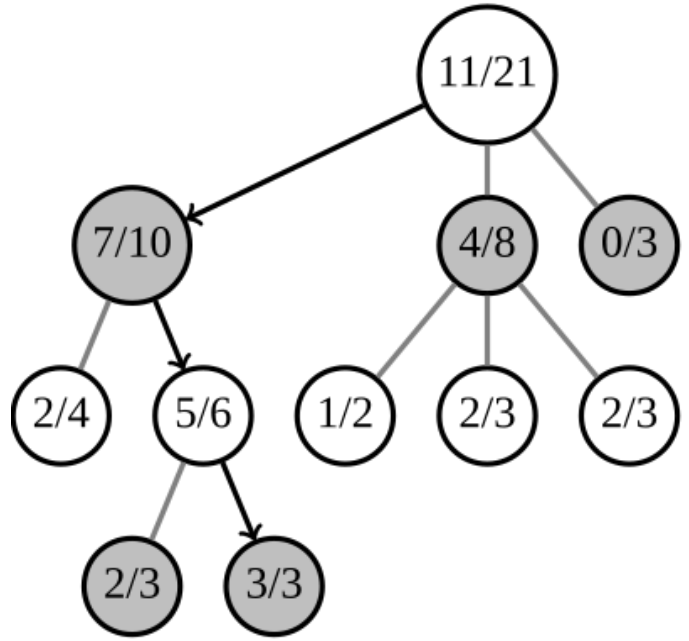
Wykład 10. O rywalizacji człowieka z komputerem

Piotr Morawiecki

19 maja 2025

Podsumowanie poprzedniego wykładu

- Poznaliśmy algorytm *Monte Carlo Tree Search* (MCTS), który pozwala znajdować strategie w grach, w których przeszukanie całego drzewa gry jest niemożliwe.
- Polega na iteracyjnym przeszukiwaniu najbardziej obiecujących gałęzi drzewa gry, a następnie na drodze symulacji ocenianiu szans na wygraną przy wyborze danej strategii.
- Algorytmu MCTS można wzmocnić wykorzystując metody heurystyczne, np. ewaluacją pośrednich stanów gry.



Dwa słynne pojedynki człowieka z komputerem

DEEP-BLUE VS GARY KASPAROV (1997)

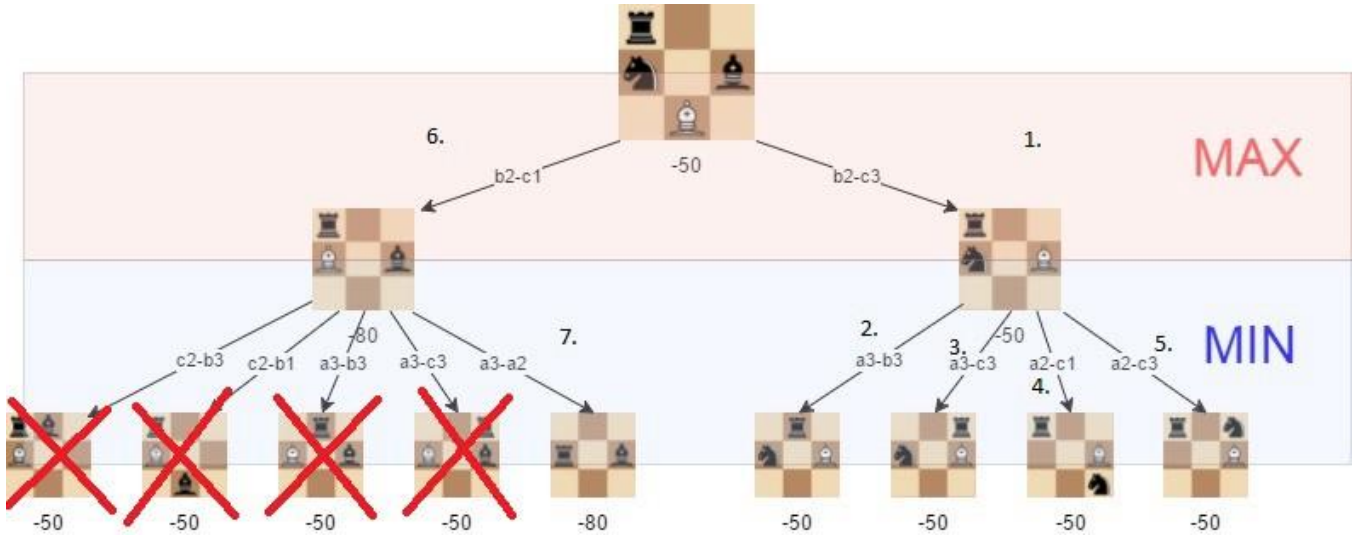
ALPHA GO VS LEE SEDOL (2016)

Algorytm DeepBlue (1/2)

- DeepBlue został zaprogramowany w oparciu o wiedzę ekspercką.
- Wykorzystywał on bazę danych zawierającą zapis 700,000 rozgrywek arcymistrzów szachowych.
- W oparciu o nią skonstruował zbiór danych (tzw. *Extended Book*), który zawierał miliony możliwych stanów gry z pierwszych 30 ruchów, pozwalając algorytmowi wybierać pierwsze ruchy.
- Ponadto zawierał obszerną bazę zakończeń gry (tzw. *Endgame Database*), zawierającą wynik gry po tym jak na planszy zostanie 6 lub mniej figur szachowych.

Algorytm DeepBlue (2/2)

- Podczas rozgrywki wykorzystywał algorytm minimax dla kilku następnych poziomów drzewa gry. Stany gry były oceniane z wykorzystaniem heurystycznych wzorów uwzględniających:
  1. wartości figur każdego gracza,
  2. położenie figur (bezpieczeństwo króla, ustawienie pionków, aktywność figur, itd.), oraz
  3. informacje z *endgame database*.
- Superkomputer analizował ok. 200 milionów stanów gry na sekundę.

