CS 378: Natural Language Processing

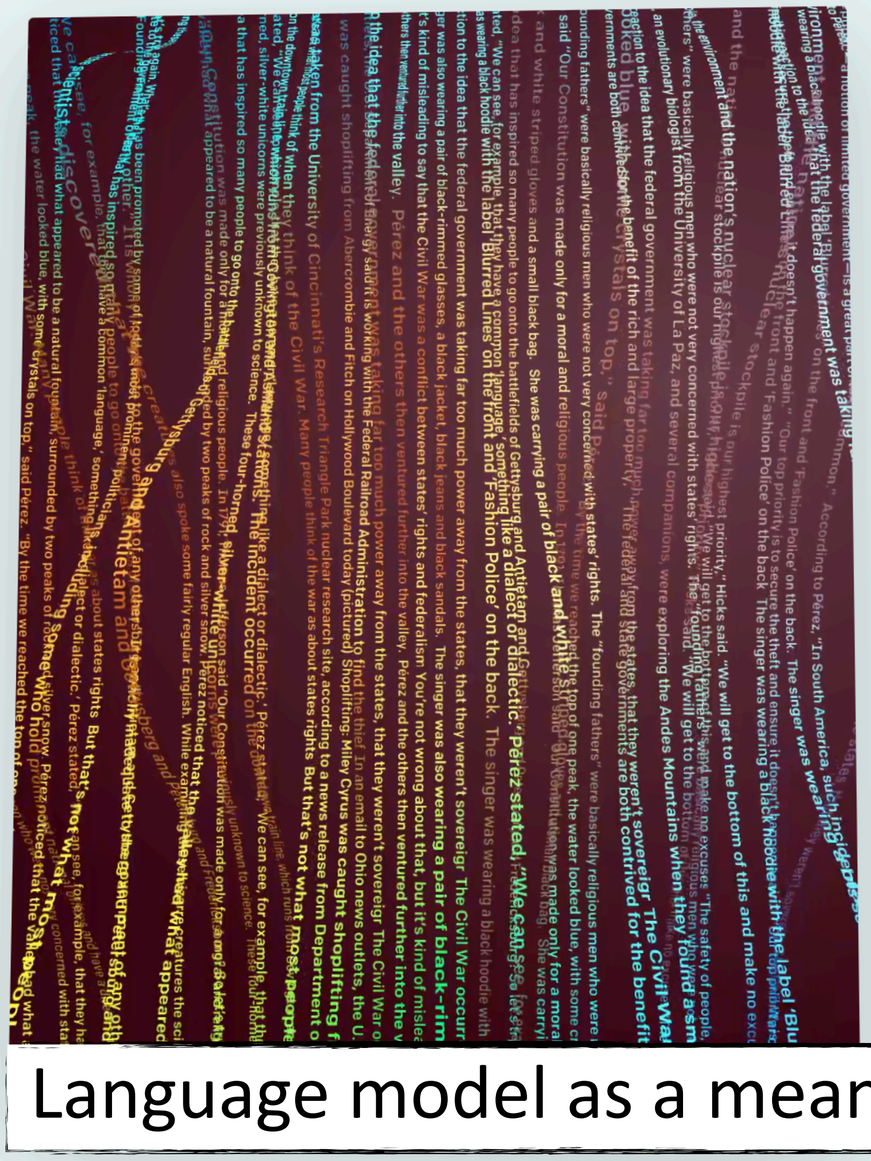
Lecture 12: Language Modeling

Eunsol Choi

Slides adapted from Greg Durrett, Yoav Artzi, Yejin Choi, Princeton NLP

Language Modeling

- Why should we care?
- N-gram models
- Evaluating language models
- Preview: Neural language model



