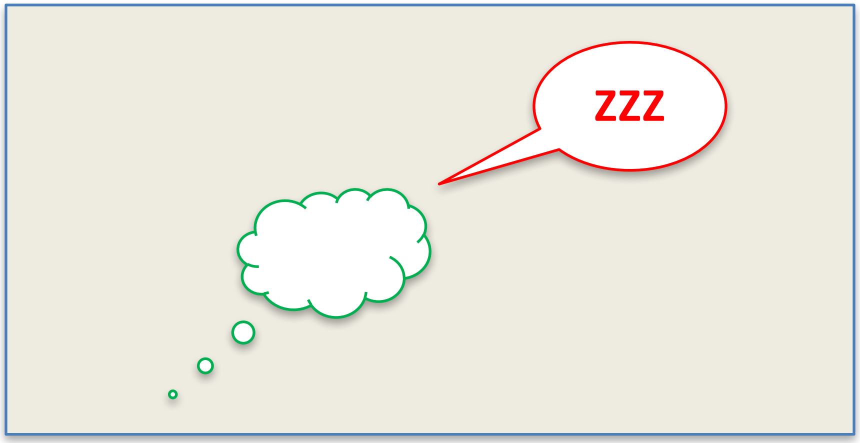
Machine Learning Method for Natural Language Processing

Lecture 2: Language Models



Colorless green ideas sleep furiously

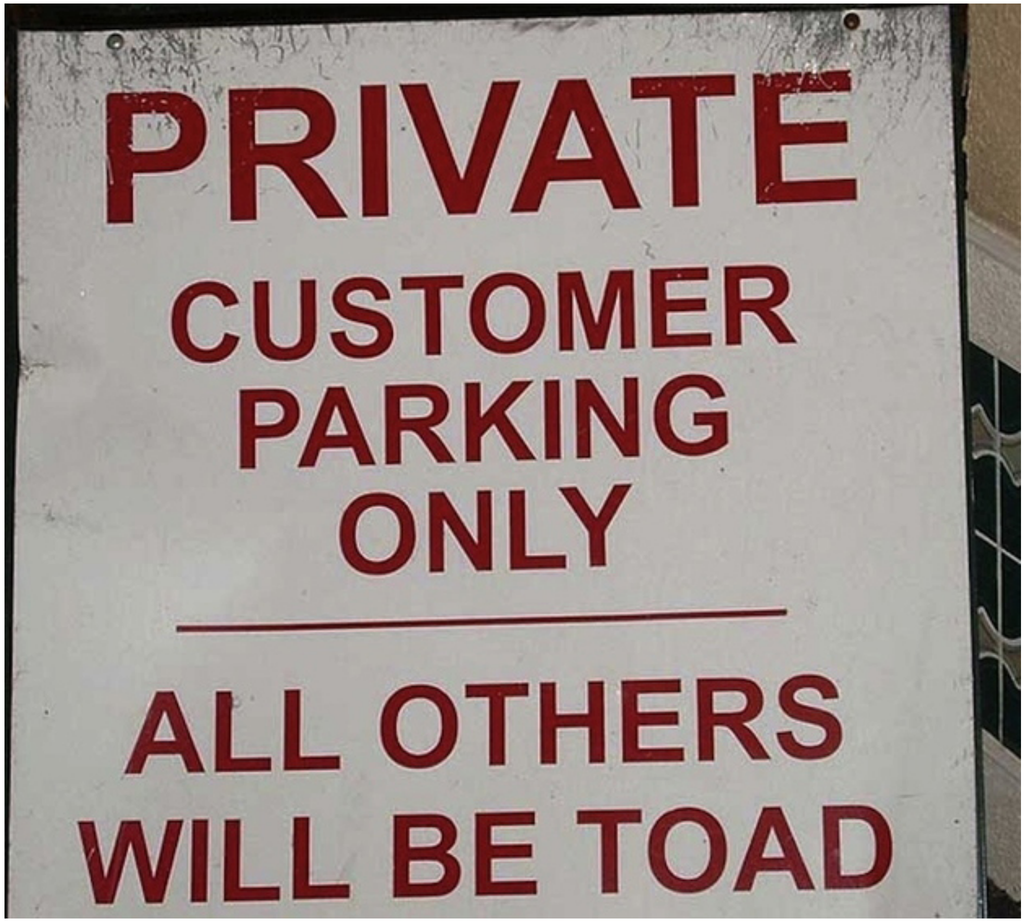
Dan Goldwasser

Purdue University

dgoldwas@purdue.edu

CS 590NLP

And now to something completely different!