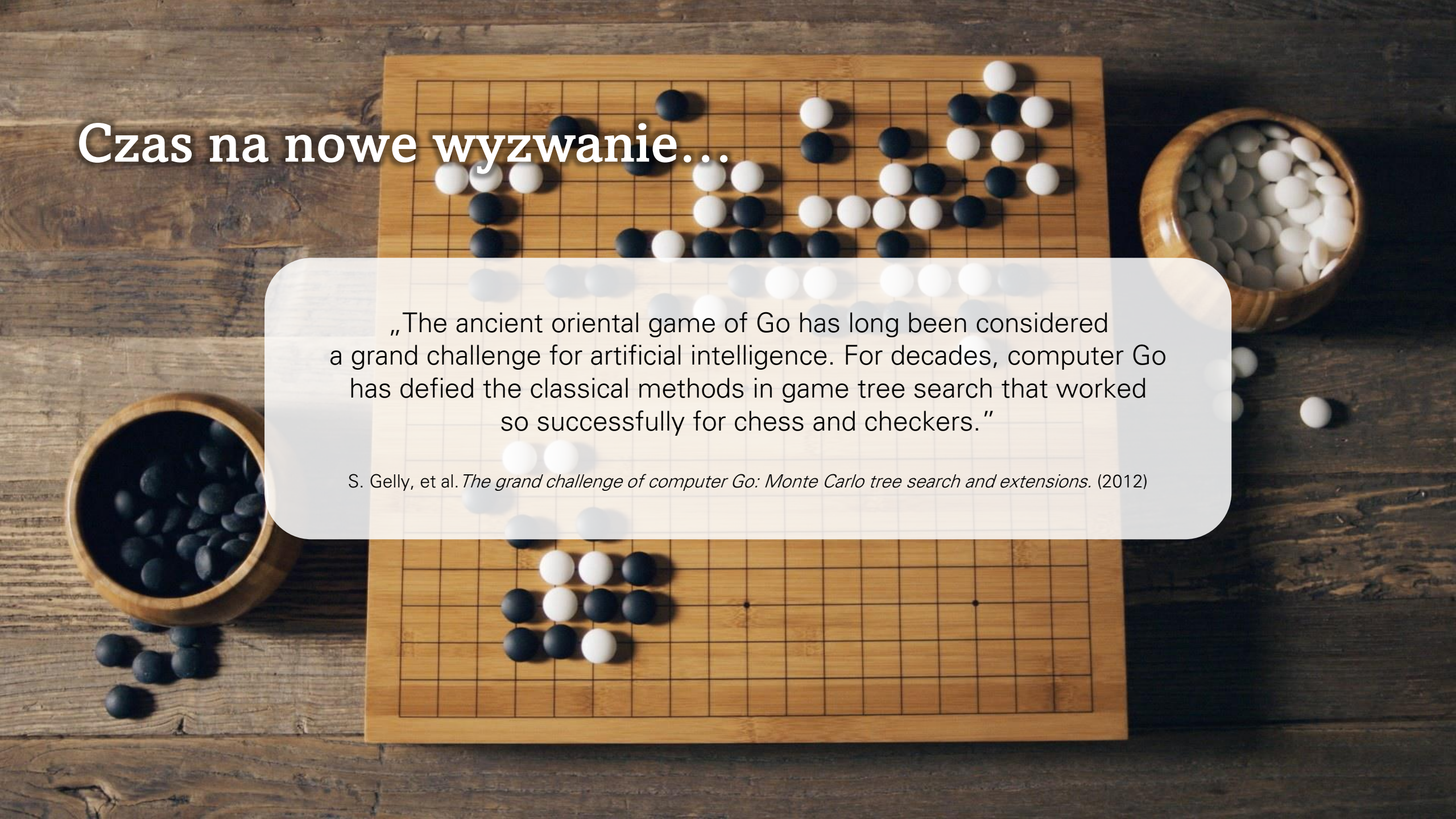


Źródło grafiki: https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/

DeepBlue vs Gary Kasparov

- DeepBlue zmierzył się w 1996 roku z mistrzem świata w szachach, Garrym Kasparovem. DeepBlue wygrał pierwszą rozgrywkę, ale później zremisował dwóch i przegrał w trzech.
- Rewanż odbył się w 1997 roku, podczas którego Deep Blue wygrał dwie rozgrywki, Garry Kasparov jedną, a pozostałe trzy zakończyły się remisem.

Czas na nowe wyzwanie...



„The ancient oriental game of Go has long been considered a grand challenge for artificial intelligence. For decades, computer Go has defied the classical methods in game tree search that worked so successfully for chess and checkers."

