**Česká zemědělská univerzita v Praze**

**Provozně ekonomická fakulta**

**Katedra informačního inženýrství**

# Diplomová práce

## Použití umělé inteligence pro stimulaci logického myšlení

**Bc. Martin Muzika**

# ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

# ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Martin Muzika

Informatika

Název práce

**Použití umělé inteligence pro stimulaci logického myšlení**

Název anglicky

**Using artificial intelligence to stimulate logical thinking**

---

**Cíle práce**

Vytvořit umělou inteligenci, která bude asistovat při řešení logických úloh a rozvíjení ve hře Gomoku. K tomu bude využito algoritmu postaveného na neuronových sítích, které se budou učit na sofistikovaně vytvořených datech. Součástí práce bude vytvoření potřebných dat pro učení Neuronových sítí za pomoci stávajících programů.

**Metodika**

Student zhodnotí možnosti použití her pro stimulaci logického myšlení.

Popíše metody umělé inteligence, které se pro hraní her používají. V praktické části pak realizuje aplikaci, kterou lze k podpoře logického myšlení použít.

Zpracuje metodu prohledávání do hloubky a v praktické části zakomponuje naučenou Neuronovou síť do daného algoritmu.

**Doporučený rozsah práce**

60

**Klíčová slova**

deep learning, neural networks

---

**Doporučené zdroje informací**

GÉRON, Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and techniques to build intelligent systems.* Beijing ; Boston ; Farnham ; Sevastopol ; Tokyo: O'Reilly, 2019. ISBN 978-1-4920-3264-9.

CHOLLET, François. *Deep learning with Python.* Shelter Island: Manning, 2021. ISBN 978-1-61729-686-4.

---

**Předběžný termín obhajoby**

2023/24 ZS – PEF

**Vedoucí práce**

doc. Ing. Arnošt Veselý, CSc.

**Garantující pracoviště**

Katedra informačního inženýrství

Elektronicky schváleno dne 28. 11. 2023

**Ing. Martin Pelikán, Ph.D.**

Vedoucí katedry

Elektronicky schváleno dne 30. 11. 2023

**doc. Ing. Tomáš Šubrt, Ph.D.**

Děkan

V Praze dne 25. 03. 2024

**Čestné prohlášení**

Prohlašuji, že svou diplomovou práci Použití umělé inteligence pro stimulaci logického myšlení jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 31. 3. 2024 _____

**Poděkování**

Rád bych touto cestou poděkoval panu doc. Ing. Arnoštu Veselému, CSc., za vedení, odborné rady a podporu během mé diplomové práce.

Zároveň bych moc rád poděkovat panu doc. Ing. Janu Tyrychtrovi, Ph.D. za umožnění využití vědeckých laboratoří PEF.

Dále bych chtěl poděkovat mé rodině, přátelům a kolegům z práce za jejich podporu a cenné rady v průběhu práce na projektu a diplomové práci.

# Použití umělé inteligence pro stimulaci logického myšlení

**Abstrakt**

Tato diplomová práce se zaměřuje na využití neuronových sítí pro analýzu dat ve strategické deskové hře Gomoku s cílem zlepšit logické myšlení hráčů. Cílem práce je prozkoumat efektivitu různých architektur neuronových sítí při klasifikaci a detekci strategických vzorů a taktik ve vizuálních datech herních pozic hry Gomoku. Práce představuje použití nejnovější literatury a metod v oblasti neuronových sítí a jejich aplikací v analýze strategické hry Gomoku. Metodologie práce zahrnuje návrh experimentů s různými architekturami neuronových sítí a jejich implementaci pro analýzu herních pozic v Gomoku. Výsledky experimentů ukazují, že konvoluční neuronové sítě dosahují významně lepších výsledků ve srovnání s dalšími metodami neuronových sítí. Práce dále diskutuje faktory ovlivňující úspěšnost neuronových sítí v analýze strategické hry Gomoku a navrhuje možnosti případné optimalizace. Závěry práce potvrzují možnosti použití daných metod v Gomoku a nevylučují možnost vyzkoušení daných způsobů v podobných strategických hrách. Kde klíčem úspěchu jsou kvalitní data, která při použití neuronových sítí ukazují na potenciál těchto metod v praktických aplikacích. Výstupy práce poskytují cenné poznatky pro výzkumnou sféru v oblasti analýzy hry Gomoku a mohou přispět k rozvoji logického myšlení hráčů. Diplomová práce přináší ucelený pohled na využití neuronových sítí v analýze hry Gomoku a přispívá k rozvoji této oblasti výzkumu a aplikací.

**Klíčová slova:** neuronové sítě, Gomoku, analýza dat, strategické hry, konvoluční sítě, logické myšlení, vizuální data, optimalizace, klasifikace, experimenty

# Using artificial intelligence to stimulate logical thinking

**Abstract**

This master's thesis focuses on the utilization of neural networks for data analysis in the strategic board game Gomoku with the aim of enhancing players' logical thinking. The objective of the thesis is to explore the effectiveness of various neural network architectures in classifying and detecting strategic patterns and tactics in visual data of Gomoku game positions. The thesis presents the utilization of the latest literature and methods in the field of neural networks and their application in the analysis of the strategic game Gomoku. The methodology of the thesis includes the design of experiments with different neural network architectures and their implementation for analyzing game positions in Gomoku. The results of the experiments show that convolutional neural networks achieve significantly better results compared to other neural network methods. Furthermore, the thesis discusses the factors influencing the success of neural networks in the analysis of the strategic game Gomoku and suggests potential optimization options. The conclusions of the thesis confirm the potential use of the methods in Gomoku and do not exclude the possibility of testing these approaches in similar strategic games, where the key to success lies in high-quality data that demonstrates the potential of these methods in practical applications. The outputs of the thesis provide valuable insights for the research community in the field of Gomoku game analysis and can contribute to the development of players' logical thinking. The master's thesis offers a comprehensive view of the utilization of neural networks in the analysis of the Gomoku game and contributes to the advancement of research and applications in this area.

**Keywords**: Neural networks, Gomoku, analysis of visual data, strategic games, convolutional networks, logical thinking, visual data, optimization, classification, experiments

# Obsah

# 1 Úvod

Hra a logické myšlení je spojení, které dříve vyvolávalo rozpaky, avšak v současné době, v éře počítačů a umělé inteligence, se stalo velmi aktuálním tématem. Pozorují se schopnosti nejlepších hráčů v dané hře a analyzují se jejich reakce za různých podmínek, zaznamenávají se statistiky (Mráček, 2022). Masivní nárůst moderních her za poslední dvacetiletí přinesl několik her, které ovládly světovou scénu. Jednou z nich je bezpochyby Minecraft, hra, která si získala srdce mnoha hráčů díky své kreativitě a prostoru pro uvažování (Carrión, 2018).

Tato práce se zaměřuje na deskové hry a jejich vliv na logické myšlení člověka v dnešní době. Specificky jsem si vybral hru Gomoku, která sdílí některé charakteristiky se světově populární hrou šachy. Od svých sedmnácti let jsem se této hře věnoval, když jsem kvůli zranění kolena nemohl pokračovat ve hraní fotbalu. Začátky mojí cesty s Gomoku nebyly slibné, ale postupně jsem objevil piškvorkové úlohy, které mě vtáhly do hodin dlouhého zkoumání a hledání optimálního řešení. Tím jsem navázal na léta na základní škole, kde jsem se zabýval luštěním sudoku. Tento proces postupného rozvíjení logického myšlení mě nakonec přivedl až do společnosti specializující se na mainframe, kde jsem mohl uplatnit své dovednosti a znalosti z Gomoku při studiu kódu v Assembleru, Cobolu a PL1.

Práce na vlastních aplikacích, včetně Gomoku library, byla pro mě přirozeným pokračováním. Cílem bylo vytvořit úlohy, které by hráčům pomohly rozvíjet logické myšlení formou hry a zábavy, stejně jako to pomohlo mně před patnácti lety. V současném kontextu umělé inteligence je důležité mít dostatek nástrojů pro podporu rozvoje logického myšlení člověka.

Hlavním cílem této diplomové práce je přiblížit problematiku spojení her, umělé inteligence a logického myšlení. V teoretické části budou popsány metody používané k podobným problémům a zkoumány možnosti využití umělé inteligence ve hře Gomoku pro rozvoj logického myšlení. Praktická část se bude zabývat vytvářením dat a následným trénováním modelů neuronových sítí, které budou kombinovány s prohledáváním do hloubky. Dále budou popsány možná vylepšení odvozená z provedeného výzkumu. Závěrečná fáze práce bude věnována testování umělé inteligence na dobrovolnících.

# 2 Cíl práce a metodika

## 2.1 Cíl práce

Vytvořit umělou inteligenci, která bude asistovat při řešení logických úloh a rozvíjení ve hře Gomoku. K tomu bude využito algoritmu postaveného na neuronových sítích, které se budou učit na sofistikovaně vytvořených datech. Součástí práce bude vytvoření potřebných dat pro učení Neuronových sítí za pomoci stávajících programů.

## 2.2 Metodika

Student zhodnotí možnosti použití her pro stimulaci logického myšlení.

Popíše metody umělé inteligence, které se pro hraní her používají. V praktické části pak realizuje aplikaci, kterou lze k podpoře logického myšlení použít.

Zpracuje metodu prohledávání do hloubky a v praktické části zakomponuje naučenou Neuronovou síť do daného algoritmu.

# 3 Teoretická východiska

V teoretické části Vás seznámím s pojmy, problematikou, možnými metodami řešení problému a samotnou hrou.

## 3.1 Myšlení

> Určitý proces, který umožňuje zpracovat vstupní informace na základě kterých, pak dojde k určitému výsledku. Daný proces pak má za následek změny v jinak předvídaném a očekávaném výsledku. (Čtvrtníková, 2018)

## 3.2 Strategie

Uvažujme množinu možných situací, které mohou nastat. Pak určitý pokus o poskládání možných situací nad danou množinou můžeme nazvat strategií. (Muzika, 2017)

Nejlepší strategie popisují chování a množinu procesů hráčů. V teorii her se vyskytují dvě základní reprezentace strategií. (Muzika, 2017)

První je deterministická strategie, nazývaná pure strategy. Tato strategie je založená na jednotlivých akcích daných hráčů a umožňuje určit, jaká akce se má provést při uvažování dokonalé hry. (Muzika, 2017)

Druhá je nedeterministická strategie, nazývaná mixed strategy. Nedá se s jistotou říct, která akce je správná. Do hry vstupuje pravděpodobnost, která určuje, s jakou hodnotou je daná akce správná. (Muzika, 2017)

## 3.3 Logické myšlení

Jedná se o určitou část procesu myšlení člověka, kterou není snadné popsat ani přesně vysvětlit, co jí způsobuje. Z hlediska pozorování a výzkumů by se dalo uvažovat, že se skládá z jednotlivých podprocesů, které člověk dokáže nasbíranými zkušenostmi a dosaženými znalostmi zlepšovat a využívat k řešení složitějších problémů. Postupný vývoj lidstva ukazuje potenciální možnost tyto procesy trénovat a rozvíjet. (Čtvrtníková, 2018)

Logické myšlení může být ovlivněno několika významnými procesy. Mezi ně patří bezesporu intuice, paměť, vnímání, celkový stav jedince (duševní a fyzický), stres

a sebevědomí. Paměť se pak dá dále dělit na krátkodobou a dlouhodobou. (Čtvrtníková, 2018)

Výše uvedené procesy se dají dlouhodobě trénovat řešením různých hlavolamů, křížovek, matematických úloh, logických úloh, čtením knih a v neposlední řadě hraním her. (Mráček, 2022)

U jedinců, kteří se profesionálně věnují určitému druhu sportu se zaměřením na využití logického myšlení se očekává trénování jednotlivých procesů do krajních možností s minimalizací slabých míst a tím docílení maximálního využití svého potenciálu k dosažení cíle. (piškvorky.cz, 2006)

Mezi starodávné hry patří i piškvorky světově nazývané Gomoku, které svojí jednoduchostí, malým množstvím pravidel a přitom s obrovskými možnostmi strategií, ohromili svět už před dávnou dobou. Tato hra, na první pohled primitivní, má neskutečné množství možných situací, takzvaných pozic, do kterých se může dostat a tím se otevírají možnosti pro strategie. (piškvorky.cz, 2006)

## 3.4 Matematická logika

Logika v matematice systematicky analyzuje procesy za použití nástrojů běžných v matematice. Hlavními charakteristikami matematické logiky jsou formalizace a idealizace. Tím se myslí, že pro logiku má klíčový význam spíše forma usuzování, která k vyjádření používá symboly. Samotný obsah pak není tolik důležitý. Tato disciplína není omezena psychologickými faktory, a proto se může říct, že se vyhýbá zkoumání sociálních nebo individuálních vlastností. Naopak se dá říct, že se zaměřuje na abstraktní stránku deduktivního myšlení. Logika v matematice se pohybuje v oblasti pojmů dokazatelnosti, vyvratitelnosti a pravdivosti. Zkoumá jejich vztahy a tradičně se dělí na výrokovou a predikátovou logiku. Výroková logika, vytvořena stoiky, se zaměřuje na logické spojky a tvorbu složitějších výroků bez zasahování do sociálních nebo psychologických aspektů. Predikátová logika, studovaná Aristotelem a jeho studenty a učedníky, analyzuje výrazy (též

tak zvané predikáty), které přiřazují vlastnosti nebo vztahy prvkům, s důrazem na jejich logickou strukturu a abstraktní povahu. (Trombiková, 2021)

## 3.5 Hra

Hra je činnost, která obvykle probíhá v rámci určitých pravidel a má za cíl poskytnout zábavu, rekreační či vzdělávací hodnotu účastníkům. Během hry mohou jednotlivci či skupiny soutěžit či spolupracovat, často s využitím specifických pravidel, cílů a prostředků, které představují strukturu této činnosti. Hry mohou mít různé formy, včetně deskových her, sportů, videoher, kognitivních her, a mohou být motivovány soutěží, strategií, nebo jednoduše zábavou a relaxací. (Arjoranta, 2019, Muzika, 2017)

Můžeme si určit základní rysy hry asi takto: (Kopecká, 2019)

Pravidla: Hra má stanovená pravidla, která určují průběh a podmínky účasti. Pravidla mohou být formální (pevně stanovená) nebo neformální (vznikající během hry).

Cíle: Většina her má definované cíle, kterých hráči dosahují pomocí strategie, dovedností nebo náhody. Cíle mohou být individuální nebo sdílené mezi hráči.

Zábava: Hra je obvykle navržena tak, aby poskytovala zábavu a radost účastníkům. Tato zábava může být spojena s emocemi, soutěží, spoluprací nebo dokonce vzdělávacím prvkem.

Struktura: Hra může mít určitou strukturu nebo formu, která může zahrnovat kola, fáze nebo segmenty.

Interakce: Většina her vyžaduje interakci mezi účastníky. To může zahrnovat komunikaci, soutěž nebo spolupráci.

Teorie her se zabývá hlubší analýzou struktury her, jejich prvků a vlivu na účastníky. Tato oblast zkoumá strategie, rozhodování, motivaci a další aspekty spojené s hraním her.

## 3.6 Hry a logické myšlení

Logické myšlení hraje klíčovou roli v mnoha typech her a má vliv na strategii, rozhodování a úspěch hráče. Pro příklad uvedu některé způsoby, jak je logické myšlení využíváno v hrách: (Heus 2020, Mráček 2022)

Rozhodování a strategie:

V mnoha hrách, zejména v strategických nebo deskových hrách, je úspěch často spojen s logickým rozhodováním. Hráči musí přemýšlet o následcích svých akcí, předvídat tahy soupeřů a vytvářet dlouhodobé strategie. (Lin, Cheng 2022)

Řešení problémů:

Některé hry klade hráče před různé logické nebo matematické problémy, které musí řešit. To může zahrnovat hádanky, skládačky nebo situace vyžadující logické uvažování. (Heus 2020)

Logické spoje v hrách:

Výroková logika se často používá k modelování rozhodování v počítačových hrách. Hráčovo jednání nebo interakce s herním světem může být založeno na logických podmínkách a důsledcích. (Heus 2020)

Vývoj logických dovedností:

Některé edukační hry jsou navrženy tak, aby rozvíjely logické myšlení u hráčů, zejména u mladších hráčů. Tyto hry mohou obsahovat matematické úkoly, strategické hry nebo hádanky. (Lin, Cheng 2022)

Simulace a plánování:

Hry, které obsahují aspekty simulace nebo plánování, vyžadují logické myšlení. Hráči musí rozumět pravidlům herního světa, předvídat možné vývoje a reagovat na ně s ohledem na dosažení cílů. (Mráček 2022)

Kognitivní výzvy:

Některé videohry a aplikace pro trénování mozku jsou zaměřeny na rozvoj kognitivních dovedností, včetně logického myšlení. Tyto aktivity mohou zahrnovat různé logické úkoly a úlohy. (Mráček 2022)

Využití logického myšlení v hrách může být velmi rozmanité a závisí na konkrétním typu hry a jejím zaměření. Logika vytváří v hrách strukturu a umožňuje hráčům lépe porozumět pravidlům, predikovat možné výsledky a efektivně plánovat své kroky. (Heus 2020, Mráček 2022)

## 3.7 Vliv her na logické myšlení

Samotný vliv her na logické myšlení a uvažování člověka je široce pozorován a zkoumán už po mnoho let.

