

AI

---

## Artificial Intelligence



Universiteit  
Leiden

Walter Kosters

[www.liacs.leidenuniv.nl/~kosterswa/](http://www.liacs.leidenuniv.nl/~kosterswa/)

Labyrint — Thursday, February 7, 2019

**VN Detective en Thrillergids**  
18720 titels [www.vnster.nl](http://www.vnster.nl)

thuis [hulp](#) [pdf/csv](#) uitgebreid zoeken op: [titel](#) [auteur](#) [bladeren](#)

[mobiel](#) [contact](#)

U bevindt zich hier: [thuis](#) — [www.vnster.nl](#)



**Vrij Nederland**  
Detective en Thrillergids

Sinds 1980 verblijft Vrij Nederland (VN) ons iedere zomer met een Detective en/ Thrillergids. Deze bestaat onder meer uit een uitgebreide lijst met in het Nederlands verkrijgbare titels. Op 5 juni 2018 verscheen de 39ste gids. Voor de recensies leze men de papieren of digitale versie bij VN.

Er zijn in het Nederlands nog meer "spannende" boeken verschenen. Het streven is zoveel mogelijk titels in de lijst(en) op deze website op te nemen. De informatie is toegankelijk via verschillende zoekmogelijkheden, ook mobiel of met een app, en er kan direct



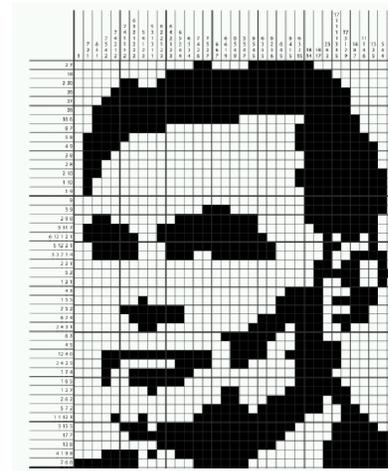
**Vrij Nederland**  
Detective en Thriller Gids

[gebladerd](#) worden.

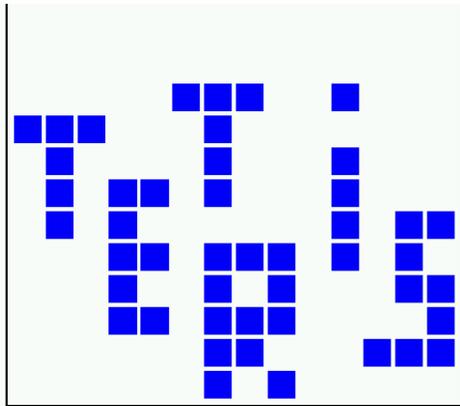
De navigatie is steeds in de kleur van het "submenu": [lichtblauw](#)—[algemeen](#), [oranje](#)—[zoeken](#), respectievelijk [donkerrood](#)—[bladeren](#) (als de muis over een "IMDb-link" of "wiki-link" gaat, komt de eerste alinea van de bijbehorende pagina in beeld; analoog voor de foto's). Een en ander is tevens in PDF-formaat beschikbaar: 500 pagina's, 14 MB, en ook in CSV-formaat. Voor vragen, opmerkingen of aanvullingen: neem gerust [contact](#) op. Of werk aan een eigen [collectie](#). Voor meer achtergronden raadplege men de [hulp](#)-pagina's. Voor de duidelijkheid: er zijn hier geen boeken te koop!

[mobiel](#) [contact](#)

[link](#)

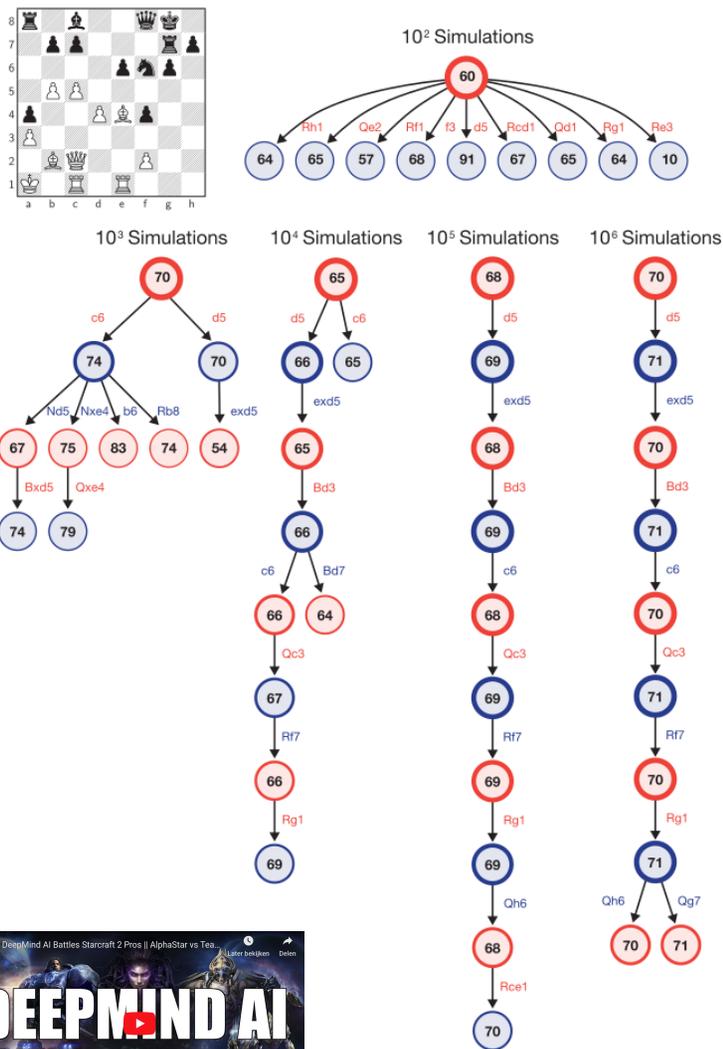


[link](#)



[link](#)

mystery novels,  
tomography  
and Tetris



December 2018  
 AlphaZero  
 Silver et al.  
 Science 362, 1140–1144



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

David Silver<sup>1,2\*</sup>, Thomas Hubert<sup>1</sup>, Julian Schrittwieser<sup>1</sup>, Ioannis Antonoglou<sup>1</sup>, Matthew Lai<sup>1</sup>, Arthur Guez<sup>1</sup>, Marc Lanzen<sup>1</sup>, Laurent Sifre<sup>1</sup>, Dhruv Neyman<sup>1</sup>, Thore Graepel<sup>1</sup>, Timothy Lillicrap<sup>1</sup>, Koray Senescevit<sup>1</sup>, David Hasselbach<sup>1</sup>

The game of chess is the longest studied domain in the history of artificial intelligence. The strongest programs are based on a combination of combinatorial search techniques, domain-specific algorithms, and handcrafted evaluation functions that have been refined by human experts over several decades. In contrast, the AlphaZero 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 autonomously identified a world champion program in the games of chess and shogi (Japanese chess), as well as Go.

of Go by representing Go knowledge with the use of deep convolutional neural networks (7, 8), trained solely by reinforcement learning from games of self-play (9). In this paper, we introduce AlphaZero, a more general version of the AlphaZero algorithm that accommodates, without special casing, a broader class of game rules. We apply AlphaZero to the games of chess and shogi, as well as Go, by using the same algorithm and network architecture for all three games. Our results demonstrate that a general-purpose reinforcement learning algorithm can learn, tabula rasa—without domain-specific human knowledge or data, as evidenced by the same algorithm performing well in multiple domains—superhuman performance across multiple challenging games.

A landmark for artificial intelligence was achieved in 1997 when Deep Blue defeated the human world chess champion (2). Computer chess programs continued to progress steadily beyond human level in the following two decades. These programs evolved gradually by using handcrafted features and carefully tuned weights, constructed by strong human players and

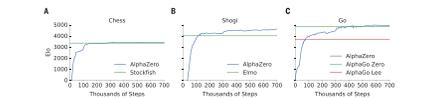
programmers, combined with a high-performance alpha-beta search that expands a vast search tree by using a large number of clever heuristics and domain-specific adaptations. On top of these heuristics and adaptations, leading on the 2016 Top Chess Engine Championship (TCEC) scores 9 world champion (Shakhovskii (2)), other strong chess programs, including Deep Blue, use very similar architectures (2, 22).

In terms of game tree complexity, shogi is a substantially harder game than chess (21, 24). It is played on a larger board with a wider variety of pieces, any captured opponent piece reenters the board and may subsequently be dropped anywhere on the board. The strongest shogi program, such as the 2017 Computer Shogi Association (CSA) world champion Elmo, has only recently defeated human champions (25). These programs use an algorithm similar to those used by computer chess programs, again based on a highly optimized alpha-beta search engine with many domain-specific adaptations.

AlphaZero replaces the handcrafted knowledge and domain-specific adaptations used in traditional game-playing programs with deep neural networks, a general-purpose reinforcement learning algorithm, and a general-purpose tree search algorithm.

Instead of a handcrafted evaluation function and move-ordering heuristics, AlphaZero uses a deep neural network (p. 9)  $V_{\pi}$  with parameters  $\theta$ . This neural network  $V_{\pi}$  takes the board position as an input and outputs a vector of move probabilities with components  $p_i = \text{Pr}(i)$  for each action  $i$  and a scalar value  $v$  estimating the expected outcome of the game from position  $x$ ,  $v = V_{\pi}(x)$ . AlphaZero learns these move probabilities and value estimates entirely from self-play; these are then used to guide its search in future games.