FEBRUARY 14, 2019

Better Language Models and Their Implications

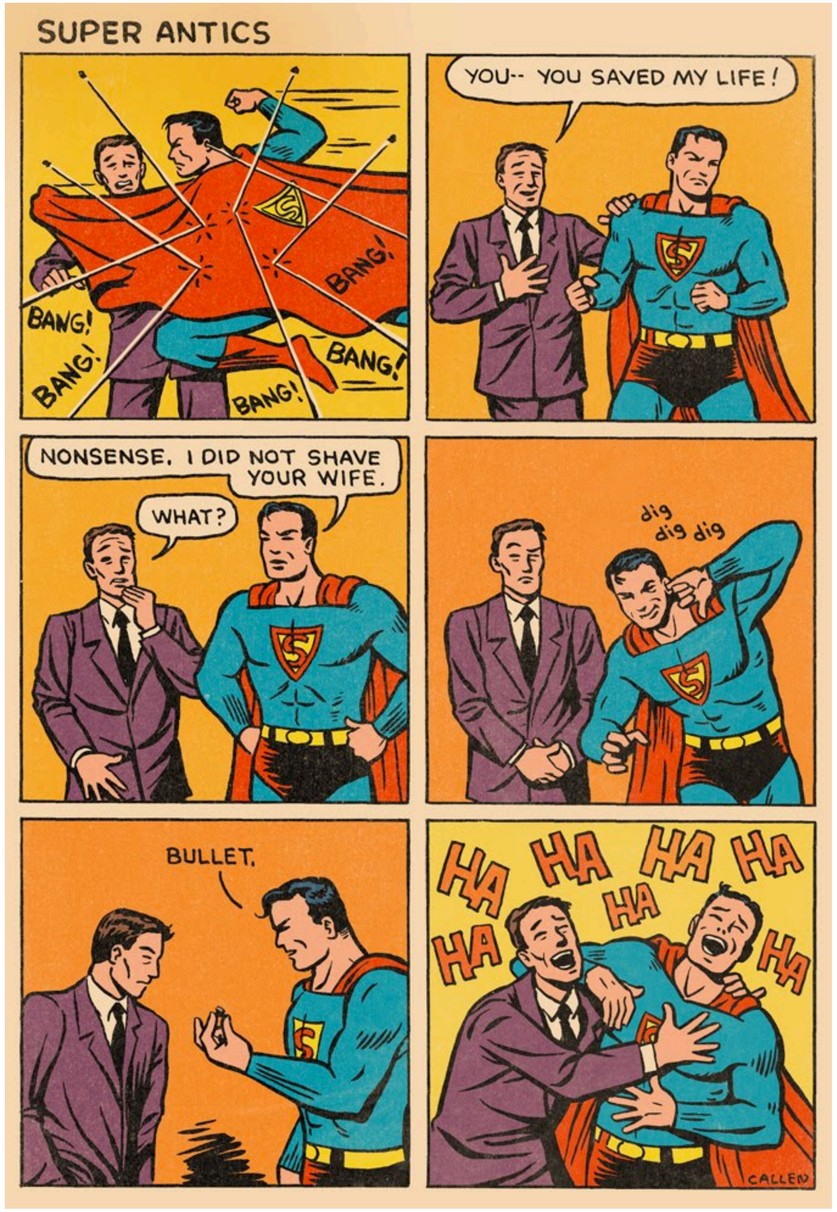
We've trained a large-scale unsupervised language model which generates coherent paragraphs of text, achieves state-of-the-art performance on many language modeling benchmarks, and performs rudimentary reading comprehension, machine translation, question answering, and summarization — all without task-specific training.

Language model as a means to get a representation of text

READ PAPER

READ MORE

Why do we care?



- Useful for many NLP tasks
  - automatic speech recognition
    - audio in, speech out
- Language generation
  - Is the model generated sentence high quality?

Applications of Language Modeling

- All generation tasks: translation, dialogue, text simplification, paraphrasing, etc.
- Grammatical error correction
- Predictive text
- Pretraining!

The noisy channel model

- We want to predict a sentence given acoustics:

$$w^* = \arg\max_w P(w|a)$$

- Noisy channel approach:

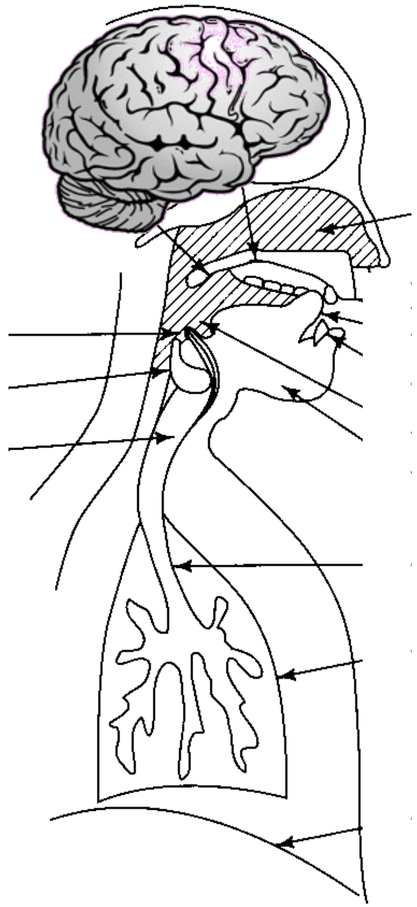
$$w^* = \arg\max_w P(w|a)$$

$$= \arg\max_w P(a|w)P(w)/P(a)$$

$$\propto \arg\max_w P(a|w)P(w)$$

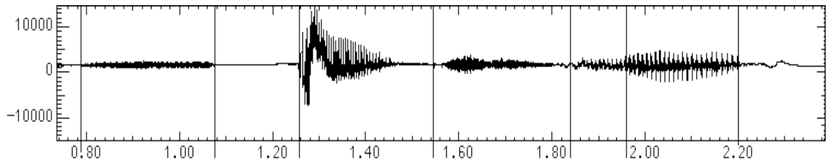
Acoustic model: Distributions over acoustic waves given a sentence

