Language Models

Basic Terminologies

- Wordform
  - full inflected or derived form of a word
  - Amusing, amused, amusement etc
- **Lemma**
  - Dictionary form (canonical form) of each word
  - amusements => amusement
  - amused = > amuse
- **Stem**
  - the part of the word that never changes even when morphologically inflected
  - amused, amusing, amusement => amus

Basic Terminologies

- Word-Type
  - Concerning unique words
- Word-Token
  - Concerning occurrences of words

- "*A problem clearly stated is a problem half solved*"

  => 9 word-tokens

  => 7 word-types

Predict next word

- Once upon a …
- Introduction to Natural Language …
- I am taking CSE 628 this …

- Language Models = “word sequence prediction” as a probabilistic model.

Language Models (LMs)

- Also known as “N-gram models”
  - N-gram models
    - Definition: $p(x_n \mid x_{n-1}, \ldots, x_{n-N+1})$
    - predict the next word given N-1 previous words
  - 1-gram = unigram
    - $p(x_n)$
  - 2-gram = bigram
    - $p(x_n \mid x_{n-1})$
  - 3-gram = trigram
    - $p(x_n \mid x_{n-2}, x_{n-1})$
- Probability value is ***invariant*** with respect to ‘n’
- “N-gram” (without “models”) means N-word sequences

Markov Assumption in N-gram models

- Markov Assumption: next prediction depends only on the recent history rather than the entire history
  - K'th-order Markov Assumption: next prediction depends only on K most recent states
  - N-gram model is based on K-1'th order Markov assumption.
  - In N-gram models,
    $p(x_n \mid x_1, \ldots, x_{n-1})$ is approximated by
    $p(x_n \mid x_{n-N+1}, \ldots, x_{n-1})$

- Why make this assumption?

Training Language Models

- Training LMs means measuring (estimating) the probability of each possible n-gram in a “corpus”.

- Unigram: $p(\text{"}natural\text{"})$

$$= \frac{C\,(\text{"}natural\text{"})}{\sum_x C(x)}$$

$$= \frac{C\,(\text{"}natural\text{"})}{size\ of\ corpus\ in\ \#\ of\ words}$$

Training Language Models

- Recall the definition of conditional probability

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

- Bigram : $p(\text{"language"} \mid \text{"natural"})$

$$= \frac{C(\text{"natural language"})}{\sum_x C(\text{"natural } x\text{"})}$$

$$= \frac{C(\text{"natural language"})}{C(\text{"natural"})}$$

What about the joint probability P ("natural language") ?

Training Language Models

- Bigram: $p(x_n \mid x_{n-1})$

$$= \frac{C\,(x_{n-1}\, x_n)}{C\,(x_{n-1})}$$

- N-gram : $p(x_n \mid x_{n-N+1}, \dots, x_{n-1})$

$$= \frac{C\,(x_{n-N+1} \dots x_n)}{C\,(x_{n-N+1} \dots x_{n-1})}$$

- Does this estimation scheme yield a valid probability distribution?
- The number of "parameters" (combinations of different word sequences to count) grow exponentially as N increases.