Today

First step in statistical NLP

- Let’s start with something simple
- *Context sensitive spelling correction*
  - Many words look and sound similar, confusing even native speakers

“affect - effect”, “their – they're” “**towed-toad**”

  - Not a spelling mistake, but not the intended word
  - Depends on the context in which it is used

NLP Error Correction Example

"I don't know {whether,weather} to laugh or cry"

- *What is the decision problem?*
  - Function : sentence ➔ {whether, weather}
- Can you write a program to decide?
- *How can you apply ML techniques to solve it?*
  - Can you learn a classifier?
  - **What are positive and negative examples?**
- Is there a simpler way?
  - **Assumption: Most text written is correct**

Example based on Dan Roth slides

Statistical Language Modeling

- **Intuition**: by looking at large quantities of text we can find statistical regularities
  - *Distinguish between correct and incorrect sentences*
- Language models define a probability distribution over strings (e.g., sentences) in a language.
- ***We can use language models to score and rank sentences***

"I don't know {whether,weather} to laugh or cry"

P("I don't.. weather to laugh..") **><** P("I don't.. whether to laugh..")

Language Modeling

- We have a finite vocabulary
  - V = {the, a, Alice, Bob, likes, sleeps, apples} U {STOP}

- Based on this vocabulary we can generate an infinite number of strings:
  - the STOP
  - a Stop
  - the Apples sleeps STOP
  - Alice likes Bob STOP

Some of these strings make more sentence than others!

Language Modeling

- Let's assume we have a set of "real" strings
  - For example, English sentences samples
- We want to learn a probability distribution over strings

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \quad p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

- **For example:**
  - *p(the STOP) = 0.0000001*
  - *p(the Apples sleeps STOP) =0.000000001*
  - *p(Alice likes Bob STOP) = 0.00001*

"reasonable" strings are more likely than unreasonable

Language Modeling

- A good way for doing that:
  - Collect a very large corpora of reasonable sentences
    - All of English Wikipedia.
    - All the web!
  - Use this data to estimate a probability distribution directly
- Assume we have N sentences
  - For any sentence $x_1,\ldots,x_n$, we will denote $c(x_1,\ldots,x_n)$ the number of times this sentence is observed in the data
- Simple Estimate: $p(x_1 \ldots x_n) = \frac{c(x_1 \ldots x_n)}{N}$

What is a potential problem with this approach?

Language Modeling

- We will need an unreasonable amount of data if we want reasonable estimates of sentences
  - Instead, we notice that the probability of a sentence can be viewed as a product of the probabilities of its words
    - E.g., *Mr. Smith goes to Washington*

$$\begin{aligned} P(X_1, X_2, \ldots, X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)....P(X_n|X_1, ...X_{n-1}) \\ &= P(X_1)\prod_{i=2}^{n} P(X_i|X_1 \ldots X_{i-1}) \end{aligned}$$

  - Can we use that to get simpler estimates?
    - *p(Mr.) p(Smith) p(goes) p(to) p(Washington)*
  - ***We will need to make some simplifying assumptions!***

Independence

Two random variables X *and* Y *are* ***independent*** *if*

$$P(X,Y) = P(X)P(Y)$$

If X and Y are independent, then $P(X \mid Y) = P(X)$

$$\begin{aligned} P(X \mid Y) &= \frac{P(X,Y)}{P(Y)} \\ &= \frac{P(X)P(Y)}{P(Y)} \quad (X, Y \quad \text{independent}) \\ &= P(X) \end{aligned}$$

Language Modeling with N-grams

- A language model over a given vocabulary V assigns probabilities to strings drawn from V*

$$P(w_1 \dots w_i) \quad = \quad P(w_1)P(w_2|w_1)P(w_3|w_1 w_2)\dots P(w_i|w_1 \dots w_{i-1})$$

