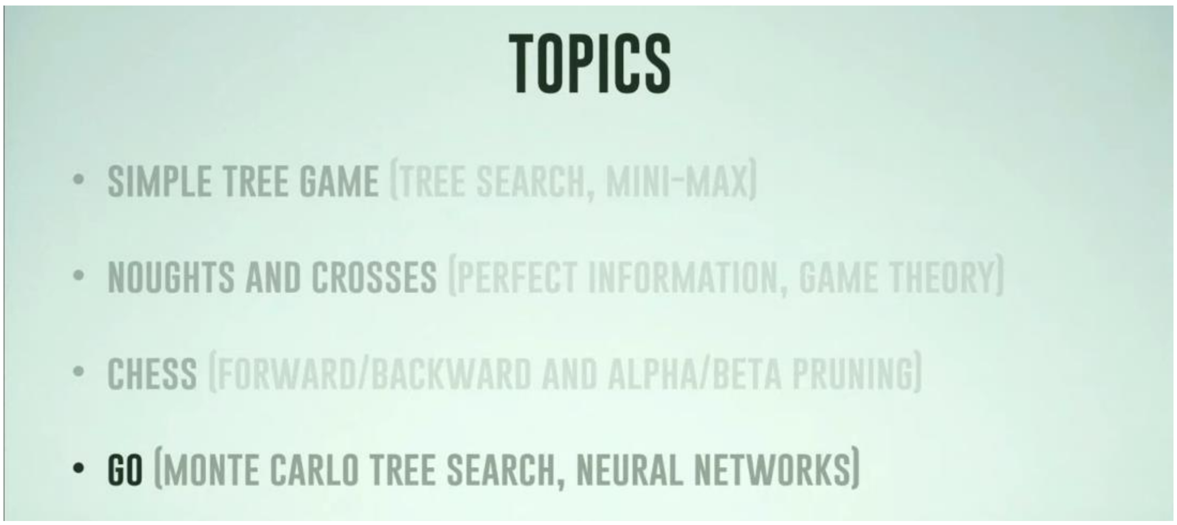
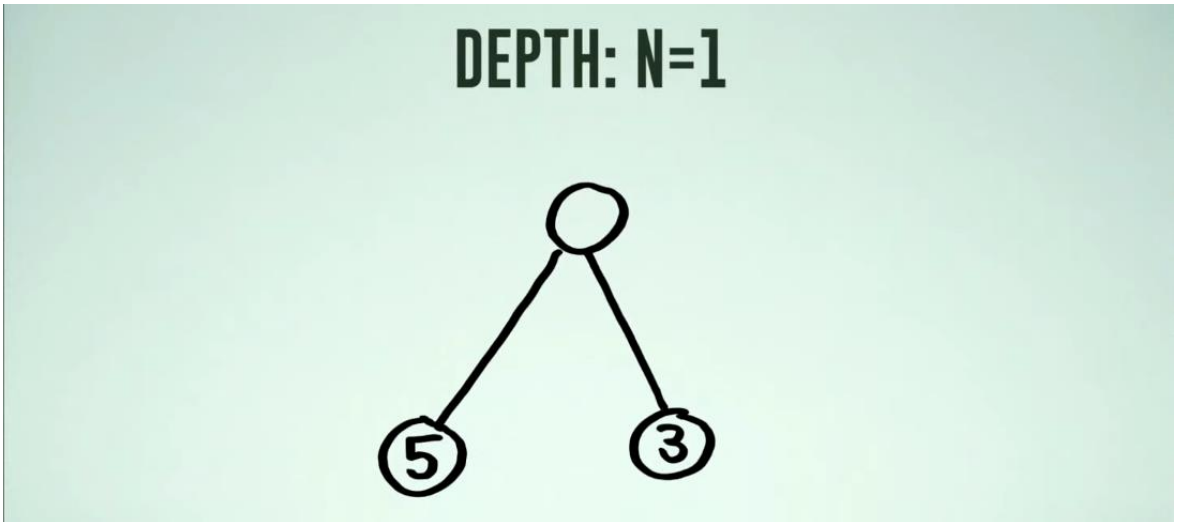
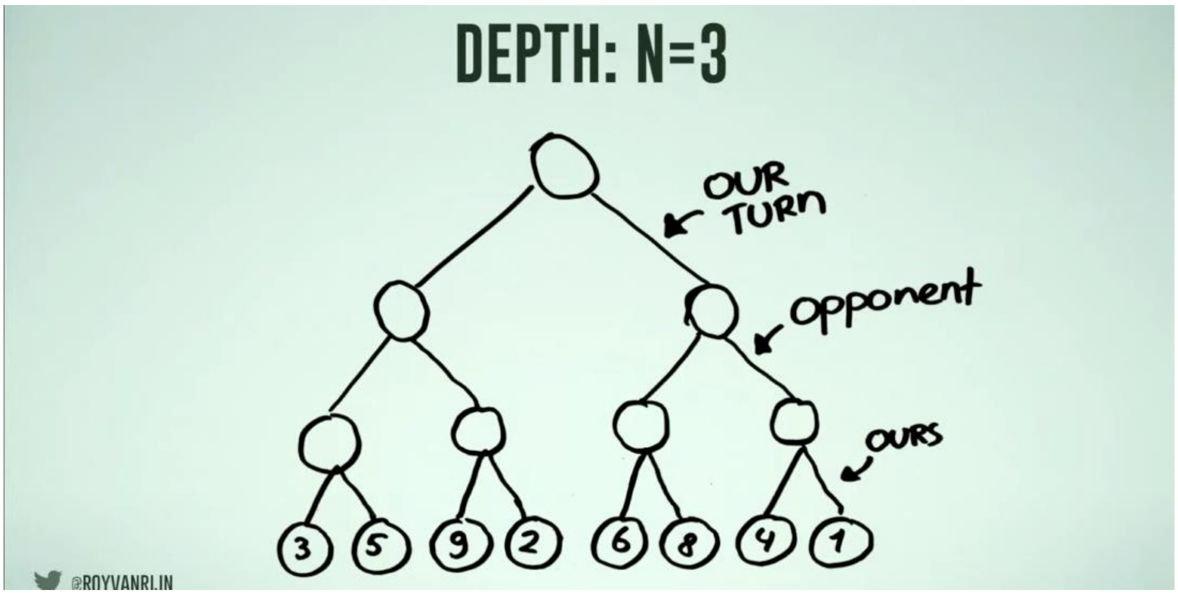
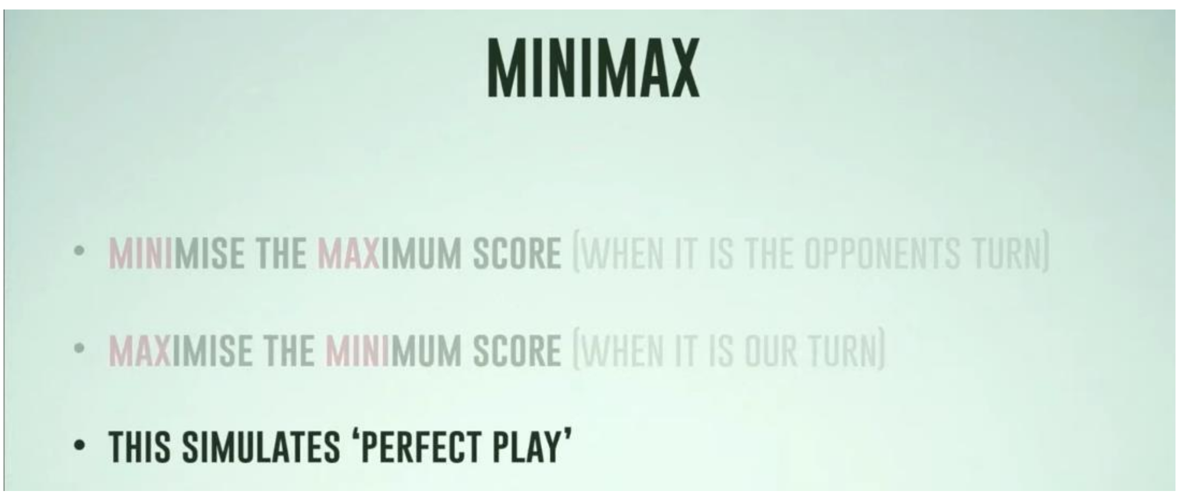
Google DeepMind's AlphaGo was an extraordinary breakthrough for Artificial Intelligence. The game of Go has 1.74×10^172 unique positions and is about a 'googol' times harder to calculate than chess. Experts thought it would take at least another decade before A.I. would be able to beat the best human players. So how did DeepMind tackle this problem? What algorithms did they [...]

