TTIC 31190: Natural Language Processing

Kevin Gimpel

Winter 2016

Lecture 7: Sequence Models

Announcements

- Assignment 2 has been posted, due Feb. 3
- Midterm scheduled for Thursday, Feb. 18
- Project proposal due Tuesday, Feb. 23
- Thursday's class will be more like a lab / flipped class
  - we will use the whiteboard and implement things in class, so bring paper, laptop, etc.

Roadmap

- classification
- words
- lexical semantics
- language modeling
- sequence labeling
- syntax and syntactic parsing
- neural network methods in NLP
- semantic compositionality
- semantic parsing
- unsupervised learning
- machine translation and other applications

Language Modeling

- goal: compute the probability of a sequence of words:

$$p(w_1 ... w_n) = \prod_{i=1}^{n} p(w_i \mid w_1 ... w_{i-1})$$

Markov Assumption for Language Modeling

Andrei Markov

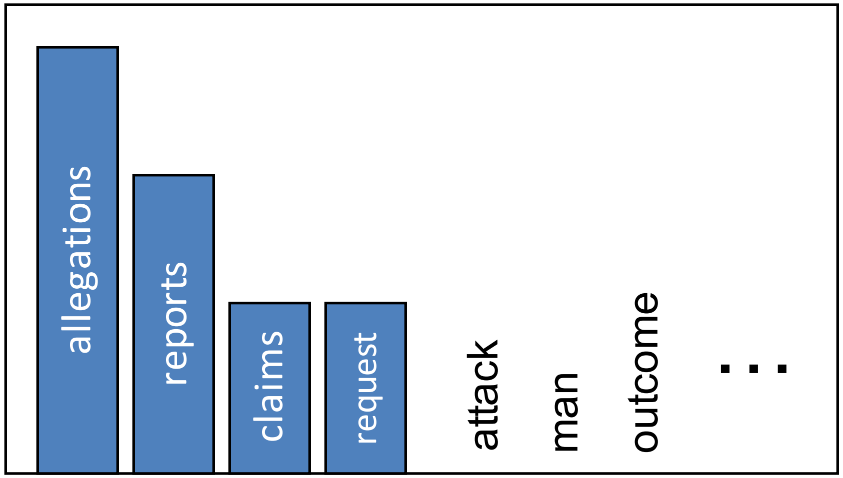
$$p(w_1 \ldots w_n) = \prod_{i=1}^{n} p(w_i \mid w_1 \ldots w_{i-1})$$

$$p(w_1 \ldots w_n) = \prod_{i=1}^{n} p(w_i \mid w_{i-k} \ldots w_{i-1})$$

Intuition of smoothing (from Dan Klein)

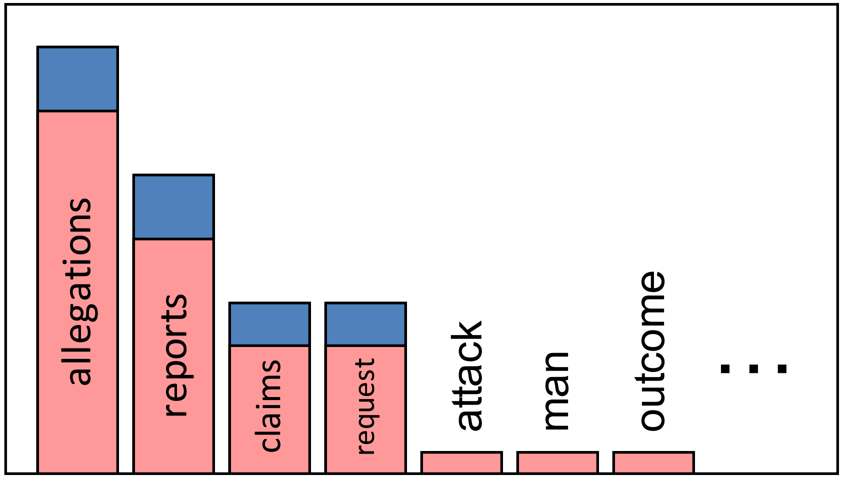
- When we have sparse statistics:

  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



“Add-1” estimation

- also called Laplace smoothing
- just add 1 to all counts!

