CRFs for DOTA 2 Team Prediction

Nabeel Sarwar and Brian Kelly

What is the problem

- Given an enemy team, pick a team of five heroes that maximizes the likelihood of a victory
- The problem is semi-supervised because we seek to evaluate team quality relative to an enemy team, but lack a direct metric
- We use the probability of victory as a proxy for relative team quality

DOTA 2

- DOTA 2 is an massive online battle arena (MOBA) game produced by Valve
- Two teams
  - 5 on each team
  - Each person selects a character (hero or champion)
- In order to pick a winning team, players will balance character archetypes (carry, tank, etc.), hero synergies and the ability to neutralize the enemy team
- Predicting most common hero (choosing mode) for each position ignores both synergies in certain situations and ability to neutralize the opposing team

Short little clip

https://youtu.be/yBEidvm_tZQ?t=32m20s

Previous Related Work

- Conley and Perry construct a logistic regression model based on win statistics
  - Does not account for pairwise character interactions
  - Does not account for asymmetry in map
- Yu and colleagues trained a bi-LSTM on professional games to optimize the entire team based on team cohesion
- OpenAI Five is a end-to-end system that selects five heroes and plays them against five human players
  - Can beat professional teams
  - AlphaGo of MOBA

Conditional Random Fields (CRFs)

- A **CRF** is a Markov Random Field (MRF) where the clique potentials are conditioned on the input features (Murphy, 684)

$$p(y|x,w) = \frac{1}{Z(x,w)} \prod_{c} \psi_c(y_c|x,w)$$ where $$\psi_c(\mathbf{y}_c|\mathbf{x},\mathbf{w}) = \exp(\mathbf{w}_c^T \phi(\mathbf{x},\mathbf{y}_c))$$

- For Pairwise CRF: $$p(\mathbf{y}|\mathbf{x},\mathbf{w}) = \frac{1}{Z(\mathbf{w},\mathbf{x})} \exp\left(\mathbf{w}^T \phi(\mathbf{y},\mathbf{x})\right)$$
- Advantages over MRFs:
  - Discriminative rather than generative
  - Clique potentials can be made data-dependent
- Disadvantages
  - Requires labeled data
  - Slower to train

Why not model CRF as an MRF?

- Avoid building a generative model that may have a complex parameter space
- Leverage the discriminative nature of CRF and more explicitly model potentials based on data
- CRF allows us to model potentials using weights and features
- Avoid wasting resources modeling the enemy team that is always observed

Training CRFs (Gradient Descent)

- Training is slow
- The gradient for MRFs is given by

$$\frac{\partial \ell}{\partial \boldsymbol{\theta}_c} = \left[\frac{1}{N}\sum_i \boldsymbol{\phi}_c(\mathbf{y}_i)\right] - \mathbb{E}\left[\boldsymbol{\phi}_c(\mathbf{y})\right]$$

- Similarly, the gradient for CRFs is given by (ignoring regularization)

$$\frac{\partial \ell}{\partial \mathbf{w}_c} = \frac{1}{N}\sum_i \left[\boldsymbol{\phi}_c(\mathbf{y}_i, \mathbf{x}_i) - \mathbb{E}\left[\boldsymbol{\phi}_c(\mathbf{y}, \mathbf{x}_i)\right]\right]$$

Problems with CRF

- Training involves expectation
    - Inference needed to compute expected sufficient Stats
    - Expectation can be approximated by sampling
    - Would need to fit new CRF at each epoch
- What if we just Compute the MAP estimate?
    - Simpler than computing marginals
    - Use structured output classifier using fast MAP solving
- Find weight of CRF through Structural Support Vector Machine

Structured SVM

- We want a more efficient method that learns the clique weights
- Structured SVM (SSVM) is a generalization of SVM that allows training for a general set of structured output labels
- Probabilities are of the same form

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w}^T \phi(\mathbf{x}, \mathbf{y}))}{Z(\mathbf{x}, \mathbf{w})}$$

- Similar loss (reduces to regularized CRF loss with 0-1 Loss)

$$-\log p(\mathbf{w}) + \sum_{i=1}^{N} \log \left[\sum_{\mathbf{y}} L(\mathbf{y}_i, \mathbf{y}) p(\mathbf{y}|\mathbf{x}_i, \mathbf{w}) \right]$$

Why are SSVMs Viable way to estimate weights?