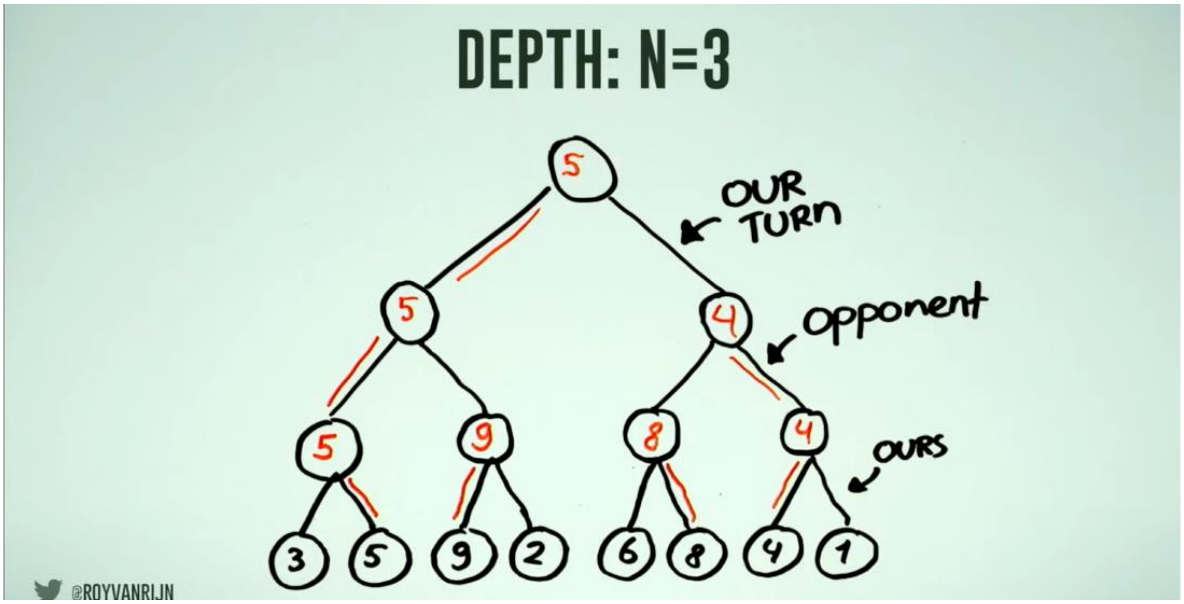






This is no longer trivial and we can use the minimax algorithm for this

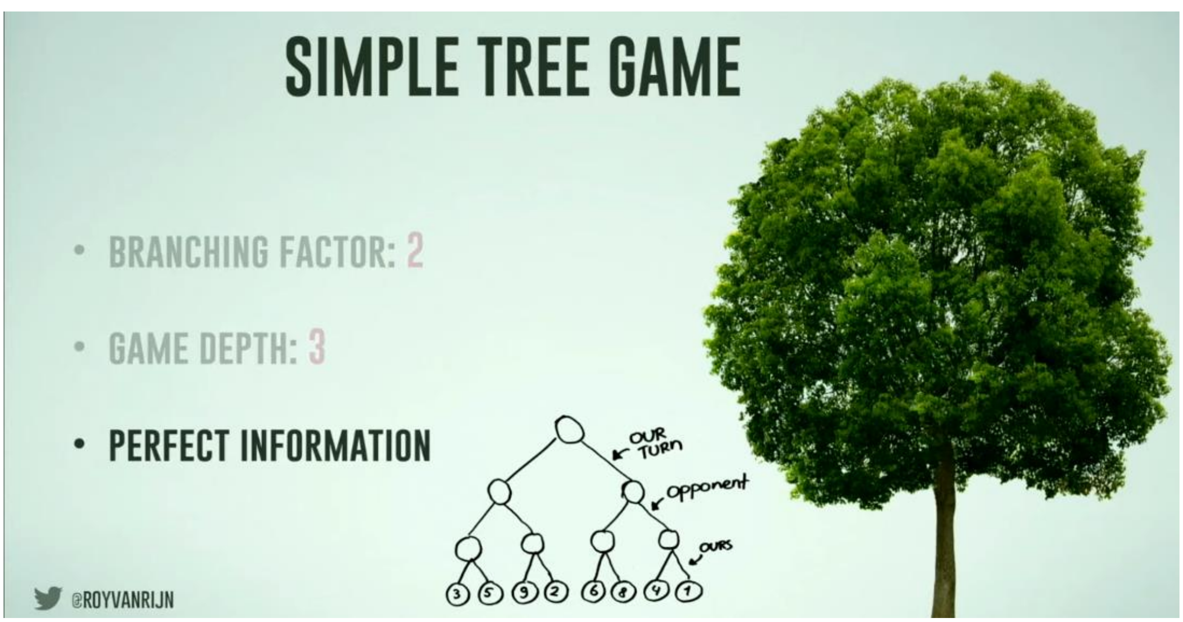


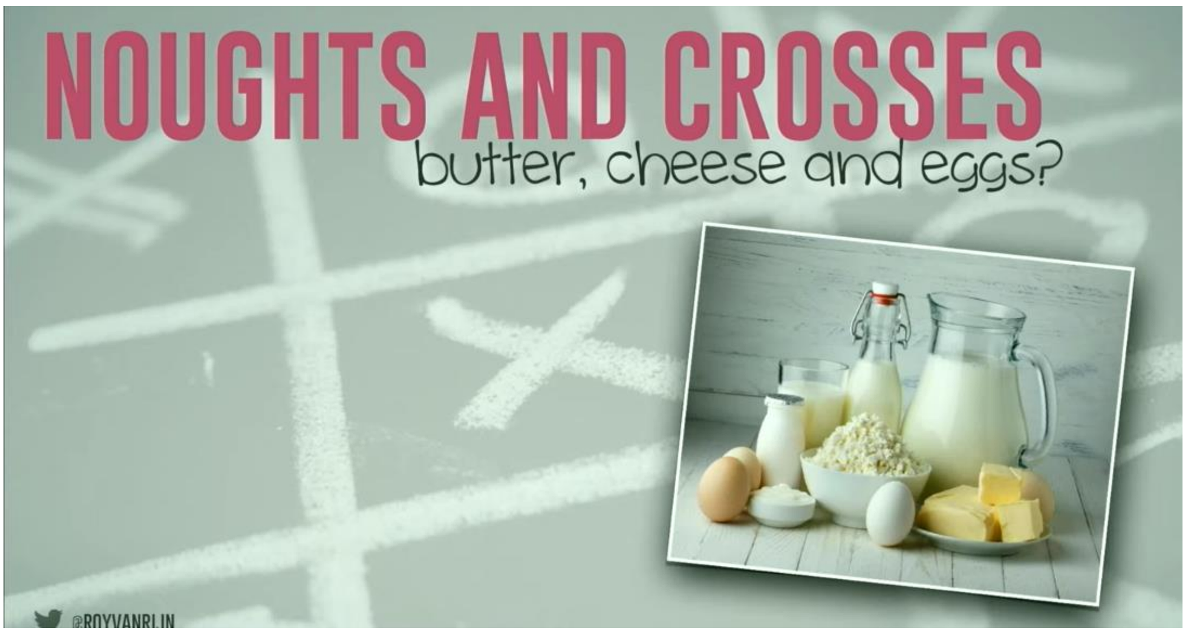


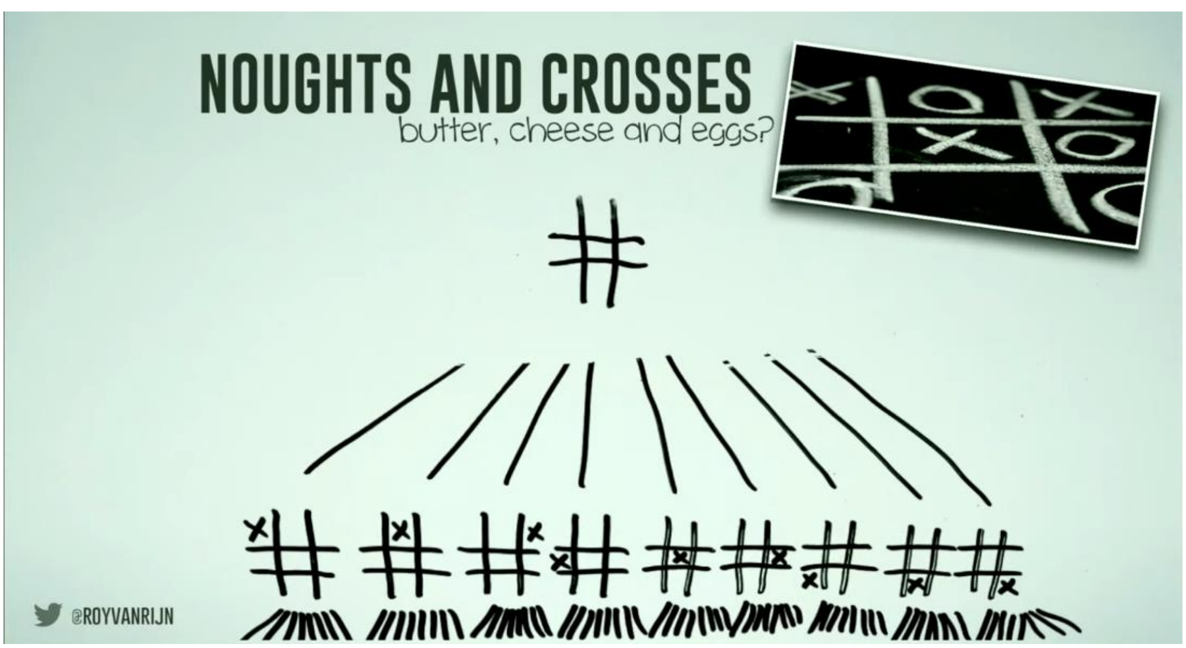
We generally start and the bottom and evaluate all the nodes to pick the node with the highest value

```java
int minimax(Node node, boolean maximizingScore) {
    if(node.isEndNode()) {
        return node.evaluate();
    }

    int bestScore = maximizingScore ? Integer.MIN_VALUE : Integer.MAX_VALUE;
    for(Node child: node.getChildren()) {
        int score = minimax(child, !maximizingScore);
        if(maximizingScore) {
            bestScore = Math.max(score, bestScore);
        } else {
            bestScore = Math.min(score, bestScore);
        }
    }
    return bestScore;
}
```