Backoff and Interpolation

- sometimes it helps to use **less** context
  - condition on less context for contexts you haven't learned much about
- **backoff**:
  - use trigram if you have good evidence, otherwise bigram, otherwise unigram
- **interpolation**:
  - mixture of unigram, bigram, trigram (etc.) models

- interpolation works better

J&M/SLP3

Linear Interpolation

- simple interpolation:

$$\begin{aligned} \hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1 P(w_n|w_{n-2}w_{n-1}) \\ &+\lambda_2 P(w_n|w_{n-1}) \\ &+\lambda_3 P(w_n) \end{aligned} \qquad \sum_i \lambda_i = 1$$

Kneser-Ney Smoothing

- better estimate for probabilities of lower-order unigrams!
  - Shannon game: *I can't see without my reading*____________?
  - "*Francisco*" is more common than "*glasses*"
  - … but "*Francisco*" always follows "*San*"
- unigram is most useful when we haven't seen bigram!
- so instead of unigram $P(w)$ ("How likely is w?")
- use $P_{\text{continuation}}(w)$ ("How likely is w to appear as a novel continuation?")
  - for each word, count # of bigram types it completes:

$$P_{CONTINUATION}(w) \propto \left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|$$

J&M/SLP3

Kneser-Ney Smoothing

- how many times does *w* appear as a novel continuation?

$$P_{CONTINUATION}(w) \propto \left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|$$

- normalize by total number of
  word bigram types: $\left|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}\right|$

$$P_{CONTINUATION}(w) = \frac{\left|\{w_{i-1} : c(w_{i-1}, w) > 0\}\right|}{\left|\{(w_{j-1}, w_j) : c(w_{j-1}, w_j) > 0\}\right|}$$

J&M/SLP3

N-gram Smoothing Summary

- add-1 estimation:
  - OK for text categorization, not for language modeling
- for very large N-gram collections like the Web:
  - stupid backoff
- most commonly used method:
  - modified interpolated Kneser-Ney

Roadmap

- classification
- words
- lexical semantics
- language modeling
- sequence labeling
- syntax and syntactic parsing
- neural network methods in NLP
- semantic compositionality
- semantic parsing
- unsupervised learning
- machine translation and other applications

Linguistic phenomena: summary so far…

- words have structure (stems and affixes)
- words have multiple meanings (senses) → word sense ambiguity
  - senses of a word can be homonymous or polysemous
  - senses have relationships:
    - hyponymy ("is a")
    - meronymy ("part of", "member of")
- variability/flexibility of linguistic expression
  - many ways to express the same meaning (as you saw in Assignment 1)
  - word vectors tell us when two words are similar
- today: **part-of-speech**

Part-of-Speech Tagging

Some questioned if Tim Cook ’s first product

would be a breakaway hit for Apple .

Part-of-Speech Tagging

determiner	verb (past)	prep.	proper noun	proper noun	poss.	adj.	noun
Some	questioned	if	Tim	Cook	’s	first	product
modal	verb	det.	adjective	noun	prep.	proper noun	punc.
would	be	a	breakaway	hit	for	Apple	.

Part-of-Speech (POS)

- functional category of a word:
  - noun, verb, adjective, etc.
  - how is the word functioning in its context?
- dependent on context like word sense, but different from sense:
  - sense represents word meaning, POS represents word function
  - sense uses a distinct category of senses per word, POS uses same set of categories for all words

Penn Treebank tag set

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

Universal Tag Set

- many use smaller sets of coarser tags
- e.g., "universal tag set" containing 12 tags:
  - noun, verb, adjective, adverb, pronoun, determiner/article, adposition (preposition or postposition), numeral, conjunction, particle, punctuation, other

sentence:	The	oboist	Heinz	Holliger	has	taken	a	hard	line	about	the	problems	.
original:	DT	NN	NNP	NNP	VBZ	VBN	DT	JJ	NN	IN	DT	NNS	.
universal:	DET	NOUN	NOUN	NOUN	VERB	VERB	DET	ADJ	NOUN	ADP	DET	NOUN	.

Figure 1: Example English sentence with its language specific and corresponding universal POS tags.

Petrov, Das, McDonald (2011)

Twitter Part-of-Speech Tagging

other verb article noun pronoun

intj pronoun prep adj prep verb

ikr smh he asked fir yo last name so he can

add u on fb lololol =D #lolz

verb prep intj emoticon hashtag

pronoun proper noun

adj = adjective
prep = preposition
intj = interjection

- we removed some fine-grained POS tags, then added Twitter-specific tags:

  hashtag

  @-mention

  URL / email address

  emoticon

  Twitter discourse marker

  other (multi-word abbreviations, symbols, garbage)

word sense vs. part-of-speech

	word sense	part-of-speech
semantic or syntactic?	semantic: indicates meaning of word in its context	syntactic: indicates function of word in its context
number of categories	$\|V\|$ words, ~5 senses each → $5\|V\|$ categories!	typical POS tag sets have 12 to 45 tags
inter-annotator agreement	low; some sense distinctions are highly subjective	high; relatively few POS tags and function is relatively shallow / surface-level
independent or joint classification of nearby words?	independent: can classify a single word based on context words; structured prediction is rarely used	joint: strong relationship between tags of nearby words; structured prediction often used

How might POS tags be useful?

- text classification
- machine translation
- question answering

Classification Framework

inference: solve argmax

modeling: define score function

$$\text{classify}(x, \boldsymbol{\theta}) = \operatorname*{argmax}_{y} \ \text{score}(x, y, \boldsymbol{\theta})$$

learning: choose $\boldsymbol{\theta}$

Applications of our Classification Framework

text classification:

$$\text{classify}^{\text{linear}}_{\text{text}}(\boldsymbol{x}, \boldsymbol{\theta}) = \operatorname*{argmax}_{y \in \mathcal{L}} \sum_i \theta_i f_i(\boldsymbol{x}, y)$$

$\mathcal{L}$ = {objective, subjective}

x	*y*
the hulk is an anger fueled monster with incredible strength and resistance to damage .	objective
in trying to be daring and original , it comes off as only occasionally satirical and never fresh .	subjective

Applications of our Classification Framework

word sense classifier for *bass*:

$$\text{classify}^{\text{linear}}_{\text{bassWSD}}(\boldsymbol{x}, \boldsymbol{\theta}) = \operatorname*{argmax}_{y \in \mathcal{L}_{\text{bass}}} \sum_i \theta_i f_i(\boldsymbol{x}, y)$$

$\mathcal{L}_{\text{bass}}$ = {*bass*$_1$, *bass*$_2$, …, *bass*$_8$}

x	*y*
he's a bass in the choir .	*bass*$_3$
our bass is line-caught from the Atlantic .	*bass*$_4$

- S: (n) **bass** (the lowest part of the musical ra
- S: (n) **bass**, bass part (the lowest part in pol
- **S: (n) bass, basso (an adult male singer w**
- S: (n) sea bass, **bass** (the lean flesh of a salt Serranidae)
- S: (n) freshwater bass, **bass** (any of various with lean flesh (especially of the genus Micr
- S: (n) **bass**, bass voice, basso (the lowest ad
- S: (n) **bass** (the member with the lowest ran instruments)
- S: (n) **bass** (nontechnical name for any of nu freshwater spiny-finned fishes)

Applications of our Classification Framework

skip-gram model as a classifier:

$$\text{classify}_{\text{skipgram}}(x, \boldsymbol{\theta}) = \underset{y \in \mathcal{L}}{\text{argmax}} \ \ \boldsymbol{\theta}^{(\text{in},x)} \cdot \boldsymbol{\theta}^{(\text{out},y)}$$

$\mathcal{L}$ = V (the entire vocabulary)

x	*y*
agriculture	<s>
agriculture	is
agriculture	the

corpus (English Wikipedia):

agriculture is the traditional mainstay of the cambodian economy .

but benares has been destroyed by an earthquake .

…

Applications of our Classifier Framework so far

task	input (x)	output (y)	output space ($\mathcal{L}$)	size of $\mathcal{L}$
text classification	a sentence	gold standard label for x	pre-defined, small label set (e.g., {positive, negative})	2-10
word sense disambiguation	instance of a particular word (e.g., *bass*) with its context	gold standard word sense of x	pre-defined sense inventory from WordNet for *bass*	2-30
learning skip-gram word embeddings	instance of a word in a corpus	a word in the context of x in a corpus	vocabulary	$\|V\|$
part-of-speech tagging	a sentence	gold standard part-of-speech tags for x	all possible part-of-speech tag sequences with same length as x	$\|P\|^{\|x\|}$

