Better Computer Go Player with Neural Network and Long-term Prediction

Yuandong Tian[1] *Yan Zhu*[1,2] [1]Facebook AI Research [2]Rutgers University

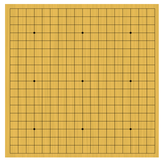
http://yuandong-tian.com

Arxiv Link: http://arxiv.org/abs/1511.06410

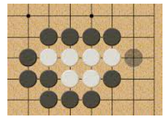
Introduction

Go, originated in ancient China more than 2,500 years ago, is a two-player zero-sum board game with full information. The possible board situations of Go are much more than the atoms of universe, rendering any brute-force search intractable.

Rules



Black and white take turns to place stones on a 19*19 board.



4-connecetd group dies if surrounded by enemy

The player with more territory wins

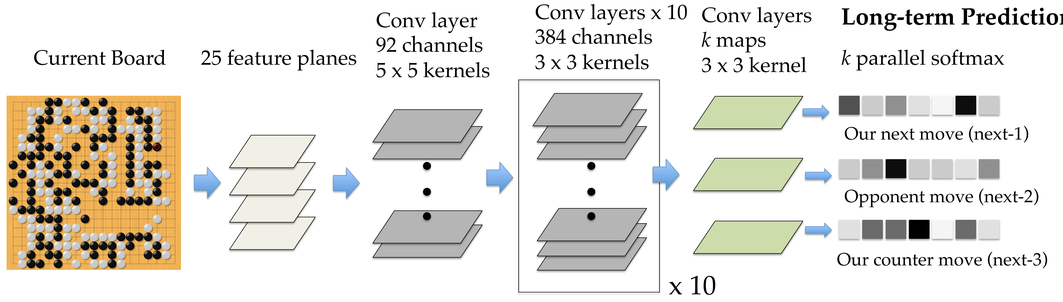
Method

Rather than brute-force search, human plays Go with both intuitions and reasoning: first think about a few possible alternatives, and then find the best move by careful analysis. With the advancement of Deep Learning, it is now possible to model human's intuition more precisely, yielding a better Computer Go player.

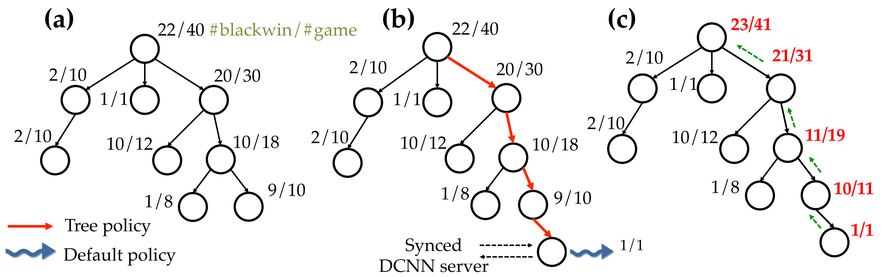
Policy Network

Feature Name
Our/enemy liberties (6)
Ko location (1)
Our/enemy/empty/(3)
Our/enemy history (2)
Enemy rank (9)

- Relative coding: our/enemy
- Easy-to-compute features



Monte Carlo Tree Search (MCTS)



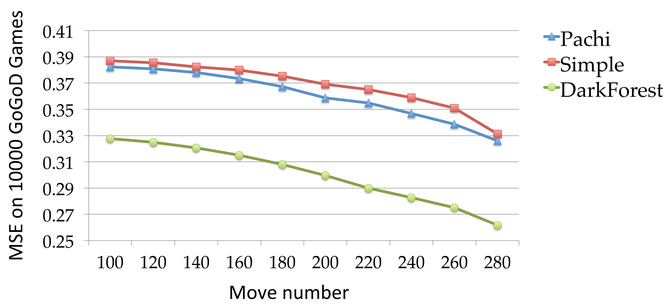
Original Design:
- Top 3-5 moves are picked from DCNN
- No prior once the moves are selected.
- Add noise to win rate estimate to diverge threads.

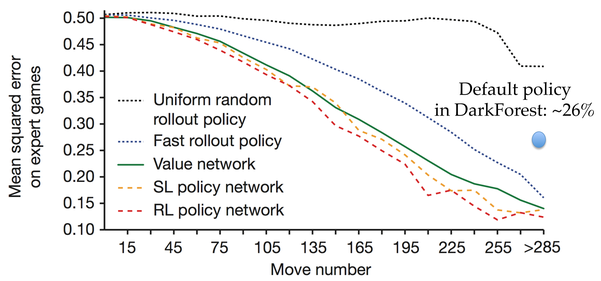
Current Design:
- Top 7 moves from DCNN.
- Use PUCT and virtual loss. Remove win rate noise.
- 5% of threads skip DCNN evaluation and run the default policy.