Training Language Models

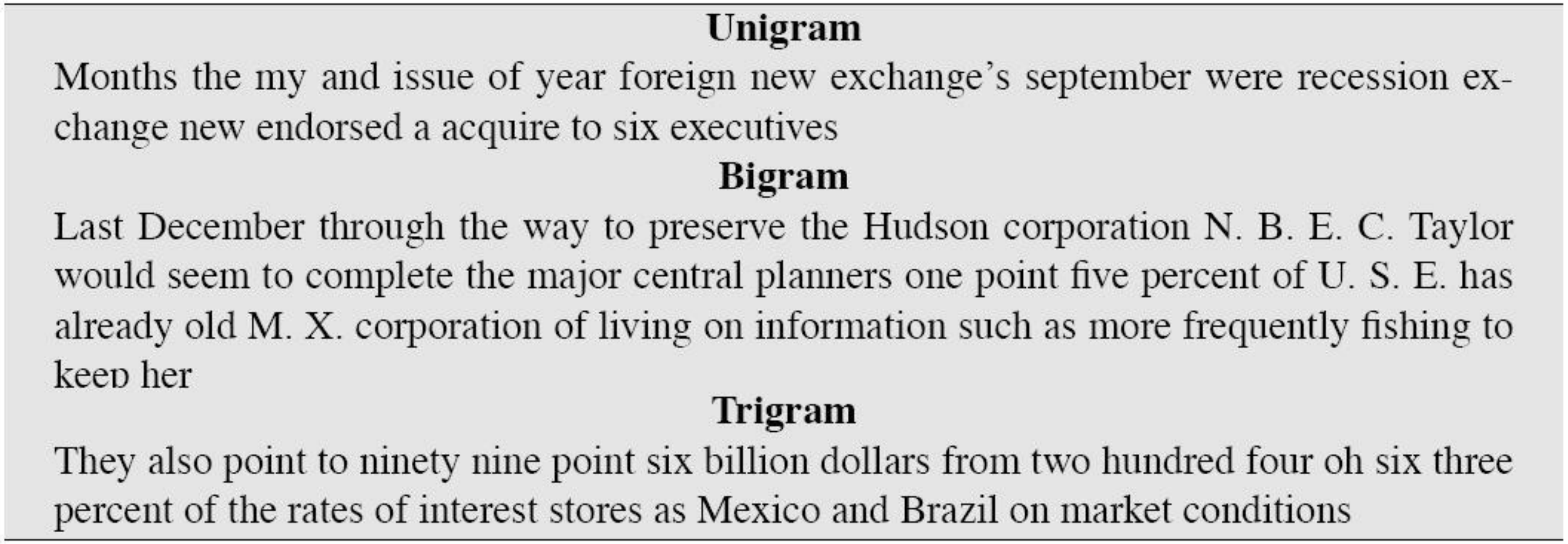
- N-gram : $p(x_n \mid x_{n-N+1}, \ldots, x_{n-1})$

$$= \frac{C\,(x_{n-N+1} \ldots x_n)}{C\,(x_{n-N+1} \ldots x_{n-1})}$$

- Above way of estimation based on counts is often called as "**Maximum likelihood estimation**" or **MLE**, because such estimation maximizes the likelihood of the training data.

- Count of a particular n-gram varies a lot depending on the genre and the general topic of the corpus.
  - Count of "natural language processing" in Shakespeare?
  - Therefore, trained language models also vary a lot depending on the corpus.

Natural Language "Generation" using LMs
--- by randomly sampling n-grams from LMs