Instead of an alpha-beta search with domain-specific enhancements, AlphaZero uses a general-purpose Monte Carlo tree search (MCTS) algorithm. Each search consists of a series of simulated games of self-play that traverse a tree from root state  $x_{\text{root}}$  until leaf state is reached. Each simulation proceeds by selecting in each state  $x$  a move  $a$  with the highest count (not previously frequently explored), high move probability, and high value (averaged over the leaf states of



**Fig. 1. Training AlphaZero for 700,000 steps.** Elo ratings were constructed from games between different players where each player was given 1-p per move. (A) Performance of AlphaZero in chess, compared with the 2016 TCEC world champion program Stockfish. (B) Performance of AlphaZero in shogi compared with the 2017 CSA world champion program Elmo. (C) Performance of AlphaZero in Go compared with AlphaGo Lee and AlphaGo Zero (GO ratings over 3 days).

Silver et al., Science 362, 1140–1144 (2018) | 7 December 2018

Downloaded from www.sciencemag.org on January 21, 2018

Artificial Intelligence (AI) tries to address many questions:

- *robotics*: How to program a robot?
- *data mining*: What is hidden in WikiLeaks?
- *law*: Can a machine act as judge? NLP
- *linguistics*: “the spirit is willing but the flesh is weak”  
→ ... → “the vodka is good but the meat is rotten”? ↙
- *gaming*: Can a computer play Fortnite?
- *neural networks*: How to predict the stock exchange?
- *cognition*: How can we mimic the human brain?

One can look at AI from different angles:

1. from *psychology* or *philosophy*:

What is the difference between a human and a computer?

Can a machine/computer think?

2. from *computer science*:

How to make a chess program?

How does a Mars robot function?

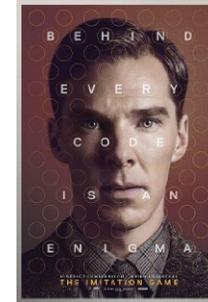
machine learning

“Do androids dream of electric sheep?” →



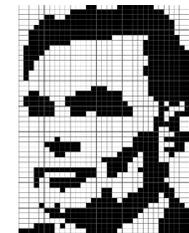
AI lets a computer do things that are called **intelligent** if humans would do it like that. And learning. And autonomy?

The famous **Turing test** from 1950 asks (“the imitation game”):



We can communicate, by means of a keyboard and a text screen, with an entity in a locked room.

Can we decide whether it is a human or a machine?



By the way, the original problem was: man ↔ woman.

And how about John Searle’s Chinese Room experiment (understanding and consciousness)?

Already in 1964–66 Joseph Weizenbaum from MIT/Boston wrote the nice psychotherapy program **ELIZA/DOCTOR**. An example conversation from Jason Hutchens' HAL/HeX (1997):

my name is bob

Pleased to meet you.

how are you doing, jas?

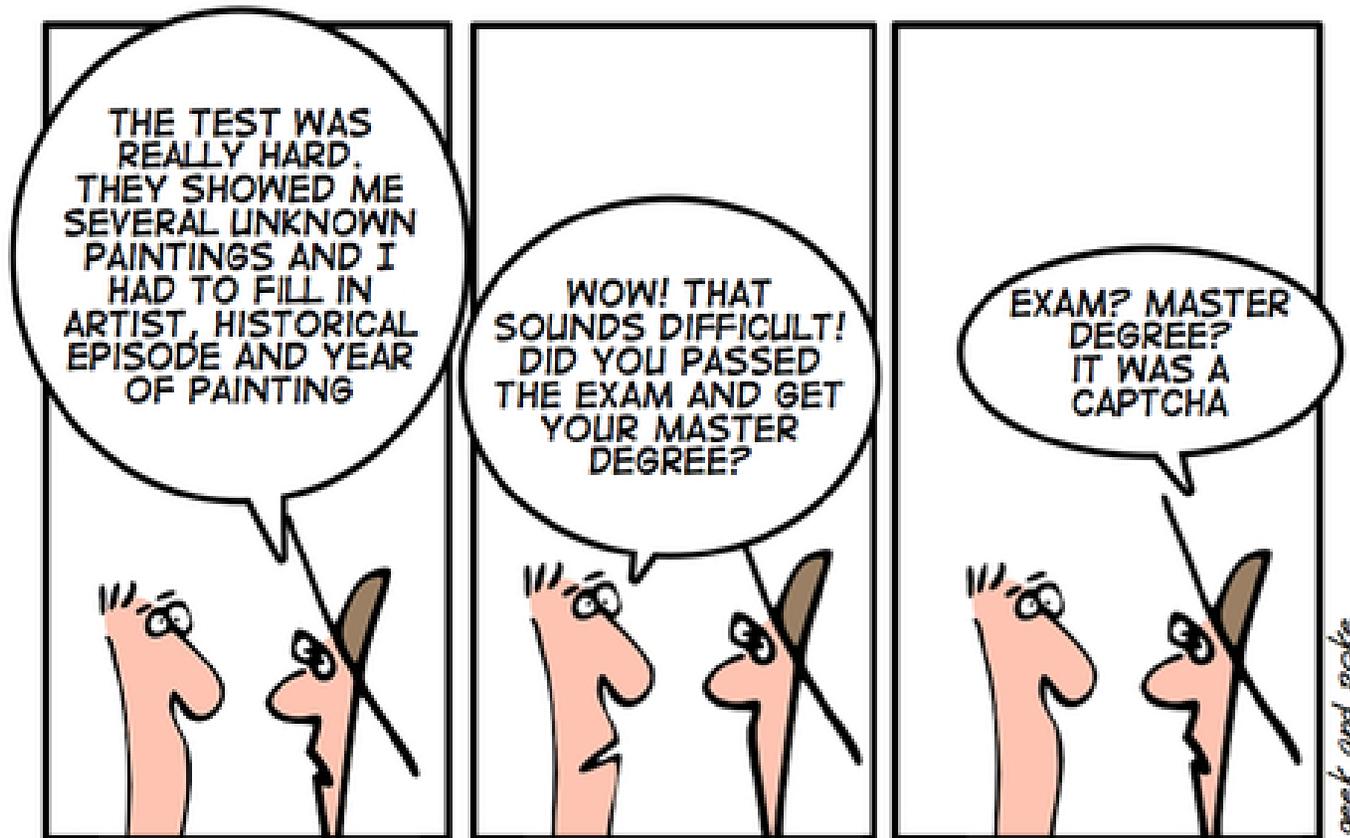
I am just fine, thanks! And you?

i am feeling great!

If you are feeling great, it's your business, mate!



The [Loebner Prize](#) tries to implement the Turing test.



geek and punk

elzoufzi  
keyetr  
mathics  
numexpa  
ommarie  
ntednbus

IN THE FUTURE SOPHISTICATED CAPTCHAS WILL LOCK OUT ANY BOT

In 2011 IBM used a computer to play “Jeopardy!”:

1990 POP CULTURE	10 <sup>TH</sup> CANNON	KNOW YOUR BORDERS	WHAT'S YOUR SIGN	FLY LIKE AN EAGLE	NATIONAL PASTRIES
\$100	\$100	\$100	\$100	\$100	\$100
\$200	\$200	\$200	\$200	\$200	\$200
\$300	\$300	\$300	\$300	\$300	\$300
\$400	\$400	\$400	\$400	\$400	\$400
\$500	\$500	\$500	\$500	\$500	\$500

**IN 2013 ROB FORD,  
MAYOR OF THIS 4th-  
LARGEST CITY IN N.  
AMERICA, FIRST SAID  
HE SMOKED WEED,  
NOT CRACK...THEN  
YES, OK, CRACK, TOO**



What is  
**Toronto????**



A **robot** acts in the physical world (what about softbots?). The word originates from 1921 or earlier, and is coined by the Čapek brothers from Czechia.

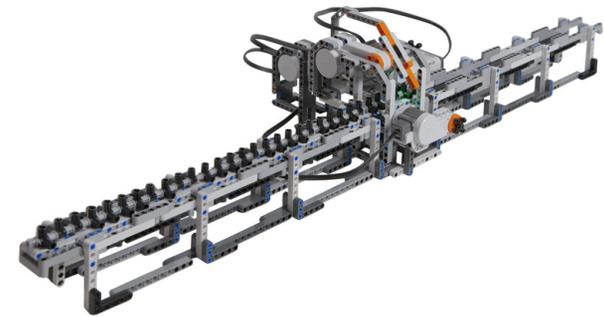
The science fiction author Isaac Asimov (“I, Robot”) conceived the three laws of **robotics**:

1. A robot must not harm a human.
2. A robot must obey human orders, unless this contradicts law 1.
3. A robot must protect it(?)self, unless this contradicts law 1 or 2.



A simple program for a **Lego** robot makes it/him/her move randomly, which looks quite intelligent to some.

```
task main ( ) {  
    while ( true ) {  
        OnFwd (OUT_A + OUT_C);  
        Wait (Random (100) + 40);  
        OnRev (OUT_A);  
        Wait (Random (85) + 30);  
    }  
}
```



Lego Turing machine

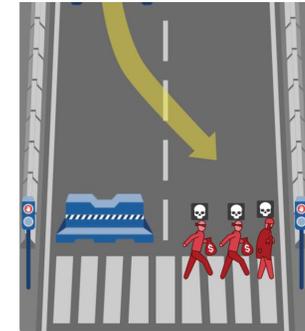
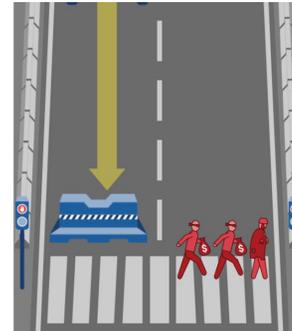


Chang'e-3 Moon lander



Pepper robot from Softbank (2016)

Currently self-driving cars perform rather well.



↑computer vision, knowledge, law, ethics↑, ...



Udacity and Coursera have **MOOCs** on self-driving cars.

**Maxi** and **Mini** play a simple game: **Maxi** chooses a (horizontal) row, and then **Mini** chooses a (vertical) column:

	3	12	8
	2	4	6
①	14	5	2

②

Example: **Maxi** ① chooses row 3, and then **Mini** ② chooses column 2: kind, but stupid; the game outcome is 5.

**Maxi** wants the outcome to be as high as possible, **Mini** as low as possible.

How to analyze this?

If **Maxi** chooses row 1, **Mini** chooses column 1 (gives 3);  
if **Maxi** chooses row 2, **Mini** chooses column 1 (gives 2);  
if **Maxi** chooses row 3, **Mini** chooses column 3 (gives 2).

3	12	8
2	4	6
14	5	2

So **Maxi** must choose row 1!

Indeed, **Maxi** chooses the row with the *highest minimum*. This is in fact Von Neumann's 1928 *minimax algorithm*.

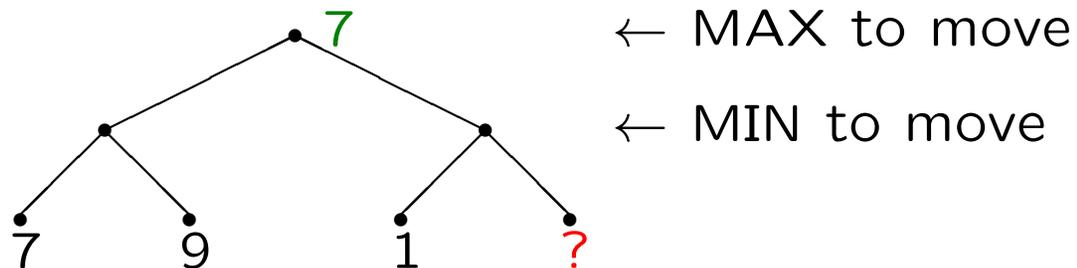


cf. Prisoner's dilemma

Now notice that the same analysis holds if we do not know what the ?s are.

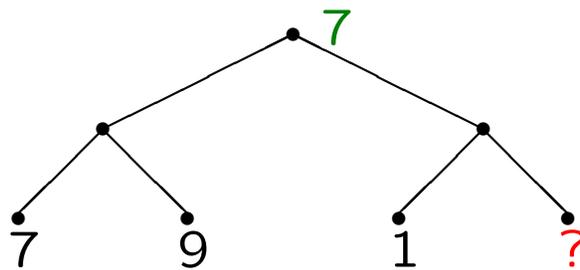
3	12	8
2	?	?
14	5	2

The  $\alpha$ - $\beta$ -algorithm uses this observation, and until recently was at the basis of every chess program.





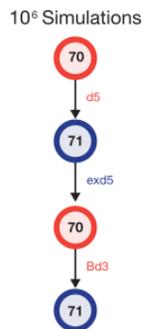
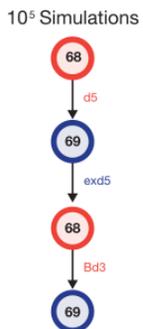
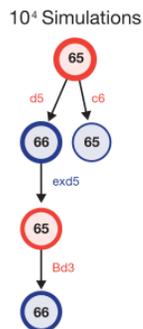
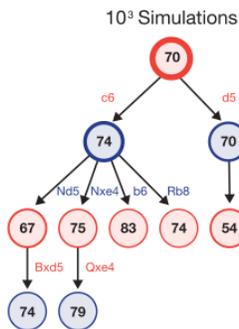
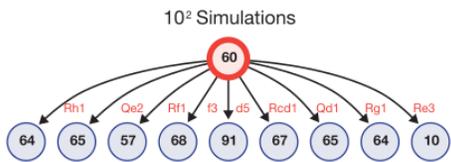
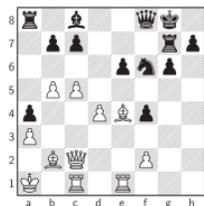
Deep Blue (with minimax/ $\alpha$ - $\beta$ ) vs. Garry Kasparov, 1997



← MAX to move

← MIN to move

cf. Adriaan de Groot



December 2018

AlphaZero



Silver et al.

