AlphaGo

На основе статьи Mastering the game of Go with deep neural networks and tree search

Альмухаметова Гузель

Краткий план

- Описание игры
- Описание алгоритма
  - SL policy
  - Rollout policy
  - RL policy
  - MCTS
- Итоги

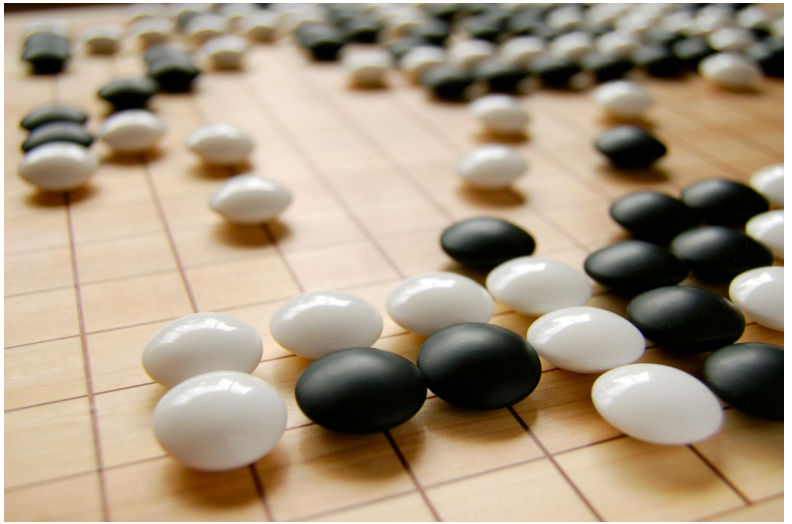
*Бонус

Игра Go

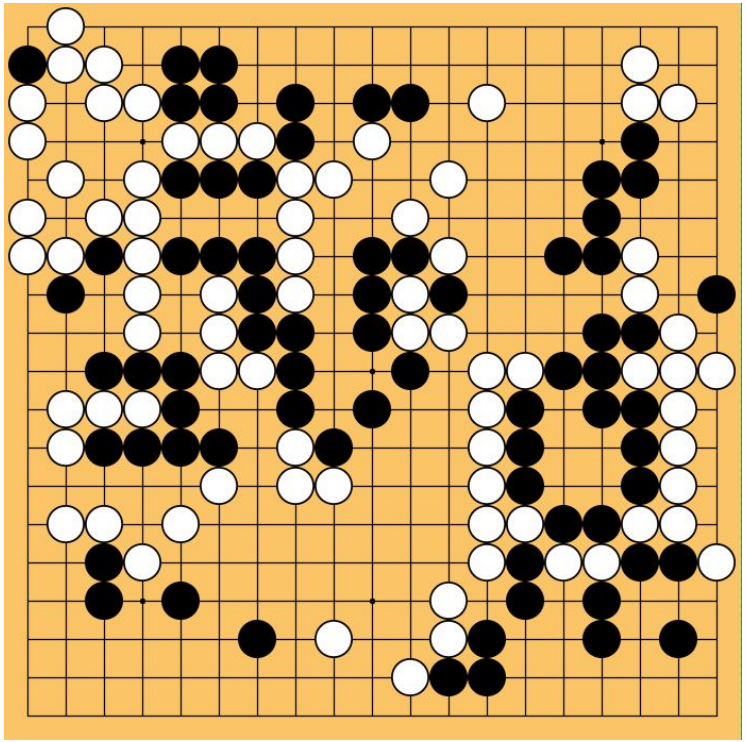
Цель игры - захват территории

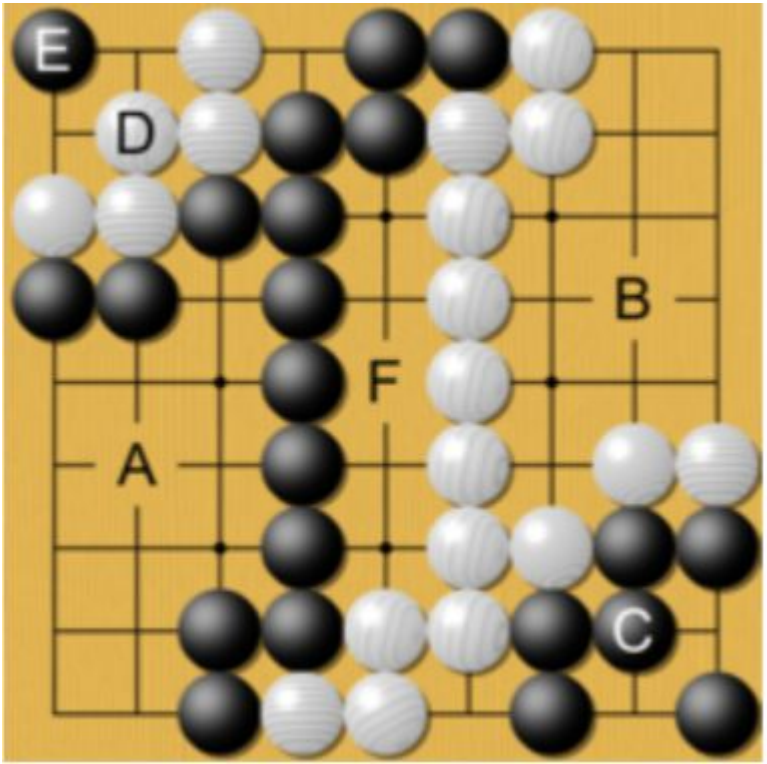
Основные правила:

- Первыми ходят чёрные
- Игрок выставляет один свой камень на доску в любую не занятую точку пересечения линий
- По завершении игры подсчитываются очки, набранные игроками. Игрок получает по одному очку за каждый из пунктов доски, окружённых камнями только его цвета, и по одному очку за каждый захваченный камень противника



Возможная расстановка фигур во время игры



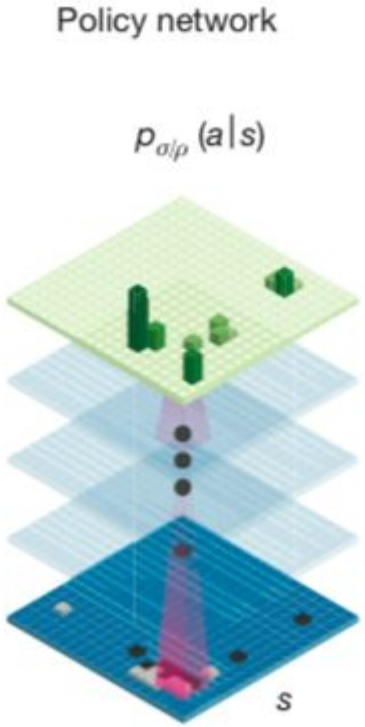


Policy networks

Учимся предсказывать ходы людей, поэтому обучение происходит на 160000 доступных в онлайне игр игроках довольно высокого уровня

SL policy network
Архитектура сети — 13 уровней сверток с нелинейностью и softmax на каждую клетку в конце, для предсказания хода по текущему состоянию. Используем для выбора хода.

Rollout policy
Быстрая логистическая регрессия на большом количестве признаков. Используется для игры с самой собой.



SL policy network

Input Layer: 19x19x48 image stack

1 hidden layer:

- zero padding to 23x23
- convolving 192 filters of size 5x5 with stride equal to 1
- applying rectifier unit

2-12 hidden layer:

- zero padding to 21x21
- convolving 192 filters of size 3x3 with stride equal to 1
- applying rectifier unit

Output layer:

- convolving 1 filter of size 1x1 with stride equal to 1
- applying softmax to get probability distribution over all possible moves

SL policy network

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

Extended Data Table 2: **Input features for neural networks.** Feature planes used by the policy network (all but last feature) and value network (all features).

Rollout policy

Feature	# of patterns	Description
Response	1	Whether move matches one or more response features
Save atari	1	Move saves stone(s) from capture
Neighbour	8	Move is 8-connected to previous move
Nakade	8192	Move matches a *nakade* pattern at captured stone
Response pattern	32207	Move matches 12-point diamond pattern near previous move
Non-response pattern	69338	Move matches 3×3 pattern around move
Self-atari	1	Move allows stones to be captured
Last move distance	34	Manhattan distance to previous two moves
Non-response pattern	32207	Move matches 12-point diamond pattern centred around move

Extended Data Table 4: **Input features for rollout and tree policy.** Features used by the rollout policy (first set) and tree policy (first and second set). Patterns are based on stone colour (black/white/empy) and liberties $(1, 2, \geq 3)$ at each intersection of the pattern.

