ANTONIS ANASTASOPOULOS
CS499 INTRODUCTION TO NLP

LANGUAGE MODELING

https://cs.gmu.edu/~antonis/course/cs499-spring21/

PROJECT IDEA

Due tomorrow

STRUCTURE OF THIS LECTURE

1. Why LM?
2. Smoothing
3. Evaluation

LANGUAGE MODELING

ARE THESE SENTENCES OK?

Jane went to the store.

store to Jane went the.

Jane went store.

Jane goed to the store.

The store went to Jane.

The food truck went to Jane.

CALCULATING THE PROBABILITY OF A SENTENCE

Jane went to the store .

$$P(X) = \prod_{i=1}^{n} P(x_i)$$

Unigram

How can we calculate this?

$$P(\text{Jane went to the store}) = P(\text{Jane}) \times P(\text{went}) \times P(\text{to}) \times P(\text{the}) \times P(\text{store}) \times P(.) \times P(\text{</s>})$$

$$P(the) = \frac{c(the)}{N}$$

bahia cocoa review
showers continued throughout the week in the bahia cocoa zone, alleviating the drought since early january and improving prospects for the coming temporao, although normal humidity levels have not been restored, comissaria smith said in its weekly review. the dry period means the temporao will be late this year. arrivals for the week ended february 22 were 155,221 bags of 60 kilos making a cumulative total for the season of 5.93 mln against 5.81 at the same stage last year.

A UNIGRAM DISTRIBUTION

[rank = 1]
p(the) = 0.038
p(of) = 0.026
p(and) = 0.021
p(to) = 0.017
p(is) = 0.013
p(a) = 0.012
p(in) = 0.012
p(for) = 0.09
…

…
[rank = 1000]
p(joint) = 0.00014
p(relatively) = 0.00014
p(plot) = 0.00014
p(DEL1SUBSEQ) = 0.00014
p(rule) = 0.00014
p(62.0) = 0.00014
p(9.1) = 0.00014
…

As a generator:

pressure & , dlrs export , ethanol prohibiting brussels two high defence corp mln . rumors cts , buying bushel of the tonnes . his the agreement very each the rains > . , later end the seen lehman leaders u.s. john bcs.l paid to offers or day 30.055 by 6,974,554 lending 2,650,000 for 's 57 billion february to , dealers added 1,247,000 product industry the pueblo the earlier pipeline . reported of and for commission 150 & to .

CALCULATING THE PROBABILITY OF A SENTENCE

Jane went to the store .

$$P(X) = \prod_{i=1}^{n} P(x_i)$$

Unigram

$$P(\text{Jane went to the store .}) = P(\text{Jane}) \times P(\text{went}) \times P(\text{to}) \times P(\text{the}) \times P(\text{store}) \times P(.) \times P(\text{</s>})$$

$$P(the) = \frac{c(the)}{N}$$

But word order and context matters!

?????

$$P(\text{Jane went to the store.}) = P(\text{went the store Jane to.})$$

CALCULATING THE PROBABILITY OF A SENTENCE

$$P(X) = \prod_{i=1}^{I} P(\underbrace{x_i}_{\text{Next Word}} \mid \underbrace{x_1, \ldots, x_{i-1}}_{\text{Context}})$$

BIGRAMS

$$P(X) = \prod_{i=1}^{I} P(\underbrace{x_i}_{\text{Next Word}} \mid \underbrace{x_1, \ldots, x_{i-1}}_{\text{Context}})$$

How can we calculate this?

$$P(\text{to} \mid \text{went}) = \frac{c(\text{went,to})}{c(\text{went})}$$

$$\begin{aligned} P(\text{Jane went to the store}) = {} & P(\text{Jane} \mid \text{<s>}) \times P(\text{went} \mid \text{Jane}) \times \\ & \boxed{P(\text{to} \mid \text{went})} \times P(\text{the} \mid \text{to}) \times \\ & P(\text{store} \mid \text{the}) \times P(\,.\mid \text{store}) \\ & P(\text{</s>} \mid .\,) \end{aligned}$$

