

TTIC 31190: Natural Language Processing

Kevin Gimpel

Winter 2016

Lecture 9: Sequence Models

Announcements

- on Thursday, class will be in Room 530 (the room directly behind you)

Announcements

- we will go over part of Assignment 1 today (grades coming soon)
- Assignment 2 was due Wed. Feb. 3, now due Fri., Feb. 5
- project proposal due Tuesday, Feb. 16
- midterm on Thursday, Feb. 18

Other Naturally-Occurring Data

- quality of scientific journalism:

What Makes Writing Great? First Experiments on Article Quality Prediction in the Science Journalism Domain

Annie Louis

University of Pennsylvania
Philadelphia, PA 19104
lannie@seas.upenn.edu

Ani Nenkova

University of Pennsylvania
Philadelphia, PA 19104
nenkova@seas.upenn.edu

Abstract

Great writing is rare and highly admired. Readers seek out articles that are beautifully written, informative and entertaining. Yet information-access technologies lack capabilities for predicting article quality at this level. In this paper we present first experiments on article quality prediction in the science journalism domain. We introduce a corpus of great pieces of science journalism, along with typical articles from the genre. We imple-

done before. The fawn, known as Dewey, was developing normally and seemed to be healthy. He had no mother, just a surrogate who had carried his fetus to term. He had no father, just a “donor” of all his chromosomes. He was the genetic duplicate of a certain trophy buck out of south Texas whose skin cells had been cultured in a laboratory. One of those cells furnished a nucleus that, transplanted and rejiggered, became the DNA core of an egg cell, which became an embryo, which in time became Dewey. So he was wildlife, in a sense, and in another sense elaborately synthetic. This is the sort of news

Other Naturally-Occurring Data

- memorability of quotations:

You had me at hello: How phrasing affects memorability

Cristian Danescu-Niculescu-Mizil Justin Cheng Jon Kleinberg Lillian Lee

Department of Computer Science
Cornell University

cristian@cs.cornell.edu, jc882@cornell.edu, kleinber@cs.cornell.edu, llee@cs.cornell.edu

Abstract

Understanding the ways in which information achieves widespread public awareness is a research question of significant interest. We consider whether, and how, the way in which the information is phrased — the choice of words and sentence structure — can affect this process. To this end, we develop an analysis framework and build a corpus of movie quotes, annotated with memorability information, in which we are able to control for both the speaker and the setting of the quotes.

Building on a foundation in the sociology of diffusion [27, 31], researchers have explored the ways in which network structure affects the way information spreads, with domains of interest including blogs [1, 11], email [37], on-line commerce [22], and social media [2, 28, 33, 38]. There has also been recent research addressing temporal aspects of how different media sources convey information [23, 30, 39] and ways in which people react differently to information on different topics [28, 36].

Beyond all these factors, however, one's everyday

Other Naturally-Occurring Data

- sarcasm (remove #sarcasm hash tag from tweets):

Contextualized Sarcasm Detection on Twitter

David Bamman and Noah A. Smith

School of Computer Science

Carnegie Mellon University

{dbamman,nasmith}@cs.cmu.edu

Abstract

Sarcasm requires some shared knowledge between speaker and audience; it is a profoundly *contextual* phenomenon. Most computational approaches to sarcasm detection, however, treat it as a purely linguistic matter, using information such as lexical cues and their corresponding sentiment as predictive features. We show that by including extra-linguistic information from the context of an utterance on Twitter – such as properties of the author, the audience and the immediate communicative environment – we are able to achieve gains in accuracy compared to purely linguistic features in the detection of this complex phenomenon, while also shedding light on features of interpersonal interaction that enable sarcasm in conversation.

people who know each other well than between those who do not.

In all of these cases, the relationship between author and audience is central for understanding the sarcasm phenomenon. While the notion of an “audience” is relatively well defined for face-to-face conversations between two people, it becomes more complex when multiple people are present (Bell 1984), and especially so on social media, when a user’s “audience” is often unknown, underspecified or “collapsed” (boyd 2008; Marwick and boyd 2011), making it difficult to fully establish the shared ground required for sarcasm to be detected, and understood, by its intended (or imagined) audience.