Reinforcement learning of policy networks

Каждая итерация состоит из n матчей, играемых параллельно между policy network и противником, который выбирается из пула прошлых версий сети.

Пул обновляется каждый 500 итераций.

p_{ϱ} - обучаемая сеть

$p_{\varrho'}$ - случайный противник

Веса ϱ и ϱ' инициализируются с весами σ SL policy network

RL policy

Играем i-ю игру до её завершения - T^i

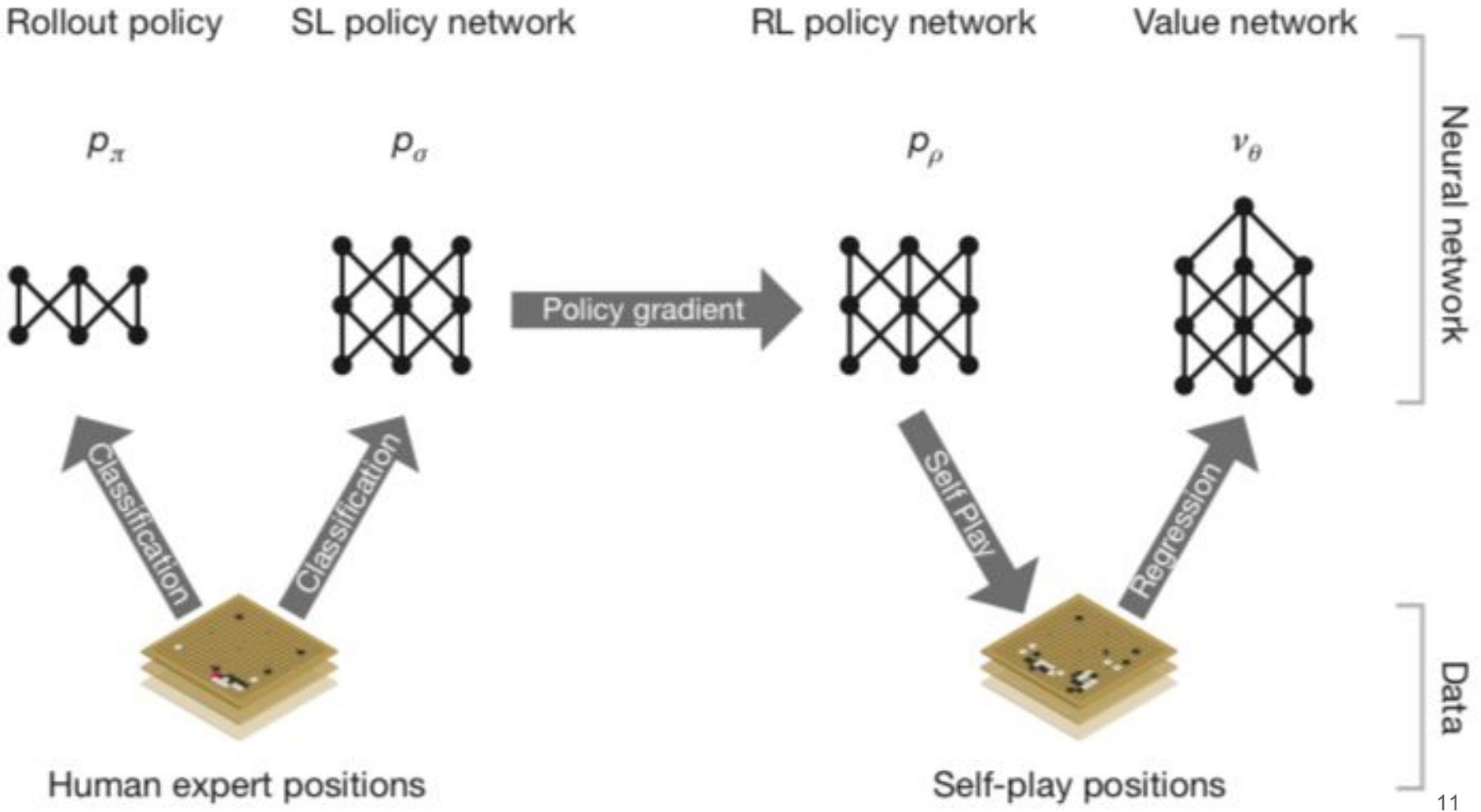
Результат игры $z_t^i = \pm r(s_{Ti})$

v - оценка вероятности выигрыша от value network

Обновляем веса:

$$\Delta p = \frac{\alpha}{n} \sum_{i=1}^{n} \sum_{t=1}^{T^i} \frac{\partial log p_\rho(a_t^i \mid s_t^i)}{\partial \rho} (z_t^i - v(s_t^i))$$

