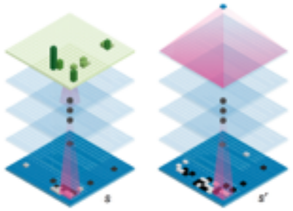
AlphaGo

COMP7607

Lingpeng Kong
Department of Computer Science, The University of Hong Kong

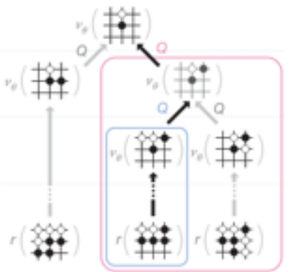
Outline

- Representation of Board (deep convolutional neural networks, CNN)
- Imitating expert moves (supervised learning)
- Predicting the wining possibility given a board configuration (reinforcement learning)
- Select a move, more wisely (Monte Carlo tree search)



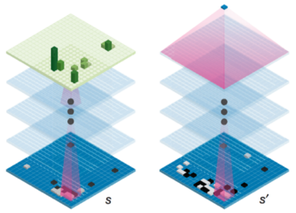
Policy network (SL)

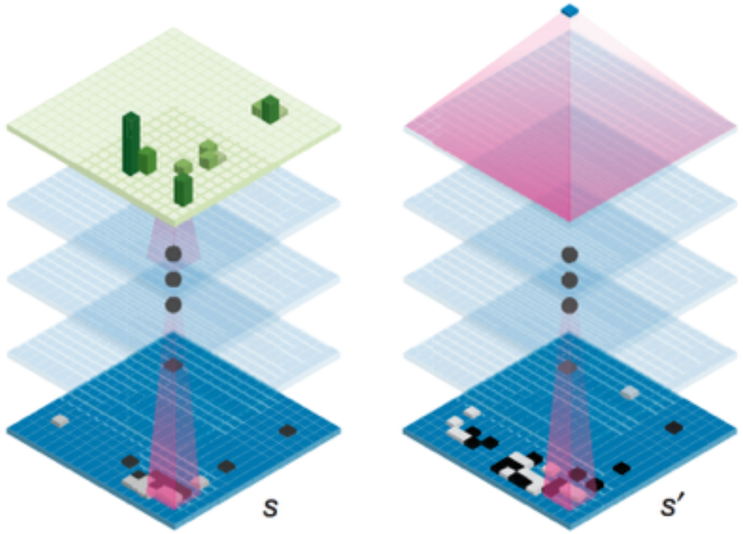
Value network



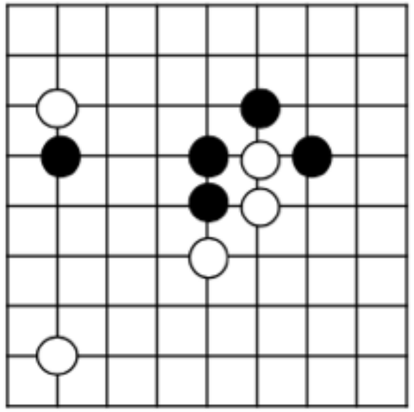
Deterministic MDP

- Representation of Board (deep convolutional neural networks, CNN)





Current Board

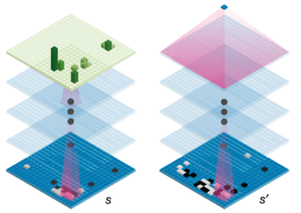


Current Board

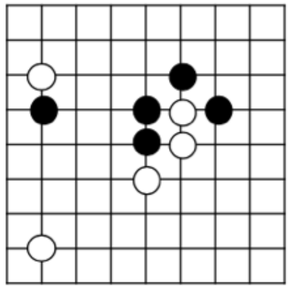
00 000 0000
00 000 1000
0-100 1-110 0
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

s

- Representation of Board (deep convolutional neural networks, CNN)



Current Board



Current Board

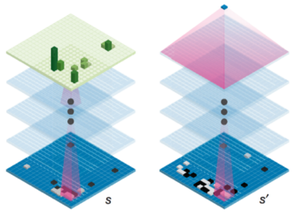
```
00 000 0000
00 000 1000
0-100 1-110 0
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000
```

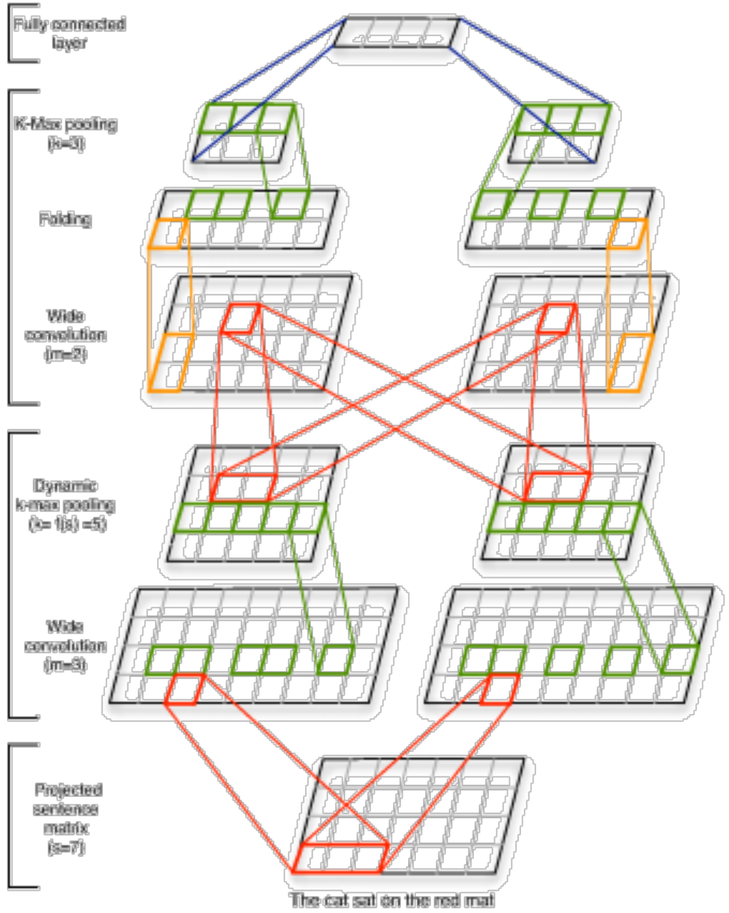
$$\theta^T \phi(s)$$

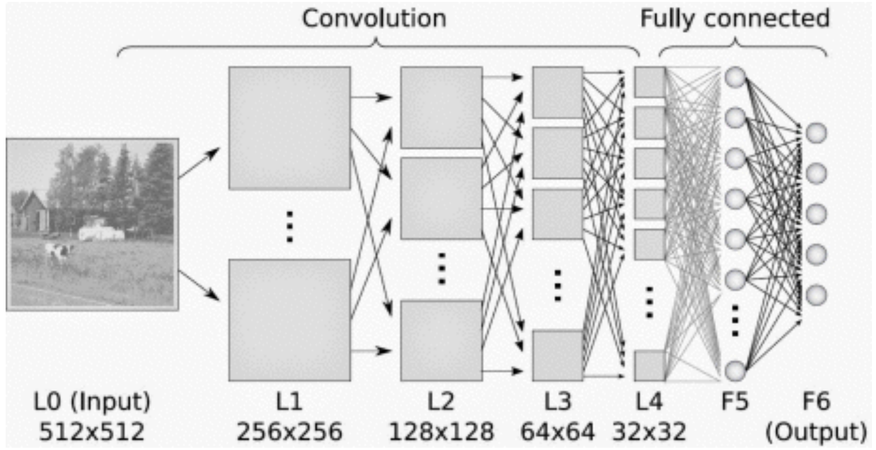
Feature	# of planes	Description
Stone colour	3	Player stone / opponent stone / empty
Ones	1	A constant plane filled with 1
Turns since	8	How many turns since a move was played
Liberties	8	Number of liberties (empty adjacent points)
Capture size	8	How many opponent stones would be captured
Self-atari size	8	How many of own stones would be captured
Liberties after move	8	Number of liberties after this move is played
Ladder capture	1	Whether a move at this point is a successful ladder capture
Ladder escape	1	Whether a move at this point is a successful ladder escape
Sensibleness	1	Whether a move is legal and does not fill its own eyes
Zeros	1	A constant plane filled with 0
Player color	1	Whether current player is black

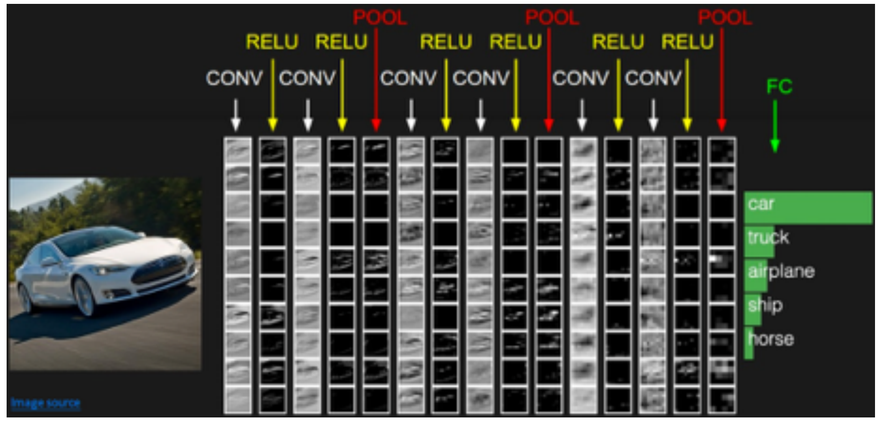
Feature planes used by the policy network (all but last feature) and value network (all features).