We present here a series of experiments to discern the effect of extra-linguistic information on the detection of sarcasm, reasoning about features derived not only from the

Other Naturally-Occurring Data

- opening weekend movie revenue prediction from critic reviews:

Movie Reviews and Revenues: An Experiment in Text Regression*

Mahesh Joshi Dipanjan Das Kevin Gimpel Noah A. Smith

Language Technologies Institute

Carnegie Mellon University

Pittsburgh, PA 15213, USA

{maheshj, dipanjan, kgimpel, nasmith}@cs.cmu.edu

Abstract

We consider the problem of predicting a movie's opening weekend revenue. Previous work on this problem has used metadata about a movie—e.g., its genre, MPAA rating, and cast—with very limited work making use of text *about* the movie. In this paper, we use the text of film critics' reviews from several sources to predict opening weekend revenue. We describe a new dataset pairing movie reviews with metadata and revenue data, and show that review text can substitute for metadata, and even improve over it, for prediction.

correlation between actual revenue and sentiment-based metrics, as compared to mention counts of the movie. (They did not frame the task as a revenue prediction problem.) Zhang and Skiena (2009) used a news aggregation system to identify entities and obtain domain-specific sentiment for each entity in several domains. They used the aggregate sentiment scores and mention counts of each movie in news articles as predictors.

While there has been substantial prior work on using critics' reviews, to our knowledge all of this work has used polarity of the review or the number of stars given to it by a critic, rather than the review

Other Naturally-Occurring Data

- predicting novel success from text of novels:

Success with Style: Using Writing Style to Predict the Success of Novels

Vikas Ganjigunte Ashok Song Feng Yejin Choi

Department of Computer Science

Stony Brook University

Stony Brook, NY 11794-4400

`vganjiguntea, songfeng, ychoi@cs.stonybrook.edu`

Abstract

Predicting the success of literary works is a curious question among publishers and aspiring writers alike. We examine the quantitative connection, if any, between *writing style* and successful literature. Based on novels over several different genres, we probe the predictive power of statistical stylometry in discriminating successful literary works, and identify characteristic stylistic elements that are more prominent in successful writings. Our study

fore they are picked up by a publisher.¹

Perhaps due to its obvious complexity of the problem, there has been little previous work that attempts to build statistical models that predict the success of literary works based on their intrinsic content and quality. Some previous studies do touch on the notion of stylistic aspects in successful literature, e.g., extensive studies in Literature discuss literary styles of significant authors (e.g., Ellegård (1962), McGann (1998)), while others consider content characteristics such as plots, characteristics of charac-

Project Proposal

- due Feb. 16 (in two weeks)
- 1-2 pages
- one per group
- include the following:
 - members of your group
 - describe the task you are going to work on (could be a new task you create or an existing task)
 - describe the methods you will use/develop for the task
 - give a brief review of related work; i.e., situate your project with respect to the literature (www.aclweb.org and Google Scholar are useful for this!)
 - a proposed timeline

Project Proposal (cont'd)

- your results do not have to beat the state-of-the-art!
- but your project does have to be carefully done, so that you can draw conclusions
- you are welcome to start by replicating an NLP paper (I can give suggestions if you need some)
- during the week of Feb. 22, please schedule a meeting with me to discuss your project
 - details to follow

Class Presentations

- final two class meetings (March 3rd and March 8th) will be mostly used for in-class presentations
- one presentation per group
- 10-15 minutes per presentation (will be determined once I know how many groups there are)
- you will each take notes and email me questions/feedback for the presenter, which I will anonymize and send

Project

- final report due Thursday, March 17 (original date of the final exam)
- so the presentation will be more like an “interim progress report”

Roadmap

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

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	.

Formulating segmentation tasks as sequence labeling via B-I-O labeling:

Named Entity Recognition

O O O B-PERSON I-PERSON O O O
Some questioned if Tim Cook 's first product

O O O O O O B-ORGANIZATION O
would be a breakaway hit for Apple .