Unigram

Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

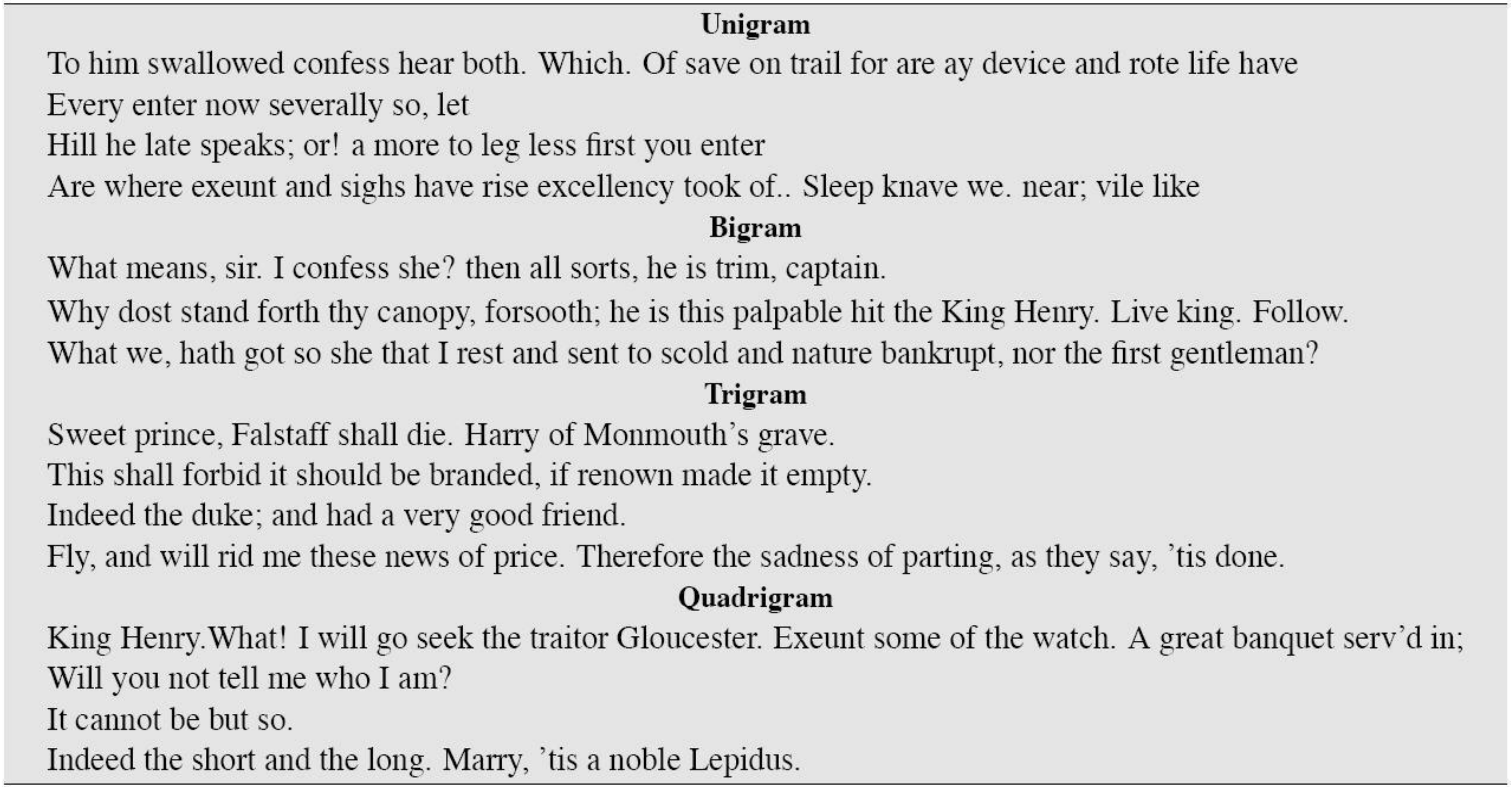
Bigram

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

Trigram

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

Natural Language "Generation" using LMs



Unigram

To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
Every enter now severally so, let
Hill he late speaks; or! a more to leg less first you enter
Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

Bigram

What means, sir. I confess she? then all sorts, he is trim, captain.
Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

Trigram

Sweet prince, Falstaff shall die. Harry of Monmouth's grave.
This shall forbid it should be branded, if renown made it empty.
Indeed the duke; and had a very good friend.
Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

Quadrigram

King Henry.What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;
Will you not tell me who I am?
It cannot be but so.
Indeed the short and the long. Marry, 'tis a noble Lepidus.

Probability of a Sentence?

Given a sentence $\boldsymbol{x} = x_1, x_2, x_3, \ldots xn$

How can we estimate the probability of the entire sentence P($x_1, x_2, x_3, \ldots xn$) ?

Say, P (“I couldn’t submit my homework, because my horse ate it”) ?

➔ Direct estimation is practically impossible. Why?

