Maximum Entropy Models (for tagging)

Stephen Clark

Lent 2013

Machine Learning for Language Processing: Lecture 2

MPhil in Advanced Computer Science
 Discriminative Models

- Classification requires the class-posterior $P(\omega_j|x)$
  - can just directly model the posterior distribution
  - avoids the complexity of modelling the joint distribution $P(x, \omega_j)$

- Form of model called a discriminative model

- Many debates of generative versus discriminative models:
  - discriminative model criterion more closely related to classification process
  - not dependent on generative process being correct
  - joint distribution can be very complicated to accurately model
  - only final posterior distribution needs to be a valid distribution
Recap on Tagging

- Find the best tag sequence *given the sentence* (conditional probability):

  \[
  \text{argmax}_{t_1 \ldots t_n} p(t_1 \ldots t_n \mid w_1 \ldots w_n)
  \]

- Alternatively maximise \( p(t_1 \ldots t_n, w_1 \ldots w_n) \) (joint probability):

  \[
  \text{argmax}_{t_1 \ldots t_n} p(t_1 \ldots t_n \mid w_1 \ldots w_n) = \text{argmax}_{t_1 \ldots t_n} \frac{p(t_1 \ldots t_n, w_1 \ldots w_n)}{p(w_1 \ldots w_n)} = \text{argmax}_{t_1 \ldots t_n} p(t_1 \ldots t_n, w_1 \ldots w_n)
  \]
Recap on Markov Model Tagging

• Maximise the joint probability:

\[ p(t_1 \ldots t_n, w_1 \ldots w_n) = p(t_1 \ldots t_n)p(w_1 \ldots w_n|t_1 \ldots t_n) \]

• Tag sequence probability (first order Markov Model):

\[ p(t_1 \ldots t_n) \approx p(t_1)p(t_2|t_1)p(t_3|t_2) \ldots p(t_n|t_{n-1}) \]

• Word sequence probability (given the tags):

\[ p(w_1 \ldots w_n|t_1 \ldots t_n) \approx p(w_1|t_1)p(w_2|t_2) \ldots p(w_n|t_n) \]
Problems with Markov Model Taggers

• unreliable zero or very low counts
  – does a zero count indicate an impossible event?
  $\Rightarrow$ smoothing the counts solves this problem

• Words not seen in the data are especially problematic
  $\Rightarrow$ would like to include word internal information
  e.g. capitalisation or suffix information

• Cannot incorporate diverse pieces of evidence for predicting tags
  e.g. global document information
Feature-based Models

- Features encode evidence from the context for a particular tag:

  (title caps, NNP)  
  Citibank, Mr.

  (suffix -ing, VBG)  
  running, cooking

  (next word Inc., I-ORG)  
  Lotus Inc.

  (previous word said, I-PER)  
  said Mr. Vinken
Complex Features

• Features can be arbitrarily complex
  – e.g. document level features
    \[(\text{document} = \text{cricket} \& \text{current word} = \text{Lancashire, I-ORG})\]
    \[\implies \text{hopefully tag Lancashire as I-ORG not I-LOC}\]

• Features can be combinations of atomic features
  – \((\text{current word} = \text{Miss} \& \text{next word} = \text{Selfridges, I-ORG})\)
    \[\implies \text{hopefully tag Miss as I-ORG not I-PER}\]

• Features are not assumed to be (conditionally) independent (given the label)
  – unlike the Naive Bayes classifier
Feature-based Tagging

• How do we incorporate features into a probabilistic tagger?

• Hack the Markov Model tagger to incorporate features

• Maximum Entropy (MaxEnt) Tagging
  – principled way of incorporating features
  – requires sophisticated estimation method
Features in Maximum Entropy Models

• Features encode elements of the context $C$ useful for predicting tag $t$

• Features are binary valued functions, e.g.

$$f_i(C, t) = \begin{cases} 
1 & \text{if } \text{word}(C) = \text{Moody} \& t = I-\text{ORG} \\
0 & \text{otherwise}
\end{cases}$$

• $\text{word}(C) = \text{Moody}$ is a *contextual predicate*

• Features determine (contextual_predicate, tag) pairs
The Model

\[ p(t|C') = \frac{1}{Z(C')} \exp \left( \sum_{i=1}^{n} \lambda_i f_i(C, t) \right) \]

- \( f_i \) is a feature
- \( \lambda_i \) is a weight (large value implies informative feature)
- \( Z(C') \) is a normalisation constant ensuring a proper probability distribution
- Also known as a log-linear model
- Makes no independence assumptions about the features
- Can be used as a general classifier (outside of tagging, e.g. text classification)
Tagging with Maximum Entropy Models

