

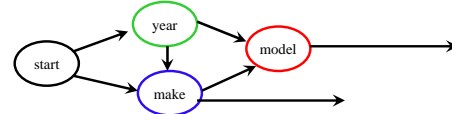
Graphical Model-based Extraction Tools

Matthew Michelson & Craig A. Knoblock
CSCI 548 – Lecture 2

Information Ext: Graphical Models

- Task: Given an input string, and set of states (labels) with probabilities (we define later)
 - What is set of states that produces input sequence?

Input: 2001 Ford Mustang GT V-8 Convertible - \$12700



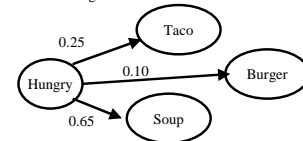
Probabilistic Generative Models

- Generative Models
 - Probability of X and Y $\rightarrow P(X,Y)$
 - There is a water bowl, 1 dog, 1 cat and 1 chicken?
 - What is probability of seeing dog and cat drink together?
 - Look at the # times dog and cat drink together and divide by the # times all animals drink together
- *Markov assumption*: prob. in current state only depends on previous and current state
 - Independence:
 - You just saw the cat drink, but you can't use this information to predict the next drinkers!
 - Standard model
 - Hidden Markov Model (HMM)

Markov Process: Definitions

Let's say we're independent of time, then we can define

- $a_{ij} = P(q_t=S_j|q_{t-1}=S_i)$ as a STATE TRANSITION from S_i to S_j
 - What is the prob. of moving to new state from old state?
 - cat/dog drink from no-one drinking?
- $a_{ij} \geq 0$
- $\sum_{j=1}^N a_{ij} = 1$
- Conserve "mass" of probability \rightarrow all outgoing probabilities sum to 1



Markov Process: More Defs

- Two more terms to define:
 - $\pi_i = P(q_1=S_i)$ = probability that we start in state S_i
 - $b_j(k) = P(k|q_t = S_j)$ = probability of observation symbol k in State j.
 - "Emission probability"
- So, lets say symbols = {cat,dog}, then we could have something like $b_1(\text{cat}) = P(\text{cat}|\text{not-drinking-state})$
i.e. what is the probability that we output cat in not-drinking-state?

Hidden Markov Model

- A Hidden Markov Model (HMM)
 - Set of states, Set of a_{ij} , Set of π_i , Set of $b_j(k)$
- Training
 - From sequences of observations, transition probs (a_{ij}), emission probs ($b_j(k)$), starting probs (π_i)
- Decoding (after training)
 - Input comes in, fits the model's observations
 - Output best state transition sequence that produces input observation
- Can observe the sequence of emissions, but you do not know what state the model is in \rightarrow "Hidden"
 - If 2 states output "yes", all I see is "yes." I have no idea what state or set of states produced this!

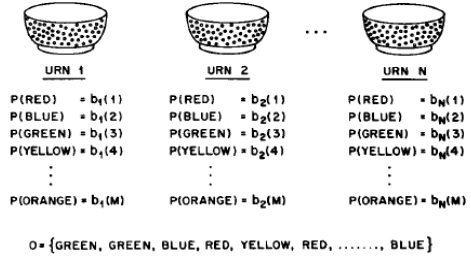
HMM - Example

Urn and Ball Model

Each urn has M distinct colored balls.
Randomly pick an urn, and pick out a colored ball. Repeat!

S = set of states = Set of Urns
Transition Probs = Choice of Next Urn
 $b_i(\text{color})$ = Prob. of picking that colored ball from urn $_i$

Urn and Ball Problem

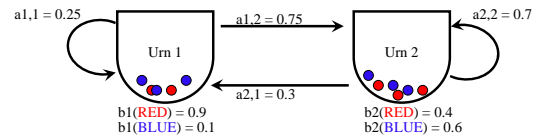


Urn and Ball Example: Training

Let's say we have the following:

- 2 urns
- 2 colors for marbles (Red, Blue)
- $a_{i,j}$ = pick from urn i , go to urn j
- $a_{1,1} = 0.25$ $a_{1,2} = 0.75$
- $a_{2,1} = 0.3$ $a_{2,2} = 0.7$
- $b_1(\text{Red}) = 0.9$, $b_1(\text{Blue}) = 0.1$
- $b_2(\text{Red}) = 0.4$, $b_2(\text{Blue}) = 0.6$

Urn and Ball Example: Decoding

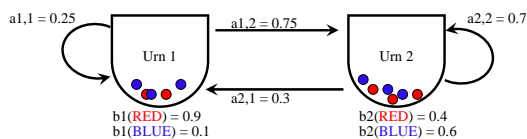


Let's say it's perfectly random to start with either urn, i.e. $\pi_1 = \pi_2 = 0.5$

Now, what if we observe the following marble picks: {RED, RED, BLUE}.

What is the most probable state sequence to produce it?
Urn 1, Urn 2, Urn 2? Urn 2, Urn 1, Urn 1?

Urn and Ball Example: Viterbi!

