Natural Language Processing Workshop

Amy Hemmeter, MSA Class of ‘18
Artificial Intelligence Engineer, Interactions Digital Roots

“We are seeing a surge in demand for NLP applications across sectors and use cases. Our clients are implementing NLP solutions for deeper customer insights, risk mitigation, and operational efficiency.”

- Nathan Peifer, formerly of EY, now at Elicit, personal communication, 2020

“We’re looking for better skills for dealing with unstructured data, NLP is becoming increasingly important as much of the data available is text.”

- Garima Sharma, Manager, Enterprise Data Intelligence at Domino’s (Women in Data Science Conference 2020, Detroit)

Housekeeping

- Assignment/quizzes after each class on Moodle will be how your grade will be determined
- “Office hours” will mostly be done by email but you can also email me to set up an appointment at amy.hemmeter@gmail.com

Natural Language Processing

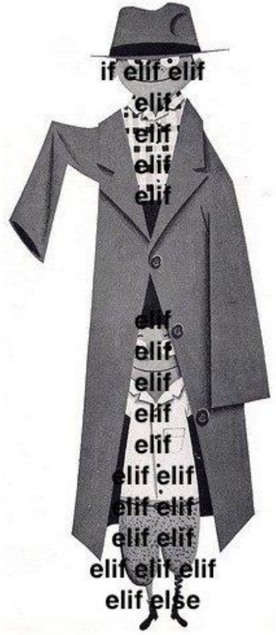
- NLP is a subfield of artificial intelligence that deals with understanding and in some cases producing, human ("natural") language
- We're going to cover the following NLP topics:
  - Language Modeling
  - Classification
  - Tagging
  - Practical tips for text data exploration

NLP in the pre-Machine Learning Era

Rule-Based NLP

- A series of rules in code that very explicitly spelled out how a computer is supposed to understand human language
- Often based on keywords and collocations (words that occur together)
- Regular expressions play a big role
- Very time-consuming and expensive for the company
- Fairly old-fashioned, no one's first choice and doesn't show up on job postings that request NLP skills very often
- However, many systems are still partially rule-based -- including those of many MSA employers (from my experience of talking to them in 2018)

Expert AI Revealed To Be 1,000,000 If-Else Statements Stacked in a Trenchcoat



What’s missing from this picture?

What’s missing from this picture?

Math!!

“I thought the movie was terribly well done, bravo to all the actors.”

“Garbage”.

1. Words are not numbers

1. Words are not numbers
2. Input can be different lengths

1. **Words are not numbers**
2. Input can be different lengths

Traditional Dataset

Number of Bedrooms	Number of Bathrooms	Age in Years	Walkability
2	1	10	87.2
5	3	2	50.2
3	1.5	25	95.2

Output is a number

$$\beta 0 + X_1\beta_1 + X_2\beta_2 + X_3\beta_3 + X_4\beta_4 = Y$$

Inputs are numbers

NLP Example - Sentiment Analysis

Output - Valence

NLP Example - Sentiment Analysis

Output - Valence

...

NLP Example - Sentiment Analysis

Output - Valence

…

A NUMBER!

NLP Example - Sentiment Analysis

Output - Valence

…

A NUMBER!

What’s our input?

Early ML Solutions to NLP

Early ML Solutions for NLP

- **Simple n-gram models for language modeling**
- Naive Bayes, Logistic Regression for classification
- Various flavors of Markov Models for tagging
- Now, however, almost all solutions are based on deep learning

What is language modeling?

- The task of predicting the probability of a sentence

“I’ll text you when I get ______”

Other Uses for Language Modeling

- Speech recognition
  - We want to know that the sounds that are very similar in “wreck a nice beach” are more likely “recognize speech” -- we want to get a probability of the sentence
- Machine Translation
  - Change idiomatic “large winds tonight” from another language to more naturally English “High winds tonight”
- Spelling correction
  - “The office is about 15 minuets from my house” → “the office is about 15 minutes from my house”
- Anytime you need to know the probability of a sentence!