- The conditional probability of a tag sequence \( t_1 \ldots t_n \) is

\[
p(t_1 \ldots t_n|w_1 \ldots w_n) \approx \prod_{i=1}^{n} p(t_i|C_i)
\]

given a sentence \( w_1 \ldots w_n \) and contexts \( C_1 \ldots C_n \)

- The context includes previously assigned tags (for a fixed history)

- Beam search or Viterbi is used to find the most probable sequence (Ratnaparkhi, 1996)

- Later in the course we will see an alternative (more principled) conditional formulation of the global probability (in the form of CRFs)
Model Estimation

\[ p(t|C) = \frac{1}{Z(C)} \exp \left( \sum_{i=1}^{n} \lambda_i f_i(C, t) \right) \]

- Model estimation involves setting the weight values \( \lambda_i \)
- The model should reflect the data
  \( \implies \) use the data to *constrain* the model
- What form should the constraints take?
  \( \implies \) constrain the *expected value* of each feature \( f_i \)
The Constraints

\[ E_p f_i = \sum_{C,t} p(C,t) f_i(C,t) = K_i \]

- Expected value of each feature must satisfy some constraint \( K_i \)

- A natural choice for \( K_i \) is the average empirical count:

\[ K_i = E_p f_i = \frac{1}{N} \sum_{j=1}^{N} f_i(C_j, t_j) \]

derived from the training data \((C_1, t_1), \ldots, (C_N, t_N)\)
Choosing the Maximum Entropy Model

- The constraints do not uniquely identify a model

- From those models satisfying the constraints: choose the Maximum Entropy model

- Conditional entropy of a model $p$:

$$H(p) = - \sum_{C,t} \hat{p}(C)p(t|C) \log p(t|C)$$
The Maximum Entropy Model

- The maximum entropy model is the \textit{most uniform model} \implies makes no assumptions in addition to what we know from the data

- MaxEnt model is also the \textit{Maximum Likelihood Log-Linear} model

- Set the weights to give the MaxEnt model satisfying the constraints \implies use \textit{Generalised Iterative Scaling} (GIS)
Generalised Iterative Scaling (GIS)

- Set $\lambda_i^{(0)}$ equal to some arbitrary value (e.g. zero)

- Repeat until convergence:

$$
\lambda_i^{(t+1)} = \lambda_i^{(t)} + \frac{1}{C} \log \frac{E \tilde{p} f_i}{E_{p(t)} f_i}
$$

where

$$
C = \max_{x,y} \sum_{i=1}^{n} f_i(x, y)
$$

- Many formulations of GIS specify the need for a “correction feature”, but see Curran and Clark (2003)
Smoothing

• Models which satisfy the constraints exactly tend to overfit the data

• In particular, empirical counts for low frequency features can be unreliable
  – often leads to very large weight values

• Common smoothing technique is to ignore low frequency features
  – but low frequency features may be important

• Use a prior distribution on the parameters
  – encodes our knowledge that weight values should not be too large
Smoothing

- Standard technique is to use a *Gaussian prior* over the parameters (Chen and Rosenfeld 1999)
  - penalises models with extreme feature weights

- This is a form of *maximum a posteriori* (MAP) estimation

- Can be thought of as relaxing the model constraints - requires a modification to the update rule

- Can also be thought of as a form of *regularisation*
Pos Tagger Features

- The tagger uses binary valued features, e.g.

\[
f_i(x, y) = \begin{cases} 
1 & \text{if } \text{word}(x) = \text{the} \land y = \text{DT} \\
0 & \text{otherwise}
\end{cases}
\]

- \text{word}(x) = \text{the} is a contextual predicate

- Contextual predicates:

\[
\begin{align*}
t_{i-1} &= X & \text{previous tag history} \\
t_{i-2}t_{i-1} &= XY & \text{previous two tags history} \\
w_i &= X & \text{current word} \\
w_{i-1} &= X & \text{previous word} \\
w_{i-2} &= X & \text{previous previous word} \\
w_{i+1} &= X & \text{next word} \\
w_{i+2} &= X & \text{next next word}
\end{align*}
\]
Pos Tagger Features for Rare Words

- These predicates apply to words seen less than 5 times in the data

  \[ \begin{align*}
  &X \text{ is prefix of } w_i, \ |X| \leq 4 \\
  &X \text{ is suffix of } w_i, \ |X| \leq 4 \\
  &w_i \text{ contains a digit} \\
  &w_i \text{ contains uppercase char} \\
  &w_i \text{ contains a hyphen}
  \end{align*} \]

- Otherwise the current word predicate applies
Evaluation Measures

Acc  overall per-word accuracy

Uword accuracy on previously unseen words

Utag accuracy on previously unseen word-tag pairs

Amb accuracy on words seen with more than one tag in the Treebank

• Training data sections 2-21, development section 00, testing section 23 from the WSJ Penn Treebank
Results on the Development Set