- Representation of Board (deep convolutional neural networks, CNN)

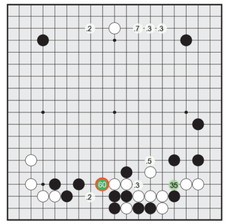


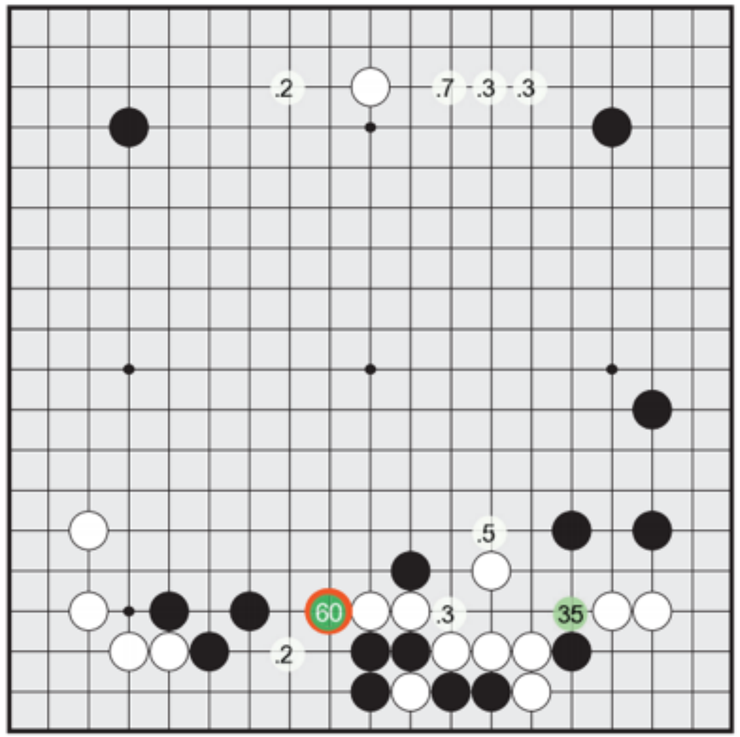


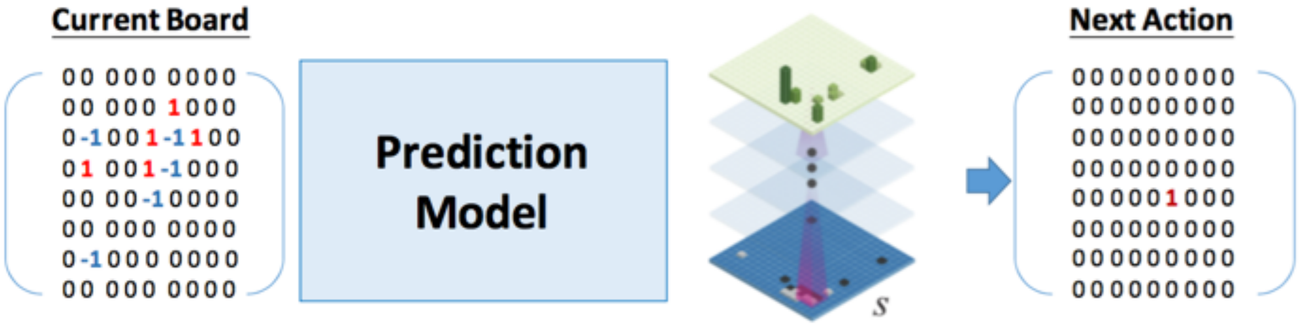




- Imitating expert moves (supervised learning)

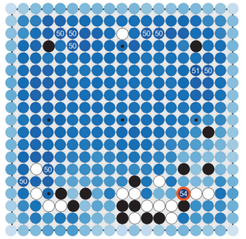


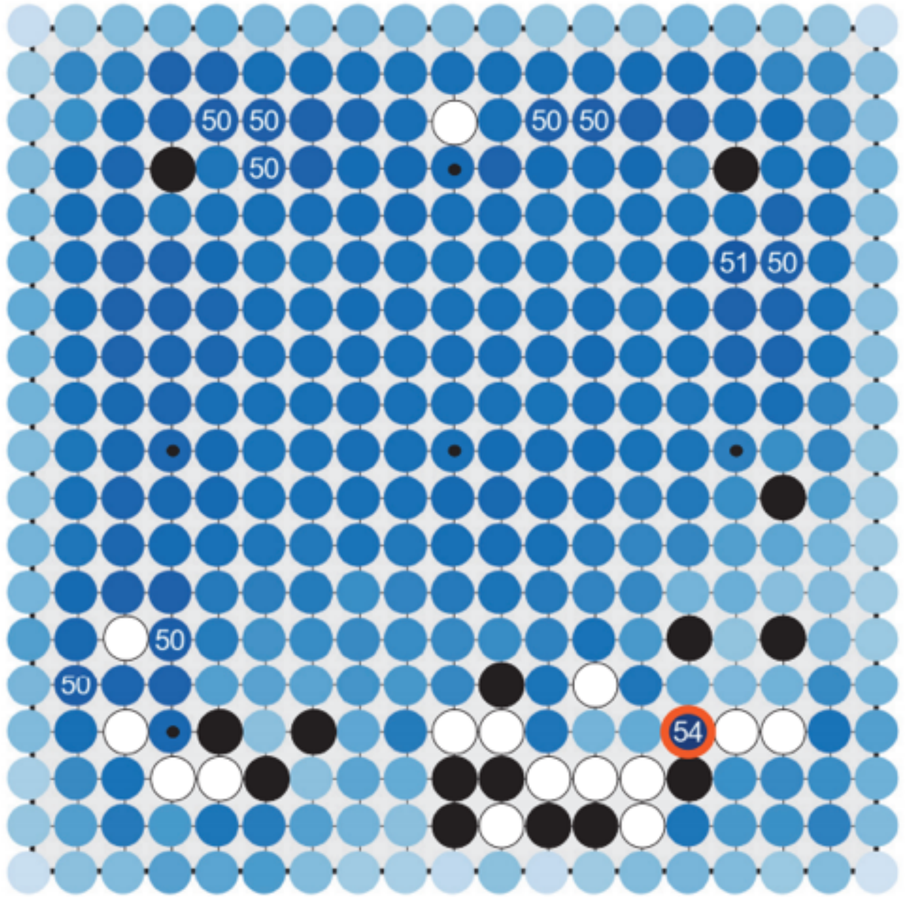


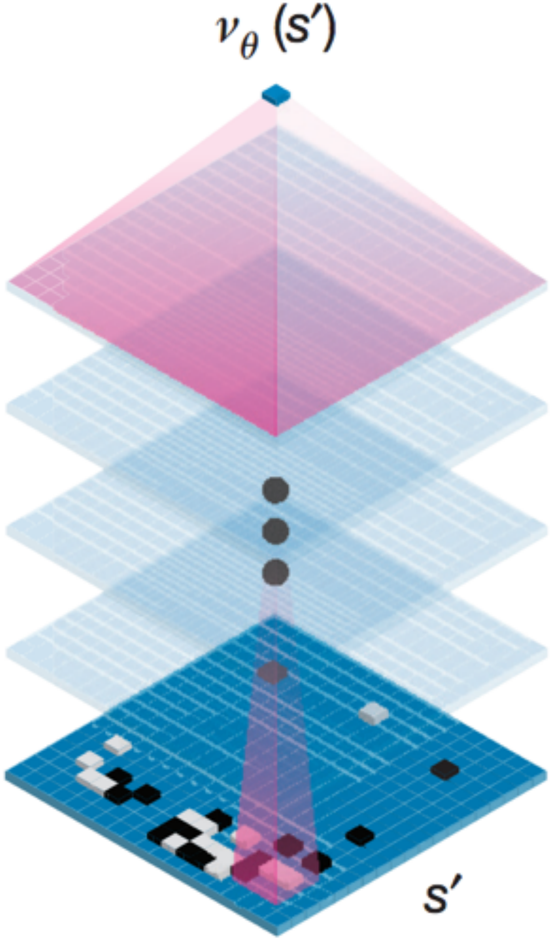


Initializing policy

- Predicting the wining possibility given a board configuration (reinforcement learning)

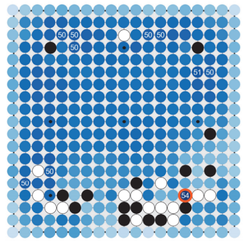


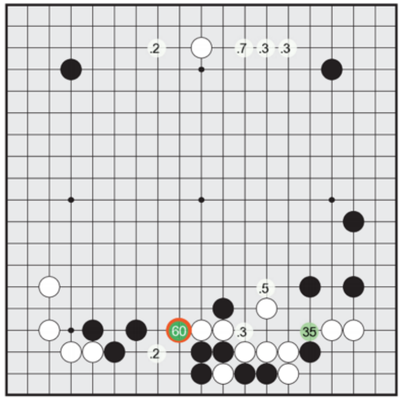




fitted value iteration algorithm

- Predicting the wining possibility given a board configuration (reinforcement learning)

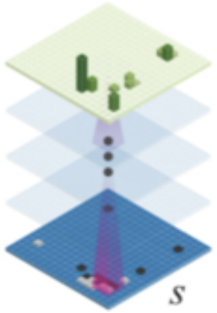




Current Board

00 000 0000
00 000 1000
0-100 1-1100
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

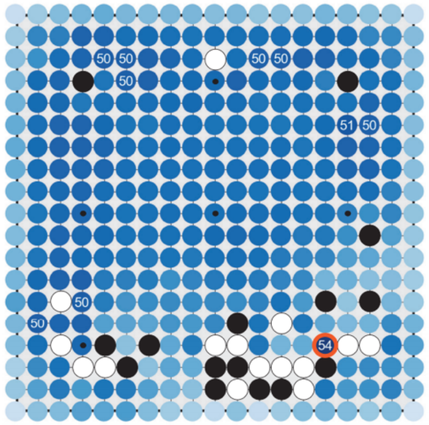
Prediction Model



Next Action

000000000
000000000
000000000
000000000
000001000
000000000
000000000
000000000

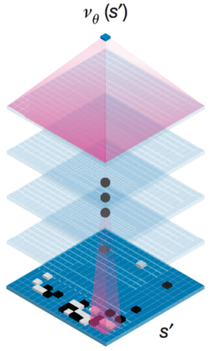
s ***g: s → p(a|s)*** ***p(a|s)*** **argmax** ***a***



Current Board

00 000 0000
00 000 1000
0-100 1-1100
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

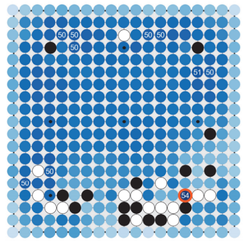
Prediction Model

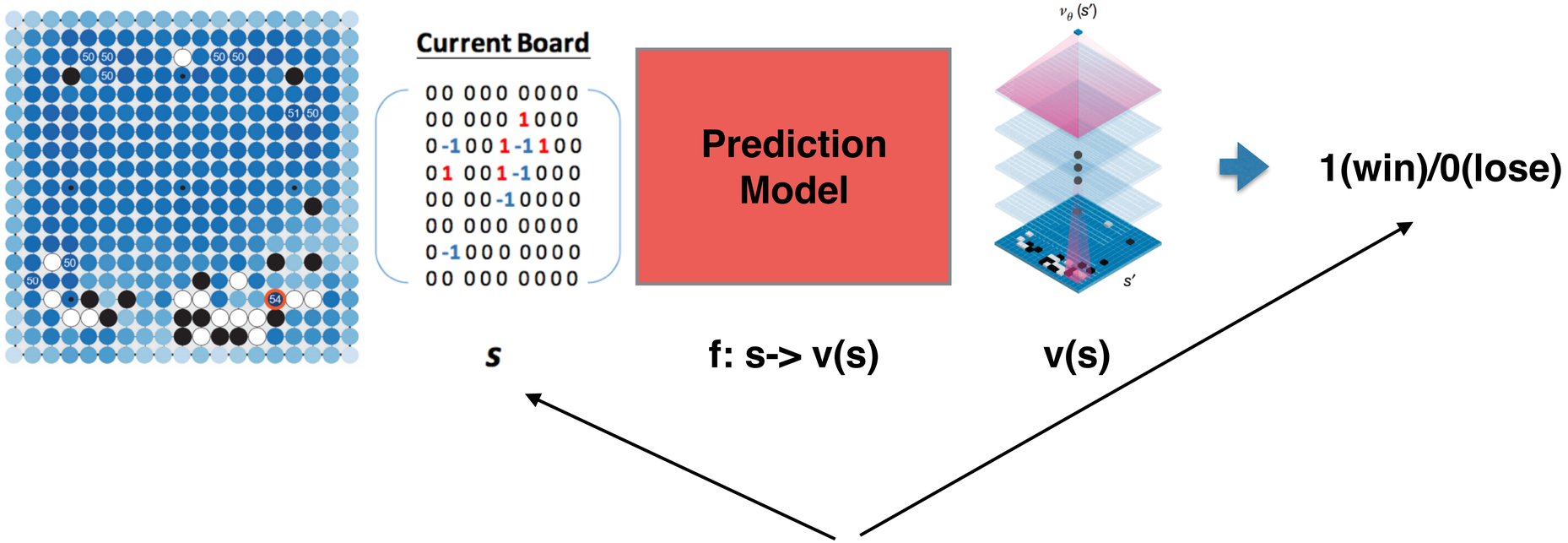


1(win)/0(lose)

s **f: s-> v(s)** **v(s)**

- Predicting the wining possibility given a board configuration (reinforcement learning)



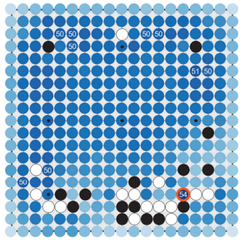


(S, Z) — training pair (x,y)

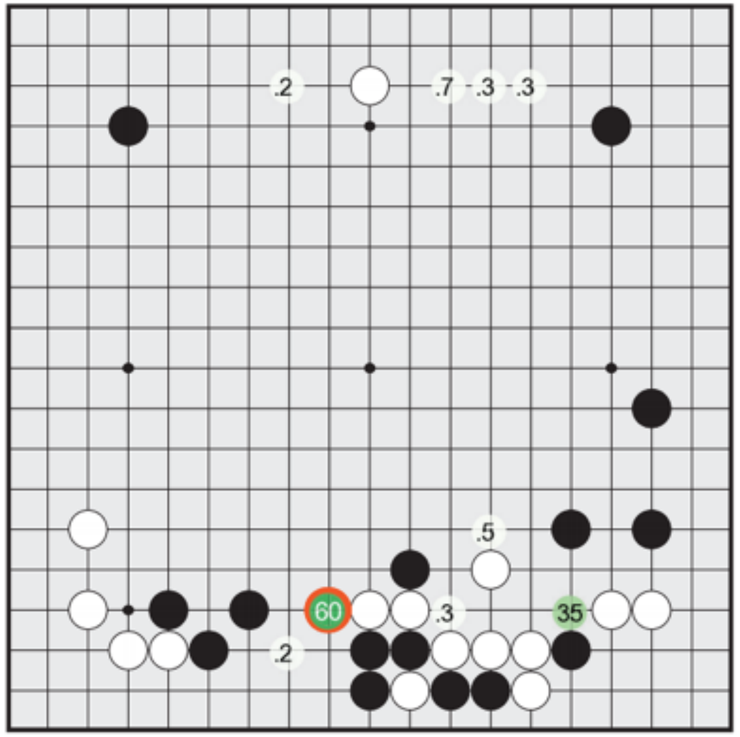
1 *"29,400,000 positions from 160,000 games played by KGS 6 to 9 dan human players"*

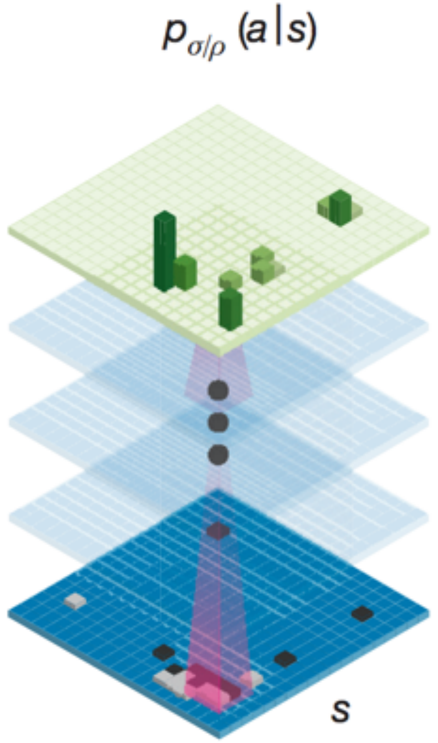
Overfitting, Training error rate 0.19, Test error rate 0.37

- Predicting the wining possibility given a board configuration (reinforcement learning)



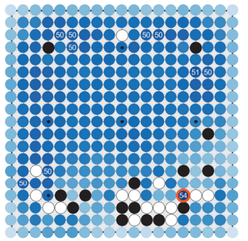
(S, Z) — training pair (x,y) — self play?



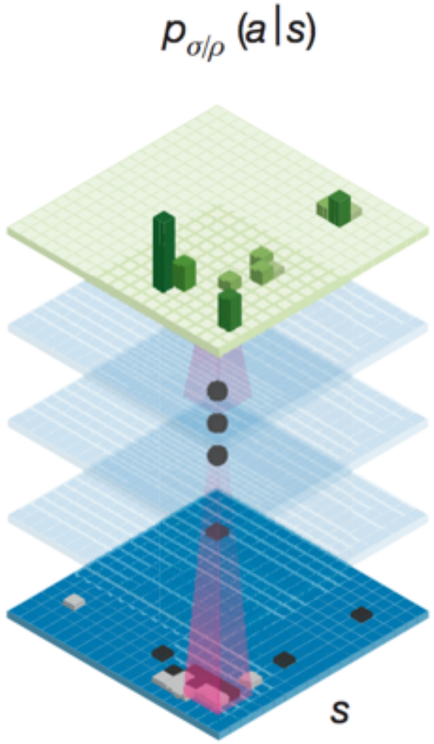


Supervised Learning policy (SL policy)

- Predicting the wining possibility given a board configuration (reinforcement learning)



(S, Z) — training pair (x,y) — Better policy?

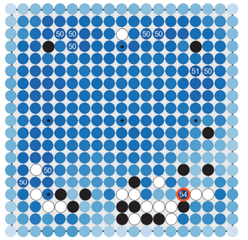


better policy → better estimation of the value

Reinforcement Learning policy (SL policy)

policy gradient

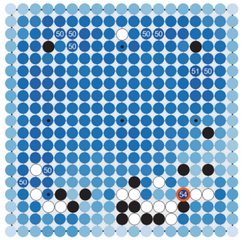
- Predicting the wining possibility given a board configuration (reinforcement learning)



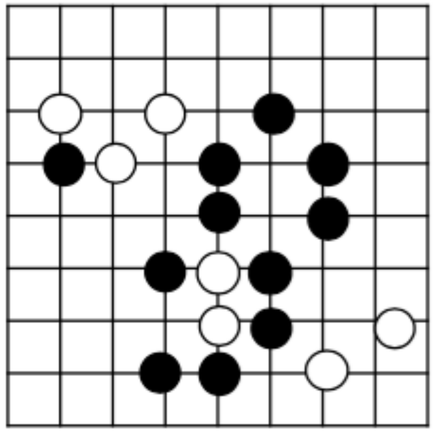
ver. 0 = Supervised Learning policy (SL policy)

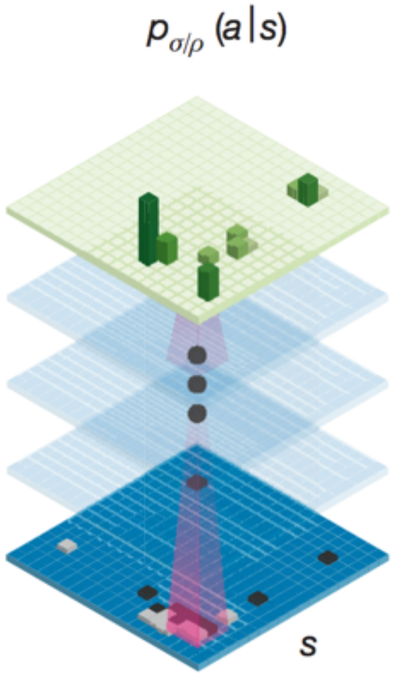
Policy Network ver.0	VS	Policy Network ver.0	→	Win/loss
.....				
Policy Network ver. 200	VS	Policy Network ver.153	→	Win/loss
Policy Network ver. 201	VS	Policy Network ver.78	→	Win/loss
.....				
Policy Network ver. 20000	VS	Policy Network ver. 12531	→	Win/loss

- Predicting the wining possibility given a board configuration (reinforcement learning)



Board position





win z = 1

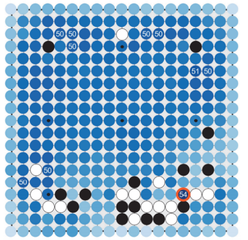
lose z = -1

Update

$$\Delta\rho \propto \frac{\partial \log p_\rho(a_t|s_t)}{\partial\rho} z_t$$

policy gradient

- Predicting the wining possibility given a board configuration (reinforcement learning)

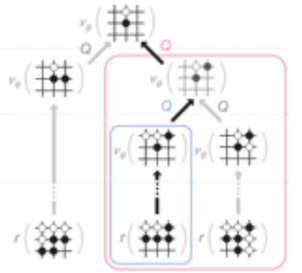


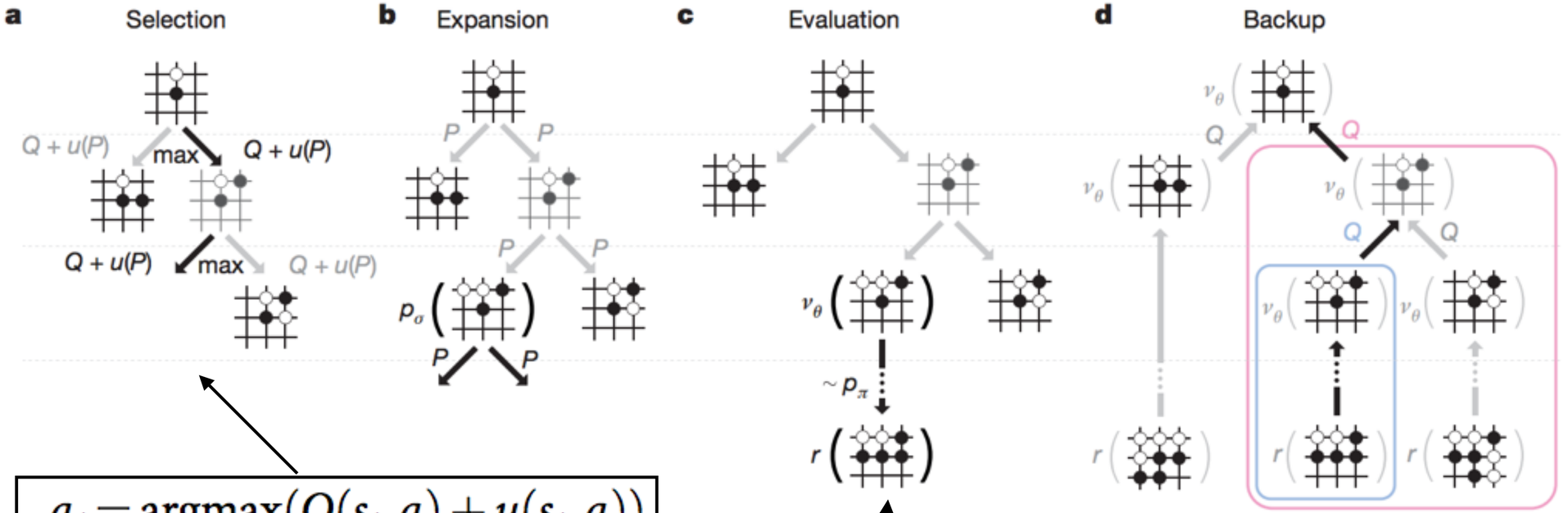
Policy Network ver. 20000

Reinforcement Learning policy (RL policy)

2 *“a new self-play data set consisting of 30,000,000 positions, each sampled from a separate game”*

- Select a move, more wisely (Monte Carlo tree search)





$$a_t = \underset{a}{\operatorname{argmax}}(Q(s_t,a) + u(s_t,a))$$

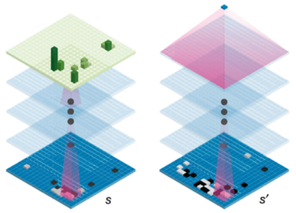
$$u(s,a) \propto \frac{P(s,a)}{1+N(s,a)}$$

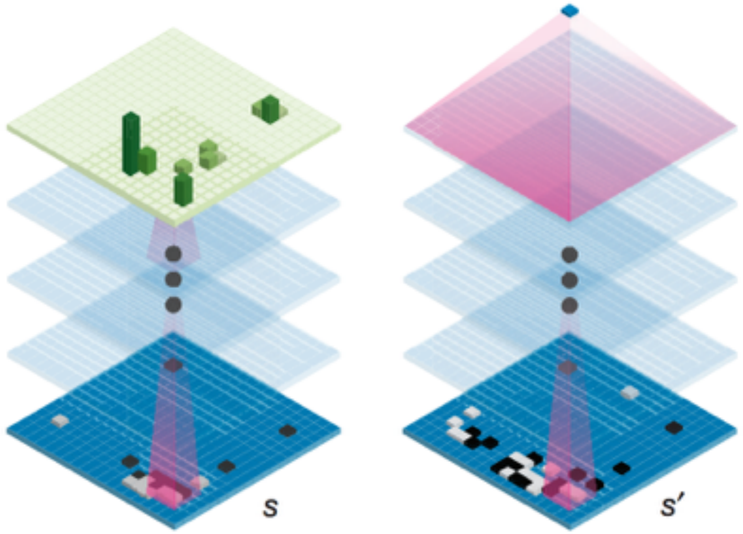
$$V(s_L) = (1-\lambda)v_\theta(s_L) + \lambda z_L$$

$$N(s,a) = \sum_{i=1}^{n} 1(s,a,i)$$

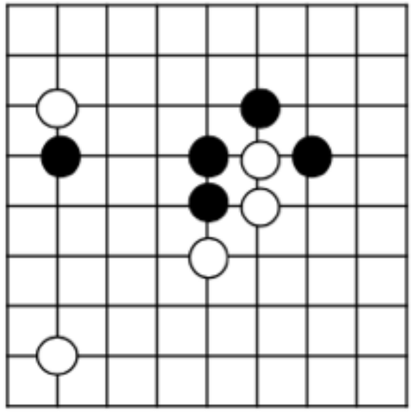
$$Q(s,a) = \frac{1}{N(s,a)} \sum_{i=1}^{n} 1(s,a,i) V(s_L^i)$$

- Representation of Board (deep convolutional neural networks, CNN)





Current Board

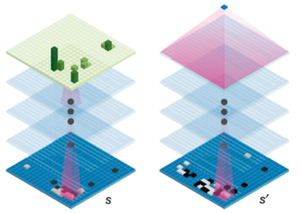


Current Board

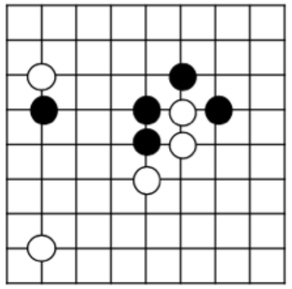
00 000 0000
00 000 1000
0-100 1-110 0
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

s

- Representation of Board (deep convolutional neural networks, CNN)



Current Board



Current Board

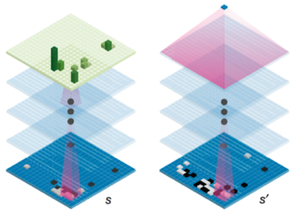
```
00 000 0000
00 000 1000
0-100 1-110 0
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000
```

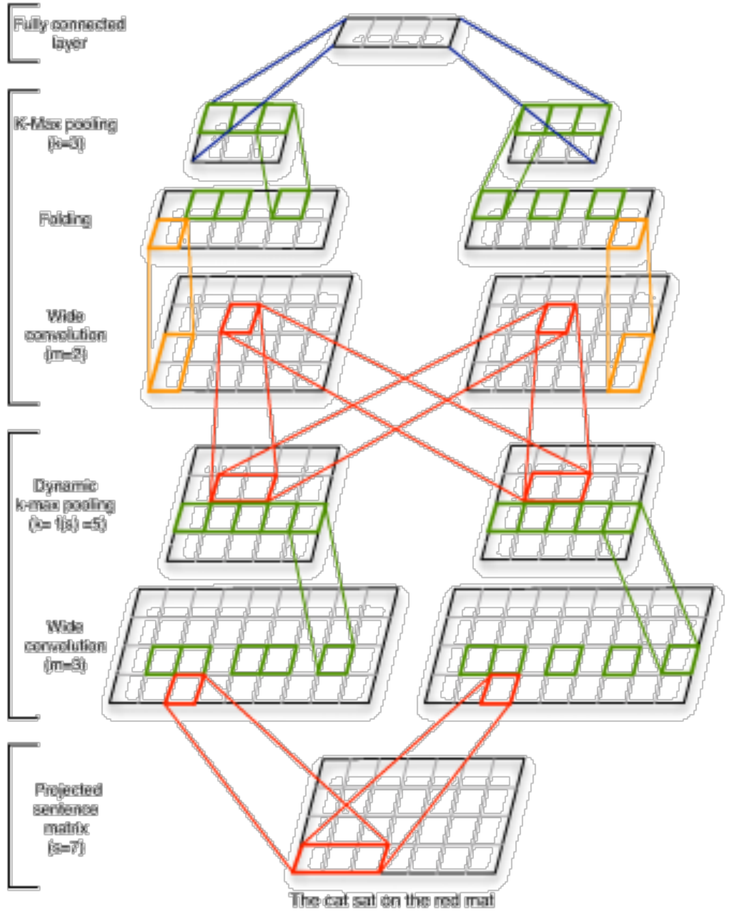
$$\theta^T \phi(s)$$

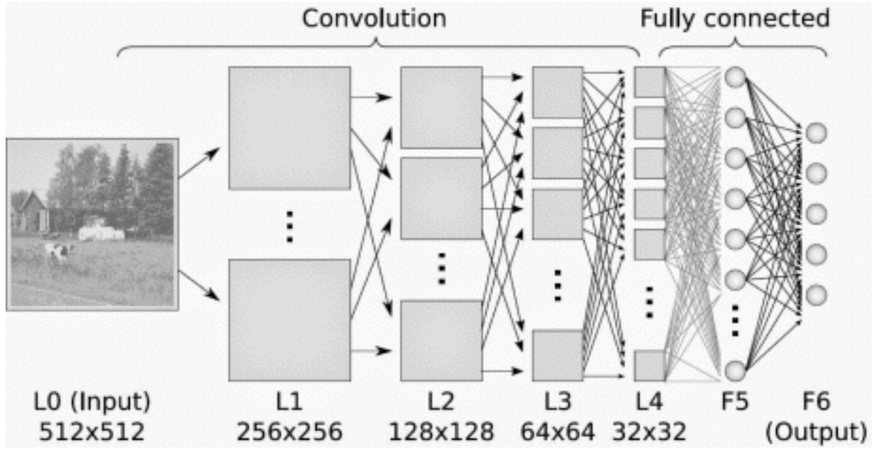
Feature	# of planes	Description
Stone colour	3	Player stone / opponent stone / empty
Ones	1	A constant plane filled with 1
Turns since	8	How many turns since a move was played
Liberties	8	Number of liberties (empty adjacent points)
Capture size	8	How many opponent stones would be captured
Self-atari size	8	How many of own stones would be captured
Liberties after move	8	Number of liberties after this move is played
Ladder capture	1	Whether a move at this point is a successful ladder capture
Ladder escape	1	Whether a move at this point is a successful ladder escape
Sensibleness	1	Whether a move is legal and does not fill its own eyes
Zeros	1	A constant plane filled with 0
Player color	1	Whether current player is black

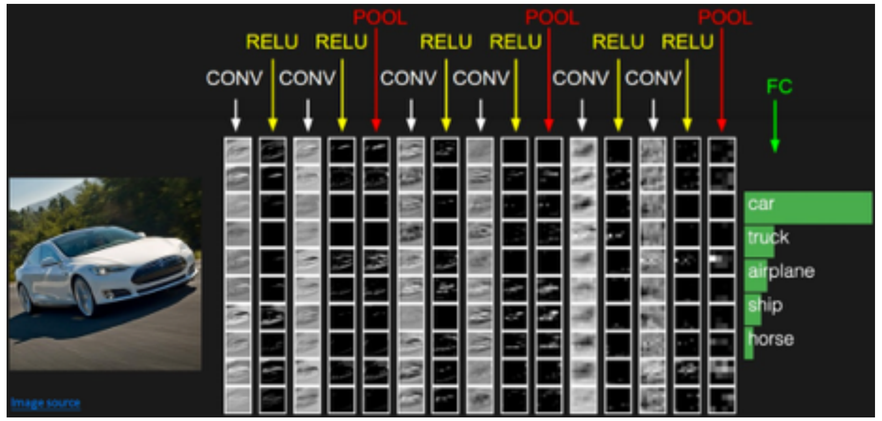
Feature planes used by the policy network (all but last feature) and value network (all features).

- Representation of Board (deep convolutional neural networks, CNN)

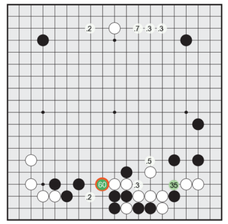


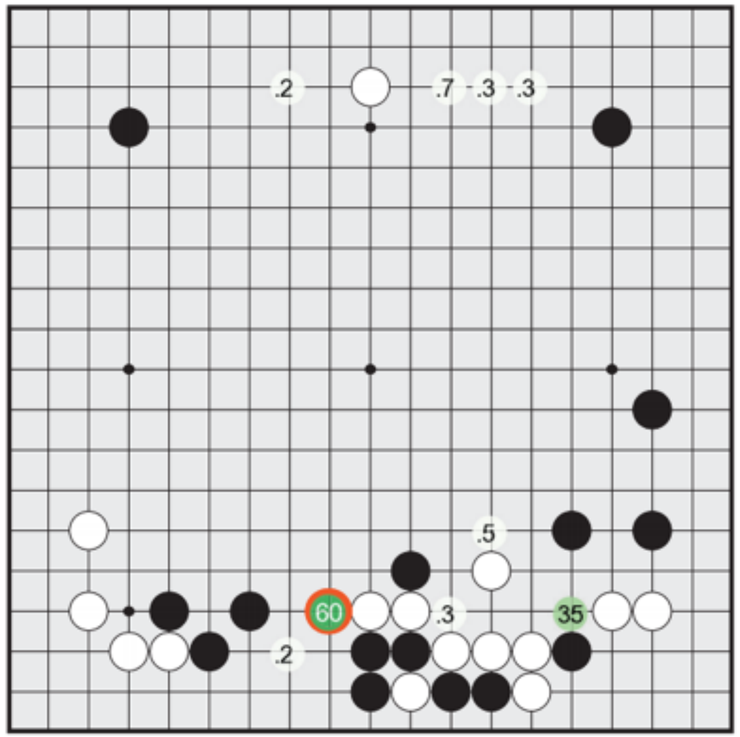


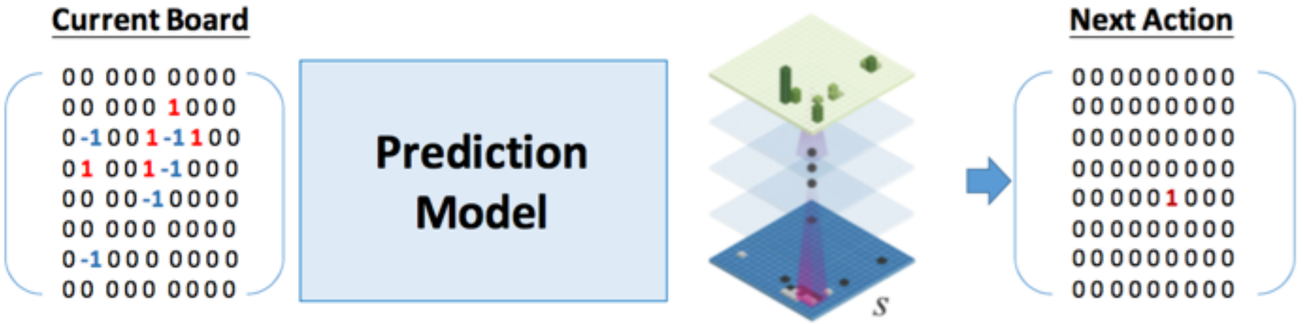




- Imitating expert moves (supervised learning)

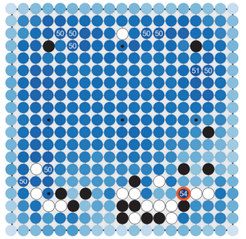


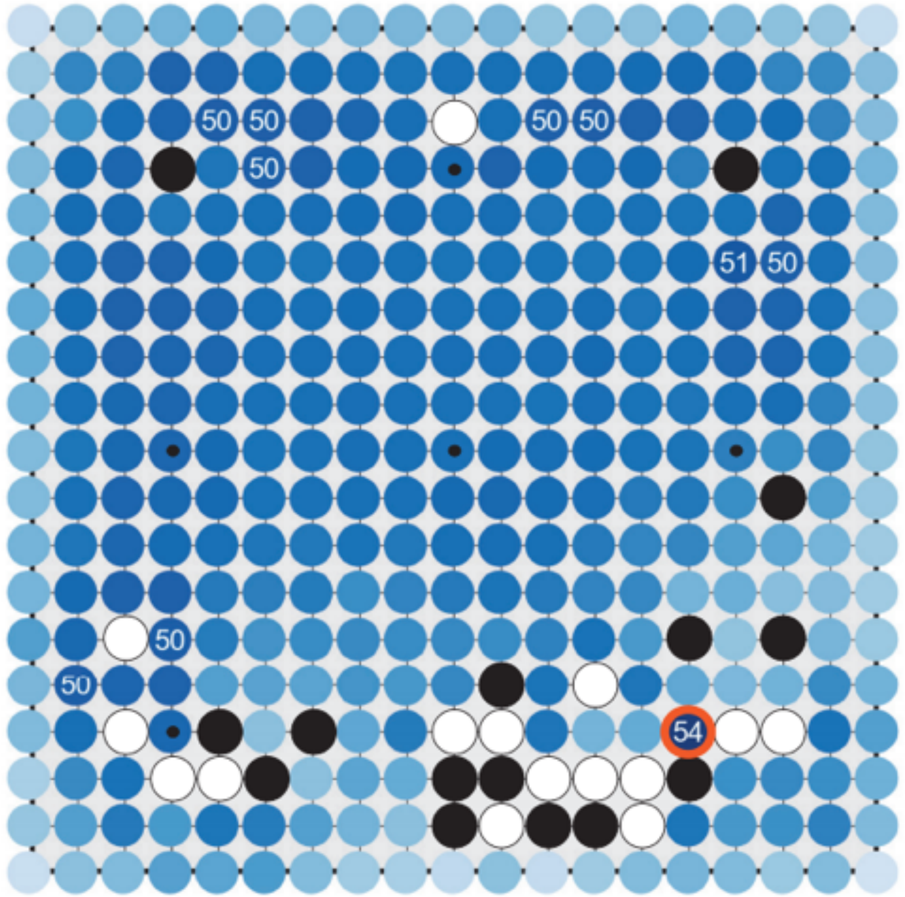


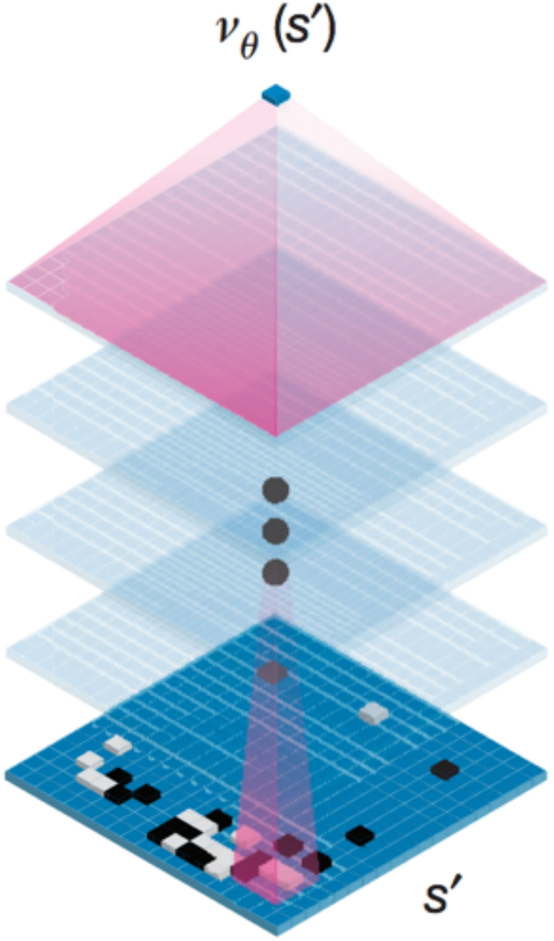


Initializing policy

- Predicting the wining possibility given a board configuration (reinforcement learning)

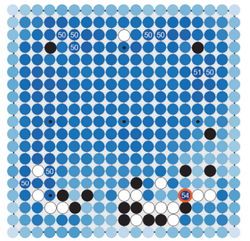


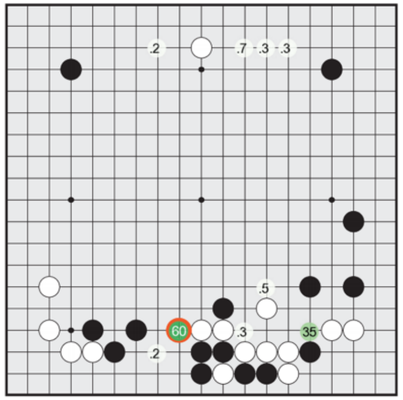




fitted value iteration algorithm

- Predicting the wining possibility given a board configuration (reinforcement learning)

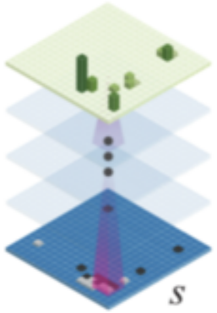




Current Board

00 000 0000
00 000 1000
0-100 1-1100
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

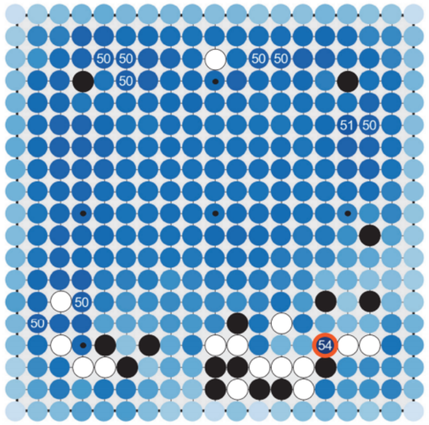
Prediction Model



Next Action

000000000
000000000
000000000
000000000
000001000
000000000
000000000
000000000

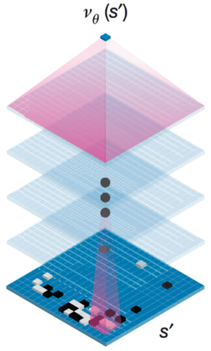
s ***g: s → p(a|s)*** ***p(a|s)*** **argmax** ***a***



Current Board

00 000 0000
00 000 1000
0-100 1-1100
01 001-1000
00 00-10000
00 000 0000
0-1000 0000
00 000 0000

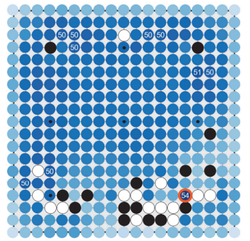
Prediction Model

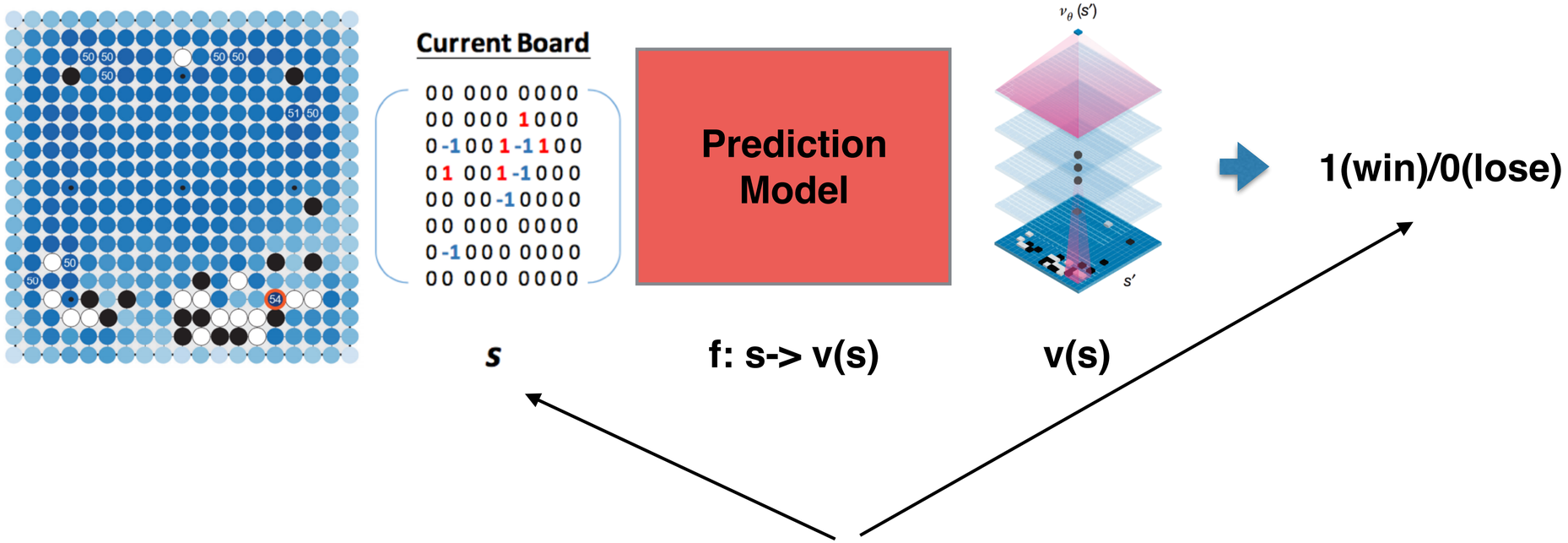


1(win)/0(lose)

s **f: s-> v(s)** **v(s)**

- Predicting the wining possibility given a board configuration (reinforcement learning)



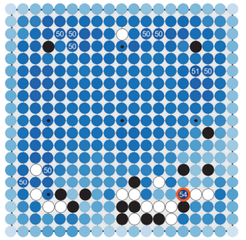


(S, Z) — training pair (x,y)

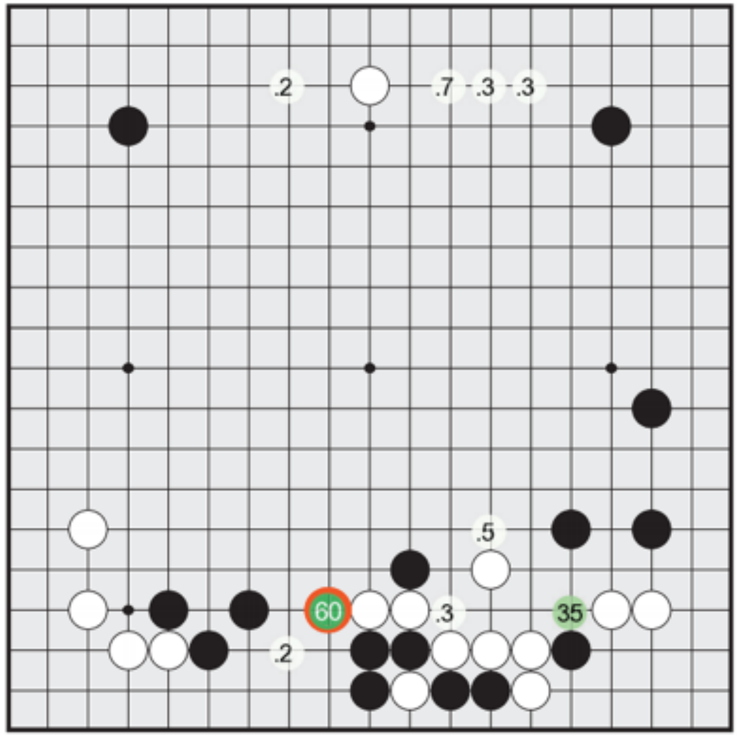
1 *"29,400,000 positions from 160,000 games played by KGS 6 to 9 dan human players"*

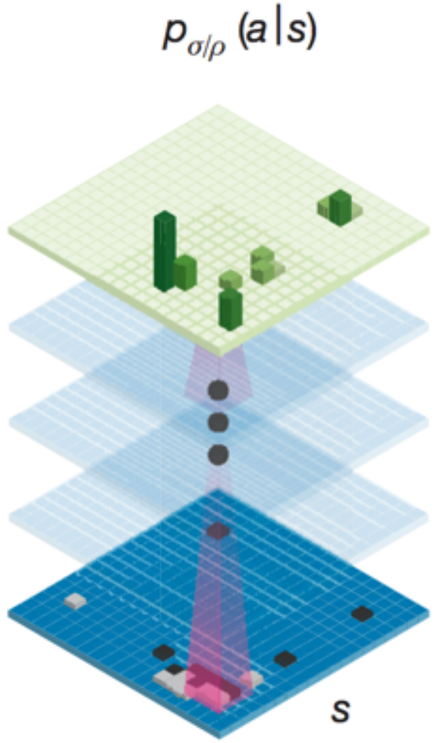
Overfitting, Training error rate 0.19, Test error rate 0.37

- Predicting the wining possibility given a board configuration (reinforcement learning)



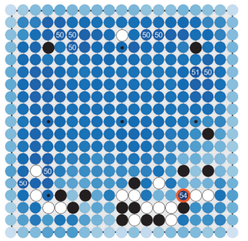
(S, Z) — training pair (x,y) — self play?



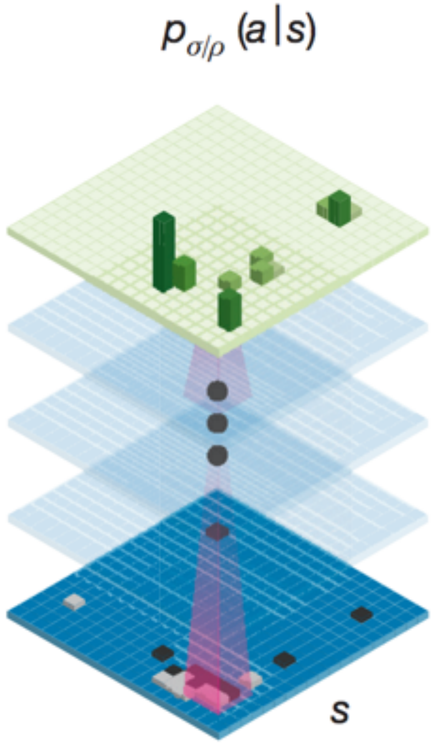


Supervised Learning policy (SL policy)

- Predicting the wining possibility given a board configuration (reinforcement learning)



(S, Z) — training pair (x,y) — Better policy?

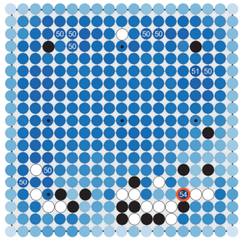


better policy → better estimation of the value

Reinforcement Learning policy (SL policy)

policy gradient

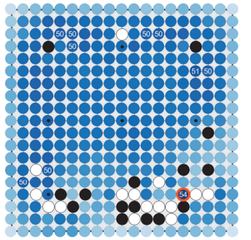
- Predicting the wining possibility given a board configuration (reinforcement learning)



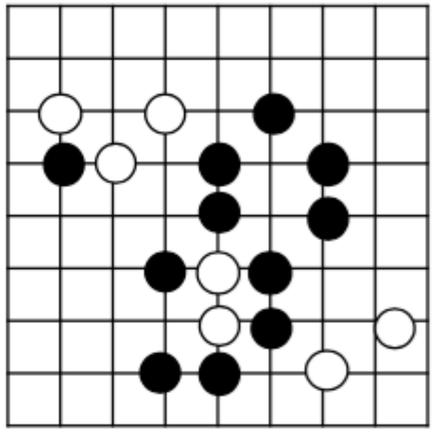
ver. 0 = Supervised Learning policy (SL policy)

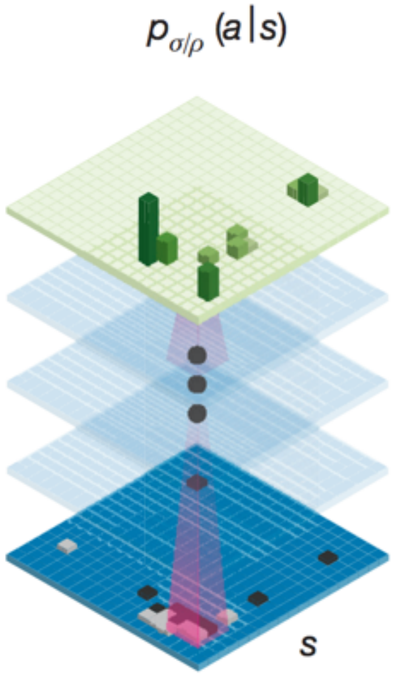
Policy Network ver.0	VS	Policy Network ver.0	→	Win/loss
.....				
Policy Network ver. 200	VS	Policy Network ver.153	→	Win/loss
Policy Network ver. 201	VS	Policy Network ver.78	→	Win/loss
.....				
Policy Network ver. 20000	VS	Policy Network ver. 12531	→	Win/loss

- Predicting the wining possibility given a board configuration (reinforcement learning)



Board position





win z = 1

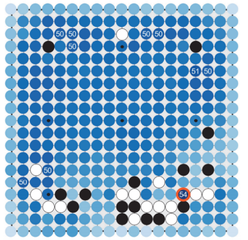
lose z = -1

Update

$$\Delta\rho \propto \frac{\partial \log p_\rho(a_t|s_t)}{\partial\rho} z_t$$

policy gradient

- Predicting the wining possibility given a board configuration (reinforcement learning)

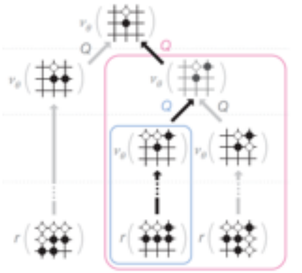


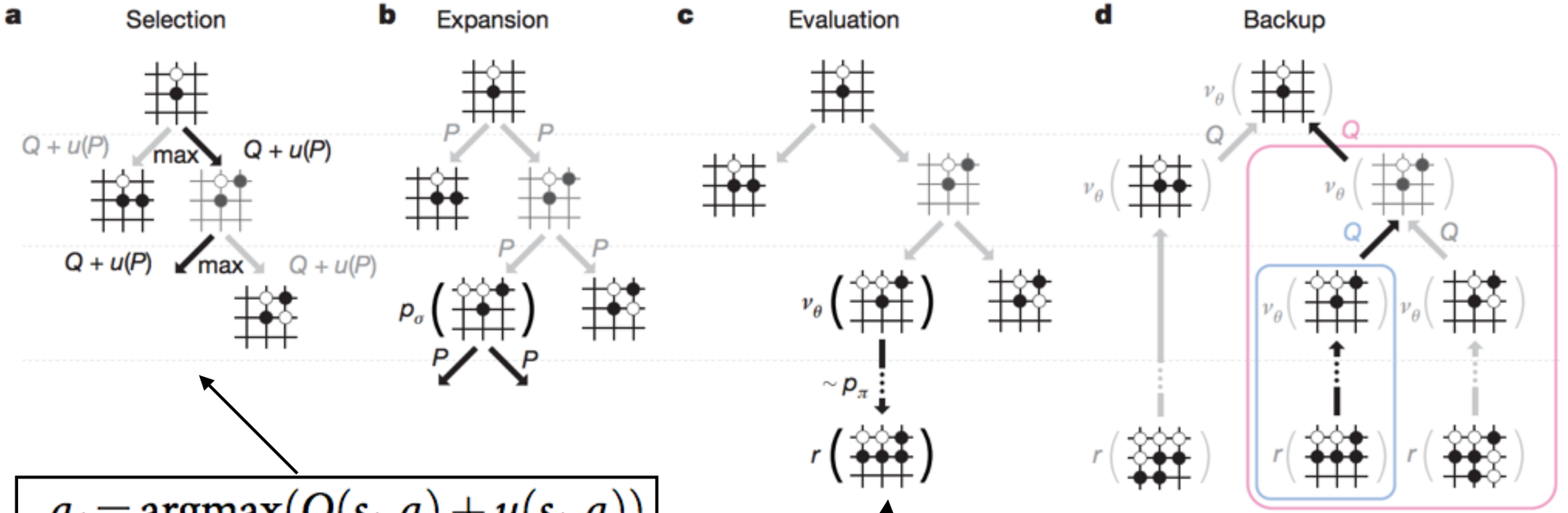
Policy Network ver. 20000

Reinforcement Learning policy (RL policy)

2 *“a new self-play data set consisting of 30,000,000 positions, each sampled from a separate game”*

- Select a move, more wisely (Monte Carlo tree search)





$$a_t = \underset{a}{\operatorname{argmax}}(Q(s_t,a) + u(s_t,a))$$

$$u(s,a) \propto \frac{P(s,a)}{1+N(s,a)}$$

$$V(s_L) = (1-\lambda)v_\theta(s_L) + \lambda z_L$$

$$N(s,a) = \sum_{i=1}^{n} 1(s,a,i)$$

$$Q(s,a) = \frac{1}{N(s,a)} \sum_{i=1}^{n} 1(s,a,i) V(s_L^i)$$