B = “begin”

I = “inside”

O = “outside”

- there are many downloadable part-of-speech taggers and named entity recognizers:
 - Stanford POS tagger, NER labeler
 - TurboTagger, TurboEntityRecognizer
 - Illinois Entity Tagger
 - CMU Twitter POS tagger
 - Alan Ritter’s Twitter POS/NER labeler

Stanford Named Entity Tagger

Classifier:

Output Format:

Preserve Spacing:

Please enter your text here:

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

Submit

Clear

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

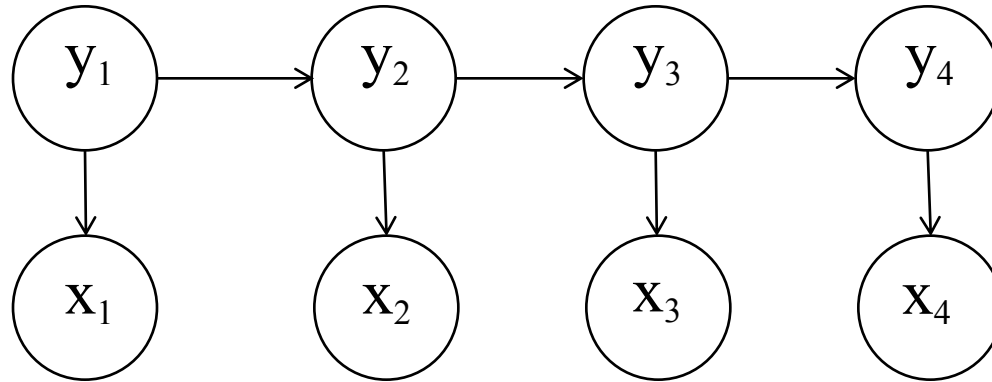
Potential tags:

ORGANIZATION

LOCATION

PERSON

Hidden Markov Models



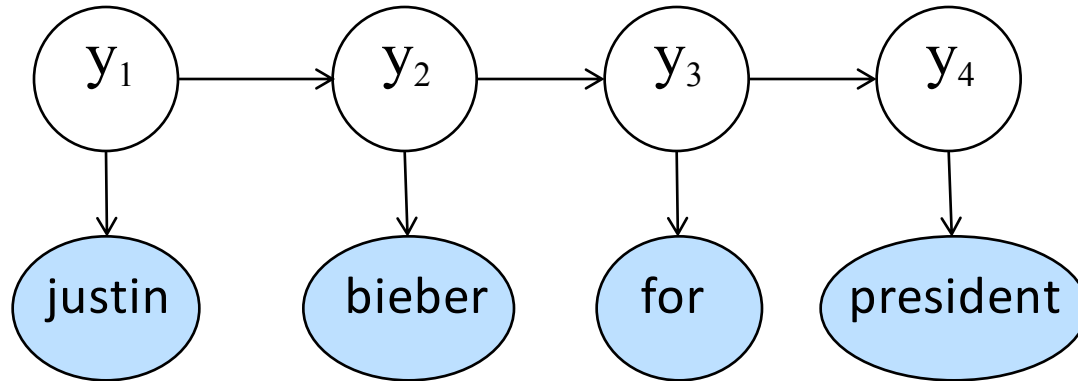
$$p_{\theta}(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{|\mathbf{x}|} p_{\tau}(y_i | y_{i-1}) p_{\eta}(x_i | y_i)$$