<table>
<thead>
<tr>
<th>Tagger</th>
<th>ACC</th>
<th>UWORD</th>
<th>UTAG</th>
<th>AMB</th>
</tr>
</thead>
<tbody>
<tr>
<td>MXPOST</td>
<td>96.59</td>
<td>85.81</td>
<td>30.04</td>
<td>94.82</td>
</tr>
<tr>
<td>BASE</td>
<td>96.58</td>
<td>85.70</td>
<td>29.28</td>
<td>94.82</td>
</tr>
<tr>
<td>SMOOTHED</td>
<td>96.75</td>
<td>86.74</td>
<td>33.08</td>
<td>95.06</td>
</tr>
</tbody>
</table>

- MXPOST is Ratnaparkhi’s original tagger (feature cutoff 5, no smoothing)
- Gaussian smoothing improves results
### Results with varying feature cut-offs

<table>
<thead>
<tr>
<th>Cut-off</th>
<th>ACC</th>
<th>UWORD</th>
<th>UTAG</th>
<th>AMB</th>
</tr>
</thead>
<tbody>
<tr>
<td>≥ 1</td>
<td>96.82</td>
<td>87.20</td>
<td>30.80</td>
<td>95.07</td>
</tr>
<tr>
<td>≥ 2</td>
<td>96.77</td>
<td>87.02</td>
<td>31.18</td>
<td>95.00</td>
</tr>
<tr>
<td>≥ 3</td>
<td>96.72</td>
<td>86.62</td>
<td>31.94</td>
<td>94.94</td>
</tr>
<tr>
<td>≥ 4</td>
<td>96.72</td>
<td>87.08</td>
<td>34.22</td>
<td>94.96</td>
</tr>
</tbody>
</table>

- No cutoff gives best results
- Gaussian smoothing allows all features to be used without overfitting
## Results on the Test Set

<table>
<thead>
<tr>
<th>Tagger</th>
<th>ACC</th>
<th>UWORD</th>
<th>UTAG</th>
<th>AMB</th>
</tr>
</thead>
<tbody>
<tr>
<td>MXPOST</td>
<td>97.05</td>
<td>83.63</td>
<td>30.20</td>
<td>95.44</td>
</tr>
<tr>
<td>C&amp;C</td>
<td>97.27</td>
<td>85.21</td>
<td>28.98</td>
<td>95.69</td>
</tr>
</tbody>
</table>

## Cross-validation results

<table>
<thead>
<tr>
<th>Tagger</th>
<th>ACC</th>
<th>$\sigma$</th>
<th>UWORD</th>
<th>UTAG</th>
<th>AMB</th>
</tr>
</thead>
<tbody>
<tr>
<td>MXPOST</td>
<td>96.72</td>
<td>0.12</td>
<td>85.50</td>
<td>32.16</td>
<td>95.00</td>
</tr>
<tr>
<td>TNT</td>
<td>96.48</td>
<td>0.13</td>
<td>85.31</td>
<td>0.00</td>
<td>94.26</td>
</tr>
<tr>
<td>C&amp;C</td>
<td>96.86</td>
<td>0.12</td>
<td>86.43</td>
<td>30.42</td>
<td>95.08</td>
</tr>
</tbody>
</table>
Performance

- Training takes around 10 minutes for 100 GIS iterations
- Tagging is very fast (around 100,000 words per second)
Named Entity Tagging

- Language independent NER for CoNLL-02, CoNLL-03 competitions
- English, German, Dutch
- LOC, PER, ORG, MISC, O
## Contextual Predicates used by the NE tagger

<table>
<thead>
<tr>
<th>Condition</th>
<th>Contextual predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(w_i) &lt; 5$</td>
<td>$X$ is prefix/suffix of $w_i$, $</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$w_i = X$  $w_{i-1} = X, w_{i-2} = X$  $w_{i+1} = X, w_{i+2} = X$</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$\text{POS}<em>i = X$  $\text{POS}</em>{i-1} = X, \text{POS}<em>{i-2} = X$  $\text{POS}</em>{i+1} = X, \text{POS}_{i+2} = X$</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$\text{NE}<em>{i-1} = X$  $\text{NE}</em>{i-2}\text{NE}_{i-1} = XY$</td>
</tr>
</tbody>
</table>
### Additional Contextual Predicates

<table>
<thead>
<tr>
<th>Condition</th>
<th>Contextual predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(w_i) &lt; 5$</td>
<td>$w_i$ contains period</td>
</tr>
<tr>
<td></td>
<td>$w_i$ contains punctuation</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is only digits</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is a number</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is {upper,lower,title,mixed} case</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is alphanumatic</td>
</tr>
<tr>
<td></td>
<td>length of $w_i$</td>
</tr>
<tr>
<td></td>
<td>$w_i$ has only Roman numerals</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is an initial (x.)</td>
</tr>
<tr>
<td></td>
<td>$w_i$ is an acronym (ABC, A.B.C.)</td>
</tr>
</tbody>
</table>
## Additional Contextual Predicates