- Map/Reshape Clique Features/Potentials to the Joint Features of SSVM
  - Can still use Inference algorithms of CRFs (like max-product)
- After finding approximate convex upper bound for SSVM loss and choosing Gaussian Spherical Prior for weights:

$$\vec{w} = \arg\min \frac{1}{2}\vec{w}^T\vec{w} + C\sum_i [\vec{w}^T\phi(x_i, y_i) - \max_{y\in\mathcal{Y}}[\Delta(y_i, y') + \vec{w}^T\phi(x_i, y')]]$$

- For CRF:

$$\vec{w} = \arg\min \frac{1}{2}\vec{w}^T\vec{w} + C\sum_n [-\vec{w}^T\phi(x_i, y_i) + \log(Z(\mathbf{x}, \mathbf{w}))]$$

- Become more similar with loss-augmentation

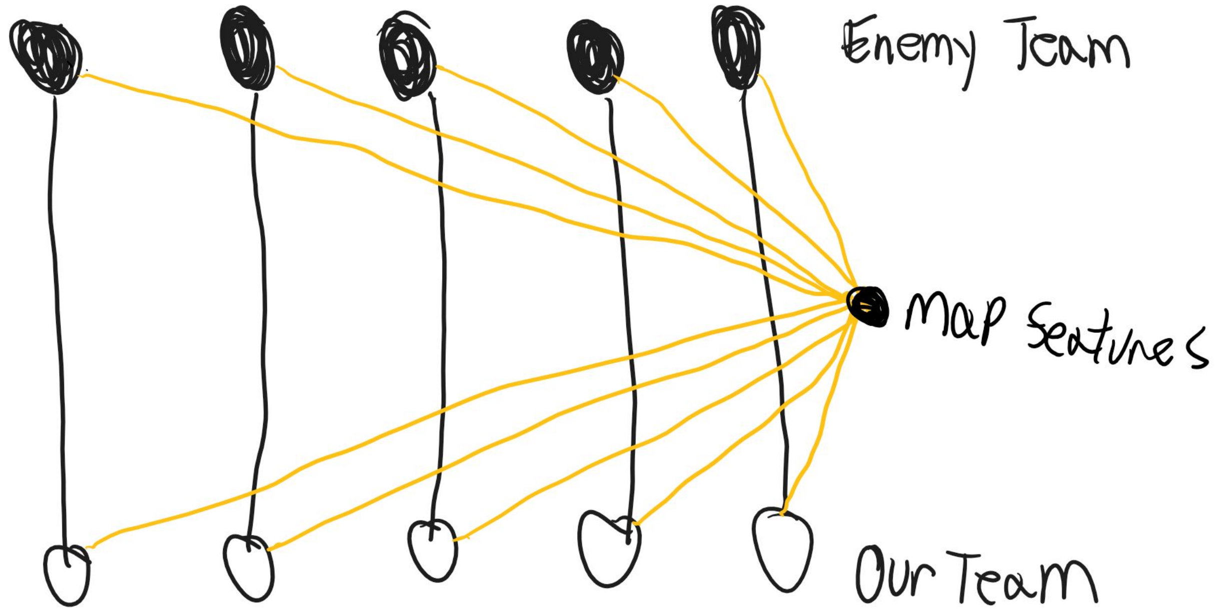
Data

- DOTA 2 match data was mined using OpenDOTA and Stratz APIs
- Only included matches with the most common 32 heroes
    - Make problem easier
- Match data was used to construct conditional unary and pairwise win rates between heroes
- Train: 300 matches
- Test: 59 matches

CRF Model For this Problem

- Nodes correspond to heroes
- Edges correspond pairwise interactions between heroes
- Node features: unary conditional win rates against 32 heroes
- Edge features: pairwise conditional win rates against pairs of the 32 heroes
- The CRF was trained using a One Slack and Frank Wolfe SSVM