How to do n-gram language modeling

We’re trying to find $P(w|h)$

How to do n-gram language modeling

We're trying to find $P(w|h)$

$$P(the|its\ water\ is\ so\ transparent\ that)$$

How to do n-gram language modeling

We're trying to find $P(w|h)$

$$P(the|its\ water\ is\ so\ transparent\ that)$$

$$= \frac{C(its\ water\ is\ so\ transparent\ that\ the)}{C(its\ water\ is\ so\ transparent\ that)}$$

How to do n-gram language modeling

$$P(B|A) = \frac{P(A,B)}{P(A)}$$

therefore

$$P(A, B) = P(A)P(B|A)$$

If you add more variables:

$$P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, B, C)$$

How to do n-gram language modeling

$$P(w_1, w_2, \ldots w_{n-1}) \quad =$$

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned}$$

How to do n-gram language modeling

We can apply a simplifying assumption:

$$P(the|its\ water\ is\ so\ transparent\ that) \approx P(the|that)$$

Or perhaps:

$$P(the|its\ water\ is\ so\ transparent\ that) \approx P(the|transparent\ that)$$

Unigram Language Model

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i)$$

Unigram Language Model

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i)$$

fifth, an, of, futures, the, an, incorporated, a, a, the, inflation, most, dollars, quarter, in, is, mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the

Bigram Language Model

$$P(w_1, w_2, w_3, w_4) = P(w_1)P(w_2|w_1)P(w_3|w_2)P(w_4|w_3)$$

Bigram Language Model

$$P(w_1, w_2, w_3, w_4) = P(w_1)P(w_2|w_1)P(w_3|w_2)P(w_4|w_3)$$

texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached

this, would, be, a, record, november

How many Ns in your n-gram?

- N-gram models can also work for trigrams, 4-grams, and 5-grams
- Note that this is not a full model of language because language contains long-range dependencies

“The *people* who have worked the longest on this project *are* the most industrious.”

How to do n-gram language modeling

Maximum Likelihood Estimate

- Counts from a corpus (a large body of text that serves as training data) and normalizes the counts so that they end up between 0 and 1

How to do n-gram language modeling

Maximum Likelihood Estimate

- Counts from a corpus (a large body of text that serves as training data) and normalizes the counts so that they end up between 0 and 1

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

How to do n-gram language modeling

An example*:

$$P(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

<s> I am Sam </s>

<s> Sam I am </s>

<s> I do not like green eggs and ham </s>

$P(\texttt{I} \mid \texttt{<s>}) = \frac{2}{3} = .67$ $\quad P(\texttt{Sam} \mid \texttt{<s>}) = \frac{1}{3} = .33$ $\quad P(\texttt{am} \mid \texttt{I}) = \frac{2}{3} = .67$

$P(\texttt{</s>} \mid \texttt{Sam}) = \frac{1}{2} = 0.5$ $\quad P(\texttt{Sam} \mid \texttt{am}) = \frac{1}{2} = .5$ $\quad P(\texttt{do} \mid \texttt{I}) = \frac{1}{3} = .33$

*From Dan Jurafsky's lecture slides

How to do language modeling

- In order to avoid “floating point underflow”, in practice we do all of our calculations in log space rather than multiplying

$$log(p_1 p_2 p_3 p_4) = logp_1 + logp_2 + logp_3 + logp_4$$

How to evaluate your n-gram language model

- Perplexity

How to evaluate your n-gram language model

- Perplexity

How to evaluate your n-gram language model

- Perplexity

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

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

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

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

How to evaluate your n-gram language model

- Pretend you have a sentence that has random digits
- The perplexity of this sentence according to a model that assigns p=1/10 to each digit:

$$
\begin{aligned}
\mathrm{PP}(W) &= P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} \\
&= (\frac{1}{10}^N)^{-\frac{1}{N}} \\
&= \frac{1}{10}^{-1} \\
&= 10
\end{aligned}
$$

How to evaluate your n-gram language model

A lower perplexity = a better model

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

Time to look at some code

Let's see how an implementation of an n-gram language model could work.

On Moodle, go to the course page and do the following to follow along:

1. Click on the link that says "2020 files"
2. Move the train.txt, dev.txt, and test.txt into a new folder you create within that folder called "data"
3. Open NGram_Language_Model.ipynb with Jupyter Notebook