotated texts ('self-supervision')
- Skip-gram model predicts surrounding words (context) given a target word.
- What is the probablity that context word will be around (before or after) target word?
- **Today:** we consider **language models** which compute the probability of linguistic sequences (e.g, sentences).

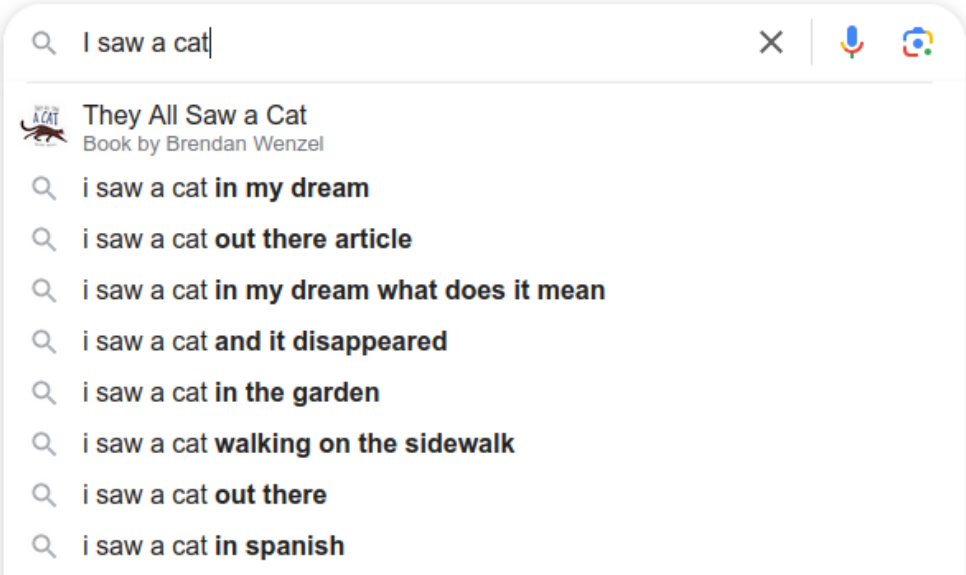
Humans are excellent language models!



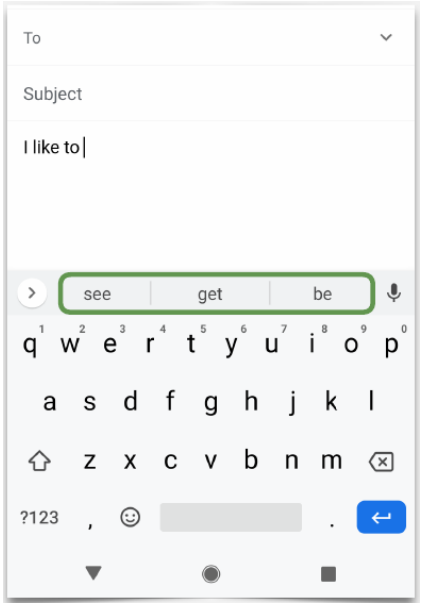
Which one of these is more likely?

- P(the cat slept peacefully) vs P(slept the peacefully cat)
- P(the cat slept peacefully) vs P(the cat slept furiously)
- P(the cat slept furiously) vs P(the cat slept analogously)

Even before GPT language models were popular



Even before GPT language models were popular

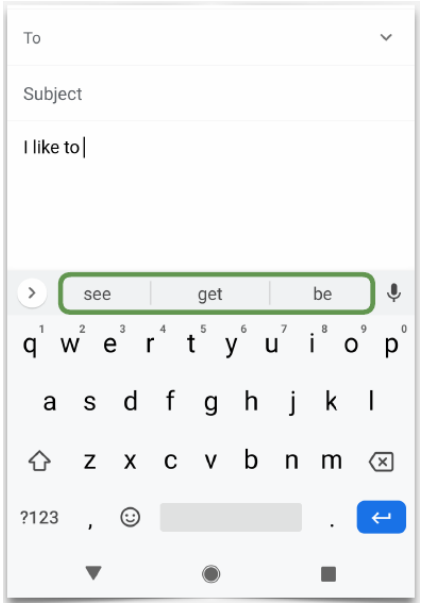


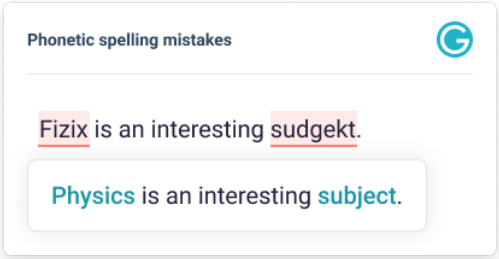
Phonetic spelling mistakes

Fizix is an interesting sudgekt.