Hraní her může mít pozitivní vliv na rozvoj a posílení logického myšlení. Zde uvádím některé způsoby, jak mohou hry ovlivnit tuto schopnost:

Rozvoj strategického myšlení:

Hry, zejména strategické a taktické, vyžadují od hráčů plánování a rozhodování. Hráči se učí vážit si možností, předvídat následky svých akcí a vyvíjet dlouhodobé strategie. Tím si navíc vytvářejí znalost, kterou si rozvíjejí i paměť. (Gentile, Allegra, Söbke 2018)

Řešení problémů:

Mnohé hry obsahují různé úrovně obtížnosti a hádanky, což vyžaduje od hráčů logické uvažování při hledání řešení. To může posílit schopnost řešení problémů a analytické myšlení. Na tuto část se zaměřuje výzkum této práce. Kde výstup bude využit v Gomoku library k lehčímu generování logických úloh zaměřených na zlepšování znalostí a schopností ze samotné hry, tak i zlepšení logického myšlení, představivosti, intuice i v neposlední řadě řešení problémů v reálném životě. (Trombiková 2021, Chou 2017)

Rozvoj kritického myšlení:

Některé hry klade hráče do situací, kde musí kriticky hodnotit informace, provádět racionální rozhodnutí a odhadovat pravděpodobnostní scénáře. (Mráček 2022)

Trénování rychlého rozhodování:

Akční a rychlé hry mohou trénovat hráče ve schopnosti rychle rozhodovat a reagovat na nové situace. To může posílit logické myšlení v dynamickém prostředí. (Mráček 2022)

Rozvoj logických dovedností:

Některé hry jsou navrženy tak, aby přímo posilovaly logické dovednosti hráčů. To může zahrnovat hraní her, které se zaměřují na matematické úkoly, logické hádanky nebo hry s konceptem programování. (Trombiková 2021)

Využití různých logických prvků:

Hraní různorodých her může zapojit různé logické dovednosti, jako jsou deduktivní a induktivní úvahy, abstraktní myšlení nebo logická analýza. (Trombiková 2021)

Zlepšení koordinace rukou a očí:

Některé videohry vyžadují rychlé a přesné pohyby, což může posílit koordinaci mezi okem a rukou a rozvoj motorických dovedností. (Mráček 2022)

Nicméně, je nesmírně důležité brát v úvahu, že vliv hraní her na logické myšlení může být individuální a závisí na různých faktorech. Ať se jedná o typ her, délky hraní nebo přístupu k nim. Mírné, vyvážené a edukační hraní her může být pro mnoho jednotlivců pozitivním prvkem v rozvoji logického myšlení. (Lin, Cheng 2022)

Zajímavým příkladem může být začlenění Gomoku desek a kamenů do základní školy Antonínská v Brně. Žáci mají možnost hrát tuto deskovou hru o přestávkách na chodbách školy nebo přímo v určitých třídách.

## 3.8 Teorie her

Abychom mohli postoupit dále k praktické části, musíme nejdříve být schopni pochopit a porozumět problematice, která vzniká při hraní her a nazývá se teorie her. Využití her pro stimulaci logického myšlení bezesporu překrývá tuto část. Bez znalosti alespoň základů této problematiky není možné pochopit hlubší význam této práce.

Teorie her je odvětvím matematiky a ekonomie, které se zabývá studiem strategické interakce mezi racionálními rozhodovacími subjekty. Základními prvky teorie her jsou hráči, strategie, informace, akce a výsledek. (Hrůša, 2018) Tyto prvky se pokusím následně vysvětlit:

**Hráč:**

Teorie her se zabývá interakcemi mezi jednotlivci nebo skupinami, kteří jsou považováni za hráče. Z toho plyne, že hráči mohou být různí aktéři, od jednotlivců, skupin lidí až po firmy nebo dokonce země. Do této množiny by bylo možné zahrnout i některé stroje, které rozhodují na základě vstupů různě. (Muzika, 2017, Hrůša, 2018, Shoham, 2008)

**Strategie:**

Každý hráč má sadu možných akcí, které může podniknout. Tyto akce jsou nazývány strategie. Hráči se snaží vybrat nejlepší strategii na základě jejich cílů a informací. (Muzika, 2017, Shoham, 2008)

**Informace:**

Informace hraje klíčovou roli v teorii her. Existuje rozlišení mezi hrami s úplnou informací, kde všichni hráči mají plný přehled o situaci, a hrami s neúplnou informací, kde hráči nemají úplný přehled o rozhodnutí ostatních. (Muzika, 2017, Korotovskaia, 2020)

**Akce:**

Hráči podnikají akce v souladu se svými strategiemi. Tyto akce vedou k určitým výsledkům. (Muzika, 2017)

**Výsledek:**

Výsledek hry závisí na strategiích a akcích všech hráčů. Každý hráč může mít různý výsledek na základě toho, jak ostatní reagují na jeho akce. (Muzika, 2017)

Teorie her má široké uplatnění v ekonomii, politice, biologii, psychologii a dalších oborech. Hra může být zkoumána formálně pomocí různých modelů a matematických

nástrojů, jako jsou matice hry, Nashovy rovnováhy, koaliční hry a další. (Hrůša, 2018, Dosedělová, 2015)

## 3.9 Umělá inteligence

V dnešní době se pravidelně setkávám s názory lidí, kteří si myslí různé věci o umělé inteligenci. Otázky typu: "Převezme umělá inteligence moji práci?", "Nezpůsobí umělá inteligence vyhubení lidstva?" a podobné další otázky, které ukazují na strach z umělé inteligence, a to i přesto, že jí většina lidí využívá denně, neboť například Google vyhledávač by se dal považovat za umělou inteligenci. (Pazika, 2009)

Obecně se jedná se o odvětví informatiky, popřípadě počítačové vědy, které se zabývá vytvářením algoritmů a systémů, jež by byly schopni provádět úkoly, které vyžadují lidskou inteligenci. Umělá inteligence (UI) se snaží o vytvoření strojů, které jsou schopni učení, řešení problémů a vykonávání úkolů. Jako příklad může být uvedení rozpoznávání obrazů, plánování a rozhodování. (Antebi, 2021)

Hlavní cíl je a stále ještě bude simulovat a optimalizovat lidské myšlení. Tím se umožňuje možnost zlepšit efektivitu a inovaci v mnoha oborech. (Pazika, 2009)

## 3.10 Metody umělé inteligence

V tuto chvíli budu uvažovat pouze algoritmy a metody umělé inteligence, které jsou spojené a využitelné v řešení problémů v deskových hrách typu Go, šachů a Gomoku. Na úvod vysvětlím základní princip Minimaxu, pak vylepšení v podobě Alpha-Beta pruning, následně MCTS a na závěr Neuronovy sítě.

### 3.10.1 Minimax

Minimax je rekurzivní algoritmus pro nalezení nejlepší možné akce v dané situaci. Jedná se o formu hry, kdy máme úplnou informaci. (Muzika, 2017)

Používá se převážně v počítačovém zpracování her a v hrách se zero-sum strategií dvou hráčů. (Mráček, 2022)

Cílem algoritmu je minimalizovat ztrátu v nejhorším možném scénáři. Druhý hráč naopak chce maximalizovat svoje možnosti. Z toho důvodu název minimax. V obou případech se uvažuje optimální chování protihráče. (Muzika, 2017)

Níže uvádím pseudokód pro algoritmus minimax. Postup lze popsat následovně: algoritmus prochází strom nodu rekurzivně, kde pro daný node vždy vybere nejlevějšího potomka (následníka) dokud nedosáhne listu. V takovém případě vrátí hodnotu listu svému předchůdci, který porovná s nejlepší hodnotu a následně, pokud má další potomky, tak pokračuje k dalšímu potomku, dokud takto neprojde všechny následníky. Jakmile situace nastane, vrátí se z rekurze o úroveň výše a takto se pokračuje, dokud nedosáhne kořene stromu. (Muzika, 2017)

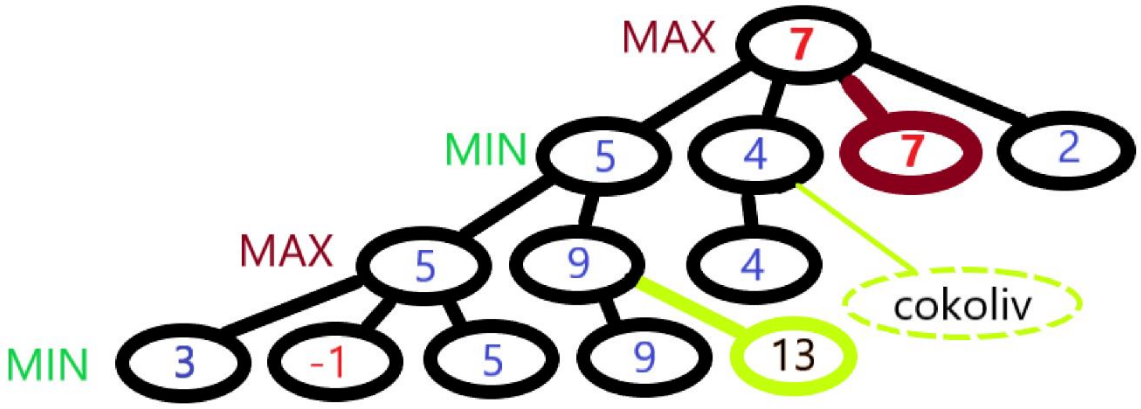
```
minimax(node, depth, maximizingPlayer):
    if depth is 0 or node is a terminal node:
        return evaluate(node)
    if maximizingPlayer:
        bestValue = negative infinity
        for each child of node:
            value = minimax(child, depth - 1, false)
            bestValue = max(bestValue, value)
        return bestValue
    else:
        bestValue = positive infinity
        for each child of node:
            value = minimax(child, depth - 1, true)
            bestValue = min(bestValue, value)
        return bestValue

evaluate(node):
    // tady by se provedla evaluace listu a vrátila hodnota odpovídající nodu
```

*Obrázek 1: pseudocode minimaxu*

### 3.10.2 Alpha-Beta pruning

Nejznámějším vylepšením minimaxu je Alpha-Beta pruning. Po důkladném prostudování minimaxu lze vidět, že může navštívit všechny nody, než dojde k svému cíli. To se dá vylepšit tím, že si budeme držet hodnoty alpha a beta, kde alpha je nejlepší hodnota pro maximalizujícího hráče a beta je nejlepší dosažená hodnota pro minimalizujícího hráče. Lze z toho vytvořit interval, který budeme upravovat podle hodnot daných nodů. Ve chvíli, kdy zjistíme, že interval je prázdný, přestaneme prohledávat. Tím umožníme odříznout některé podstromy a tím ušetřit obrovské množství navštívených nodů. Ani toto vylepšení nezaručí, že nenastane situace, kdy navštívíme všechny uzly.

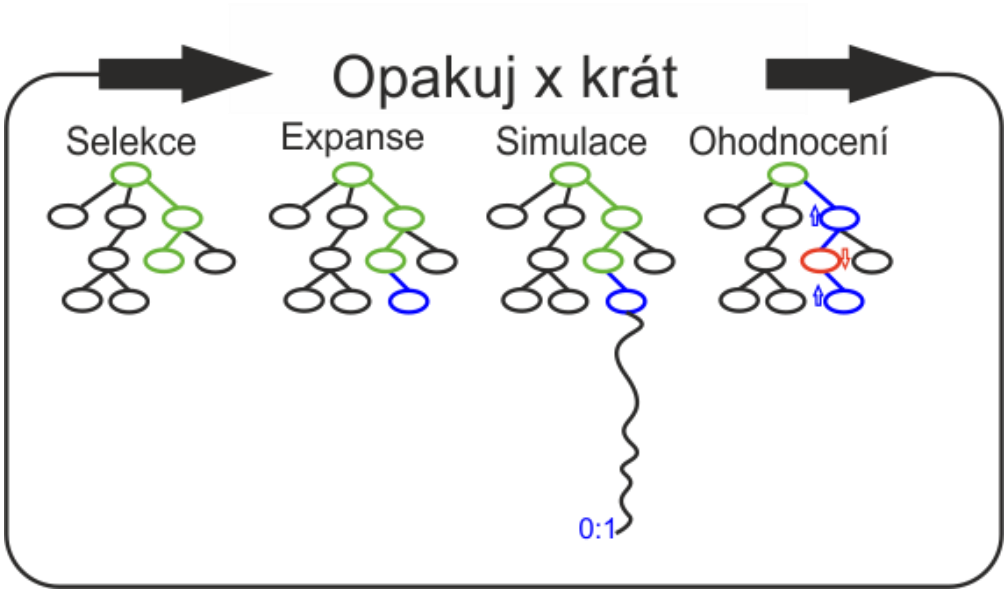
V praxi existují ještě další způsoby vylepšení. Ať už se jedná o transpoziční tabulky nebo například limity hloubky, množství navštívených nodů nebo omezení času. (Muzika, 2017)




*Obrázek 2: Ukázka Alpha-beta. Můžeme vidět odříznuté větve už v první části. Zajímavá je druhá větev, kde vstupem s alpha=5 dojde k odříznutí všeho co se nachází v "cokoliv". Další posunem může být seřazení uzlů. Pak by výpočet vypadal jinak.*

### 3.10.3 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) je stochastický algoritmus používaný pro hledání optimální strategie v rozhodovacích stromech. Základní koncept je postavený na prohledávání stromu a náhodné simulaci. (Trunda, 2013) MCTS je často využívaný ve hrách s konečným počtem tahů. Jako příklad můžou být uvedené hry GO, šachy, poker nebo Gomoku, kde prohledávací prostor je konečný, ale pro minimax příliš velký. (Kang, 2016)




*Obrázek 3: Ukázka základního principu MCTS*

Popis základního MCTS je následující: (Trunda, 2013)

Algoritmus MCTS staví strom takovýmto způsobem:

- Selekce – výběr nodů na základě zvolené heuristiky dokud nedorazíme k listu stromu. V případě nesplnění ukončovací podmínky, následuje krok 2) jinak 4).
- Expanse – vytvoří node, který se stane následníkem (potomkem) listu z 1). Následně zkontroluje ukončovací podmínku a pokud není splněna, následuje krok 3) jinak 4).
- Simulace – program náhodně hraje hru od nodu, který přidal v 2). Po dosažení ukončovací podmínky simulace následuje bod 4)
- Hodnota se zpropaguje přes všechny prošlé nody (zahrnuje všechny node z 1) a 2)).
- Tyto čtyři kroky opakuje x krát, dokud nenastane nějaká z ukončovacích podmínek.

Tento postup umožňuje MCTS systematicky prohledávat strom a přidávat hodnoty k uzlům na základě vyhodnocení simulace. Algoritmus se často opakuje a tím posiluje svoji strategii.

Bylo dokázáno, že MCTS konverguje k výsledkům minimaxu (Lisy, 2013). Nicméně základní MCTS konverguje velmi pomalu.

### 3.10.4 Neuronovy sítě

Neuronova síť je matematický model inspirovaný rozložením a fungováním neuronů lidského mozku. Tento model se využívá k řešení úkolů strojového učení a v umělé inteligenci. Síť se skládá z jednotlivých neuronů, které jsou mezi sebou propojeny váhami a rozdělují se do vrstev. (Géron, 2019)

Účel neuronovy sítě je naučit se složité vzory a reprezentace ze vstupních dat. (Géron, 2019)

Neuronova síť má čtyři základní prvky:

**Neurony**

Základní prvky sítě. Jejich funkcí je přijímání vstupu, zpracování vstupu na základě vah a aktivační funkce. Výsledek pak vrátí jako výstup. Dle typu sítě se pak dělí na vstupní, vnitřní a výstupní. (Chollet, 2021)

**Váhy**

Jedná se o parametry, které ovlivňují přenos signálu mezi jednotlivými neurony. Tyto parametry jsou optimalizovány v průběhu procesu učení Neuronovy sítě. Díky tomu je Neuronova síť schopna lépe modelovat nebo klasifikovat vstupní data. (Chollet, 2021)

**Vrstvy**

Jak už bylo napsáno výše, Neuronova síť se rozděluje do vrstev, kde vstupní vrstva zpracovává vstupní data, skryté vrstvy zpracovávají informace a výstupní vrstva vytváří výstup. (Chollet, 2021)

**Aktivační funkce**

Každý neuron má aktivační funkci, která určuje výstup daného neuronu na základě vstupních hodnot a vah. Aktivační funkce přidávají nelinearitu a umožňují učit se komplexnějším problémům. (Chollet, 2021)

Důležité je si uvědomit, že existují různé architektury. Mezi známé architektury patří vícevrstvé perceptrony (MLP), rekurentní neuronové sítě (RNN) a konvoluční neuronové sítě (CNN), kterým se budu v této diplomové práci věnovat. Důvod různých architektur je způsoben tím, že každá jednotlivá architektura je vhodná pro jiný typ úkolu. Ať už se jedná o klasifikaci, zpracování sekvencí nebo detekci a rozpoznávání. (Graupe, 2019)

### 3.10.5 Konvoluční neuronové sítě (CNN-tensorflow, 2023)

CNN je typ neuronové sítě, který jsem zvolil jako finální verzi praktické části. Hodí se pro zpracování vizuálních dat jako jsou obrázky a videa. Architektura CNN využívá takzvané konvoluční vrstvy, které se hodí pro efektivní extrahování charakteristických rysů ze vstupních dat.

Pokud model neuronové sítě obsahuje alespoň jednu konvoluční vrstvu, pak se jedná o model spadající do konvoluční neuronové sítě.

Mezi základní složky konvoluční sítě patří konvoluční, agregační též nazývané pooling, a plně propojené vrstvy. (Machine learning glossary, 2023)

#### 3.10.5.1 Konvoluční vrstvy

Konvoluční vrstvy používají filtry (jádra) k provádění operace (konvoluce) nad vstupními daty. Filtry jsou v tomto případě malé matice, které se posouvají přes vstupní data provádějí základní elementární operace a díky tomu umožňují detekci různých vzorů. (CNN-tensorflow, 2023)

Druhá část konvoluční vrstvy je aktivační funkce, která následuje hned po provedení konvoluce. Jedná se o aplikaci nelineární aktivační funkce. Mezi často používanými je Rectified Linear Unit (ReLU). Tato aktivační funkce přidává do modelu nelinearitu. (CNN-tensorflow, 2023)

#### 3.10.5.2 Agregační vrstvy

Agregační vrstvy známé jako pooling vrstvy jsou klíčovou složkou CNN. Pooling vrstvy slouží k redukci rozměrů dat a tím k zjednodušení reprezentace. Díky tomu dochází k efektivnímu zpracování vložených informací. (CNN-tensorflow, 2023)

Mezi nejznámější a nejpoužívanější způsob patří max pooling. Dalšími používanými jsou average pooling, global average pooling a agregační krok. (CNN-tensorflow, 2023)

3.10.5.3Plně propojené vrstvy

V plně propojených vrstvách jsou všechny neurony vzájemně propojeny s neurony předchozí i následující vrstvy. Tato vrstva se většinou nachází na konci architektury a přeměňuje informace ze skrytých (v tomto případě konvolučních) vrstev na výstupy nebo klasifikace. (CNN-tensorflow, 2023)

Plně propojeným vrstvám se též říká dense layers.