Science 362, 1140-1144

RESEARCH

COMPUTER SCIENCE

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

David Silver<sup>1,2\*</sup>, Thomas Hubert<sup>1</sup>, Julian Schrittwieser<sup>1</sup>, Ioannis Antonoglou<sup>1</sup>, Matthew Lai<sup>1</sup>, Arthur Guez<sup>1</sup>, Marc Lanzen<sup>1</sup>, Laurent Sifre<sup>1</sup>, Shariq Khan<sup>1</sup>, Thore Graepel<sup>1</sup>, Timothy Lillicrap<sup>1</sup>, Koray Senescevit<sup>1</sup>, David Hasselbach<sup>1</sup>

The game of chess is the longest studied domain in the history of artificial intelligence. The strongest programs are based on a combination of combinatorial search techniques, domain-specific algorithms, and handcrafted evaluation functions that have been refined by human experts over several decades. In contrast, the AlphaZero 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 autonomously identified a world champion program in the games of chess and shogi (Japanese chess), as well as Go.

The study of computer chess is as old as computer science itself. Charles Hubbs, Alan Turing, Claude Shannon, and John von Neumann devised hardware, algorithms, and theory to analyze and play the game of chess. Chess subsequently became a grand challenge task for a generation of artificial intelligence researchers, culminating in high-performance computer chess programs that play at a superhuman level (2, 3). However, these systems are highly tuned to their domain and cannot be generalized to other games without substantial human effort, whereas general game-playing systems (4, 5) remain comparatively weak. A long-standing ambition of artificial intelligence has been to create programs that can instead learn for themselves from their experience (6, 7). Recently, the AlphaZero algorithm achieved superhuman performance in the game

of Go by representing Go knowledge with the use of deep convolutional neural networks (7, 8), trained solely by reinforcement learning from games of self-play (9). In this paper, we introduce AlphaZero, a more general version of the AlphaZero algorithm that accommodates, without special casing, a broader class of game rules. We apply AlphaZero to the games of chess and shogi, as well as Go, by using the same algorithm and network architecture for all three games. Our results demonstrate that a general-purpose reinforcement learning algorithm can learn, tabula rasa—without domain-specific human knowledge or data, as evidenced by the same algorithm succeeding in multiple domains—superhuman performance across multiple challenging games.

A landmark for artificial intelligence was achieved in 1997 when Deep Blue defeated the human world chess champion (1). Computer chess programs continued to progress steadily beyond human level in the following two decades. These programs evolved gradually by using handcrafted features and carefully tuned weights, constructed by strong human players and

programmers, combined with a high-performance alphabeta search that expands a vast search tree by using a large number of clever heuristics and domain-specific adaptations. On 10 we describe these adaptations, focusing on the 2016 Top Chess Engine Championship (TCEC) series. It world champion Stockfish (2); other strong chess programs, including Deep Blue, use very similar architectures (2, 22).

In terms of game tree complexity, shogi is a substantially harder game than chess (23, 24). It is played on a larger board with a wider variety of pieces, any captured opponent piece reenters side and may subsequently be dropped anywhere on the board. The strongest shogi program, such as the 2017 Computer Shogi Association (CSA) world champion Elmo, have only recently defeated human champions (25). These programs use an algorithm similar to those used by computer chess programs, again based on a highly optimized alphabeta search engine with many domain-specific adaptations.

AlphaZero replaces the handcrafted knowledge and domain-specific adaptations used in traditional game-playing programs with deep neural networks, a general-purpose reinforcement learning algorithm, and a general-purpose tree search algorithm. Instead of a handcrafted evaluation function and move-ordering heuristics, AlphaZero uses a deep neural network (p, v) (26) with parameters  $\theta$ . This neural network  $Q(\theta)$  takes the board position as an input and outputs a vector of move probabilities with components  $p_i = \text{Pr}(i)$  for each action  $i$  and a scalar value  $v$  estimating the expected outcome of the game from position  $x$ ,  $v = Q(x)$ . AlphaZero learns these move probabilities and value estimates entirely from self-play; these are then used to guide its search in future games.

Instead of an alphabeta search with domain-specific enhancements, AlphaZero uses a general-purpose Monte Carlo tree search (MCTS) algorithm. Each search consists of a series of simulated games of self-play that traverse a tree from root state  $x_{\text{root}}$  until leaf state is reached. Each simulation proceeds by selecting in each state  $x$  a move  $a$  with the visit count (not previously frequently explored), high move probability, and high value (averaged over the leaf states of

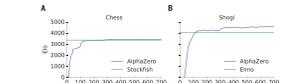


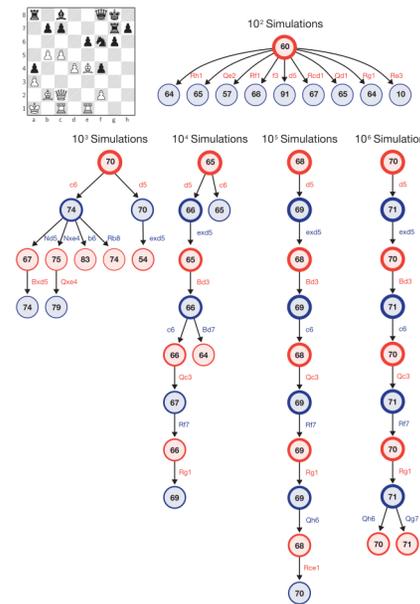
Fig. 1. Training AlphaZero for 700,000 days. Elo ratings were constructed from games between different players where each player was given 1-p per move. (A) Performance of AlphaZero in chess, compared with the 2016 TCEC world champion program Stockfish. (B) Performance of AlphaZero in shogi compared with the 2017 CSA world champion program Elmo. (C) Performance of AlphaZero in Go compared with AlphaGo Zero and AlphaGo Zero (20) networks over 3 days.