S. Gelly, et al. *The grand challenge of computer Go: Monte Carlo tree search and extensions.* (2012)

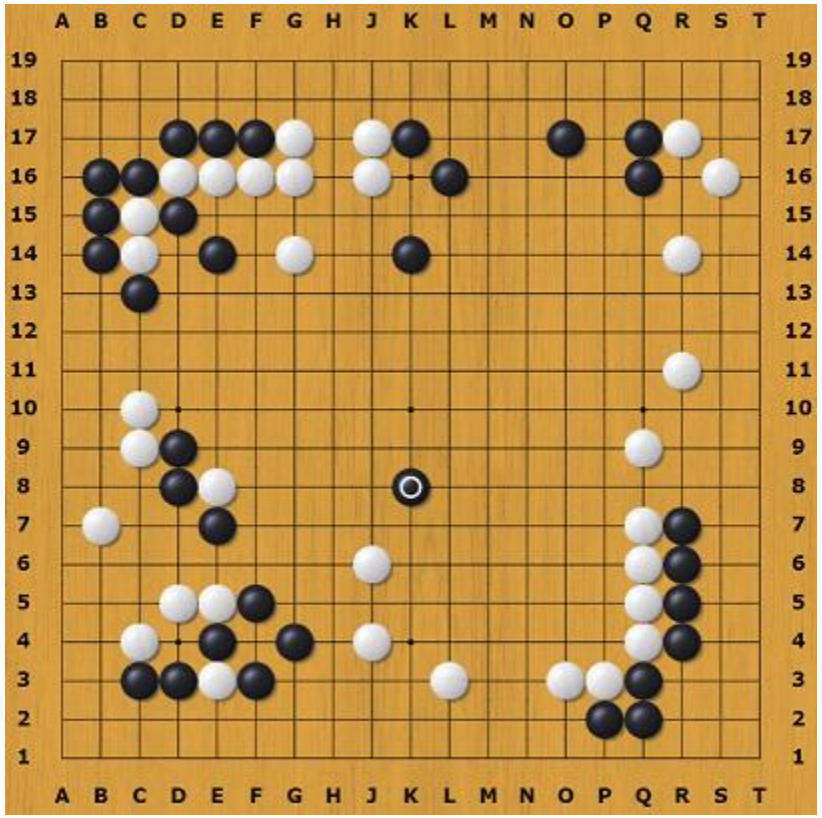
Zasady Go

Na czym polega trudność Go?

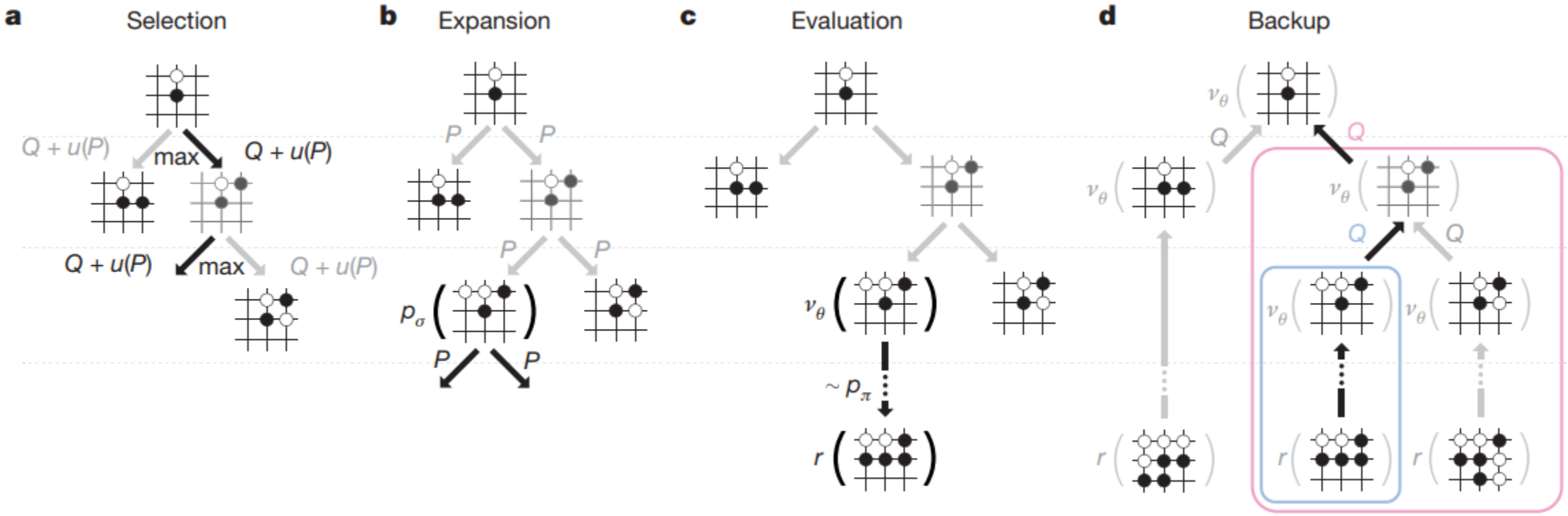
1. Ogromna liczba możliwych ruchów

	Szachy	Go
Liczba kombinacji dwóch pierwszych ruchów	$20^2 = 400$	$361 \cdot 360 \approx 130{,}000$
Typowa liczba tur w trakcie jednej partii	$40 - 60$	≈ 200
Szacowana liczba możliwych stanów gry	10^{47}	10^{170}

2. Trudność dokonaniem ewaluacji stanu gry (każdy ruch może mieć niespodziewane konsekwencje na dużo późniejszym etapie gry)