Language model: Distributions over sequences of words (sentences)



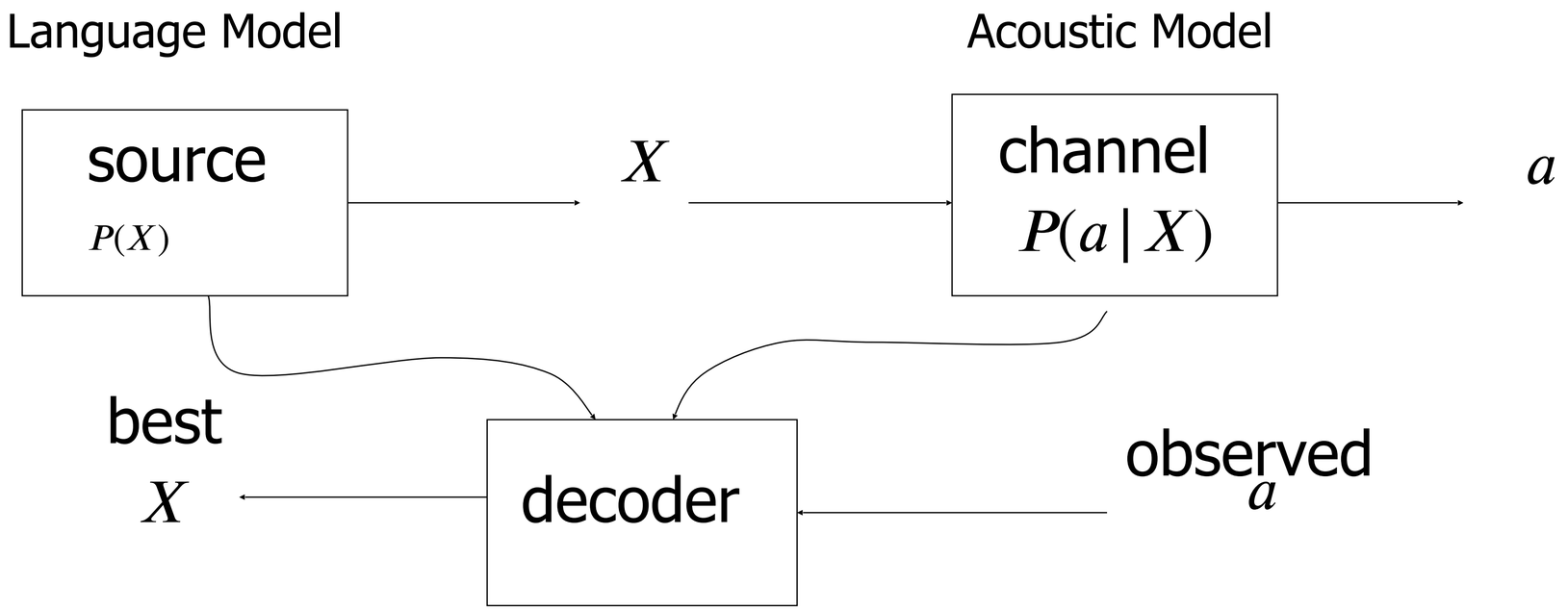
Acoustically Scored Hypotheses

$$\propto \arg\max_{w} P(a|w)P(w)$$



the station signs are in deep in english	-14732
the stations signs are in deep in english	-14735
the station signs are in deep into english	-14739
the station 's signs are in deep in english	-14740
the station signs are in deep in the english	-14741
the station signs are indeed in english	-14757
the station 's signs are indeed in english	-14760
the station signs are indians in english	-14790
the station signs are indian in english	-14799
the stations signs are indians in english	-14807
the stations signs are indians and english	-14815