- An n-gram model assume each word depends only on the previous n-1 words:

$$P_{ngram}(w_1 \dots w_i) \quad := \quad P(w_1)P(w_2|w_1)\dots P(\underbrace{w_i}_{nth\ word} \ | \underbrace{w_{i-n-1} \dots w_{i-1}}_{prev.\ n-1\ words})$$

Unigram model	$P(w_1)P(w_2)\dots P(w_i)$
Bigram model	$P(w_1)P(w_2\|w_1)\dots P(w_i\|w_{i-1})$
Trigram model	$P(w_1)P(w_2\|w_1)\dots P(w_i\|w_{i-2}\ w_{i-1})$

Model Estimation

- The last remaining issue – how can we estimate the models parameters ***$p(w_i|w_{i-2},w_{i-1})$***
- Simple solution – *counting*!
  - *Also known as* ***Maximum likelihood estimate***

$$p(w_i|w_{i-2},w_{i-1}) = \frac{\mathrm{count}(w_{i-2}, w_{i-1}, w_i)}{\mathrm{count}(w_{i-2}, w_{i-1})}$$

How many parameters does the model need to estimate?

Model Estimation

- **How many parameters does the model need to estimate?**
  - Let’s assume a trigram model, defined over vocabulary V
  - The number of parameters is $|\mathbf{V}|^3$
  - Let’s assume: $|V| = 20K \quad < |V_{\text{Shakespeare}}|$
  - We’ll have to estimate $\mathbf{8 \times 10^{12}}$ parameters
- **How many will we need to estimate for a unigram model?**
  - **Why not just do that?**

Generating from a distribution

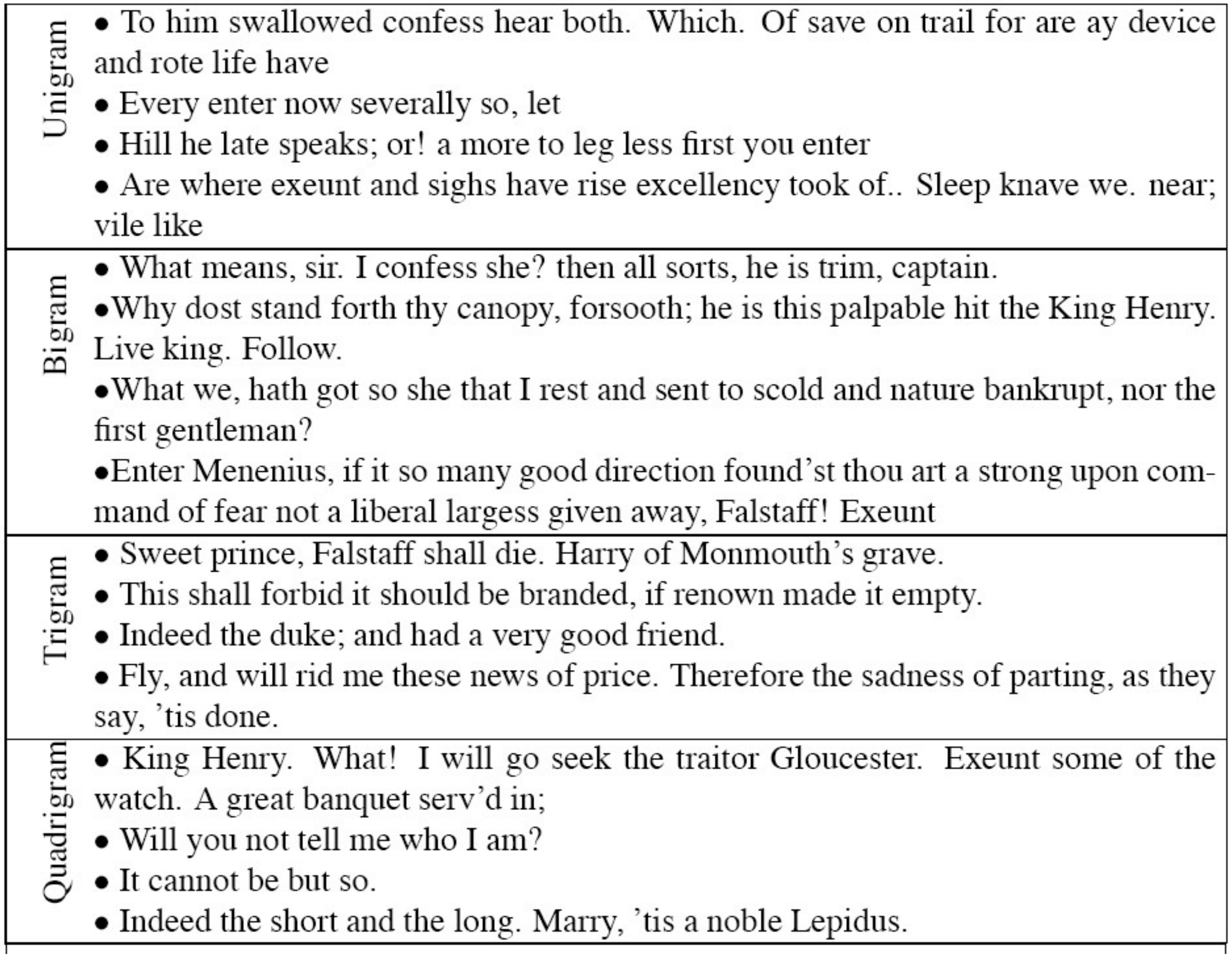
- You can sample text from language models