Předpokládejme vstupní data a filtr 3x3:

| | | | |
|---|---|---|---|
| 172 | 93 | 56 | 48 |
| 59 | 71 | 63 | 73 |
| 128 | 36 | 81 | 137 |
| 158 | 47 | 94 | 186 |

| | | |
|---|---|---|
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |

Pak první výpočet by byl:

| | | |
|---|---|---|
| 172 | 93 | 56 |
| 59 | 71 | 63 |
| 128 | 36 | 81 |

| | | |
|---|---|---|
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |

>

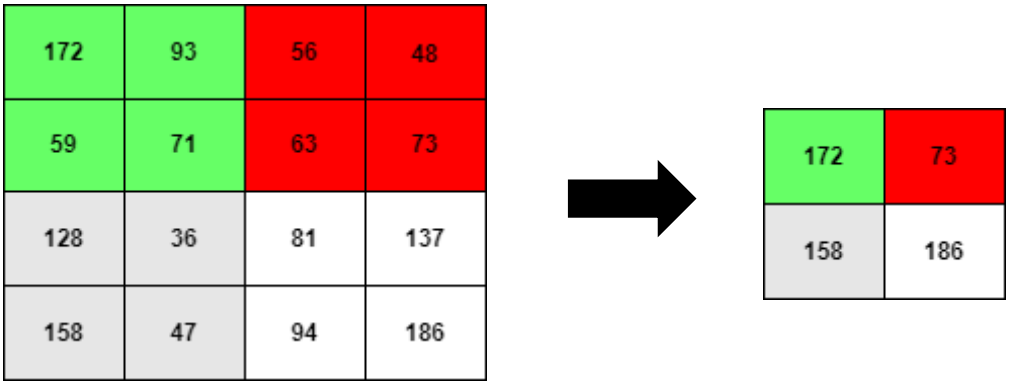
| | | |
|---|---|---|
| 172 | 0 | 0 |
| 0 | 71 | 63 |
| 128 | 0 | 0 |

Kde 172 + 71 + 63 + 128 se rovná 434. Celkový výsledek by vypadal takto:

| | |
|---|---|
| 434 | 265 |
| 334 | 336 |

Můžeme si všimnout, že aplikováním filtru 3x3 jsme z původní matice 4x4 dostali 2x2. Je možné vynutit takzvaných padding, kdy původní vstupní se doplní o y řádků a x sloupců hodnot (většinou nul), kde x je x-ová velikost filtru minus 1 a y je y-ová velikost filtru minus 1. Díky tomu bychom dostali výslednou matici opět 4x4. Tím bychom zamezili určité ztrátě informace.

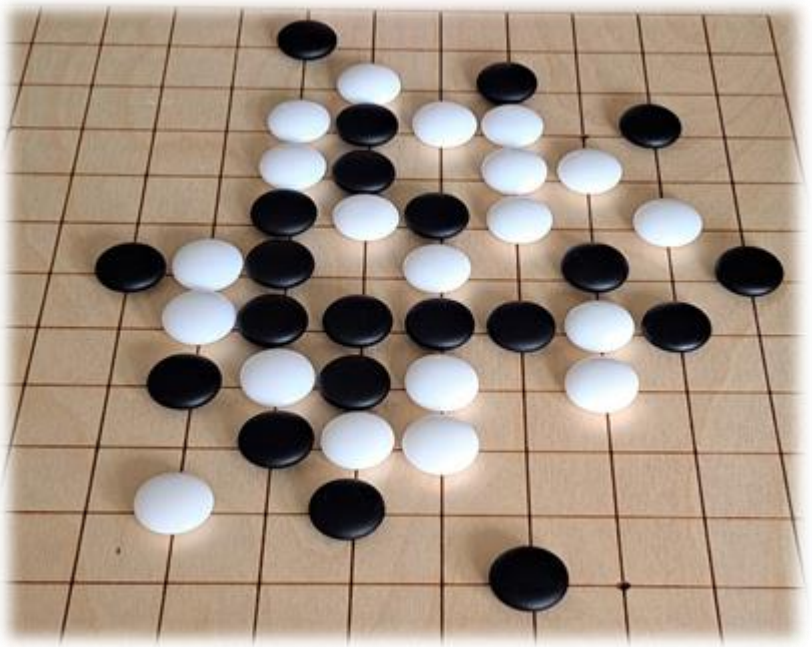
Nyní předpokládejme stejný vstup, filter (2 x 2) a stride (2, 2), pak výsledkem bude matice 2x2 ukázána vpravo.



## 3.11 GOMOKU

Gomoku, zajímavá a na pravidla poměrně jednoduchá hra, je ve světě známá pod mnoha jmény. Nejznámější je určitě tic-tac-toe, o něco méně pak five in a row. V České republice je tato hra známá pod názvem piškvorky. (Muzika, 2017, piškvorky.cz, 2006)

Gomoku je oficiálně hra pro dva hráče. Hraje se na desce s černými a bílými kameny, které se pokládají na průsečíky. Deska má 225 průsečíků, které jsou ve čtvercové struktuře 15x15. Hru začíná vždy černý hráč. Každý kámen musí být položený na průsečík nikoliv na čtverce. Hráči musí brát v potaz, že každý kámen položený na desku zůstane na svém místě až do konce hry. Hráči nemohou žádný kámen odebrat nebo změnit pozici kamene. Vítězem se stává ten hráč, který jako první poskládá přesně pět kamenů v řadě v jakémkoliv směru. Více jak pět kamenů se v oficiálních pravidlech nebere jako výhra. (Muzika, 2017, piškvorky.cz, 2006)



*Obrázek 4: Ukázka hry Gomoku na dřevěné desce s kameny*

Hru si můžeme představit jako stavový prostor, který je konečné velikosti, ale příliš velký z pohledu paměti a výpočetního výkonu. Každá pozice hry je jedním stavem v daném prostoru prohledávání a přidáním kamene se přesuneme do jiného stavu. Existují stavy, do kterých se můžeme dostat více způsoby a z toho důvodu se nejedná o strom. (Levin, 2018)

### 3.11.1 Historie a vývoj hry

Gomoku v základních pravidlech je hra stará tisíce let. Nejvíce byla populární v 18. a 19. století v Japonsku, kde se hra postupně rozvinula do takové míry, že základní pravidla přestala stačit, neboť Japonci si uvědomili, že začínající má obrovskou výhodu. Díky tomu vznikl první termín surewin, nebo-li jasná výhra. Ať už to bylo jen ze statistiky, nebo přímo analýzou hry, to se neví. Nicméně to vedlo k úpravám pravidel.

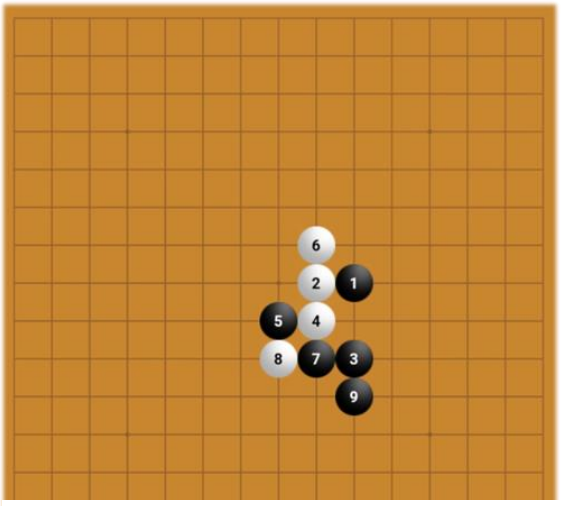
Jeden z prvních pokusů byl ten, že přestalo platit více jak pět v řadě. Následovala další řada úprav, která ze základního Gomoku začala vytvářet podobné hry. Jako příklad může být remíza, pro, swap, swap2, pente nebo renju, kterému se profesionálně věnují hráči v Číně a Japonsku.

Vývoj se vedl dvěma směry. Varianty typu remíza, pente a renju se zaměřili na změny pravidel takovým směrem, že hra byla ovlivňována ukončováním hry a tím změnil smysl klasických piškvorek. Druhý směr změn variant základních piškvorek vedl přes omezování začátku hry, kde po omezeném začátku se pokračovalo stejným způsobem jako v klasických piškvorkách, jak je každý zná. Mezi tyto způsoby patří varianty pro, long-pro, swap a swap2, který se stal světově uznávaným pravidlem a je používaným pravidlem na všech velkých turnajích.

Další velký milník nastal v roce 1994, kdy L. Victor Allis dokázal surewin v Gomoku na základě matematiky a prohledávání v kombinaci s databází. Zároveň i dokázal, že přesně pět v řadě k zabránění výhry začínajícího nestačí. (Allis, 1994)

#### 3.11.1.1 Klasické piškvorky

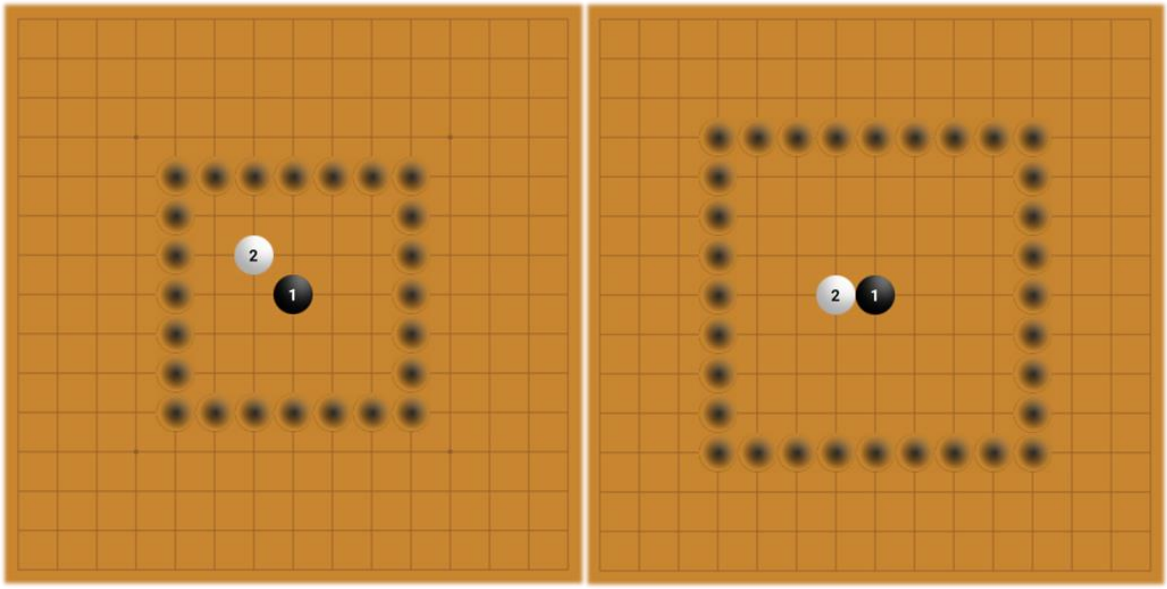
První hráč začíná hru, pokládá černý kámen a až do konce hry hraje za černé kameny. Druhý hráč hraje druhý, pokládá bílý kámen a až do konce hry hraje za bílé kameny. Ve hře se střídají po jednom tahu a vyhraje ten, kdo dříve poskládá pět a více kamenů v řadě.




*Obrázek 5: Ukázka klasických piškvorek. Černý má dvě nebráněné hrozby a hru velmi rychle vyhraje.*

#### 3.11.1.2 Varianty pro a long-pro

Hráči se střídají v tazích stejně jako v klasických piškvorkách. Rozdíl je pouze v omezení na začátku hry. První (černý) kámen se položí na střed hrací desky (h8). Druhý hráč může položit druhý (bílý) kámen v přímém sousedství prvního kamene. Má tedy 8 možností, kam položit druhý kámen. Doteď byly obě varianty shodné. Následně první hráč ve variantě pro musí položit třetí kámen mimo středový čtverec 5x5. Ve variantě long-pro musí první hráč položit třetí kámen mimo středový čtverec 7x7. V tuto chvíli se ví, že varianta pro je stále výhra začínajícího hráče za předpokladu ideálních tahů obou hráčů. Varianta long-pro i přes existenci hratelných variant se nepoužívá, neboť po vyřazení špatných variant zůstane jen pár hratelných začátků.




*Obrázek 6: Znázornění variant pro a long-pro. Rozostřené body znázorňují nejbližší možné tahy k středu prvního hráče při zahrání třetího kamene.*

### 3.11.2 Swap a swap2

Hráči piškvorek si byli vědomi omezení variant pro a long-pro. Výhoda černého v některých variantách byla tak značná, že se pak hrálo jen pár variant a hra pomalu konvergovala ke klasickým piškvorkám. Ukázkou může být mistrovství světa v Gomoku z roku 1989 a 1991, kde se hrála varianta *pro*. Hra se postupně stávala méně atraktivní ažturnaje typu mistrovství světa v Gomoku, a až do roku 2009 zmizeli.

#### 3.11.2.1 Swap

Naštěstí přišla chvíle, kdy bylo vymyšleno pravidlo **swap**.

Body pravidla swap:

1) Hru začíná první hráč, který položí 2 černé a 1 bílý kámen na desku kamkoliv bude chtít.
2) Druhý hráč má dvě volby:
   a) Vybere si bílé a v tu chvíli je na tahu a až do konce hry hraje za bílé.
   b) Vybere si černé a až do konce hry hraje za černé kameny. V tu chvíli je na tahu první hráč, který až do konce hry hraje za bílé kameny.
3) Hra od této chvíle pokračuje až do konce hry stejně jako klasické piškvorky. Hráči se střídají po tahu.

Představte si, že jste vášnivý hráč šachu a najednou vám řeknou, že pravidla se mění. Místo původního rozestavení figurek, máte možnost na začátku hry rozestavit figurky podle sebe. Jediný háček bude v tom, že volbu strany, za kterou barvu bude kdo hrát, určí druhý hráč po rozestavení figurek prvním hráčem. Přesně tak se změnila hra piškvorky ve chvíli, kdy bylo vymyšleno pravidlo swap.

Pravidlo se zdálo dokonalé, ale přesto se časem ukázala ještě jedna slabina. Přečtěte si pravidlo ještě jednou a popřemýšlejte, co se stane, když se naučíte jednu variantu tří kamenů tak dobře, že ji budete umět hrát dokonale. Soupeř s největší pravděpodobností prohraje, neboť nejspíše nebude mít znalost, čas nebo schopnosti situaci vyřešit správně. Z toho důvodu vzniklo pravidlo swap2.

#### 3.11.2.2 Swap2 (gomokuworld.com, 2013)

Body pravidla swap2:

1) Hru začíná první hráč, který položí 2 černé a 1 bílý kámen kamkoliv na desku bude chtít.
2) Druhý hráč má tři volby:
   a) Vybere si bílé a v tu chvíli je na tahu a až do konce hry hraje za bílé.
   b) Vybere si černé a až do konce hry hraje za černé kameny. V tu chvíli je na tahu první hráč, který až do konce hry hraje za bílé kameny.
   *c) Přidá 1 bílý a 1 černý kámen kamkoliv na desku a přenechá volbu barvy prvnímu hráči, který už si musí vybrat barvu.*

i) *Vybere si bílé a v tu chvíli je na tahu a až do konce hry hraje za bílé.*

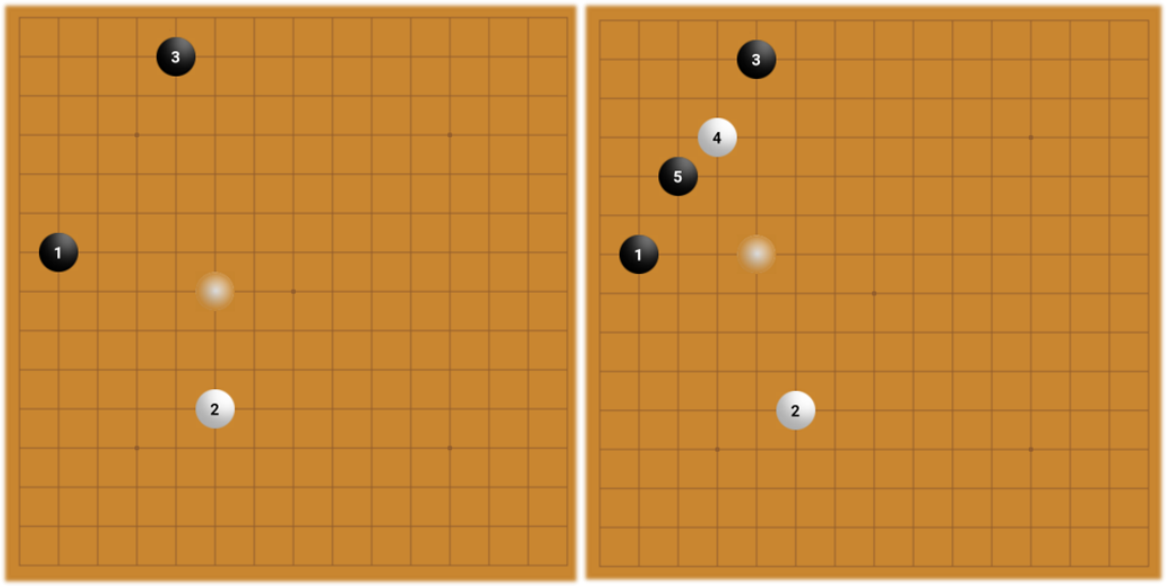
ii) *Vybere si černé a až do konce hry hraje za černé kameny. V tu chvíli je na tahu druhý hráč, který až do konce hry hraje za bílé kameny.*

3) Hra od té chvíle pokračuje až do konce hry stejně jako klasické piškvorky. Hráči se střídají po tahu.

Přidáním třetí možnosti pro druhého hráče vznikla možnost eliminovat znalosti prvního hráče a zároveň touto změnou bylo umožněno druhému hráči kartu obrátit vlastní přípravou na soupeřovo zahájení.

Hra se tímto stala vyrovnanou a otevřela nepřeberné množství strategií.

Pravidlo swap3 a více bylo prozatím zamítnuto kvůli zbytečnému navýšení komplexnosti a nepotřebnosti.




*Obrázek 7: Ukázka netypicky rozložených 3 kamenů takzvaně "do schématu", kde rozostřené body vždy značí nejlepší následující tah. Druhá ukázka je pak využití swap2 pravidla, kdy druhý hráč přidal 2 kameny (bílý a černý) a tím rozbil původní teorii začátku.*

### 3.11.3 Potenciální možnosti Gomoku na rozvoj logického myšlení

Důležité je si uvědomit, že se jedná o starodávnou hru, která má velmi jednoduchá pravidla, a přesto dává možnost neskutečným strategiím. Zároveň hra obsahuje obrovský potenciál na rozvoj představivosti, zlepšování intuice a otevírá možnost pro kreativitu. Hráči i jen pozorující diváci často počítají, aniž si to uvědomují. Hra obvykle vyžaduje po hráči být schopen si jednotlivé tahy vizualizovat a vnímat, v jakých prostorách bude třeba hrát a zda je nutné reagovat na soupeřovy hrozby.

Z toho dostáváme tyto části, které hráč přímo trénuje:

- Strategie
- Rozvíjení představivosti, intuice a kreativity

Kreativita je nejlépe vidět přímo při výběru prvních tří kamenů a následné volbě, kdy hráč může přidat dva kameny. Intuice je velmi využívaná část při hře, neboť z důvodů nedostatku času nebo výpočetní kapacity je potřeba ohodnotit tahy bez možnosti propočítání hry do hloubky.

