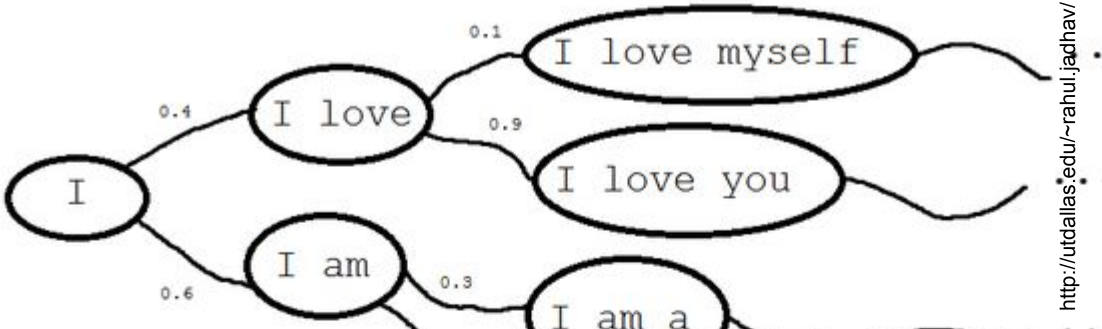
Language modeling-I

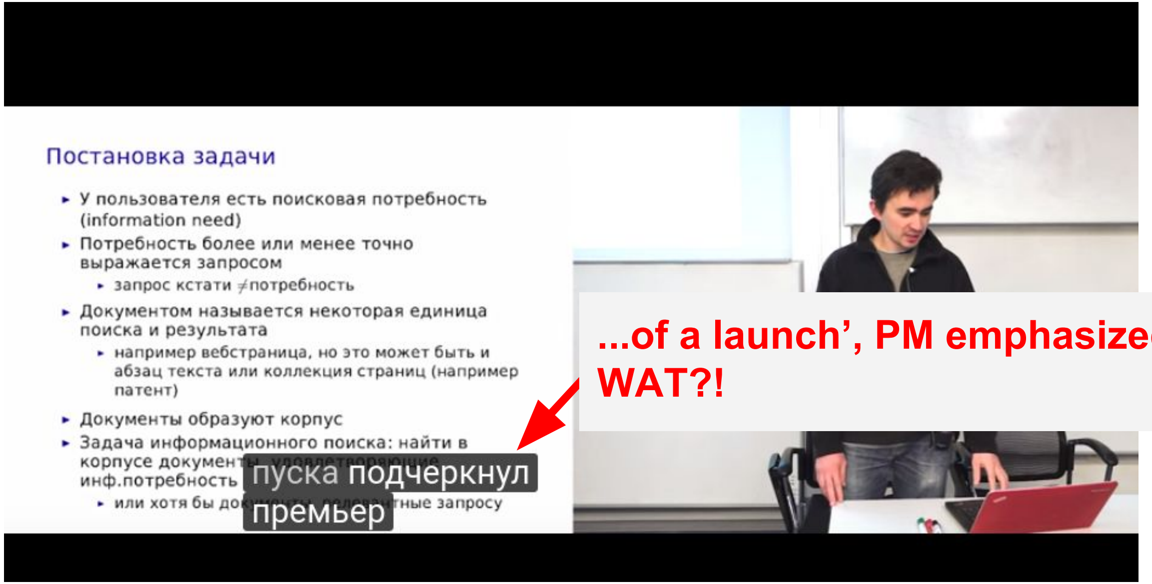
Lectures: Anton Alekseev, Steklov Mathematical Institute in St Petersburg
NRU ITMO, St Petersburg, 2018



http://utdallas.edu/~rahul.jadhav/

Motivation

In many tasks one has to estimate whether the text is ‘natural’ or ‘comprehensible’. Sometimes a clever way to estimate the word sequence probability is enough



...of a launch’, PM emphasized WAT?!