Visualized CRF



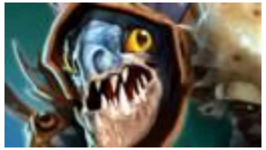
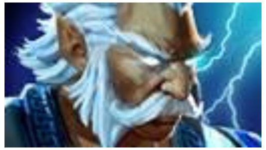
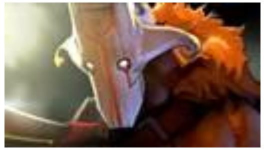
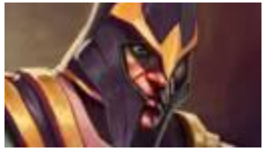
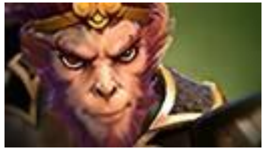
The image above ignores pairwise interactions for diagram simplicity and readability

Results

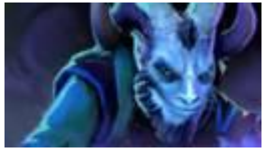
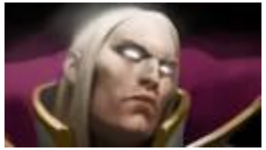
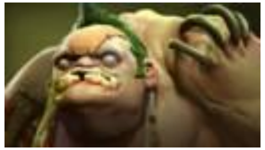
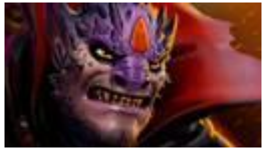
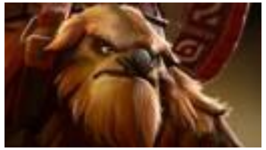
- Accuracies on test set
    - OneSlackSSVM: 0.075
    - FrankWolfeSSVM: 0.061
- Exact match is a poor indicator of model quality
    - Some teams that won may have been low quality
    - Given an enemy team, often there exists multiple effective teams
- Manually inspected 40 cases for each SSVM:
    - ~60% predicted teams were “similar” quality
    - ~30% predicted teams were “better” quality
    - ~10% predicted teams were “worse” quality teams
- OneSlackSSVM tended to suggest “better” teams but was worse in putting heroes in the right roles

Example Correction

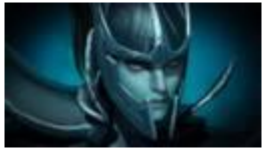
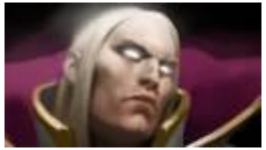
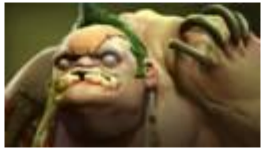
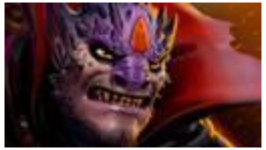
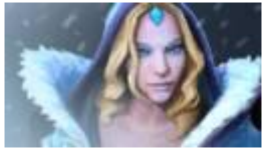
What the enemy has

What was chosen

What we suggested

Issues

- Data contained terrible winning teams
- Model often included a hero on both teams
- Only used pairwise interactions
- Hero role (e.g. carry, support) determined using gold farmed
- Model included multiples of a given hero role (e.g. two carries)

Potential Improvements

- Use data from only highly rated players
- Set potentials for heroes that were already chosen explicitly to 0
- Higher order features
- Generating the targets with more sophisticated proxies
- Hard code character roles and set potential for interaction terms to be 0 when sharing a role

References

Andreas C. Mueller, Sven Behnke. “PyStruct - Structured prediction in Python.” Journal of machine learning, 2014

Conley, Kevin, and Perry, Daniel. “How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2.”

OpenAI Five. Accessed 13 Nov 2019. https://openai.com/five/

Yu C., Zhu W., Sun Y. (2019) E-Sports Ban/Pick Prediction Based on Bi-LSTM Meta Learning Network. In: Sun X., Pan Z., Bertino E. (eds) Artificial Intelligence and Security. ICAIS 2019. Lecture Notes in Computer Science, vol 11632. Springer, Cham https://link.springer.com/chapter/10.1007/978-3-030-24274-9_9

Murphy, K. P. (2013). *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press. ISBN: 9780262018029 0262018020

Questions?