Đại học Quốc gia Hà Nội - Trường Đại học Khoa học Tự nhiên

AI chơi cờ vây

Nhóm 20

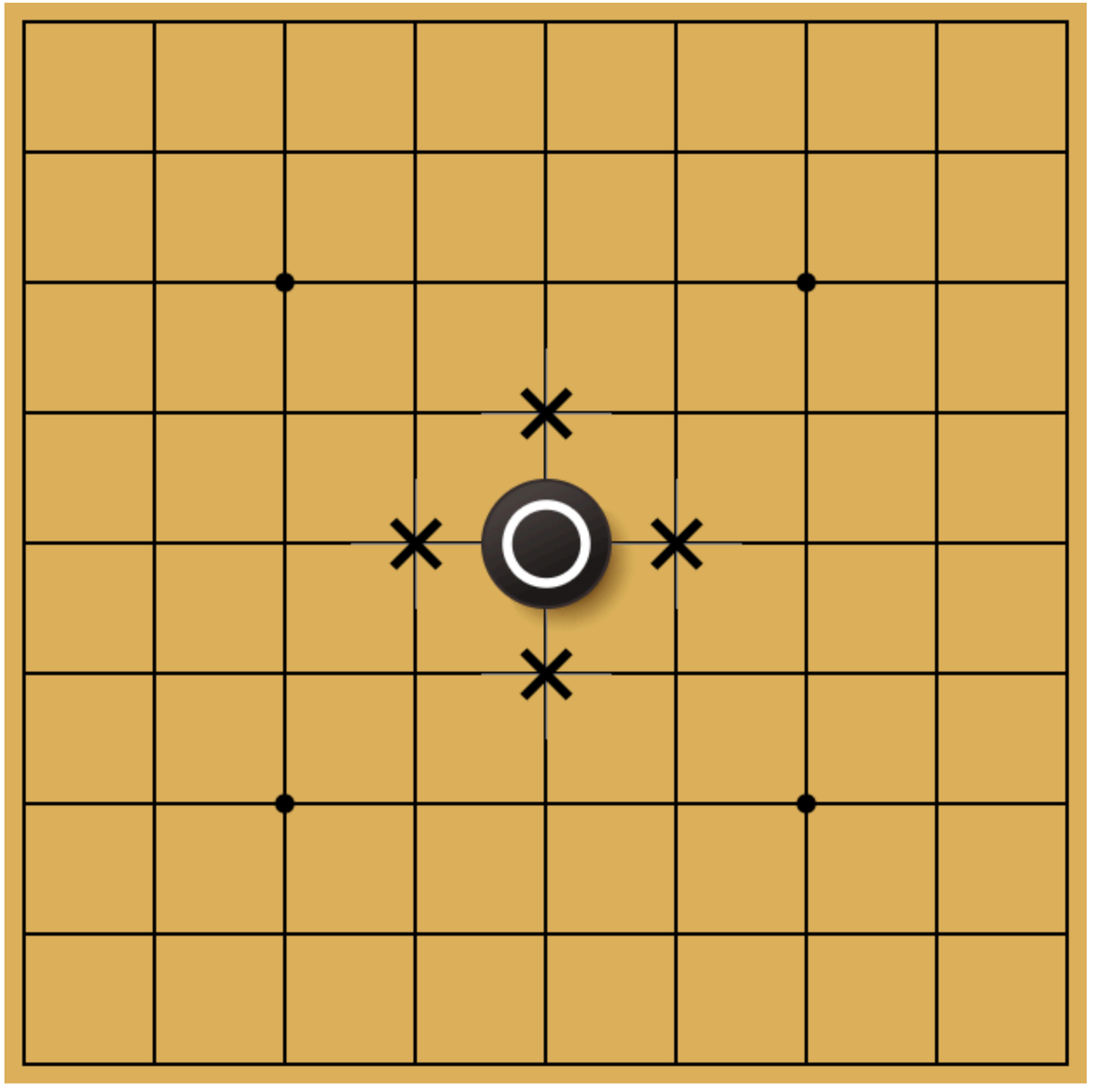
Hồ Quang Chung
Bùi Đức Hiếu
Nguyễn Khánh Toàn