➔ N-gram models can be used to compute probabilities of longer word sequences, such as P(sentence), P(document), or even P(corpus)

Probability of a long sequence by N-gram approximation

- Word sequences

$$w_1^n = w_1 \dots w_n$$

- Chain rule of probability

$$P(w_1^n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2)\dots P(w_n \mid w_1^{n-1}) = \prod_{k=1}^{n} P(w_k \mid w_1^{k-1})$$

- Bigram approximation $P(w_1^n) = \prod_{k=1}^{n} P(w_k \mid w_{k-1})$

- N-gram approximation

$$P(w_1^n) = \prod_{k=1}^{n} P(w_k \mid w_{k-N+1}^{k-1})$$

Unknown Words (OOV)

- How to handle words in the test corpus that did not occur in the training data, i.e. ***out of vocabulary*** (OOV) words?
- Train a model that includes an explicit symbol for an unknown word (<UNK>).
  - Choose a vocabulary in advance and replace other words in the training corpus with <UNK>.
  - Replace the first occurrence of each word in the training data with <UNK>.

Evaluation of Language Models

- Evaluate on a test dataset
  - Must be disjoint from the training dataset!!!!!!

Training Data	Held-out Data	Develop-ment Data	Test Data

- Held-out Data (optional)
  - often used to tune extra parameters
- Development Data
  - pretend it is a test data while playing with the models (so that we don’t overfit or cheat on the test data)
  - Sometimes people use the term “held-out data” and “development data” interchangeably.

Evaluation of Language Models

- Ideally, evaluate the use of the model in a specific application (***extrinsic***, ***in vivo***)
  - Ideal but expensive

- Evaluate on the ability to model the test corpus (***intrinsic***).
  - Improvement in intrinsic evaluation does not guarantee improvement in extrinsic evaluation. (hopefully they correlate.)
  - Cheaper & faster

Perplexity

- Measure of how well a model “fits” the test data.
- Uses the probability that the model assigns to the test corpus.
- Normalizes for the number of words in the test corpus and takes the inverse.

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

- Why normalize?

- Models with lower perplexity is better

➔ Higher Perplexity means model is ***“perplexed or surprised”*** to see the test data

Evaluation on WSJ

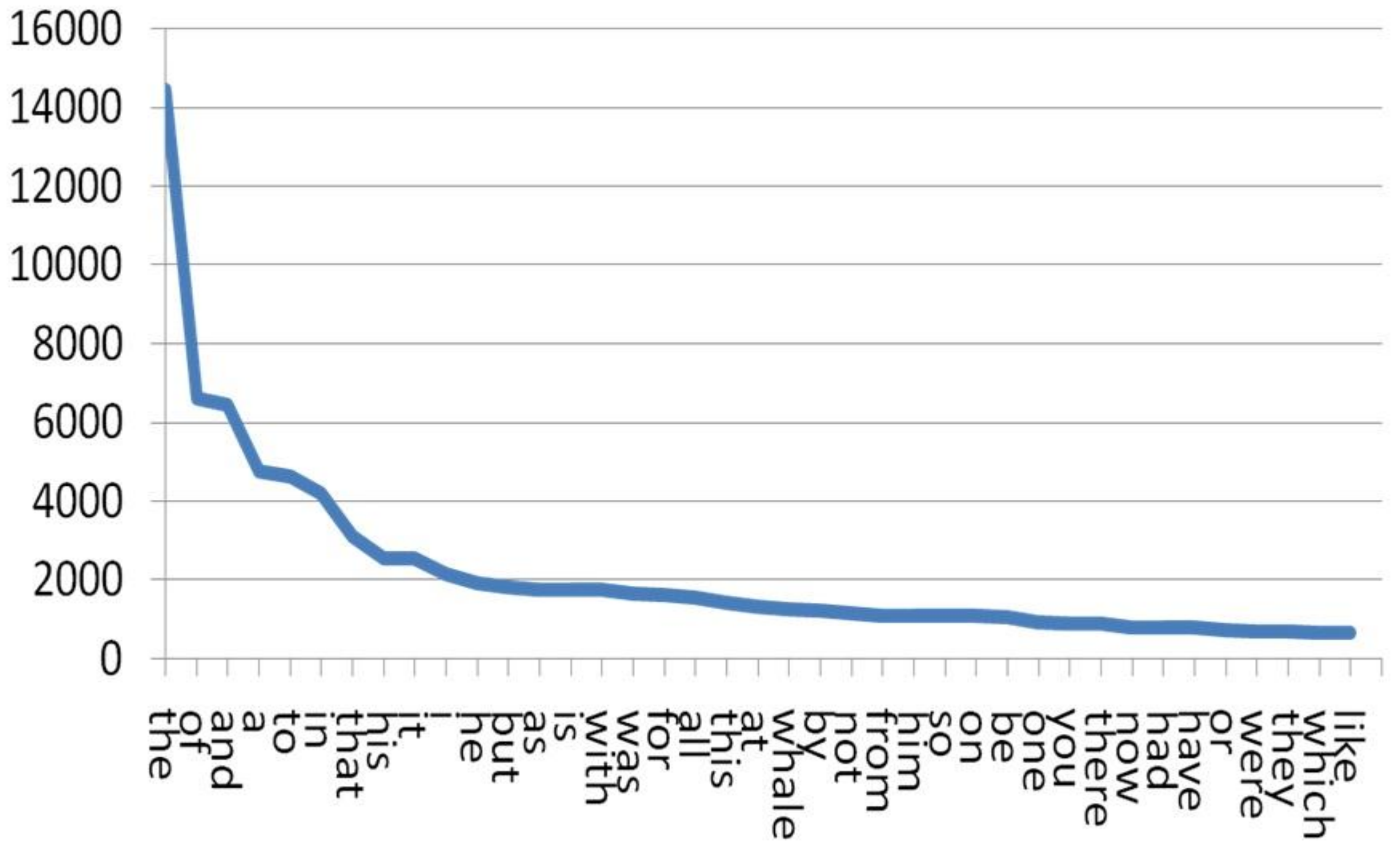
- Models trained on 38 million words from the Wall Street Journal (WSJ) using a 19,979 word vocabulary.
- Evaluate on a disjoint set of 1.5 million WSJ words

