Lecture notes by Edward Loper

Course: CIS 639 (Statistical Approaches to Natural Language Processing)
Professor: Mitch Marcus
Institution: University of Pennsylvania

`http://www.cis.upenn.edu/~mitch/cis639.html`

1 Logistics

go over section III of manning carefully go over mike collin's & adwait's code..

Fred Jelloneck?

Bulk pack

Topics

- brill taggers
- hmms
- maxent
- pcfgs
- generative statistical models
- svms
- memory based learning
- voting methods

2 Why corpus based approaches?

in 80's, people said parser problem was solved informal IBM study in 1990: parsers get <40% correct

Why did people say the parser problem was solved? People can "magically adapt" to the capabilities of the system.. e.g., zork.

The apparent problem: NL grammars are very big

- lexical ideosyncracies?

Pervasive ambiguity working hypothesis: build systems that learn? → Supervised learning

Thursday, January 10, 2002

3 Plan for the 1st part of the class

- HMM basics
- Chapter 10 (tagging), quickly
- Chapter 9 (HMMs), in depth
- chunking

4 Tagging

4.1 Langauge Modelling

Task: predict the next item in a sequence

4.2 Markov Models

We can think of a bigram model as a markov model.

Start out with a finite state automaton:

$$\begin{tikzcd}
\texttt{farm} \arrow[r] \arrow[dr] & \texttt{subsidy} \arrow[r] \arrow[dr] & \texttt{for} \\
\texttt{form} \arrow[r] \arrow[ur] & \texttt{subsidies} \arrow[ur] & \texttt{far}
\end{tikzcd}$$

Instead of using a transition function, use a transition matrix: a(i,j) = probability of going from state i to state j.

Estimate a(i,j):

$$a(i,j) = \frac{count(w_i,w_j)}{count(w_i)}$$

4.3 POS tagging

corpus-based techniques do very well

- unigram → 91%
- simple, impossible approach: use argmax$_T$ P(T|W)..
  - T = sequence of tags, W = sequence of words
  - sparse data (zeros)
  - computationally expensive
  - But we only have probability estimates..

What can we estimate well?

If we make assumptions about the distributions, then we can figure things out.

First, assume there's a uniform distribution of 5k words, and 40 part of speech tags.

Then we can figure out what the best case # of samples/instance we can get. E.g., for ⟨ word,tag⟩, we have 5 samples/instance, on average.

Accurate models require a very large amount of data

5 A practical statistical tagger

By bayes rule:

```
            P(W|T) P(T)
P(T|W) = ------------
               P(W)
```

So we want to maximize P(W|T) P(T).

What distribution are we actually drawing from? (Joint of W and T)

5.1 Computing P(T)

$$P(T) \approx P(t_1) * P(t_2|t_1) * P(t_3|t_1,t_2) * \ldots * P(t_n|t_1 \ldots t_{n-1})$$

But we can't estimate P(T). So make the markov assumption:

$$P(t_i|t_1, \ldots, t_{i-1}) = P(t_i|t_{i-1})$$

5.2 Computing P(W|T)

$$P(w_i|t_1,\ldots t_n) \approx P(w_i|t_i)$$

5.3 HMM..

Use an HMM to implement the statistical tagger

$$\begin{tikzcd}
\texttt{[Det]} \arrow[loop above] \arrow[r] \arrow[r, bend right] & \texttt{[Adj]} \arrow[loop above] \arrow[r] \arrow[r, bend right] & \texttt{[Noun]} \arrow[loop above]
\end{tikzcd}$$

(fully connected)

$P(w|t);\ p(t|t_{i-1}$

P(T,W) reduces to:

$\pi(t1) * \prod_i P(w_i|t_i) * \prod_i P(t_i|t_{i-1})$

So the markov model gives us the same equation as the baysian rule..

5.4 Noisy Channel

The noisy channel is memoryless

$\longrightarrow P(e_k|e_{k-1}) \longrightarrow P(f_k|e_k) \longrightarrow \texttt{out}$

5.5 HMMs again

But HMMs can be trained without a tagged corpus. In particular, if we have a set of possible tags for words, and a large unannotated corpus, we can learn all HMM parameters.

Thursday, January 17, 2002

6 Symbolic Learning for POS Tagging

Fun with Brill taggers!

- Use iterative improvement with transformational rules
- Use transformational rule templates

Problems with overfitting?

