SEQUENCE LABELING

CS 562/662: Natural Language Processing

2015-02–02

In sequence labeling problems, we cannot always label each sequence element independently (i.e., in isolation), as the best prediction for the y_t may depend on the label at y_{t-1} and so on…

PART OF SPEECH TAGS

- The “parts of speech” consist of several major categories—the key four being *noun*, *verb*, *adjective*, and *adverb*—plus adpositions, particles, discourse markers, fillers, numbers, punctuation, and so on, but most systems also encode additional morphological distinctions

- Sets vary from about a dozen (the “Universal tagset”; Petrov et al. 2012) to several hundred (Hungarian; Oravecz & Dienes 2002)

UNIVERSAL TAGSET

- `ADJ`: adjective
- `ADP`: adposition
- `ADV`: adverb
- `CONJ`: conjunction
- `DET`: determiner
- `NOUN`: noun
- `NUM`: number
- `PRON`: pronoun
- `PRT`: particle
- `VERB`: verb
- `.`: punctuation
- `X`: all other

[Source: Petrov et al. 2012]

PENN TREEBANK TAGSET

- `JJ`, `JJR`, `JJS`: adjective
- `IN`: preposition
- `RB`, `RBR`, `RBS`, `WRB`: adverb
- `CC`: conjunction
- `DT`, `EX`, `PDT`, `WDT`: determiner
- `NN`, `NNS`, `NNP`, `NNPS`: noun
- `CD`: number
- `PRP`, `PRP$`, `WP`, `WP$`: pronoun
- `POS`, `RP`, `TO`: preposition
- `MD`, `VB`, `VBD`, `VBG`, `VBN`, `VBZ`: verb
- `#`, `$`, ``, ‘’, `-LRB-`, `-RRB-`, `.`, etc.: punctuation
- `FW`, `LS`, `SYM`, `UH`: others

[Source: Santorini 1990]

```
Pierre/NNP Vinken/NNP ,/, 61/CD years/
NNS old/JJ ,/, will/MD join/VB the/DT
board/NN as/IN a/DT nonexecutive/JJ
director/NN Nov./NNP 29/CD ./.
```

```
Pierre/NOUN Vinken/NOUN ,/. 61/NUM
years/NOUN old/ADJ ,/. will/VERB join/
VERB the/DET board/NOUN as/ADP a/DET
nonexecutive/ADJ director/NOUN Nov./NOUN
29/NUM ./.
```

USING TAGS

- Find all instances of some morphological class
- But more often, used as “downstream” feautures for:

  Parsing (shallow, dependency, constituency)

  Word sense disambiguation (e.g., dog_n vs. dog_v)

NOUN PHRASE CHUNKING

- A *NP-chunk* is a non-recursive, non-overlapping noun phrase:

```
[Pierre Vinken] , [61 years] old , will
join [the board] as [a nonexecutive
director] [Nov. 2] .
```

- Can encode this using tags (BIO encoding):

```
[Pierre/B Vinken/I] ,/O [61/B years/I] old/
O ,/O will/O join/O [the/B board/I] as/O
[a/B nonexecutive/I director/I] [Nov./B 29/
I] ./O
```

[Source: Ramshaw & Marcus 1995, Tjong Kim Sang & Veenstra 1999]

OTHER SEQUENCE TAGGING APPLICATIONS

- Joint POS tagging/chunking

- Named entity recognition:

```
[PERS Pierre Vinken] … will join
[ORG the board] …
```

- Time expression recognition:

```
… [Nov. 2](1990-11-02) …
```

GENERATIVE TAGGING MODELS

- Objective: find a tag sequence $t_{i...n}$ for a sentence $w_{i...n}$

- This is represented by the joint probability distribution $P(w_{i...n}, t_{i...n})$, which may either be observed (supervised learning) or inferred from $w_{i...n}$ sequences (unsupervised learning via the forward-backward algorithm)

- At inference time, we ask “what tag sequence gave rise to this sentence?” and select the tag sequence $t_{i...n}$ maximizing the conditional probability

GENERATIVE HIDDEN MARKOV MODEL TAGGING

- Construct a *lattice* of size |*n*| x |*t*|, each state of which stores the probability of observation *i* being labeled *t*

- Simple case (bigram tagger):

$$P(w_i \mid t_i)\, P(t_i \mid t_{i-1})$$

- Complex case (HunPos tagger):

$$\mathrm{P}(w_i \mid t_i, t_{i-1}, t_{i-2})\, P^*(t_i)$$

where $P^*(t_i) = \lambda_1 P(t_i) + \lambda_2 P(t_i \mid t_{i-1}) + \lambda_3 P(t_i \mid t_{i-1}, t_{i-2})$

DISCRIMINATIVE HIDDEN MARKOV MODEL TAGGING

- We're really interested in $P(t_i \mid w_{i\ldots}, t_{i-1\ldots})$ but the generative model requires a difficult-to-construct estimate for $P(w_i \mid t_i\ldots)$

- Discriminative methods directly estimate $P(t_i \mid w_{i\ldots}, t_{i-1\ldots})$ from observed word/tag sequences

- At inference (tagging) time, we ask "what is the most likely tag sequence for this sentence?"

DISCRIMINATIVE POS TAGGING FEATURES

- “Emission” features:
  - Token identity: `w_i='FOUR'`
  - Identity of adjacent tokens: `w_i-1='SCORE'`
  - (String) prefixes and suffixes of token: `suf3(w_i)='NTY'`
  - Capitalization of token: `case(w_i)=*title*`
- “Transition” features:
  - Tag history unigram: `t_i-2='NN'`
  - Tag history suffix: `t_i-2='NN',t_i-1='IN'`

HMM DECODING

- In *greedy decoding*, we generate the "transition" features at time *t* using current best hypotheses for $y_{1...t-1}$,. The prediction is the one which maximizes the score of the emission features at *t* and these hallucinated transition features.

- In *Viterbi decoding* (Collins 2002), we construct a lattice of scores for each possible label at each time point. Let the *j*-th state at time *t* be denoted by $\sigma_{j,t}$, and its score be given by $s(\sigma_{j,t})$. We first select the most likely prior state at time *t*, selected by identifying the prior state $\sigma_{j,t-1}$ maximing $s(\sigma_{i,t-1} \rightarrow \sigma_{j,t})$, defined as the sum of the score of the prior state —$s(\sigma_{j,t})$ —and the model score for the $\sigma_{i,t-1} \rightarrow \sigma_{j,t}$ transition. Then, $s(\sigma_{j,t})$ is the sum of $\sigma_{j,t-1} + s(\sigma_{i,t-1} \rightarrow \sigma_{j,t})$ and the model score for emitting *j* at time *t.* Once the lattice is complete, we find the most likely label at the last position and follow backtraces to generate the most likely sequence.