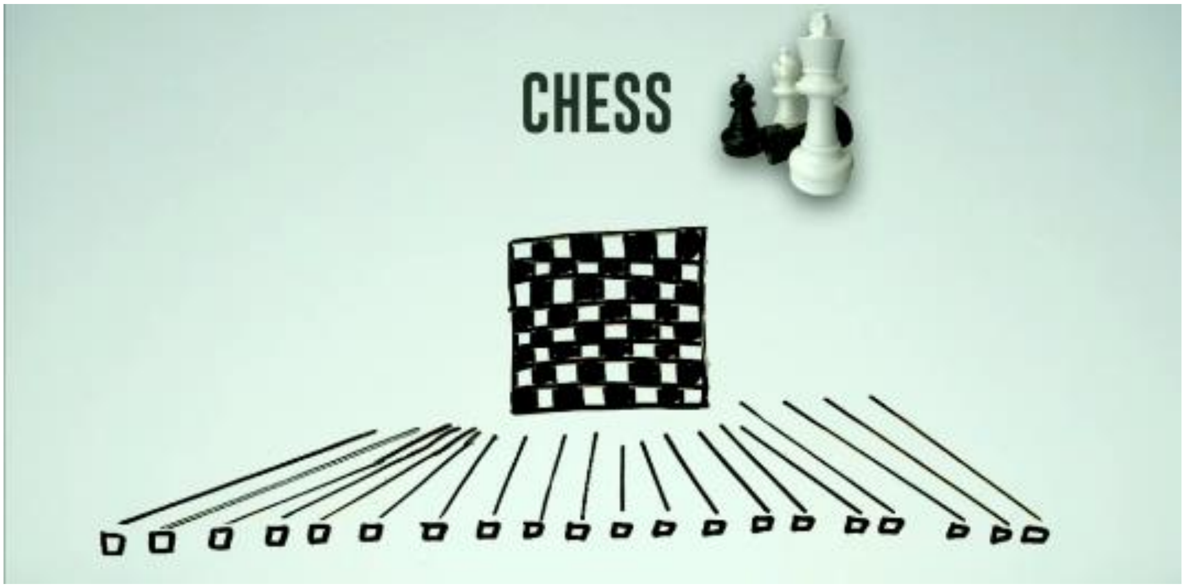




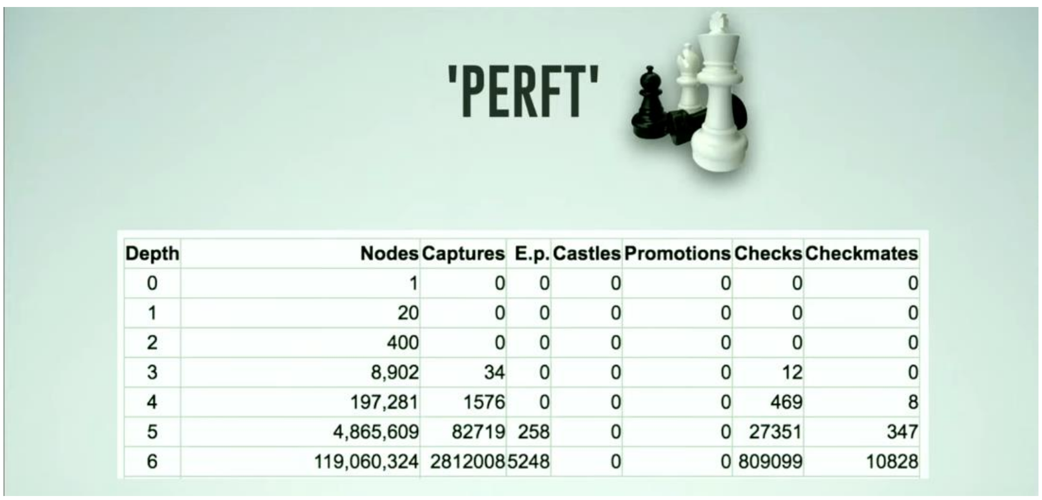
You can get very large possibilities for the choice here, but you can eliminate some duplicates if you take the 1st choice and rotate it to get the 3rd, 7th and 9th choices. You can also mirror the 2nd choice to get the 8th choice.

We can evaluate the nodes using the above approach

This is a perfect information game where we can calculate the entire game easily, we always end up in one of the terminal positions in at most 9 moves.



Chess is no longer trivial and takes much longer to play



Depth	Nodes	Captures	E.p.	Castles	Promotions	Checks	Checkmates
0	1	0	0	0	0	0	0
1	20	0	0	0	0	0	0
2	400	0	0	0	0	0	0
3	8,902	34	0	0	0	12	0
4	197,281	1576	0	0	0	469	8
5	4,865,609	82719	258	0	0	27351	347
6	119,060,324	2812008	5248	0	0	809099	10828

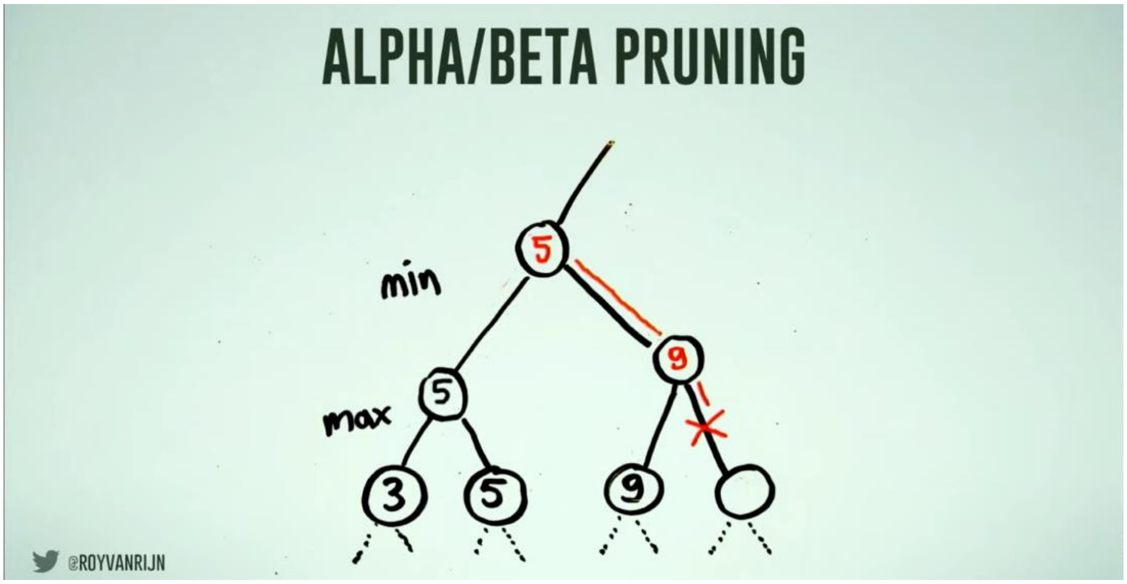
This is the Perft table for the chess start board. A game engine’s suitable is in how fast it can generate all the possible moves at each depth and choose the best next move.