- Some rule templates can have many many parameters
- You basically don't get overfitting – you're only selecting some rules, so few parameters?

6.1 Why does it work?

- Try scorings other than (right-wrong)?
    - They don't help
    - GPS: Generalized problem solver
    - Newell, Shaw, Simon 1958
    - Means-ends analysis

6.2 So what: Brill's Existential Despair

What if we just have an undergrad do this?

- in a short time, they can produce rules that are just as good.

7 Formalizing HMMs

$\mathtt{HMM} = \langle A, B, \Pi, Q, V \rangle$
$A = \{A_{ij}\}$
$A_{ij} = P(q_j \text{ at } t+1 | q_i \text{ at } t)$
$B = \{B_{jk}\}$
$B_{jk} = P(v_k | q_j \text{ at } t)$
$\Pi = \{\pi_i\}$
$\pi_i = P(q|i \text{ at } 0)$
v_k = vocabulary items
Q = states

Vector quantization:

- method of producing small fixed vocab (v_k)
- first, make a vector discrete by quantizing
- then produce a fixed-size vocab by picking the best n exemplars, and then vocab item selected is whatever's closest.

(For speech recognition: use a cepstrum, not an FFT.)

For speech recognition:

- use fixed number of transitions/state. eg:

$$\begin{tikzcd}
\mathtt{[Det]} \arrow[r] \arrow[loop above] \arrow[r, bend right] & \mathtt{[Adj]} \arrow[r] \arrow[loop above] \arrow[r, bend right] & \mathtt{[Noun]} \arrow[loop above]
\end{tikzcd}$$

- use vector quantization to produce vocab

Arc Emission HMMs:

$B = \{B_{ijk}\}$
$B_{jk} = P(v_k \in V | q_i \text{ at } t,\ q_j \text{ at } t+1)$

State emission HMMs are a special case of arc emission HMMs.

We love Viterbi!

What's P(i$\rightarrow$ j with output v_k)?

$= a_{i,j} b_{i,j,k}$

$\alpha_t[i] = P(q_i \text{ at } t | \text{given the output})$

$\alpha_{t+1}[j] = \sum_i \alpha_t[i]\ a[i,j]\ b[i,j,k]$

7.1 Decoder lattice

Arc is the joint probability on a transition and an output.

$s_i \text{ --}a(i,j)b(i,j,o_k)\longrightarrow s_j$

Node is the joint probability of being in a state and having seen a partial output:

$\alpha(j,t) = P(x_t{=}s_j, o_{0\ldots k} | M)$

We can compute it in O(N^2T) time. Basically, this is because of the markov property: the only things we need to know about the probability for a state is the probabilities for the last state (and the input symbol). Locality is essential for dynamic programming.

Tuesday, January 29, 2002

8 Forward-backward algorithm

(aka Baum-Welsch algorithm)

For any $1 \leq t \leq T+1$:

```
[Eq. 8] P(O|M) = $\sum_{i=1}^{N} \alpha(i,t)\beta(i,t)$
```

Find the best model for this output:

```
$\max_m P(O|m)$
```

8.1 Expectation phase

Compute:

```
$P_{arc}$(i,j,t)
    (=$P_t$(i,j) in the book)
```

The probability that we go from i to j at time t, given the output and the model.

$$P_{arc}(i,j,t) = \frac{\alpha_t(i) a_{i,j} b_{i,j,ot} \beta_{t+1}(j)}{P(O|M)}$$

This is basically just:

```
=P($x_i$ at t; $x_j$ at t+1 | O,M)
```

Note that for all t:

$$\sum_i \sum_j P_{arc}(i,j,t) = 1$$

8.2 Maximization

$$\pi'_i = \sum_j P_{arc}(i,j,1)$$

$$a'_{i,j} = \frac{\sum_t P_{arc}(i,j,t)}{\sum_t \sum_j P_{arc}(i,j,t)}$$

$$b'_{i,j,k} = \frac{\sum_{t \text{ s.t. } ot=k} p_{arc}(i,j,t)}{\sum_t P_{arc}(i,j,t)}$$

9 Speech recognition

9.1 Vowels

- characterized by 3 resonance frequencies. Model throat as tube with semi-divider:

```
||--------------------------
||
||              |
||--------------------------
```

We can move the divider left/right (affects height of freqs) or up/down (affects the relative power of formants)

Tuesday, February 5, 2002

10 Statistical MT

We want:

```
ehat = argmax_e P(E|F)
```

Noisy channel:

```
P(E) → P(F|E) →
```

- P(E) is language model
- P(F|E) is an HMM with state=word

What's wrong with this model for P(F|E)?