- Perplexity: Trigram < Bigram < Unigram (in this case)
- Which N-gram is the best in general?

	Unigram	Bigram	Trigram
Perplexity	962	170	109

Zipf's Law (Power Law)

- We say natural language follows Zipf's Law distribution. (a kind of "Power Law distribution"

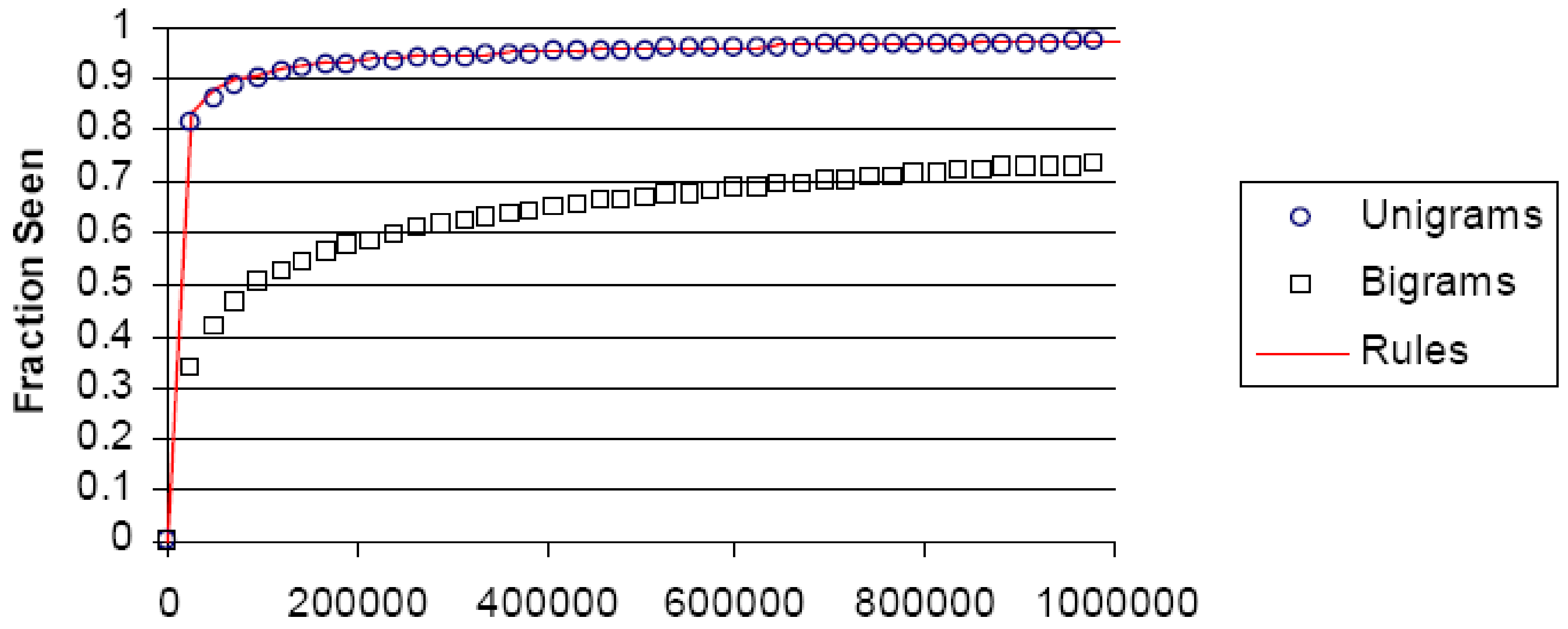


Zipf's Law (Power Law)

- many types of data studied in the physical and social sciences can be approximated with a Zipf's distribution

- The frequency of any word is inversely proportional to its rank in the frequency table