- Nezamýšlené počítání

Při hře dochází k situacím, kdy je potřeba pozici spočítat, než se rozhodnete zahrát další tah. V součtu se zápalem a touhou po vyřešení dané pozice a snahou neprohrát, dojde k počítání, které nebylo zamýšlené.

- Prostorové a časové vnímání

Čas a prostor patří mezi základní prvky hry. Bez vnímání rozložení kamenů a vlivu hran není možné ve hře uspět. Trénink těchto částí je nezbytný nejen z pohledu hry.

K tomu musíme přidat:

- Trénink koncentrace a trpělivosti

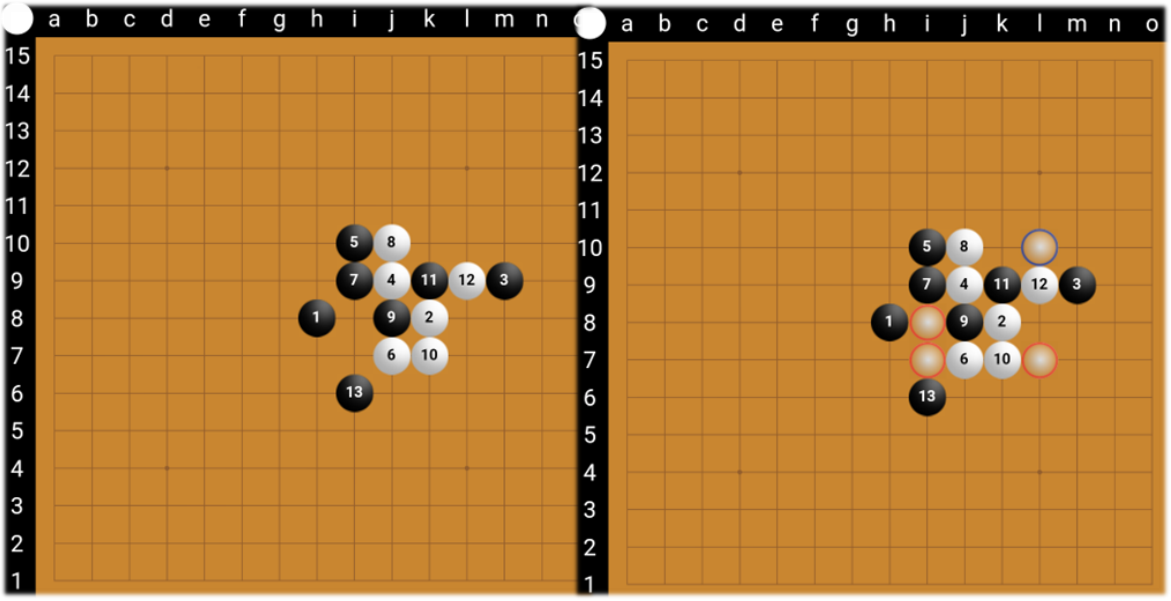
Neboť piškvorky, stejně jako šachy, obnáší náročné partie, které jsou často rozhodnuty malými detaily, kde koncentrace hraje velkou roli. Trpělivost pak ve hře má také své místo, neboť je důležité jednotlivé tahy neuspěchat. A to nás dovádí k poslednímu velkému bodu:

- Rozhodování

Téměř v každém tahu má hráč příležitost se rozhodnout mezi různými možnostmi, kam by měl zahrát. Přitom jednotlivé možnosti se dají postupně dopočítat nebo odvodit.

Spojením detailů pak vzniká ucelená hra. Problém rozhodování je zapotřebí, neboť většinou není k dispozici dostatek času nebo výpočetní schopnosti k dopočítání hry.

Jako příklad uvedu následující situaci (obr 8.), kterou jsem nechal řešit různé věkové skupiny:




*Obrázek 8: Ukázka úlohy zaměřující se na následující nejlepší tah. Modrý kruh znázorňuje správný tah a červené kruhy znázorňují špatná řešení.*

Na tahu je hráč hrající za bílé, označme si hráče W.

K vyřešení situace výše si W musí být vědom následujících detailů:

Hráč hrající za černé kameny (označíme B) zabránil 13. kamenem hrozbě trojky a vytvořil si potenciální dvojitou hrozbu na pozici i7, kterou někteří řešitelé neuvažovali a pokusili se zahrát na l7. Po oznámení se přidali k dalším řešitelům, kteří zahráli 14. Kámen na i7. V takovém případě hráč B zahraje na l7 a hráč W přijde o přímou výhru. Další možný tah, který byl hrán, je i8, který se zdá být jako velmi zajímavý. Hráč W brání přímou hrozbu hráče B a zároveň si vytváří další dvě dvojice (i8, j9 a i8, j7). Nicméně, hráč B zahraje l10, následuje i7 (W) a h7 (B).

Pouze velmi pokročilí řešitelé dokázali vymyslet l10, který jako jediný vede k rychlé výhře bílého. Úloha č.5 Gomoku library (mobilní aplikace).

Z výše uvedeného příkladu můžeme vidět, že je potřeba umět odvodit a dopočítat tahy, které jsou proherní a tím odfiltrovat možnosti, které se nabízí, ale jsou špatně. Tím si hráči trénují další důležitou vlastnost, která se spojuje s logickým myšlením:

- Dedukce

Řešitelé si postupně vštěpují důležité prvky pro řešení problémů. Důsledky jednotlivých rozhodnutí daných řešitelů mají velkou váhu. Postupně učí řešitele, že méně je často více. Neboť čím méně kamenů je na desce zahraných, tím větší šance hru ovlivnit a určit směr, kterým se bude daná partie ubírat.

Při hře se pak samotná partie stává často výzvou a pro mnoho lidí i adrenalinem.

*Obrázek 9: Dvě ukázky z Mistrovství světa v Gomoku, kde je znázorněna strategie, která započítává čas pomocí přepínacích hodin, který pak ovlivní hru.*

## 3.12 Využití znalostí v praxi

Myšlení je bezesporu nutnou součástí života. Zda a proč trénovat logické myšlení jsou otázky, které mnoho lidí napadne téměř ihned. Ve škole se učí matematika, logika, fyzika a další předměty, které bezesporu spojuje logické myšlení. Nástupem do praxe a běžného studia se člověk ocitne před úkoly, které se často neshodují s ukázkovými příklady ze školy. Přesto se dané problémy vyřeší, neboť logické myšlení umožňuje přijít na způsob, jak daný problém zvládnout.

V mém případě využívám vytrénované schopnosti v práci téměř každý den. Zkoumání starého assemblerového kódu, hledání chyb v místech kódu, které vyžaduje studování a analyzování takzvaného "DUMPu" mi samotnou hru často připomínají. Na začátku se nachází volné pole působnosti, kde netušíte, co můžete najít, velmi podobné začátku samotné hry. Postupně sbíráte drobné informace a snažíte se pochopit daný problém, aniž byste se snažili naučit celý produkt. Někdy se podaří pochopit dané části kódu a myšlení programátora, který daný kód před třiceti lety napsal. Taková situace se podobá partii, kde se nacházíte v nové pozici, kterou vyřešíte a zpětně zjistíte, že jste přišli na nejlepší postup. Následně přicházejí podobné otázky jako po hře. "Jak to mohl někdo takto napsat?" "Jak je možné, že to většinou funguje?" "Proč se na to nepřišlo dříve?"

Jednoduché otázky položené při řešení problémů, které stojí firmy obrovské množství času a peněz. Často se viní autor kódu, přitom chybou může být nepochopení původního plánu nebo využití kódu jiným směrem, než bylo zamýšleno. Většina lidí v dnešní době preferuje psát od začátku. Pochopení a přepsání starého kódu stojí obrovské úsilí. Tím vzniká určitá část, kterou nikdo nechce, nemůže nebo nemusí dělat. Přesto existují části kódu, které fungují desítky let a v tuto chvíli existuje jen malé množství lidí, kteří by byli schopní pochopit a opravit daný kód. K tomu bezesporu potřebují nejen dobrou znalost jazyka, ale i velmi dobré logické myšlení. V opačném případě by nebyli schopni pochopit komplexní situace a místo opravy by často způsobili větší škody.

Logické myšlení nemusí být spojené jen s programováním. Lze ho využít u běžných prací. Často lidi dělají věci způsobem, které by šlo řešit lépe. Ať už se jedná o úklid domu a zahrady, skladování zásob, hospodaření s penězi a časem, trávení volného času, pěstování ovoce a zeleniny, vždy bude možnost logické myšlení využít.

S Gomoku znalostmi získanými z řešení úloh, analýzou her, nebo soupeřením s umělou inteligencí je to podobné.

V průběhu let jsem zaznamenal velké množství případů, kdy logickým myšlením šlo zakrýt nedostatek znalostí daného problému a díky tomu vyřešit případy v mnohem kratším čase.

# 4 Vlastní práce

V praktické části jsem se zaměřil na vytvoření umělé inteligence, která bude sloužit jako pomocník pro zlepšování logického myšlení. Umělá inteligence (AI) se bude skládat z neuronových sítí, které budou schopny v jednotlivých pozicích hry vrátit přijatelné tahy a tyto tahy pak bude druhá část AI propočítat do hloubky pomocí alpha beta algoritmu.

Neuronovy sítě budou sloužit nejen jako generátor možných tahů, ale též budou jednotlivé tahy řadit a tím pomohou alpha beta snáze identifikovat lepší větve stromů a tím urychlí samotný výpočet AI.

V první části praktické práce se zaměřuji na vytváření dat pro neuronovy sítě, neboť v předchozí práci (Muzika, 2017) se ukázal postup založený pouze na datech hraných lidmi jako velmi špatný. Této části jsem věnoval zhruba 12 měsíců průběžného připravování a počítání dat. Jednotlivé kroky jsem pak konzultoval s odborníky z praxe.

V druhé části praktické práce se zaměřuji na trénování neuronových sítí, předělávání dat, zapojení neuronových sítí do alpha beta algoritmu a prvním experimentům. Vzhledem k obsáhlosti problematiky by bylo možné pokračovat ještě několik dalších let k úplnému dotažení práce k zdárnému konci.

Do praktické části jsem zahrnul všechny své postřehy, nápady a pokusy, kterých jsem docílil postupným zkoumáním, konzultováním s odborníky a testováním.

V průběhu posledních 12 měsíců jsem posbíral několik cenných rad od programátorů úspěšných projektů zaměřujících se na vývoj AI v Gomoku a opakovaně se účastnících Gomocup soutěže. Část z nich jsem v průběhu vývoje zahrnul do projektu a zbylé rady postupně zahrnul do této práce, neboť se jedná o cenné rady, které mohou mnoha čtenářům usnadnit vývoj jejich vlastní AI a třeba posunout hranici o kus dále. Přesto všechno je dost důležité mít ujasněnou představu, jakým směrem se vydat při tvorbě AI.

V závěru pak shrnu celkový výstup práce, zhodnocení cílů a stručně popíšu nabyté zkušenosti.

## 4.1 Tvorba dat

Data pro neuronovy sítě mohou být pojmuty několika různými způsoby. Po zkušenostech z dřívějších pokusů jsem si byl vědom důležitosti kvality dat (Muzika, 2017) a z toho důvodu jsem přikládal samotné tvorbě dat největší důležitost a tím jsem se snažil poučit z chyb vzniklých v předchozí práci (Muzika, 2017).

Samotný plán, jak data vytvořit, vznikl postupně. Cílem bylo dosáhnutí takové podoby, která by zaručila kvalitu dat, a zároveň by stále data vycházela z reálných her. Z toho důvodu jsem se rozhodl vzít hry z offline partií.

Jednotlivé hry jsem rozdělil na pozice takovým způsobem, že každý odehraný tah v dané hře vytvořil novou pozici. Tímto způsobem jsem získal n pozic z jedné partie, která měla n kamenů. Tím skončila lidská část zahrnuta v datech pro neuronovy sítě.

V dalším kroku jsem vzal jednu pozici, pro kterou jsem vždy zahrál následující tah na každou volnou souřadnici partie a nechal AI spočítat ohodnocení dané hry a tím jsem získal ohodnocení pro každou volnou souřadnici.

Vzhledem k velké početní výkonnosti a délce času jsem se rozhodl postupovat na etapy, kde v první části jsem ukládal celý výstup AI pro každou volnou souřadnici a až pak následně výstup upravoval dle vlastních potřeb pro využití trénování neuronových sítí.

Tím jsem získal přidanou hodnotu, kterou byla možnost využití dat více způsoby. Této výhody bylo využito hned ve dvou případech a tím se ukázala volba jako velmi dobrá i přes časovou náročnost výpočtu. Poprvé ve chvíli, kdy přišlo rozhodnutí, které rozsahy ohodnocení v jednotlivých pozicích uvažovat. Podruhé ve chvíli, kdy jsem potřeboval pročistit data. Do budoucna to umožní zkusit pokročilejší operace nad daty, zužitkování nejlepší sekvence následujících tahů a zároveň to umožní opakované úpravy.

### 4.1.1 Popis využitých programů pro tvorbu dat

K vypočítání dat byl použit program Embryo od českého programátora Fontána, který v letech 2019-2021 dominoval v gomocup turnaji umělých inteligencí v Gomoku a renju.

Scripty pro vytváření dat, učení neuronových sítí a vytvoření grafů, byly psány v pythonu. Zhruba polovina dat byla spočítána pomocí laboratoře umělé inteligence PEF (provozně ekonomická fakulta) zemědělské univerzity. Za tuto možnost, kterou mi škola poskytla, děkuji.

Druhá polovina dat byla spočítána na vlastním PC v průběhu posledních dvanácti měsíců. Následné učení Neuronových sítí a všechny testy byly prováděny na vlastním PC z důvodu nepravidelnosti času, který jsem v průběhu roku byl schopen věnovat projektu. Tato volba přinesla ztížení v podobě limitace výkonu daného parametry grafické karty.

Embryo po ukončení jednotlivých výpočtů vrací zprávu v následující podobě:

*MESSAGE depth H ev +V n N n/ms T tm 4 pv Xi8*

Kde H značí hloubku prohledávání, V je ohodnocení, T je čas výpočtu, N počet navštívených uzlů a za pv následuje nejlepší sekvence tahů, kde Xi8 značí křížek na pozicí i8. Sekvence končí dvěma hrozbami, po kterých je jistá výhra daného hráče.

Příklad z dat spojený k obrázku 10 – po bílém tahu h8:

*MESSAGE depth 20-40 ev 473 n 1677k n/ms 1708 tm 982 pv Xi6 Oh7 Xh9 Oi4 Xg3 Of2 Xe7 Of8 Xi9 Oi8 Xg8 Of7 Xe9 Og9 Xe5 Oe6 Xd5 Oc5 Xf3 Oe3 Xd4 Od6 Xk7 Oj6 Xf10 Oc6 Xf6 Od8 Xc3 Ob2 Xj8 Oh10 Xi3 Og5*

- Ohodnocení 473
- v hloubce 20 – 40
- s prohledáním 1677 tisíc nodů
- s průměrem 1708 nodů za milisekundu
- celková spotřebovaná doba na výpočet 982 milisekund
- nejlepší sekvence začínající i6 pro křížek (černý kámen), h7 (bílý kámen) atd.

### 4.1.2 Popis dat

Soubor her obsahoval 14032 partií a následným rozložením vzniklo 342041 jednotlivých pozic, které sloužily jako vstup pro embryo (dále jen AI).

Zápis originálních partií byl ve tvaru:

1. h8 g9 2. e8 a16 3. f7 h9 4. i8 f6 5. g7 h7 6. h6 h11 7. h10 f9 8. g10 g8 9. e10 f8 10. d8 i10 11. g12 j11 12. k12 a1 0-1,

kde čtvrtá souřadnice a16 není součástí hry. Data byla originálně tvořena pro jiný program, který sloužil pro renju zobrazení dat a z toho důvodu čtvrtý symbol ignoroval. Hra skončila rezignací černých kamenů 23. tahem na a1.

Na každý výpočet AI byla přidělena jedna vteřina, která se zdála být jako optimální pro oddělení nevhodných tahů a zároveň dostačující pro ohodnocení jednotlivých pozic. V rámci testu bylo provedeno i několik tisíc výpočtů s dvěma a deseti vteřinami. Výsledek nebyl odlišný až na pár vyjímek. Pouze zpřesnění ohodnocení bylo v zhruba pěti procentech přesnější. V ukázkové úloze níže je přímo patrné, proč se tak děje. AI už v době jedna vteřina spočítala 205 souřadnic v optimálním řešení a zbylých 11 souřadnic se k ideálnímu řešení blíží.

Data obsahovala hry z variant klasické piškvorky, pro a swap. Varianta long-pro je obsažena v hrách swap. Pro hlubší modelování by bylo důležité doplnit i variantu swap2, která se liší třetí možností druhého hráče.

Každou pozici jsem zaznamenal v podobě dvou jednodimenzionálních polí, první pro vstupní data a druhé pro očekávaný výstup.

Každé políčko ve vstupním poli mělo ohodnocení:

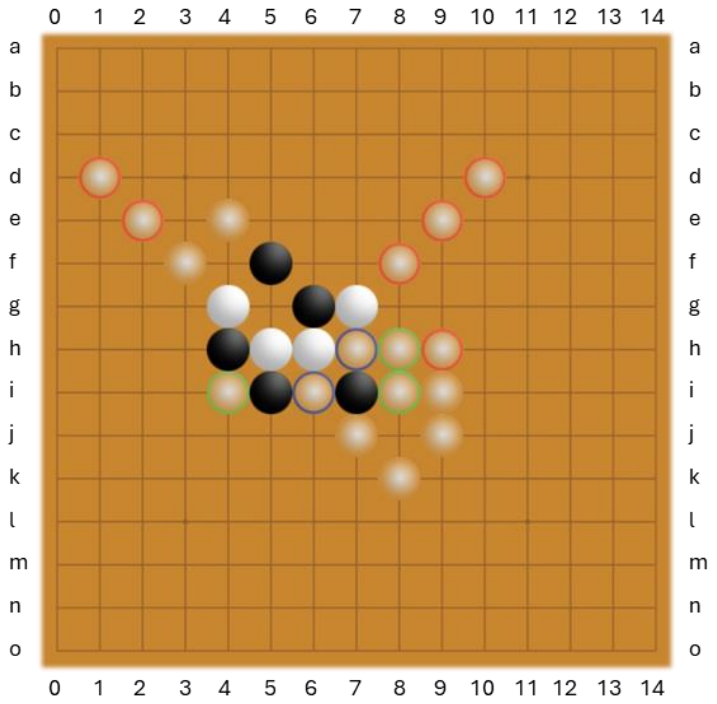
- 0 - v případě prázdného pole
- 1 - v případě bílé barvy
- 2 - v případě černé barvy

Každé políčko ve výstupním poli mělo ohodnocení:

- 0 - pro každé i-tý políčko, které ve vstupním poli nerovná se 0
- MESSAGE depth H ev +V n N n/ms T tm 4 pv Xx Oo …. - ostatní

#### 4.1.2.1 Ukázka jedné pozice

Pro ukázku jsem vybral pozici s identifikací 3722 z datasetu 9 kamenů.