Applications of our Classifier Framework so far

task	input (x)	output (y)	output space ($\mathcal{L}$)	size of $\mathcal{L}$
text classification	a sentence	gold standard label for x	pre-defined, small label set (e.g., {positive, negative})	2-10
word sense disambiguation	instance of a particular word (e.g., *bass* … its cont…	gold standard …	pre-defined sense inventory from …	2-30
learning skip-gram word embeddings	instance … word in a c…	…	…	…
part-of-speech tagging	a sentence	gold standard part-of-speech tags for x	all possible part-of-speech tag sequences with same length as x	$\lvert P \rvert^{\lvert x \rvert}$

exponential in size of input!
"structured prediction"

Simplest kind of structured prediction: Sequence Labeling

Part-of-Speech Tagging

determiner	verb (past)	prep.	proper noun	proper noun	poss.	adj.	noun
Some	questioned	if	Tim	Cook	’s	first	product

modal	verb	det.	adjective	noun	prep.	proper noun	punc.
would	be	a	breakaway	hit	for	Apple	.

Named Entity Recognition

Some questioned if Tim Cook’s first product would be a breakaway hit for Apple.

PERSON

ORGANIZATION

Learning

$$\text{classify}(x, \boldsymbol{\theta}) = \operatorname*{argmax}_{y} \; \text{score}(x, y, \boldsymbol{\theta})$$

learning: choose $\boldsymbol{\theta}$

Empirical Risk Minimization with Surrogate Loss Functions

- given training data: $\mathcal{T} = \{\langle \boldsymbol{x}^{(i)}, y^{(i)} \rangle\}_{i=1}^{|\mathcal{T}|}$
  where each $y^{(i)} \in \mathcal{L}$ is a label
- we want to solve the following:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{|\mathcal{T}|} \operatorname{loss}(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

many possible loss functions to consider optimizing

Loss Functions

name	loss	where used
cost ("0-1")	$\mathrm{cost}(y, \mathrm{classify}(\boldsymbol{x}, \boldsymbol{\theta}))$	intractable, but underlies "direct error minimization"
perceptron	$-\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} \mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta})$	perceptron algorithm (Rosenblatt, 1958)
hinge	$-\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \max_{y' \in \mathcal{L}} (\mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \mathrm{cost}(y, y'))$	support vector machines, other large-margin algorithms
log	$-\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x})$ $= -\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta}) + \log \sum_{y' \in \mathcal{L}} \exp\{\mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta})\}$	logistic regression, conditional random fields, maximum entropy models

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) = \frac{\exp\{\mathrm{score}(\boldsymbol{x}, y, \boldsymbol{\theta})\}}{\sum_{y' \in \mathcal{L}} \exp\{\mathrm{score}(\boldsymbol{x}, y', \boldsymbol{\theta})\}}$$

(Sub)gradients of Losses for Linear Models

name	entry *j* of (sub)gradient of loss for linear model
cost ("0-1")	not subdifferentiable in general
perceptron	$-f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \hat{y}), \text{ where } \hat{y} = \text{classify}(\boldsymbol{x}, \boldsymbol{\theta})$
hinge	$-f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \tilde{y}), \text{ where } \tilde{y} = \text{costClassify}(\boldsymbol{x}, y, \boldsymbol{\theta})$
log	

$$\text{classify}(\boldsymbol{x}, \boldsymbol{\theta}) = \operatorname*{argmax}_{y' \in \mathcal{L}} \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta})$$

whatever loss is used during training, **classify** (**NOT costClassify**) is used to predict labels for dev/test data!

$$\text{costClassify}(\boldsymbol{x}, y, \boldsymbol{\theta}) = \operatorname*{argmax}_{y' \in \mathcal{L}} \text{score}(\boldsymbol{x}, y', \boldsymbol{\theta}) + \text{cost}(y, y')$$

(Sub)gradients of Losses for Linear Models

name	**entry *j* of (sub)gradient of loss for linear model**
cost ("0-1")	not subdifferentiable in general
perceptron	$-f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \hat{y}), \text{ where } \hat{y} = \text{classify}(\boldsymbol{x}, \boldsymbol{\theta})$
hinge	$-f_j(\boldsymbol{x}, y) + f_j(\boldsymbol{x}, \tilde{y}), \text{ where } \tilde{y} = \text{costClassify}(\boldsymbol{x}, y, \boldsymbol{\theta})$
log	$-f_j(\boldsymbol{x}, y) + \mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x})}[f_j(\boldsymbol{x}, \cdot)]$

expectation of feature value with respect to distribution over *y* (where distribution is defined by theta)

alternative notation:

$$-f_j(\boldsymbol{x}, y) + \mathbb{E}_{y' \sim p_{\boldsymbol{\theta}}(Y \mid \boldsymbol{x})}[f_j(\boldsymbol{x}, y')]$$

Sequence Models

- models that assign scores (could be probabilities) to sequences
- general category that includes many models used widely in practice:
  - *n*-gram language models
  - hidden Markov models
  - "chain" conditional random fields
  - maximum entropy Markov models

Hidden Markov Models (HMMs)

- HMMs define a joint probability distribution over input sequences ***x*** and output sequences ***y***:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y})$$

- conditional independence assumptions ("Markov assumption") are used to factorize this joint distribution into small terms
- widely used in NLP, speech recognition, bioinformatics, many other areas

Hidden Markov Models (HMMs)

- HMMs define a joint probability distribution over input sequences ***x*** and output sequences ***y***:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y})$$

- assumption: output sequence ***y*** "generates" input sequence ***x***:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{|\boldsymbol{x}|} p(y_i \mid y_1, y_2, \ldots, y_{i-1}) p(x_i \mid y_1, y_2, \ldots, y_i)$$

- these are too difficult to estimate, let's use Markov assumptions

Markov Assumption for Language Modeling

Andrei Markov

$$p(w_1 \ldots w_n) = \prod_{i=1}^{n} p(w_i \mid w_1 \ldots w_{i-1})$$

$$p(w_1 \ldots w_n) = \prod_{i=1}^{n} p(w_i \mid w_{i-k} \ldots w_{i-1})$$

trigram model:

$$p(w_1 \ldots w_n) = \prod_{i=1}^{n} p(w_i \mid w_{i-2} w_{i-1})$$

Independence and Conditional Independence

- **Independence**: two random variables *X* and *Y* are independent if:

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

(or $P(x, y) = P(x)P(y)$)

for all values *x* and *y*

- **Conditional Independence**: two random variables *X* and *Y* are conditionally independent given a third variable *Z* if

$$P(x, y \mid z) = P(x \mid z)P(y \mid z)$$

for all values of *x*, *y*, and *z*

(or $P(x \mid y, z) = P(x \mid z)$)

Markov Assumption for Language Modeling

Andrei Markov

$$p(w_1 ... w_n) = \prod_{i=1}^{n} p(w_i \mid w_1 ... w_{i-1})$$

trigram model:

$$p(w_1 ... w_n) = \prod_{i=1}^{n} p(w_i \mid w_{i-2} w_{i-1})$$

$$w_i \perp w_{i-3} \mid w_{i-2}, w_{i-1}$$

Conditional Independence Assumptions of HMMs

- two *y*'s are conditionally independent given the *y*'s between them:

$$y_i \perp y_{i-2} \mid y_{i-1}$$

- an *x* at position *i* is conditionally independent of other *y*'s given the *y* at position *i*:

$$x_i \perp y_{i-1} \mid y_i$$

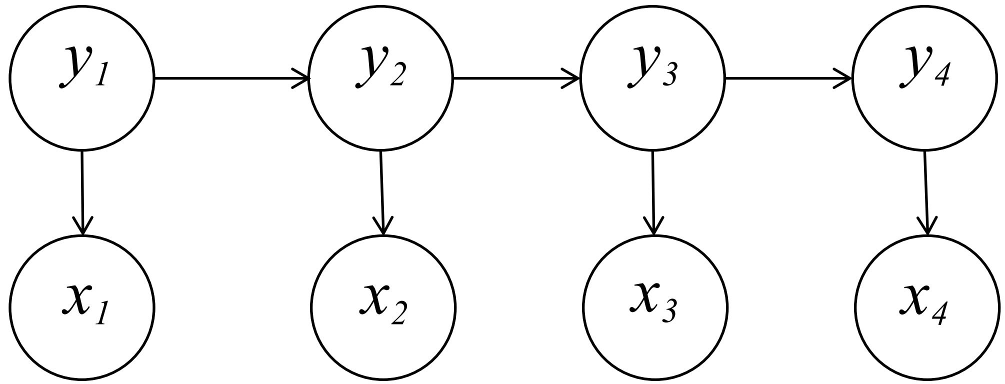
$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{|\boldsymbol{x}|} p(y_i \mid y_1, y_2, \ldots, y_{i-1}) p(x_i \mid y_1, y_2, \ldots, y_i)$$

$$\Downarrow$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{|\boldsymbol{x}|} p_{\boldsymbol{\tau}}(y_i \mid y_{i-1}) \; p_{\boldsymbol{\eta}}(x_i \mid y_i)$$