General formula for n-gram model: $p(x_i \mid x_{i-n+1:i-1}) = \dfrac{c(x_{i-n+1:i})}{c(x_{i-n+1:i-1})}$

SMOOTHING

DEALING WITH UNSEEN N-GRAMS

[Option 1]: Limit the vocab

- pretend that the most rare types in the corpus are actually unknown
- (substitute rare words with <unk> in the corpus)
- compute probabilities as usual
- simple, and common in neural models

DEALING WITH UNSEEN N-GRAMS: ADDITIVE SMOOTHING

[Option 2]: **Laplace** or add-one smoothing

- Pretend we've seen every word type one more time that we actually have
- Add 1 to all counts (or add **ε** to all counts)
- you still need to know the whole vocabulary

$$p(w) = \frac{c(w)}{\boxed{|\Sigma|}} \qquad p(w) = \frac{c(w) + \delta}{|\Sigma| + \delta \boxed{|V|}} \qquad p(\text{UnKnOwNwOrD}) = \frac{0 + \delta}{|\Sigma| + \delta|V|} = \frac{1}{|\Sigma| + \delta|V|}$$

Σ = all tokens (without <s>) **V** = all types (with <unk>)

EXAMPLE

Corpus:

the cat sat on the mat .
a dog chased the cat .
a mouse ate some cheese .
the mouse ran under a mat .

$$
\begin{aligned}
p(\text{A cat chased the mouse .}) = {} & p(\text{<s>} \mid \text{a}) \times \\
& p(\text{cat} \mid \text{a}) \times \\
& p(\text{chased} \mid \text{cat}) \times \\
& p(\text{the} \mid \text{chased}) \times \\
& p(\text{mouse} \mid \text{the}) \times \\
& p(\text{.} \mid \text{mouse}) \\
& p(\text{</s>} \mid \text{.})
\end{aligned}
$$

EXAMPLE

Corpus:

the cat sat on the mat .
a dog chased the cat .
a mouse ate some cheese .
the mouse ran under a mat .

$$
\begin{aligned}
p(\text{A cat chased the bat .}) = \; & p(\text{<s>} \mid \text{a}) \times \\
& p(\text{cat} \mid \text{a}) \times \\
& p(\text{chased} \mid \text{cat}) \times \\
& p(\text{the} \mid \text{chased}) \times \\
& p(\text{bat} \mid \text{the}) \times \\
& p(\text{.} \mid \text{bat}) \\
& p(\text{</s>} \mid \text{.})
\end{aligned}
$$

MORE READING ON SMOOTHING

https://www3.nd.edu/~dchiang/teaching/nlp/2019/notes/chapter3v2.pdf

DEALING WITH UNSEEN N-GRAMS: ABSOLUTE DISCOUNTING

[Option 3]: Additive smoothing penalizes common words way more

- intuitively, the counts of each word shouldn't decrease by more than ~1

n_{1+} = number of types seen at least 1 time

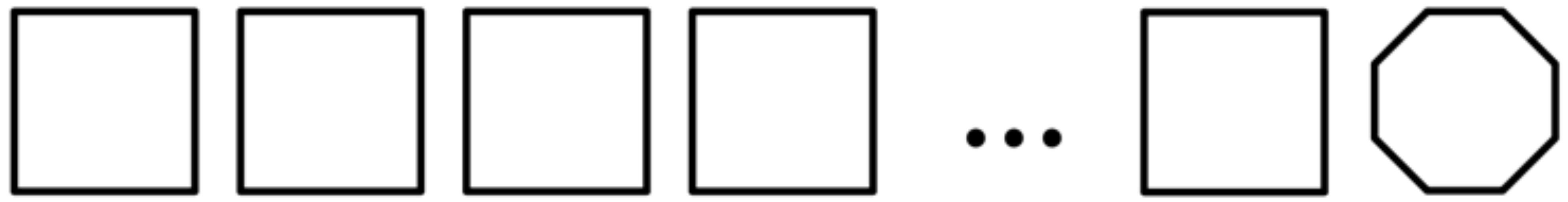
$$p(w) = \frac{\max(0, c(w) - d)}{|V|} + \frac{n_{1+} d}{V} \frac{1}{|\Sigma|}$$

