Techniques-NNS For-IN Tagging-VBG Parts-NNS of-IN Speech-NN

Presented by: Steve Engels

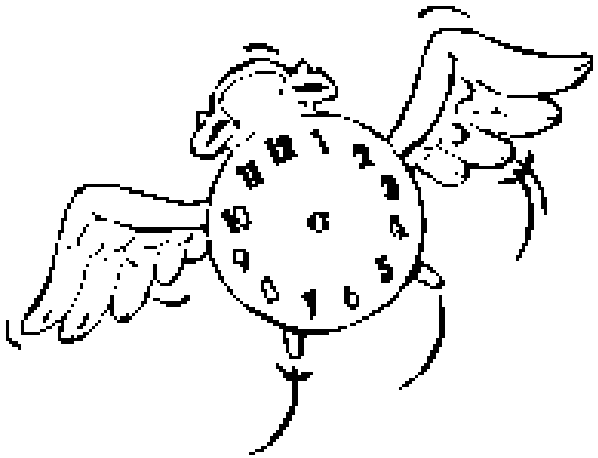
Date: April 2, 2001

Overview to Tagging

- Tagging = Labeling each word in a sentence with its part of speech (e.g. noun, adjective, etc.)
- Example: "Time flies like an arrow"
  - NN VBZ RB AT NN
    - Time is flying in the same way an arrow would (p=0.01)
  - NN NNS VB AT NN
    - "Time flies" are fond of an arrow (p=0.01)
  - NN NNS RB AT NN
    - Fragment: "Time flies" similar to an arrow (p=0.001)
  - NN VBZ VB AT NN
    - Meaningless sentence fragment (p=0)

Overview to Tagging

- Example Tags:
  - **AT** article
  - **BEZ** the word "is"
  - **IN** preposition
  - **JJ** adjective
  - **JJR** comparative adjective
  - **MD** modal
  - **NN** singular or mass noun
  - **NNP** singular proper noun
  - **NNS** plural noun
  - **PERIOD** . : ? !
  - **PN** personal pronoun
  - **RB** adverb
  - **RBR** comparative adverb
  - **TO** the word "to"
  - **VB** verb, base form
  - **VBD** verb, past tense
  - **VBG** verb, present participle, gerund
  - **VBN** verb, past participle
  - **VBP** verb, non-3rd person singular present
  - **VBZ** verb, 3rd singular present
  - **WDT** *wh*-determiner (what, which, etc.)



Overview of Tagging

- Tagging parts of speech can represent the semantic layer between unlabeled and parsed data
- Tagging is easier to approach, because of its more limited scope
- Most tagging approaches achieve 96%-97% accuracy on a per-word basis
  - Bad for sentences, since this amounts to ~1 error/sentence

Information Sources for Tagging

- Syntagmatic structure = given context words or tags, what is the most likely mapping of a word to a tag
  - not very useful, since most words can have various parts of speech, therefore hard to determine a word's context
- Lexical information = most likely tag is assigned to a word (Charniak: "dumb tagger")
  - performance is better (~90%), used as baseline for taggers
- Best methods combine both sources of information

Information Sources for Tagging

- Statistical information
  - frequency information captures main uses of a word. Top usage is labeled "basic" part of speech
  - better than deterministic tagging, which is tempted to omit more obscure tags of a word
  - added information can be inferred about other meanings, assuming that secondary tags are "derived" from basic POS

Markov Model Taggers

- Two main properties:
  - Limited horizon $P(x_{i+1}=t^j|x_1...x_i) = P(x_{i+1}=t^j|x_i)$
  - Time invariant $P(x_{i+1}=t^j|x_i) = P(x_2=t^j|x_i)$
- These prove problematic for extracting certain phenomena, such as *wh*-extraction
- Using HMMs, probability of a tag s_j given word k and previous tag si is: $P(o_n=k|x_n=s_i, x_{n+1}=s_j)$
- Probability of a word given a tag is: $P(w|t) = \frac{C(w,t)}{C(t)}$

Markov Model Taggers

- So applying Bayes' Rule to get best tag sequence $t_{1,n}$:

$$\begin{aligned}\arg\max_{t_{1,n}} P(t_{1,n} \mid w_{1,n}) &= \arg\max_{t_{1,n}} \frac{P(w_{1,n} \mid t_{1,n}) P(t_{1,n})}{P(w_{1,n})} \\ &= \arg\max_{t_{1,n}} P(w_{1,n} \mid t_{1,n}) P(t_{1,n}) \\ &= \arg\max_{t_{1,n}} \prod_{i=1}^{n} \left[P(w_i \mid t_i) \times P(t_i \mid t_{i-1}) \right]\end{aligned}$$

assuming that words are independent of each other, and only depend on their tag

Variations on Markov Model Taggers

- Unknown words:
  - Generally treated as any part of speech, with likelihood distribution that reflects lexicon as a whole
  - This loses lexical information, so other refinements are used:
    - destemming
    - likelihood of <unknown> given POS
    - capitalisation
    - roots & suffixes (Charniak)

Variations on Markov Model Taggers

- Trigram taggers
  - Motivation: capture information lost in Limited Horizon
  - encounters sparse data problems (e.g. commas)
  - possible to combine unigram, bigram & trigram with weights to produce a better model
  - Variable Markov Models (VMMs) use variable sequence length, determined by splitting states and invoking longer sequences based on the current input

Ha ha, made you look.