Silver et al., Science 362, 1140–1144 (2018) | 7 December 2018

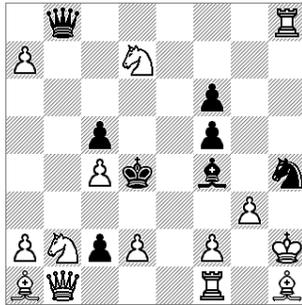


**AlphaZero** quickly teaches itself to play chess, shogi and Go at super-human level. What are the main ingredients of this breakthrough?

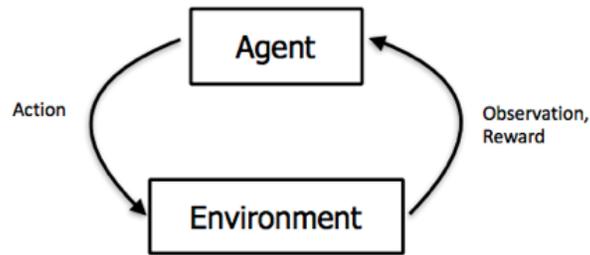
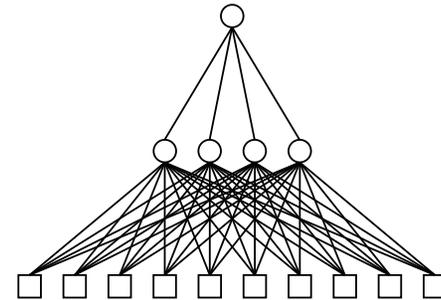
- reinforcement learning
- self-play
- Monte Carlo Tree Search →
- deep neural networks
- unlimited resources (Google: men, machines, . . . )



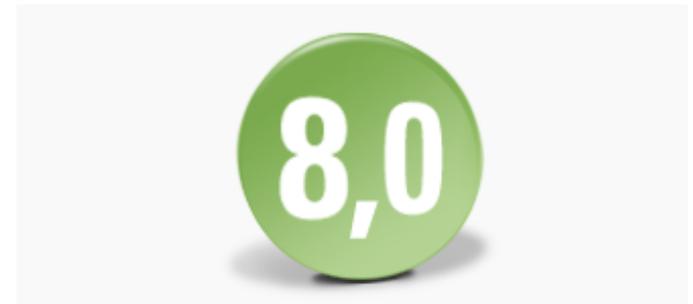
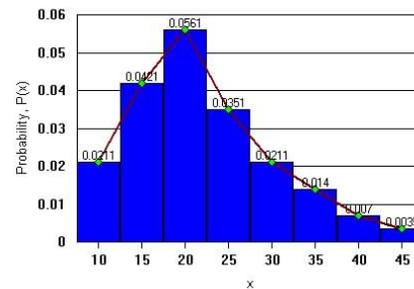
But no human game experts (“tabula rasa” )!



Output units  $a_i$   
 $W_{j,i}$   
Hidden units  $a_j$   
 $W_{k,j}$   
Input units  $a_k$