- word order problem: english and french word orders may differ.
- fertility: n-to-m translations (e.g., not → ne...pas)

Use a generative model (HMMs are a generative model)

Why estimate P(F|E) rather than just directly estimating P(E|F)? Well, one problem is that our model of P(E|F) gives a lot of probability mass to giberish. This happens because there are many more sentences that are not english than those that are. But now consider running P(E)P(F|E). Here, P(E) helps us select things that are not gibberish; and then it doesn't matter that P(F|E) assigns lots of probability mass to gibberish; we throw that out. Since we're doing an argmax, that's ok.

Picture:

```
[  JUNK   --]
[     --  -\]\
[  [    ]\--]>---[    ]
[  [ Fr ]---]----[ En ]
[  [    ]-x-]>---[    ]
[        x-/]/
```

Put another way: we have a generative model that overgenerates. How do we know where it's overgenerating?

Use "alignment" to take care of order and fertility

- consists of connections: eg., ⟨2,1⟩.
- for an english sentence with length l, and a french target length m, there are L*m possible connections, so 2^{l*m} possible alignments.

Generative model (roughly model 1):

1. pick target length
2. pick connections (independantly)
3. for each connection, generate a word (conditional only on the english word)

cepts = concepts: map via an intermediate generative locus, which allow us to handle fertility more gracefully. Empty → empty cept; multi→ helps us with order.

Adding alignments:

```
P(F=f,A=a|E=e)
```

So to get P(f|e):

```
P(f|e) = ∑_a P(f,a|e)
```

To get P(f,a|e) we can do:

```
P(f,a|e) = P(f|a,e)P(a|e)
```

Note that the length m is hidden in the choice of alignment.

10.1 Model 1

Limit ourselves to n-to-1 ($n \geq 0$). (n English words to 1 French word, if we're translating French$\rightarrow$ English)

- Add a single ø symbol to each French sentence, to generate any 0-to-1 translations.

The following is true (derived from chain rule)

$$P(f,a|e) = P(m|e)\prod_{j=1...m} P(a_j|a_1^{j-1},f_1^{j-1},m,e) \\ P(f_j|a_1^{j},f_1^{j-1},m,e)$$

Where:

$a_1^j = a_1 a_2 ... a_j$
$f_1^j = f_1 f_2 ... f_j$
a_j = position in English of French word j.

Note that this a_j notation imposes the constraint that each French word connects to exactly one English word.

This equation says:

```
For each i: first, generate the next alignment for i; then
pick the word for that alignmnent.
```

Now simplify it, because of sparse data problem: use backoff to simpler things.

Tuesday, February 12, 2002

fun with mt

10.2 More Model 1

Estimating P(m|e)

- Assume that it's independant of both e and m.
- Assume that there's a maximum length $1/\epsilon$.

So:

$$P(m|e) \approx \epsilon$$

Estimating $P(a_j|a_1^{j-1},f_1^{j-1},m,e)$

This is basically the alignment probability. Some plausible alternatives:

- Condition it on j
- Condition it on a_{j-1}

But we'll be even more simple. Free word order, but make sure you at least get some word. So:

$$P(a_j|a_1^{j-1},f_1^{j-1},m,e) \approx 1/(l+1)$$

(The "+1" is for the empty cept)

Estimating $P(f_j|a_1^{j},f_1^{j-1},m,e)$

Use the alignment to directly translate words. Estimate probability of french words using the "translation probability:"

$$P(f_j|a_1^{j},f_1^{j-1},m,e) \approx P(f_j|e[a_j])$$

Use frequency, or maybe smoothed frequencies

Putting it all together

$$P(f,a|e) = \epsilon * (l+1)^{-m} \prod_j P(f_j|e[a_j])$$

Sum that over all alignments.. But there are $(l+1)^m$ possible alignments. We could just enumerate all alignments, and sum the P's.

10.3 Model 2

Estimating $P(a_j|a_1^{j-1},f_1^{j-1},m,e)$

Make a smarter model: alignment depends on j, a_j, m, and l.

$$P(a_j|a_1^{j-1},f_1^{j-1},m,e) \approx P(a_j|j,m,l)$$

Impose the constraint:

$$\sum_i P(a_i|j,m,l) = 1$$

Putting it all together

$$P(f,a|e) = \epsilon * \prod_j P(f_j|e[a_j])P(a_j|j,m,l)$$

10.4 Model 3

New basic model. Explicitly model the cepts.

