Factorization Ranking Model for Move Prediction in the Game of Go

Chenjun Xiao and Martin Müller
Department of Computing Science, University of Alberta
{ chenjun, mmuelle } @ualberta.ca

Computer Go

Ancient Chinese two player board game. Black and White take turns to place a single stone on an empty intersection on a Go board, in order to surround a larger total area of the board with their stones than the opponent by the end of the game.

Until very recently, one of the last classical board games that have not been mastered by computer programs. Expert knowledge can improve the strength of Go playing programs [1].

This work is built on top of the open source Fuego framework, mostly developed at University of Alberta.

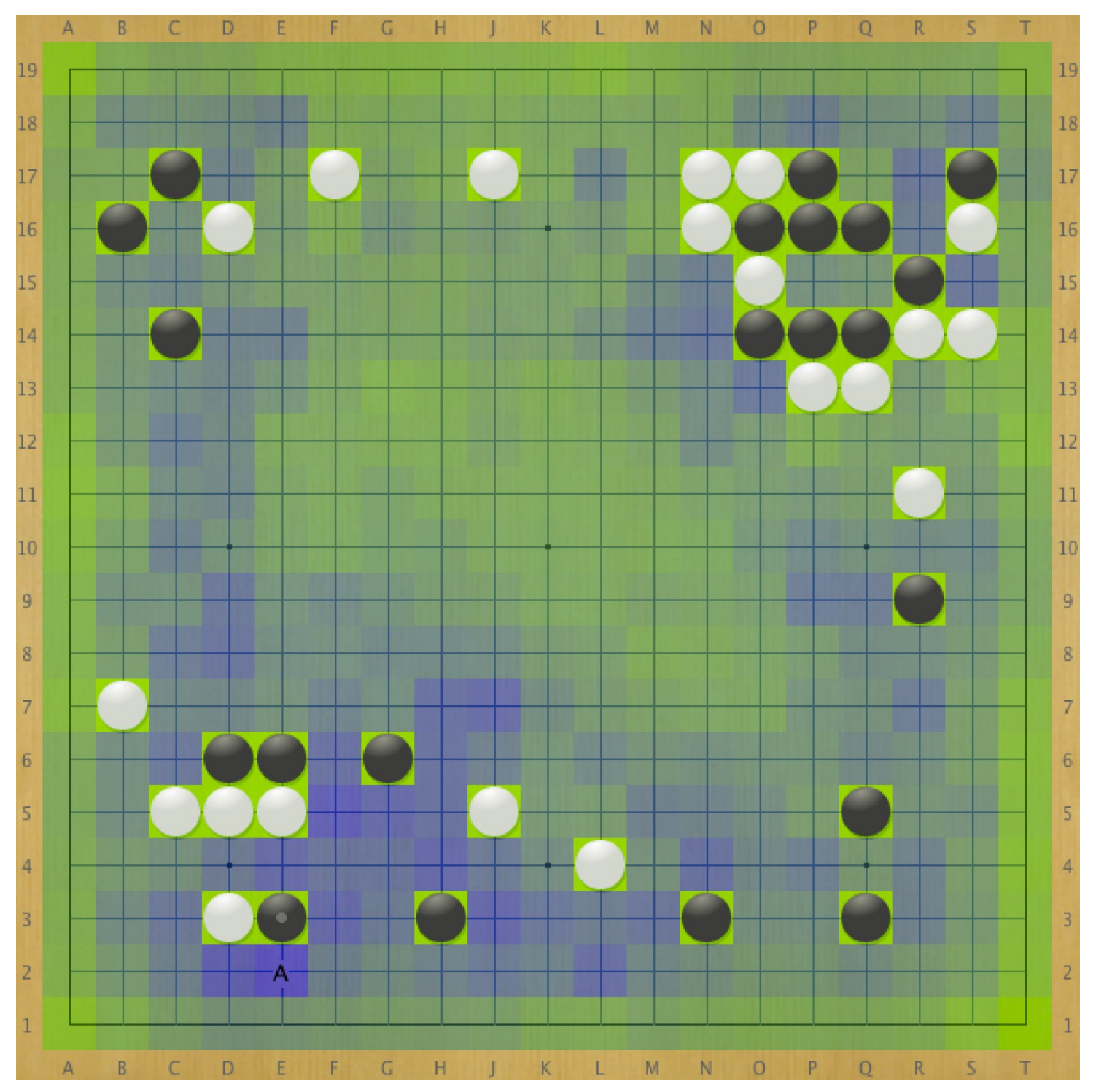

Figure 1: Move evaluation with learned features in Fuego. Game: January 2013, Chang Hao (Black), 9 dan professional vs. Tuo Jiaxi, 9 dan professional