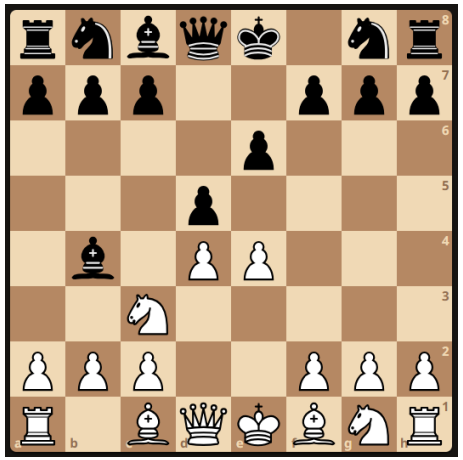
Algorytm AlphaGo: MCTS

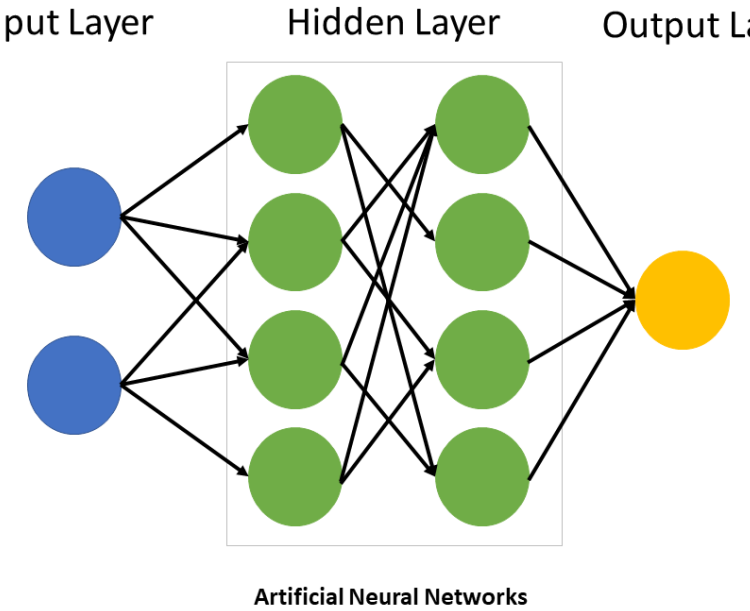


Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Czym są sieci neuronowe?

Wejście: stan gry



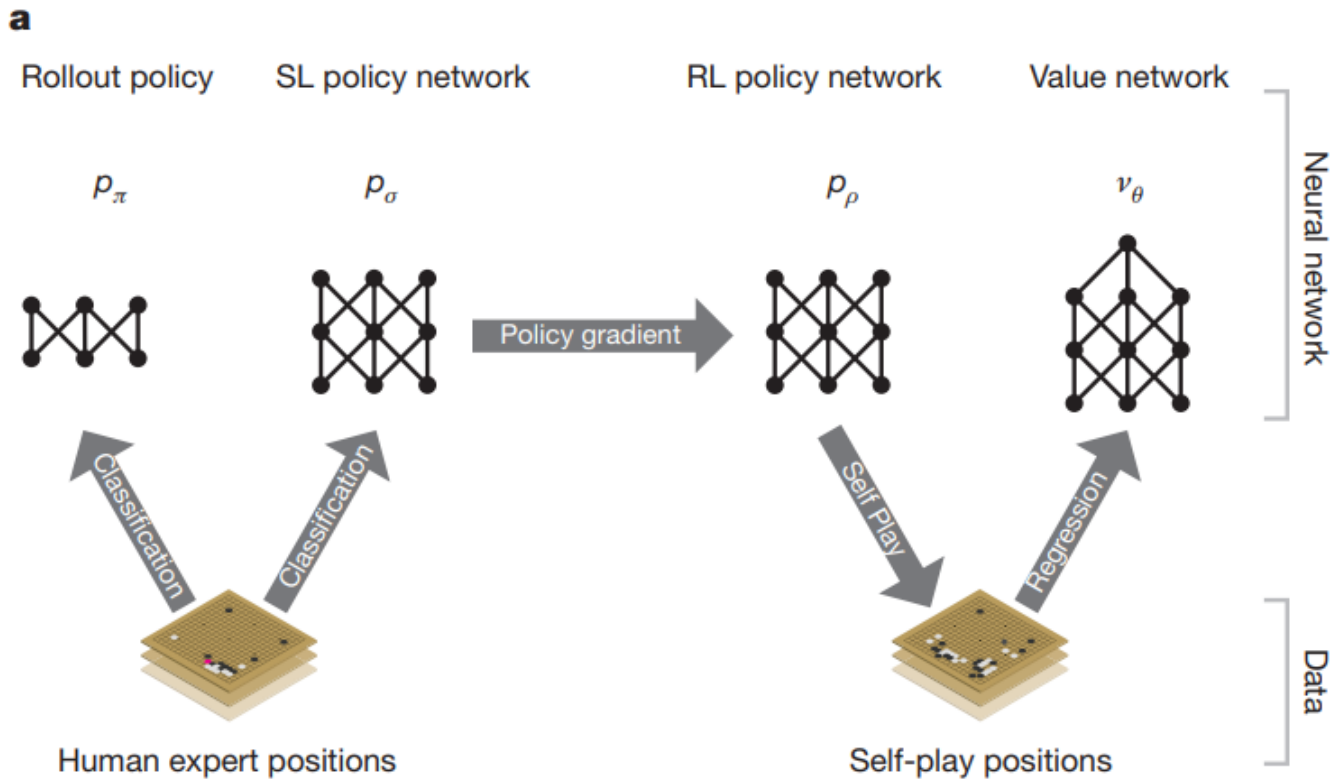


Wyjście:

prawdopodobieństwo wygranej danego gracza

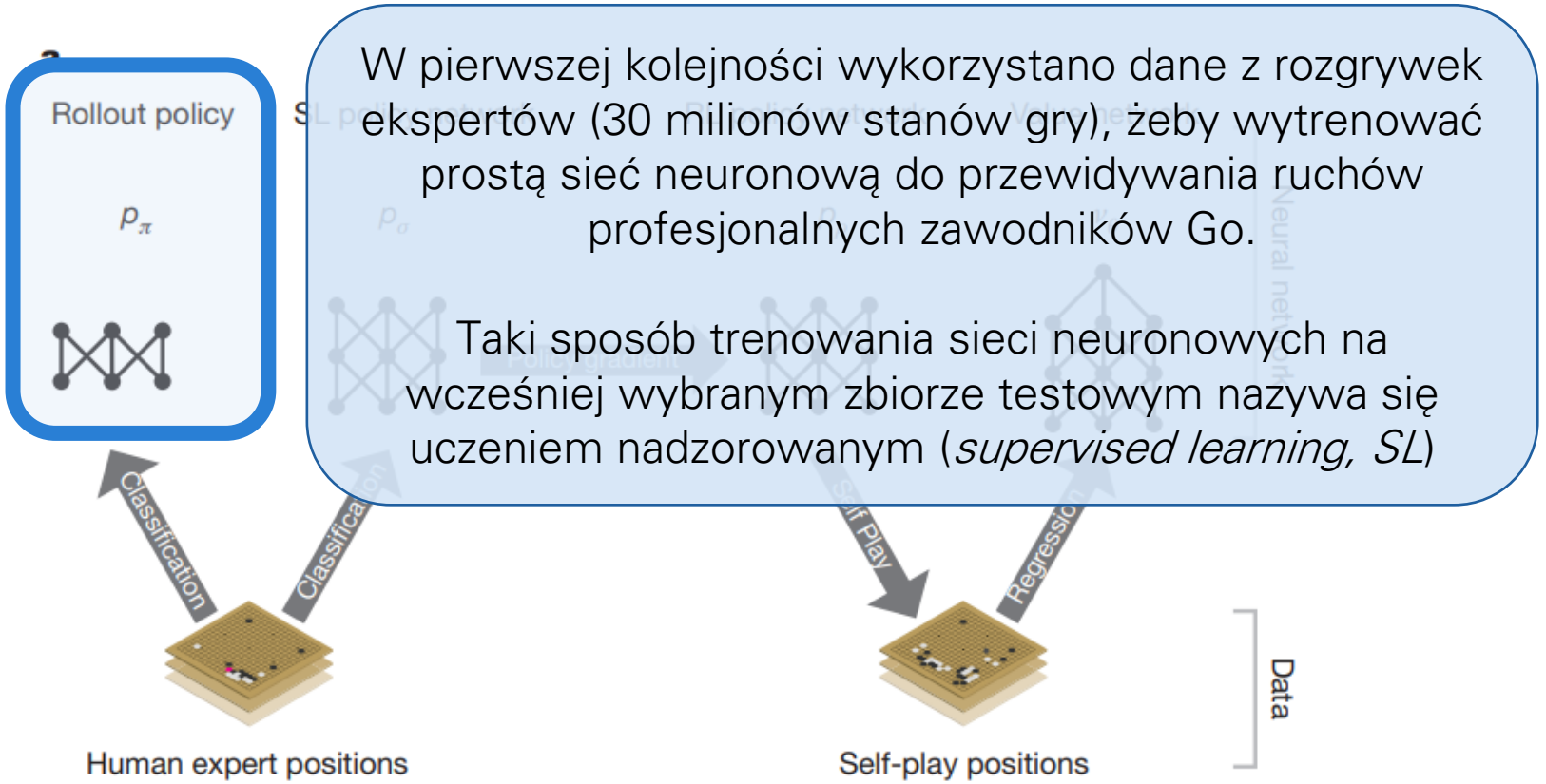
Interaktywna demonstracja: http://playground.tensorflow.org/

Algorytm AlphaGo: trenowanie sieci neuronowych



Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: trenowanie sieci neuronowych