- ϕ Fertility
- τ Tableau (translation table for a given fertility)
- π Permutation

Full model:

$$\begin{aligned}
&P(\tau,\pi|e) = \\
&\quad \prod_{i=1\ldots l} P(\phi_i|\phi_1^{i-1},e) * \\
&\quad P(\phi_0|\phi_1^{l},e) * \\
&\quad \prod_{i=0\ldots l}\prod_{k=1\ldots\phi_i} P(\tau_{ik}|\tau_{i1}^{k-1},t_0^{i-1},\phi_0^{l},e) * \\
&\quad \prod_{i=1\ldots l}\prod_{k=1\ldots\phi_i} P(\pi_{ik}|\pi_{i1}^{k-1},\pi_1^{i-1},\tau_0^{l},\phi_0^{l},e) * \\
&\quad \prod_{k=1\ldots\phi_0} P(\pi_{0k}|\pi_{01}^{k-1},\pi_1^{l},\tau_0^{l},\phi_0^{l},e)
\end{aligned}$$

- Line 0: to find the probability of a translation…
- Line 1: pick a fertility for each word
- Line 2: pick a fertility for the empty cept
- Line 3: pick a translation for each word
- Line 4: pick a permutation for each word
- Line 5: pick a permutation for the empty cept.

What independance assumptions do we want to use? (Too many parameters!)

Model with independences:

$$\begin{aligned}
&P(\tau,\pi|e) = \\
&\quad \prod_{i=1\ldots l} P(\phi_i|e_i) * \\
&\quad \mathrm{CHOOSE}(\phi_1+\ldots\phi_l,\ \phi_0)(1-p_1)^{\phi_1+\ldots+\phi_l-\phi_0}p_1^{\phi_0} \\
&\quad \prod_{i=0\ldots l}\prod_{k=1\ldots\phi_i} P(\tau_{ik}|e_i,\phi_i) * \\
&\quad \prod_{i=1\ldots l}\prod_{k=1\ldots\phi_i} P(\pi_{ik}|l,\ m,\ i) * \\
&\quad \prod_{k=1\ldots\phi_0} P(\pi_{0k}|\pi_{01}^{k-1},\pi_1^{l},\tau_0^{l},\phi_0^{l})
\end{aligned}$$

- Line 1: fertility of normal words just depends on the word
- Line 2: assume we can get fertility of the emtpy cept by using a probability p_1 for getting an empty cept for a given word..
- Line 3: translation depends on the english word & its fertility
- Line 4: permutation depends on the length of the english & french sentences; and the index of the current (english) word.
- Line 5: permutation of the empty cept: put the empty cept only in places where there weren't already words. Give it even distribution in empty spots..

Tuesday, February 19, 2002

11 Parsing by ”Chunking”

find non-recursive grammatical chunks

non-nested NPs up to the head.

Thursday, February 28, 2002

12 Smoothing

Smoothing is fun! Smoothing is good!

12.1 Theory

- Small counts yield bad estimtes

Tuesday, March 5, 2002

12.2 Good-Turing Estimation

Make the assumption that the material is binomial. I.e., words in a document are iid.

- Let N_r be the number of items that occur r times
- Insight: N_r can provide a better estimate of r
- Adjusted frequency r^*:

$$r^* = \frac{E[N_{r+1}]}{E[N_r]} (r+1)$$

Works well for language modelling, despite the fact that the binomial condition doesn't really hold..

Problems:

- To estimate r^* for r=0, we must know how many things never occured (=N_0)
- For large r, N_r gets small, so $E[N_r]$'s must be smoothed

12.3 Terminology

- Smoothing: average two distributions
- Backoff: switch from one distribution to another distribution, depending on (some aspect of) the input.
- So in some cases, backoff is a subset of smoothing.

Tuesday, March 5, 2002

13 Smoothing in Practice...

Dan Bikel

Smoothing..

- We want to estimate the likelihood of things that weren't observed (esp zero-count items).