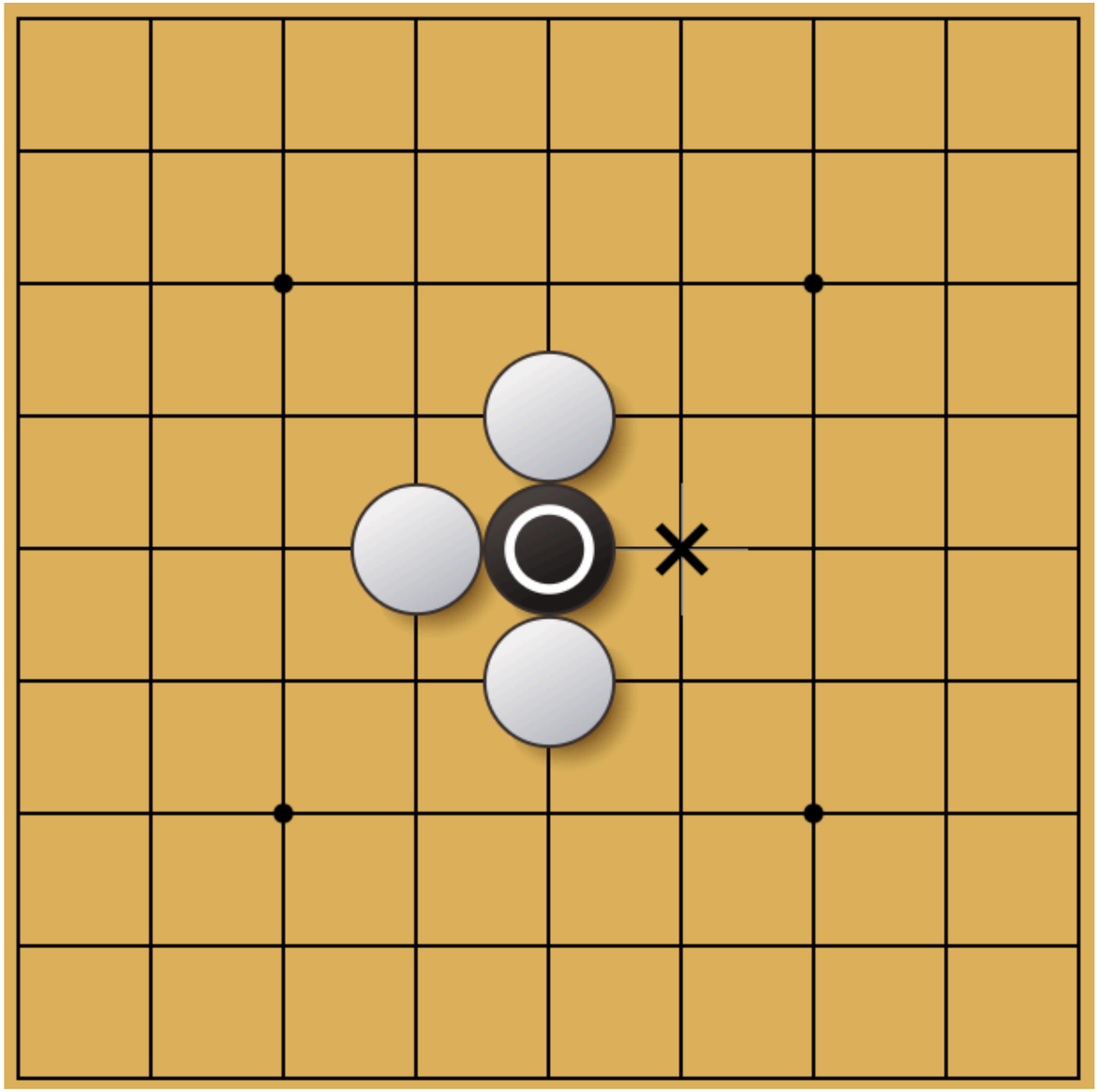
MAT3508 - Nhập môn Trí tuệ Nhân tạo

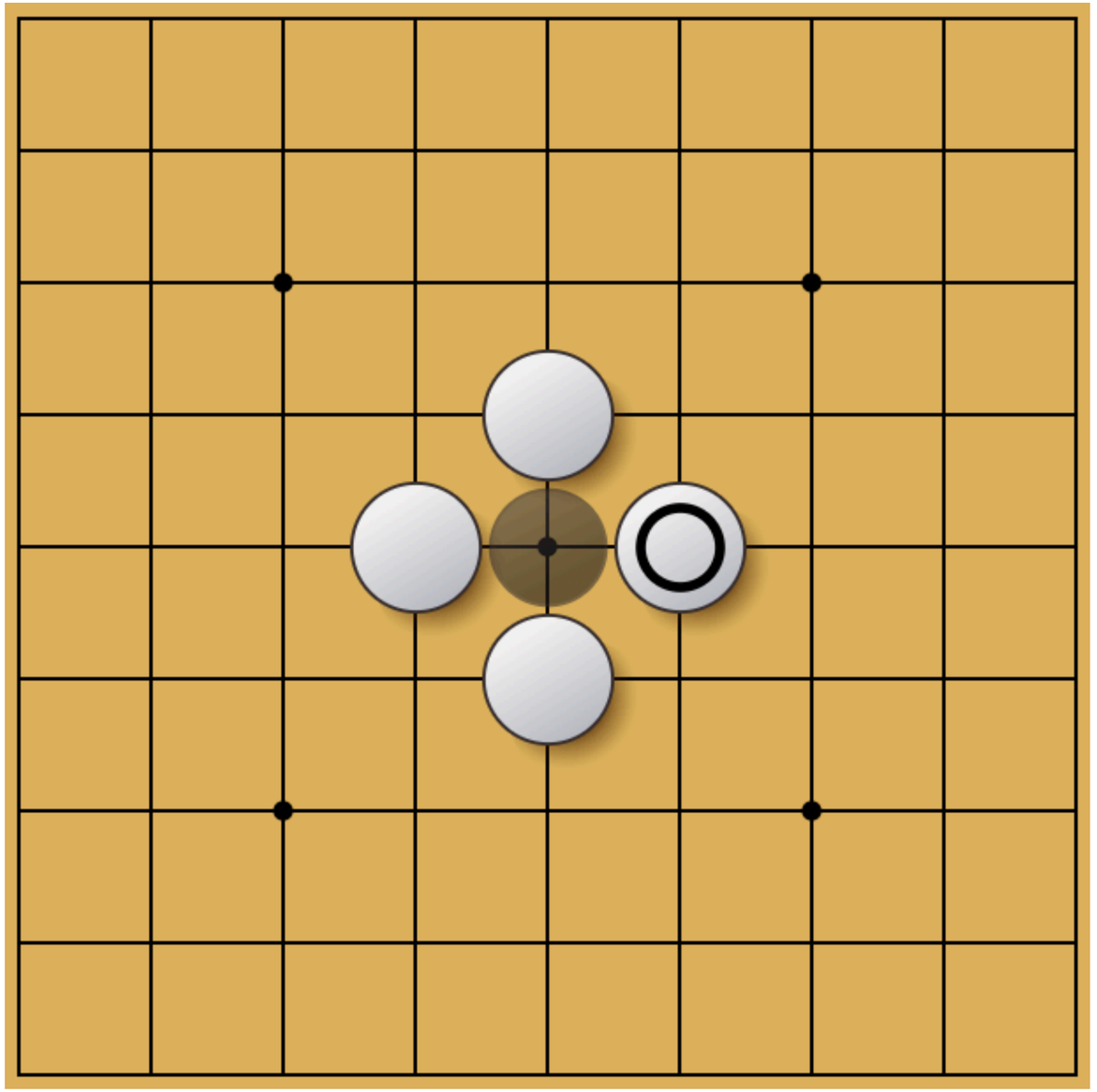
Đặt vấn đề

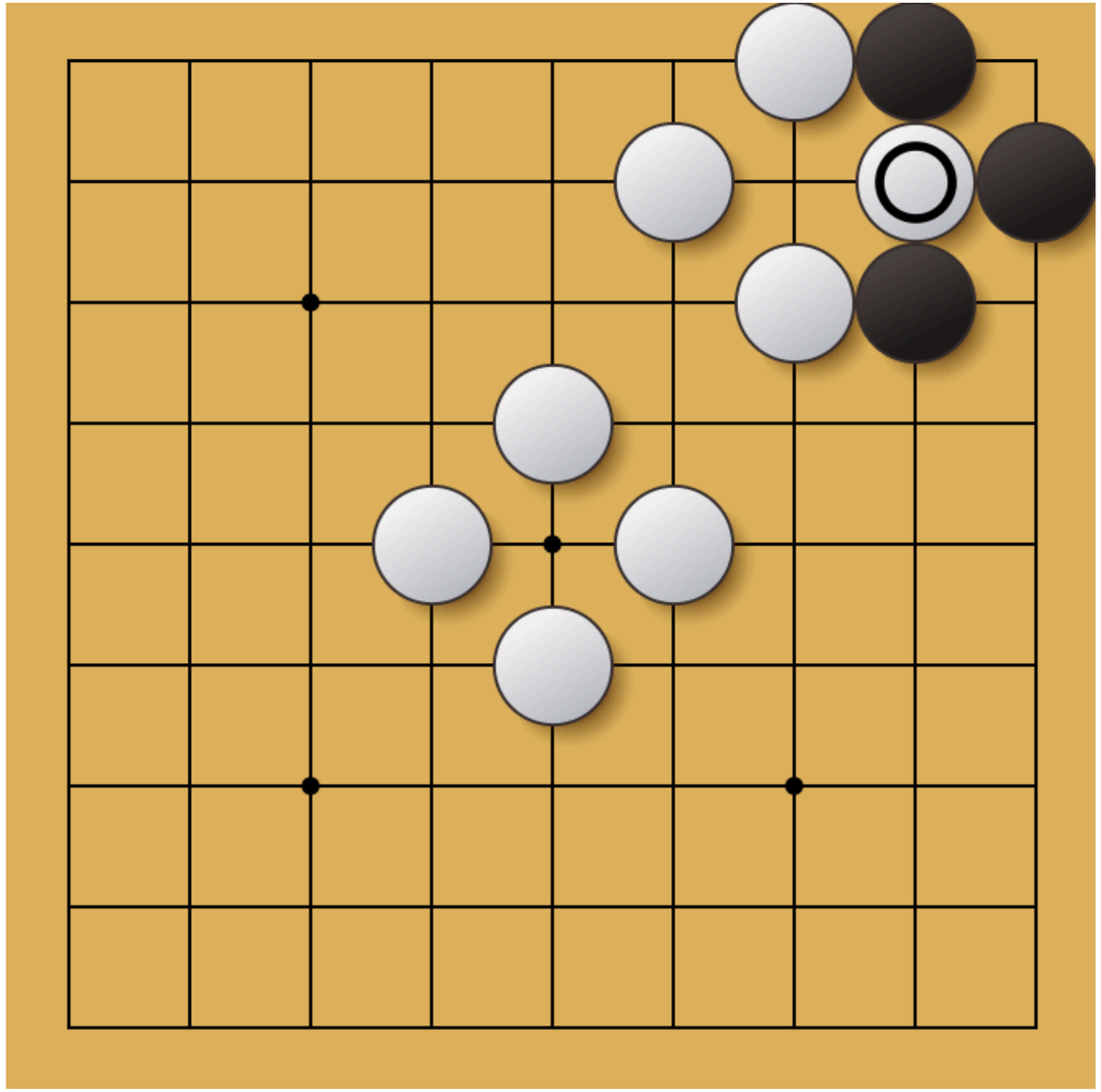
Trên bàn cờ 19 × 19, có tổng cộng 361 giao điểm. Ước tính số lượng trạng thái hợp lệ của bàn cờ lên đến khoảng 2,1 × 10^170, một con số khổng lồ vượt xa tổng số nguyên tử trong vũ trụ mà chúng ta quan sát được (khoảng 10^80)

