hochschule mannheim

Entwicklung eines Efficiently Updatable Neural Network (NNUE) zur Evaluation von Schach Positionen

Marvin Karhan

Bachelor-Thesis

zur Erlangung des akademischen Grades Bachelor of Science (B.Sc.)

Studiengang Informatik

Fakultät für Informatik

Hochschule Mannheim

31.08.2022

Betreuer

Prof. Jörn Fischer, Hochschule Mannheim

Prof. Thomas Ihme, Hochschule Mannheim

Karhan, Marvin:
Entwicklung eines NNUE zur Evaluation von Schach Positionen / Marvin Karhan. –
Bachelor-Thesis, Mannheim: Hochschule Mannheim, 2022. 10 Seiten.

Karhan, Marvin:
Development of an NNUE for the Evaluation of Chess Positions / Marvin Karhan. –
Bachelor Thesis, Mannheim: University of Applied Sciences Mannheim, 2022. 10 pages.

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich bin damit einverstanden, dass meine Arbeit veröffentlicht wird, d. h. dass die Arbeit elektronisch gespeichert, in andere Formate konvertiert, auf den Servern der Hochschule Mannheim öffentlich zugänglich gemacht und über das Internet verbreitet werden darf.

Mannheim, 31.08.2022

Marvin Karhan

Abstract

Entwicklung eines NNUE zur Evaluation von Schach Positionen

Jemand musste Josef K. verleumdet haben, denn ohne dass er etwas Böses getan hätte, wurde er eines Morgens verhaftet. Wie ein Hund! sagte er, es war, als sollte die Scham ihn überleben. Als Gregor Samsa eines Morgens aus unruhigen Träumen erwachte, fand er sich in seinem Bett zu einem ungeheueren Ungeziefer verwandelt. Und es war ihnen wie eine Bestätigung ihrer neuen Träume und guten Absichten, als am Ziele ihrer Fahrt die Tochter als erste sich erhob und ihren jungen Körper dehnte. Es ist ein eigentümlicher Apparat, sagte der Offizier zu dem Forschungsreisenden und überblickte mit einem gewissermaßen bewundernden Blick den ihm doch wohl bekannten Apparat. Sie hätten noch ins Boot springen können, aber der Reisende hob ein schweres, geknotetes Tau vom Boden, drohte ihnen damit und hielt sie dadurch von dem Sprunge ab. In den letzten Jahrzehnten ist das Interesse an Künstlern sehr zurückgegangen. Aber sie überwanden sich, umdrängten den Käfig und wollten sich gar nicht fortrühren.

Development of an NNUE for the Evaluation of Chess Positions

The European languages are members of the same family. Their separate existence is a myth. For science, music, sport, etc, Europe uses the same vocabulary. The languages only differ in their grammar, their pronunciation and their most common words. Everyone realizes why a new common language would be desirable: one could refuse to pay expensive translators. To achieve this, it would be necessary to have uniform grammar, pronunciation and more common words. If several languages coalesce, the grammar of the resulting language is more simple and regular than that of the individual languages. The new common language will be more simple and regular than the existing European languages. It will be as simple as Occidental; in fact, it will be Occidental. To an English person, it will seem like simplified English, as a skeptical Cambridge friend of mine told me what Occidental is.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Computer Schach ist ein viel betrachtetes Thema. Schon Alan Turing und Claude Shannon hab sich damit befasst [1], [2].

Dieses Konzept wurde erstmals 2018 von Nasu in seinem Japanischem paper vorgestellt [3]. Shogi ist die japanische Variante des Schachspiels. Shogi unterscheidet sich in einigen Punkten von herkömmlichem Schach, es hat unter anderem eine andere Spielfeldgröße und erlaubt es geschlagene Figuren wieder einzusetzen. Trotzdem eignet sich Nasus Ansatz für traditionelles Schach, da die Zuggenerierung sowie die Evaluation ähnlich ist. Außerdem gibt es in beiden Varianten einen König was praktisch für die Auswahl eines passenden Feature Sets ist, wie in Unterabschnitt 4.1.1 genauer erläutert ist.

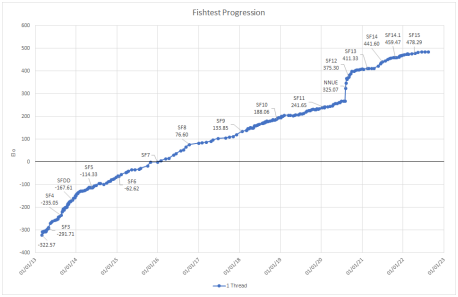

Abbildung 1.1

Nur zwei Jahre später zeigte ein port des Konzepts starke Verbesserungen in dem Schachcomputer Stockfish, der sich durch NNUE um mehr als 80 elo verbessern konnte [4]. Die

Ein Schachcomputer besteht aus drei Teilen: Suche, Zuggenerierung (Boardrepräsentation) und Evaluation [5]. Als Basis für diese Arbeit wird ein simpler Schach-

computer, der in dem Modul Künstliche Intelligenz für autonome Systeme (KIS) entwickelt wurde, verwendet. Dieser Schachcomputer verfügt über eine simple Suche und eine hand-crafted evaluation (HCE) [6]. Gegenstand dieser Arbeit ist es die HCE durch ein NNUE zu ersetzen.

Ziel dieser Arbeit ist es ein NNUE zu trainieren und in einen bestehenden Schachcomputer, der 2021 im Modul KIS entwickelt wurde, einzubinden. Die Erstellung neuer Eingabedaten für NNUEs ist nicht teil dieser Arbeit.

Kapitel 2

Grundlagen

In diesem Kapitel wird das Wissen vermittelt, welches benötigt wird um zu Verstehen, wie NNUEs im Rahmen von Schachcomputern funktionieren. Zuerst wird die Evaluation, wie sie in herkömmlichen Schachcomputern funktioniert erklärt, auch HCE genannt.

2.1 Hand-crafted Evaluation

Die Evaluation einer Schachposition ist eine heuristische Methode der Position einen numerischen Wert zuzuordnen. Hätten wir unendliche Ressourcen könnten wir aus jeder Position alle mögliche Zugfolgen sehen und der Positionen einer der drei Werte: -1 (Verlust), 0 (remis), 1 (Gewinn) geben. In der Realität ist es nicht möglich den exakten Wert der Stellung zu kennen, deshalb wird in der HCE versucht anhand von Menschen festgelegten Kriterien einen Wert der Position zuzuordnen. Vor der Verbreitung von neuronale Netze (NNs), war HCE die einzige Form der Positions-Evaluation.

2.2 Neuronale Netze

künstliche neuronale Netze (KNNs) oder einfach NNs genannt, sind Computer Systeme, die dem biologischen Vorbild des Gehirns nachempfunden sind. In Abbildung Abbildung 2.1 ist ein einfaches neuronales Netz zu sehen. Es besteht aus

NN bestehen aus

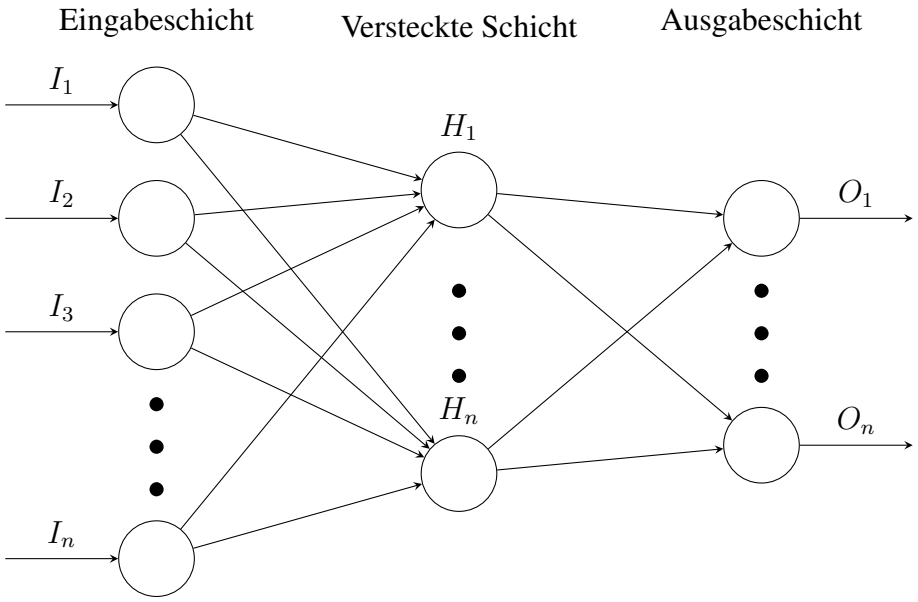

Abbildung 2.1: Ein exemplarisches neuronales Netz mit

2.2.1 Das Neuron

Das Neuron ist der elementare Bestandteil eines NNs. Um Neuronen eines NNs zu verstehen, schauen wi

2.2.2 Backpropagation

Als Backpropagation wird das Verfahren der Fehlerrückführung beschrieben. Es gehört zu der Familie der überwachten Lernverfahren

2.2.3 Convolutional Neural Networks

2.2.4 Loss funcntions

2.2.5 Quantisierung

Quantisierung ist ein Signalverarbeitungsverfahren, bei welchem Eingabewerte auf eine vorher festgelegte kleinere Menge von Ausgabewerten abgebildet wird. Ein simples Beispiel für Quantisierung ist das Abbilden von rationalen Zahlen auf ganze Zahlen, hierfür müssen die rationalen Zahlen zu der nächsten ganzen Zahl gerundet werden. Im Bereich der Informatik werden für Gleitkomma Eingabewerte oft Fest-

kommazahlen oder Ganzzahlen als Ausgabewerte gewählt [7]. Egal wie und welche die Quantisierung stattfindet, das Ziel ist es weniger Speicherkapazität und weniger Berechnungszeit zu benötigen mit minimalen Präzisionsverlust.

Dieses Verfahren eignet sich gut für Anwendungsgebiete mit wenig Speicher- und Rechenkapazität, wie beispielsweise der Einsatz von NNs bei Mobilgeräten [7], [8]. Der Grund dafür ist zweierlei. Erstens sorgt Quantisierung dafür, dass weniger Platz im cache der CPU gebraucht wird, wodurch weniger Schreib- und Lesezugriffe ausgeführt werden und somit die Berechnung schneller ist. Zweitens ermöglicht die Abbildung auf ganzzahlige Typen einen Performance-Gewinn, durch die Verwendung von Prozessor internen Recheneinheiten die beispielsweise Single instruction, multiple data (SIMD) unterstützen.

Das Problem der Quantisierung ist das Einbauen von „Fehlern“. Bei NNs wird oft von Fehler-Kumulierung gesprochen, da bei der Aktivierung eines NNs in jedem Quantisierten Neuron der Fehler wächst **Park2018**.

2.3 SIMD

SIMD

2.3.1 Memory Alignment

Kapitel 3

Verwandte Arbeiten

Kapitel 4

NNUE

Ziel dies Kapitels ist es,

Kapitel Abschnitt 2.1 zeigt wie die herkömmliche Art und Weise der Positions-Evaluation funktioniert. Nach kurzer Überlegung wird aber klar, dass die HCE nur so gut sein kann wie die Schachspieler die sie Entwickeln. Natürlich können die darin verwendeten Parameter durch Optimierungsalgorithmen wie genetische Algorithmen oder Simulated Annealing maximiert werden, letztendlich bleibt der limitierende Faktor das Spielverständnis der Entwickler.

4.1 Architektur

4.1.1 Feature Set

4.1.2 Eingabeschicht

4.1.3 Versteckte Schicht

4.1.4 Ausgabeschicht

4.2 Training

4.2.1 Eingabedaten

Die Erzeugung der Eingabedaten ist nicht teil dieser Arbeit. Jedoch ist es wichtig zu wissen wie die Eingabedaten generiert werden und wie sie in den Trainer geladen werden, um zu verstehen, wie das neuronale Netz lernt. Im Training für diese Arbeit wurden drei verscheiden generierte Datensätze verwendet. Diese Datensätze wur-

den von Stockfish für das Training der neusten Variante ihres NNUEs verwendet **StockfishNewestNetJul04**. Sie

4.3 Integration in einen Schachcomputer

4.3.1 Eingabeschicht

4.3.2 Versteckte Schicht

- simple -> quantization

Kapitel 5

Ergebnisse

5.1 Verbesserungen

Kapitel 6

Fazit und Ausblick

Wie in Kapitel 5 festgestellt, hat das NNUE hat nicht die erwarteten Ergebnisse geliefert. Dafür kann es mehrere Gründe geben:

Die Implementierung des Netzes in dem Schachcomputer kann Fehler enthalten. Die rekursive Art und Weise, der Zugsuche macht es nicht einfach mögliche Fehlerquellen zu identifizieren. In einer zeitlich limitierten Anstrengung Fehler in der Implementierung zu finden, wurde das Einlesen der Gewichte und Bias, sowie das Verhältnis zwischen updates und refreshes des Akkumulators überprüft. Es konnten keine Unregelmäßigkeiten festgestellt werden. Fehler an anderen stellen der Implementierung sind wahrscheinlich und können ausschlaggebend für die vorzufindenden Ergebnisse sein. Leider waren weitere Tests im Rahmen dieser Arbeit nicht möglich.

Ein weiter Grund für eine nicht vorhandene Steigerung der Spielstärke kann außerhalb der Evaluation liegen. Die zugrundeliegende Suche von des im Modul KIS entwickelten Schachcomputers ist nur minimal optimiert. Sie besteht aus einem Negamax Depth-First Suchalgorithmus [9] mit Move Ordering, Transposition Tables, Quiescence Search und Iterative Deepening. Es fehlen viele weit verbreitete pruning Techniken wie unter anderem: razoring [10, S. 123-128]

Abkürzungsverzeichnis

NNUE Efficiently Updatable Neural Network
SIMD Single instruction, multiple data
HCE hand-crafted evaluation
KNN künstliches neuronales Netz
NN neuronales Netz
KIS Künstliche Intelligenz für autonome Systeme

Tabellenverzeichnis

Abbildungsverzeichnis

Literatur

[1] Alan Turing, *Faster Than Thought - A Symposium on Digital Computing Machines*. London: Sir Isaac Pitman & Sons, 1953, 1953.

[2] Claude E. Shannon, „XXII. Programming a computer for playing chess“, *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Jg. 41, Nr. 314, S. 256–275, März 1950. DOI: 10.1080/14786445008521796.

[3] Yu Nasu. „NNUE: Efficiently Updatable Neural-Network-based Evaluation Functions for Computer Shogi“. (2018), Adresse: https://www.apply.computer-shogi.org/wcsc28/appeal/the_end_of_genesis_T.N.K.evolution_turbo_type_D/nnue.pdf (besucht am 28. 04. 2018).

[4] Stockfish. „Introducing NNUE Evaluation“. (2020), Adresse: https://stockfishchess.org/blog/2020/introducing-nnue-evaluation/ (besucht am 07. 08. 2020).

[5] Eduardo Vazquez-Fernandez und Carlos A. Coello Coello, „An adaptive evolutionary algorithm based on tactical and positional chess problems to adjust the weights of a chess engine“, in *2013 IEEE Congress on Evolutionary Computation*, IEEE, Juni 2013. DOI: 10.1109/cec.2013.6557727.

[6] Joel Staubach und Marvin Karhan. „Ein Algorithmenbasierter Schachcomputer“. (2022), Adresse: https://github.com/marvinkarhan/chess-engine/blob/master/Karhan_Staubach_Ein_algorithmenbasierter_Schachcomputer.pdf (besucht am 17. 08. 2022).

[7] Philipp Gysel, Mohammad Motamedi und Soheil Ghiasi, *Hardware-oriented Approximation of Convolutional Neural Networks*, 2016. DOI: 10.48550/ARXIV.1604.03168.

[8] Jiali Ma, Zhiqiang Zhu, Leyu Dai und Songhui Guo, „Layer-by-Layer Quantization Method for Neural Network Parameters“, in *Proceedings of the International Conference on Industrial Control Network and System Engineering Research*, Ser. ICNSER2019, Shenyang, China: Association for Computing Machinery, 2019, S. 22–26. DOI: 10.1145/3333581.3333589.

[9] Murray S. Campbell und T.A. Marsland, „A comparison of minimax tree search algorithms“, *Artificial Intelligence*, Jg. 20, Nr. 4, S. 347–367, Juli 1983. DOI: 10.1016/0004-3702(83)90001-2.

[10] David Levy, Hrsg., *Computer Chess Compendium*. Springer New York, 1988. DOI: 10.1007/978-1-4757-1968-0.