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

Generating Shakespeare



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? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
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.

Generating Shakespeare

The Shakespeare corpus consists of N=884,647 word **tokens** and a vocabulary of V=29,066 word **types**

Only 30,000 word types occurred.
Any word that does not occur in the training data has zero probability!

Shakespeare produced 300,000 bigram types out of V^2= 844 million possible bigram types. 99.96% of the possible bigrams were never seen

Only 0.04% of all possible bigrams occurred. Any bigram that does not occur in the training data has zero probability!

Simple Solution: Smoothing (Add-1, Laplacian)

$$\text{MLE} \quad P(w_i) = \frac{C(w_i)}{\sum_j C(w_j)} = \frac{C(w_i)}{N}$$

$$\text{Add One} \quad P(w_i) = \frac{C(w_i)+\mathbf{1}}{\sum_j (C(w_j)+\mathbf{1})} = \frac{C(w_i)+\mathbf{1}}{N+\mathbf{V}}$$

A general tradeoff in ML, we will meet it again!

Evaluating Language Models

- Assuming that we have a language model, how can we tell if it's good?
- **Option 1**: ***try to generate Shakespeare..***
  - This is know as *Qualitative evaluation*
- ***Option 2: Quantitative evaluation***
  - ***Option 2.1:*** *See how well you do on Spelling correction*
    - *This is known as **Extrinsic** Evaluation*
  - ***Option 2.2:*** *Find an independent measure for LM quality*
    - *This is known as **Intrinsic** Evaluation*

Perplexity

- Assume we have a language model
- Sample new (test) data for evaluation: $s_1,\ldots,s_m$
- We will look at the probability of the test data under out model: $\prod_{i=1}^{m} p(s_i)$
- Or, for convenience the log of that probability:

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i)$$

- **Perplexity is defined as:**

$$\text{Perplexity} = 2^{-l} \qquad l = \frac{1}{M} \sum_{i=1}^{m} \log p(s_i)$$

(M is the number of instances in the test data)

Perplexity

- Given a vocabulary V of size |V| = N
- We have a very "bad" model:

  $p(w|w_{i-2}, w_{i-1}) = 1/N$

- What is the perplexity of this model?

Perplexity = 2^{-l}

$$l = \frac{1}{M} \sum_{i=1}^{m} \log p(s_i)$$

$$\Downarrow$$

$$l = \log \frac{1}{N}$$

➔The perplexity is N

Simple Intuition: Given the context what is *the effective "branching factor" for the current word.* ➔ Lower is better!