I must admit it was too hard to find a good example of lousy generated English subtitles

Actually Dmitriy said:

*…**поиск по патентам, например.***

...patent search, for example

[***https://youtu.be/APcwsxUpGrQ?t=1m38s***](https://youtu.be/APcwsxUpGrQ?t=1m38s)

Motivation

- **Speech recognition / machine translation / spelling correction / augmentative communication**
  e.g.: having generated several possible decodings of the phrase, one has to choose ‘the most probable’ (from the language’s point of view)

- **Information retrieval**
  ranking: for every document **d** we build ‘its language model’ and sort all documents by **P(q|d)** (where **q** is a query)

- **Fun!** Text generators, imitating the provided text collection’s style

Plan

1. Intuition
2. N-gram modeling
3. Language models quality evaluation
4. Zeros and smoothing
    a. Kneser-Ney smoothing

- Libraries
- Datasets

Intuition

- **Language model** allows us to estimate the probability of any sequence of words (alternative formulation: to estimate the probability of the next word)

- How to estimate the probability of ‘*Everything was in confusion in the Oblonskys' house*…’?

- Let us turn to conditional probability

Intuition: total recall

- Conditional probability

$$P(Y|X) = \frac{P(X,Y)}{P(X)} \Rightarrow P(X,Y) = P(Y|X)P(X)$$

- Chain rule for greater number of variables:

$$P(x_1 x_2 ... x_n) = P(x_n | x_1 ... x_{n-1}) ... p(x_2 | x_1) p(x_1)$$

- So can we compute it all easily?

$$P(x_i | x_1 ... x_{i-1}) = \frac{Count(x_1 ... x_{i-1} x_i)}{Count(x_1 ... x_{i-1})}$$

* Here and further Count(...) is the same as C(...) и c(...)

Intuition: total recall

- Conditional probability

$$P(Y|X) = \frac{P(X,Y)}{P(X)} \Rightarrow P(X,Y) = P(Y|X)P(X)$$

- Chain rule for greater number of variables:

$$P(x_1 x_2 ... x_n) = P(x_n|x_1 ... x_{n-1}) ... p(x_2|x_1)p(x_1)$$

- So can we compute it all easily?

$$P(x_i|x_1 ... x_{i-1}) = \frac{Count(x_1 ... x_{i-1} x_i)}{Count(x_1 ... x_{i-1})}$$

$$P(happy\ families\ are\ all) = P(all \mid happy\ families\ are) \times$$

$$\times P(are|happy\ families) \times P(families|happy) \times P(happy)$$

Intuition: total recall

- Conditional probability

$$P(Y|X) = \frac{P(X,Y)}{P(X)} \Rightarrow P(X,Y) = P(Y|X)P(X)$$

- Chain rule for greater number of variables:

$$P(x_1 x_2 ... x_n) = P(x_n|x_1 ... x_{n-1}) ... p(x_2|x_1)p(x_1)$$

- So can we compute it all easily?

$$P(x_i|x_1 ... x_{i-1}) = \frac{Count(x_1 ... x_{i-1} x_i)}{Count(x_1 ... x_{i-1})}$$

(nope! long chains are rare events!)

What do we do?

- Assumption is here to help: text satisfies the Markov property

$$P(x_i|x_1 \ldots x_{i-1}) = P(x_i|x_i - K \ldots x_{i-1})$$

  ...which means that current event depends on not more than on K preceding ones
- Examples:
  - $K = 0$ (unigram model)

$$P(\textit{happy families are all}) =$$

$$P(\textit{all}) \times P(\textit{are}) \times P(\textit{families}) \times P(\textit{happy})$$

  - $K = 1$ (bigram model)

$$P(\textit{happy families are all}) = P(\textit{all} \mid \textit{are}) \times$$

$$\times P(\textit{are} \mid \textit{families}) \times P(\textit{families} \mid \textit{happy}) \times P(\textit{happy})$$

Plan

1. ~~Intuition~~
2. N-gram modeling
3. Language models quality evaluation
4. Zeros and smoothing
    a. Kneser-Ney smoothing

- Libraries
- Datasets

N-gram model

- Model:

$$P(x_1, ...x_n) = \prod_{i=1}^{n} P(x_i | x_{i-N+1} ... x_{i-1})$$

  one has to add $N-1$ terms «begin» ^and «end» $ from both sides (padding)
- We can estimate the probability like that

$$P(x_i | x_{i-N+1} ... x_{i-1}) = \frac{Count(x_{i-N+1} ... x_{i-1} x_i)}{Count(x_{i-N+1} ... x_{i-1})}$$