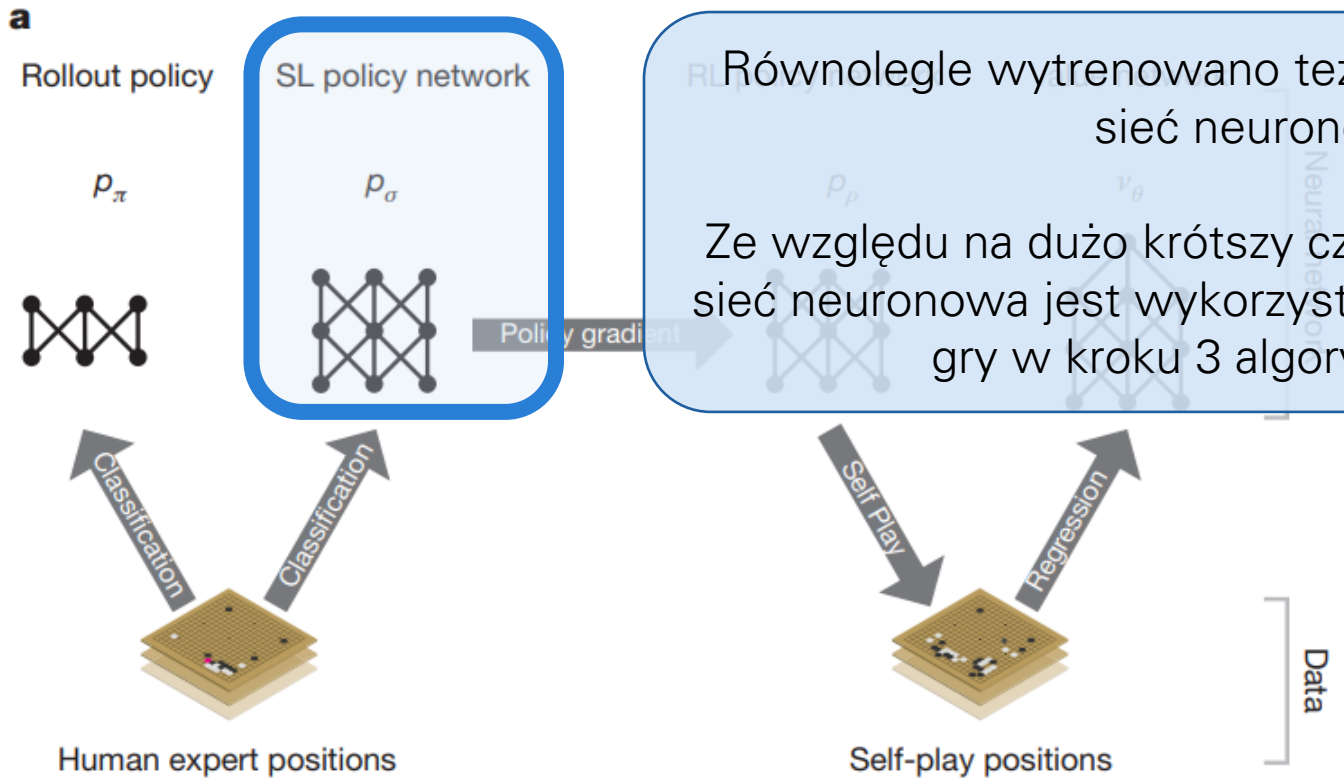


W pierwszej kolejności wykorzystano dane z rozgrywek ekspertów (30 milionów stanów gry), żeby wytrenować prostą sieć neuronową do przewidywania ruchów profesjonalnych zawodników Go.

Taki sposób trenowania sieci neuronowych na wcześniej wybranym zbiorze testowym nazywa się uczeniem nadzorowanym (*supervised learning, SL*)

Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: trenowanie sieci neuronowych

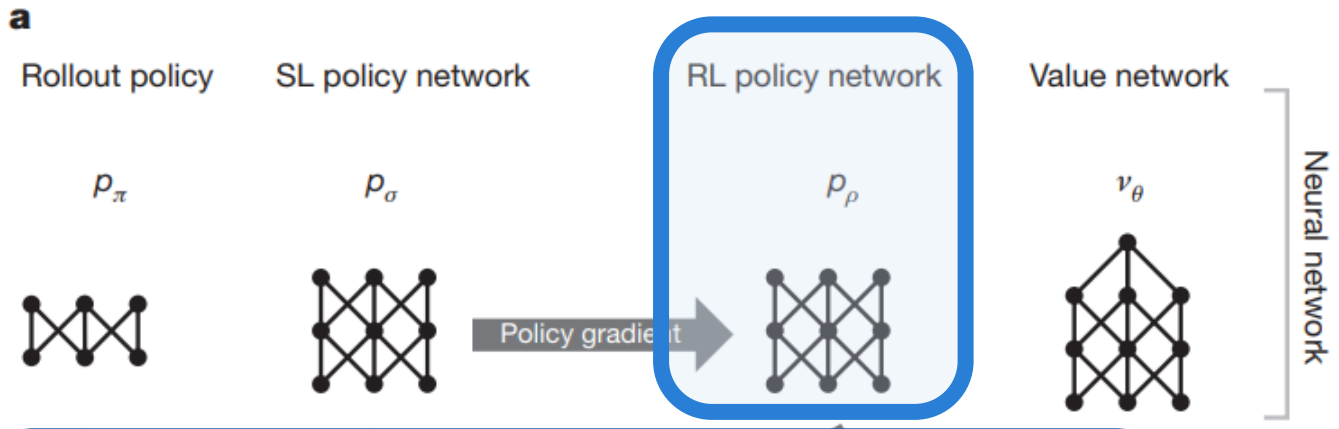


Równolegle wytrenowano też dużo bardziej złożoną sieć neuronową.

Ze względu na dużo krótszy czas obliczeń to prostsza sieć neuronowa jest wykorzystywana do symulowania gry w kroku 3 algorytmu MCTS.

Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: trenowanie sieci neuronowych

