Learning CRFs with Hierarchical Features: An Application to Go

Scott Sanner – University of Toronto
Thore Graepel
Ralf Herbrich
Tom Minka

(with thanks to David Stern and Mykel Kochenderfer)
The Game of Go

• Started about 4000 years ago in ancient China
• About 60 million players worldwide
• 2 Players: Black and White
• Board: 19×19 grid
• Rules:
  – Turn: One stone placed on vertex.
  – Capture.
• Aim: Gather territory by surrounding it
Territory Prediction

• Goal: predict territory distribution given board...

• How to predict territory?
  – Could use search:
    • Full $\alpha$-\(\beta\) impractical (avg. 180 moves/turn, >100 moves/game)
    • Monte Carlo averaging a reasonable estimator, but costly
  – We learn to directly predict territory:
    • Learn $P(\vec{s} | \vec{c})$ from expert data
Talk Outline

• Hierarchical pattern features

• Independent classifiers
  – What is best way to combine features?

• CRF models
  – Coupling factor model (w/ patterns)
  – Exact inference intractable
  – What is best training / inference method to circumvent intractability?

• Evaluation and Conclusions
Hierarchical Patterns

• Centered on a single position

• Exact config. of stones

• Fixed match region (template)

• 8 nested templates

• 3.8 million patterns mined
Models

(a) Independent pattern-based classifiers

(b) (Coupling) CRF

(c) Pattern CRF

Vertex Variables: $s_i, c_i$

Unary pattern-based factors:
$$
\psi_i(s_i = +1, \overline{c}) = \exp \left( \sum_{\overline{\pi} \in \overline{\Pi}} \lambda_{\overline{\pi}} \cdot \mathbb{I}_{\overline{\pi}}(\overline{c}, i) \right)
$$

Coupling factors:
$$
\psi_{(i,j)}(s_i, s_j, c_i, c_j) = \exp \left( \sum_{k=1}^{36} \lambda_k \cdot \mathbb{I}_k(s_i, s_j, c_i, c_j, k) \right)
$$
Independent Pattern-based Classifiers
Inference and Training

• Up to 8 patterns may match at any vertex

• Which pattern to use?
  – Smallest pattern
  – Largest pattern

• Or, combine all patterns:
  – Logistic regression
  – Bayesian model averaging…

\[
\psi_i(s_i = +1, \bar{c}) = P(s_i | \tilde{\pi}_{\text{min}}(\bar{c}, i))
\]
\[
\psi_i(s_i = +1, \bar{c}) = P(s_i | \tilde{\pi}_{\text{max}}(\bar{c}, i))
\]
\[
\psi_i(s_i = +1, \bar{c}) = \exp \left( \sum_{\tilde{\pi} \in \tilde{\Pi}} \lambda_{\tilde{\pi}} \cdot I_{\tilde{\pi}}(\bar{c}, i) \right)
\]
Bayesian Model Averaging

• Bayesian approach to combining models:

\[
P(s_j|\bar{c}, D) = \sum_{\tau \in \mathcal{Y}} P(s_j|\tau, \bar{c}, D)P(\tau|\bar{c}, D)
\]

• Now examine the model “weight”:

\[
P(\tau|\bar{c}, D) = \frac{P(D|\tau, \bar{c})P(\tau|\bar{c})}{\sum_{\tau \in \mathcal{Y}} P(D|\tau, \bar{c})P(\tau|\bar{c})}
\]

• Model \( \tau \) must apply to all data!
Hierarchical Tree Models

• Arrange patterns into decision trees $\tau$:

$$\tilde{\pi}_1 = \bullet$$

$$\tilde{\pi}_2 = \begin{array}{c} \circ \\ \bullet \\ \end{array}$$

$$\tilde{\pi}_3 = \begin{array}{c} \circ \\ \bullet \\ \circ \\ \bullet \\ \end{array}$$

$\tau_1$

$\pi_1$ $\pi_2$ $
\theta_{\tilde{\pi}_1\tau_1}$ $\theta_{\emptyset \tau_1}$

$\tau_2$

$\pi_1$ $\pi_2$ $\theta_{\emptyset \tau_2}$

$\tau_3$

$\pi_1$ $\pi_2$ $\theta_{\emptyset \tau_3}$

• Model $\tau$ provides predictions on all data
Coupling CRF & Pattern CRF
Inference and Training

• Inference
  – Intractable for 19x19 grids
  – Loopy BP is biased
    • but an option
  – Sampling is unbiased
    • but much slower

• Training
  – Max likelihood requires inference!

\[
\frac{\partial l}{\partial \lambda_j} = \sum_{d \in D} \left( \mathbb{I}_j(\tilde{s}^{(d)}, \tilde{c}^{(d)}) - \sum_{\tilde{s}} \mathbb{I}_j(\tilde{s}, \tilde{c}^{(d)}) P(\tilde{s} | \tilde{c}^{(d)}) \right)
\]
  – Other approximate methods…
Pseudolikelihood

• Standard log likelihood:

\[ l(\tilde{\lambda}) = \sum_{d \in D} \log P(s^{(d)} | \tilde{c}^{(d)}) \]