ASR noisy channel



$$\arg\max_X P(X \mid a) = \arg\max_X P(a \mid X)P(X)$$

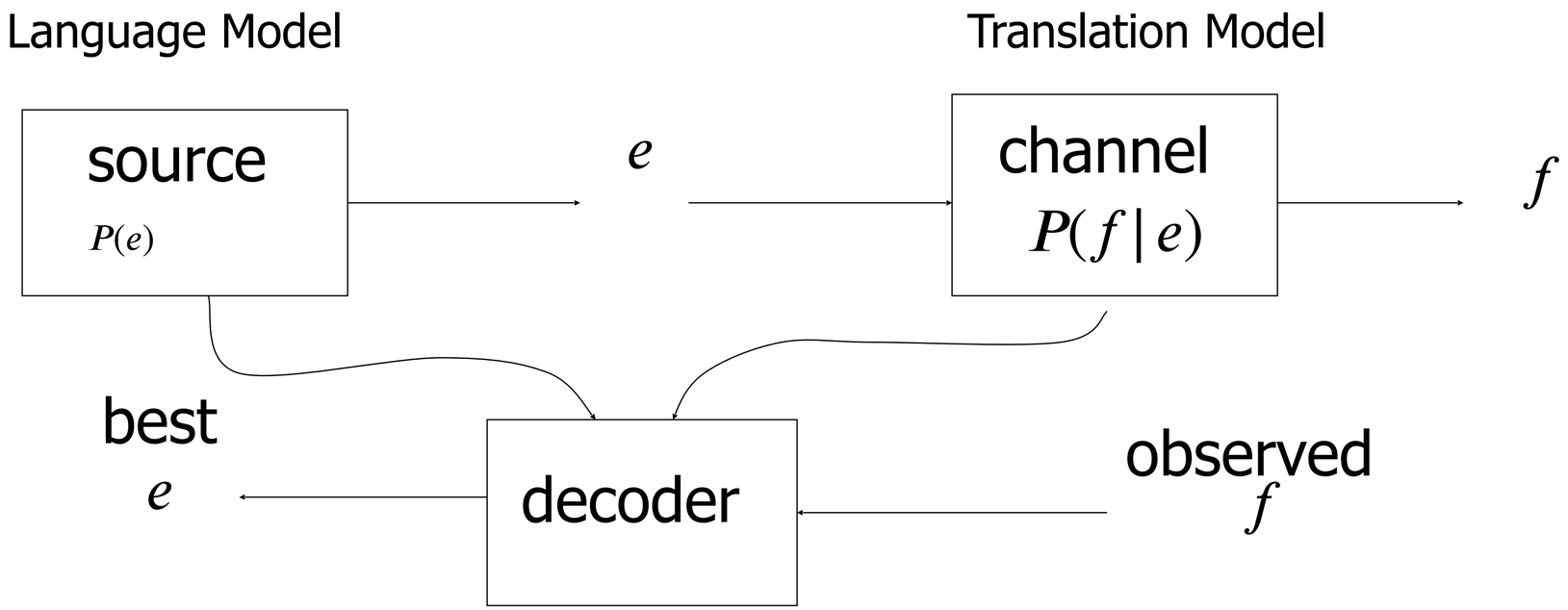
Translation as Code Breaking

"Also knowing nothing official about, but having guessed and inferred considerable about, the powerful new mechanized methods in cryptography—methods which I believe succeed even when one does not know what language has been coded—one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.' "