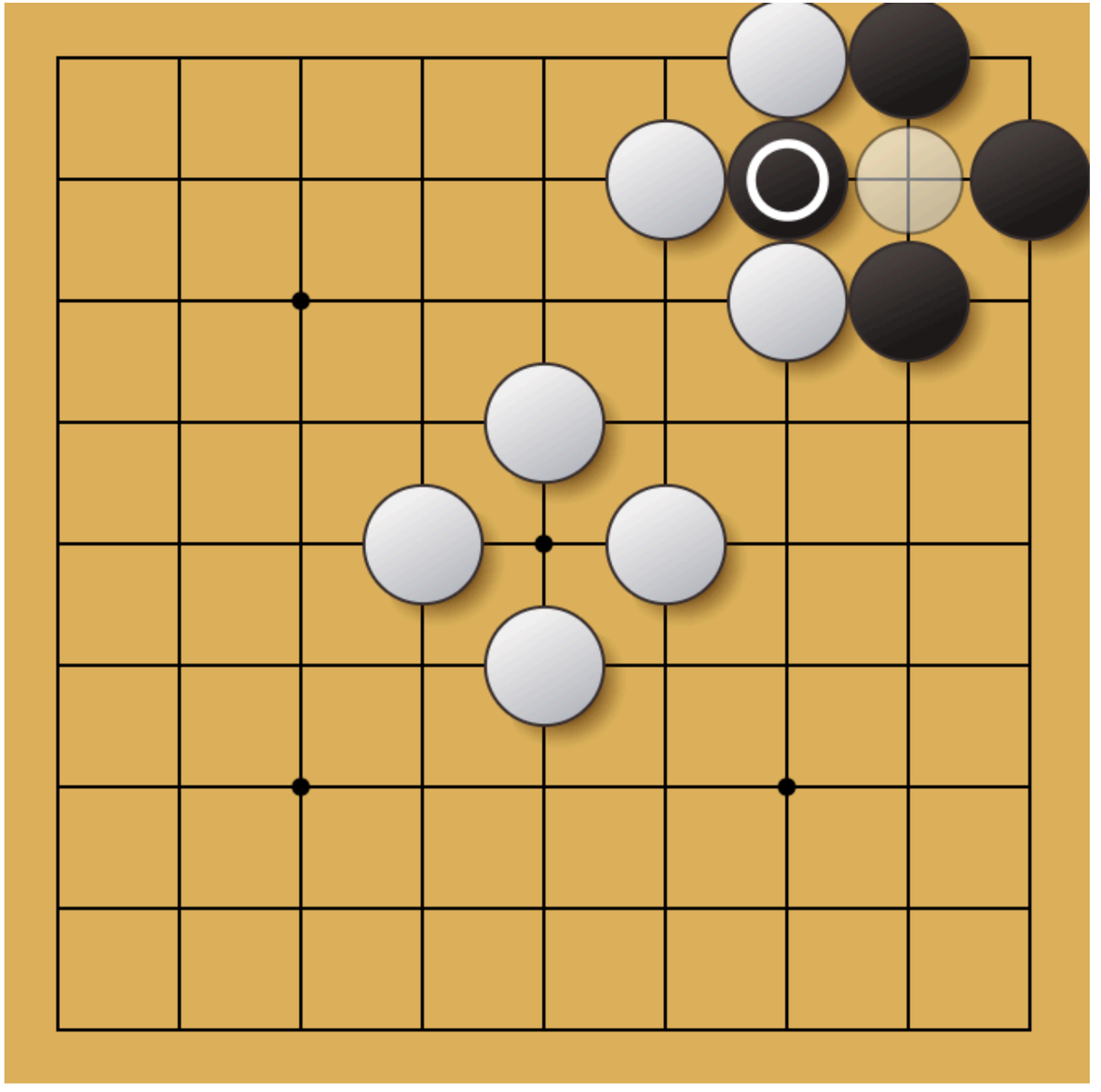
LUẬT CHƠI











Người chơi có thể chọn bỏ lượt. Khi cả 2 người cùng bỏ lượt sẽ đến phần tính điểm

Người ta thường chỉ đếm quân và đất của một bên (thường là bên Đen) để xác định kết quả:

- Nếu Tổng điểm của Đen (Đất + Quân) > 184.25 (thực tế là ≥ 185 trên bàn cờ): Đen Thắng.
- Nếu Tổng điểm của Đen ≤ 184.25: Trắng Thắng

PHƯƠNG PHÁP

Kiến trúc hệ thống

Hệ thống hoạt động theo mô hình Client-Server phân tán, tách biệt giữa giao diện người dùng, logic trò chơi và tính toán AI.

- Frontend (ReactJS): Đóng vai trò lớp Presentation. Xử lý tương tác người dùng, hiển thị bàn cờ, gửi dữ liệu nước đi và nhận kết quả từ Server.
- Backend (.NET 10): Được thiết kế theo mô hình Domain-Driven Design (DDD). Chịu trách nhiệm quản lý trạng thái ván đấu (State Management), thực thi luật cờ vây, và cung cấp API RESTful/WebSocket.
- AI Core: Module tích hợp trong Backend, sử dụng ONNX Runtime để chạy mô hình Deep Learning và thuật toán MCTS để đưa ra quyết định nước đi.

Dữ liệu

Sử dụng dataset từ featurecat/go-dataset, tập trung lọc các ván đấu của kỳ thủ chuyên nghiệp (Professional Dan rank: 1p - 9p).

Bàn cờ 19×19 được biểu diễn dưới dạng Tensor đa kênh với kích thước [Batch_Size, 19, 19, C].
Các kênh thông tin bao gồm:

- Quân ta (Player Stones): 1 nếu có quân, 0 nếu không.
- Quân đối thủ (Opponent Stones): 1 nếu có quân, 0 nếu không.
- Vị trí trống (Empty): Đánh dấu các điểm chưa có quân.
- Lịch sử (History): Trạng thái bàn cờ tại các thời điểm $t-1$, $t-2$, . . . để giúp mạng nhận diện luật Ko (cướp lại).
- Lượt đi (Color): Plane chứa toàn số 1 (nếu Đen đi) hoặc 0 (nếu Trắng đi)

Residual CNN - ResNet

Sử dụng dataset từ featurecat/go-dataset, tập trung lọc các ván đấu của kỳ thủ chuyên nghiệp (Professional Dan rank: 1p - 9p).

Bàn cờ 19×19 được biểu diễn dưới dạng Tensor đa kênh với kích thước [Batch_Size, 19, 19, C].
Các kênh thông tin bao gồm:

- Quân ta (Player Stones): 1 nếu có quân, 0 nếu không.
- Quân đối thủ (Opponent Stones): 1 nếu có quân, 0 nếu không.
- Vị trí trống (Empty): Đánh dấu các điểm chưa có quân.
- Lịch sử (History): Trạng thái bàn cờ tại các thời điểm $t-1$, $t-2$, . . . để giúp mạng nhận diện luật Ko (cướp lại).
- Lượt đi (Color): Plane chứa toàn số 1 (nếu Đen đi) hoặc 0 (nếu Trắng đi)

Residual CNN - ResNet

Residual Block: $y = \sigma(F(x, \{W_i\}) + x)$

Cấu trúc Dual-head: Mạng chia sẻ các lớp Convolutional ban đầu và tách thành hai nhánh ở cuối:

1. Policy Head (Chiến thuật):
   - Output: Vector xác suất p kích thước 19 × 19 + 1 (bao gồm nước Pass).
   - Hàm kích hoạt: Softmax.
   - Ý nghĩa: pa = P (a|s) là xác suất chọn nước đi a tại trạng thái s.
2. Value Head (Giá trị):
   - Output: Giá trị thực v ∈ [−1, 1].
   - Hàm kích hoạt: Tanh.
   - Ý nghĩa: Dự đoán khả năng thắng (1) hoặc thua (-1) từ trạng thái hiện tại

Residual CNN - ResNet

Residual Block: $y = \sigma(F(x, \{W_i\}) + x)$ Loss Function: $L = (z - v)^2 - \pi^T \log(p) + c||\theta||^2$

Cấu trúc Dual-head: Mạng chia sẻ các lớp Convolutional ban đầu và tách thành hai nhánh ở cuối:

1. Policy Head (Chiến thuật):
   - Output: Vector xác suất p kích thước 19 × 19 + 1 (bao gồm nước Pass).
   - Hàm kích hoạt: Softmax.
   - Ý nghĩa: pa = P (a|s) là xác suất chọn nước đi a tại trạng thái s.
2. Value Head (Giá trị):
   - Output: Giá trị thực v ∈ [−1, 1].
   - Hàm kích hoạt: Tanh.
   - Ý nghĩa: Dự đoán khả năng thắng (1) hoặc thua (-1) từ trạng thái hiện tại

Residual CNN - ResNet

Việc sử dụng CNN và ResNet Dual-head là giải pháp tối ưu cho Cờ Vây vì:

- CNN: Xử lý bàn cờ 19x19 như một ảnh, giúp mạng nhận diện hiệu quả các mẫu hình và đặc trưng cục bộ như "mắt cờ" hay "dáng quân" (shape).
- ResNet: Sử dụng các kết nối tắt ($y = F(x) + x$) để xây dựng mạng rất sâu. Điều này chống lại vấn đề biến mất đạo hàm (vanishing gradient), cho phép AI có tầm nhìn chiến lược toàn cục (ví dụ: Chinh quân) thay vì chỉ chiến thuật cục bộ.