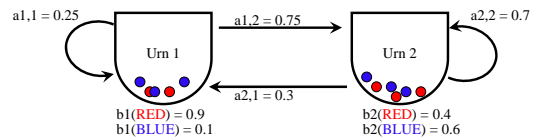


Viterbi algorithm to decode observations recursively:
Define $\zeta(i) = \max P[q_1, q_2, \dots, q_i = i, O_1, O_2, \dots, O_n | \text{HMM model}]$

(Remember, q_t = current state, O are observations)

So, $\zeta_{t+1}(i) = \max[\zeta_t(i) * a_{i,j}] * b_j(O_{t+1})$

Urn and Ball Example: First Pass



Our Observations: {RED, RED, BLUE}.

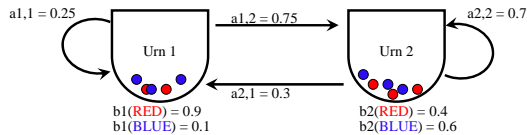
We need a first set of initialized values:

$\zeta_1(i) = \pi_i * b_i(O_1 = \text{RED})$ $i = \{1, 2\}$

$\zeta_1(1) = \pi_1 * b_1(O_1 = \text{RED}) = 0.5 * 0.9 = 0.45$

$\zeta_1(2) = \pi_2 * b_2(O_1 = \text{RED}) = 0.5 * 0.4 = 0.2$

Urn and Ball Example: Recursion



Our Observations: {RED, RED, BLUE}.

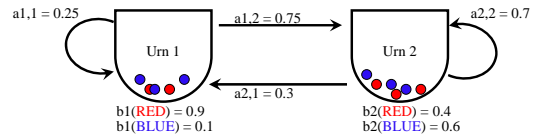
$$\zeta_2(1) = \max(\{\zeta_1(1)*a_{1,1}, \zeta_1(2)*a_{2,1}\}) * b_1(O_2 = \text{RED})$$

$$= \max(\{0.45*0.25, 0.2*0.3\}) * 0.9 = 0.10125$$

$$\zeta_2(2) = \max(\{\zeta_1(1)*a_{1,2}, \zeta_1(2)*a_{2,2}\}) * b_2(O_2 = \text{RED})$$

$$= \max(\{0.45*0.75, 0.2*0.7\}) * 0.4 = 0.135$$

Urn and Ball Example: Recursion



Our Observations: {RED, RED, BLUE}.

$$\zeta_3(1) = \max(\{\zeta_2(1)*a_{1,1}, \zeta_2(2)*a_{2,1}\}) * b_1(O_3 = \text{BLUE})$$

$$= \max(\{0.10125*0.25, 0.135*0.3\}) * 0.1 = 0.00405$$

$$\zeta_3(2) = \max(\{\zeta_2(1)*a_{1,2}, \zeta_2(2)*a_{2,2}\}) * b_2(O_3 = \text{BLUE})$$

$$= \max(\{0.10125*0.75, 0.135*0.7\}) * 0.6 = 0.0567$$

Urn and Ball Example: Decoded

So, we see that at each step, maximally we have:

$$\zeta_3(2) = 0.0567$$

$$\zeta_2(2) = 0.135$$

$$\zeta_1(1) = 0.45$$

So, working backwards, know the most likely state transitions went Urn 2 \leftarrow Urn 1 \leftarrow Urn 1.

So, given observation {RED, RED, BLUE} we say that the most probable State transition set is {Start in Urn 1/red, Go to Urn 2/red, Stay Urn 2/blue}

What if urns were made, model, year? \rightarrow EXTRACTION!

HMM Issues

1 – Independence Assumption

Current observation only depends on what state you are in right now.

(remember the cat?)

Or, to say it differently, the current output has no dependence on previous outputs.

Example: Can't model that if Urn1 outputs a red ball, then Urn2 should decrease it's probability of doing so.

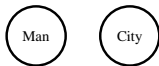
HMM Issues

2 – Multiple Features Issue

□ What about many, interacting features?

□ E.g. $x = \{\text{Doug}\}$

■ *noun* and *capitalized*



■ (*noun* and *capitalized*) \rightarrow *masculine*



HMM Issues

3 – an abundance of training data for one state has no effect on the others

So, if we have lots of data about Urn 2 and we think this could affect how we model Urn 1, we can't use this information...

4 – Must enumerate all possible observation sequences

Conditional Model

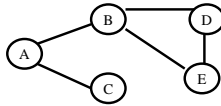
- We prefer a model that is trained to maximize a *conditional* probability rather than *joint* probability: **P(s|o) instead of P(s,o):**
 - Allow arbitrary, non-independent features on the observation sequence
 - Transition probabilities between states might depend on past (and future!) observations
 - Conditionally trained:
 - Given some observations (your input), what are the likely labels (states) that the model traverses given this input?

Conditional Random Fields (CRFs)

- CRFs
 - Conditional model
 - Based on “Random Fields”
 - Undirected acyclic graph
 - Allow some transitions “vote” more strongly than others depending on the corresponding observations
 - Remember the point: Given some observations, what are the labels (states) that likely produced labels?