Warren Weaver

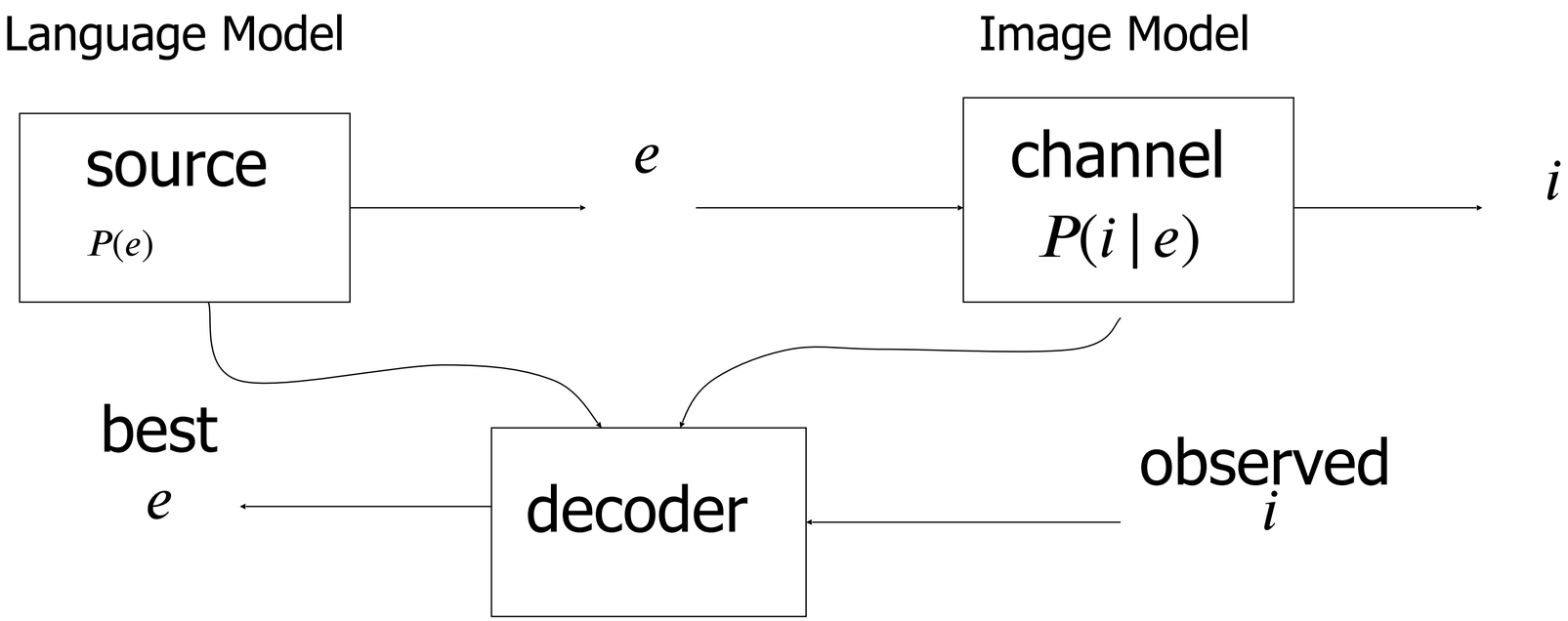
(1955:18, quoting a letter he wrote in 1947)

MT noisy channel model



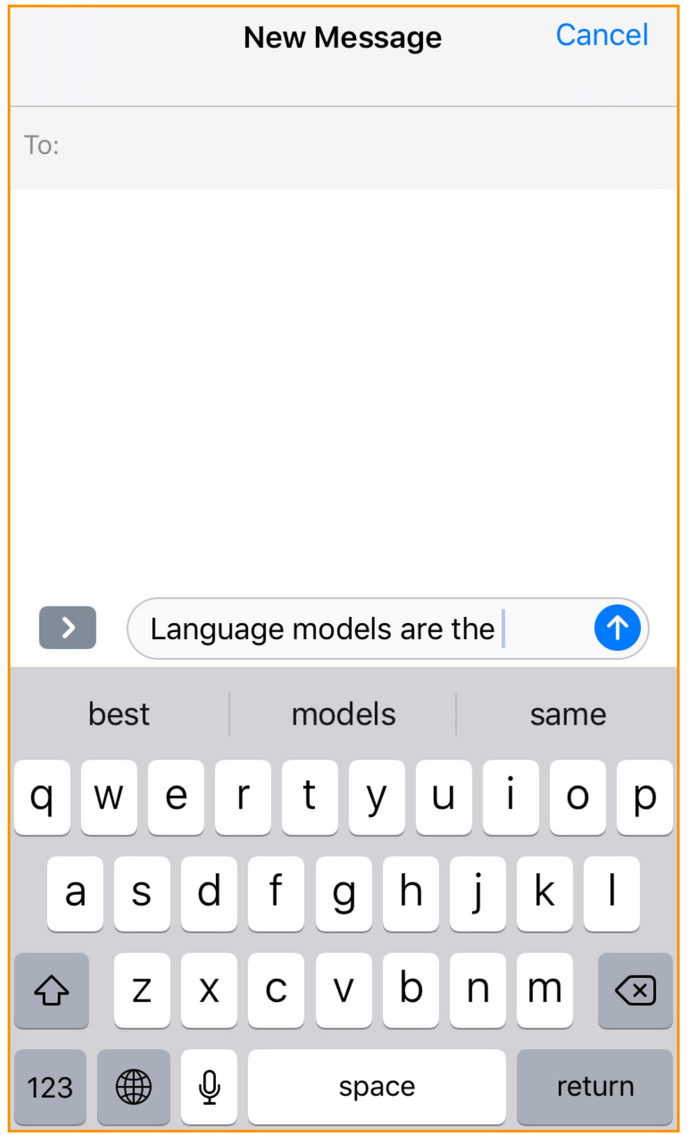
$$\arg\max_e P(e \mid f) = \arg\max_e P(f \mid e)P(e)$$

Caption generation noisy channel model



$$\arg\max_e P(e \mid i) = \arg\max_e P(i \mid e)P(e)$$

Language models are in production!