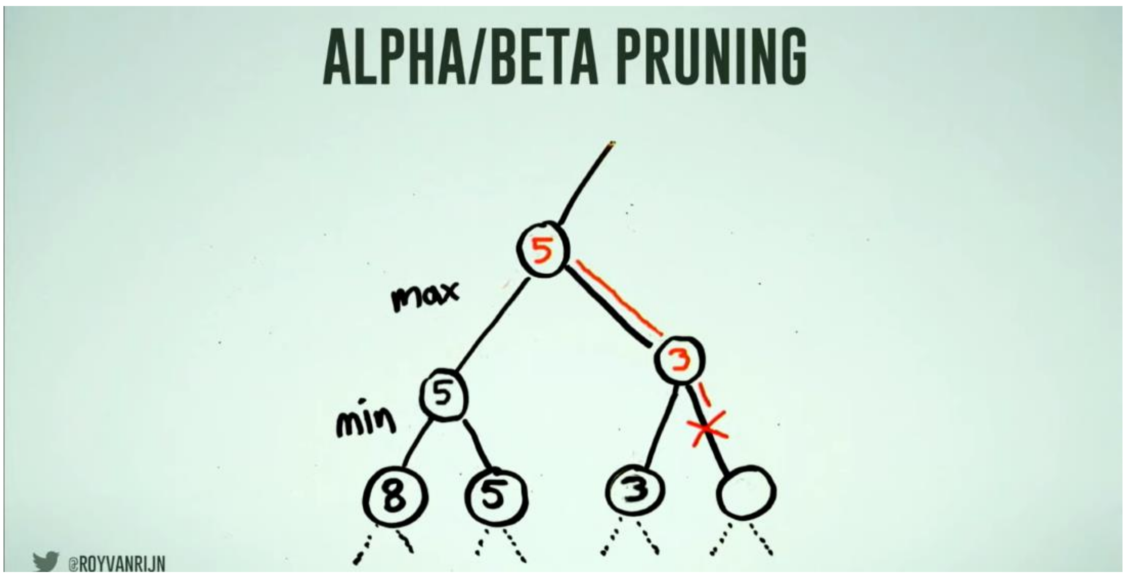
This allows us to write an evaluation function that we can use to evaluate the board positions up to a certain depth, assign it a value,


HORIZON PROBLEM


PRUNING
• WE NEED TO CUT BACK THE TREE
• FORWARD PRUNING: RISKY
• BACKWARD PRUNING: SAFE
@ROYVANRIJN


FORWARD PRUNING
• IF A MOVE IS TOO BAD: STOP EVALUATING
• IF A MOVE IS TOO GOOD: STOP EVALUATING






```java
int minimax(Node node, boolean maximizingScore) {
    if(node.isEndNode()) {
        return node.evaluate();
    }

    int bestScore = maximizingScore ? Integer.MIN_VALUE : Integer.MAX_VALUE;
    for(Node child: node.getChildren()) {
        int score = minimax(child, !maximizingScore);
        if(maximizingScore) {
            bestScore = Math.max(score, bestScore);
        } else {
            bestScore = Math.min(score, bestScore);
        }
    }
    return bestScore;
}
```

```java
int alphaBeta(Node node,                          boolean maximizingScore) {
    if(node.isEndNode()) {
        return node.evaluate();
    }

    int bestScore = maximizingScore ? Integer.MIN_VALUE : Integer.MAX_VALUE;
    for(Node child: node.getChildren()) {
        int score = alphaBeta(child,                 !maximizingScore);
        if(maximizingScore) {
            bestScore = Math.max(bestScore, score);

        } else {
            bestScore = Math.min(bestScore, score);

        }


    }
    return bestScore;
}
```

@ROYVANRIJN

```java
int alphaBeta(Node node, int alpha, int beta, boolean maximizingScore) {
    if(node.isEndNode()) {
        return node.evaluate();
    }

    int bestScore = maximizingScore ? Integer.MIN_VALUE : Integer.MAX_VALUE;
    for(Node child: node.getChildren()) {
        int score = alphaBeta(child, alpha, beta, !maximizingScore);
        if(maximizingScore) {
            bestScore = Math.max(bestScore, score);
            alpha = Math.max(alpha, bestScore);
        } else {
            bestScore = Math.min(bestScore, score);
            beta = Math.min(beta, bestScore);
        }
        if(beta <= alpha)
            break; /* stop evaluating */
    }
    return bestScore;
}
```

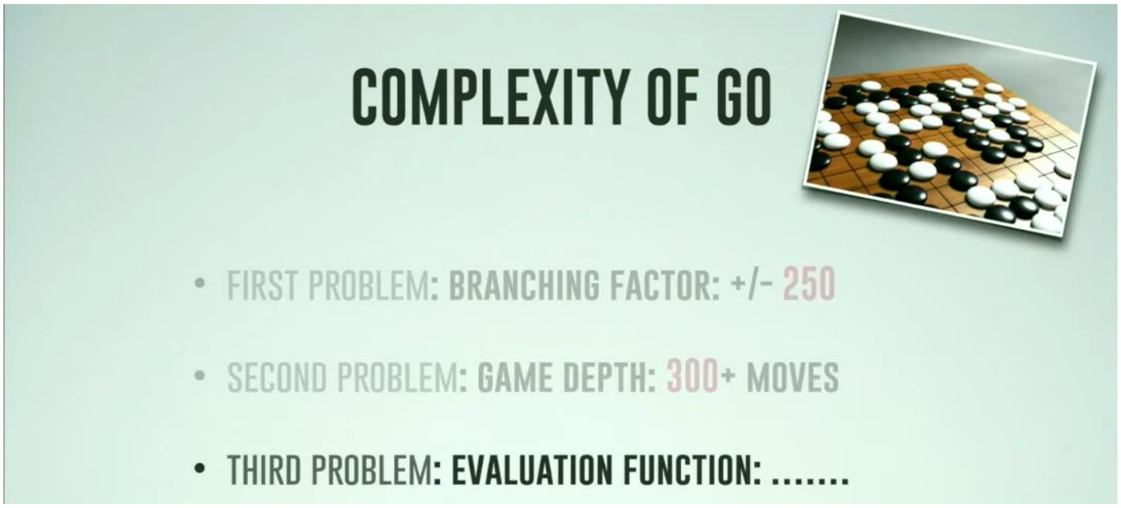
@ROYVANRIJN

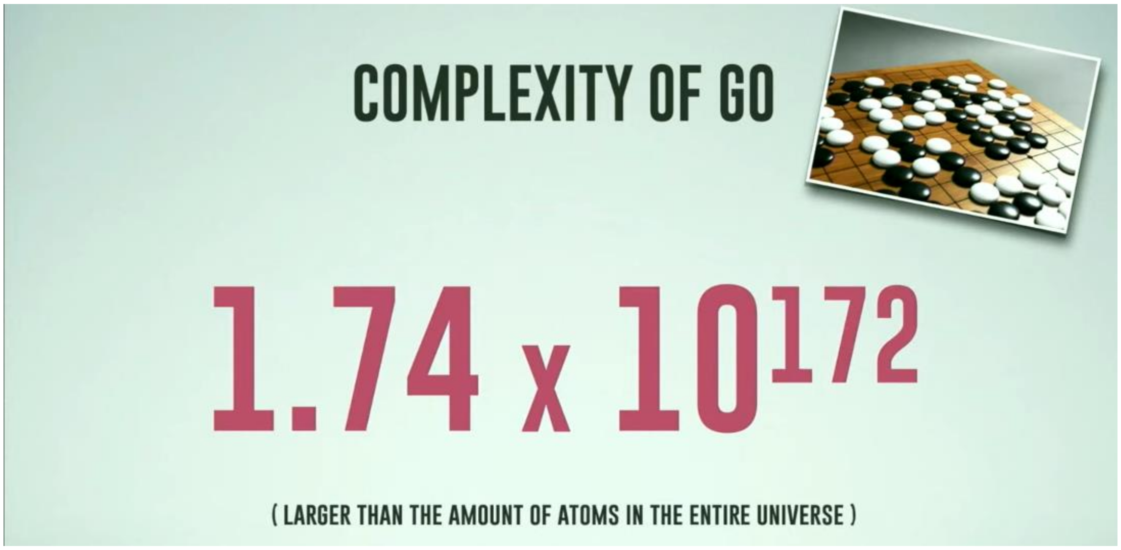
Using pruning and an evaluation function to pick a move, we can now have a much deeper search and a much better chess engine