Kneser-Ney smoothing: $d = \frac{n_1}{n_1 + 2n_2}$

Based on Good-Turing smoothing, invented by Alan Turing

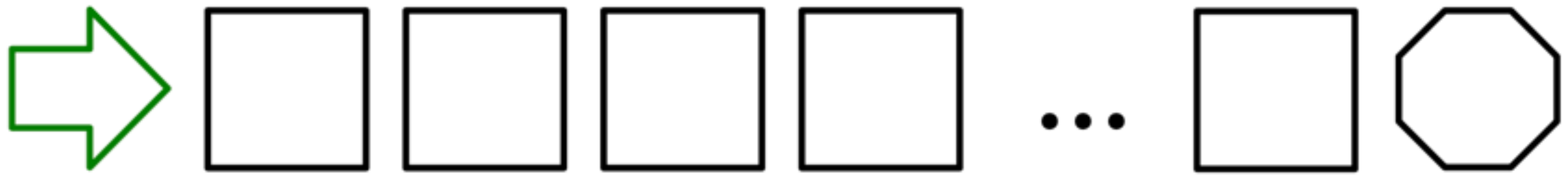
STARTING AND STOPPING

Unigram model



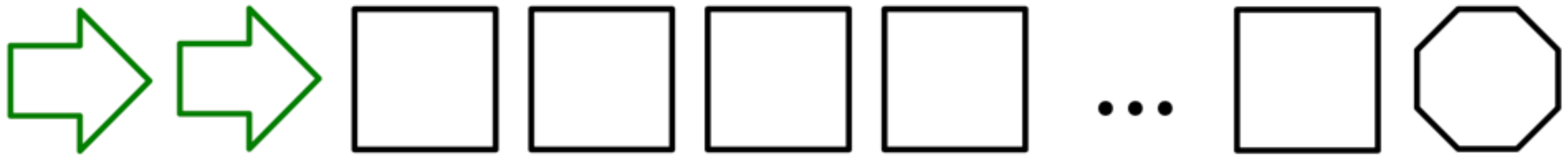
$$\prod_{i=1}^{L+1} p(w_i)$$

Bigram model



$$\prod_{i=1}^{L+1} p(w_i \mid w_{i-1})$$

Trigram model



$$\prod_{i=1}^{L+1} p(w_i \mid w_{i-2}, w_{i-1})$$

EVALUATION

EVALUATION

Log-likelihood:

$$LL(\mathcal{E}_{test}) = \sum_{E \in \mathcal{E}_{test}} \log P(E)$$

Per-word Log Likelihood:

$$WLL(\mathcal{E}_{test}) = \frac{1}{\sum_{E \in \mathcal{E}_{test}} |E|} \sum_{E \in \mathcal{E}_{test}} \log P(E)$$

Per-word (Cross) Entropy:

$$H(\mathcal{E}_{test}) = \frac{1}{\sum_{E \in \mathcal{E}_{test}} |E|} \sum_{E \in \mathcal{E}_{test}} - \log_2 P(E)$$

Perplexity:

$$ppl(\mathcal{E}_{test}) = 2^{H(\mathcal{E}_{test})} = e^{-WLL(\mathcal{E}_{test})}$$

PERPLEXITY OF DIFFERENT MODELS

Better models will have lower perplexity

WSJ — unigram: 962; bigram: 170; trigram: 109

Different tasks/datasets have different perplexity

— WSJ (109) vs Bus Information Queries (~25)

Higher the conditional probability → lower the perplexity

Perplexity is the *average branching rate*

Checkout this blog post: https://sjmielke.com/comparing-perplexities.htm

ASSIGNMENT 2

Due next week

NEXT CLASS PREVIEW

Neural Language Models