*Obrázek 10: Ukázka pozice 3722. Šmouhy jsou znázorněné pozice souřadnic, které nejsou zahrnuty v množině 199 pozic souřadnic +M5. Červené kruhy znázorňují velmi špatné možnosti, šmouhy bez kruhů znázorňují špatné možnosti, zelené kruhy pak znázorňují zajímavé možnosti a modré kruhy nejlepší možné příležitosti.*

Pro vybranou pozici dostanem 199 krát ohodnocení +M5 pro následující množinu souřadnic: [a0 - a14, b0, - b14, c0 - c14, d0 - d9, d11 - d14, e0, e1, e3, e5 - e8, e10 - e14, f0 - f2, f4, f6, f7, f9 - f14, g0 - g3, g5, g8 - g14, h0 - h3, h10 - h14, i0 - i3, i10 - i14, j0 - j6, j8, j10 - j14, k0 - k7, k9 - k14, l0 - l14, m0 - m14, n0 - n14, o0 - o14] s následujícím tahem Xi8. Pro upřesnění, +M5 je z pohledu pozice desky po zahrání daného tahu z výše uvedené množiny. Nebo-li si můžeme říct, že pro vybranou souřadnici platí -M6. Tato množina zahrnuje všechny souřadnice průsečíků, které v případě zahrátí následujícím kamenem, budou následováni tahem na i8 a hra skončí za pět kamenů (+M5 - zahrnuje i8). K této množině můžeme přiřadit možné tahy na f8, e9, d10, které nezabrání následujícímu tahu na i8, ale hru protáhnou na sedm kamenů (+M7), neboť po i8 bude smět bílý kámen zahrát čtyři kameny v řadě a tím oddálit prohru o dva tahy.

Zbylé možnosti e2, d1, h9 (+M9), e4 (+M17), j7, j9, k8 (+M19), f3 (+M21), i8 (+M23), i9 (+M25), i4 (+M27), h8 (+473), h7 (+280), i6 (+258) znemožňují přímo zahrát 11. kámen hry na i8 a tím hru změní a prodlouží. V tuto chvíli se naskytly dvě možnosti, jak danou situaci zhodnotit z pohledu přípravy dat. Uvažujme například i4, který byl spočítán jako jistá prohra nejpozději za 27 kamenů a i6, který má ohodnocení -258. Z hlediska dokonalé hry by i6 byl brán jako mnohem lepší tah, neboť i4 je jistá prohra. Druhý pohled může být brán z pohledu patternu, kde i4 je obranný tah vyskytující se často jako jeden z nejlepších možných tahů, zatímco i6 je trojka a v podobných patternech vyjde často hůře v porovnání s druhou možností.

Hlubší analýzou bychom se dostali postupně k diskuzi, zda je lepší bránit a nechat soupeře vymýšlet útok, nebo naopak zahrát útočný tah po kterém jsou jen dvě možnosti kam může být zahrán další tah a obě vychází s vysokým ohodnocením pro druhou stranu.

V obou případech lze množinu souřadnic s ohodnocením +M5, +M7 a +M9 vyřadit z možných následujících tahů.

### 4.1.3 Úprava dat

V této fázi jsem měl opět několik možností, jak postupovat:

- Zvolit vždy pouze nejlepší tah a učit Neuronovy Sítě (NN) pouze hrát.
    - Vybrat pouze neproherní pozice
- Zvolit až x možných následujících tahů a učit NN hledat nejlepší možné tahy pro danou pozici.
    - Vybrat pouze neproherní pozice

V této práci jsem se zaměřil na druhou variantu, kdy jsem chtěl, aby se NN učily hledat přijatelnou množinu tahů v dané pozici, která lépe splňovala určený cíl pomáhat lidem v učení.

Velkou změnou vstupních hodnot byl menší trik. Vzhledem k tomu, že barvy se mění, docházelo u NN ke zmatení, za které barvy v určité chvíli hraje. Tento problém byl zlepšený změnou barev. Všechny pozice byly změněny tak, že na tahu byla vždy stejná barva (v mém případě černá).

Po určení, která data budu využívat, jsem musel data převést do tvaru, který místo “message” v podobě stringu, byla v číselné podobě. Tady přišla úvaha, kde jsem vybral nejlepší následující tah, kterému jsem přiřadil hodnotu 1 a ostatním možným tahům jsem přidal decimální hodnotu z intervalu <0,1>, kde rozdíl byl přeškálovaný podle hodnoty ohodnocení vrácené z výpočtu AI.

Tím jsem dostal pozice, které byly v číselné podobě, ale už ztratily přesnou informaci, kolik bylo ohodnocení pozice vůči celkové hře.

Následující fáze úpravy zahrnovala převedení dat do formátu TFRecord (Géron, 2019). V průběhu převádění dat jsem provedl poslední dvě úpravy. První se týkala očekávaného výstupu, kde došlo k přeškálování hodnot, aby výsledný součet byl roven 1. Druhá se týkala změny vstupu, kde jsem přeškáloval barvy kamenů z 2 (černá) na 1 a z 1 (bílá) na -1. Tato změna byla provedena z důvodu lepší symetrie pro NN.

Samotný formát pak obsahoval:

- Images = jednodimenzionální pole obsahující hodnoty vstupní pozice
- Labels = jednodimenzionální pole obsahující očekávaný výstup, kde součet výstupu = 1

Poslední fáze obsahovala převedení jednotlivých vstupů a výstupů do trojdimenzionálního pole, neboť konvoluční 2D layer očekává trojdimenzionální pole na vstupu (CNN-tensorflow, 2023).

#### 4.1.3.1 Čištění dat

Samotná data po provedení úprav obsahují takzvané “noisy” data, která často způsobují špatnou nebo dokonce žádnou konvergenci. Data je z toho důvodu dobré očistit.

V našem případě mohou být problémy těchto typů:

- Příliš mnoho možných přijatelných řešení
- Obsahují šest kamenů jako správné řešení způsobené chybou AI
- Nevyvážený počet kamenů daných úrovní

- Nevyvážený počet správných řešení pro dané souřadnice
- Rozdílné patterny pro nízký a vysoký počet kamenů na ploše¨
- Často opakující se pattern s nízkou váhou pro hru
- Tahy založené na přímé výhře, ale chybné z generalizačního pohledu
- Nedostatečný počet dat

Některé problémy, například nedostatečný počet dat, je v tomto případě přijatý fakt, který v případě úspěšných výsledků, lze odstranit dopočítáním pouze potřebných dat. Jednotlivé postupy výpočtů a testů postupně ukazovaly, že nejpodstatnější jsou data v rozsahu od pěti do dvaceti kamenů. I přesto, že data od 20 do 30 kamenů jsou důležitá, je potřeba brát v potaz důkladné čištění neboť obsahují často data, která jsou chybná pro nižší počet kamenů a zároveň obsahují patterny širšího pole. Z toho důvodu by bylo vhodné do budoucna napočítat mnohem více dat od 1 až do 30 kamenů a toto pozorování ověřit.

Důležitou informací je konec hry, který velmi často obsahuje sekvenci hraní takzvaných “čtyřek”, kdy útočící strana hraje svůj kámen takovým způsobem, že vytváří čtyři kameny v rozsahu pěti polí a soupeřící strana má pouze jednu možnost kam zahrát a to až do posledního svého tahu, kde už tah neovlivní délku hry a následným tahem útočící strany skončí hra.

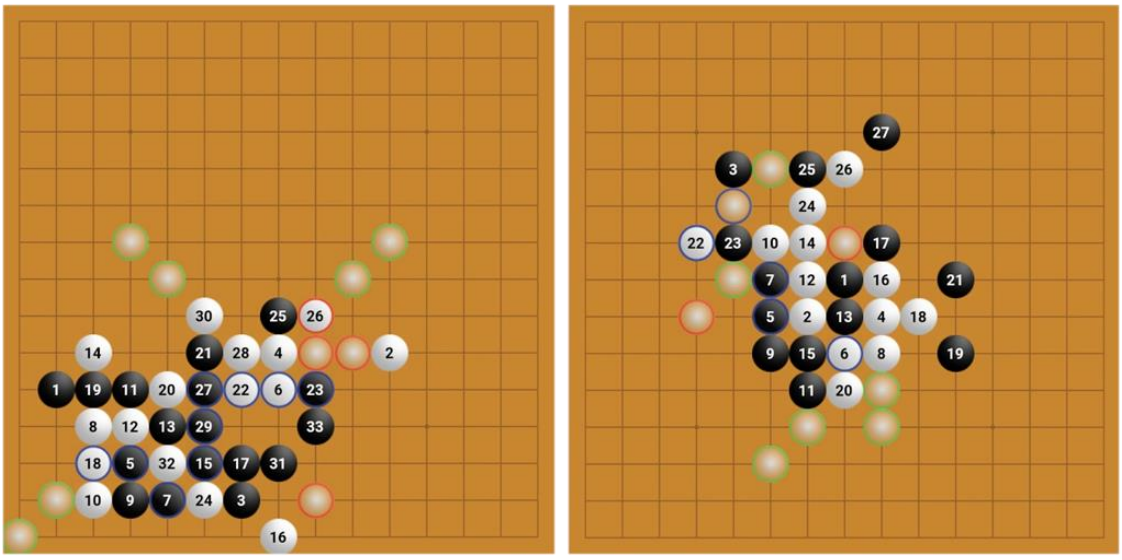
Tento postup se dá rozvinnout i na sekvenci tahů kombinace takzvaných “trojek” a “čtyřek”, ale pro ukázku a vysvětlení je dostačující útok přes “čtyřky”. Tento útok může být dlouhý od tří kamenů až do zaplnění desky. Z pravidla bývá útok častěji kratší délky a to v rozsahu 3 až 21 kamenů.

Pro tuto práci je tato informace velmi důležitá, neboť hraní čtyř kamenů v řadě je pattern, který se ve hře vyskytuje velmi často, ale bez dopočítání do konce není jednoduché určit, zda daná čtyřka je nutná. V opačném případě, zahráním špatné čtyřky v nevhodnou chvíli, se situace změní v negativním směru pro danou stranu.

Tento jednoduchý pattern vytvářel v datech velký šum a bylo obtížné problém dlouho identifikovat. Neboť výsledky na testovacích a validačních datech vypadaly velmi dobře, ale v reálné hře pak bylo patrné časté přeučení tohoto patternu na úkor všech ostatních.

Na základě těchto pozorování jsem přistoupil k provizornímu řešení, které popíšu v části umělá modifikace dat. Mohu pouze poznamenat, že touto úpravou jsem přišel o důležitou informaci z hlediska samotné hry, ale na modelech pak bylo znát mnohem lepší generalizování.

Pro upřesnění uvádím dvě ukázky. Na levém obrázku jsou tahy, které neovlivňují situaci hry označeny zelenými kruhy. Tyto tahy budou v originálních datech zahrnuty jako nejlepší tahy v danou chvíli. Plyne to z algoritmu výpočtu, kde se pomocí čtyřek dosáhne nejdelší prohry. Červené tahy značí možnosti, které jsou důležité z pohledu komplexnosti hry. Nejdůležitější tah v dané pozici není čtyřkou. Na obrázku v pravo, máme znázorněné čtyřky zeleně a červeně. K tomu pak máme modře znázorněný nejlepší tah, který je nutný zahrát. Červeně označujeme ty čtyřky, které hru pokazí, zeleně pak jsou označeny čtyřky, které hru prodlouží a po nich bude následovat opět modrý kámen. Můžeme vidět, že za dva případy hned 12 čtyřek bylo označeno jako přijatelné a v polovině případů i njelepší řešení dané pozice.




*Obrázek 11:Dvě ukázky znázorňující pattern čtyřek, který pak způsobí přeučení*

## 4.2 Učení Neuronových sítí

S nadsázkou si Neuronovy sítě můžeme představit jako skříňku, která má miliony koleček. Pokaždé, když nějaké kolečko trochu pootočíte, vypadne vám ze skříňky nový model se specifickými vlastnostmi.

Z tohoto důvodu jsem se v prvních pokusech inspiroval modely od tensorflow a z dostupných github modelů (CNN-tensorflow, 2023). Nepřekvapivě, modely nekonvergovaly, a tak přišel čas na hlubší studování a zjišťování, jak matematicky znázornit jednotlivé layery. K tomu bylo nutné zohlednit specifikaci dat, neboť počet dat byl velmi malý k teoretické množině dat daného problému a naopak obsahoval poměrně široké spektrum dat.

První velký pokrok přišel s rozhodnutím data rozdělit do menších datasetů podle počtu kamenů a učit NN postupně těmito datasety. V tu chvíli začaly modely konvergovat k hranici 40 %.

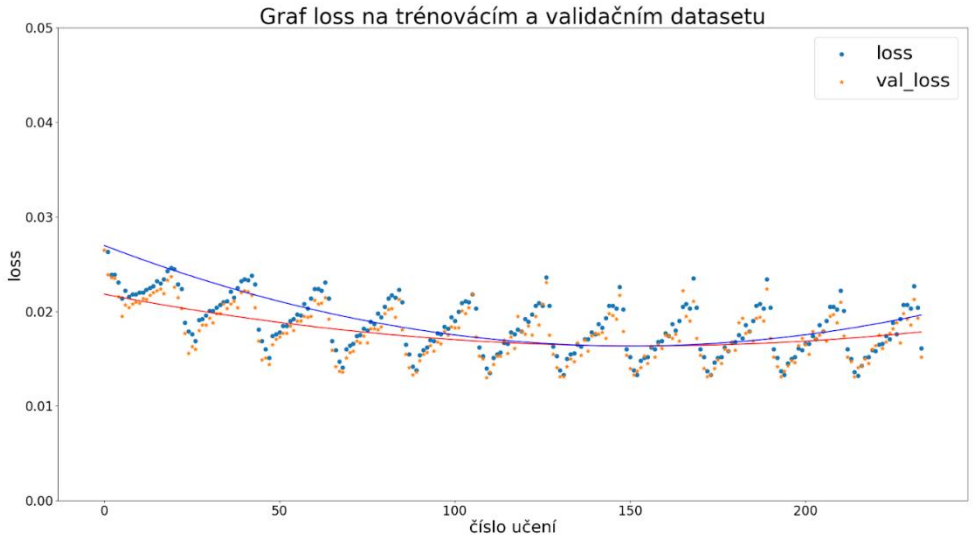
(přidat ukázku porovnání učení)

Hlavní parametry, které jsem zahrnul do mého testování:

- Počet layerů
    - Pouze dense layers
    - Convolutional layers zakončené dense layer
    - Convolutional layers zakončené dvěma dense layers
- Velikost jádra convolutional layer od 3x3 po 11x11, dodatečně 1x1
- Počet filtrů v jednotlivých layerech
- Aktivační funkce - relu, leaky_relu, linear, softmax, None, atd.
- Flatten a GlobalAveragePooling2D
- Dropout a kernel regularization
- Learning rate
- Padding
- MaxPooling2D

Za loss pro učení modelu jsem zvolil BinaryCrossentropy, neboť mi záleželo na hledání všech možných tahů v dané pozici a z toho důvodu jsem nemohl zvolit CategoricalCrossentropy, která by se mi hodila za předpokladu, že by pro každý vstup existovala pouze jedna jediná třída, do které by se daný výstup klasifikoval.

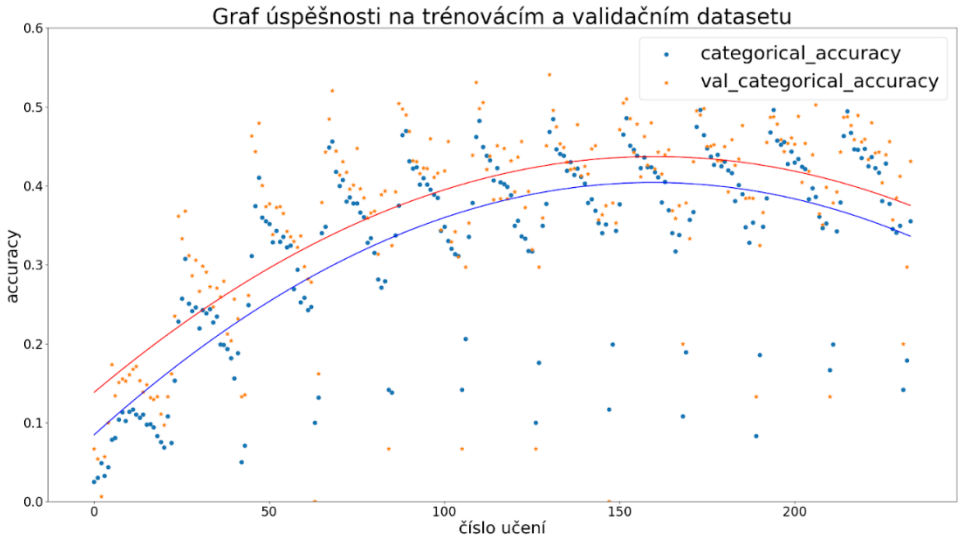
Další metriku, kterou jsem nastavoval byla accuracy, která ač nemá přímý vliv na učení modelu, stále nepřímo ovlivňuje učení modelu, neboť na základě accuracy často nastavujeme předčasné zastavení, změnění learning rate a v neposlední řadě má accuracy vliv na evaluaci modelu. V mém případě jsem zvolil CategoricalAccuracy, neboť BinaryAccuracy by chybně vracelo velmi vysoké úspěšné hodnocení a to jen díky tomu, že většina souřadnic je v očekávání výsledku nulová.




*Graf 1: Ukázka loss po odebrání první hodnoty pro možnost vidět "schody" jednotlivých datasetů. Nutné si uvědomit, že je graf velmi detailní.*

Graf loss na trénovacím a validačním datasetu, zahrnující model 1, ukazuje schody, které jsou způsobené jednotlivými datasety. Přidal jsem omezení šířky loss, kvůli první hodnotě, která byla velmi vysoká a z toho důvodu pak nebyly dobře vidět schody.

V průběhu pokusů jsem se zaměřoval na minimalizaci jednotlivých “špiček”, neboť jak je vidět v grafu, špičky naopak rostly a tím kazily celkové učení modelů. Jednalo se o datasety s pozicemi obsahujícími velmi málo kamenů. Obzvláš pozice s třemi kameny postupně divergovaly a jako jediný dataset se nedostaly pod chybu 0.02.