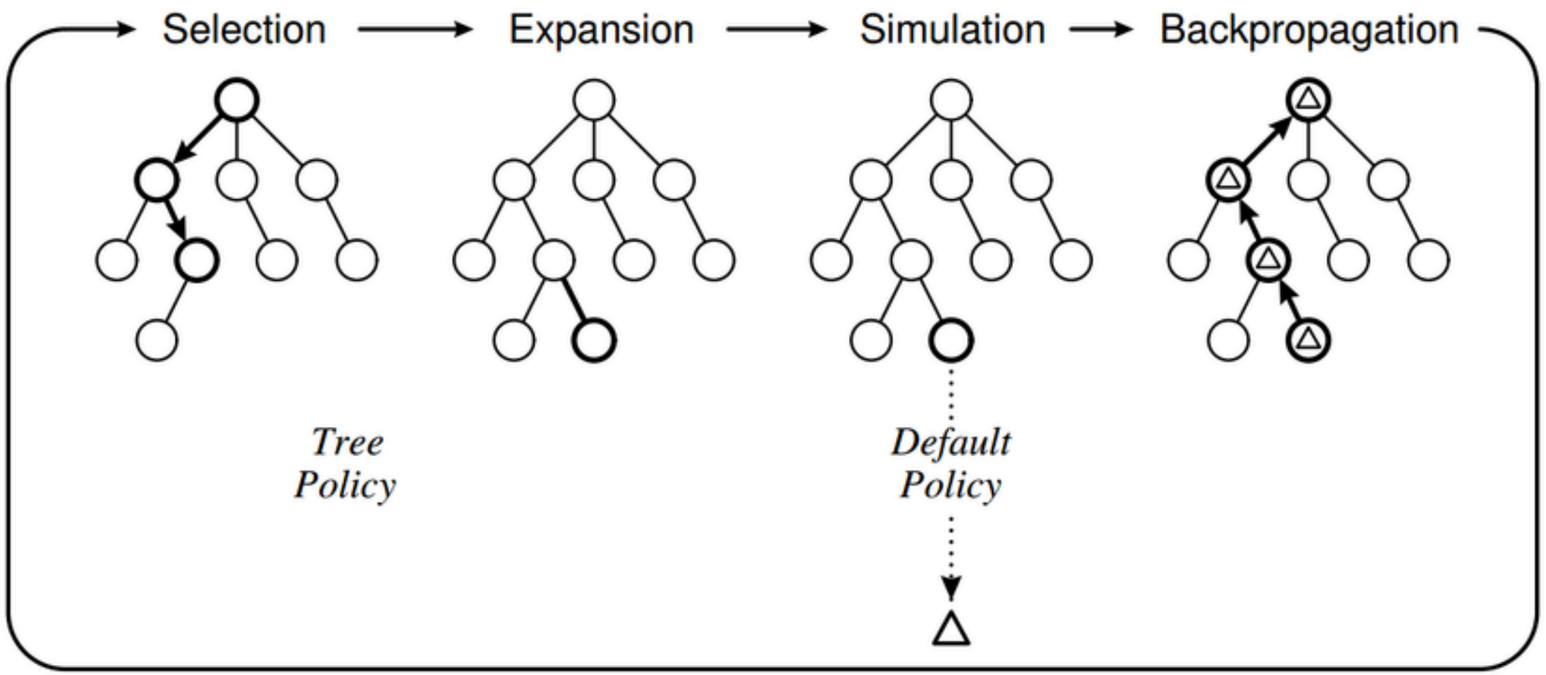
MCTS(PUCT)

Việc chọn Monte Carlo Tree Search (MCTS) thay cho Minimax/Alpha-Beta là bắt buộc do đặc điểm của cờ vây:

- Bùng nổ Tổ hợp (Combinatorial Explosion): cờ vây có trung bình 250 nước đi/lượt và không gian trạng thái khổng lồ (10^170), khiến các thuật toán duyệt cạn (như Minimax) không thể tính toán sâu được.
- Giải pháp: MCTS giải quyết vấn đề này bằng cách duyệt cây theo xác suất (ngẫu nhiên có định hướng). Thuật toán này chỉ tập trung tài nguyên tính toán vào các nhánh triển vọng do Mạng nơ-ron gợi ý, thay vì lãng phí thời gian duyệt tất cả các nước đi kém hiệu quả.

MCTS(PUCT)

Quy trình:



MCTS(PUCT) + Neural Network

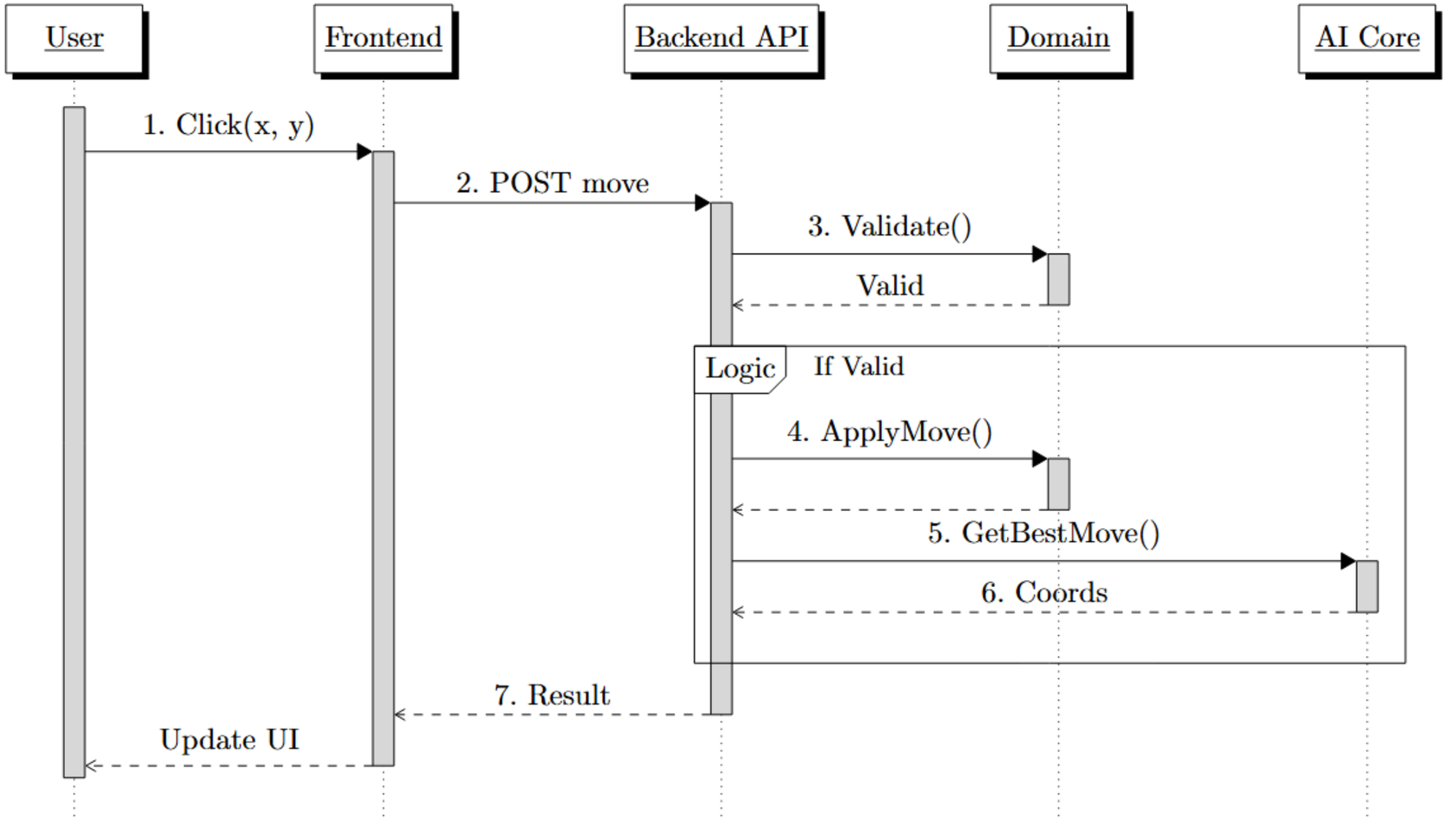
Công thức lựa chọn nút trong MCTS (PUCT):

$$U(s,a) = C_{puct} \cdot P(s,a) \cdot \frac{\sqrt{\sum_b N(s,b)}}{1 + N(s,a)}$$

Mạng Chính sách (Policy Network) P(s, a) giảm chiều rộng tìm kiếm bằng cách loại bỏ các nước đi vô nghĩa, trong khi Mạng Giá trị (Value Network) V(s) giảm chiều sâu tìm kiếm bằng cách đánh giá nhanh thế cờ thay vì phải mô phỏng toàn bộ ván đấu (Rollout)

Frameworks

Thành phần	Công nghệ / Thư viện	Phiên bản (Dự kiến)
Frontend	ReactJS, TypeScript, Vite	React 18+, Vite 5.x
UI/UX	TailwindCSS, HTML5 Canvas (Render bàn cờ)	v3.4+
Backend Runtime	.NET 10 (C#)	Preview/Stable
Kiến trúc Backend	Domain-Driven Design (DDD), Clean Architecture	-
AI Inference	ONNX Runtime (CPU/GPU)	Microsoft.ML.OnnxRuntime
AI Training	Python, PyTorch, NumPy	PyTorch 2.x
Database	PostgreSQL hoặc SQLite (Dev)	Latest
Containerization	Docker, Docker Compose	-


User
Frontend
Backend API
Domain
AI Core
1. Click(x, y)
2. POST move
3. Validate()
Valid
Logic
If Valid
4. ApplyMove()
5. GetBestMove()
6. Coords
7. Result
Update UI

Tiền xử lý dữ liệu

- Chỉ chọn các ván bàn 19x19, không chấp quân (Handicap = 0) và có kết quả thắng thua rõ ràng, chỉ chọn những ván chơi từ những cờ thủ chuyên nghiệp (tầm 10000 ván)
- Lưu dữ liệu thành nhiều file nhỏ, mỗi file chứa khoảng 2000-5000 ván để giảm tải

Huấn luyện

Sử dụng kiến trúc ResNet Dual-head lấy cảm hứng từ AlphaGo Zero.

- Backbone: 1 lớp Convolutional đầu vào + 10 khối Residual Blocks (mỗi khối gồm Conv3x3, Batch Normalization, ReLU và Skip Connection).
- Policy Head (Đầu ra chiến thuật): Output vector 362 chiều (xác suất nước đi).
- Value Head (Đầu ra giá trị): Output scalar [−1, 1] (dự đoán khả năng thắng