The Language Modeling task

- Setup: Assume a (finite) vocabulary of words, (infinite) set of sentences.

$$\mathcal{V} = \{\text{the, a, man, telescope, Beckham, two, Madrid, ...}\}$$

$$\mathcal{V}^\dagger = \{\text{the, a, the a, the fan, the man, the man with the telescope, ...}\}$$

- Data: a training set of example sentences
- Task: estimate a probability distribution over sentences

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1$$

and $p(x) \geq 0$ for all $x \in \mathcal{V}^\dagger$

$p(\text{the}) = 10^{-12}$

$p(\text{a}) = 10^{-13}$

$p(\text{the fan}) = 10^{-12}$

$p(\text{the fan saw Beckham}) = 2 \times 10^{-8}$

$p(\text{the fan saw saw}) = 10^{-15}$

...

Building Language Model

- Goal: Assign useful probabilities P(x) to sentences x
  - Input: many observations of training sentences x
  - Output: system capable of computing P(x)

- Simplest option: empirical distribution over the training sentences

$$p(\mathbf{x}) = \frac{\mathbf{count}(\mathbf{x})}{N}$$

- Decompose the probability: chain rule

$$P(\mathbf{x}) = P(x_1)P(x_2\,|\,x_1)P(x_3\,|\,x_2, x_1)\ldots$$

$$P(\text{the cat sat on the mat}) = P(\text{the}) * P(\text{cat}|\text{the}) * P(\text{sat}|\text{the cat})$$
$$*P(\text{on}|\text{the cat sat}) * P(\text{the}|\text{the cat sat on})$$
$$*P(\text{mat}|\text{the cat sat on the})$$

Problem with long context

		N-gram
$P(w\|\text{to})$	*to* occurs 10M times in corpus	1
$P(w\|\text{go to})$	*go to* occurs 100,000 times in corpus	2
$P(w\|\text{to go to})$	*go to* occurs 10,000 times in corpus	3
$P(w\|\text{want to go to})$	*want to go to*: only 100 occurrences	4

- Probability counts get very sparse, and we often want information from 5+ words away

This Lecture

- Language modeling
  - Why should we care?
  - N-gram models
  - Evaluating language models
  - Preview: Neural language model

Simplest Model: Unigram

- Generative process: pick a word, pick a word, … until you pick STOP

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i)$$

[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]

[]

[after, any, on, consistently, hospital, lake, of, of, other, and, factors, raised, analyst, too, allowed, mexico, never, consider, fall, bungled, davison, that, obtain, price, lines, the, to, sass, the, the, further, board, a, details, machinists, the, companies, which, rivals, an, because, longer, oakes, percent, a, they, three, edward, it, currier, an, within, in, three, wrote, is, you, s., longer, institute, dentistry, pay, however, said, possible, to, rooms, hiding, eggs, approximate, financial, canada, the, so, workers, advancers, half, between, nasdaq]

- Big Problem with Unigrams: $P(\text{the the the the}) >> P(\text{I like ice cream})$!

Bigram Model

- Generative process: pick START, pick a word conditioned on previous one, repeat until to pick STOP

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i \mid x_{i-1})$$

  - [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]
  - [although, common, shares, rose, forty, six, point, four, hundred, dollars, from, thirty, seconds, at, the, greatest, play, disingenuous, to, be, reset, annually, the, buy, out, of, american, brands, vying, for, mr., womack, currently, sharedata, incorporated, believe, chemical, prices, undoubtedly, will, be, as, much, is, scheduled, to, conscientious, teaching]
  - [this, would, be, a, record, november]

- Any disadvantage compared to unigram model?

N-Gram language model

- n-gram models: distribution of next word is a multinomial conditioned on previous n-1 words

$$p(x_i \mid x_1, \ldots x_{i-1}) = P(x_i \mid x_{i-n+1} \ldots x_{i-1})$$

Unigram $$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i)$$

Bigram $$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i \mid x_{i-1})$$

and Trigram, 4-gram, and so on.