*Graf 2: Graf znázorňující úspěšnost jednotlivých učení*

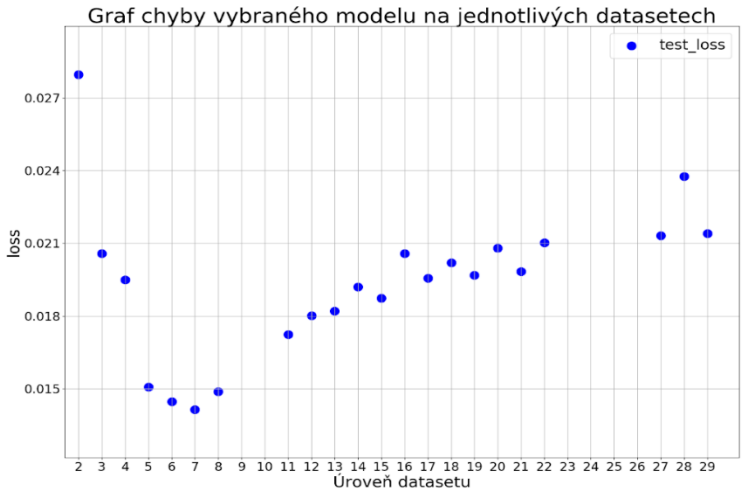
Graf ze stejného učení, ale pro znázornění úspěšnosti jednotlivých učení. Proložil jsem data křivkou polynomu druhého stupně. Červená barva je pro validační datasety a modrá pro trénovací datasety. Jev, kdy úspěšnost modelu je na trénovacích datech horší než na validačních datech, jsem vídal velmi často. Může to být způsobeno několika možnostmi:

- Komplexní data
- Nedostatek parametrů
- Příliš jednoduchý model
- Špatně rozdělená data

Při studování dat jsem vyškrtl pouze poslední bod, neboť komplexnější modely s velkým množstvím parametrů tento jev odstranily, ale měly pak mnohem větší tendenci přeučit se. Ve většině případech to pak dospělo k mnohem horší až nulové generalizaci.

Dále na grafu je opět vidět několik bodů, které se pohybují v rozmezí 0,05 až 0,2 i po mnoho iterací. Jedná se o datasety s pozicemi obsahující 2 až 4 kameny. Tato chyba může být způsobena i špatně spočítanými daty, neboť omezení jedné vteřiny na tah může být pro Embryo nedostačující. Z toho důvodu jsem dal počítat tyto datasety s větším časem, ale ani tak to situaci moc nezlepšilo. Hlubší zkoumání ukázalo, že na většinu pozic s dvěma kameny bylo potřeba často i několik desítek vteřin až minut, než byly výsledky adekvátní. Úplný

výpočet těchto pozic by přesáhl celkovou dobu spotřebovanou na všechny ostatní pozice. K takovému kroku jsem se rozhodl nepřistoupit.

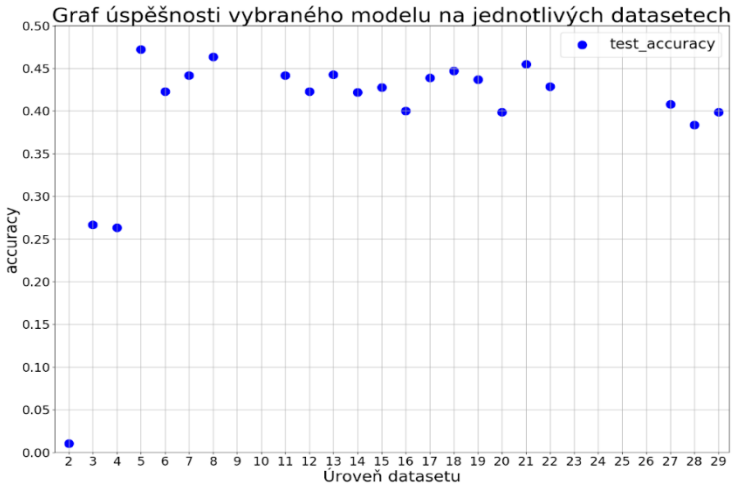



*Graf 3: Testovací chyba na jednotlivých datasetech*

Pro úplnost přidávám i grafy zobrazující výsledky modelu 1 na testovacích datech podle jednotlivých datasetů testovací sady. Chyba modelu na datasetu 2 je mnohem větší než na všech ostatních datasetech. Můžeme vidět velmi nízkou chybu u datasetů 5 až 8 a z toho lze usoudit, že trénování modelu v tu chvíli probíhalo na datasetu nízké úrovně.

Graf úspěšnosti modelu 1 na jednotlivých datasetech testovací sady ukazuje velmi špatnou úspěšnost datasetu 2 až 4. Kde obzvlášť dataset 2 byl téměř na nule.

Pro hodnocení modelů jsem pak používal vážený průměr chyby a úspěšnosti podle počtu dat v jednotlivých datasetech.




*Graf 4: Testovací úspěšnost na jednotlivých datasetech*

Na začátku trénování NN jsem zvolil nižší počet layerů (2-4) s malým počtem filtrů (1-16), které jsem postupně zvyšoval podle předchozích výsledků. Počet filtrů jsem volil vždy velikosti 2na x, kde x bylo kladné celé číslo včetně nuly. Následovala volba aktivačních funkcí. Pro výstupní layer jsem přiřadil softmax, kde výstup se sčítal do 1. U skrytých vrstev jsem dlouho používal relu, který je široce používán i přes mizící gradient. Následně jsem přešel k Leaky ReLU (Leaky Rectified Linear Unit), který umožňuje řešit problém s mizícím gradientem a často nazývaným "dying ReLU" a tím se nestane, že by neuron přestal updatovat své váhy a neuvízne v negativní části aktivační funkce. Zároveň umožňuje být méně citlivý na malé šumy ve vstupních datech. Díky tomu může trénování vést k mnohem více robustním modelům.

Další částí, kterou jsem podrobil testu byl padding. Ukázkové příklady modelů od tensorflow nedoplňují padding a tudíž se vstupní rozměry dat každým layerem zmenšují o n - 1, respektive m - 1 polí, kde n, m jsou rozměry jádra.

Jako příklad můžeme uvést vstupní data ve formátu: (15, 15, 1)

Pak

- Conv2D(x, (7, 7)) zmenší vstup do dalšího layeru na (8, 8, 1)
- MaxPooling2D((2, 2)) zmenší vstup do dalšího layeru na (4, 4, 1)
- Conv2D(x, (3, 3)) zmenší vstup do dalšího layeru na (2, 2, 1)

Další layery budou už velmi omezené. Samotný model se bude velmi rychle učit do určité úrovně (v našem případě do 20% accuray) a bude malé velikosti. Nicméně, jeho limity učení budou velké a ztráta informací po krajích desky bude markantní.

Z těchto důvodů jsem po krátké době přešel na padding="same".

Úplně první pokusy učení probíhaly čistě s modely zahrnující pouze dense vrstvy a měly tendenci se učit přímo dané varianty a chybělo jim dostatečné generalizování. Z toho důvodu jsem velmi brzo přešel na modely obsahující konvoluční vrstvy.

Prvních 100 úspěšných modelů neobsahovalo dropout ani batchNormalization. Tyto techniky zlepšení trénování jsem přidal až postupně, kdy všechny pokusy modelů nebyly schopny přesáhnout hranici 40% úspěšnosti na testovacích datech. Za pomoci těchto dvou technik pro zlepšení trénování jsem byl schopen trénovat širší škálů parametrů. Nebo-li zavedení obou částí pomohlo modelům nebýt tak citlivé na šumy a nepřesnosti v datech a tím jsem postupně konvergoval ke složitým strukturám modelů.

Ani výše uvedené techniky nepomohly k výraznému zvýšení úspěšnosti na testovacích datech. Nejlepší modely zahrnující 4-8 skrytých vrstev, se pohybovaly kolem 42% úspěšnosti na testovacích datech.

K velkému překvapení došlo ve chvíli, kdy v rámci testování jsem přidal AI složenou pouze z neuronových sítí (jeden z úspěšnějších modelů z první 100) do vlastní aplikace Gomoku Library. Pro jednoduchost jsem uvažoval vždy nejlepší tah z vrácených tahů, ale pro jistotu jsem umožnil po vrácení kamenu vidět všechny tahy, které neuronova síť vrátila i s přiřazeným ohodnocením. Z mého pohledu poměrně úspěšného hráče, hrála AI velmi špatně. Přesto jsem dal možnost AI vyzkoušet uživatelům aplikace a sesbírat první dojmy uživatelů. Reakce byly zajímavé. AI kupodivu dokázala často i vyhrát, ale z pohledu vývojáře ukázala mnoho nedostatků. Nebránění čtyřky nebo volné trojky často vedlo k zbytečné výhře a naopak časté nezahrání nebráněné čtyřky z obou stran nebo občas nezahrání přesně pěti kamenů v řadě vedlo k zbytečné ztrátě výhry. Tyto postřehy mi postupně ukazovaly, že není vhodné vždy spoléhat pouze na nejlepší tah a rozhodl jsem se kontrolovat pět kamenů v řadě pro obě barvy.

Moje hodnocení bylo pak upraveno, kdy jsem mohl po pár desítek hrách konstatovat, že s drobnou úpravou konce hry, byla neuronova síť schopna často vyhrát. Jedinou starost mi dělalo zjištění, že často chybí tah v množině možných tahů, který po hlubším zkoumání vedl k nejlepšímu řešení. Po konzultacích s odborníky na tuto problematiku mi bylo vysvětleno, že správně bych měl mít ještě jeden zdroj, který doplní tahy ve chvíli, kdy neuronova síť vrátí chybné, nebo dokonce žádné přijatelné řešení.

Tento problém jsem se rozhodl pro tuto práci řešit pouze povrchně, neboť problematika se tím rozšířila mimo vytyčené cíle této práce.

První krok bylo minimalizování tohoto problému. Zajistil jsem kontrolu, která doplnila obranu čtyřky, pětky, volné trojky a zároveň jsem trochu vylepšil ukončení hry. Druhým krokem jsem chtěl vylepšit neuronovy sítě takovým způsobem, aby se více zaměřovaly na generování pozičních tahů na místo přímého útoku přes trojky a čtyřky. Přímý útok jsem pak měl v plánu řešit pomocí alpha beta, která by neměla mít problém spočítat přímé výhry do určité hloubky.

V tu chvíli přišel problém, neboť neuronovy sítě se odmítaly učit hrát pozičně a dlouho to vypadalo na problém špatného modelu. Od chvíle, kdy jsem zahrnul do modelů dropout a batchNormalization a prozkoumal širší spektrum možností, začalo být vidět, že

problém se schovává jinde. Modely se sice naučily na lepší přesnost s menší chybou, ale generalizace zmizela. Podmínka, kdy tah musel být větší než 2 %, byla často moc přísná pro získání alespoň jednoho tahu. Naopak tahy typu čtyřky nebo volné trojky byly hrány s velkou přesností a vysokou pravděpodobností.

Vzhledem k omezenému času pouhých pár zbývajících měsíců, jsem musel začít opět zkoumat data a zjišťovat, proč se tato situace děje. Po hlubším zamyšlení bylo jasné, že data obsahují často čtyřku a trojku jako možné řešení. Tím, že uvažuji nejlepší možné tahy, vznikají situace uvedené na obrázku (obr. x) velmi často. Na dané ukázce trojka, ani čtyřka nejsou klíčové k výhře, ale zahráním těchto tahů se výhra v ideálním případě pouze oddálí o 2 kameny. Vzhledem k omezenému času na vytvoření jednotlivých pozic mohlo často dojít i k nepřesnosti, kdy Embryo našlo výhru, která ale nebyla optimální. Nestačilo spočítat, že daná trojka, ani čtyřka nejsou nutné zahrát. V obou případech pak volba nejlepších x možností v dané pozici zahrnovalo zahrátí čtyřky nebo volné trojky a tím došlo k velmi častému opakování čtyřky a trojky v daných pozicích. První reakce byla, že daná chyba nepůjde opravit.

Získané modely, obzvlášť z první stovky úspěšných modelů byly schopny hrát velmi zajímavě a se spojením algoritmu prohledávání do hloubky mohly stačit ke splnění požadavků na vytvoření modelu, který bude schopen pomáhat učit a rozvíjet logické myšlení lidí. Horší už to bylo s rozhodnutím, zda daný nápad s daty je dobrý nebo aspoň použitelný.

Tato chvíle dobře ukazuje, jak je často výhodné, problémy řešit v týmu a ne jako jednotlivec. Jedna tříminutová diskuze s kolegou, kdy jsem téměř mluvil jen já, mi přinesla nápad, jak problém řešit.

Data obsahují hodně čtyřek, ale jak už jsem uvedl dříve, ty mohu, v reálné hře, snadno doplnit pomocí jednoduché validace a pomocí algoritmu prohledávání do hloubky ohodnotit, zda se jedná o nutný tah. V kombinaci se zkušenostmi z praxe Gomoku, je známé pravidlo: “nehraj čtyřku, pokud není nutná”. Toto pravidlo se dá rozšířit na trojky a další tahy, ale pro náš problém bude dostačující a mnohem jednodušší vysvětlit pouze základní “nepsané” pravidlo, které ovlivňuje hru neustále a je součástí ukončení hry. Ve většině případů se jedná o sekvenci těchto tahů, které neumožňují druhému hráči více možností než jednu až do předposledního kamenu, kde má hráč více možností, ale už neovlivní délku hry.

Vysvětlení pravidla “nehraj čtyřku, pokud není nutná”:

Uvažujme, že hráč A hraje za černé kameny a hráč B hraje za bílé kameny.

Uvažujme, že hráč B nemá na desce žádnou možnost zahrát 5 kamenů v řadě následujícím svým tahem.

Uvažujme, že hráč A má možnost zahrát čtyři kameny v řadě z jedné strany bráněné bílým kamenem. Můžeme to zapsat ve tvaru ?XXXO. Kde ? značí volné místo. Aby vznikla hrozba, která bude nutit hráče B bránit, musí být vedle ? další volné místo a tím získáme ??XXXO. Pro upřesnění, zleva od otazníků se nesmí nacházet X, neboť pak by vzniklo 6 v řadě a hrozba na místech označených ? by v tomto směru nevznikala.

Pak platí, že hráč A má přesně 2 možnosti kam může zahrát v danou chvíli, aby v dané ose vznikla hrozba, kterou bude muset hráč B řešit.

Následně platí, že ať se hráč A rozhodne hrát na první nebo druhý otazník, hráč B bude muset zahrát na daný otazník, který nebyl vybrán hráčem A.

Tím se dostáváme k několika hlavním poznatkům:

- čtyřku zahrajeme, soupeř má pouze jednu možnost kam hrát a nemusí vynaložit žádné úsilí na následující tah.
- každou čtyřkou přidáme 2 kameny na plochu a zmenšíme možný prostor pro kombinaci
- Bez znalosti přímé sekvence tahů, je 50% šance, že za pár tahů budeme potřebovat zahrát na druhý otazník, kde už bude soupeřův kámen.
- Pokud je možné zahrát čtyřku až v dalším tahu, aniž by soupeř nemohl situaci přerušit následujícím tahem bez následků, pak není nutné hrát čtyřku hned.

Tyto poznatky jsou velice cenné a při diskuzi, uvedené výše, jsem si uvědomil, jak zkusit naučit toto nepsané pravidlo neuronovu síť a tím vyřešit problém, který se mi po dobu několika měsíců nedařilo eliminovat.

### 4.2.1 Umělá modifikace dat

Na začátek této podkapitoly je důležité zmínit, že každá větší modifikace dat vyústila v nutnost opětovného hledání správných parametrů pro modely. Konvergence u modelů z předešlých pokusů buď úplně zmizela, nebo se výrazně zhoršila.

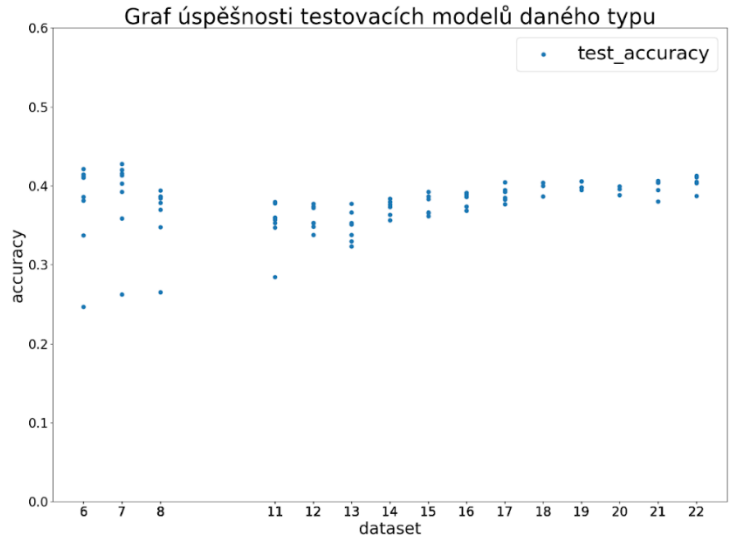
Za jako první modifikaci dat bych považoval výběr pozic pouze obsahující až n možností, neboť tím omezuji neuronovu síť o možnost vidět pozice, které mají hodně možností, kam lze zahrát další kámen.

Mezi další modifikace dat, které jsem v průběhu vyzkoušel, patří omezení pouze na n možných datasetů. Vybral jsem skupiny datasetů, které jsem postupně přidával do učení neuronovy sítě. Osvědčilo se, učit neuronovy sítě nejdříve na datasetech zahrnujících pozice s počty kamenů od 3 do 10. Postupné přidávání datasetů až k pozicím s 20 kameny umožnilo učit robustnější a komplexnější modely.

Pokusy o zamíchání dat nevedly nikdy k dobrému konci. Může to být způsobeno malým vzorkem dat, kdy pak jednotlivé batche obsahovaly velkou škálu rozdílných patternů. Dá se předpokládat, že existuje model, který by se dal naučit tímto způsobem, ale pro naše účely je způsob postupného učení datasety s pozicemi podle počtu kamenů mnohem efektivnější a účelnější.

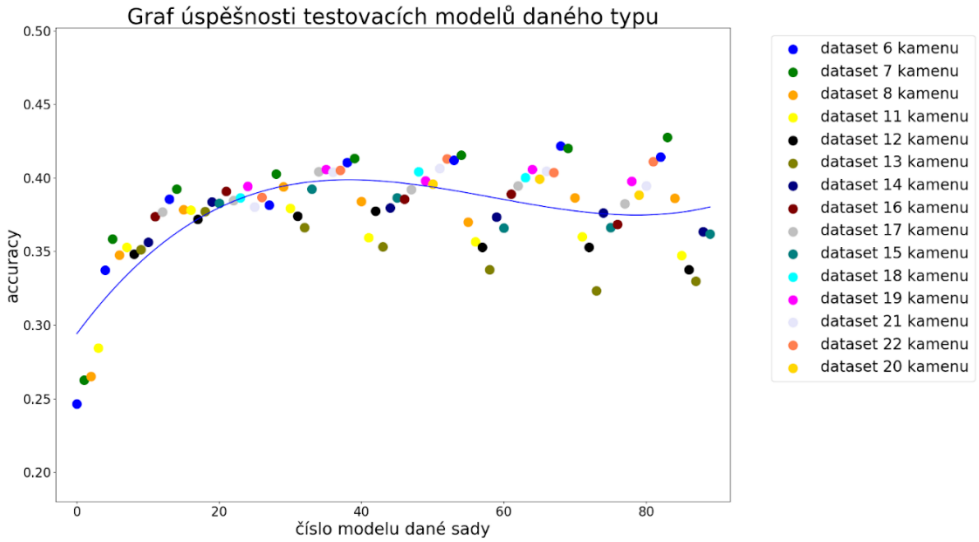
Za zmínku stojí i vyzkoušení omezení jedné barvy a učení pouze daty jedné strany. Tento pokus byl čistě z experimentálního hlediska a nepřinesl žádné kladné výsledky z pohledu mého cíle.

Zajímavější byl experiment, kdy jsem vyselektoval pouze neproherní pozice. Tímto krokem jsem opět zmenšil počet dat, ale modely, které jsem získal, měly pozitivní rysy a generalizace byla přijatelná. Tento experiment jsem testoval z podstatného důvodu, kdy běžně dostupné AI v Gomoku hrají při zjištěné prohře tahy, které hru protáhnou, ale neovlivní samotný důvod vzniklé prohry. Pokus ukázal, že tento fakt, který je všeobecně známý a je způsobený algoritmem prohledáváním do hloubky, má na data velký vliv. Z tohoto experimentu jsem usuzoval, že většina pozic nad 20 kamenů má velký potenciál být matoucí pro neuronovy sítě a spadat do šumu v datech. Pozitivní výsledek tohoto experimentu je i získání nejlepšího modelu, který vychází právě z tohoto testování.




*Graf 5: Úspěšnost modelů na testovacích datech rozdělených dle úrovní datasetů*

Graf x nám ukazuje datasety s pozicemi obsahujícími daný počet kamenů a modré tečky nám znázorňují jednotlivé modely, které splnily podmínku 40% úspěšnosti na validačních datech a tím se dostaly mezi modely daného typu pro testování na testovací množině dat. Osa y pak ukazuje úspěšnost jednotlivých modelů na testovacích datech. Graf ukazuje hned několik zajímavých pozorování k zamyšlení. Prvním pozorováním jsou bezesporu chybějící datasety 9, 10 a 23 až 29. Opravdu žádné učení sítě pomocí datasetů 9 a 10 nesplnilo hranici? Důvod byl už při omezování dat, kdy datasety 9 a 10 neprošly podmínkami a z toho důvodu pak chyběly v učení i testování. Naopak u datasetů nad 25 už se jednalo o nesplnění hranice úspěšnosti na validačních datech. Dalším pozorováním je nečekaně úspěšné hodnocení datasetů s nízkým počtem kamenů. V porovnání s ostatními experimenty to byla velmi nečekaná věc, neboť ostatní experimenty měly problémy s datasety obsahujícími pozice s nízkým počtem kamenů. Tento jev se promítl i do počtu učení s nízkými datasety, které častěji uspěly na validačních datech a dostaly se do testovací části. Pro lepší představu uvádím graf v zobrazení s pořadím jednotlivých modelů za sebou. V grafu byla vynechána všechna učení, která neuspěla na validačních datech. Graf jsem proložil křivkou polynomu třetího stupně. Opět je krásně vidět, jak datasety 6 a 7 postupně rostou v úspěšnosti, zatímco datasety 11, 12 a 13 stagnují. Můžeme předpokládat, že data obsahují těžší vzorce a v případě našich dat mohlo dojít k velkému rozdílu kvůli malé množině trénovacích dat.




*Graf 6: Graf úspěšnosti jednotlivých datasetů ku pořadí učení*

Největší zlom přišel ve chvíli, kdy jsem se rozhodl odstranit čtyřky z očekávaných tahů v daných pozicích. Při odstraňování jsem udělal chybu a nerozlišoval, kdo je v danou chvíli na tahu. Tím jsem odstranil i případné obrany trojek nebo čtyřek. Tahy jsou přímo dané a opět spadají do všech tahů, které by měly být spočítány za každých okolností. Z toho důvodu jsem se rozhodl prozatím pokračovat s těmito daty a nechat chybějící tahy na alpha-beta algoritmu. Neboť mi tato úprava umožnila se zaměřit na patterny typu takzvaných "pozičních" tahů.

Jedná se o takové tahy, které:

- nevytváří volnou trojku, ani čtyřku
- vytváří možnost útoku následujícím kamenem stejné barvy
- mohou zkrátit přímý útok
- jsou extrémně těžké
- nejdou hrát v případě volné trojky nebo čtyřky druhé barvy
- k pochopení smyslu daného tahu vyžadují pochopení všech náznaků útoku
- těžko se identifikují přímo v alpha-beta algoritmu

Tyto tahy jsou nesmírně důležité pro samotnou hru a lidem dělají obrovské problémy a z toho důvodu by bylo dobré mít AI, která by "poziční" tahy lidem zvýrazňovala a do budoucna i přímo ukazovala spojení s jednotlivými hrozbami útoku.

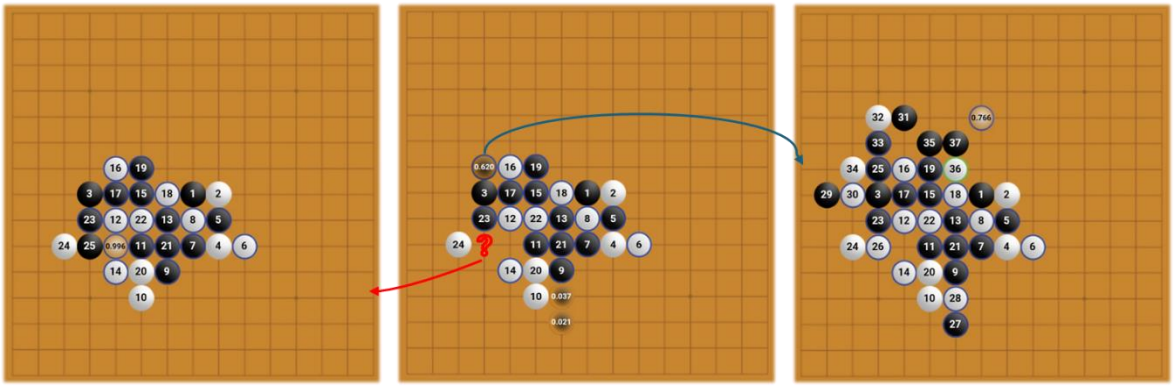
Z hráčského hlediska mám nasbírané zkušenosti z mnoha turnajů a mohu konstatovat, že jen pár lidí na světě dokáže poskládat aspoň dva poziční tahy po sobě při hře v pozici, kterou dříve neviděli. Více jak dva poziční tahy už je obrovská rarita, která není běžná ani mezi nejlepšími hráči světa. Uvažuji pouze takzvané "koncovky" her.

Moje druhá zkušenost je nasbíraná od řešitelů mých úloh, kteří mi opakovaně oznamovali, že úlohy na "poziční" tahy jsou pro ně velmi těžké, téměř vždy neintuitivní a často nepochopitelné i po vyřešení. Tyto zpětné vazby nejen potvrzují důležitost těchto tahů, ale i nutnost programů, které by byly schopny ukazovat lidem tyto tahy a zároveň jim dávat i vysvětlení pokud možno s co nejlepšími informacemi k pochopení daného tahu.

Z těchto důvodů jsem uvítal možnost rozdělit data ještě detailněji a zaměřit se přímo na nejtěžší části hry, které by měly zlepšit výpočty AI, zpřesnit koncovku hry a umožnit tak přiblížit se k vytyčenému cíli.

Obzvlášť u této modifikace dat došlo ke změně struktury modelů. Staré struktury nefungovaly a muselo se opět odhadovat, které modely by mohly nejlépe vyhovovat na řešení daného problému a zároveň by byly schopny dobře generalizovat. Tento problém se dal očekávat, neboť tahy, které zvedaly úspěšnost modelů jsem přímo vyřadil z dat touto úpravou.

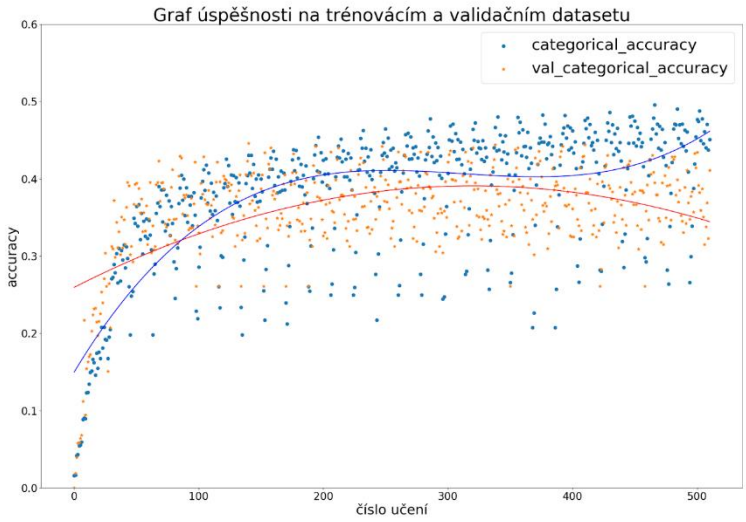
Zároveň jsem musel vytvořit generátor tahů, které jsem z dat odstranil. Výhodou bezesporu bylo zajištění stoprocentní jistoty existence tahu v dané množině. Na následujícím obrázku můžeme vidět, že neuronová síť, která se učila na všech původních datech (model



*Obrázek 12: Prostřední deska znázorňuje pozici s ohodnoceními modelu. Levá deska znázorňuje ideální pokračování. Pravá deska pak ukazuje ideální posloupnost po zvolení tahu určeného modelem.*

1) nemá správný tah v množině přípustných tahů a z toho důvodu oddálí výhru

a z jednoduchého řešení vytvoří komplexní počítací úlohu. Pro lepší pochopení jsem přidal ideální sekvenci tahů špatném 25. tahu.




*Graf 7: Graf úspěšnosti na trénovacích a validačních datech v závislosti na jednotlivých učeních*

Tento způsob modifikace dat umožnil i mnohem hlubší sítě. Jako ukázku přidávám graf z učení neuronovy sítě s 10 skrytými vrstvami v konvoluční části modelu. Opět můžeme vidět postupné přeučení trénovacími daty. Přesto tu vzniká zajímavé pozorování, kdy můžeme vidět, že některé datasety z trénovací množiny se pohybují hluboko pod křivkou. Opět se jedná o datasety s menším počtem kamenů. Model vznikl při trénování 19. datasetu a 511. iteraci.

### 4.2.2 Nejlepší modely

Nejlepší model z prvních dvou stovek zajímavých modelů: Model: "sequential" 1

```
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d (Conv2D)              (None, 15, 15, 128)       15616
 conv2d_1 (Conv2D)            (None, 15, 15, 8)         50184
 conv2d_2 (Conv2D)            (None, 15, 15, 8)         3144
 conv2d_3 (Conv2D)            (None, 15, 15, 8)         3144
 flatten (Flatten)            (None, 1800)              0
 going_to_end (Dense)         (None, 64)                115264
 ouputlayer (Dense)           (None, 225)               14625
=================================================================
  Total params: 201,977  Trainable params: 201,977
  Non-trainable params: 0
```

Velikosti kernel jednotlivých layerů: (11, 11), (7, 7), (7, 7), (7, 7)

**Úspěšnost na testovacich datech: 42,77%**

V kontrastu s druhým nejlepším modelem: Model: "sequential" 2

```
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d (Conv2D)              (None, 15, 15, 32)        3904
 conv2d_1 (Conv2D)            (None, 15, 15, 32)        82976
 conv2d_2 (Conv2D)            (None, 15, 15, 32)        82976
 flatten (Flatten)            (None, 7200)              0
 going_to_end (Dense)         (None, 64)                460864
 ouputlayer (Dense)           (None, 225)               14625
=================================================================
  Total params: 645,345 Trainable params: 645,345
  Non-trainable params: 0
```

Velikosti kernel jednotlivých layerů: (11, 11), (9, 9), (9, 9)

**Úspěšnost na testovacich datech: 41,11%**

Tyto modely, kromě dobré úspěšnosti na testovacích datech, ukázaly možnost velmi dobré generalizace na neznámých datech při reálných hrách s hráči a byly výsledkem postupného zkoumání, kde se velmi rychle ukázaly tři až čtyři konvoluční vrstvy ideálním počtem pro učení. Dvě vrstvy se dokázaly též učit, ale výsledek byl zhruba o 10 % horší na testovací sadě dat než u třech nebo čtyř konvolučních vrstev. Naopak pět konvolučních vrstev už nekonvergovalo při žádném z pokusů až do chvíle, kdy jsem přidal techniky pro lepší stabilitu trénování v podobě dropout, kernel regularizace a batchNormalization. Následně jsem byl schopen mít až šest konvolučních vrstev, ale rozšíření na tolik vrstev už vyžadovalo velkou výpočetní kapacitu a stávalo se mnohem těžším určit počet filtrů

k jednotlivým vrstvám. Samotná síť pak začala být velmi náchylná jak na přeučení, tak i na neučení.

Uvedené dva modely mají v kontrastu rozložení počtu parametrů v jednotlivých částech sítě způsobené rozložením počtu filtrů v jednotlivých vrstvách konvoluční části modelu. První model má velký počet filtrů v první vrstvě modelu, následovaný třemi vrstvami s malým počtem filtrů, zatímco druhý model má všechny tři konvoluční vrstvy se stejným počtem filtrů, které jsou mnohem vyšší než filtry ve skrytých vrstvách prvního modelu a díky tomu má druhý model vysoký počet parametrů při přechodu z konvoluční části modelu do dense části, kde

15 * 15 = 7200 * 64 = 460800 + 64 = 460864 = počet parametrů.

Vzhledem k tak vysokému počtu parametrů u layeru going_to_end je překvapivé, že nedošlo k přeučení, obzvlášť, když nebyly použity žádné techniky pro lepší zajištění učení.

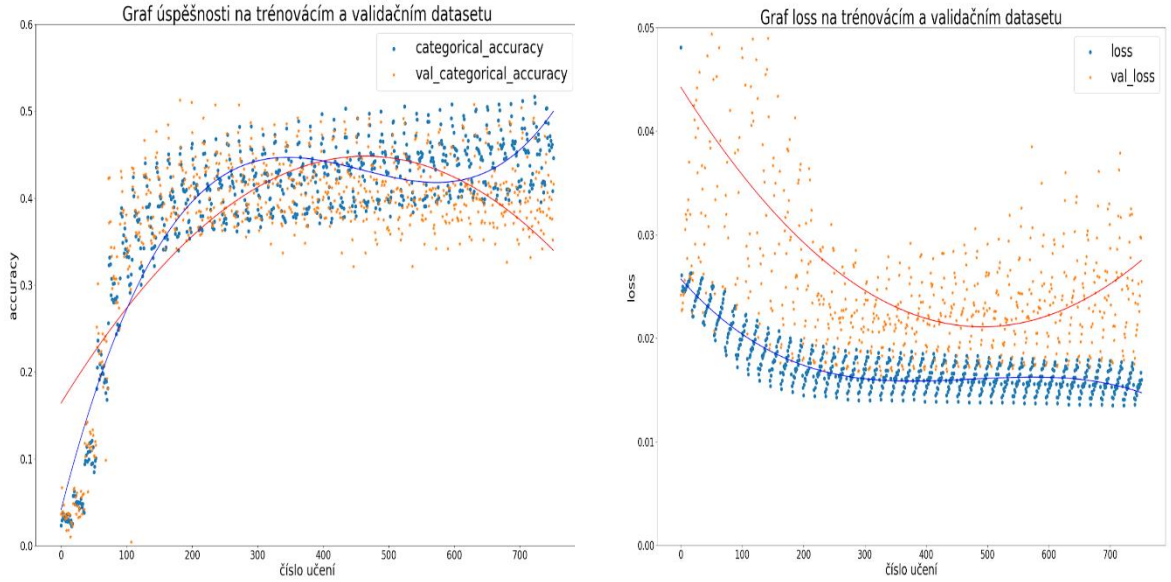
Ukázka modelu s více vrstvami, ale mnohem méně parametry: Model: "sequential" 3

```
  Layer (type)              Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 15, 15, 64)        1664
 batch_normalization         (None, 15, 15, 64)        256
 conv2d_1 (Conv2D)           (None, 15, 15, 16)        9232
 batch_normalization_1       (None, 15, 15, 16)        64
4x
 conv2d_2 (Conv2D)           (None, 15, 15, 16)        2320
 batch_normalization_2       (None, 15, 15, 16)        64
 conv2d_6 (Conv2D)           (None, 15, 15, 1)         145
 batch_normalization_6       (None, 15, 15, 1)         4
 flatten (Flatten)           (None, 225)               0
 going_to_end (Dense)        (None, 64)                14464
 dropout (Dropout)           (None, 64)                0
 ouputlayer (Dense)          (None, 225)               14625
  =================================================================
  Total params: 49,990 Trainable params: 49,700
  Non-trainable params: 290
```

Velikosti kernel jednotlivých layerů: (5, 5), (3, 3), (3, 3), (3, 3), (3, 3), (3, 3), (3, 3).