13.1 Deleted interpolation

- Create a smoother distribution by linearly interpolating several (hopefully) related distributions

$p(A|B) = \sum_i \alpha_i P(A|\phi_i(B))$

13.2 Witten-Bell

- Instead of modifying "rough" estimates from one part of the corpus, using counts gathered from a held-out section, try to estimate the confidence in estimates directly.
- We want direct confidence estimates for probability estimates (which can be used as α s in smoothing)
- How do we estimate confidence in a conditional probability estimate?
  - Base it on the *shape* of the distribution

Begin Digression... Define probability theory:

- Ω: set of events
- $F \subseteq 2^\Omega$
- P: $F \rightarrow R$

Define expectation:

- $E[X] = \sum_x xp(x)$ is the center of mass (μ_x).
- Also consider the center of mass in the y dimension, μ_y. This quantity is related to entropy (in particular, entropy is the expected value of log[p(x)]; μ_y is the expected value of p(x)).

End Digression...

For more uniform distributions, we have less confidence; for less uniform distributions, we have more confidence. For example, MLE will do very badly if everything occurs exactly once. This is esp bad if we don't know the underlying set of events.. But still true otherwise.

So, we trust distributions with lower entropy, and distrust distributions with higher entropy.

13.3 Basic Witten-Bell

confidence for a pdf P(A|B):

```
              c(B)
λ = --------------------
       |{Aᵢ:c(Aᵢ,B)>0}|+c(B)
```

c is count.

or simpler notation:

```
λ = d/(d+u) = 1/(1+u/d)
```

("u" for unique, aka diversity)

The link to entropy:

```
u/d = 1/nbar
nbar = average of n..
```

Using weights..

```
λ₁e₁ + (1-λ₁)[λ₂e₂ + (1-λ₂)e₃]
```

Chen & Goodman (96, 98) did analysis of smoothing techniques for language modeling. They found that Witten-Bell was very bad for language modeling.. Why? Does this mean we shouldn't use Witten-Bell?

- They didn't explore the formula as it actually get used

People actually use 1/1+(k*u/d) instead of 1/1+(u/d). k is a "fudge factor," typically at least one. "Fudges" the number of unique outcomes for some history. This allows us to reserve more of the probability mass.

But what do we use for k? Use held-out data to optimize k..

Other tricks..

- Add a factor to compensate for equi-trained submodels: if we have 2 models that are equally well trained for some instance, then we tend to trust the one with more confidence.

```
     [    c(φ_{i-1}(B)) ]      d_i
λ_i = [ 1 - ---------- ] * -------
     [    c(φ_i(B))    ]    d_i+k*u_i
```

- Use an additive factor

Witten-Bell just takes one pass through the data. We can count things directly, etc. Good for estimating probabilities of unseen events. Fast reestimation, etc.

check newsgroup... (next time: PCFGs & probablistic parsing)

Thursday, March 7, 2002

14 Introduction to Statistical Parsing

Determining Grammatical Structure.. We need (roughly):

- A grammar that specifies which sentences are legal
- A parsing algorithm that assigns possible structures to new word strings.
- A method for resolving ambiguities

Begin digression

Terminology: ”recursive transition networks” are those FSA things which consist of a set of named FSAs, where edges can be labeled with the name of an FSA.. eg:

```
[S] --NP⟶[ ]--VP⟶[E]

              --
             /  \ adj
             ↘ /
[NP] --det⟶[ ]--Noun⟶[E]
 \           ↗
  -NP→[ ]-'s
```

augmented transition networks: recursive transition networks with registers.. Woods ’69

End digression

Tuesday, April 2, 2002

Assembling Current Parsing Technology

- Inside algorithm – PCKY
- (outside prob) * (inside prob) = prob that constituant in sentence (used to do a beam search of the space; usually, approximate outside prob).
- lexicalized CFGs: associate a head word with each node. Gives us a good stand-in for context sensitivity. But creates a *lot* of rules.
- So we need to deal with sparse data

Today:

- Prepositional phrase attatchment as language modelling
- sparse data – backoff
- "linguistic" analysis & sparse data
- steting & nagao

15 Presentatoin schedule

Thurs 4	[Cardie & Pierce]	Erwin, Seung-Yun
Mon 8	[Veenstra]	Mike, Dave
Tues 9	[ADK]	Xiayi, Shudong
Thurs 11	[TKS, Veenstra]	Edward, Nikhil
Mon 15	[MPRZ]	Jinying Chen, Libin Shen
Tues 16	[Kudo, Matsumoto]	Alex, Anne
Thurs 18		Fernando