Fast Move Prediction Problem in Go

Problem:

- Training data includes tens of thousands of games
- Each training case consists of possible moves in one game position, with one specified move chosen by the expert
- Moves are described by a set of features
- Supervised training of a model from game records to predict expert moves as accurately as possible

Question:
How to model each training case?

Two dominating fast algorithms:

- Elo Rating: consider move selection as a competition among groups of features.
- LRF: take the interaction between features within a group into account.

Our Solution:
Combine them: model interaction between features within a group, and model move selection as a group competition

Contrast: Deep neural networks are better move predictors, but several orders of magnitude slower.

Factorization Bradley Terry Model

Objective:
Modelling competition among groups of features by taking the interaction between features into account.

Description:
$\mathcal{F}$: set of features. $\mathbf{w} \in \mathbb{R}^{|\mathcal{F}|}$: strength of features,
$\mathbf{v} \in \mathbb{R}^{|\mathcal{F}| \times k}$: interaction features.
Strength of a group $\mathcal{G}$ is defined by:

$$E_{\mathcal{G}} = \sum_{f \in \mathcal{G}} w_f + \frac{1}{2} \sum_{f \in \mathcal{G}} \sum_{g \in \mathcal{G}, g \neq f} \langle v_f, v_g \rangle$$

Given N groups $\{\mathcal{G}^1, \dots, \mathcal{G}^N\}$,

$$P(\mathcal{G}^i \text{ wins}) = \frac{exp(E_{\mathcal{G}^i})}{\sum_{j=1}^{N} exp(E_{\mathcal{G}^j})}$$

Training:

- Model each training case with Factorization Bradley Terry (FBT) model
- Train the model using log-loss and l_2 regularization.

$$\min_{\mathbf{w} \in \mathbb{R}^{|\mathcal{F}|}, \mathbf{v} \in \mathbb{R}^{|\mathcal{F}| \times k}} \frac{1}{n} \sum_{i=1}^{n} l_i + \frac{\lambda_w}{2} ||\mathbf{w}||^2 + \frac{\lambda_v}{2} ||\mathbf{v}||^2$$

- Optimization with Stochastic Gradient Decent Gradient for a single model parameter $\theta \in \mathbf{w} \cup \mathbf{v}$ is

$$\nabla_j \theta = -h_{\mathcal{G}_j^*}(\theta) + \frac{\sum_{i=1}^{|\Gamma(s_j)|} exp(E_{\mathcal{G}_j^i}) h_{\mathcal{G}_j^i}(\theta)}{\sum_{i=1}^{|\Gamma(s_j)|} exp(E_{\mathcal{G}_j^i})}$$

where if $\theta = w_s \in \mathbf{w}$, $h_{\mathcal{G}}(\theta) = 1$, if $\theta = v_{s,q} \in \mathbf{v}$, $h_{\mathcal{G}}(\theta) = \frac{1}{2} \sum_{t \in \mathcal{G}, t \neq s} v_{t,q}$.

Accelerating Gradient Computation

- Efficient Gradient Computation
  - Maintain a value of $E_{\mathcal{G}}$ for each $\mathcal{G}$, update it according to $E'_{\mathcal{G}} = \theta' h_{\mathcal{G}}(\theta) + g_{\mathcal{G}}(\theta) = E_{\mathcal{G}} + (\theta' - \theta) h_{\mathcal{G}}(\theta)$.
  - Calculate the h term by incrementally updating $R'_{\mathcal{G},q} = R_{\mathcal{G},q} + \frac{1}{2}(v'_{s,q} - v_{s,q})$, and applying $h_{\mathcal{G}}(v_{s,q}) = R_{\mathcal{G},q} - \frac{1}{2} v_{s,q}$
- Approximate Gradient Create a mini-batch of groups created by sampling M groups $\{\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(M)}\}$ from $\mathcal{D}_j$ with probability distribution $P_j = (P_j^1, \dots, P_j^{|\Gamma(s_j)|})$. Construct the unbiased estimated gradient:

$$\hat{\nabla}_j \theta = -h_{\mathcal{G}_j^*}(\theta) + \frac{1}{M} \sum_{i=1}^{M} h_{\mathcal{G}_j^{(i)}}(\theta)$$