- $$P(x_i | x_{i-1}) = Count(x_i, x_{i-1}) Count(x_{i-1})$$
- E.g. for bigrams:

$$P(hello, i, love, you) =$$

$$= P(hello | \hat{}\,) P(i | hello) P(love | i) P(you | love) P(\$ | you)$$

Plan

1. ~~Intuition~~
2. ~~N-gram modeling~~
3. Language models quality evaluation
4. Zeros and smoothing
    a. Kneser-Ney smoothing

- Libraries
- Datasets

Quality evaluation techniques

- **Extrinsic**
  Checking quality by inducing the model into a bigger useful task
  (machine translation, spelling correction, ...).
  If the target metric (where the money is: translators work time, editor’s time, clicks count, earned money, etc.) goes up, **the model has become better**

- **Intrinsic**
  ~~Evaluation for the poor~~ we need estimates when extrinsic evaluation is too expensive or when one doesn’t want the results to be related to some specific application (if the model is universal to certain extent); also a metric that shows us how ‘good’ the model is

Quality evaluation techniques

- **Extrinsic**
  Checking quality by inducing the model into a bigger useful task (machine translation, spelling correction, ...).
  If the target metric (where the money is: translators work time, editor's time, clicks count, earned money, etc.) goes up, **the model has become better**

Not this time (totally different story)

- **Intrinsic**
  ~~Evaluation for the poor~~ when we need estimates when extrinsic evaluation is too expensive or when one doesn't want the results to be related to some specific application (if the model is universal to certain extent); also a metric that shows us how 'good' the model is

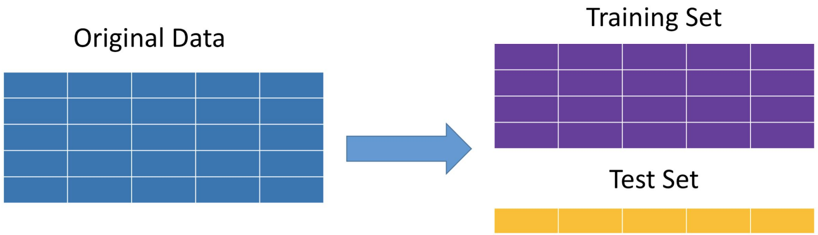
Quality evaluation

We have the data, we have the metric

We split the data into

- train set (for tuning models) and
- test set (for trained models evaluation)

We have to believe that train and test set data samples are from “the same distribution” (otherwise we won’t be able to train anything useful)



https://jessesw.com/images/Rec_images/Traintest_ex.png

Quality evaluation

Deadly Sin №1
Test data leaks into train set
(this way we lose generalization capability and estimates validity)

Deadly Sin №2
Tuning hyperparameters on test set

But how do we tune the parameters? Ideas?

DataFest sticker

Quality evaluation: data splitting

TRAIN	DEV	TEST

1. TRAIN - training model
2. DEV - evaluating quality + analyzing errors + tuning hyperparameters
3. TEST - blind quality evaluation: looking at quality metric ONLY + not too often, so as not to overfit

Model quality evaluation

- The larger the probability of the test text, the closer the model is to life
- Perplexity — inverse probability of the text normalized by words sequence length

$$PP(W) = P(x_1 ... x_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(x_1 ... x_N)}} =$$

$$= \sqrt[N]{\frac{1}{\prod_{i=1}^{N} P(x_i | x_1 ... x_{i-1})}}$$

It is evident that less is better.

- To those who know some information theory, the formula may seem familiar:

$$PP(W) = P(x_1 ... x_N)^{-\frac{1}{N}} = e^{-\frac{1}{N}\sum_{i=1}^{N} \log P(x_i | x_1 ... x_{i-1})}$$

Quality evaluation: example

Training on 38M tokens
Testing on 1.5M
Dataset: Wall Street Journal

	1-gram	**2-gram**	**3-gram**
Perplexity	962	170	109

from Martin/Jurafsky

To be continued...

Language modeling-I

Lectures: Anton Alekseev, Steklov Mathematical Institute in St Petersburg
NRU ITMO, St Petersburg, 2018