  - There is a constant *k* such that *freq* * *rank* = *k*
- Each corpus has a different, fairly small "working vocabulary"
- In aggregate, the long tail of low frequency words comprise most of the text.

“Data Sparsity”
– A big problem in statistical NLP



What can we do with Language Models?

- Speech recognition
  - “I ate a cherry”

    VS
  - “Eye eight uh Jerry”

  - It is hard to recognize speech.

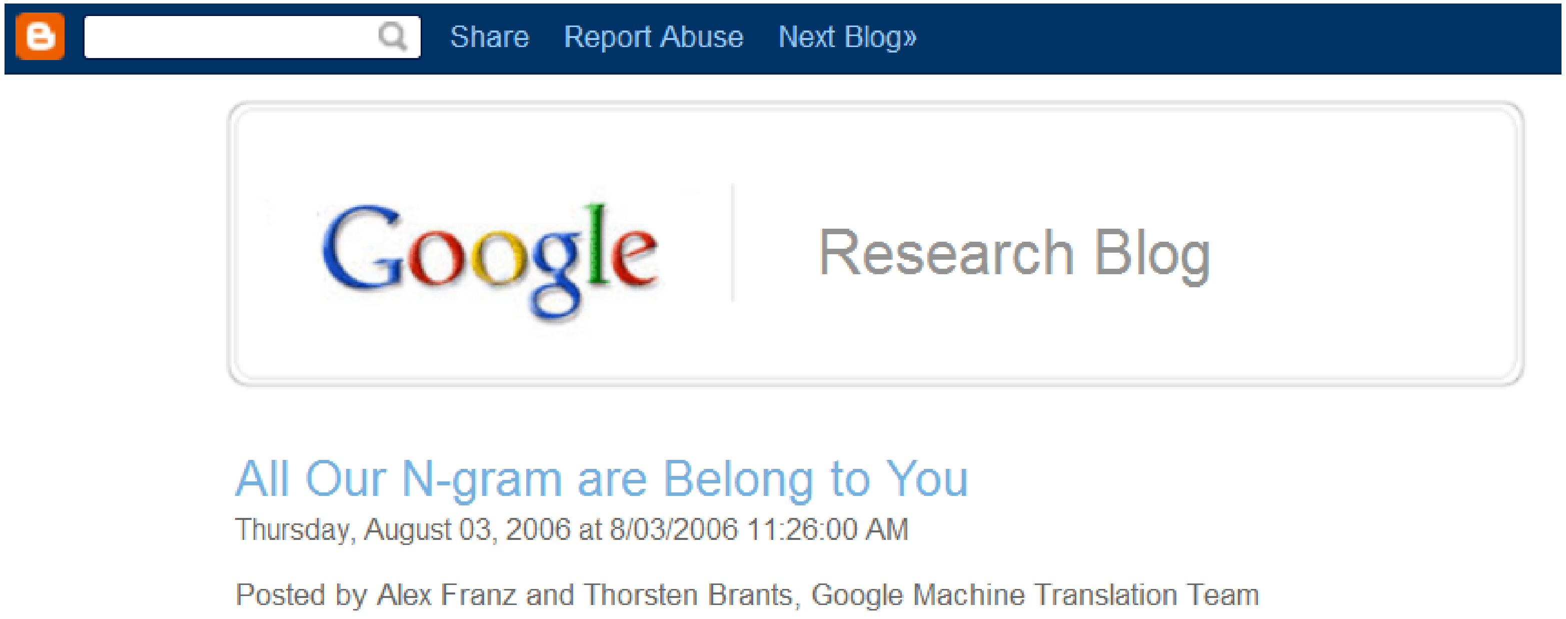
    VS
  - It is hard to wreck a nice beach.