Graphical Model for an HMM

(for a sequence of length 4)



a graphical model is a graph in which:

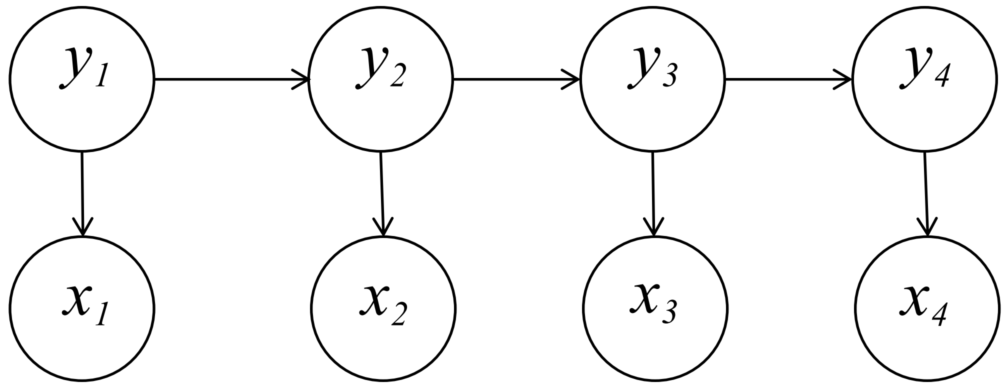
each node corresponds to a random variable

each directed edge corresponds to a conditional probability distribution of the target node given the source node

conditional independence statements among random variables are encoded by the edge structure

Graphical Model for an HMM

(for a sequence of length 4)



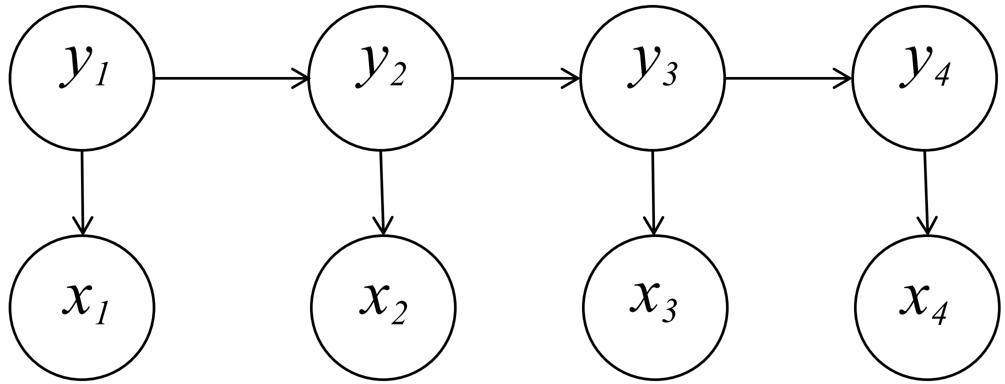
conditional independence statements among random variables are encoded by the edge structure → we only have to worry about **local distributions:**

transition parameters: $p_{\boldsymbol{\tau}}(y_i \mid y_{i-1})$

emission parameters: $p_{\boldsymbol{\eta}}(x_i \mid y_i)$

Graphical Model for an HMM

(for a sequence of length 4)



$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{|\boldsymbol{x}|} p_{\boldsymbol{\tau}}(y_i \mid y_{i-1}) \, p_{\boldsymbol{\eta}}(x_i \mid y_i)$$

transition parameters: $p_{\boldsymbol{\tau}}(y_i \mid y_{i-1})$

emission parameters: $p_{\boldsymbol{\eta}}(x_i \mid y_i)$

"Brown Clustering"

Class-Based *n*-gram Models of Natural Language

Peter F. Brown*
Peter V. deSouza*
Robert L. Mercer*
IBM T. J. Watson Research Center