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

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

HMMs for Word Clustering

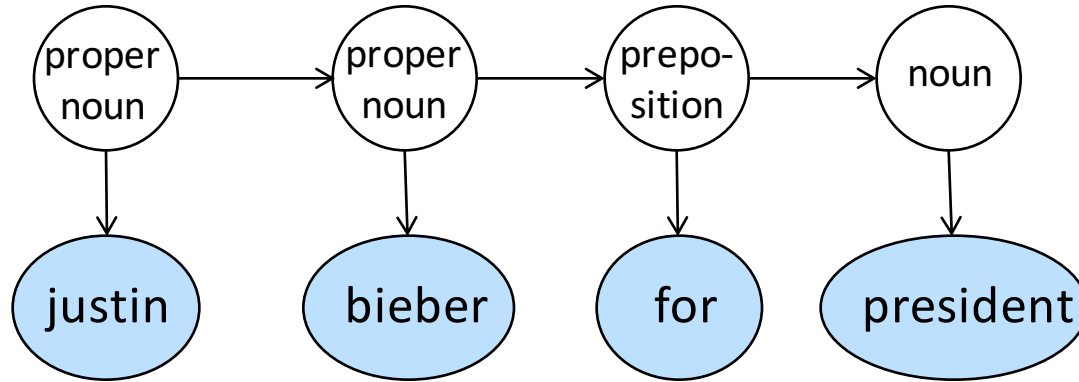
(Brown et al., 1992)



each $y_i \in \mathcal{L}$ is a cluster ID

so, label space is $\mathcal{L} = \{1, 2, \dots, 100\}$

HMMs for Part-of-Speech Tagging



each $y_i \in \mathcal{L}$ is a part-of-speech tag
so, label space is $\mathcal{L} = \{\text{noun, verb, ...}\}$

what parameters need to be learned?

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

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

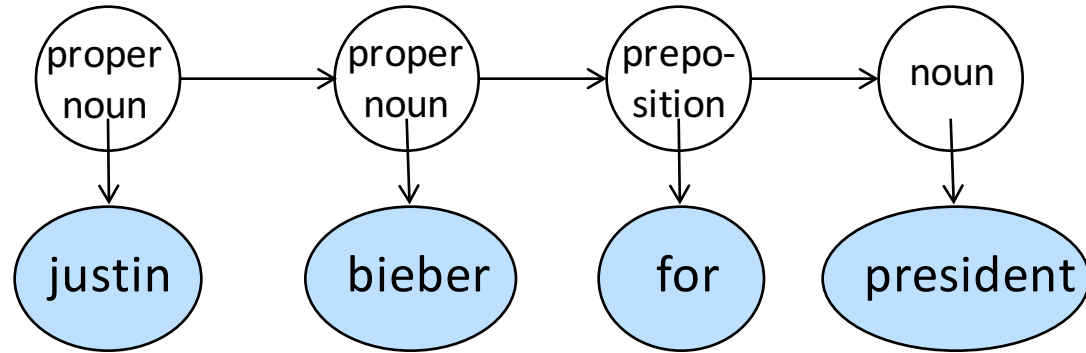
How should we learn the HMM parameters?

transition parameters: $p_{\tau}(y_i | y_{i-1})$ $p_{\tau}(\text{verb} | \text{noun})$
 $p_{\tau}(\text{verb} | \text{adjective})$
...

emission parameters: $p_{\eta}(x_i | y_i)$ $p_{\eta}(\textit{for} | \text{verb})$
 $p_{\eta}(\textit{walk} | \text{verb})$
...

Supervised HMMs

- given a dataset of input sequences and annotated outputs:



- to estimate transition/emission distributions, use maximum likelihood estimation (count and normalize):

$$p_{\tau}(y | y') \leftarrow \frac{\text{count}(y' y)}{\text{count}(y')} \qquad p_{\eta}(x | y) \leftarrow \frac{\text{count}(y, x)}{\text{count}(y)}$$

$$p_{\tau}(\text{verb} | \text{noun}) \leftarrow \frac{\text{count}(\text{noun verb})}{\text{count}(\text{noun})} \qquad p_{\eta}(\text{walk} | \text{verb}) \leftarrow \frac{\text{count}(\text{verb, walk})}{\text{count}(\text{verb})}$$

Estimates of Tag Transition Probabilities

	proper noun	modal verb	infinitive verb	adjective	noun	adverb	determiner
	NNP	MD	VB	JJ	NN	RB	DT
< <i>s</i> >	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Figure 9.5 The A transition probabilities $P(t_i|t_{i-1})$ computed from the WSJ corpus without smoothing. Rows are labeled with the conditioning event; thus $P(VB|MD)$ is 0.7968.