Features used in Experiments

- Common non-pattern features, always used: **Pass, Capture, Extension, Self-atari, Atari, Line and Position (edge distance perpendicular to Line), Distance to previous move, Distance to second-last move, Fuego Playout Policy, Side Extension, Corner Opening Move, CFG Distance**. All of these features are implemented in Fuego [1].
- Two feature sets: **Small pattern feature set** adds only 3×3 patterns. **Large pattern feature set** includes large patterns [2] harvested from training set. Distribution of patterns with different size harvested at least 10 times in different game phases is provided in **Figure 2**.

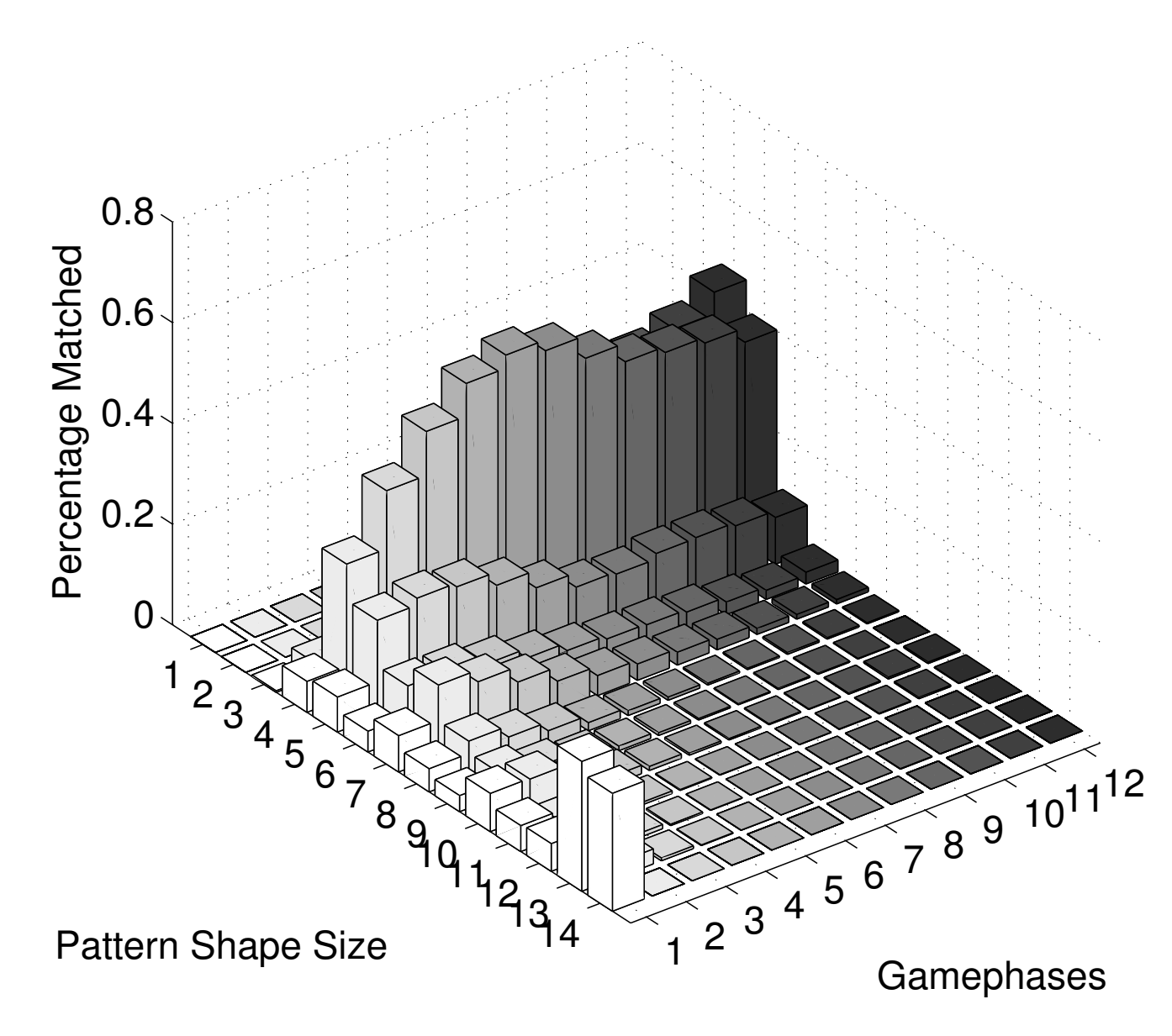

Figure 2: Pattern Size Distribution

Experiments