Variations on Markov Model Taggers

- Smoothing
  - Used to increase probabilities of generating words that do not occur in training corpus
- Reversibility
  - "Decoding" tags usually occurs from left to right, but right to left is equivalent, assuming that first & last tags are the same (e.g. PERIOD)

$$P(t_{1,n}) = P(t_1)P(t_{1,2}\,|\,t_1)P(t_{2,3}\,|\,t_2)..P(t_{n-1,n}\,|\,t_{n-1})$$

$$= \frac{P(t_1)P(t_{1,2})P(t_{2,3})..P(t_{n-1,n})}{P(t_1)P(t_2)..P(t_{n-1})}$$

$$= P(t_n)P(t_{1,2}\,|\,t_2)P(t_{2,3}\,|\,t_3)..P(t_{n-1,n}\,|\,t_n)$$

Variations on Markov Model Taggers

- Tag by tag
  - choose most likely tag based on $P(t_i|w_{1,n})$ for all i, instead of $P(t_{1,n}|w_{1,n})$
  - "Time flies" example
  - Should hopefully eliminate errors Viterbi makes if a state is chosen incorrectly, since tag-by-tag assumes state independence

The clocks represent "time flies", if you haven't figured that out yet.

Hidden Markov Models

- Initialising
  - random assignment too unconstrained; should account for non-existent entries in lexicon
  - Jelinek's method:

$$b_{j.l} = \frac{b^*_{j.l} C(w^l)}{\sum_{w^m} b^*_{j.m} C(w^m)}$$

$b_{j.l}$ = probability of word l given tag j

$b^*_{j.l}$ = 0 if $p(t^j|w^l)==0$

1/(# tags of w^l) otherwise

Hidden Markov Models

- Initialising (cont'd)
  - Kupiec's method:
    - create "metawords" which group words with the same possible parts of speech (e.g. NN & JJ)
    - Jelinek's method is then used on metawords instead of words, which reduces the number of parameters to tune (i.e. not every word, just general word categories)
      - more suitable for sparse data conditions, doesn't work as well under non-sparse conditions
- Training findings
  - given tagged training corpus & similar test set, Visible Markov Models are the best approach
  - given different test set or lexical info with no tagged training corpus, a few iterations of Forward-Backward algorithm improves performance

Transformation-Based Tagging

- Basic idea:
  - need tagged corpus and dictionary
  - label each word with most likely tag (dictionary)
  - transform tags based on surrounding context (triggers)
    - E.g. NN becomes VB when preceded by TO

      VB becomes NN when preceding TO is preceded by VB
  - Transformations are hand-crafted rules that change the tags from most likely to correct tags (seem hand-crafted)
  - Greedy search finds transformations that minimize tag error
  - After training, transformations are ranked and applied to each word of unseen data

Transformation-Based Tagging

- Variations:
  - delayed-effect transformations (e.g. AAAA)
  - unsupervised learning as alternative to HMM tagging
    - E.g. "the can is open"

      AT ____ BEZ JJ
- Similar to decision trees, except no splitting and application of rules to branches (reduces overfitting)
- Probabilistically, this does not quantify the likelihood of a tagging being correct
  - statistical only during training

Transformation-Based Tagging

- Automata
  - Finite-state transducers that take in symbol and output another symbol on each arc, then combine together to create equivalent transformation machine
  - Takes RKn steps (R=number of transformations, K=length of triggering environment, n=length of input)

Transformation-Based Tagging

- Advantages
  - triggers on richer set of events
  - easier to understand; more intuitive
  - resistant to overfitting
  - applicable to parsing and word sense disambiguation
- Disadvantages (not mentioned)
  - rules hand-crafted, subject to human error
  - rules can overlap, interfere with each other
  - other reasons?

Other Tagging Stuff

- Other methods (Chap 16)
  - neural networks
  - decision trees
  - k nearest neighbor
  - maximum entropy models
  - EngCS (Constraint Grammar)
- Performance Factors
  - amount of training data
  - tag set size (larger = more ambiguity)
  - difference between training and test corpora
  - unknown words (bigger dictionary = bigger coverage)
  - "favourable conditions" e.g. Charniak's Brown-based lexicon

Other Tagging Stuff

- Applications
  - partial parsing
  - information extraction
  - indexing for information retrieval
  - question answering

Yep, I am one seriously funny guy.