Hàm mất mát: $$L_{total} = L_{policy} + L_{value}$$

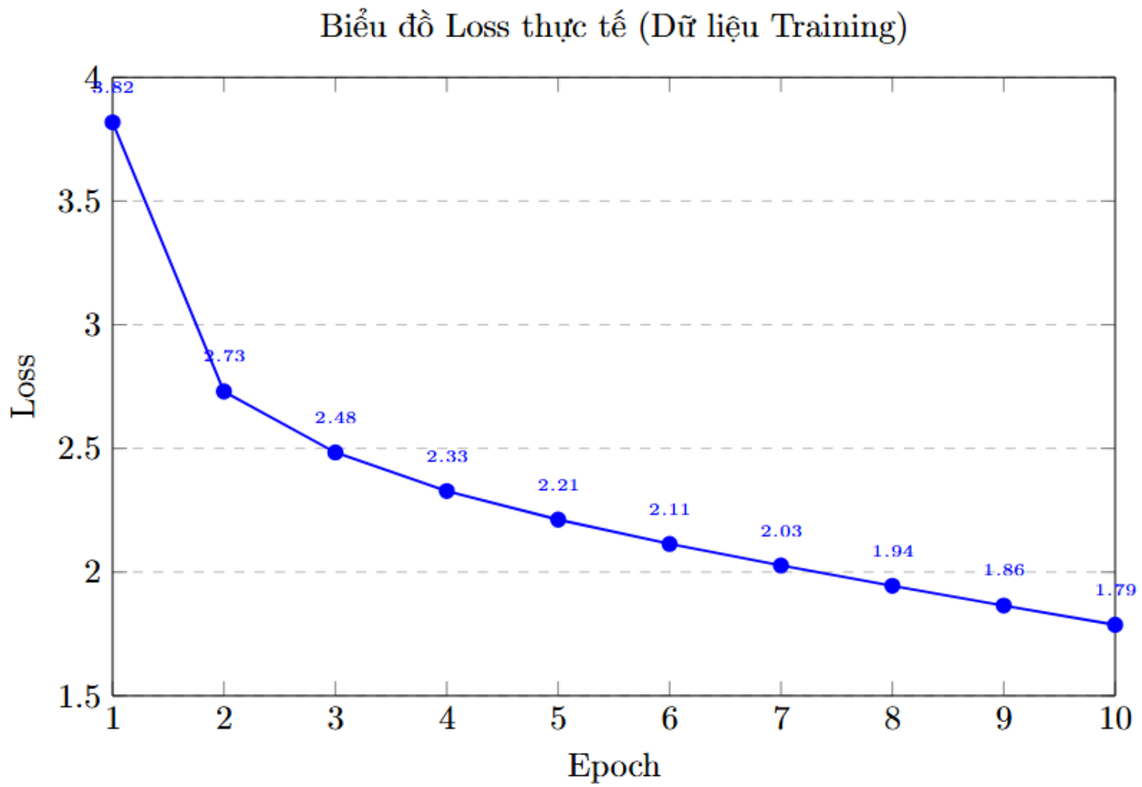
Fine-tuning

Khi cần cập nhật model với dữ liệu mới hoặc phong cách chơi mới:

1. Load trọng số (Weights) từ file model ‘.pth‘ đã train trước đó.
2. Giữ nguyên kiến trúc mạng.
3. Giảm Learning Rate xuống thấp (thường là 10−4 hoặc 10−5) để tránh hiện tượng Catastrophic Forgetting (quên kiến thức cũ).
4. Train trên tập dữ liệu mới với số lượng Epochs ít hơn

KẾT QUẢ

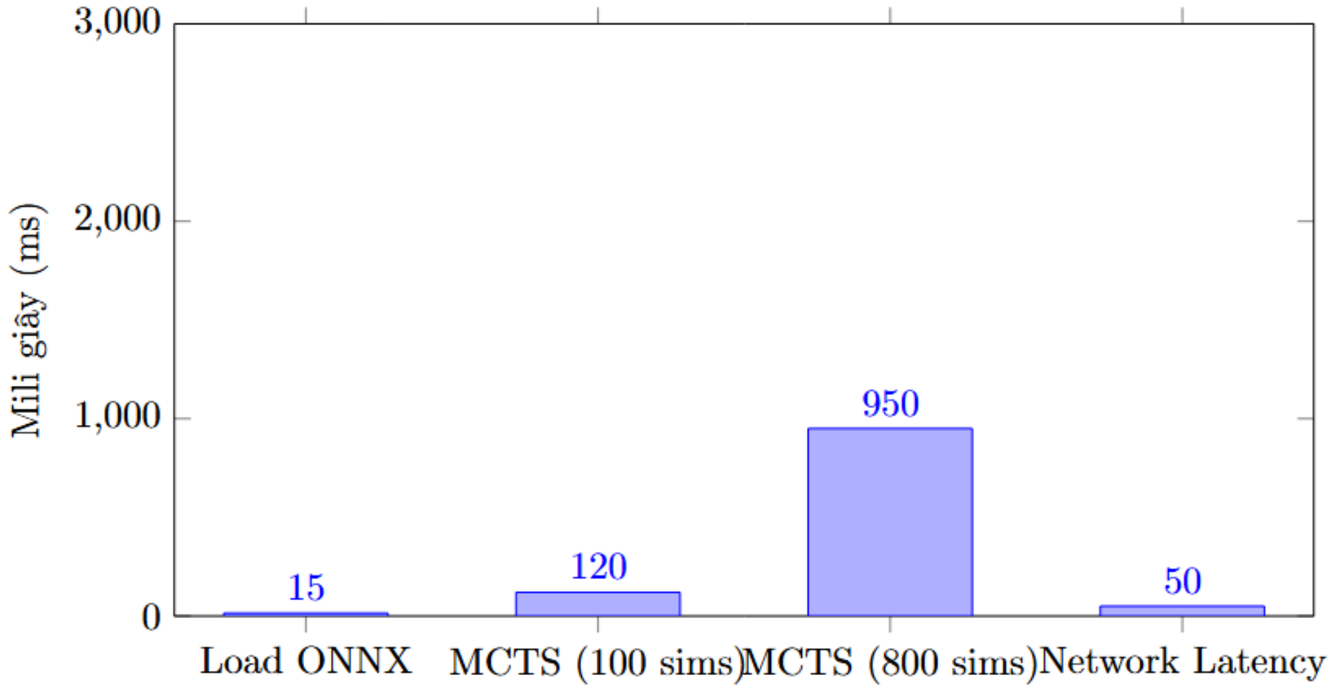
Quá trình huấn luyện được thực hiện với Batch Size = 256 trên Google Colab T4 GPU. Kết quả ghi nhận trong 10 Epochs đầu tiên cho thấy sự hội tụ tích cực của mô hình



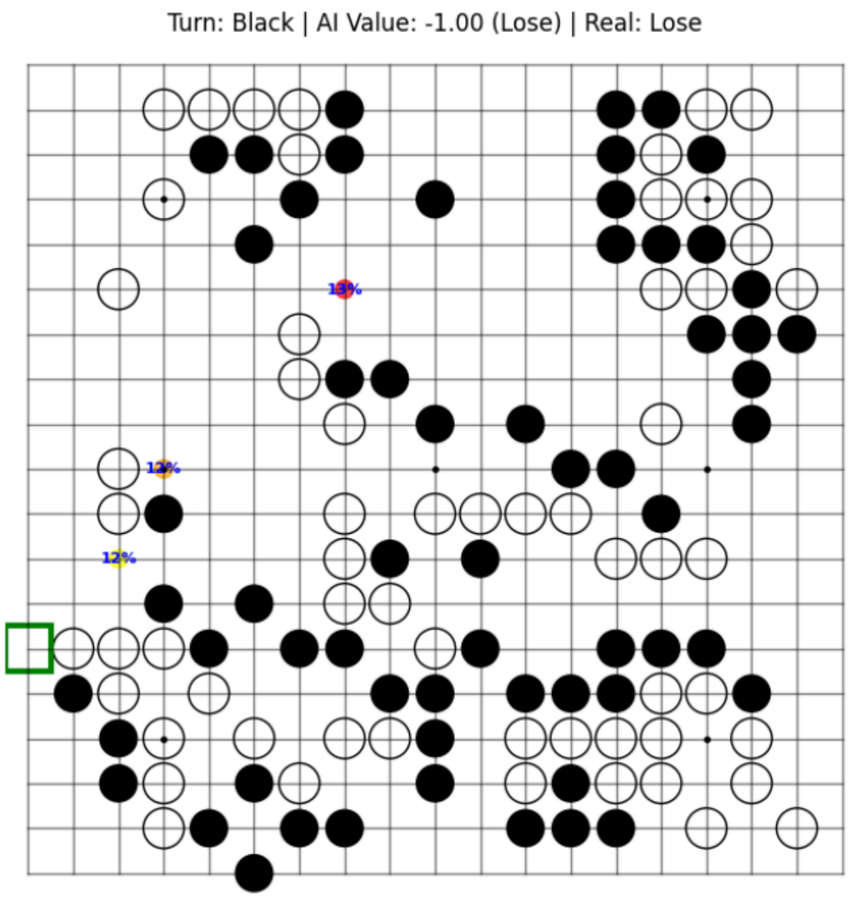
Epoch	Policy Top-1 (%)	Policy Top-5 (%)	Value Accuracy (%)	Value MSE
1	22.5%	45.1%	52.3%	0.401
3	35.8%	62.4%	58.1%	0.302
5	41.2%	70.5%	70.7%	0.254
8	46.8%	75.8%	85.4%	0.112
10 (Final)	**57.4%**	**87.7%**	**97.0%**	**0.074**

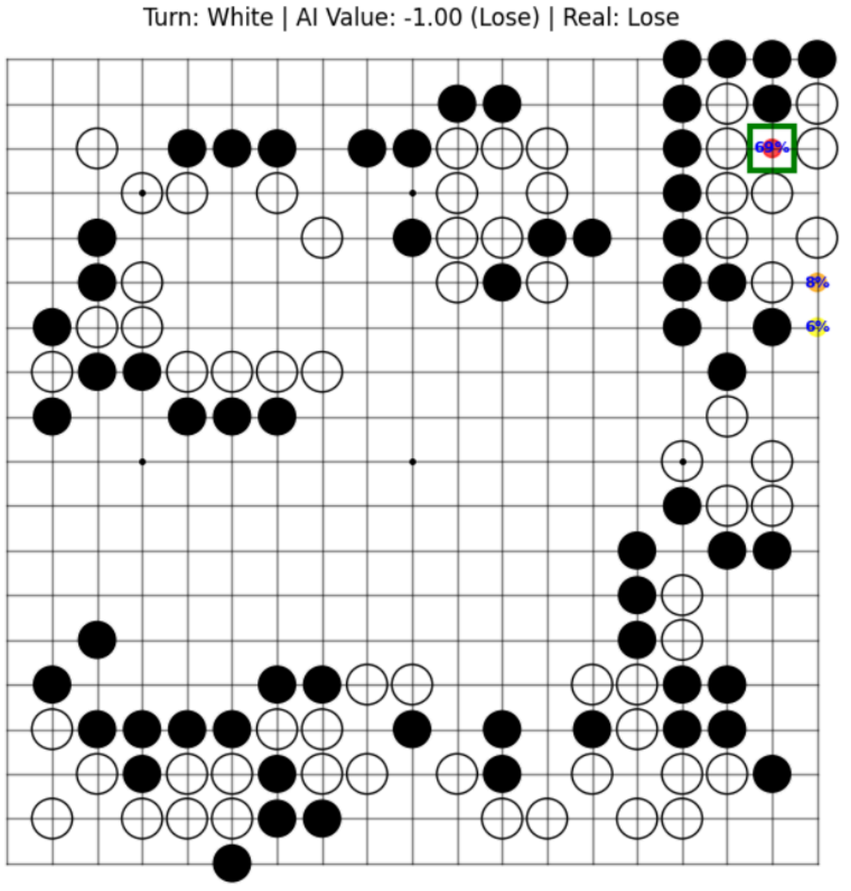
Nhìn chung, mô hình dự đoán nước đi có độ chính xác cao. Tuy nhiên, do việc dự đoán chính xác kết quả thắng/thua ngay từ khai cuộc là khó, nên việc dự đoán kết quả ván cờ còn khó khăn.

- Thời gian thực thi MCTS với 800 simulations là khoảng 950ms (gần 1 giây). Đây là mức thời gian chấp nhận được cho trải nghiệm người dùng Web thời gian thực.



- Việc sử dụng thư viện Microsoft.ML.OnnxRuntime trên .NET cho tốc độ inference trung bình khoảng 1.2ms/batch (với Batch=1 trên CPU), đảm bảo khả năng mở rộng khi có nhiều người chơi cùng lúc


Turn: Black | AI Value: -1.00 (Lose) | Real: Lose
13%
12%
12%


Turn: White | AI Value: -1.00 (Lose) | Real: Lose
8%
6%

Kết luận

- Kết quả huấn luyện xác nhận mô hình AI Cờ Vây hiệu quả và tin cậy. Mô hình ResNet Dual-head bắt chước hành vi chuyên nghiệp tốt và đánh giá tình thế vượt trội, hỗ trợ thuật toán MCTS.
- Mô hình đạt cân bằng giữa "Trực giác" (Policy Network 46% accuracy) và "Suy luận" (MCTS 800 sims). Hệ thống hoạt động ổn định trên .NET 10, sẵn sàng cho ứng dụng cờ vây trực tuyến bán chuyên nghiệp và mở rộng chế độ chơi.

Hướng phát triển

1. **Mở rộng Trải nghiệm Người dùng (UX Community):**
- Chế độ chơi giữa người với người qua server.
- Tính năng: Bảng xếp hạng, hệ thống Elo, lịch sử ván đấu.
- Chế độ phân tích ván đấu: AI phân tích và cung cấp điểm số, nước đi tối ưu.
- Trình chỉnh sửa ván cờ: Tải lên, lưu trữ, xem lại ván đấu định dạng SGF.
2. **Công cụ Huấn luyện:**
- Học theo chủ đề: Bài tập về sống/chết và khai cuộc với AI.
- Tính năng: Kiểm tra giải đố, gợi ý, hiển thị giải pháp tối ưu.
- Chế độ gợi ý và giải thích: AI cung cấp nước đi tốt nhất và giải thích.
3. **Tối ưu hóa Hiệu năng Tech Stack:**
- Triển khai phân tán: Kiến trúc microservices và GPU/TPU cho MCTS.
- Huấn luyện lại mô hình: Cải thiện AI qua chơi tự động.