10.000 итераций 128 игр занимает 1 день


Rollout policy
SL policy network
RL policy network
Value network
p_π
p_σ
p_ρ
v_θ
Policy gradient
Classification
Classification
Self Play
Regression
Neural network
Data
Human expert positions
Self-play positions

MCTS

Каждый узел (s, a) отвечает за состояние доски, ход и хранит внутри себя информацию:

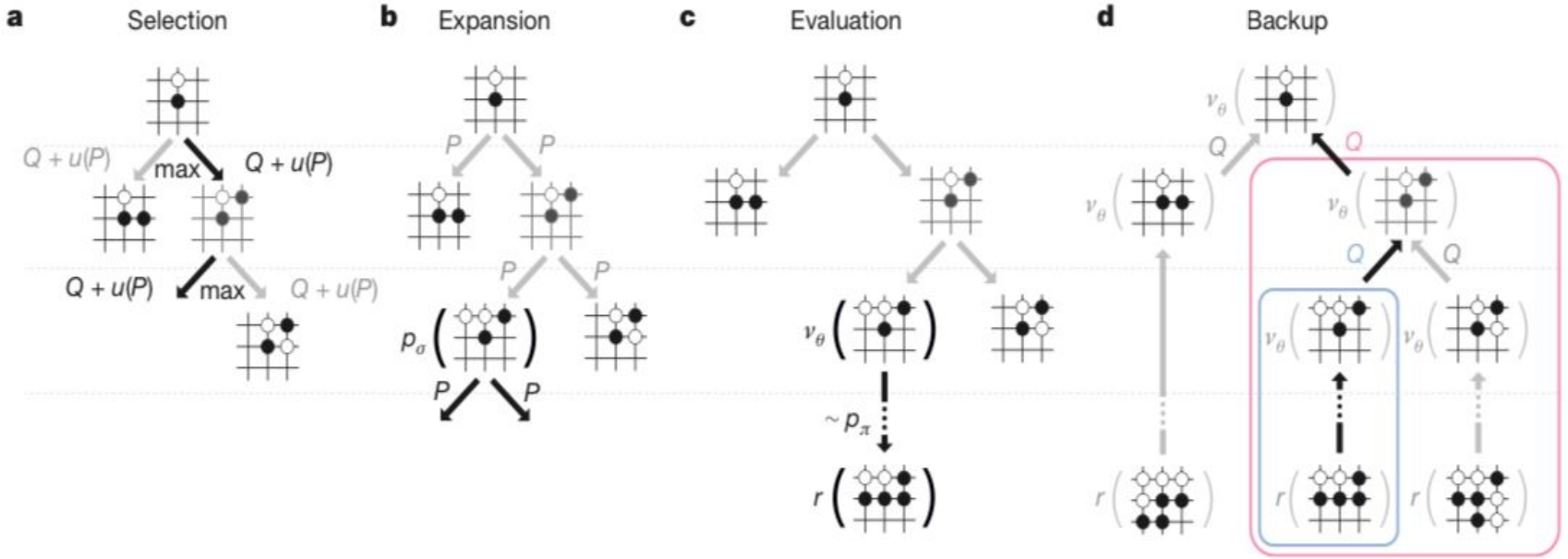
- action value(s, a)
- количество посещений N(s, a)
- априорная вероятность P(s, a)

Имея эти данные в каждом узле выбираем ход:

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

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

MCTS



- Q - средняя оценка выигрыша
- u(P) - то, насколько ходим расширять наше исследование
- V - значения value network

Подведение итогов

+ 2015 год AlphaGo выиграла матч у четырехкратного чемпиона Европы Фань Хуэя.
+ 2016 год AlphaGo выиграла матч у Ли Седоля.
+ Хороший скор

- Сложность реализации
- Большой объем данных
- Трудозатратно по ресурсам

Бонус

Список литературы:

Mastering the game of Go with deep neural networks and tree search
(https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf)