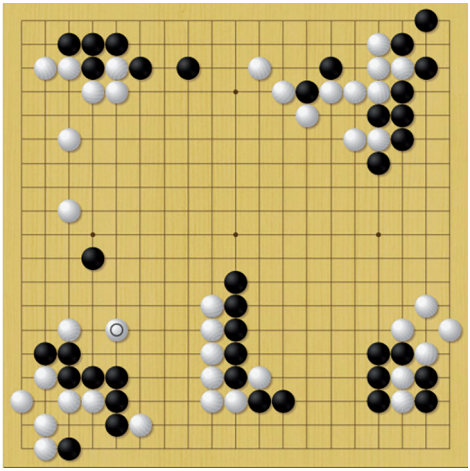
Ein Liebesbrief an KataGo …

von Branton DeMoss



KataGo vs. Leela Zero: B+Resign[1]

Um einen Computer so zu programmieren, dass er eine vernünftige Go-Partie – und nicht nur eine legale Partie – spielt, ist es notwendig, die Prinzipien einer guten Strategie zu formalisieren oder ein Lernprogramm zu entwerfen. Die Prinzipien sind qualitativer und mysteriöser als beim Schach und hängen mehr vom Urteilsvermögen ab. Daher wird es meiner Meinung nach noch schwieriger sein, einen Computer so zu programmieren, dass er eine vernünftige Partie Go spielt, als beim Schach.

I. J. Good, The Mystery of Go, 1965

Das Go-Spiel hat etwas Magisches an sich. Seit Tausenden von Jahren fesselt es die Phantasie derer, die lernen wollen, was es heißt zu lernen, und die darüber nachdenken, was Denken bedeutet.

Mit dem jüngsten Aufkommen einer starken, quelloffenen Go-KI, die Spitzenprofis schlagen kann, lohnt es sich, die Geschichte des Spiels nachzuvollziehen, um zu ergründen, warum es so lange so schwierig war, Menschen zu schlagen – und was die Zukunft des Go für uns bereit hält.

[1] *https://bit.ly/3jM5Imz*

Klassische KI

Auf die eine Weise betrachtet, weiß jeder, was Intelligenz ist; auf eine andere Weise betrachtet, weiß das niemand.

Robert Sternberg, 2000

Wie wir KI definieren, hat sich im Laufe der Zeit verändert. Ältere, naive Definitionen beschäftigten sich meist mit der Fähigkeit, bestimmte Aufgaben zu erfüllen, und definierten KI als:

Die Wissenschaft von der Herstellung von Maschinen, die in der Lage sind, Aufgaben auszuführen, die, wenn sie von Menschen ausgeführt würden, Intelligenz erfordern würden.

Marvin Minsky, 1968

Definitionen wie diese sind von Natur aus instabil, denn wenn wir diese „intelligenten" Computersysteme bauen und uns auf ihre (zunächst) erstaunlichen Fähigkeiten normalisieren, hören wir auf, ihre Aufgabenleistung als Demonstration irgendeiner Art von Intelligenz zu betrachten. Die Definition von Intelligenz im Verhältnis zu den Fähigkeiten bei bestimmten Aufgaben, egal wie schwierig sie auch sein mögen, lässt die KI in einer Art „Gott der Lücken"-Situation[2] zurück.

Einige haben vorgeschlagen, dass Intelligenz die Fähigkeit ist, viele Aufgaben gut auszuführen, oder die Fähigkeit, Aufgaben in einer Vielzahl von Umgebungen zu lösen.[3] Andere behaupten, dass Intelligenz die Fähigkeit ist, durch Lernen neue Fähigkeiten zu erwerben.[4] In jüngerer Zeit gab es Vorschläge,[5] dass Intelligenz ein Maß für die Effizienz des Erwerbs von Fähigkeiten ist. Wenn man zwei Agenten mit dem gleichen Wissen und einer festen Trainingszeit für eine neuartige Aufgabe einsetzt, ist der intelligentere Agent derjenige, der am Ende die besseren Fähigkeiten erwirbt.

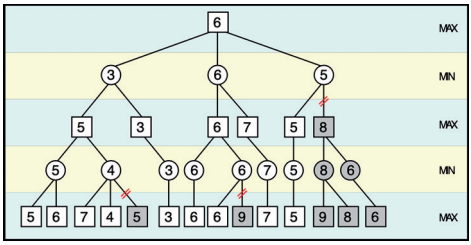
[2] *en.wikipedia.org/wiki/God_of_the_gaps*

[3] Legg and Hutter: A Collection of Definitions of Intelligence, *arxiv.org/abs/0706.3639*

[4] Hernandez-Orallo: Evaluation in artificial intelligence: from task-oriented to ability-oriented measurement. Artificial Intelligence Review, 2017, S. 397–447.

[5] Chollet: On the Measure of Intelligence, *arxiv.org/pdf/1911.01547.pdf*

Das beliebteste KI-System des letzten Jahrhunderts war Deep Blue, ein Schachspielsystem, das von Forschern bei IBM entwickelt wurde. Das System bestand aus einer händisch programmierten Brettbewertungsfunktion, einer Baumsuche, um den erwarteten Brettzustandswert bei einem feindseligen Gegner zu maximieren, und maßgeschneiderter Hardware zur Beschleunigung dieser Operationen, wobei Geschwindigkeiten von etwa 100 Millionen Stellungsbewertungen pro Sekunde erreicht wurden.



Alpha-Beta-Entscheidungsbaum[6]

Bewertungsfunktionen messen die „Güte“ von Zuständen (= wie wahrscheinlich es ist, dass sie zum Sieg führen). Sie nehmen den Zustand des Bretts als Input und geben eine Schätzung des Wertes der aktuellen Stellung aus. Die Erstellung aussagekräftiger Bewertungsfunktionen ist keine einfache Aufgabe – in der Tat bestand die Deep-Blue-Bewertungsfunktion aus 8000 handkodierten Heuristiken![7] Programmierer setzten sich mit Schachexperten zusammen, um verschiedenen Brettzuständen einen Wert zuzuweisen – Türme auf dem hinteren Rang, freistehende Bauern, Sicherheit des Königs usw. All diese einzelnen Heuristiken wurden sorgfältig gewichtet und kombiniert, um einen ungefähren singulären Brett-Gesamtwert zu erhalten, der dann über eine Baumsuche durch viele mögliche zukünftige Brettzustände optimiert wurde.

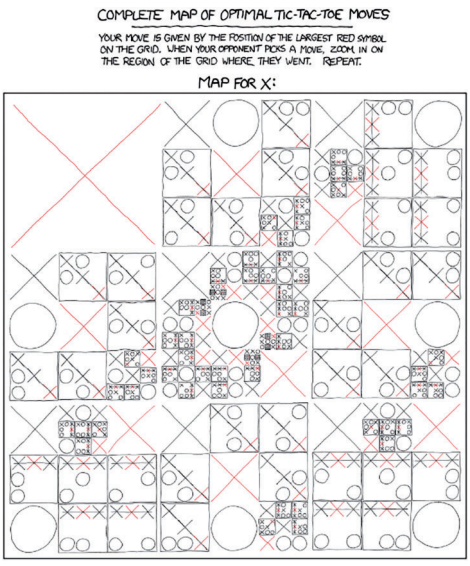
Mit einer gut abgestimmten Bewertungsfunktion und einer leistungsfähigen Baumsuche zur wertmaximierende Vorausberechnung gelang Deep Blue 1997 ein Sieg über Schachweltmeister Garry Kasparow.[8]

[6] en.wikipedia.org/wiki/Alpha–beta_pruning
[7] Campbell et al: Deep Blue, core.ac.uk/download/pdf/82416379.pdf
[8] en.wikipedia.org/wiki/Deep_Blue_versus_Garry_Kasparov

Deep Blue ist ein Beispiel für ein „Expertensystem“ – eines, in dem menschliches Expertenwissen kodiert ist. Es hat nicht aus seinem Spiel gelernt oder neuartige Heuristiken oder Erkenntnisse generiert, es maximierte allein den Wert des Brettzustands entsprechend der von Menschen vorgegebenen Wertfunktion.

Komplexität

Go ist wie Schach ein deterministisches Spiel mit perfekter Information. Es gibt keine Zufallskomponente, keinen verborgenen Zustand. Anders als beim Schach, bei dem es im Durchschnitt etwa 35 legale Züge pro Zug zu berücksichtigen gibt, sind bei Go im Durchschnitt etwa 250 legale Züge zu berücksichtigen. Und bei Tic-Tac-Toe können wir den gesamten Partiebaum durchsuchen und leicht die optimale Antwort für jede Stellung finden. „Xkcd“ hat dies in einem Bild schön zusammengefasst:[9]



Obwohl es prinzipiell möglich ist, einen solchen Baum für Go zu erstellen, da es sich um ein endliches Spiel handelt, ist der Zustandsraum von Go extrem groß: Die Anzahl der legalen Stellungen[10] im Go beträgt etwa $2{,}1 \times 10^{170}$.

[9] xkcd.com/832/
[10] Tromp: Number of legal Go positions, *tromp.github.io/go/legal.html*

Da ein Spiel eine Kurve durch legale Brettzustände ist (mit einigen Übergangsbeschränkungen), ist die Anzahl der möglichen Go-Spiele erheblich größer. Die Anzahl der einzigartigen Go-Spiele ist liegt zwischen ($10^{10^{108}}$, $10^{10^{171}}$).[11]

Intuition und Lektüre

Da der Zustandsraum bei Go zu groß ist, um aufgezählt und durchsucht zu werden, müssen die Spieler lernen, sich bei der Betrachtung möglicher Spielzustandsverläufe (Varianten) nur auf vielversprechende Züge zu konzentrieren, d. h., die Spieler müssen ein intuitives Gespür dafür entwickeln, welche Züge gut sein könnten, und es vermeiden, Zeit mit zweifelhaften Möglichkeiten zu verschwenden. Empirisch erweist sich die Definition einer solchen Wertfunktion für Go als viel schwieriger als für Schach – Bemühungen, das Schach in die Richtung von Go zu spiegeln, brachten keine guten Ergebnisse.

Während die Zugauswahl von der Intuition geleitet wird, stärkt das Auslesen von Varianten die Intuition durch eine Form von Selbstargumentation: Da Go ein Nullsummen-Spiel ist, ist die Zugauswahl notwendigerweise von einem gegnerischen Spieler abhängig. Die Ziele des Spielers sind perfekt gegeneinander ausgerichtet, so dass eine optimale Strategie konstruiert werden kann, indem man in Betracht zieht, den zukünftigen Zustandswert bei einem minimierenden Gegner zu maximieren (diese Logik ist im Minimax-Algorithmus schön kodiert).[12]

Angesichts der Schwierigkeiten, eine aussagekräftige Bewertung für Go zu entwickeln, war eine modifizierte Baumsuche, Monte Carlo Tree Search (MCTS)[13], ein Ansatz, der einige Erfolge verzeichnen konnte. MCTS sucht nach dem Zufallsprinzip legale Züge in der aktuellen Stellung und durchsucht den Spielbaum bis zum Ende, wobei jede Stellung aus Zufallszügen entwickelt wird. Der Wert des Anfangszugs bemisst sich am Anteil an Durchgängen, die zu einer gewonnenen Partie führen. Etwas überraschend war, dass Go-Bots mit MCTS in der Lage waren, fortgeschrittenes Spiel auf Amateurniveau (Low-Mid-Dan) zu erreichen.

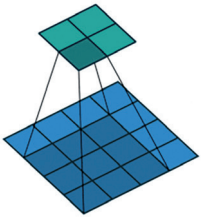
Es liegt etwas sehr Interessantes in der Tatsache, dass die Definition von Zustandswerten durch die Auswertung zufälliger Rollouts bis zum Ende tatsächlich eine sinnvolle Annäherung an den „wahren Wert" darstellt. Es scheint tautologisch zu sein, wenn man es ausspricht, aber wirklich „gute" Züge haben wirklich einen größeren Anteil an Entwicklungen, die zum Sieg führen, und eine Zufallsauswahl reicht aus, um sich ihrem Wert anzunähern.

Neuronale Netzwerke

Wenn die Heuristik der Spielbrettbewertung und der Zugauswahl so schwer zu programmieren, ja sogar zu spezifizieren ist, wieso können Menschen so gut Go spielen? Einige Profis können viele Varianten sehr schnell auslesen, aber das ist nichts im Vergleich zu den Hunderten von Millionen Stellungen pro Sekunde, auf die Deep Blue kommt.

Die menschliche Intuition bei der Zugauswahl ist ausgezeichnet. Auf den ersten Blick fällt eine sehr kleine Anzahl von Zügen als überlegenswert auf. Aus der Erfahrung vieler Go-Partien scheinen wir in der Lage zu sein, ein scharfes Gespür dafür zu entwickeln, welche Züge funktionieren und welche nicht. Wir haben auch den Vorteil, dass wir die Go-Theorie lesen können, die die destillierte Erfahrung vieler anderer über Jahrtausende hinweg darstellt (die Einbeziehung symbolischen Wissens in Lernsysteme ist ein ungelöstes Problem).

Wie kann man KI-Systemen diese ausgezeichnete Intuition vermitteln? Durch sogenannte Convolutional Neural Networks (faltende neuronale Netze)!



Ein konvolutionärer Kern (dunkelblau) wird auf die Eingabe (blau) angewendet, um die Ausgabe (cyan) zu erzeugen

[11] Untergrenze: Walraet/Tromp: A Googolplex of Go Games, *matthieuw.github.io/go-games-number/AGoogolplexOfGoGames.pdf*. Obergrenze: Tromp/Farneback: Combinatorics of Go, *tromp.github.io/go/gostate.pdf*

[12] *en.wikipedia.org/wiki/Zero-sum_game* und *en.wikipedia.org/wiki/Minimax*

[13] Coulom: Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search, *www.remi-coulom.fr/CG2006/CG2006.pdf*

Kurz gesagt: Convolutional Neural Networks (CNN) sind Beispiele für ein neuronales Netz, das lokale Verbindungen verwendet, die besonders geschickt beim Lernen und Verarbeiten räumlich korrelierter Merkmale in Bildern sind. Das obige Bild (siehe letzte Seite) zeigt einen gelernten konvolutionären Filter, der um ein Bild herum gleitet und eine Darstellung in niedrigerer Dimension erzeugt. Typische Netzwerke enthalten Millionen solcher gelernter Parameter und können eine Vielzahl von Aufgaben in der Bildverarbeitung erfüllen.[14]

Da Convolutional Neural Networks bei Bilderkennungsaufgaben vielversprechend zu sein begannen[15] und da sich diese Netzwerke jeder Funktion annähern können,[16] begannen die Leute darüber nachzudenken, sie zur Bestimmung der Bewertungsfunktion zu verwenden und den Brettzustand als „Bild"-Eingabe an das CNN zu behandeln. Die Idee ist einfach: Bei einem bestimmten Brettzustands- und Endergebnispaar (s,r) trainiere dein CNN zur Vorhersage von r aus s – die Vorhersage von r (eingeschränkt zwischen [0,1] zu interpretieren als Gewinnwahrscheinlichkeit) stellt unseren Zustandswert dar. Mehr noch, von diesem Zustand ausgehend können wir lernen, den nächsten Zug – als Zugvorschlagsgenerator – vorherzusagen (Policy genannt).

Und so begannen die Leute, Hunderttausende von Go-Spielen herunterzuladen, die von starken Amateuren online gespielt worden waren und trainierten CNNs, Züge und Gewinnraten vorherzusagen. Allein die Agenten, die von der reinen Policy aus spielten, konnten einige der schwächeren Go-Bots übertreffen, hatten aber gegen starke MCTS-Bots immer noch ihre Schwierigkeiten.

[14] Dumoulin/Visin: Convolution arithmetic, *github.com/vdumoulin/conv_arithmetic*

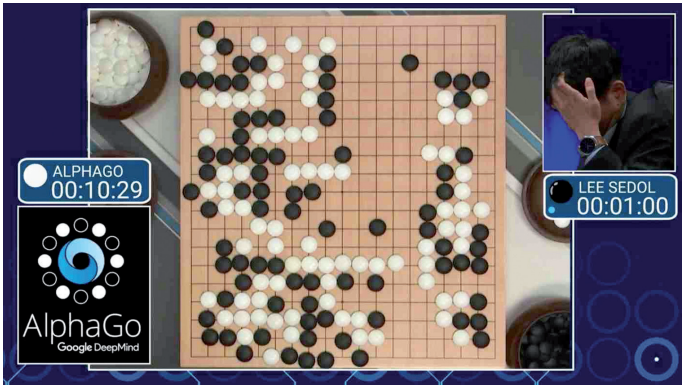
[15] Krizhevsky et al.: ImageNet Classification with Deep ConvolutionalNeural Networks, *papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf*

[16] *en.wikipedia.org/wiki/Universal_approximation_theorem*

Durch die Kombination von CNN-basierter Policy- und Wertbestimmung mit MCTS (d.h., anstelle von zufälligen Stichproben von Spielzügen gewichten wir die MCTS-Stichprobe mit der Policy aus dem CNN und anstatt einen Endzustand zu erreichen, schätzen wir den Wert des aktuellen Zustands aus dem Wertnetz), begannen diese prototypischen CNN-Bots alle anderen zu übertreffen, aber professionelle Menschen waren immer noch unerreichbar.

AlphaGo

Obwohl MCTS das Spiel der CNN-basierten Bots verbesserte, wurden die neuronalen Netze ausschließlich auf menschliche Spiele trainiert und hatten keine Möglichkeit, sich über das menschliche Wissen hinaus zu verbessern. Sie konnten die menschliche Intuition nur schwach imitieren.



Um dieses Problem zu lösen, setzte AlphaGo Spiele gegen sich selbst bzw. sogenanntes Reinforcement Learning ein, um die Policy- und Werteinschätzungen zu verbessern. Im Großen und Ganzen nehmen Reinforcement Learning Agents Aktionen in einem Umfeld vor, erhalten Belohnungen und Beobachtungen aus ihrer Umgebung und lernen, ihre Aktionen so anzupassen, dass sie zukünftige Belohnungen maximieren können. In diesem Fall ist die „Umgebung" ein simuliertes Go-Spiel und die Belohnung ist das Endergebnis des Spiels (d.h. die Belohnungen sind spärlich und werden erst nach vielen Aktionen ausgegeben).

Da der MCTS in AlphaGo für einen maximalen Wert optimiert (der die Gewinnwahrscheinlichkeit misst), kann das Netzwerk durch die Produktion und Partien gegen sich selbst weiter trainiert werden, um bessere Wertvorhersagen zu machen und, was wichtig ist, kann das Policy-Netzwerk anhand der MCTS-Suchwerte trainiert werden, d.h. wir können das Policy-CNN anhand der endgültigen Policyverteilung trainieren, die der MCTS während des Selbstspiels gefunden hat. Mit Hilfe des Reinforcement Learnings im Selbstspiel, bei dem das Policy- und Wertnetzwerk kontinuierlich trainiert wird, konnte AlphaGo die menschliche Leistung übertreffen und am Ende Lee Sedol mit 4:1 besiegen.

Die Art und Weise, in der AlphaGo vom menschlichen Spiel abweicht, ist ebenso schockierend wie die Art und Weise, in der es das nicht tut. Es gibt keinen besonderen Grund anzunehmen, dass AlphaGo überhaupt etwas wie Menschen spielen sollte – es ist nur auf Sieg optimiert und doch beginnt AlphaGo wie Menschen immer noch damit, Steine in die Ecken zu platzieren, spielt immer noch „normale" Annäherung und macht Testzüge. Dass sich AlphaGo dem menschlichen Spiel in vielen Aspekten von Go annährt ist beruhigend: Wir verstehen das Spiel nicht völlig falsch. Eine externe Intelligenz stimmt den Zügen, die wir machen, weitgehend zu – etwas an unserem Spiel ist „richtig", unsere Vorstellungen über das Spiel sind nicht völlig abwegig.

Und doch gibt es Divergenzen. AlphaGo spielt Züge und bringt Ideen hervor, die Menschen noch nie gesehen haben, die aus unserer Sicht einfach „falsch" aussehen. Es ist viel über Zug 37 in der zweiten Partie gegen Lee Sedol gesagt worden: Ein Schulterschlag der fünften Linie, der sich falsch anfühlt – nicht einer der Züge, die ein Mensch ernsthaft in Erwägung ziehen würde, den er nicht einmal auf seinem Radar hätte.

Aber diese Züge funktionieren. AlphaGo gewinnt. Er einwickelt Ideen in seinem Partien, die wir nicht verstehen.

AlphaZero

Obwohl es AlphaGo gelang, den Weltmeister Lee Sedol zu schlagen, wollte das Team von DeepMind das Reinforcement Learning weiter vorantreiben. AlphaGos Policy- und Wertnetzwerke waren über viele Tausende von menschlichen Partien erlernt worden, bevor das Reinforcement Learning im Selbstspiel gestartet wurde.

Zusätzlich zum Rohzustand des Spielbretts beinhalteten die Inputs von AlphaGo für jede Bewertung Folgendes:

Extended Data Table 2 | Input features for neural networks

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

Feature planes used by the policy network (all but last feature) and value network (all features).

Feature-Ebenen von AlphaGo[17]

In gewisser Weise gab es immer noch Go-spezifisches Wissen, mit dem AlphaGo „programmiert" worden war, und das Team wollte sehen, ob ein Bot mit null spielspezifischen Kenntnissen ähnliche Leistungen würde erbringen können.

Im Jahr 2017 veröffentlichte das Team das AlphaGo-Zero-Paper[18] mit drei wesentlichen Verbesserungen:

- Die Netzwerke werden ausschließlich durch Reinforcement Learning im Selbstspiel trainiert, ausgehend von einem Zufallsspiel ohne vorheriges Training auf der Basis menschlicher Daten.
- Nur die Positionen der schwarzen und weißen Steine werden als Eingabemerkmale für die Netzwerke verwendet.
- Die Policy- und Wertnetzwerke werden zu einem einzigen Netzwerk kombiniert, mit flachen Policy- und Value-Heads oben drauf.

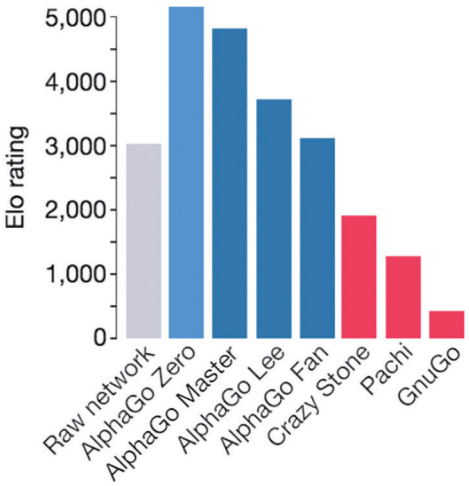
[17] Silver et al: Mastering the game of Go with deep neural networks and tree search, *www.nature.com/articles/nature16961*

[18] Silver et al: Mastering the game of Go without human knowledge, *www.nature.com/articles/nature24270*

Mit diesen Änderungen übertraf AlphaGo Zero die Leistung von AlphaGo bei weitem und gewann durch das kombinierte Policy- und Wertnetz sowie durch die Verwendung einer ResNet-ähnlichen Architektur[19] anstelle eines vollständig gefalteten Netzwerks massive Steigerungen der Trainingseffizienz.

Um die relative Spielstärke verschiedener Agenten zu messen, ist eine häufig verwendete Metrik die Elo-Bewertung. Während die vollständigen Einzelheiten des Elo-Ratings (und wie es den Handicap-Steinen entspricht) den Rahmen dieses Artikels sprengen würden, kodiert das Elo-Rating kurz gesagt die relative Gewinnwahrscheinlichkeit. Bei Verwendung der Standardskalen kodiert z.B. eine Bewertungsdifferenz von 100 Punkten eine Erwartung, dass der höher bewertete Spieler eine 64%ige Chance hat, seinen Gegner zu schlagen; wenn die Differenz 200 beträgt, liegt die Erwartung bei 76%.

Es gibt eine wunderbare Darstellung der Elo-Bewertungen verschiedener Bots aus dem AlphaGo Zero-Papier:



Elo-Vergleich verschiedener Computer-Go-Programme. Lee Sedol wäre zum Zeitpunkt seiner Partien gegen AlphaGo etwa 3500 auf dieser Skala

Beachten Sie, dass die Stärke des rohen Netzwerks (das gerade den vom Policy-Netz empfohlenen Top-Zug spielt) etwa 3.000 beträgt, während der vollständige AlphaZero-Bot (unter Verwendung des Policy-Netzes + MCTS + Wertnetz) eine Bewertung >5.000 erreicht. Dies gibt uns eine Vorstellung davon, um wie viel stärker die Baumsuche und die Werteinschätzung des rohen Netzwerks intuitiv stärker spielen lässt.

Im Zusammenhang mit unserer früheren Definition von Intelligenz als Maß für die Lerneffizienz wäre es hervorragend gewesen zu sehen, wie sich die Elo-Stärke als Funktion von Selbstspiel-Spielen von AlphaGo zu AlphaGo Zero verändert hat.

Schließlich dehnte das DeepMind-Team seine AlphaGo-Zero-Methode auf Schach und Shogi aus, entfernte alle Go-spezifischen Aspekte des Programms (z. B. keine zusätzlichen Trainingsproben aus der Symmetrie des Brettes zu generieren[20]) und veröffentlichte es erneut unter dem Namen AlphaZero.

Zero-Explosion

AlphaGo erschütterte sowohl die Go-Welt als auch die KI-Forschungsgemeinschaft, aber DeepMind ließ seine Arbeit weitgehend hinter sich und ging zu anderen Themen über. Mit nur den Forschungspapieren als Leitfaden begannen andere aber mit der Neuimplementierung von AlphaZero.

Bereits mit dem ersten veröffentlichten Papier über AlphaGo begannen viele Privatunternehmen, insbesondere in China, Südkorea und Japan (wo kommerzielle Go-Produkte realisierbar sind), AlphaGo/Zero neu zu erstellen. Während diese Bots für diejenigen hilfreich waren, die sich den Zugang leisten konnten, konnte die Go-Gemeinschaft ihre Vorteile erst dann voll ausschöpfen, als Open-Source-Bots sich verbreiteten.

Der bekannteste Open-Source-Bot ist Leela Zero,[21] eine originalgetreue Neuimplementierung von AlphaZero, die GPU-Computer aus der Cloud verwendet, um Selbstspiel-Spiele zu produzieren und das Netzwerk zu trainieren. Leela Zero trainiert seit Ende 2017 und hat bis Mai 2020 etwa 20 Millionen Partien gegen sich selbst absolviert.

Als Leela Zero und andere Bots der Öffentlichkeit für die Analyse und zum Spielen zur Verfügung standen, erlebte Go einen kulturellen

[19] He et al: Deep Residual Learning for Image Recognition, *arxiv.org/abs/1512.03385*

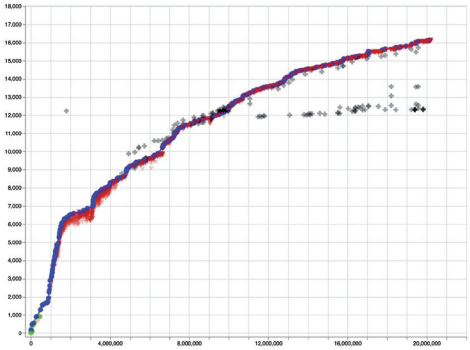
[20] en.wikipedia.org/wiki/Dihedral_group

[21] zero.sjeng.org/home

Wandel, wie es ihn noch nie zuvor gegeben hatte. Plötzlich hatte jeder Zugang zu übermenschlicher Spielberatung und konnte von einem der stärksten Spieler aller Zeiten Meinungen zu bestimmten Varianten einholen. Während AlphaZero ein Durchbruch für die KI-Gemeinschaft war, waren Leela und die Open-Source-Bots wie das wahre Geschenk des Himmels für die Go-Gemeinschaft. Anstatt einfach nur die Züge von AlphaZero nachzuahmen, konnten die Spieler diese Open-Source-Bots für eingehende Überprüfungen und Studien verwenden. Shin Jinseo bringt z. B. angeblich überall ein iPad mit Leela Zero mit, um Ideen und Varianten zu überprüfen. Als sich der Einfluss von AlphaZero und Leela Zero auf die Metaebene des Spiels durchsetzte, stellten Forscher bei Facebook fest, dass die Spieler schneller als je zuvor in der Geschichte stärker wurden.[22]

Obwohl sie eine große Ressource für die Go-Gemeinschaft darstellten, hatten diese Zero-Bots aber immer noch Probleme: Sie waren teuer im Training und brauchten Monate oder Jahre, um eine übermenschliche Spielstärke mit „normaler“ Rechenleistung zu erzielen, sie waren (zunächst) überraschend schlecht bei Treppen, teilten mit AlphaGo die Neigung, „schlaffe“ Züge zu machen, wenn sie vorne lagen, konnten nicht mit variablem Komi spielen und spielten unausgeglichen in Handicap-Partien.

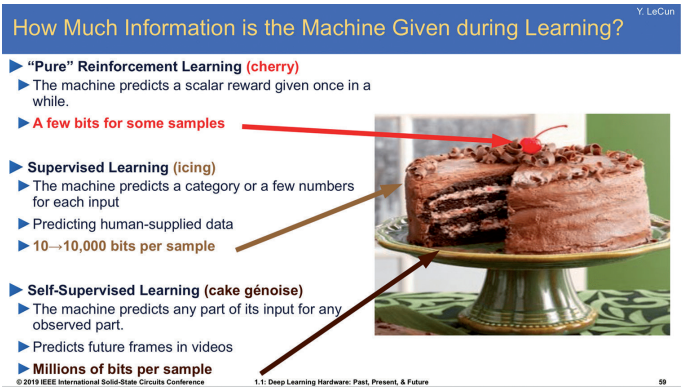
Bei einer KI-Weltmeisterschaft 2019 scheiterte Leela am Podium und verlor den 3. Platz an HanDol, einen kommerziellen Bot aus Korea, der später gegen Lee Sedol bei dessen letzter Profipartie spielen sollte. Enttäuschenderweise zerstörten die kommerziellen Bots den Open-Source-Bot Nr. 1 Leela, wahrscheinlich aufgrund der ihnen zur Verfügung stehenden weitaus größeren Rechenressourcen für das Training. Zudem ist unklar, welche algorithmischen Unterschiede, wenn überhaupt, die kommerziellen Bots gegenüber AlphaGo haben.



Leela Zero Elo-Bewertung vs. Anzahl der Spiele des Selbstspiels.[20] *Diese Selbstspiel-Elo ist allerdings etwas aufgeblasen und entspricht nicht dem zuvor gezeigten Elo-Diagramm.*

Yann LeCun's Cake of Learning

Der „Pate der KI“, Miterfinder der Convolutional Neural Networks und Turing-Preisträger Yann LeCun, möchte von seinem Kuchen erzählen.



„If intelligence is a cake, the bulk of the cake is self-supervised learning, the icing on the cake is supervised learning, and the cherry on the cake is reinforcement learning.“

Yann LeCun

Yanns Tortenmetapher soll darauf hinweisen, dass der Großteil der in den Daten enthaltenen Informationen „unstrukturiert“ ist. Beim Reinforcement Learning wird ein extrem verrauschtes Signal mit

[22] ai.facebook.com/blog/open-sourcing-new-elf-opengo-bot-and-go-research/

geringer Informationsdichte, z.B. ein Gewinn-Verlust-Signal aus einem Selbstspielspiel, genommen und dieses Lernsignal durch viele Brettzustände propagiert (z.B. Training des Wertnetzwerks zur Vorhersage der Gewinnrate aus einem gegebenen Brettzustand). Da das Signal-Rausch-Verhältnis dort so schlecht ist, ist das Reinforcement Learning extrem datenintensiv.

Supervised Learning ist etwas besser: Würden wir ein CNN aufbauen, um Bilder von Hunden zu klassifizieren, würde jedes Trainingsbeispiel aus einem Bild und einem von Menschenhand erstellten korrekten Etikett bestehen, das nur durch ein einziges Bild rückpropagiert würde. Die Informationsdichte pro Sample ist viel höher und das Label-Rauschen ist nicht vorhanden (es sei denn, der Mensch hat das Bild falsch beschriftet). Supervised Learning erfordert im Allgemeinen weit weniger Beispiele als Reinforcement Learning, um eine gute Leistung zu erzielen.

Schließlich gibt es noch das, was Yann als „selbstüberwachtes" (self-supervised) Lernen bezeichnet, bei dem „das System lernt, einen Teil seines Inputs aus anderen Teilen seines Inputs vorherzusagen".[22] Die Idee dahinter ist, dass die unstrukturierten Eingabedaten weit mehr Informationen enthalten, als jede überwachte Kennzeichnung es jemals könnte, und so führt die Suche nach Wegen, Teile der Eingaben geschickt vorherzusagen, zu einer viel besseren Lerneffizienz und ermöglicht die Verwendung eines viel größeren Satzes nicht gekennzeichneter Daten.

Ein lustiges aktuelles Beispiel für erfolgreiches selbstüberwachtes Lernen ist die monokulare Tiefenschätzung:[23]

Schätzung der monokularen Tiefe[24]

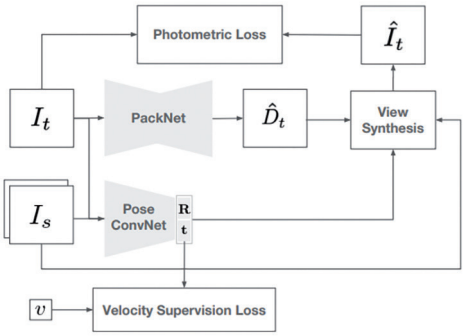
[22] Yann Lecun Twitter, *twitter.com/ylecun/status/1123235709802905600?lang=en*

[23] Godard et al: Digging Into Self-Supervised Monocular Depth Estimation, *arxiv.org/abs/1806.01260*

[24] Guizilini et al: 3D Packing for Self-Supervised Monocular Depth Estimation, *github.com/tri-ml/packnet-sfm*

Es wäre sehr nützlich, pixelgenaue Tiefenkarten aus monokularen Kamerabildern abzuschätzen. Da der Mensch nicht in der Lage ist, pixelgenaue Tiefenkarten zu beschriften, und die Erfassung von LiDAR-Daten teuer ist, können wir irgendwie ein Netzwerk zur Schätzung der Tiefe aufbauen, das nur Rohbilder als Trainingsbeispiele verwendet?

Es stellt sich heraus, ja! Durch Ausnutzung der von Bild zu Bild konsistenten Szenengeometrie und der Tatsache, dass aufeinander folgende Videobilder viele gleiche Objekte enthalten, können wir ein Netzwerk eine Tiefe und eine Kameraposition erraten lassen und einige clevere differenzierbare geometrische Transformationen verwenden, um diese Szene aus dem erratenen Blickwinkel eines nachfolgenden Videobildes zu rekonstruieren – und den Konsistenzverlust aus den rekonstruierten und tatsächlichen Bildern zurückpropagieren, um szenenkonsistente Tiefen- und Einstellungsschätzungen zu erzwingen. Es stellt sich heraus, dass die einzige Möglichkeit für das Netzwerk, die konsistente Tiefe und Kameraeinstellung von Bild zu Bild zu erraten, darin besteht, die korrekte Tiefe und Position zu erraten. Mit dieser Methode können Netzwerke lernen, genaue Tiefenkarten mit nur rohem Videoinput vorherzusagen.



Selbstüberwachtes monokulares Netzwerkdiagramm

KataGo

Ende 2017 begann David „Lightvector" Wu, ein Go-Enthusiast und Forscher, mit der Arbeit an einem Bot im AlphaGo-Stil für persönliche Experimente. Denjenigen, die an den Details der Entwicklung interessiert sind, empfehle ich dringend,

sich das Repository auf Github anzuschauen (*github.com/lightvector/GoNN*), um seine Experimente nachzuverfolgen. Das Projekt entwickelte sich zu einer echten Forschungsarbeit und wurde letztlich zu KataGo.

Wie AlphaGo verwendet auch KataGo ein CNN zur Schätzung der Gewinnrate (Wert) und der Zugauswahl (Policy), aber es verzichtet auf einen Teil der Zero-Methodik, bei der Go-spezifische Informationen nicht einbezogen werden, sondern stattdessen relevante Merkmale wie Treppe und Freiheitsstatus als Input für das Netzwerk einbezogen werden. Insbesondere wird b = Brettbreite ein b×b×18 Tensor von …

Kanäle	Feature
1	Stellung ist auf dem Brett
2	Stellung hat {eigenen,gegnerischen} Stein
3	Stellung hat Stein mit {1,2,3} Freiheiten
1	Zug illegal wegen Ko/Superko
5	die letzten 5 Zug-Orte
3	in Treppe fangbare Steine vor {0,1,2} Zügen
1	Zug fängt den Gegner in der Treppe
2	{eigene, gegnerische} Fläche ist Pass-lebendig

… wird als Eingabe an das CNN weitergeleitet, zusammen mit einem zusätzlichen Eingabe-Vektor einiger globaler Zustandseigenschaften einschließlich Ko- und Komi-Details.

KataGo nimmt eine Reihe scheinbar kleiner Änderungen am AlphaGo/Zero-System vor, die sich zu enormen Effizienzsteigerungen beim Lernen summieren und die Verbesserungen der Benutzerfreundlichkeit für die Go-Gemeinschaft mit sich bringen.

Wie AlphaGo wird auch KataGo von Grund auf durch Reinforcement Learning im Selbstspiel trainiert. Es gibt vier wesentliche Verbesserungen der Lerneffizienz:

1. Playout-Randomisierungsbegrezung:
Wie im KataGo-Papier[25] angemerkt, besteht eine „Spannung zwischen Policy- und Werttraining [... denn] der Spielergebnis-Zielwert ist stark datenbeschränkt, mit nur einem verrauschten binären Ergebnis pro ganzer Partie", während das optimale Policy-Training etwa 800 MCTS-Playouts pro Zug verwenden würde. Mit anderen Worten, das Wertnetz wünscht sich, dass mehr Partien schneller gespielt werden, aber das Policy-Netz wünscht sich, dass das MCTS während des Selbstspiels tiefer geht, um bessere Policy-Ziele zu erhalten, so dass zwischen diesen beiden Zielen aufgrund der begrenzten Berechnung Spannungen bestehen. Um dieses Problem zu lösen, führt KataGo während des Selbstspiels gelegentlich eine „vollständige Suche" von 600 Playouts zur Zugauswahl durch, verwendet aber meistens nur 100 Playouts, um Partien schneller zu beenden. Nur die Züge der „Vollsuche" werden zum Training des Policy-Netzwerks verwendet, aber da es mehr Spielergebnisse gibt, verfügt das Wertnetz über mehr Trainingsbeispiele.

2. Erzwungene Playouts und Strategiezielbeschneidung:
Beim Verstärkungslernen gibt es einen klassischen Kompromiss zwischen Erkundung und Ausbeutung: Soll das erlernte Wissen genutzt werden, um optimale Züge zu spielen oder sollten scheinbar nicht-optimale Züge erforscht werden, um neue Strategien zu entdecken? KataGo versucht, dieses Problem mit erzwungenen Playouts zu lösen, bei denen jedes Kind der Wurzel, das Playouts erhalten hat, eine Mindestanzahl von Playouts erhält. Weitere Einzelheiten können im KataGo-Paper nachgelesen werden.

3. Globales Pooling:
Eine relativ einfache Verbesserung zeigt sich in KataGo durch die Einführung gelegentlicher globaler Pooling-Schichten, so dass das Netzwerk Bereiche auf dem Brett konditionieren kann, die außerhalb des Wahrnehmungsradius der Faltungsschichten liegen können. Experimente mit KataGo zeigten, dass dies spätere Trainingsphasen erheblich verbessert und, „da Go explizit nicht-lokale Taktiken (Ko) enthält, ist dies nicht überraschend".

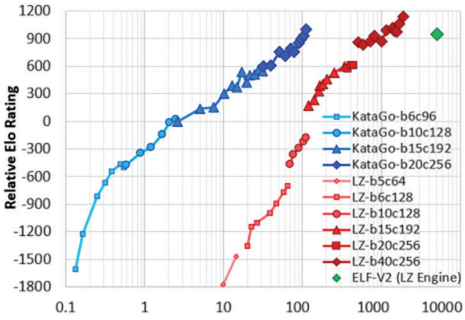
4. Policy-Hilfsziele:
Ich denke, dies ist die interessanteste Änderung in KataGo, da sie einige Gemeinsamkeiten mit Ideen wie selbstüberwachtem Lernen aufweist: das Training zusätzlicher Policy-Ziele. Typischerweise sagen Bots im AlphaZero-Stil nur die Policy und den Wert voraus und verwenden die MCTS-Suche bzw. das Endergebnis des Spiels

[25] Wu: Accelerating Self-Play Learning in Go, *arxiv.org/abs/1902.10565*

als Label. Ausgehend von der Idee aus LeCuns Folie, dass das Lernen durch das Hinzufügen von mehr Trainingszielen (in diesem Fall ganze Teile der Eingabedaten) verbessert werden kann, versucht KataGo, eine größere Anzahl von Spielergebnissen als nur den Wert vorherzusagen. Insbesondere sagt KataGo auch die endgültige Gebietskontrolle, die Endergebnisdifferenz und von jedem Brett den nächsten Zug des Gegners voraus. Zitat aus dem Papier:

Es mag überraschend sein, dass diese Ziele auch über die frühesten Stadien hinaus helfen. Wir bieten eine Intuition an: Denken Sie an die Aufgabe der Aktualisierung nach einer Partie, die in erster Linie aufgrund einer Fehleinschätzung einer bestimmten Region des Spielbretts verloren wurde. Mit nur einem binären Endergebnis kann das neuronale Netz nur „raten", welcher Aspekt der Brettstellung den Verlust verursacht hat. Im Gegensatz dazu erhält das neuronale Netz bei einem Besitzziel eine direkte Rückmeldung darüber, welcher Bereich des Bretts falsch vorhergesagt wurde, wobei große Fehler und Gradienten auf den falsch vorhergesagten Bereich beschränkt sind. Das neuronale Netz sollte daher weniger Stichproben benötigen, um eine korrekte Zuweisung und Aktualisierung durchführen zu können.

Als Ergebnis dieser Verbesserungen übertrifft KataGo Leela Zero und den ELF-Bot von Facebook bei der Lerneffizienz massiv. KataGo erreicht eine Verbesserung der Trainingseffizienz um den Faktor fünfzig im Vergleich zu ELF:

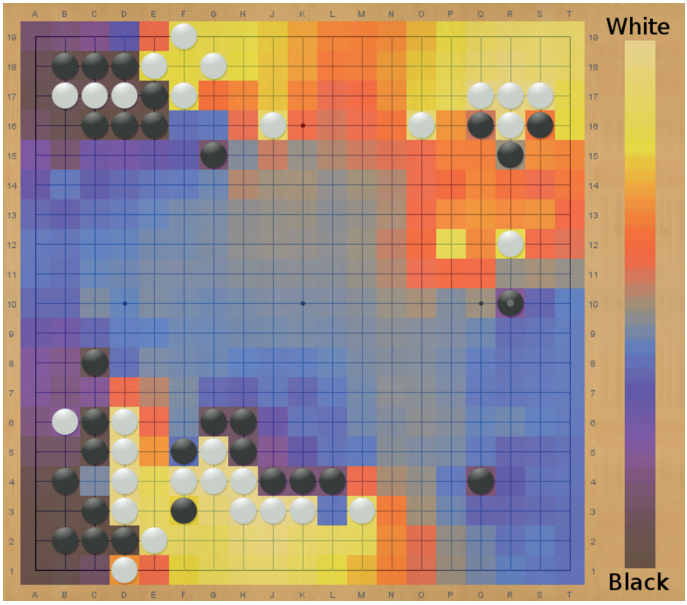


Relative Elo-Bewertung vs. Selbstspielkosten in Milliarden äquivalenter Abfragen von 20 Blöcken × 256 Kanälen (logarithmische Skala)

Zusätzlich zu diesen Verbesserungen optimiert KataGo auch direkt für eine maximale Punktzahl (mit einigen Vorbehalten) und eliminiert weitgehend die „nachlässigen" Züge, die man bei anderen Bots im Zero-Stil findet. KataGo spielt auch Handicap-Spiele gegen schwächere Versionen von sich selbst während des Trainings, spielt auf mehreren Brettgrößen und mit variablen Komi und Regelsätzen, so dass es unter Permutationen dieser Spieleinstellungen flexibel ist.

Mit all diesen zusätzlichen Funktionen stellt KataGo das bisher nützlichste Analysewerkzeug für Go dar, das den Spielern einen besseren Einblick in die Gedankenwelt eines übermenschlichen Go-Agenten gibt.

KataGo ist jetzt wahrscheinlich der stärkste Open-Source-Go-Bot auf dem Markt und



Visualisierung von Gebietsvorhersagen von KataGo

hat kürzlich die CGOS-Rangliste[26] in allen Brettgrößen angeführt.

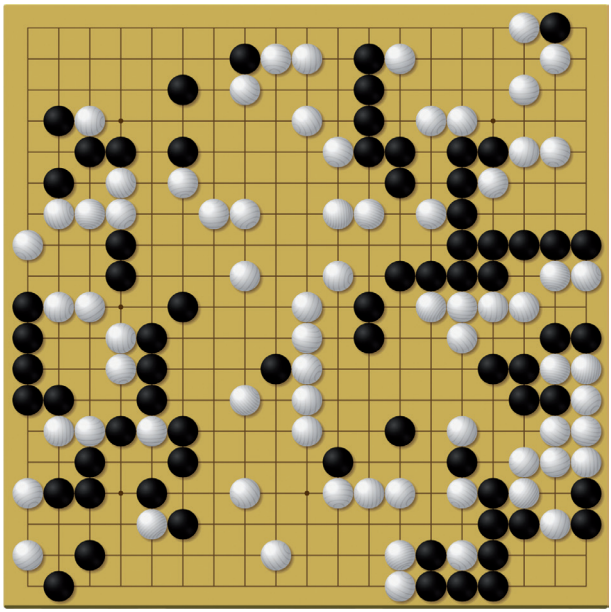
Ich empfehle allen Interessierten wärmstens, sich die Originalarbeit zu KataGo anzuschauen – es ist eine äußerst zugängliche Lektüre.

Zurückgeben

Während es Spaß macht zuzusehen, wie die Elo-Werte der Top-Bots unaufhaltsam nach oben klettern, liegt der wahre Nutzen einer starken Go-KI in dem, was sie den Menschen zurückgeben können. Go-KI ist eines der ersten Beispiele für übermenschliche KI-Agenten, die es in der Welt bereits gibt und die von echten Menschen benutzt werden, um ein Spiel, das sie lieben, besser zu verstehen.

Das *Igo Hatsuyoron*[27] ist eine Sammlung von Go-Problemen von vor etwa 300 Jahren, von denen ein Großteil von dem damals stärksten Spieler Japans geschaffen wurde. Das 120. Problem in der Sammlung wird oft als das herausforderndste Go-Problem der Welt angesehen – es handelt sich um ein Ganzbrett-Problem, das den Leser auffordert, das Spiel für beide Seiten optimal zu beenden und den Sieger und die Punktzahl zu bestimmen. Seit hunderten von Jahren haben Go-Spieler versucht, es zu lösen, aber es besteht immer noch Ungewissheit über seine wahre Lösung. Die Tesujis und Semeais, die in der scheinbar richtigen Lösung auftauchen, sind höchst unintuitiv und komplex und konnten bisher noch nicht endgültig optimal sortiert werden.

Ende 2019 wurde KataGo speziell trainiert, um dieses Problem zu lösen. Das zu diesem Zeitpunkt stärkste KataGo-Netzwerk wurde gebracht, viele Spiele von der Startposition des Problems gegen sich selbst bis zu einem Endzustand zu spielen und dabei viele der seltsamen und komplexen Formen zu entdecken, die Menschen bei diesem Problem gefunden hatten. Nun war KataGo in der Lage, neue Züge entlang der Lösungshauptvariante vorzuschlagen, einfachere Widerlegungen alter menschlicher Ideen zu finden und schließlich ein anderes Ergebnis vorauszusagen als das, was man für das richtige Ergebnis gehalten hatte.[26]



Igo Hatsuyoron 120 – Schwarz am Zug. Was ist das optimale Ergebnis?

KataGo war in der Lage, eine neue Perspektive auf dieses jahrhundertealte Problem zu entwickeln und der Go-Gemeinschaft Wissen zurückzugeben. Wie Bots wie diese weiterhin Bewertungen und neue Einsichten zu menschlichen Probleme beisteuern werden, ist eine offene Frage und ich bin gespannt, wie sie in den kommenden Jahren beantwortet wird. Wir treten in eine Ära ein, in der Computer neue Einsichten zur menschlichen Wissensbasis hinzufügen.

Zukunft

In einem kürzlich geführten Interview sagte der leitende Forscher von AlphaGo, David Silver, dass er erwartet, dass sich die Bots im AlphaZero-Stil in den nächsten 100 Jahren weiter verbessern werden, dass die Spielstärkehorizont von Go immer noch außer Sicht ist. KataGo gibt ein Bild davon,

[26] *www.yss-aya.com/cgos/19x19/bayes.html*
[27] *en.wikipedia.org/wiki/Igo_Hatsuyoron*
[28] *blog.janestreet.com/deep-learning-the-hardest-go-problem-in-the-world/*

wie sich die Verbesserungen weiter fortsetzen werden und wie dabei ein Mehrwert für die menschlichen Spieler geschaffen werden kann.

Um etwas mehr darüber zu erfahren, wohin sich die Dinge für die Zukunft von Go entwickeln könnten, wandte ich mich an den KataGo-Autor David Wu, um eine Perspektive zu erhalten:

Vielleicht sogar noch mehr als die Steigerung der Spielstärke war es spannend zu beobachten, wie KataGo seinen Nutzen als Analysewerkzeug verbessert hat, z. B. durch die Schätzung von Spielständen und die Vorhersage von Gebietsbesitz. Konzentriert sich die Zukunft von KataGo mehr darauf, den Spielern zusätzliche Ebenen der Interpretierbarkeit und des Analyse-Nutzens zu bieten oder sind Verbesserungen der Spielstärke und der Trainingseffizienz Ihr Hauptaugenmerk?

Wu: Ich interessiere mich für beides! Aber Trainingseffizienz und Spielstärke sind vor allem deshalb wichtig, weil viele der netten Experimente, die man machen möchte, um die Interpretierbarkeit und den Analyse-Nutzen zu erhöhen, von Dingen abhängen, die grundlegend bei der Modellierung eingeübt werden müssen. Zum Beispiel, wenn Sie das Ergebnis vorhersagen wollen? Der einzige Weg ist, wenn das neuronale Netz darauf trainiert wurde, den Spielstand vorherzusagen, und es gibt viele Ansätze, das zu versuchen. Vielleicht möchten Sie, dass der Bot in der Lage ist, Ihnen den Status einer Gruppe mitzuteilen? Auch dafür müssen Sie einen Weg finden, wie Sie das trainieren können. Vielleicht möchten Sie, dass das Programm ein gewisses Maß an „Unsicherheit" im Vergleich zu „Zuversicht" in Bezug auf seine eigene Bewertung meldet? Es gibt Möglichkeiten, wie Sie versuchen können, das im Nachhinein hinzuzufügen, aber auch hier könnten Sie bessere Ergebnisse erzielen, wenn es von Anfang an in das Training eingebaut wäre.

Wenn das Training effizient ist, erweitert es Ihre Fähigkeit, viele Experimente für solche Dinge durchzuführen (wie es KataGo mit Ergebnisschätzung, japanischen Regeln und Handicap-Spieltraining tat) – je schneller Sie in jedem neuen Experiment auf ein starkes Niveau trainieren können, desto interessantere Experimente können Sie machen. Aber natürlich macht es auch Spaß, zu versuchen, der Stärkste zu sein und dem optimalen Spiel immer näher zu kommen.