CHESS
AVERAGE BRANCHING FACTOR: 35
AVERAGE GAME DEPTH: 40-50 MOVES
EVALUATION FUNCTION: RELATIVELY EASY
ADVANCED CHESS A.I. CAN LOOK 20+ MOVES AHEAD


THE GAME OF GO
@ROYVANRIJN


ABOUT THE GAME
BOARD: 19X19
BLACK AND WHITE STONES
SURROUND AND CAPTURE AREAS





How can we also use random search to make a better Go playing engine?

MONTE CARLO TREE SEARCH

- PICK A NODE
- PLAY (SEMI-) RANDOM MOVES TO THE END (AS OFTEN AS POSSIBLE)
- THIS GIVES A STRONG INDICATION OF THE STRENGTH

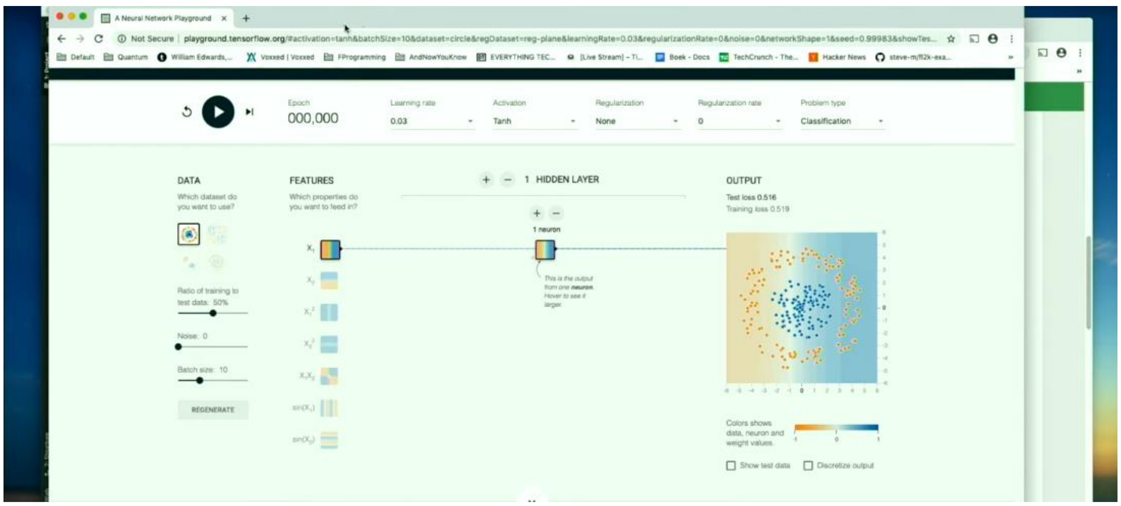
EXPERTS IN 2015:

"IT WILL PROBABLY TAKE 10 TO 15 YEARS BEFORE A COMPUTER CAN BEAT A PROFESSIONAL GO PLAYER"