<table>
<thead>
<tr>
<th>Condition</th>
<th>Contextual predicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall w_i$</td>
<td>memory NE tag for $w_i$</td>
</tr>
<tr>
<td></td>
<td>unigram tag of $w_{i+1}$</td>
</tr>
<tr>
<td></td>
<td>unigram tag of $w_{i+2}$</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$w_i$ in a gazetteer</td>
</tr>
<tr>
<td></td>
<td>$w_{i-1}$ in a gazetteer</td>
</tr>
<tr>
<td></td>
<td>$w_{i+1}$ in a gazetteer</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$w_i$ not lowercase and $f_{lc} &gt; f_{uc}$</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>unigrams of word type</td>
</tr>
<tr>
<td></td>
<td>bigrams of word types</td>
</tr>
<tr>
<td></td>
<td>trigrams of word types</td>
</tr>
</tbody>
</table>
The Word Type Features

• Moody $\Rightarrow$ Aa

• A.B.C. $\Rightarrow$ A.A.A.

• 1,345.00 $\Rightarrow$ 0,0.0

• Mr. Smith $\Rightarrow$ Aa. Aa
## Baseline Results on English Data

<table>
<thead>
<tr>
<th>English</th>
<th>Precision</th>
<th>Recall</th>
<th>$F_{\beta=1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCATION</td>
<td>90.78%</td>
<td>90.58%</td>
<td>90.68%</td>
</tr>
<tr>
<td>MISC</td>
<td>85.80%</td>
<td>81.24%</td>
<td>83.45%</td>
</tr>
<tr>
<td>ORGANISATION</td>
<td>82.24%</td>
<td>80.09%</td>
<td>81.15%</td>
</tr>
<tr>
<td>PERSON</td>
<td>92.02%</td>
<td>92.67%</td>
<td>92.35%</td>
</tr>
<tr>
<td>OVERALL</td>
<td>88.53%</td>
<td>87.41%</td>
<td>87.97%</td>
</tr>
</tbody>
</table>

- Reuters newswire data
- 200,000 words training, 50,000 words test
### Full System Results on English Data

<table>
<thead>
<tr>
<th>English</th>
<th>Precision</th>
<th>Recall</th>
<th>$F_{\beta=1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCATION</td>
<td>91.75%</td>
<td>93.20%</td>
<td>92.47%</td>
</tr>
<tr>
<td>MISC</td>
<td>88.34%</td>
<td>82.97%</td>
<td>85.57%</td>
</tr>
<tr>
<td>ORGANISATION</td>
<td>83.54%</td>
<td>85.53%</td>
<td>84.52%</td>
</tr>
<tr>
<td>PERSON</td>
<td>94.26%</td>
<td>95.39%</td>
<td>94.82%</td>
</tr>
<tr>
<td>OVERALL</td>
<td><strong>90.15%</strong></td>
<td><strong>90.56%</strong></td>
<td><strong>90.35%</strong></td>
</tr>
</tbody>
</table>

- Good NER performance requires a wide range of features
- One of the best performing systems in CoNLL-03
German Results

<table>
<thead>
<tr>
<th>German</th>
<th>Precision</th>
<th>Recall</th>
<th>$F_{\beta=1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCATION</td>
<td>70.91%</td>
<td>71.11%</td>
<td>71.01%</td>
</tr>
<tr>
<td>MISC</td>
<td>68.51%</td>
<td>46.12%</td>
<td>55.13%</td>
</tr>
<tr>
<td>ORGANISATION</td>
<td>68.43%</td>
<td>50.19%</td>
<td>57.91%</td>
</tr>
<tr>
<td>PERSON</td>
<td>88.04%</td>
<td>72.05%</td>
<td>79.25%</td>
</tr>
<tr>
<td>OVERALL</td>
<td><strong>75.61%</strong></td>
<td><strong>62.46%</strong></td>
<td><strong>68.41%</strong></td>
</tr>
</tbody>
</table>

- German newspaper text (200k training, 50k test)

- German is harder than English (capitalisation)
Conclusion

- Tagging (and other NLP tasks) require a wide range of features for good performance

- Maximum entropy models (with Gaussian smoothing) can handle a large number of diverse features

- GIS is relatively simple and performs well for maximum entropy taggers
Other Work

- MaxEnt (CRF) models for wide-coverage CCG parsing (Clark & Curran, 2007)
- Statistical parsing requires a wide range of features for good performance
- Generative parsing models lack the flexibility of maximum entropy models
- Training is computationally expensive and requires dynamic programming methods
- GIS is too slow for parsing models - use more general numerical optimisation methods
References