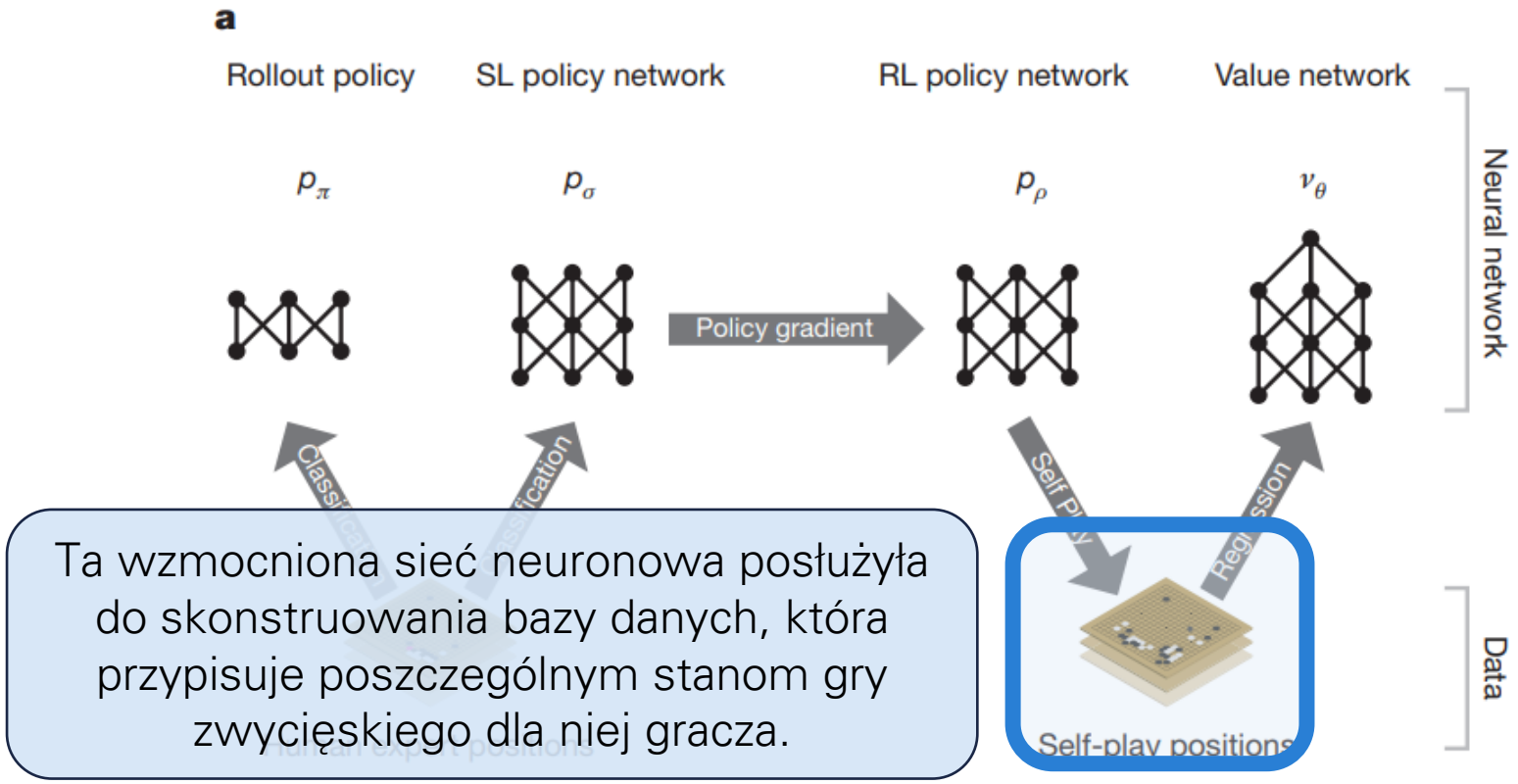


Bardziej złożona sieć została dalej udoskonalona z wykorzystaniem tzw. uczenia wzmocnionego (*reinforcement learning, RL*).

Algorytm rozgrywał partie Go sam ze sobą, za każdym razem modyfikując wagi poszczególnych neuronów, tak żeby zmaksymalizować szansę na wygraną.

Źródło: D. Silver, A. Huang, [illegible] *tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: trenowanie sieci neuronowych



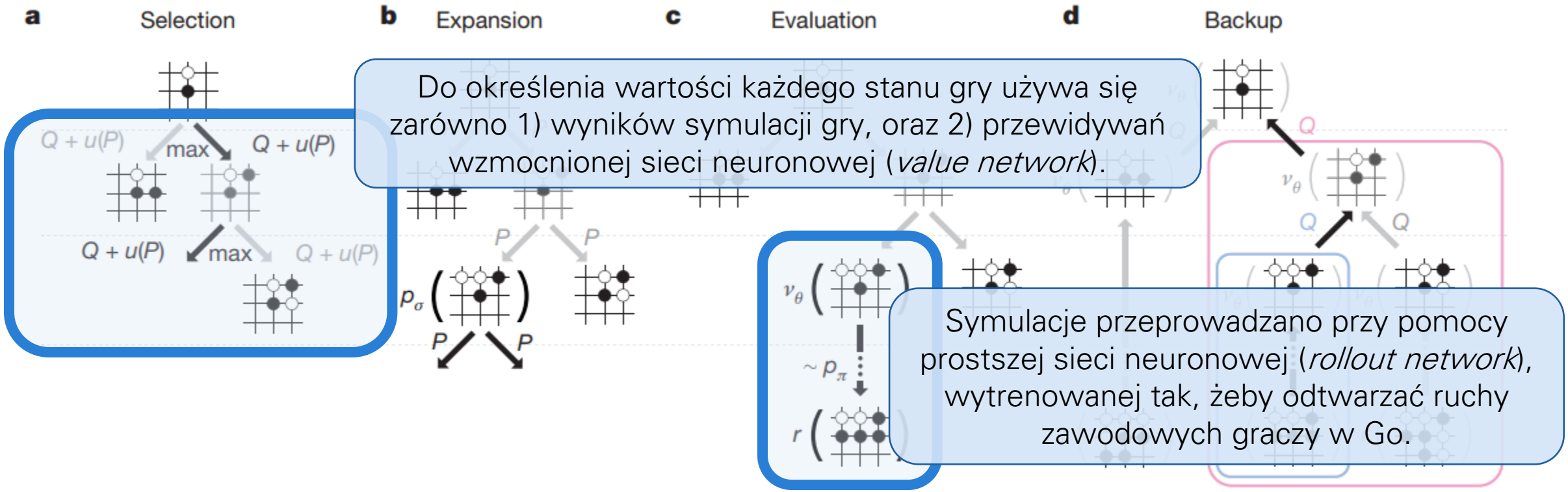
Ta wzmocniona sieć neuronowa posłużyła do skonstruowania bazy danych, która przypisuje poszczególnym stanom gry zwycięskiego dla niej gracza.

Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: trenowanie sieci neuronowych

Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

Algorytm AlphaGo: MCTS



Do określenia wartości każdego stanu gry używa się zarówno 1) wyników symulacji gry, oraz 2) przewidywań wzmocnionej sieci neuronowej (*value network*).

Symulacje przeprowadzano przy pomocy prostszej sieci neuronowej (*rollout network*), wytrenowanej tak, żeby odtwarzać ruchy zawodowych graczy w Go.

Źródło: D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search*. Nature 529, 484–489 (2016).

AlphaGo vs mistrzowie Go

- W **październiku 2015** AlphaGo pokonał mistrza w Go, Fan Hui, wygrywając wszystkie 5 rund.
- W **marcu 2016** AlphaGo stoczył pojedynek z jednym z czołowych arcymistrzów w Go, Lee Sedol, wygrywając 4 z 5 rund.

Algorytm AlphaZero

- Do **grudnia 2017** powstała nowa wersja algorytmu, *AlphaGo Zero*, który został wytrenowany bez wykorzystania wiedzy nt. strategii stosowanych przez ekspertów.
- Do **grudnia 2018** na jego bazie został opracowany algorytm *AlphaZero*, który był w stanie grać na poziomie mistrzowskim nie tylko w Go, ale też szachy i shogi (japońską odmianę szachów).