Random Field: Definition

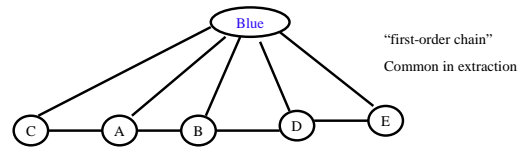
- For graph $G(V,E)$, let V_i be a random variable
If $P(V_i | \text{all other } V) = P(V_i | \text{its neighbors})$
Then G is a random field



$$P(A | B,C,D,E) = P(A | B,C) \rightarrow \text{Random Field}$$

CRF: A Conditional random field

Remember, we have observations X (marbles) and labels Y (states, such as Urns)
Ex) States = {A, B, C, D, E} and some observation **Blue**

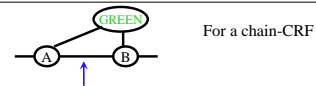


A CRF is s.t. $P(A | \text{Blue}, B, C, D, E) = P(A | \text{Blue}, B, C)$
Or generally: $P(Y_i | X, \text{all other } Y) = P(Y_i | X, \text{neighbors of } Y_i)$

CRF: Usefulness

- CRF gives us $P(\text{label} | \text{obs, model})$
 - Extraction: Find most probable label sequence (y 's), given an observation sequence (x 's)
 - 2001 Ford Mustang GT V-8 Convertible - \$12700
 - No more independence assumption
 - Conditionally trained for whole label sequence (given input)
 - “long range” features (future/past states)
 - Multi-features
 - Now we can use better features!
 - What are good features for identifying a price?

CRF: Features



- Potential Function
 - Chains: operate on pairs of adjacent labels (A,B)
 - No direct probabilistic meaning
 - Instead, think of it as a “constraint on the configurations of the random variables on which the function is defined”
 - “A global configuration with a high probability is likely to have satisfied more of these constraints”

Hanna M. Wallach. [Conditional Random Fields: An Introduction](#), Technical Report MS-CIS-04-21, Department of Computer and Information Science, University of Pennsylvania, 2004.

CRF: Using features

- CRF: Normalized product of potential functions (features!)

- Potential Function:

$$\exp\left(\sum_j \lambda_j t_j(y_{i-1}, y_i, \mathbf{x}, i) + \sum_k \mu_k s_k(y_i, \mathbf{x}, i)\right)$$

defined for whole sequence

"transition" feature function (consider y_{i-1} and y_i)

state feature function (only y_i)

Estimate from training data (think weights)

CRF: More on features

- Used features $b(\mathbf{x}, i)$: [remember \mathbf{x} is whole seq]
 - $b(\mathbf{x}, i) = 1$ if observation (token) at position i is **Ford**
0 otherwise
- Transition/State features use these guys, so:
 - $t_j(y_{i-1}, y_i, \mathbf{x}, i) = b(\mathbf{x}, i)$ if $y_{i-1} = \text{NUMBER}$ and $y_i = \text{NOT-NUM}$
0 otherwise
- Put all possible transition/state features into $F_j(\mathbf{y}, \mathbf{x})$

$$\text{CRF} = p(\mathbf{y} | \mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_j \lambda_j F_j(\mathbf{y}, \mathbf{x})\right)$$

CRF: Practicality...

- We just derived CRF!
 - How do we use it now?
 - ...

MALLET

- Machine learning toolkit specifically for language tasks
 - http://mallet.cs.umass.edu/index.php/Main_Page
- Developed at U. Mass. by Andrew McCallum and his group
 - <http://www.cs.umass.edu/~mccallum/>
- For our purposes, we will use the SimpleTagger class which implements Conditional Random Fields

Getting MALLET to work...

- Install Cygwin (HW Instructions)
- Install Ant (HW Inst.)
- Install MALLET (HW Inst.)
- Train/Test/Label with SimpleTagger

SimpleTagger

- Training
 - Each line is of the form:
<feature1> <feature2> ... <featureN> <label>

Let's start with an example of a sentence:
Los Angeles is a great city!

We want to find all nouns, like the example in:
http://mallet.cs.umass.edu/index.php/SimpleTagger_example

Training CRFs

Let's say we have some tools that can identify features:

Colors → List of colors

Regex → Apostrophe finder

Regex → Capitalized

Stop-Words → Common tokens: a, the, etc.. (not etc. the word..)



Training CRFs

GOAL: Find NOUNS

Note: In SimpleTagger, the default "ignore" label is O (Used in HW)

LABELLED INPUTS:

The SW CAP not-noun

red COLOR not-noun

bear's APOS noun



Train SimpleTagger

- ❑ `java -cp "class;lib/mallet-deps.jar" edu.umass.cs.mallet.base.fst.SimpleTagger --train true --model-file SAVEDMODEL TrainingData.txt`

Labeling with SimpleTagger

- ❑ Once you have a trained model, can re-use it to label new data!
- ❑ `java -cp "class;lib/mallet-deps.jar" edu.umass.cs.mallet.base.fst.SimpleTagger --include-input true --model-file SAVEDMODEL NotLabeledText.txt > LabeledOutput.txt`

CRFs and MALLET

- ❑ Have fun!