Das Tempo des Experimentierens könnte sich in Zukunft verlangsamen, da einige von KataGos anfänglicher Unterstützung und Vorkehrungen im Hinblick auf die Rechenleistung jetzt an ihre Grenze kommen, aber ich hoffe, dass auch in Zukunft mehr Forschung möglich sein wird. Ich hoffe, dass KataGo nicht das letzte Wort ist! Es sind sehr wahrscheinlich noch viele weitere Verbesserungen möglich, daher würde ich mich freuen, wenn in den kommenden Jahren andere Projekte Wege finden würden, KataGos Ideen und Effizienzverbesserungen zu nutzen, um noch weitere Verbesserungen zu finden und noch weiter zu gehen. KataGo ist Open Source, gerade weil der Austausch von Techniken und Forschungsergebnissen, anstatt sie geheim zu halten, den Stand der Technik für alle verbessert.

Top-Profispieler sagten immer, dass nicht einmal Gott ihnen vier Steine geben könne. Wenn sich KataGo dieser Schwelle nähert, glauben Sie, dass das Training im Zero-Style auf unbestimmte Zeit weiter skalieren wird? Sowohl in Bezug auf die Spielstärke als auch in Bezug auf die Art und Weise, wie aktuelle blinde Flecken gefunden und behandelt werden: Wir haben zum Beispiel gesehen, dass KataGo das Fliegender-Dolch-Joseki (von Mi Yuting) nicht so gut versteht wie Leela Zero und dass er gegenüber bestimmten „scharfen" Positionen blind bleibt. Glauben Sie, dass Selbstspiel ausreicht, um diese Probleme langfristig zu überwinden? Oder müssen andere Lösungen gefunden werden? Ist es möglich, andere externe Go-Agenten in den Prozess der Generierung von Trainingsspielen einzubeziehen?

Wu: Ich denke, die AlphaZero-Selbstlernschleife mit MCTS ist bei solchen Dingen nicht das letzte Wort. Blinde Flecken sind nur die sichtbarsten Mängel, aber es gibt einige technische und theoretische Details, in die man sich vertiefen kann und die deutlich machen, dass es einige praktische Probleme damit gibt, wie Forschung und Zugfindung in dieser Schleife funktionieren, einige grundlegende theoretische Mängel, die eine Diskrepanz zwischen dem Training und der Nutzung des neuronalen Netzes beinhalten – und es gibt auch einige grundlegende „fehlende" Fähigkeiten in aktuellen Bots im Hinblick auf die effektive Nutzung der Suche.

Die Einbindung externer Daten ist vielleicht nur für eines dieser Probleme ein möglicher Patch, und es ist nur ein Patch – es wäre viel cooler, Wege zu finden, sie auf einer grundlegenderen Ebene zu beheben, als sie nachträglich zu flicken.

Ihr Beitrag konzentrierte sich sowohl auf Verbesserungen beim Selbstspiel, die darauf abzielen, ein besseres Gleichgewicht zwischen Erkundung und Ausbeutung beim Training zu finden, als auch auf Designmöglichkeiten für neuronale Architekturen wie das Hinzufügen spezifischer Schichten und zusätzlicher Trainingsziele. Welcher Aspekt zwischen Rohdatenberechnung, neuronaler Architektur und dem verwendeten Suchalgorithmus hat Ihrer Meinung nach das größte Potenzial zur Verbesserung der KI-Leistung in Go und in anderen Umgebungen?

Wu: Anstatt mich von oben nach unten auf weite Bereiche zu konzentrieren und zu erraten, was „mehr Potenzial" haben wird, halte ich es für praktischer, mich auf bestimmte Probleme oder Mängel oder Ideen zu konzentrieren – und von dort aus weiterzuarbeiten. Viele der derzeitigen Techniken wurden einfach durch die Beobachtung spezifischer Mängel im „Vanilla-AlphaZero-Selbstspieltraining" entwickelt und dann mit potenziellen Lösungen experimentiert, unabhängig davon, ob es sich dabei um das neuronale Netz oder die Suche oder etwas anderes handelte. Um in der Praxis Fortschritte zu erzielen, möchte man sich meist auf Probleme konzentrieren und dann mit den (möglicherweise völlig neuen!) Methoden spielen, die sie lösen könnten, und nicht umgekehrt.

Mit anderen Worten: Wenn man versucht, Probleme zu lösen, möchte man oft jedes Problem angehen, indem man das richtige Werkzeug für die Aufgabe herausfindet, anstatt mit dem Werkzeug zu beginnen, das man verwenden möchte und dann nach dem Problem zu suchen, um es damit zu lösen. Obwohl Letzteres und das Brainstorming, ob ein Werkzeug breiter auf andere Dinge angewendet werden kann, definitiv etwas ist, was man manchmal tun möchte!

Gibt es Pläne, einen von der Gemeinschaft beigesteuerten Computerpool für das Training von KataGo einzurichten, wenn der aktuelle Trainingsdurchlauf beendet ist?

Wu: Es gibt einige Versuche und Pläne und ich hoffe, sie können auf den Weg gebracht werden. Selbst wenn nicht, hat KataGo bereits ein großes Ziel erreicht, und ich hoffe, dass es einen großen Einfluss auf künftige Nachfolger oder andere davon inspirierte Projekte hat. Aber ja, ich hoffe, dass es bald einen verteilten Selbsttrainingsdurchlauf geben wird.

Gibt es Pläne, KataGo an einem der internationalen Go-KI-Wettbewerbe teilnehmen zu lassen?

Wu: Es gibt aktuell keine Pläne – obwohl es interessant sein könnte, nur zum Spaß an einem Wettbewerb teilzunehmen. Offensichtlich ist alles, was mit Reisen zu tun hat und nicht rein online ist, im Moment ein Problem durch Covid-19. Es gibt zwar keinen Grund, warum ein reiner Online-Wettbewerb nicht organisiert werden könnte, aber viele der vergangenen Wettbewerbe waren mit Reisen verbunden, und mir sind keine großen reinen Online-Turniere bekannt, die in diesem Jahr organisiert worden wären. Falls es welche gibt, würde es mich interessieren, von ihnen zu hören.

Nebenbei bemerkt schien es mir immer amüsant zu sein, dass die Betonung auf Wettbewerben liegt, bei denen ein Bot nur eine Gesamtsumme von, sagen wir, 5 oder 10 oder 20 Partien gegen andere Gegner spielt, wobei jeder KataGo oder Leela Zero oder was auch immer herunterladen kann und Hunderte, wenn nicht Tausende von Partien ausführen und weitaus mehr Daten sammeln kann, um die Bots zu vergleichen und sogar Muster in ihren individuellen Stilen, Stärken und Schwächen zu finden und so weiter. Erschwerend kommt hinzu, dass (zumindest nach meiner begrenzten Erinnerung) frühere Wettbewerbe oft Dinge wie unbegrenzte Hardware erlaubten – alles, was sich der Einzelne oder das Team oder das Unternehmen leisten konnte, was es schwierig machte, die Ergebnisse zu beurteilen. Trotzdem denke ich, dass sie gute Unterhaltung und eine gute zentrale Koordinationsstelle für Kommentare und Zuschauer sind.

Obwohl die Bots die Menschen an Spielstärke weit überholt haben, haben sie uns nicht zurückgelassen: Jetzt geben sie uns etwas zurück und tragen zu einem besseren Verständnis und mehr Freude am Go bei. Dank Projekten wie KataGo erhalten wir Einblicke in Geheimnisse, die noch immer in diesem Jahrtausende alten Spiel verborgen liegen. Auch wenn wir die wahre Bedeutung von Go oder von Intelligenz noch nicht verstehen, macht uns die Jagd danach weiterhin Freude!