Maximum Entropy Tagging
(for the Maximum Entropy method itself,
refer to NPFL067 added slides 2018/9)
The Task, Again

• Recall:
  – tagging ~ morphological disambiguation
  – tagset \( V_T \subset (C_1, C_2, \ldots, C_n) \)
    • \( C_i \) - morphological categories, such as POS, NUMBER, CASE, PERSON, TENSE, GENDER, ...
  – mapping \( w \rightarrow \{ t \in V_T \} \) exists
    • restriction of Morphological Analysis: \( A^+ \rightarrow 2^{(L, C_1, C_2, \ldots, C_n)} \)
      where \( A \) is the language alphabet, \( L \) is the set of lemmas
  – extension to punctuation, sentence boundaries (treated as words)
Maximum Entropy Tagging Model

• General
\[ p(y,x) = \frac{1}{Z} e^{\sum_{i=1..N} \lambda_i f_i(y,x)} \]
Task: find \( \lambda_i \) satisfying the model and constraints
  • \( E_p(f_i(y,x)) = d_i \)
where
  • \( d_i = E'(f_i(y,x)) \) (empirical expectation i.e. feature frequency)

• Tagging
\[ p(t,x) = \frac{1}{Z} e^{\sum_{i=1..N} \lambda_i f_i(t,x)} \] (\( \lambda_0 \) might be extra: cf. \( \mu \) in AR)
  • \( t \in \text{Tagset}, \)
  • \( x \sim \text{context (words and tags alike; say, up to three positions R/L)} \)
Features for Tagging

• Context definition
  – two words back and ahead, two tags back, current word:
    • \( x_i = (w_{i-2}, t_{i-2}, w_{i-1}, t_{i-1}, w_i, w_{i+1}, w_{i+2}) \)
  – features may ask any information from this window
    • e.g.:
      – previous tag is DT
      – previous two tags are PRP$ and MD, and the following word is “be”
      – current word is “an”
      – suffix of current word is “ing”
    • do not forget: feature also contains \( t_i \), the current tag:
      – feature #45: suffix of current word is “ing” & the tag is VBG \( \Leftrightarrow f_{45} = 1 \)
Feature Selection

• The PC\(^1\) way (see also yesterday’s class):
  – (try to) test all possible feature combinations
    • features may overlap, or be redundant; also, general or specific
      - impossible to select manually
  – greedy selection:
    • add one feature at a time, test if (good) improvement:
      – keep if yes, return to the pool of features if not
  – even this is costly, unless some shortcuts are made
    • see Berger & DPs for details

• The other way:
  – use some heuristic to limit the number of features
    • \(^1\)Politically (or, Probabilistically-stochastically) Correct
Limiting the Number of Features

• Always do (regardless whether you’re PC or not):
  – use contexts which appear in the training data (lossless selection)

• More or less PC, but entails huge savings (in the number of features to estimate $\lambda_i$ weights for):
  – use features appearing only $L$-times in the data ($L \sim 10$)
  – use $w_i$-derived features which appear with rare words only
  – do not use all combinations of context (this is even “LC$^1$”)
  – but then, use all of them, and compute the $\lambda_i$ only once using the Generalized Iterative Scaling algorithm

$^1$Linguistically Correct
Feature Examples (Context)

- From A. Ratnaparkhi (EMNLP, 1996, UPenn)
  - \( t_i = T, w_i = X \) (frequency \( c > 4 \)):
    - \( t_i = \text{VBG}, w_i = \text{selling} \)
  - \( t_i = T, w_i \) contains uppercase char (rare):
    - \( t_i = \text{NNP}, \text{tolower}(w_i) \neq w_i \)
  - \( t_i = T, t_{i-1} = Y, t_{i-2} = X \):
    - \( t_i = \text{VBP}, t_{i-2} = \text{PRP}, t_{i-1} = \text{RB} \)

- Other examples of possible features:
  - \( t_i = T, t_j \) is \( X \), where \( j \) is the closest left position where \( Y \)
    - \( t_i = \text{VBZ}, t_j = \text{NN}, Y \Leftrightarrow t_j \in \{\text{NNP}, \text{NNS}, \text{NN}\} \)
Feature Examples (Lexical/Unknown)

• From AR:
  – \( t_i = T \), suffix\( (w_i) = X \) (length \( X < 5 \)):
    • \( t_i = JJ \), suffix\( (w_i) = \text{eled} \) (traveled, leveled, ....)
  – \( t_i = T \), prefix\( (w_i) = X \) (length \( X < 5 \)):
    • \( t_i = JJ \), prefix\( (w_i) = \text{well-} \) (well-done, well-received,...)
  – \( t_i = T \), \( w_i \) contains hyphen:
    • \( t_i = JJ \), ‘-’ in \( w_i \) (open-minded, short-sighted,...)

• Other possibility, for example:
  – \( t_i = T \), \( w_i \) contains \( X \):
    • \( t_i = \text{NounPl} \), \( w_i \) contains umlaut (ä,ö,ü) (Wörter, Länge,...)
“Specialized” Word-based Features

• List of words with most errors (WSJ, Penn Treebank):
  – about, that, more, up, ...

• Add “specialized”, detailed features:
  – \( t_i = T, w_i = X, t_{i-1} = Y, t_{i-2} = Z \):
    • \( t_i = \text{IN}, w_i = \text{about}, t_{i-1} = \text{NNS}, t_{i-2} = \text{DT} \)
    – possible only for relatively high-frequency words

• Slightly better results (also, problems with inconsistent [test] data)
Maximum Entropy Tagging: Results

- Base experiment (133k words, < 3% unknown):
  - 96.31% word accuracy
- Specialized features added:
  - 96.49% word accuracy
- Consistent subset (training + test)
  - 97.04% word accuracy (97.13% w/specialized features)
    - Best in 2000; for details, see the AR paper
- Now: perceptron, ~97.4%
  - Collins 2002, Raab 2009