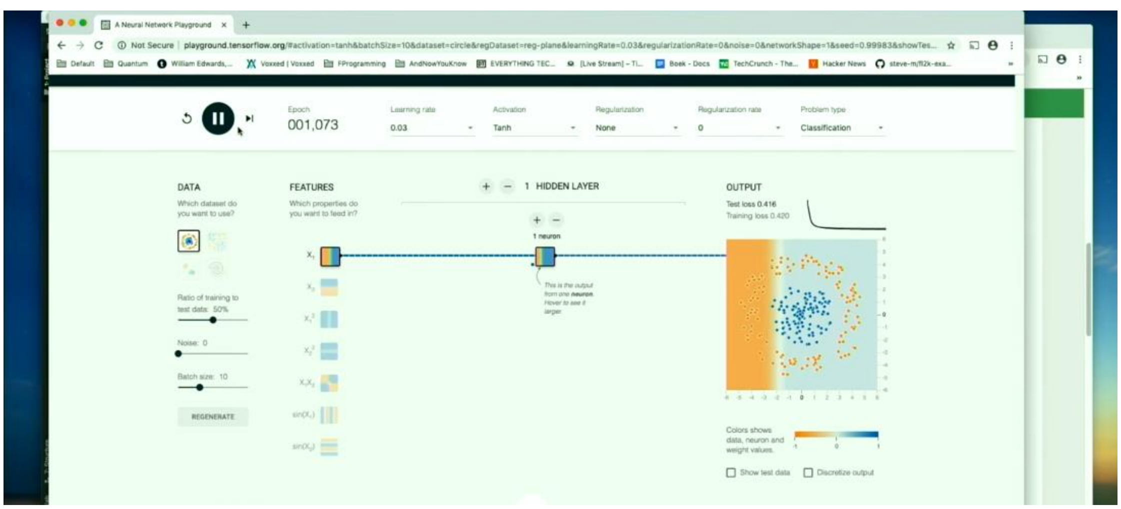
NEURAL NETWORK

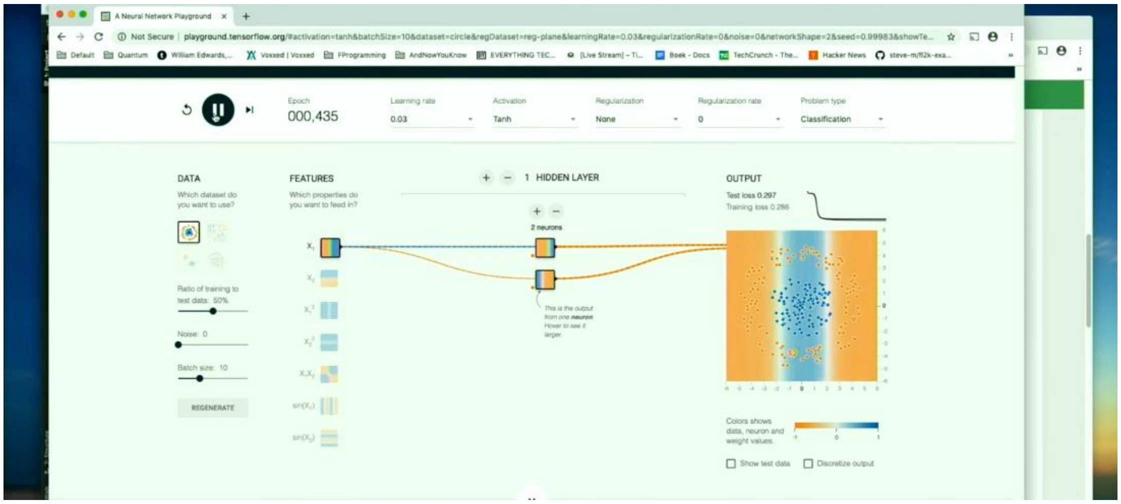
A NEURAL NETWORK IS A COMPUTER MODEL DESIGNED TO SIMULATE THE BEHAVIOUR OF **BIOLOGICAL NEURAL NETWORKS**

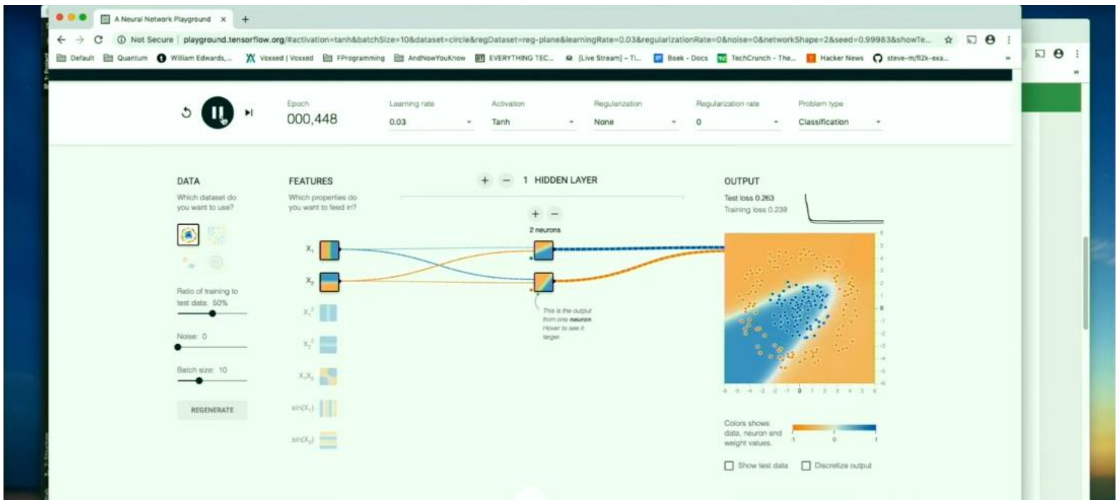
@ROYVANRIJN

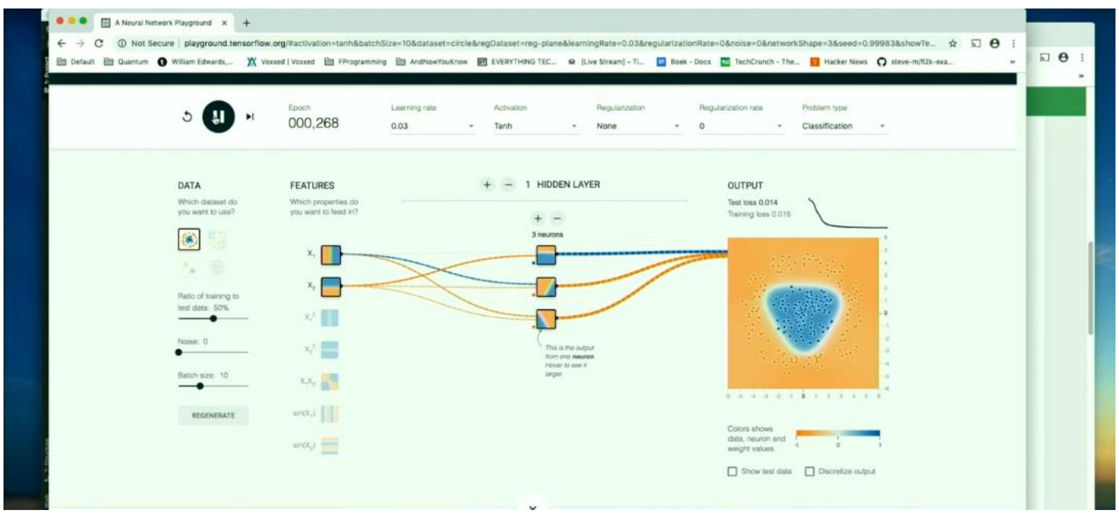


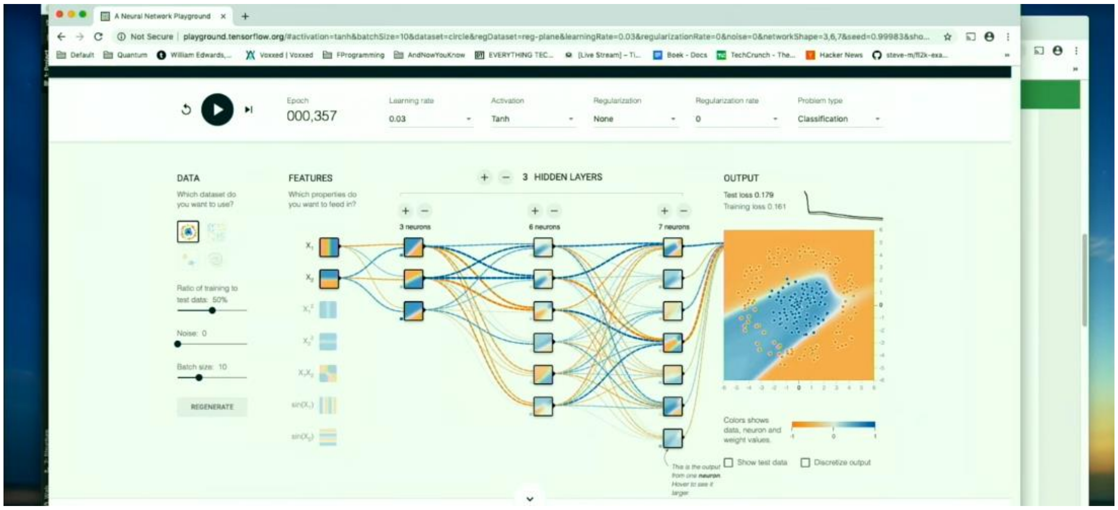
You can build a NN graphically and solve a sorting problem