ARTICLE

doi:10.1038/nature24270

Mastering the game of Go without human knowledge

David Silver[1]*, Julian Schrittwieser[1]*, Karen Simonyan[1]*, Ioannis Antonoglou[1], Aja Huang[1], Arthur Guez[1], Thomas Hubert[1], Lucas Baker[1], Matthew Lai[1], Adrian Bolton[1], Yutian Chen[1], Timothy Lillicrap[1], Fan Hui[1], Laurent Sifre[1], George van den Driessche[1], Thore Graepel[1] & Demis Hassabis[1]

A long-standing goal of artificial intelligence is an algorithm that learns, *tabula rasa*, superhuman proficiency in challenging domains. Recently, AlphaGo became the first program to defeat a world champion in the game of Go. The tree search in AlphaGo evaluated positions and selected moves using deep neural networks. These neural networks were trained by supervised learning from human expert moves, and by reinforcement learning from self-play. To produce an algorithm based solely on reinforcement learning, without human data, guidance or domain knowledge beyond game rules. AlphaGo becomes its own teacher: a neural network is trained to predict AlphaGo's own move selection and the winner of AlphaGo's games. This neural network improves the strength of tree search, resulting in higher quality move selection and stronger self-play in the next iteration. Starting tabula rasa, our new program AlphaGo Zero achieved superhuman performance, winning 100-0 against the previously published, champion-defeating AlphaGo.

RESEARCH

COMPUTER SCIENCE

A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play

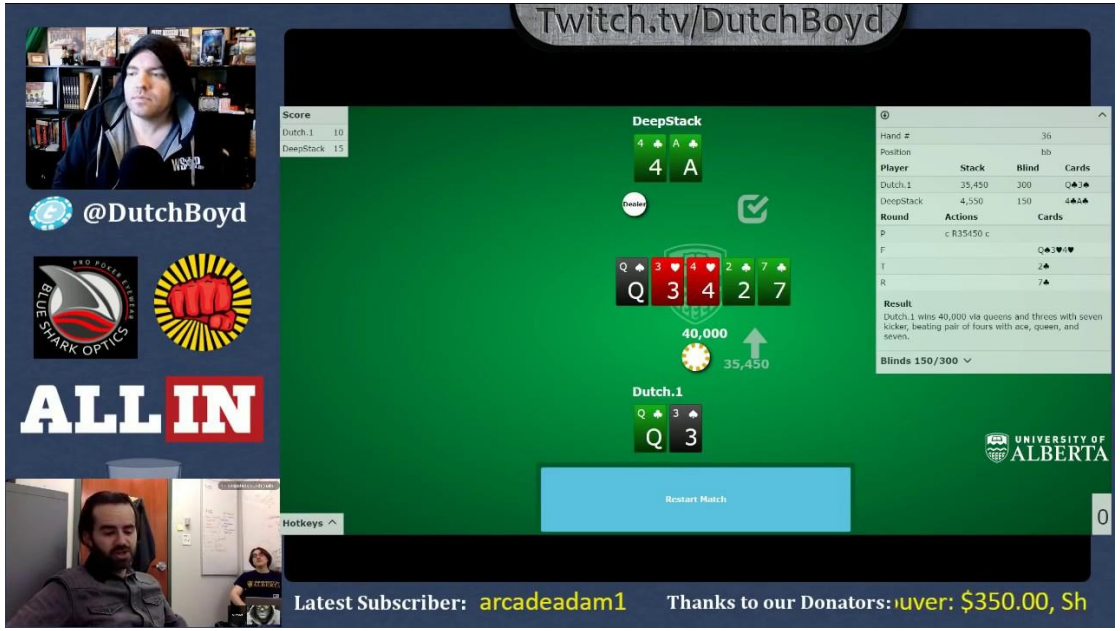
David Silver[1,2*†], Thomas Hubert[1*], Julian Schrittwieser[1*], Ioannis Antonoglou[1], Matthew Lai[1], Arthur Guez[1], Marc Lanctot[1], Laurent Sifre[1], Dharshan Kumaran[1], Thore Graepel[1], Timothy Lillicrap[1], Karen Simonyan[1], Demis Hassabis[1†]

The game of chess is the longest-studied domain in the history of artificial intelligence. The strongest programs are based on a combination of sophisticated search techniques, domain-specific adaptations, and handcrafted evaluation functions that have been refined by human experts over several decades. By contrast, the AlphaGo Zero program recently achieved superhuman performance in the game of Go by reinforcement learning from self-play. In this paper, we generalize this approach into a single AlphaZero algorithm that can achieve superhuman performance in many challenging games. Starting from random play and given no domain knowledge except the game rules, AlphaZero convincingly defeated a world champion program in the games of chess and shogi (Japanese chess), as well as Go.

Dalsze wyzwania dla sztucznej inteligencji?

STARCRAFT (ALPHASTAR)

TEXAS HOLD `EM POKER (DEEPSTACK)



Podsumowanie

- Początkowo algorytmy były trenowane na bazie wiedzy eksperckiej.
- Sieci neuronowe, wykorzystanie m.in. przez *DeepBlue*, pozwalają wytrenować algorytm do wykonywania ruchu dla dowolnego stanu gry.
- Algorytm połączenie sieci neuronowych z algorytmem MCTS w *AlphaGo*, pozwoliło na dostarczenie bardziej trafnych przewidywań wyniku gry od samych sieci neuronowych.
- Obecnie algorytmy (np. *AlphaZero*) nie wymagają wiedzy eksperckiej – trenują rozgrywając rozgrywki z samym sobą wykorzystując tzw. uczenie wzmocnione.