What can we do with Language Models?

- Speech recognition
  - “I ate a cherry” is a more likely than “Eye eight uh Jerry”
- OCR & Handwriting recognition
  - More probable sentences are more likely correct readings.
- Machine translation
  - More likely sentences are probably better translations.
- Generation
  - More likely sentences are probably better NL generations.
- Context sensitive spelling correction
  - “Their are problems wit this sentence.”
- Text Categorization
  - How?

Google N-gram Release

http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html


Share Report Abuse Next Blog»

Google Research Blog

All Our N-gram are Belong to You

Thursday, August 03, 2006 at 8/03/2006 11:26:00 AM

Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word n-gram models for a variety of R&D projects, such as statistical machine translation, speech recognition, spelling correction, entity detection, information extraction, and others.

Google N-gram Release

- We believe that the entire research community can benefit from access to such massive amounts of data. ..... We processed 1,024,908,267,229 words of running text and are publishing the counts for all 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.

  - File sizes: approx. 24 GB compressed (gzip'ed) text files
    Number of tokens: 1,024,908,267,229
    Number of sentences: 95,119,665,584
    Number of unigrams: 13,588,391
    Number of bigrams: 314,843,401
    Number of trigrams: 977,069,902
    Number of fourgrams: 1,313,818,354
    Number of fivegrams: 1,176,470,663

Google N-gram Count Examples

- ceramics collectables collectibles 55
  ceramics collectables fine 130
  ceramics collected by 52
  ceramics collectible pottery 50
  ceramics collectibles cooking 45
  ceramics collection , 144
  ceramics collection . 247
  ceramics collection </S> 120

- serve as the info 42
  serve as the informal 102
  serve as the information 838
  serve as the informational 41
  serve as the infrastructure 500
  serve as the initial 5331
  serve as the initiating 125
  serve as the initiation 63
  serve as the initiator 81

Recap (Quiz for the next class)

- What are “lemma” and “stem” of words?
- how do you compute the probability of a sentence or a corpus from LM?
- what is perplexity? what does it tell you?
- why do we normalize the perplexity by the length of the corpus?
- explain training data, test data, held out data, development data with respect to LM.
- what is the problem with unknown words?
- explain “data sparcity”
- what is zipf’s law? how do you relate it to LM?
- explain how to use language models for text categorization

Recap (Quiz for the next class)

- what are MLE estimates of unigram, bigram, trigram?

- Consider the following sequence, assuming the vocabulary V = {a, b, c}

  a b b c a c a b b b a b a c a c c

  - what are MLE estimates of unigram, bigram, trigram?
  - what is p (abb) ?
  - what is p (ab)?
  - what is p (b) ?
  - what is p (ab|b)?