• Pseudo log-likelihood (clamp Markov blanket):

\[ pl(\tilde{\lambda}) = \sum_{d \in D} \sum_{f \in \mathcal{F}} \log P(s_{f}^{(d)} | \tilde{c}_{f}^{(d)}, MB_{\mathcal{F}}(f)^{(d)}) \]

• Then inference during training is purely local
• Long range effects captured in data
• Note: only valid for training
  – in presence of fully labeled data
Local Training

• Break CRF into max likelihood trained pieces…

• Piecewise:

• Shared Unary Piecewise:
Evaluation
Models & Algorithms

• Model & algorithm specification:
  – Model / Training (/ Inference, if not obvious)

• Models & algorithms evaluated:
  – Indep / {Smallest, Largest} Pattern
  – Indep / BMA-Tree {Uniform, Exp}
  – Indep / Log Regr
  – CRF / ML Loopy BP (/ Swendsen-Wang)
  – Pattern CRF / Pseudolikelihood (Edge)
  – Pattern CRF / (S. U.) Piecewise
  – Monte Carlo
Training Time

- Approximate time for various models and algorithms to reach convergence:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Training Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indep / Largest Pattern</td>
<td>&lt; 45 min</td>
</tr>
<tr>
<td>Indep / BMA-Tree</td>
<td>&lt; 45 min</td>
</tr>
<tr>
<td>Pattern CRF / Piecewise</td>
<td>~ 2 hrs</td>
</tr>
<tr>
<td>Indep / Log Regr</td>
<td>~ 5 hrs</td>
</tr>
<tr>
<td>Pattern CRF / Pseudolikelihood</td>
<td>~ 12 hrs</td>
</tr>
<tr>
<td>CRF / ML Loopy BP</td>
<td>&gt; 2 days</td>
</tr>
</tbody>
</table>
Inference Time

- Average time to evaluate $P(\vec{s} | \vec{c})$ for various models and algorithms on a 19x19 board:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Inference Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indep / Sm. &amp; Largest Pattern</td>
<td>1.7 ms</td>
</tr>
<tr>
<td>Indep / BMA-Tree &amp; Log Regr</td>
<td>6.0 ms</td>
</tr>
<tr>
<td>CRF / Loopy BP</td>
<td>101.0 ms</td>
</tr>
<tr>
<td>Pattern CRF / Loopy BP</td>
<td>214.6 ms</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>2,967.5 ms</td>
</tr>
<tr>
<td>CRF / Swend.-Wang Sampling</td>
<td>10,568.7 ms</td>
</tr>
</tbody>
</table>
Performance Metrics

• Vertex Error: (classification error)

\[ \frac{1}{|G|} \sum_{i=1}^{\mid G \mid} \mathbb{I}(\text{sgn}(E_{P}(\tilde{s} | \tilde{c}(d))[s_i]) \neq \text{sgn}(s^{(d)}_i)) \]

• Net Error: (score error)

\[ \frac{1}{2|G|} \mid \sum_{i=1}^{\mid G \mid} E_{P}(\tilde{s} | \tilde{c}(d))[s_i] - \sum_{i=1}^{\mid G \mid} s^{(d)}_i \mid \]

• Log Likelihood: (model fit)

\[ \log P(\tilde{s}^{(d)} | \tilde{c}(d)) \]
Performance Tradeoffs I

Net Error vs. Vertex Error Tradeoff

- Indep / Smallest Pattern
- Indep / Largest Pattern
- Indep / BMA-Tree Uniform
- Indep / BMA-Tree Exp
- Indep / Log Regr
- CRF / ML Loopy BP
- CRF / ML Loopy BP / Swendsen-Wang
- Pattern CRF / Pseudolikelihood Edge
- Pattern CRF / Pseudolikelihood
- Pattern CRF / Piecewise
- Pattern CRF / S.U. Piecewise
- Monte Carlo
Why is Vertex Error better for CRFs?

• Coupling factors help realize stable configurations

• Compare previous unary-only independent model to unary and coupling model:
  – Independent models make inconsistent predictions
  – Loopy BP smoothes these predictions (but too much?)
Why is Net Error worse for CRFs?

• Use sampling to examine bias of Loopy BP
  – Unbiased inference in limit
  – Can run over all test data but still too costly for training

• Smoothing gets rid of local inconsistencies

• But errors reinforce each other!
Bias of Local Training

- Problems with Piecewise training:
  - Very biased when used in conjunction with Loopy BP
  - Predictions still good, just saturated
  - Accounts for poor Log Likelihood & Net Error…
Performance Tradeoffs II

Net Error vs. -Log Likelihood Tradeoff

Indep / Smallest Pattern
Indep / Largest Pattern
Indep / BMA-Tree Uniform
Indep / BMA-Tree Exp
Indep / Log Regr
CRF / ML Loopy BP
CRF / ML Loopy BP / Swendsen-Wang
Pattern CRF / Pseudolikelihood Edge
Pattern CRF / Pseudolikelihood
Pattern CRF / Piecewise
Pattern CRF / S.U. Piecewise
Monte Carlo
Conclusions

Two general messages:

(1) CRFs vs. Independent Models:
   • CRFs should theoretically be better
   • However, time cost is high
   • Can save time with approximate training / inference
   • But then CRFs may perform worse than independent classifiers – depends on metric

(2) For Independent Models:
   • Problem of choosing appropriate neighborhood can be finessed by Bayesian model averaging