Three data sets of increasing size, $S_1 \subset S_2 \subset S_3$, are used, which contain 1000, 10000, and 20000 master games respectively. The games are in the public domain at `https://badukmovies.com/pro_games`. We test interaction dimensions $k = 5, 10$ for both FBT and LFR.

Small pattern feature set
Prediction Accuracy: FBT vs LFR

Training Set	FBT5	FBT10	LFR5	LFR10
S_1	32.56%	**32.82%**	30.01%	**30.08%**
S_2	33.18%	**33.42%**	30.95%	**31.63%**
S_3	33.46%	**34.01%**	31.13%	**31.94%**

Large pattern feature set
Prediction Accuracy: FBT vs LFR

Training Set	FBT5	FBT10	LFR5	LFR10
S_1	32.56%	**32.82%**	30.01%	**30.08%**
S_2	33.18%	**33.42%**	30.95%	**31.63%**
S_3	33.46%	**34.01%**	31.13%	**31.94%**

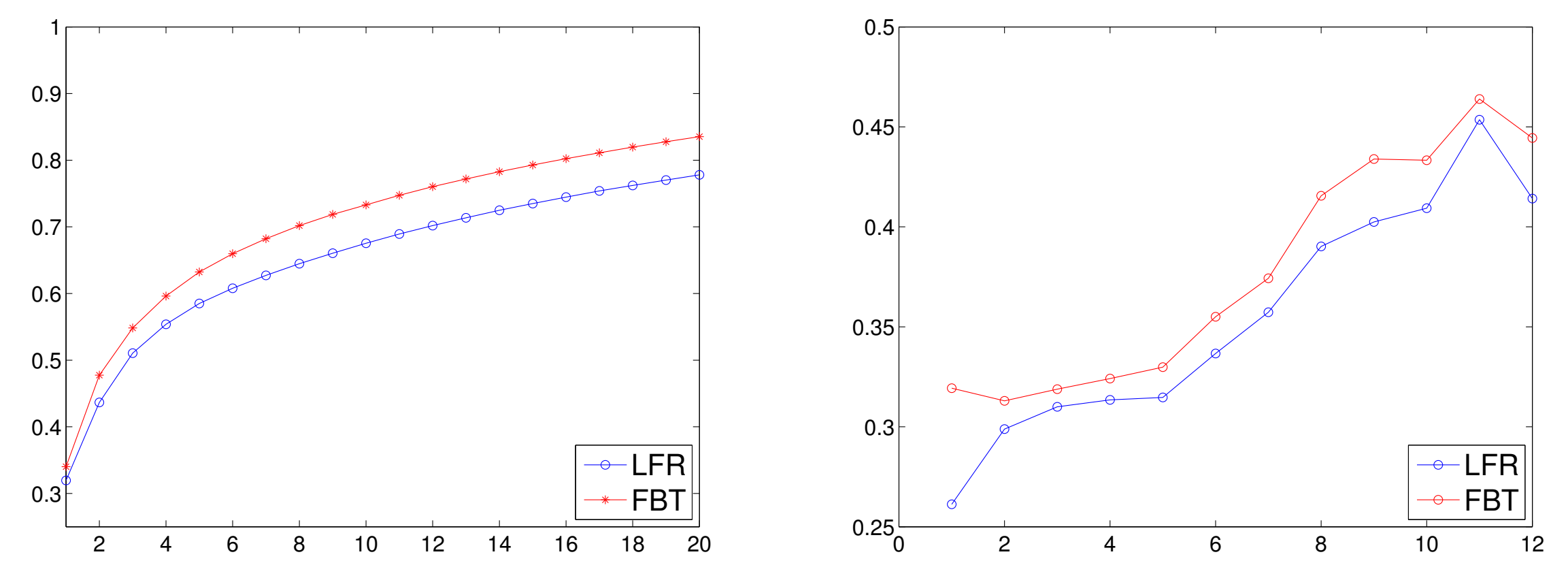

Figure 3: Left: Cumulative prediction rate. **Right:** Prediction rate per-phase. $k = 10$.