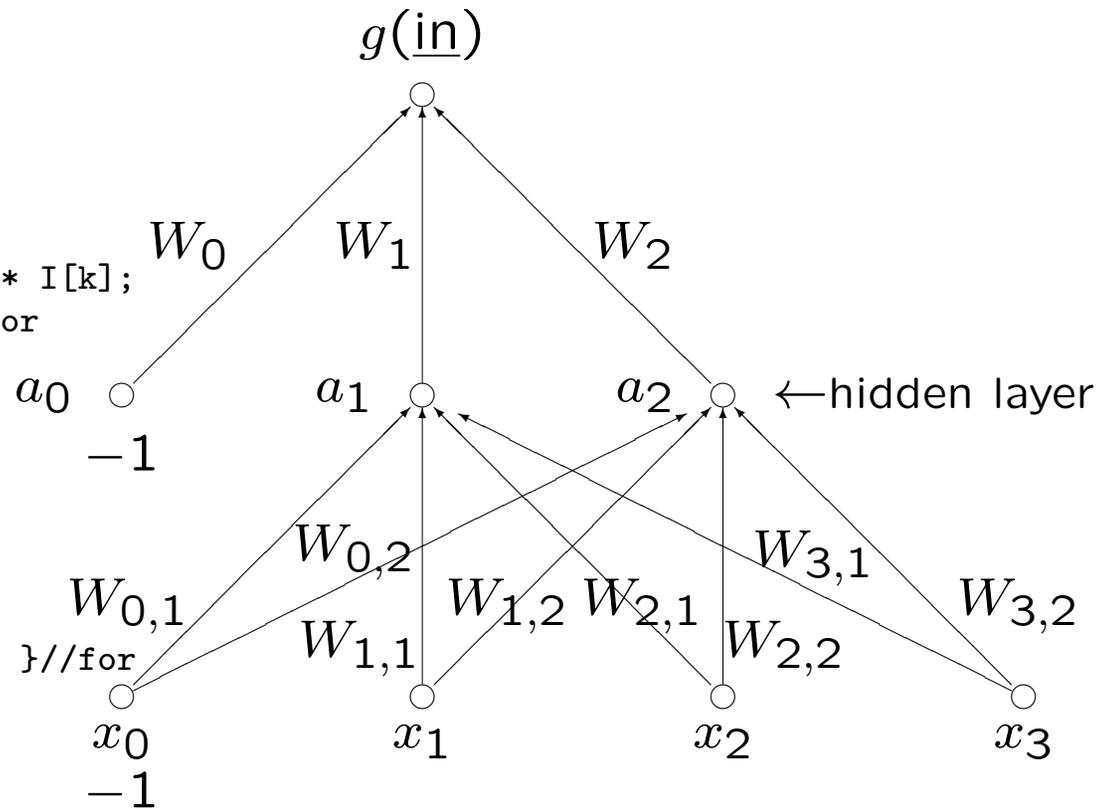
Probability Distribution



```

for ( j = 1; j <= hiddens; j++ ) {
    sum[j] = -inputtohidden[0][j];
    for ( k = 1; k <= inputs; k++ )
        sum[j] += inputtohidden[k][j] * I[k];
    hiddenacti[j] = g (sum[j]); }//for
for ( i = 0; i < outputs; i++ ) {
    sumout = -hiddentooutput[0][i];
    for ( j = 1; j <= hiddens; j++ )
        sumout += hiddentooutput[j][i]
            * hiddenacti[j];
    O[i] = g (sumout);
    deltaout[i] = gprime (sumout)
        * ( T[i] - O[i] ); }//for
for ( j = 1; j <= hiddens; j++ ) {
    delta[j] = 0;
    for ( i = 0; i < outputs; i++ )
        delta[j] += hiddentooutput[j][i]
            * deltaout[i];
    delta[j] *= gprime (sum[j]); }//for
for ( j = 0; j <= hiddens; j++ )
    for ( i = 0; i < outputs; i++ )
        hiddentooutput[j][i] += alpha * hiddenacti[j] * deltaout[i];
for ( k = 0; k <= inputs; k++ )
    for ( j = 1; j <= hiddens; j++ )
        inputtohidden[k][j] += alpha * I[k] * delta[j];

```



C++/Python/TensorFlow/Keras

convolutional NN

Artificial Intelligence (AI) will be around . . .

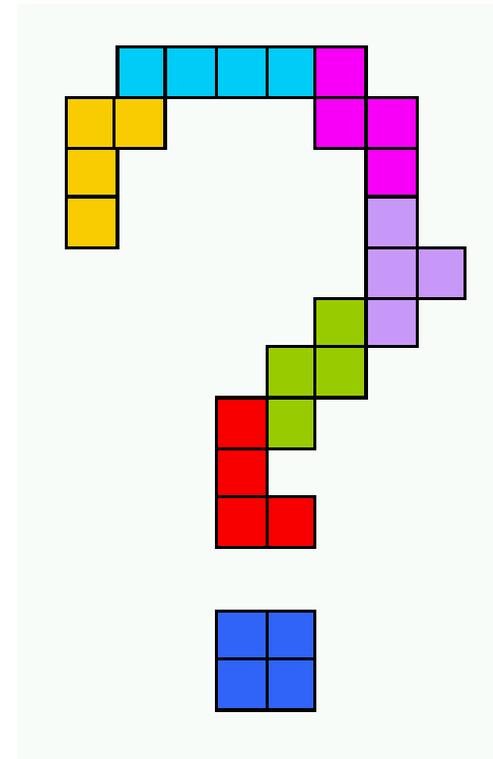
Some references:

- Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2010.
- Richard Sutton and Andrew Barto, *Reinforcement Learning*, MIT Press, 2018.
- Julian Togelius, *Playing Smart: On Games, Intelligence, and Artificial Intelligence*, MIT Press, 2018.
- Alan Turing, *Computing Machinery and Intelligence*, *Mind* 59, 433–460, 1950.

What is the role of Artificial Intelligence (AI) for Psychology in the future?



Universiteit  
Leiden



[www.liacs.leidenuniv.nl/~kosterwa/psy.pdf](http://www.liacs.leidenuniv.nl/~kosterwa/psy.pdf)