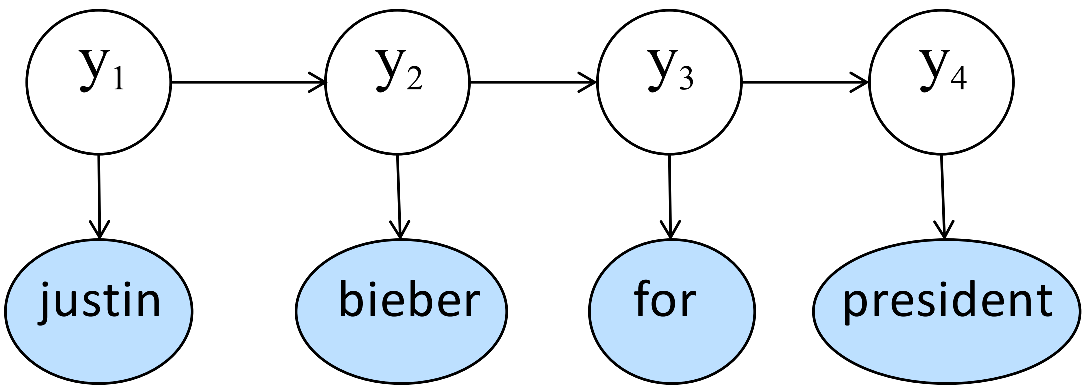
Vincent J. Della Pietra*
Jenifer C. Lai*

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays
June March July April January December October November September August
people guys folks fellows CEOs chaps doubters commies unfortunates blokes
down backwards ashore sideways southward northward overboard aloft downwards adrift
water gas coal liquid acid sand carbon steam shale iron
great big vast sudden mere sheer gigantic lifelong scant colossal

Computational Linguistics, 1992

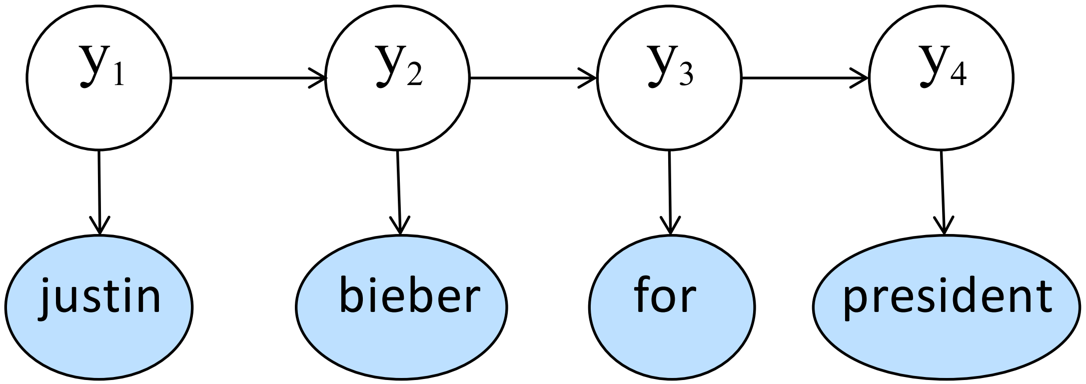
Brown Clustering (Brown et al., 1992)

hidden Markov model with one-cluster-per-word constraint



Brown Clustering (Brown et al., 1992)

hidden Markov model with one-cluster-per-word constraint

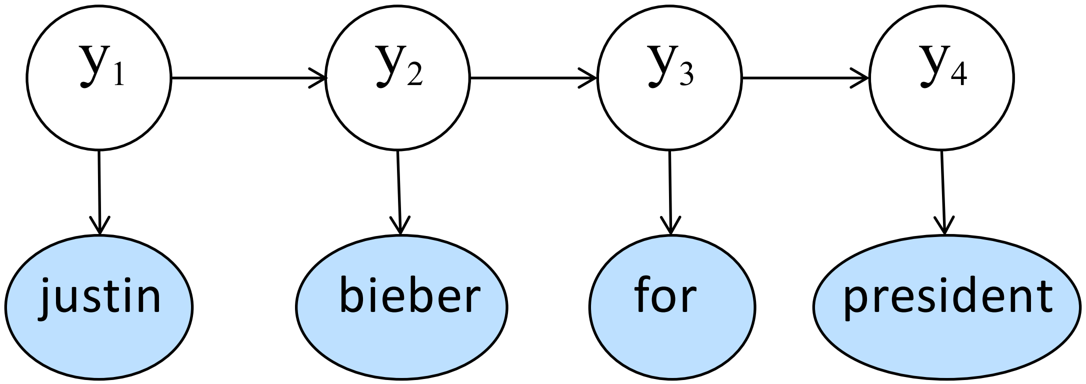


algorithm:

- initialize each word as its own cluster
- greedily merge clusters to improve data likelihood

Brown Clustering (Brown et al., 1992)

hidden Markov model with one-cluster-per-word constraint



algorithm:

- initialize each word as its own cluster
- greedily merge clusters to improve data likelihood

outputs **hierarchical** clustering

we induced 1000 Brown clusters from 56 million English tweets (1 billion words)

only words that appeared at least 40 times

(Owoputi, O'Connor, Dyer, Gimpel, Schneider, and Smith, 2013)

Example Cluster

missed loved hated misread admired underestimated resisted adored disliked regretted missd fancied luved prefered luvd overdid mistyped misd misssed looooved misjudged lovedd loooved loathed lurves lovd

Example Cluster

missed loved hated misread admired
underestimated resisted adored disliked
regretted missd fancied luved prefered luvd
overdid mistyped misd misssed looooved
misjudged lovedd loooved loathed lurves lovd

spelling
variation

“really”

really rly realy genuinely rlly reallly realllly reallyy rele realli relly reallllly reli reali sholl rily reallyyy reeeeally realllllly reaally reeeally rili

“really”

really rly realy genuinely rlly reallly realllly
reallyy rele realli relly reallllly reli reali sholl rily
reallyyy reeeeeally realllllly reaally reeeally rili
reaaally reaaaally reallyyyy rilly reallllllly
reeeeeeally reeally shol realllyyy reely relle
reaaaaally shole really2 reallyyyyy _really_
realllllllly reaaly realllyy reallii reallt genuinly relli
realllyyyy reeeeeeeally weally reaaallly reallllyyy
reallllllllly reaallly realyy /really/ reaaaaaally

“really”

really rly realy genuinely rlly reallly realllly
reallyy rele realli relly reallllly reli reali sholl rily
reallyyy reeeeally realllllly reaally reeeally rili
reaaally reaaaally reallyyyy rilly reallllllly
reeeeeally reeally shol realllyyy reely relle
reaaaaally shole really2 reallyyyyy _really_
realllllllly reaaly realllyy reallii reallt genuinly relli
realllyyyy reeeeeeally weally reaaallly reallllyyy
reallllllllly reaallly realyy /really/ reaaaaaally reallu
reaaaallly reeaally rreally reallyreally eally reeeaaally reeeaaally
reaallyy reallyyyyyy –really- reallyreallyreally rilli reallllyyyy relaly
reallllyy really-really r3ally reeli reallie realllllyyy rli realllllllllly
reaaaly reeeeeeeally

“going to”

gonna gunna gona gna guna gnna ganna qonna gonnna gana qunna gonne goona gonnaa g0nna goina gonnah goingto gunnah gonaa gonan gunnna going2 gonnnnagunnaa gonny gunaa quna goonna qona gonns goinna gonnae qnna gonnaaa gnaa

“so”

soo sooo soooo sooooo soooooo sooooooo
soooooooo sooooooooo soooooooooo
sooooooooooo soooooooooooo
sooooooooooooo soso soooooooooooooo
sooooooooooooooo soooooooooooooooo
sososo superrr sooooooooooooooooo ssooo
so0o superrrr so0 soooooooooooooooooo
sosososo sooooooooooooooooooo ssoo sssooo
soooooooooooooooooooo #too s0o ssoooo s00

Food-Related Adjectives

hot fried peanut homemade grilled spicy soy cheesy coconut veggie roasted leftover blueberry icy dunkin mashed rotten mellow boiling crispy peppermint fruity toasted crunchy scrambled creamy boiled chunky funnel soggy clam steamed cajun steaming chewy steamy nacho mince reese's shredded salted glazed spiced venti pickled powdered butternut miso beet sizzling

Adjective Intensifiers/Qualifiers

kinda hella sorta hecka kindof kindaa kinna hellla propa helluh kindda justa #slick helllla hela jii sortof hellaa kida wiggity hellllla hekka hellah kindaaa hellaaa kindah knda kind-of slicc wiggidy helllllla jih jye kinnda odhee kiinda heka sorda ohde kind've kidna baree rle hellaaaa jussa