Evaluating Language Models

- **Which one is better** – unigram, bigram, trigram?
  - Can perplexity tell us that?
- Goodman 2001: |V| =50,000
  - **Trigram** model: Perplexity =74
  - **Bigram** model: Perplexity = 137
  - **Unigram** model: Perplexity = 955
- **Is a trigram model always better? (i.e., lower perplex)**

Back-off and Interpolation

- A trigram model could *overfit* to the training data
  - *We will not have reliable estimates for many parameters!*
- **Option 1: Smoothing**
  - We briefly looked at Laplacian smoothing, many others
- **Option 2: Back-off**
  - Instead of accounting for unseen trigrams, back-off to a simpler model when needed
    - Estimate P(z|x,y) ➔ C(z,x,y) = 0 ➔ use P(z|y) ➔... ➔ P(z)
    - A little bit more complicated.. (make sure we still have a distribution)
- **Option 3: combine all the models!** (interpolate)

Linear Interpolation

- **Linear Interpolation of several models:**
  - **Unigram**: $q_{\mathsf{ML}}(w_i) = \frac{\mathsf{Count}(w_i)}{\mathsf{Count}()}$
  - **Bigram**: $q_{\mathsf{ML}}(w_i \mid w_{i-1}) = \frac{\mathsf{Count}(w_{i-1}, w_i)}{\mathsf{Count}(w_{i-1})}$
  - **Trigram**: $q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) = \frac{\mathsf{Count}(w_{i-2}, w_{i-1}, w_i)}{\mathsf{Count}(w_{i-2}, w_{i-1})}$
  - **Our combined estimate:**

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2 \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3 \times q_{\mathsf{ML}}(w_i) \end{aligned}$$

$\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i.

Linear Interpolation

- **The new estimate defines a distribution**

$$\sum_{w\in\mathcal{V}'} q(w \mid u,v)$$

$$= \sum_{w\in\mathcal{V}'} [\lambda_1 \times q_{\mathsf{ML}}(w \mid u,v) + \lambda_2 \times q_{\mathsf{ML}}(w \mid v) + \lambda_3 \times q_{\mathsf{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\mathsf{ML}}(w \mid u,v) + \lambda_2 \sum_w q_{\mathsf{ML}}(w \mid v) + \lambda_3 \sum_w q_{\mathsf{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

$$= 1$$

- *Similarly, we can show that p(w|u,v) is greater than 0*

Linear Interpolation

- **How can we set the values for λ's ?**
- Use a held out validation corpus
  - Do not use it for training!
- Choose λ that maximize the probability of that dataset
  - Estimate the language models over **training** data
  - Search over different values of λ, and look for optimal values over the **validation** set

Using Language Models

- *You now have a pretty useful statistical tool*
  - Context Sensitive Spelling correction
    - Build a language model from a **HUGE** corpus
    - Keep a list of confusing words
    - Check which candidate has the highest probability
  - What is the annotation effort required for this task?
- *There are many other uses for language models*
  - you are a spammer, who got paid to write very positive reviews on yelp for the worst bar in Purdue
  - *How can you use language models?*

History (or limitations of LM)

- Chomsky (*syntactic Structures 1957*) presented a problem with statistical LM:

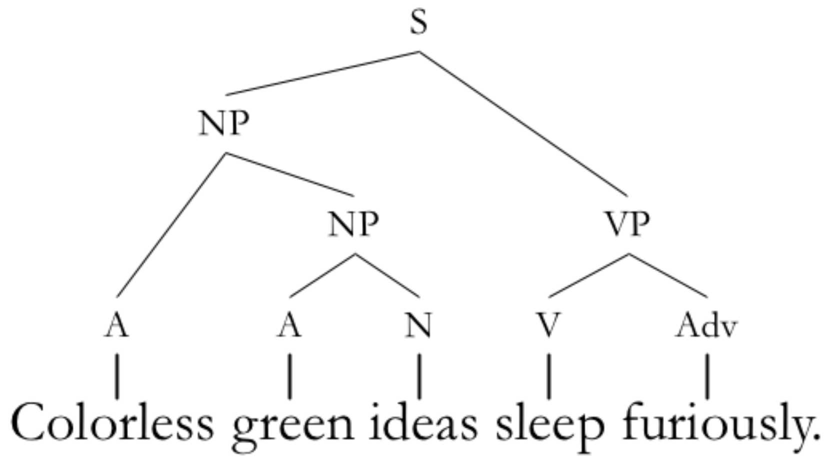
(1) Colorless green ideas sleep furiously.

(2) Furiously sleep ideas green colorless.

Both are nonsensical sentences, but (1) is grammatical and (2) is not

Assume we build a model for grammatical English.

Compare P((1)) to P((2))



Questions?