Physics is an interesting subject.

Even before GPT language models were popular





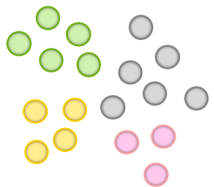
Felix is an interesting **subject**	0.03
Felix is an interesting **dialect**	0.005
Physics is an interesting **dialect**	0.0034
Physics is an interesting **subject**	0.01

How do we estimate the probability of a sentence?

- Our goal is to estimate probabilities of text fragments.
- For simplicity, let's assume we deal with **sentences**. We want these probabilities to reflect knowledge of a language.
- Specifically, we want sentences that are **more likely** to appear in a language to have a **larger probability** according to our language model.
- **How likely is a sentence to appear in a language**?

How do we estimate the probability of a sentence?

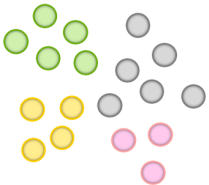
What is the probability to pick a green ball?



$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

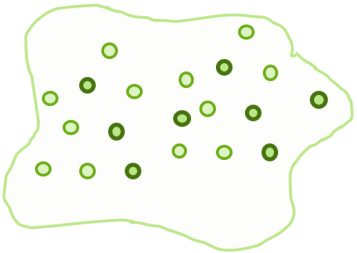
How do we estimate the probability of a sentence?

What is the probability to pick a green ball?



$$\frac{5}{5+6+4+3} = \frac{5}{18}$$

Can we do the same for sentences?



Text corpus

$$P(\textit{the Archaeopteryx soared jaggedly amidst foliage}) = \frac{0}{|corpus|} = 0$$

$$P(\textit{jaggedly trees the on flew}) = \frac{0}{|corpus|} = 0$$

Sentence Probability: Decompose into smaller parts

$$P(\text{I saw a cat on} \dots) =$$
$$P(\text{I}) \cdot P(\text{saw}|\text{I}) \cdot P(\text{a}|\text{I saw}) \cdot P(\text{cat}|\text{I saw a}) \cdot P(\text{on} \mid \text{saw a cat})$$

See animation here.

Sentence Probability: Decompose into smaller parts

$$P(\text{I saw a cat on} \dots) =$$
$$P(\text{I}) \cdot P(\text{saw}|\text{I}) \cdot P(\text{a}|\text{I saw}) \cdot P(\text{cat}|\text{I saw a}) \cdot P(\text{on} \mid \text{saw a cat})$$

See animation here.

$$\begin{aligned} P(y_1, y_2, \dots, y_n) &= P(y_1) \cdot P(y_2|y_1) \cdot P(y_3|y_1, y_2) \cdot \dots \cdot P(y_n|y_1, \dots, y_{n-1}) \\ &= \prod_{t=1}^{n} P(y_t|y_{<t}). \end{aligned}$$

Left-to-right Language Models

- Decompose the probability of text into **conditional probabilities** of each token given the previous context.
- But how do we compute $P(y_t|y_1, y_2, \ldots, y_{t-1})$?
- **N-gram Language Models**
- **Neural Language Models**
- There are other language models: Masked Language Models or models that decompose the joint probability differently (e.g., arbitrary order of tokens and not fixed as the left-to-right order).

N-gram Language Models

Estimating Conditional Probabilities

We estimate $P(y_t|y_1, y_2, \ldots, y_{t-1})$ by counting global statistics from a corpus.

$$P(y_t|y_1, \ldots, y_{t-1}) = \frac{N(y_1, \ldots, y_{t-1}, y_t)}{N(y_1, \ldots, y_{t-1})},$$

where $N(y_1, \ldots, y_k)$ is the number of times tokens $(y_1, \ldots, y_k)$ occur in the text.

Markov Assumption: the probability of a word only depends on a fixed number of previous words.

$$P(y_t|y_1, \ldots, y_{t-1}) = P(y_t|y_{t-n+1}, \ldots, y_{t-1}).$$

For example,

$$\begin{array}{lll} n = 3 & P(y_t|y_1, \ldots, y_{t-1}) & = P(y_t|y_{t-2}, y_{t-1}) \\ n = 2 & P(y_t|y_1, \ldots, y_{t-1}) & = P(y_t|y_{t-1}) \\ n = 1 & P(y_t|y_1, \ldots, y_{t-1}) & = P(y_t) \end{array}$$