Markov Assumption

$$p(\text{english} \mid \text{this is really written in}) \approx$$
$$p(\text{english} \mid \text{is really written in}) \approx$$
$$p(\text{english} \mid \text{really written in}) \approx$$
$$p(\text{english} \mid \text{written in}) \approx$$
$$p(\text{english} \mid \text{in}) \approx$$
$$p(\text{english})$$

N-gram Language Model: Parameter Estimation

- Maximum Likelihood Estimation!

$$q_{ML}(w) = \frac{c(w)}{c()}, \quad q_{ML}(w|v) = \frac{c(v,w)}{c(v)}, \quad q_{ML}(w|u,v) = \frac{c(u,v,w)}{c(u,v)}, \quad \ldots$$

Training Counts

198015222 the first
194623024 the same
168504105 the following
158562063 the world
…
14112454 the door

23135851162 the *

$$q(\text{door}|\text{the}) = \frac{14112454}{2313581162}$$

$$= 0.0006$$

Handling Low Frequency Contexts

- Add-one smoothing (Laplace Smoothing):

$$P(x_i \mid x_{i-1}) = \frac{C(x_{i-1}, x_i)}{C(x_{i-1})} \qquad P(x_i \mid x_{i-1}) = \frac{C(x_{i-1}, x_i) + 1}{C(x_{i-1}) + |V|}$$

Language Model Toy Example: Berkeley Restaurant Corpus

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day

Raw Bigram Counts

- From 9222 sentences

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Raw Bigram Probabilities

- Normalize by unigrams:

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

- Results:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Add-1 Smoothing

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

Add-1 smoothed bigrams

$$P_{\text{Add-1}}(x_i \mid x_{i-1}) = \frac{c(x_{i-1}, x_i) + 1}{c(x_{i-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Reconstructed Counts

$$P_{\text{Add-1}}(x_i \mid x_{i-1}) = \frac{c(x_{i-1}, x_i) + 1}{c(x_{i-1}) + V}$$

$$c^*(x_{i-1}, x_i) = \frac{(c(x_{i-1}, x_i) + 1)c(x_{i-1})}{c(x_{i-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Original vs. Re-constructed

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Handling Low Frequency Contexts

- Add-one smoothing (Laplace Smoothing):

$$P(x_i \mid x_{i-1}) = \frac{C(x_{i-1}, x_i)}{C(x_{i-1})} \qquad P(x_i \mid x_{i-1}) = \frac{C(x_{i-1}, x_i) + 1}{C(x_{i-1}) + |V|}$$

Problem? This will vastly overestimated the fraction of rare sequence

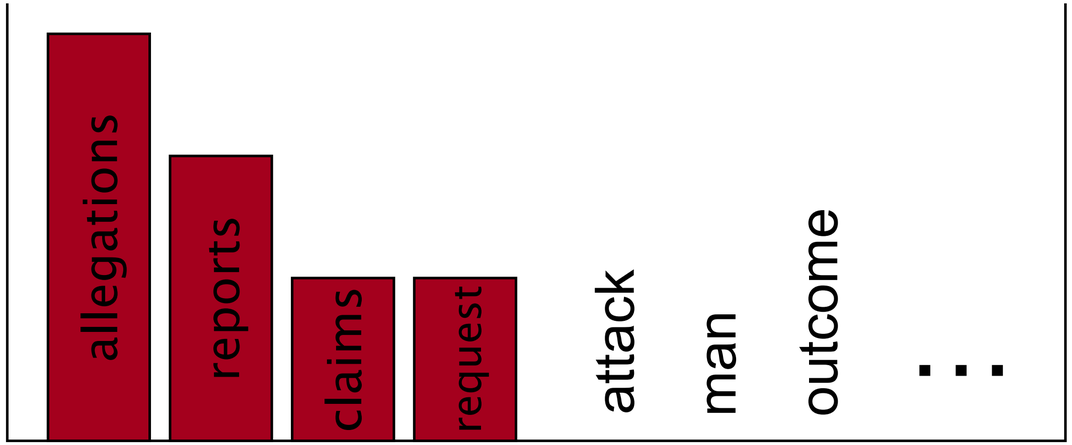
- Add-k smoothing:

$$P(x_i \mid x_{i-1}) = \frac{C(x_{i-1}, x_i) + \alpha}{C(x_{i-1}) + \alpha |V|}$$