Default Policy

- Local 3x3 pattern matching with Zobrist hashing
- Incremental board status update with heap structure
- Handle special but critical cases with rules (nakade. etc)

	DarkForest	AlphaGo*
Dataset	Tygem	Tygem
Top-1 Accuracy	30.1%	24.2%
Speed	6-7 microsecond	2 microsecond
MSE after move 280	26%	~17%





From Fig. 2(b) in AlphaGo [Silver et al, Nature 2016]

AlphaGo*: all performances are based on [Silver et al., Nature 2016]. The performance of current AlphaGo might be higher.

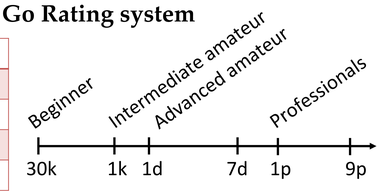
Experiments

KGS: Online amateur games
GoGoD: Professional Games

Dataset: 170k KGS / 80k GoGoD

Name	Description	KGS rank
Darkforest	Trained on KGS	1d
Darkfores1	Trained on GoGoD with nstep=3	2d
Darkfores2	Trained on GoGoD with nstep=3 and fine tuning	3d
Darkfores3	Trained on KGS with nstep=3 and fine tuning	3d

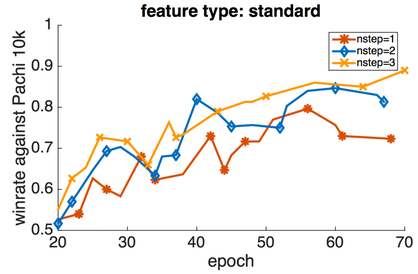
Go Rating system



Test Top-1 performance

	KGS	GoGoD
Darkforest	53.9%	48.6%
Darkfores1	54.2%	51.8%
Darkfores2	55.2%	53.3%
Darkfores3	57.6%	52.0%

feature type: standard



Training with 3 steps shows ***better*** performance than training with 1-step

Win rate for Pure DCNN

	GnuGo	Pachi 10k	Pachi 100k	Fuego 10k	Fuego 100k
Clark & Storkey (ICML 2015)	91.0	-	-	14.0	
Maddison et al. (ICLR 2015)	97.2	47.4	11.0	23.3	12.5
Darkforest	98.0 ± 1.0	71.5 ± 2.1	27.3 ± 3.0	84.5 ± 1.5	56.7 ± 2.5
Darkfores1	99.7 ± 0.3	88.7 ± 2.1	59.0 ± 3.3	93.2 ± 1.5	78.0 ± 1.7
Darkfores2	100 ± 0.0	94.3 ± 1.7	72.6 ± 1.9	98.5 ± 0.1	89.7 ± 2.1
AlphaGo* (RL)	-	-	85	-	-

Win rate for DCNN + MCTS

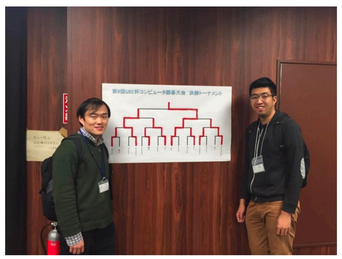
Vs Pure DCNN	DF + MCTS	DF1 + MCTS	DF2 + MCTS
1000 rollout (top5)	89.6	76.4	68.4
1000 rollout (top3)	91.6	89.6	79.2*
5000 rollout (top5)	96.8	94.3	82.3

Vs Pachi 10k	DF + MCTS	DF1 + MCTS	DF2 + MCTS
Pure DCNN	71.5	88.7	94.3
1000 rollout (top5)	88.4	94.4	97.6
1000 rollout (top3)	95.2	98.4	99.2
5000 rollout (top5)	98.4	99.6	100

*79.2: With PUCT and virtual loss, this win rate becomes 94.2%. Other win rates also increase.

Competitions

Stable KGS **5d** (kgs id: *darkfmcts3*)
3rd in KGS January Go Tournament
2nd in 9th UEC Cup for Computer Go



2nd place in 9th UEC Cup

4th Denseisen vs. Kobayashi (9p)

Our Go engine will be open-sourced!

See: https://github.com/facebookresearch/darkforest

- Standalone project with little dependency.
- Efficient Go/MCTS libraries written in C/Lua.
- Runnable on a single machine with 1-4 GPUs.
- Much stronger than existing open source engines.