Example: Trigram Model

Before	After (3-gram)
$P(\textbf{I saw a cat on a mat}) =$	$P(\textbf{I saw a cat on a mat}) =$
$\cdot P(\text{I})$	$\cdot P(\text{I})$
$\cdot P(\text{saw}\|\text{I})$	$\cdot P(\text{saw}\|\text{I})$
$\cdot P(\text{a}\|\text{I saw})$	$\cdot P(\text{a}\|\text{I saw})$
$\cdot P(\text{cat}\|\text{I saw a})$	$\cdot P(\text{cat}\|\ \textbf{I}\ \text{saw a})$
$\cdot P(\text{on}\|\text{I saw a cat})$	$\cdot P(\text{on}\|\ \textbf{I saw}\ \text{a cat})$
$\cdot P(\text{a}\|\text{I saw a cat on})$	$\cdot P(\text{a}\|\ \textbf{I saw a}\ \text{cat on})$
$\cdot P(\text{mat}\|\text{I saw a cat on a})$	$\cdot P(\text{mat}\|\ \textbf{I saw a cat}\ \text{on a})$

Example: Trigram Model

Before	After (3-gram)
$P(\textbf{I saw a cat on a mat}) =$	$P(\textbf{I saw a cat on a mat}) =$
$\cdot P(\text{I})$	$\cdot P(\text{I})$
$\cdot P(\text{saw}\|\text{I})$	$\cdot P(\text{saw}\|\text{I})$
$\cdot P(\text{a}\|\text{I saw})$	$\cdot P(\text{a}\|\text{I saw})$
$\cdot P(\text{cat}\|\text{I saw a})$	$\cdot P(\text{cat}\|\text{saw a})$
$\cdot P(\text{on}\|\text{I saw a cat})$	$\cdot P(\text{on}\|\text{a cat})$
$\cdot P(\text{a}\|\text{I saw a cat on})$	$\cdot P(\text{a}\|\text{cat on})$
$\cdot P(\text{mat}\|\text{I saw a cat on a})$	$\cdot P(\text{mat}\|\text{on a})$

But what hapens if we haven't seen the counts?

Let's imagine we deal with a 4-gram language model and consider the following example:

$$P(\text{mat}|\text{I saw a cat on a }) = P(\text{mat}|\text{cat on a}) = \frac{N(\text{cat on a mat})}{N(\text{cat on a})}$$

- What if either denominator or numerator is zero? Both these cases are not really good for the model.
- To avoid these problems is common to use **smoothing**.
- Smoothing redistributes probability mass: it "steals" some mass from seen events and give to the unseen ones.
- Lots of different methods, based on different kinds of assumptions.

Smoothing

Backoff Smoothing: use less context for context we don't know much about

- if you can, use trigram;
- if not, use bigram;
- if not, use unigram

$$P(\text{mat}|\text{cat on a}) = \frac{N(\text{cat on a mat})}{N(\text{on a mat})} \text{ is zero!}$$

$$P(\text{mat}|\text{cat on a}) \approx P(\text{mat}|\text{on a})$$
$$P(\text{mat}|\text{on a}) \approx P(\text{mat}|\text{a})$$
$$P(\text{mat}|\text{on a}) \approx P(\text{mat})$$

Linear Interpolation: mix all probabilities, unigram, bigram, trigram.

- introduce scalar positive weights, $\lambda_0, \lambda_1, \ldots, \lambda_{n-1}$ such that $\sum_i \lambda_{i=1}$
- tune coefficients λ_i on development.

$$P(\text{mat}|\text{cat on a}) = \frac{N(\text{cat on a mat})}{N(\text{on a mat})} \text{ is zero!}$$

$$\begin{aligned}\hat{P}(\text{mat}|\text{cat on a}) \approx\ & \lambda_3 P(\text{mat}|\text{cat on a}) + \\ & \lambda_2 P(\text{mat}|\text{on a}) + \\ & \lambda_1 P(\text{mat}|\text{a}) \\ & \lambda_0 P(\text{mat})\end{aligned}$$

Smoothing

Laplace Smoothing: just pretend we saw all n-grams at least one time

- just add 1 to all counts!
- instead of 1, you can add a small δ

$$P(\text{mat}|\text{cat on a}) = \frac{N(\text{cat on a mat})}{N(\text{on a mat})}$$ is zero!

$$\hat{P}(\text{mat}|\text{cat on a}) = \frac{\delta \cdot N(\text{cat on a mat})}{\delta \cdot |V| + N(\text{cat on a})}$$

where V is the vocabulary size

Most most popular smoothing for n-gram LMs is **Kneser-Ney smoothing**, a more clever variant of back-off smoothing. More details are here.

How do we know our Language Model is any good?

Perplexity (PP): of a language model on a test set is the inverse probability of the test set, normalized by the number tokens N in the test set.