Smoothing

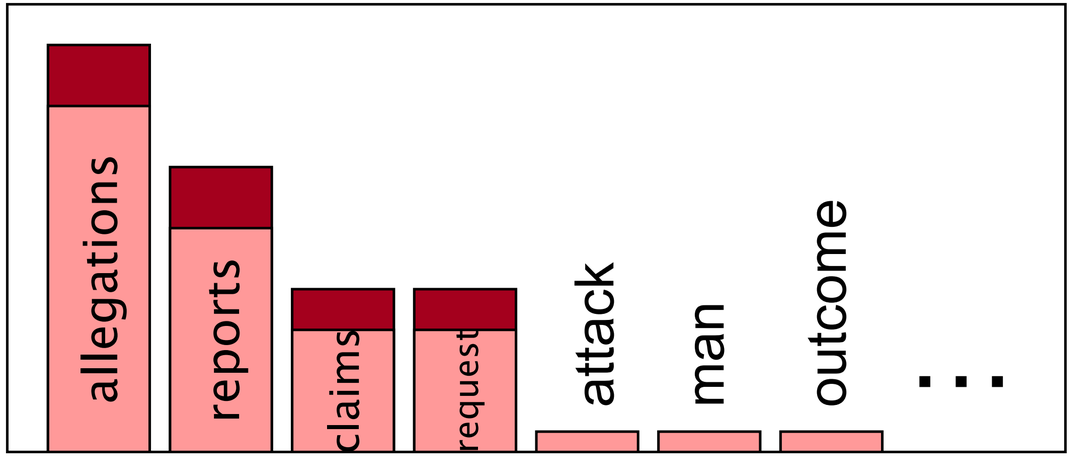
When we have sparse statistics:

P(w | denied the)
3 allegations
2 reports
1 claims
1 request
7 total



Steal probability mass to generalize better

P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other
7 total



Slide credit: Dan Klein

Handling Low Frequency Contexts

- Interpolation (use combination of different n-grams)

$$\hat{P}(x_i \mid x_{i-1}, x_{i-2}) = \lambda_1 P(x_i \mid x_{i-1}, x_{i-2}) + \lambda_2 P(x_i \mid x_{i-1}) + \lambda_3 P(x_i)$$

$$\sum_i \lambda_i = 1$$

How to choose lambdas?

- First, estimate n-gram probability on training set
- Then, select lambdas (hyper parameters) to maximize probability on the validation set

Large-scale N-grams

All Our N-gram are Belong to You

Thursday, August 03, 2006

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,

their computing resources, to play together. That's why we decided to share this enormous dataset with everyone. 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.

- serve as the incoming 92
- serve as the incubator 99
- serve as the independent 794
- serve as the index 223
- serve as the indication 72
- serve as the indicator 120
- serve as the indicators 45
- serve as the indispensable 111
- serve as the indispensible 40
- serve as the individual 234

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

Handling unknown words

- If we know all the words in advanced
  - Vocabulary V is fixed, closed vocabulary task!
- Often we don't know this
  - Out Of Vocabulary = OOV words
  - Open vocabulary task

- Instead: create an unknown word token <UNK>
  - Training of <UNK> probabilities
    - Create a fixed lexicon L of size V (e.g., rare words are not in L)
    - At text normalization phase, any training word not in L changed to <UNK>
    - Now we train its probabilities like a normal word
  - At decoding time
    - If text input: Use UNK probabilities for any word not in training

Evaluating Language Model

Language Model Evaluation

- What we would like:
  - Would the model prefer good sentences to bad ones?
  - Bad ≠ ungrammatical!
  - Bad ≈ unlikely

Measuring the Model Quality

- The Shannon Game:
  - How well can we predict the next word?

    When I eat pizza, I wipe off the ____

    Many children are allergic to ____

    I saw a ____

grease 0.5
sauce 0.4
dust 0.05
….
mice 0.0001
….
the 1e-100

Claude Shannon

- How good are we doing?
- Compute per word log likelihood (total n words):

$$l = \frac{1}{n}\sum_{i=1}^{n} \log P(x_i \,|\, x_1, x_2 \ldots x_{i-1})$$

Intrinsic Measure: Perplexity

- Evaluate LMs on the ***log likelihood*** of held-out data (averaged to normalize for length)

$$l = \frac{1}{n}\sum_{i=1}^{n} \log P(x_i \,|\, x_1, x_2 \ldots x_{i-1})$$

- Perplexity: **Lower is better!**

$$PP = 2^{-l}$$

Shannon Game intuition for perplexity

- How hard is the task of recognizing digits ‘0,1,2,3,4,5,6,7,8,9’ at random
  - Perplexity 10
- How hard is recognizing (30,000) names at random
  - Perplexity = 30,000
- If a system has to recognize
  - Operator (1 in 4)
  - Sales (1 in 4)
  - Technical Support (1 in 4)
  - 30,000 names (1 in 120,000 each)
  - Perplexity is 53
- Perplexity is weighted equivalent branching factor

$$PP = 2^{-\frac{1}{M}\sum_{i=1}^{m}\log_2(\frac{1}{10})^{|X^{(i)}|}}$$

$$= 2^{-\frac{1}{M}\sum_{i=1}^{m}|X^{(i)}|\log_2\frac{1}{10}}$$

$$= 2^{-\log_2\frac{1}{10}} = 2^{-\log_2 10^{-1}} = 10$$