A Neural Network Playground
Epoch
000,435
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
FEATURES
1 HIDDEN LAYER
2 neurons
OUTPUT
Test loss 0.297
REGENERATE

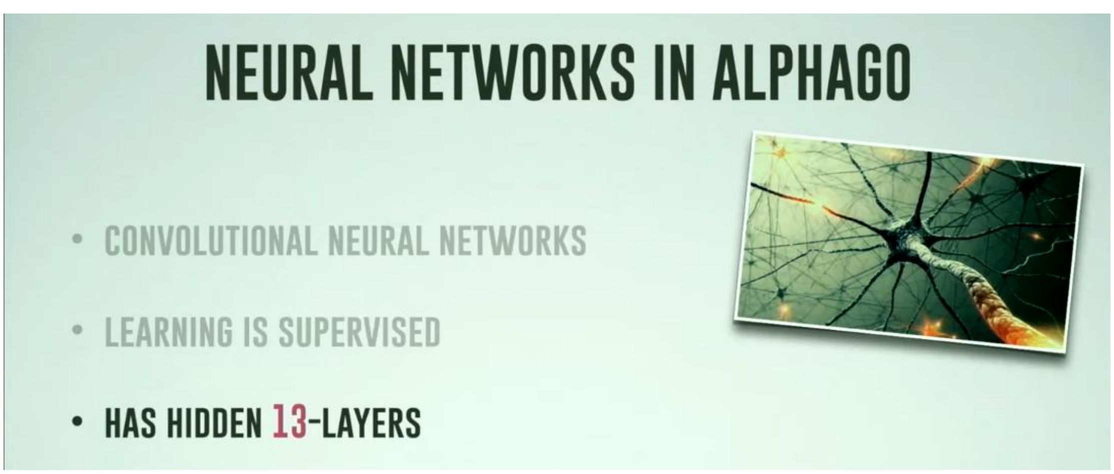

A Neural Network Playground
Epoch
000,448
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
FEATURES
1 HIDDEN LAYER
2 neurons
OUTPUT
Test loss 0.263
Training loss 0.239
REGENERATE


A Neural Network Playground
Epoch
000,268
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
FEATURES
1 HIDDEN LAYER
3 neurons
OUTPUT
Test loss 0.014
Training loss 0.015
REGENERATE


A Neural Network Playground
playground.tensorflow.org
Epoch
000,357
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
FEATURES
Which properties do you want to feed in?
3 HIDDEN LAYERS
3 neurons
6 neurons
7 neurons
OUTPUT
Test loss 0.179
Training loss 0.161
REGENERATE
Show test data
Discretize output


MORE INFORMATION
• TensorFlow (https://www.tensorflow.org)
• Deeplearning4j (https://deeplearning4j.org/)
• Caffe, Torch, Theano, etc


TELL ME HOW!
AlphaGo
@ROYVANRIJN


NEURAL NETWORKS IN ALPHAGO
• Convolutional neural networks
• Learning is supervised
• Has hidden 13-layers

#1 SUPERVISED LEARNING POLICY NETWORK

- 30 MILLION AMATEUR MATCHES
- GOAL: PREDICT THE NEXT MOVE
- RESULT: 57% CORRECT

#2 REINFORCED LEARNING POLICY NETWORK

- COPY OF SUPERVISED NETWORK
- NEW GOAL: PREDICT THE *BEST* MOVE
- NETWORK PLAYED ITSELF 1.2 MILLION TIMES (TOOK ONE DAY)
- PLAYS PACHI AND WINS: 85% OF THE TIME (WITHOUT SEARCH!)

#3 FAST ROLLOUT POLICY NETWORK

- THE REINFORCED NETWORK IS SLOW: 3MS
- TOO SLOW FOR MONTE CARLO SEARCH
- THIS IS SMALLER, BUT FASTER: 2µS 1500X

#4 VALUE NETWORK

- TRAINED USING THE SAME 30 MILLION GAMES
- PREDICTS THE WINNER BASED ON CURRENT BOARD
- INITIALLY HAD ERROR OF 0.37 (0.5 IS RANDOM)
- AFTER SELF-PLAY ERROR CAME DOWN TO ~0.23

#4 VALUE NETWORK

- TESTING THE VALUE NETWORK
- FOR A GIVEN BOARD, GENERATE ALL MOVES
- FOR ALL MOVES, EVALUATE AND PICK THE BEST NEXT MOVE
- BEATS THE STRONGEST KNOWN A.I. STILL WITHOUT TREE-SEARCH (!!)

COMBINING ALL THE PIECES

- USE POLICY NETWORK TO LOOK AT THE CURRENT BEST MOVES
- FOR THOSE MOVES, USE THE VALUE NETWORK TO DOUBLE CHECK
- USE FAST ROLLOUT NETWORK FOR MONTE CARLO TREE SEARCH

AlphaGo

THE CHALLENGE

- LEE SEDOL: THE BEST GO PLAYER OF THIS DECADE
- BEST OF 5 GAMES WINS
- WINNER GETS $1,000,000.-

@ROYVANRIJN

THE CHALLENGER

- DISTRIBUTED ALPHAGO:
  - 1202 CPU'S

AlphaGo

@ROYVANRIJN

GAME 1, MOVE 102

@ROYVANRIJN

GAME 1, MOVE 102

@ROYVANRIJN

GAME 2, MOVE 37

@ROYVANRIJN

GAME 2, MOVE 37

@ROYVANRIJN

European champion Fan Hui:

"It's not a human move. I've never seen a human play this move, so beautiful."

Game 4, move 78

Gu Li (Lee's archrival):

"This move was made with the hand of God."

Results

AlphaGo 4 - Lee Sedol 1

@ROYVANRIJN

Nobody taught AlphaGo what a good or bad move is

Nobody programmed an evaluation function for AlphaGo

AlphaGo isn't an expert system

ALPHAGO LEARNED BY WATCHING OTHERS AND SELF-PLAY

USING GENERAL MACHINE LEARNING TECHNIQUES
TO FIGURE OUT FOR ITSELF HOW TO WIN AT GO...

AS A RESPONSE TO THE SUCCESS OF ALPHAGO, SOUTH KOREA ANNOUNCED ON 17 MARCH 2016 THAT IT WOULD INVEST $863 MILLION IN ARTIFICIAL-INTELLIGENCE RESEARCH OVER THE NEXT FIVE YEARS.

- ALPHAGO ZERO VERSUS ALPHAGO: 100 - 0
- SUPERHUMAN ABILITIES FOR: CHESS, SHOGI
- PROTEIN FOLDING: ALPHAFOLD
- STARCRAFT: ALPHASTAR
- ULTIMATE GOAL: DEEPMIND HEALTH...

@ROYVANRIJN