- Measure of how suprised the language model is when it sees words on the test set.
- Depends on what language model we are using!

$$PP(W) = P(w_1, w_2, \ldots, w_N)^{-\frac{1}{N}}$$

$$\sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N)}}$$

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

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i)}} \quad \text{unigram}$$

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

Relationship between Perplexity and Cross-Entropy

Perplexity ($PP(W)$) and cross-entropy ($H(W)$) are closely related. In fact, **perplexity is simply the exponentiation of the cross-entropy**: $PP(W) = \exp(H(W))$.

$$
\begin{aligned}
H(W) &= -\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_1,...,w_{i-1}) \\
&= \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_1,...,w_{i-1})\right) \\
&= P(W)^{-\frac{1}{N}}
\end{aligned}
$$

- Perplexity is exponentiation of average negative log probability per token.
- **Cross-entropy** measures **average uncertainty** in bits (if using log base 2) or nats (if using log base e).
- **Perplexity** represents the **effective branching factor**, meaning the average number of choices the model considers at each step.

Example Calculation

Suppose a model assigns the following conditional probabilities to a test sequence of three tokens:

$$P(w_1) = 0.2, \quad P(w_2|w_1) = 0.5, \quad P(w_3|w_1, w_2) = 0.1$$

Cross-entropy:

$$H(W) = -\frac{1}{3}(\log 0.2 + \log 0.5 + \log 0.1)$$

Approximating in base e:

$$H(W) \approx -\frac{1}{3}(-1.61 - 0.69 - 2.30) = 1.53$$

Perplexity:

$$PP(W) = e^{1.53} \approx 4.64$$

Thus, the model effectively has an average of **4.64 choices per token**.

Text Generation

See animation here.

- Once we have a language model, we can use it to generate text!
- We generate **one token at a time**
- Predict probability distribution of **next token given previous context**
- **Sample** from this distribution.

Text Generation with 3-gram model

See animation here.

- Generation procedure is the same as in general case
- The only difference is how probabilities are computed
- Predict probability distribution of **next token given previous two tokens**

Let's look at some output

The output of 3-gram language model trained on 2.5 million English sentences.

```
when this option may be the worst day of amnesty
international delegations visited israel, and felt that his
sisters, that they are reserved for zyryanovsk concetrating
factory there is a member of the shire ," given as to damage
the expansion of a meeting over a large health maintenance
organization, smoking , airconditioning , designated smoking
area .
```

GPT-4o thinks the text lacks **coherence**, and has a **mix of unrelated ideas**. There are issues with grammar and syntax, unrelated topics, inconsistent use of punctuation, ambiguity and vagueness.

Let's look at some output

The output of 3-gram language model trained on 2.5 million English sentences.

```
when this option may be the worst day of amnesty
international delegations visited israel, and felt that his
sisters, that they are reserved for zyryanovsk concetrating
factory there is a member of the shire ," given as to damage
the expansion of a meeting over a large health maintenance
organization, smoking , airconditioning , designated smoking
area .
```

GPT-4o thinks the text lacks **coherence**, and has a **mix of unrelated ideas**. There are issues with grammar and syntax, unrelated topics, inconsistent use of punctuation, ambiguity and vagueness.

Let's look at some output

The output of 3-gram language model trained on 2.5 million English sentences.

```
when this option may be the worst day of amnesty
international delegations visited israel, and felt that his
sisters, that they are reserved for zyryanovsk concetrating
factory there is a member of the shire ," given as to damage
the expansion of a meeting over a large health maintenance
organization, smoking , airconditioning , designated smoking
area .
```

GPT-4o thinks the text lacks **coherence**, and has a **mix of unrelated ideas**. There are issues with grammar and syntax, unrelated topics, inconsistent use of punctuation, ambiguity and vagueness.

Let's look at some output

The output of 3-gram language model trained on 2.5 million English sentences.

```
when this option may be the worst day of amnesty
international delegations visited israel, and felt that his
sisters, that they are reserved for zyryanovsk concetrating
factory there is a member of the shire ," given as to damage
the expansion of a meeting over a large health maintenance
organization, smoking , airconditioning , designated smoking
area .
```

GPT-4o thinks the text lacks **coherence**, and has a **mix of unrelated ideas**. There are issues with grammar and syntax, unrelated topics, inconsistent use of punctuation, ambiguity and vagueness.

Let's look at some output

Output corrected by GPT-4o

```
Amnesty International delegations recently visited Israel
to assess human rights conditions.  During their visit, they
observed conditions at the Zyryanovsk Concentrating Factory,
where workers expressed concerns about limited access to
health services and designated smoking areas.  Additionally,
there was a discussion about the expansion of a local health
maintenance organization, which faced resistance due to its
potential impact on public health policies and regulations
around smoking and air conditioning standards.
```

By the end of this course, you will know how to build your own GPT!

Summary

- What is language modeling?
- How do we estimate conditional probabilities?
- N-gram Language Models (smoothing) and their evaluation
- Generation with language models
- **Next time:** Neural language models