Estimates of Emission Probabilities

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0.000097	0
NN	0	0.000200	0.000223	0.000006	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

Figure 9.6 Observation likelihoods B computed from the WSJ corpus without smoothing.

Inference in HMMs

$$\text{classify}(\mathbf{x}, \boldsymbol{\theta}) = \underset{\mathbf{y}}{\operatorname{argmax}} p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{y}}{\operatorname{argmax}} \prod_{i=1}^{|\mathbf{x}|} p_{\boldsymbol{\tau}}(y_i | y_{i-1}) p_{\boldsymbol{\eta}}(x_i | y_i)$$

- since the output is a sequence, this argmax requires iterating over an exponentially-large set
- last week we talked about using dynamic programming (DP) to solve these problems
- for HMMs (and other sequence models), the for solving this is called the **Viterbi algorithm**

Viterbi Algorithm

- recursive equations + memoization:

base case:

returns probability of sequence starting with label y for first word



$$V(1, y) = p_{\eta}(x_1 | y) p_{\tau}(y | \langle s \rangle)$$

$$V(m, y) = \max_{y' \in \mathcal{L}} (p_{\eta}(x_m | y) p_{\tau}(y | y') V(m - 1, y'))$$



recursive case:

computes probability of max-probability label sequence that ends with label y at position m

final value is in: $V(|\mathbf{x}| + 1, \langle /s \rangle)$

Example:

Janet will back the bill

proper
noun

modal
verb

infinitive
verb

determiner

noun

Janet will back the bill

proper
noun

modal
verb

infinitive
verb

determiner

noun

	NNP	MD	VB	JJ	NN	RB	DT
< <i>s</i> >	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Janet

will

back

the

bill

NNP

0.000032

0

0

0.000048

0

MD

0

0.308431

0

0

0

VB

0

0.000028

0.000672

0

0.000028

JJ

0

0

0.000340

0.000097

0

NN

0

0.000200

0.000223

0.000006

0.002337

RB

0

0

0.010446

0

0

DT

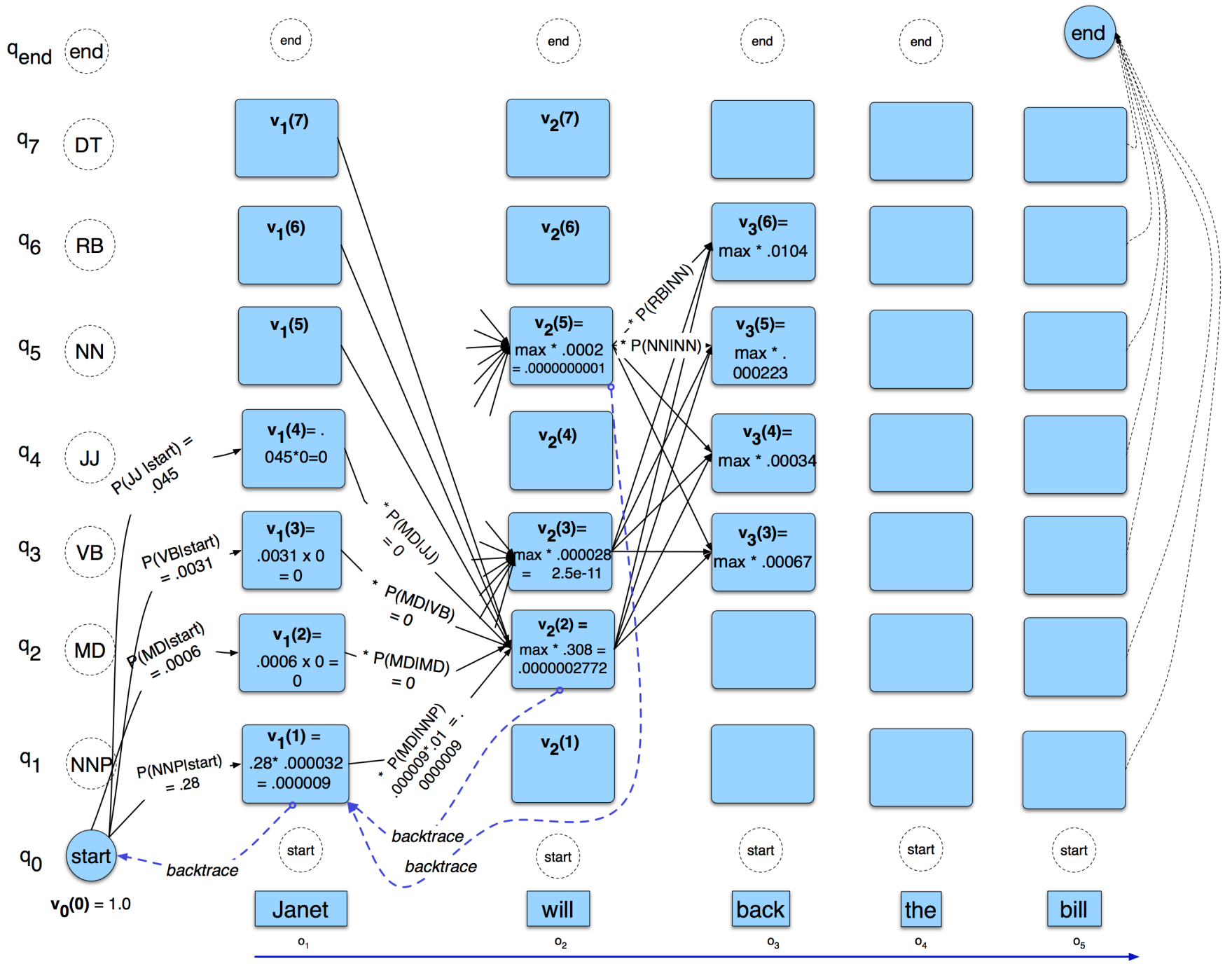
0

0

0

0.506099

0



Viterbi Algorithm

- space and time complexity?
- can be read off from the recursive equations:

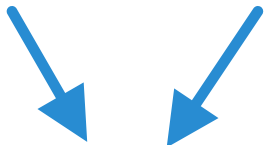
space complexity:

size of memoization table, which is # of unique indices of recursive equations

length of
sentence

*

number
of labels


$$V(m, y) = \max_{y' \in \mathcal{L}} (p_{\eta}(x_m | y) p_{\tau}(y | y') V(m - 1, y'))$$

so, space complexity is $O(|x| |L|)$

Viterbi Algorithm

- space and **time** complexity?
- can be read off from the recursive equations:

time complexity:

size of memoization table * complexity of computing each entry

length of sentence * number of labels * each entry requires iterating through the labels

$$V(m, y) = \max_{y' \in \mathcal{L}} (p_{\eta}(x_m | y) p_{\tau}(y | y') V(m - 1, y'))$$

so, time complexity is $O(|x| |L| |L|) = O(|x| |L|^2)$

Linear Sequence Models

- let's generalize HMMs and talk about linear models for scoring label sequences in our classifier framework:

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

$$\text{score}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) = \sum_i \theta_i f_i(\mathbf{x}, \mathbf{y})$$

- but first, how do we know that this scoring function generalizes HMMs?

HMM as a Linear Model

$$\text{HMM: } p_{\theta}(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{|\mathbf{x}|} p_{\tau}(y_i | y_{i-1}) p_{\eta}(x_i | y_i)$$

$$\text{linear model: } \text{score}(\mathbf{x}, \mathbf{y}, \theta) = \sum_j \theta_j f_j(\mathbf{x}, \mathbf{y})$$

$$p_{\theta}(\mathbf{x}, \mathbf{y}) \propto \exp\{\text{score}(\mathbf{x}, \mathbf{y}, \theta)\}$$

$$p_{\theta}(\mathbf{x}, \mathbf{y}) \propto \exp\left\{\sum_j \theta_j f_j(\mathbf{x}, \mathbf{y})\right\}$$

- what are the feature templates and weights?