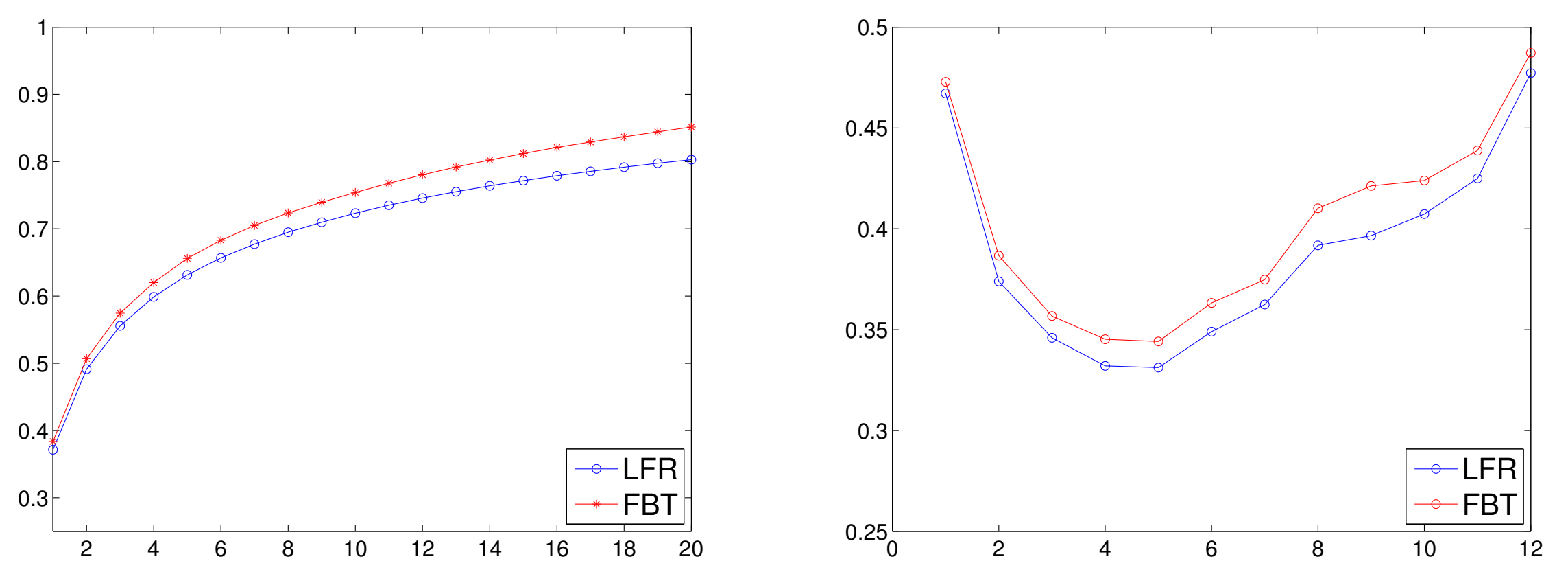

Figure 4: Left: Cumulative prediction rate. **Right:** Prediction rate per-phase. $k = 10$.

Move prediction by FBT with different sample sizes S=5,10,20,30

Training set	FBT_S5	FBT_S10	FBT_S20	FBT_S30	LFR	FBT_Full
S_1	30.04%	30.81%	31.06%	**31.78** %	30.01%	32.56%
S_3	32.07%	32.90%	32.98%	**33.03%**	30.95%	33.18%
S_3	32.54%	33.01%	33.09%	**33.12%**	31.13%	33.46%

References

[1] M. Enzenberger and M. Müller. Fuego, 2008-2015. `http://fuego.sourceforge.net`.
[2] David Stern, Ralf Herbrich, and Thore Graepel. Bayesian pattern ranking for move prediction in the game of Go. In *Proceedings of the 23rd international conference on Machine learning*, pages 873–880. ACM, 2006.