HMM as a Linear Model

$$\text{HMM: } p_{\theta}(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{|\mathbf{x}|} p_{\tau}(y_i | y_{i-1}) p_{\eta}(x_i | y_i)$$

$$\text{linear model: } \text{score}(\mathbf{x}, \mathbf{y}, \theta) = \sum_j \theta_j f_j(\mathbf{x}, \mathbf{y})$$

feature templates and weights:

$$f_{\tau(y', y'')}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} \mathbb{I}[(y_{i-1} = y') \wedge (y_i = y'')] \quad \theta_{\tau(y', y'')} = \log p_{\tau}(y'' | y')$$

$$f_{\eta(y', x')}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} \mathbb{I}[(y_i = y') \wedge (x_i = x')] \quad \theta_{\eta(y', x')} = \log p_{\eta}(x' | y')$$

Linear Sequence Models

- so, an HMM is:
 - a linear sequence model
 - with particular features on label transitions and label-observation emissions
 - and uses maximum likelihood estimation (count & normalize) for learning
- but we could use any feature functions we like, and use any of our loss functions for learning!

(Chain) Conditional Random Fields

Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data

John Lafferty^{†*}
Andrew McCallum^{*†}
Fernando Pereira^{*‡}

LAFFERTY@CS.CMU.EDU
MCCALLUM@WHIZBANG.COM
FPEREIRA@WHIZBANG.COM

*WhizBang! Labs—Research, 4616 Henry Street, Pittsburgh, PA 15213 USA

†School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA

‡Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA

Abstract

We present *conditional random fields*, a framework for building probabilistic models to segment and label sequence data. Conditional random fields offer several advantages over hidden Markov models and stochastic grammars for such tasks, including the ability to relax strong independence assumptions made in those models. Conditional random fields also avoid a fundamental limitation of maximum entropy Markov models (MEMMs) and other discrimi-

mize the joint likelihood of training examples. To define a joint probability over observation and label sequences, a generative model needs to enumerate all possible observation sequences, typically requiring a representation in which observations are task-appropriate atomic entities, such as words or nucleotides. In particular, it is not practical to represent multiple interacting features or long-range dependencies of the observations, since the inference problem for such models is intractable.

This difficulty is one of the main motivations for looking at conditional models as an alternative. A conditional model

(Chain) Conditional Random Fields

$$\text{classify}(\mathbf{x}, \boldsymbol{\theta}) = \underset{\mathbf{y} \in \mathcal{L}}{\text{argmax}} \text{score}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$$

$$\text{score}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) = \sum_i \theta_i f_i(\mathbf{x}, \mathbf{y})$$

- linear sequence model
- arbitrary features of input are permitted
- test-time inference uses Viterbi Algorithm
- learning done by minimizing log loss (DP algorithms used to compute gradients)

Max-Margin Markov Networks

Max-Margin Markov Networks

Ben Taskar Carlos Guestrin Daphne Koller
{btaskar, guestrin, koller}@cs.stanford.edu
Stanford University

Abstract

In typical classification tasks, we seek a function which assigns a label to a single object. Kernel-based approaches, such as support vector machines (SVMs), which maximize the margin of confidence of the classifier, are the method of choice for many such tasks. Their popularity stems both from the ability to use high-dimensional feature spaces, and from their strong theoretical guarantees. However, many real-world tasks involve sequential, spatial, or structured data, where multiple labels must be assigned. Existing kernel-based methods ignore structure in the problem, assigning labels independently to each object, losing much useful information. Conversely, probabilistic graphical models, such as Markov networks, can represent correlations between labels, by exploiting problem structure, but cannot handle high-dimensional feature spaces, and lack strong theoretical generalization guarantees. In this paper, we present a new framework that combines the advantages of both approaches: *Maximum margin Markov (M^3) networks* incorporate both kernels, which efficiently deal with high-dimensional features, and the ability to capture correlations in structured data. We present an efficient algorithm for learning M^3 networks based on a

Maximum-Margin Markov Networks

$$\text{classify}(\mathbf{x}, \boldsymbol{\theta}) = \underset{\mathbf{y} \in \mathcal{L}}{\text{argmax}} \text{score}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$$

$$\text{score}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) = \sum_i \theta_i f_i(\mathbf{x}, \mathbf{y})$$

- linear sequence model
- arbitrary features of input are permitted
- test-time inference uses Viterbi Algorithm
- learning done by minimizing hinge loss (DP algorithm used to compute subgradients)

Feature Locality

- **feature locality**: roughly, how “big” are your features?
- when designing efficient inference algorithms (whether w/ DP or other methods), we need to be mindful of this
- features can be arbitrarily big in terms of the input sequence
- but features **cannot** be arbitrarily big in terms of the *output* sequence!
- the features in HMMs are small in both the input and output sequences (only two pieces at a time)

Are these features big or small?

feature	big or small?
feature that counts instances of “ <i>the</i> ” in the input sentence	small
feature that returns square root of sum of counts of <i>am/is/was/were</i>	small
feature that counts “verb verb” sequences	small
feature that counts “determiner noun verb verb” sequences	pretty big!
feature that counts the number of nouns in a sentence	big, but we can design specialized algorithms to handle them if they’re the only big features
feature that returns the ratio of nouns to verbs	

Learning with linear sequence models

- given a linear sequence model with “small” features, how should we do learning?

Loss functions for learning linear sequence models

loss	entry j of (sub)gradient of loss for linear model
perceptron	$-f_j(\mathbf{x}, y) + f_j(\mathbf{x}, \hat{y})$, where $\hat{y} = \text{classify}(\mathbf{x}, \boldsymbol{\theta})$
hinge	$-f_j(\mathbf{x}, y) + f_j(\mathbf{x}, \tilde{y})$, where $\tilde{y} = \text{costClassify}(\mathbf{x}, y, \boldsymbol{\theta})$
log	$-f_j(\mathbf{x}, y) + \mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot \mathbf{x})}[f_j(\mathbf{x}, \cdot)]$

same gradients/subgradients as before, though computing these terms (inference) requires DP algorithms

Implementing DP algorithms

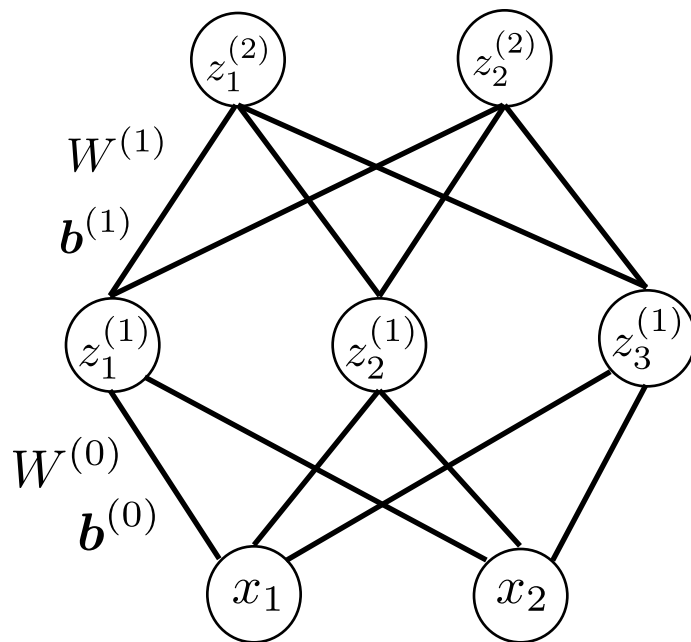
- start with counting mode, but keep in mind how the model's score function decomposes across parts of the outputs
 - i.e., how “large” are the features? how many items in the output sequence are needed to compute each feature?

Neural Networks in NLP

- neural networks
- deep neural networks
- neural language models
- recurrent neural networks and LSTMs
- convolutional neural networks

What is a neural network?

- just think of a neural network as a function
- it has inputs and outputs
- the term “neural” typically means a particular type of functional building block (“neural layers”), but the term has expanded to mean many things



$$z_i^{(0)} = x_i$$

$$z_i^{(q)} = g \left(b_i^{(q-1)} + \sum_{j=1}^{\lambda_{q-1}} W_{ji}^{(q-1)} z_j^{(q-1)} \right)$$