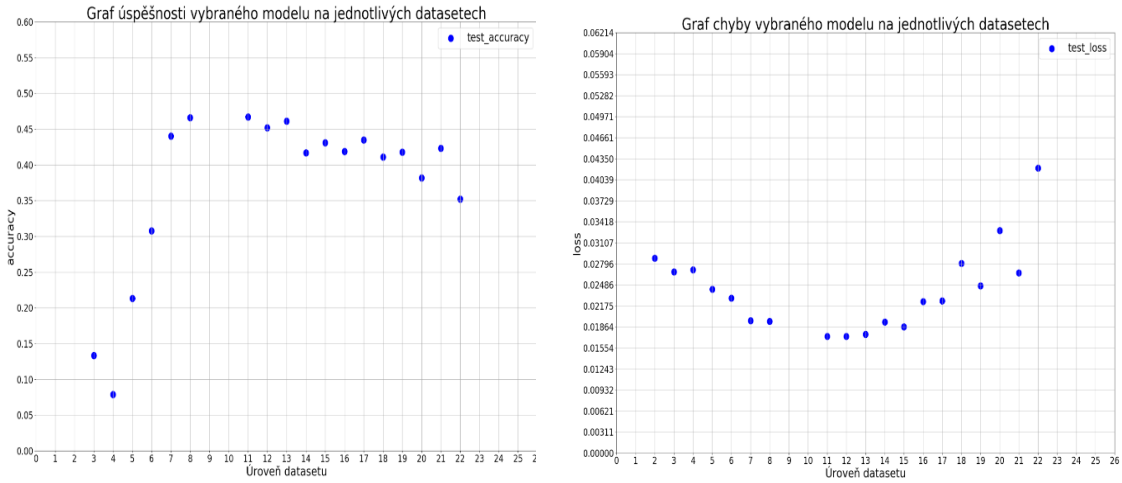
**Úspěšnost na testovacich datech: 41,41%**

V testech jsem zkusil i 9 konvolučních vrstev, které byly schopny klasifikovat testovací data s 40 %, ale v tomto případě už bylo patrné přeučení patternu čtyřek. Problém mohl být i ve velmi malém počtu parametrů. Neboť hlubší síť se učí mnohem hůře. Z grafu úspěšnosti pak můžeme vidět, že úspěšnost na trénovací sadě dat stoupá na rozdíl od úspěšnosti na validační sadě dat, kde je vidět stagnování a do jisté míry divergence. Graf chyby pak ukazuje pozvolné klesání chyby na trénovací sadě dat a velmi se měnící a zvětšující se



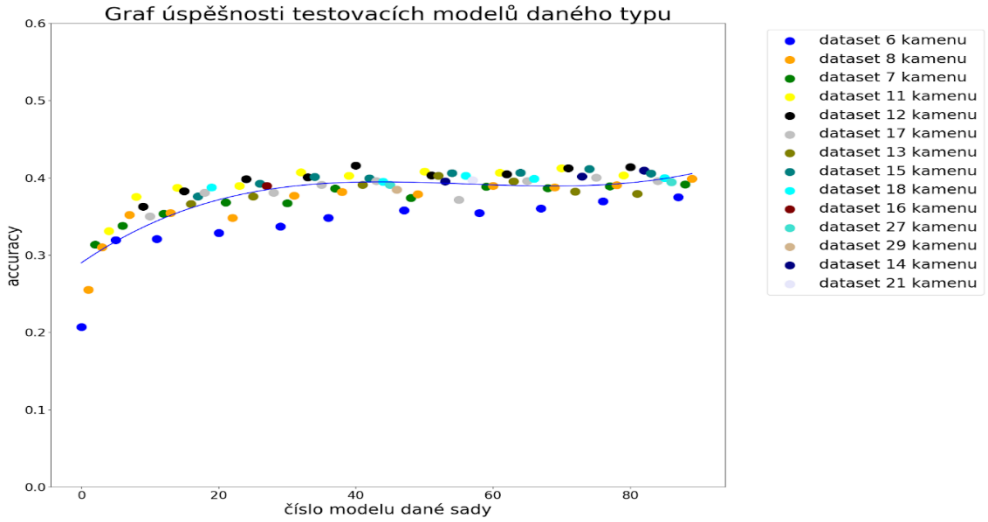

*Graf 8: Grafy úspěšnosti a chyby hlubší sítě s 9 konvolučními vrstvami*

rozptyl chyby na validační sadě dat. Z obrázku můžeme usuzovat, že ideální bylo skončit učení už kolem iterace 350. Zhruba od daného místa začala síť divergovat (přeučení/negeneralizace). Vzhledem k způsobu učení, bylo velmi obtížné implementovat ideální zastavení učení sítě. Příliš přísná měřítka byla schopna ukončit síť velmi brzy a po zpětné analýze jsem usoudil, že nejlepší je mít ukončení ve chvíli, kdy chyba začne velmi oscilovat, přesáhne hodnotu 0.2 a zároveň se nejedná o prvních x iterací. Tato podmínka byla dostačující v případech, kdy jsem učil malé sítě. U velkých sítí se i tato podmínka ukazovala jako velmi přísná. Problém ideálního ukončení popisuje ve své práci i Kriesel, kde zmiňuje, že jistotu nebo přesvědčení nám dodává situace, kdy model dosahuje opakovaně stejných výsledků (Kriesel, 2007).




*Graf 9: Grafy znázorňující chybu a úspěšnost vybraného modelu na jednotlivých úrovních datasetů*

Tato struktura uvedená u modelu 3 měla opět problém s daty nízké úrovně a tato skutečnost byla umocněná modelem, který byl naposledy učen na datasetu úrovně 12 a z toho důvodu se nacházel poměrně blízko datasetům s nízkým počtem kamenů. Číslo

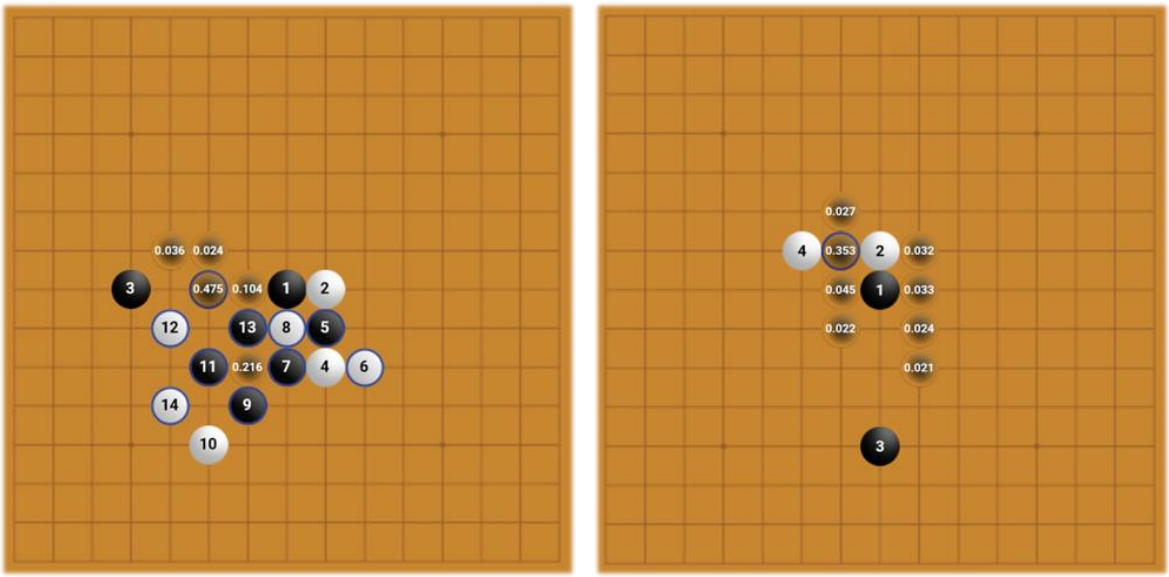



*Graf 10:Úspěšnost jednotlivých modelů vůči modelu při kterém došlo k uložení*

iterace bylo 257 a z grafů výše můžeme usuzovat, že jsme se blížili k ideálnímu bodu učení mezi trénovacími a validačními daty. Z grafu úspěšnosti testovacích modelů můžeme vidět datasety s 11 a 12 kameny nad křivkou, zatímco datasety s 6 až 8 kameny se nacházejí pod křivkou.

## 4.3 Spojení Alpha-Beta s Neuronovými sítěmi

Neuronovy sítě se často používají v kombinaci s Monte Carlo Tree Search (MCTS) algoritmem, který jsem vysvětlil v metodách umělé inteligence v teoretické části. Tuto variantu jsem zvolil i ve své bakalářské práci (Muzika, 2017), kde výsledek nebyl příznivý. Nejlepší programy v Gomoku na světě dlouho stavěly na alpha-beta algoritmu a to mě postupně motivovalo a lákalo k vyzkoušení spojení alpha-beta s neuronovými sítěmi. Cíl neuronových sítí byl zaručit ideálně všechny vhodné možnosti tahu v dané pozici. K tomu se předpokládalo seřazení, které mělo umožnit alpha-beta rychle prohledat a zkontrolovat všechny možnosti a upravit seřazené tahy na základě hloubkového výpočtu. Pro lepší představu uvádím ukázku dvou pozic, kde v první je pozice založena na přímé výhře a neuronová síť s velkou pravděpodobností může zvolit špatně a v našem případě opravdu zvolila špatně a oddálila svoji výhru o velké množství tahů. V druhé pozici se naopak nacházíme v situaci, kde neuronová síť podle vzoru zvolí správný tah, zatímco alpha-beta by potřebovala vteřiny, minuty až hodiny na správný výpočet. Dokonce tyto teorie byly stavebním kamenem úspěchu hráče Attily na mistrovství světa Gomoku v Tallinu 2013.




*Obrázek 13: Levá deska ukazuje pozici vhodnou pro alpha-beta. Pravá pak ukazuje pozici vhodnou pro neuronové sítě.*

Z těchto důvodů se jeví spojení neuronových sítí s algoritmem alpha-beta jako zajímavá možnost, kde očekávaný výstupní program bude na začátku hry spoléhat převážně na výstup neuronových sítích a s přibývajícími kameny bude přebírat váhu alpha-beta, která bude usměrňovat a zpřesňovat nabízené tahy sítí.

### 4.3.1 Testování alpha-beta s Neuronovými sítěmi

V rámci testování jsem se rozhodl vedle testování programu proti dobrovolníkům, vyzkoušet i porovnat model neuronových sítí (model A) proti stejnému modelu, který využíval alpha-beta prohledávání (model B). Vzhledem k časové náročnosti jsem omezil hloubku alpha-beta na pouhých pět kamenů. Zároveň alpha-beta nebylo umožněno využívat už předpočítané pozice z předešlých tahů. Oba modely obsahovaly kontrolu přesně pět v řadě a bránění čtyřky soupeře.

Test obsahoval 200 náhodně vygenerovaných začátků, kde první kámen byl vygenerován ve vnitřním čtverci 11x11. Druhý kámen byl vygenerován v těsném okolí prvního kamene a poslední kámen byl vygenerován bez omezení. Každý začátek jednou začínal model A a jednou model B. Z toho důvodu test obsahoval 400 her.

Zautomatizování procesu hraní přivedlo na světlo několik chyb v kódu a ukázalo sílu testů a automatizace v praxi. Za zmínku stojí neočekávané selhání testu po několika desítkách her, kdy najednou došlo k přetečení pole o 1 v místě těsně po kontrole délky. Po hlubším zkoumání byla objevena chyba, kdy kontrola čtyřky na hlavní diagonále přetekla na políčko za poslední řádek, neboť chybělo "rovnítko" v podmínce hlavní diagonály.

Výsledek testu dopadl dle očekávaní velmi dobře pro model B, který měl k dispozici základní alpha-beta a vyhrál v počtu 348:0:52. Kde 348 byl počet vyhraných her, nula remíz a 52 prohranných her. Model B tak získal 87% her.

Z hlubšího zkoumání jednotlivých začátků, bylo patrné, že model A byl schopen mít šanci vyhrát hru v pozicích, kde kameny byly blízko sebe a tím se podobaly datům, které se nacházeli v trénovacích datech.

Lze očekávat, že s přidáváním počtu povolené hloubky a umožněním využití předpočítaných pozic z předešlých tahů, bude procentuální úspěšnost modelu B stoupat. Na druhou stranu může nastat situace, kdy model B spočítá prohru už od úplného začátku. Pak bude záležet už jen na neuronových sítích, zda dokáží najít výhru i bez alpha-beta prohledávání. Za takových okolností by model B byl schopen prohrát aspoň jednu hru i s velmi velkým výpočetním výkonem modelu A. Taková situace je velmi málo pravděpodobná, ale není nulová.

Při hlubším prohledávání by už bylo vhodné měřit i délky jednotlivých her a tím lépe ohodnocovat kvalitu implementace alpha-beta v kombinaci s neuronovými sítěmi.

# 5 Výsledky a diskuse

V jednotlivých částech praktické části jsem uvedl několik výsledků a ukázek nejlepších modelů, které se mi podařilo naučit do této chvíle. Pro lepší představu doporučuji pročíst podkapitolu Nejlepší modely.

Pro ucelení představy výsledků musím nejdříve upřesnit, že úspěšnost byla uvedena pro lepší pochopení modelů a není správným měřítkem pro určení kvality. Tím je měření chyby modelu. Zda model bude vracet zpátky správné tahy, které pak pomocí algoritmu alpha-beta upřesním, je potřebné znát k správnému hodnocení. K tomu bychom potřebovali vlastní funkci úspěšnosti, která by měřila, zda se ve výsledcích modelu nachází očekávané tahy. V případě, že by tah chyběl, by pak úspěšnost klesala. Binary accuracy nelze použít, neboť ta uvažuje úspěšnost u všech možností, jak jsem popsal v praktické části. Vytvoření této funkce by bylo vhodnou částí v případě hlubších studií, neboť by umožnilo přesnější hodnocení úspěšnosti. Na samotnou chybu modelu by funkce neměla vliv.

Velmi povzbudivé jsou výsledky zakomponování modelu do algoritmu alpha-beta, kde i přes základní implementaci lze výsledek 87 % výher považovat za velmi dobrý.

Největší částí celé práce bylo vytvoření dat, neboť jsem se pokoušel o sofistikovanější, a i časově náročnější způsob přípravy jednotlivých dat. Výsledkem bylo spočítání herních pozic s velkou přesností a tím zajištění kvalitních dat. Vzniká tu velký rozdíl oproti způsobu, kdy necháte model hrát jednotlivé hry a tím zaručíte propojení jednotlivých pozic. V mém případě propojení z části chybí, neboť Embryo mohlo najít lepší následující tahy, po kterých vznikne pozice, která se nenachází v množině pozic vytvořených z daných her hráčů. Problém propojení mohl způsobovat, proč se modely nebyly schopny učit na zamíchaných datech.

Modely se ukládaly jen v případě, že při učení dosáhly v danou chvíli úspěšnost 40 % na validačním datasetu. Tato podmínka nezaručuje dobrou průměrnou úspěšnost na testovacích datasetech, ale je postačující pro výběr na validaci na testovacích datech. Tento postup zaručil uchování modelů pro případné následující testy nebo v budoucnu rozšířenější operace, kombinace nebo i mutace.

Za zmínku stojí i schopnost detekovat vzory přes celou desku, kde v některých případech model nabízel možnosti, které vyčetl z kamenů existujících v daných pozicích, nikoliv z možných následujících tahů.

Jak už jsem zmínil v praktické části, tak úprav na datech by mohlo být více. Obzvlášť zajištění zrcadlení a otáčení by mohlo pomoc zpřesnit hodnocení. Vzhledem k specifikaci a tvorbě přesných dat, není možné, bez velké chybovosti, posouvat pozice po desce. Tento problém bych do budoucna řešil posouváním pozice o jedna ve všech směrech a přepočítáním. Dostal bych následně 9 specifických pozic, které by byly sobě podobné, ale správně spočítané. Dle mých odhadů a promýšlení, by to mohlo pomoct neuronovým sítím a výpočet by se zvětšil na méně jak devítinásobek. Neboť v určitých situacích by se posuny shodovaly. Neuronové sítě by pak mohly být schopny vytáhnout vzory způsobené posunutím a snáze započítávat hrany desky. V ideálním případě by bylo vhodné sehnat více pozic, které budou mimo střed desky, neboť pozice, které jsou blízko hraně desky bývají specifické s netypickými tahy. Obzvlášť pak pozice, kde hrají roli dvě hrany neboli roh desky.

Za zvláštní pozorování považuji velmi nízké množství knih, diplomových a disertačních prací k rozvoji a trénování logického myšlení u dospělých lidí v porovnání se zdroji zabývající se logickým myšlením u dětí nebo dokonce v porovnání k neuronovým sítím. Jeden z důvodů může být i dán tím, že neuronové sítě jsou v poslední době velmi populární. Myslím si, že tato část může být ve světě velmi ceněná, neboť i v práci mohu pozorovat důležitost logického uvažování, které není lehké trénovat a ani letitá praxe není zárukou úspěchu. Vzdělávat se je potřeba stále a logické myšlení by nemělo být zapomenuto.

# 6 Závěr

V praktické části práce jsem prozkoumal různé přístupy k využití her pro stimulaci logického myšlení. Postupně jsem analyzoval myšlení, strategii, logiku a matematickou logiku, abych nakonec zkoumal samotné hry, které často vycházejí z těchto principů. Snažil jsem se přiblížit vliv, který mohou hry mít na logické myšlení jednotlivce. Za tímto účelem jsem představil základní koncepty teorie her a umělé inteligence a popsal nejznámější metody umělé inteligence.

Následně jsem se zaměřil na konkrétní logickou hru, Gomoku, a zkoumal její potenciální využití v reálném světě pro rozvoj logického myšlení a znalostí. V praktické části jsem popsal proces vytváření modelů s využitím teorie. Pro tvorbu těchto modelů jsem se obrátil k neuronovým sítím, které jsem trénoval na sofistikovaných datech, jež tvořila klíčovou součást práce.

Modely jsem následně validoval nejen na testovacích datech, ale i na dobrovolnících, kteří mi poskytli zpětnou vazbu, díky níž jsem mohl zlepšit úroveň modelů. Zároveň jsem zautomatizoval generování grafů pro lepší vizualizaci výsledků. V poslední části jsem implementoval model postavený na neuronových sítích do algoritmu prohledávání do hloubky, konkrétně alpha-beta. Prováděl jsem testy porovnání výkonnosti s modelem bez prohledávání do hloubky.

Podařilo se mi natrénovat neuronové sítě a potvrdit teorii použitelnosti metod v problematice her, zejména v kontextu Gomoku. Avšak samotné trénování vyžadovalo značné množství času na přípravu, čištění a úpravy dat. Jsem si vědom toho, že projekt by mohl pokračovat ještě několik let, jak dokládají nejlepší modely na světě (Rapfi, 2022).

Nakonec jsem byl schopen integrovat vytvořené modely do mobilní aplikace, kterou si hráči mohou stáhnout z obchodů Apple a Google pod názvem "Gomoku library". Velký přínos vidím zejména v metodě přípravy dat, kde předpočítání všech možných následujících tahů přineslo úplné pozice, které jsem následně čistil a upravoval. Výsledkem bylo vytvoření

modelu, který nezávisel pouze na nejlepším následujícím tahu, ale na množině přípustných tahů, které jsem mohl procházet pomocí prohledávacího algoritmu alpha-beta.

Tento model je schopen generovat nejlepší následující tahy v daných pozicích a může tak sloužit nejen k učení hry, ale i k rozvoji logického myšlení.

Ze získaných závěrů vyvozuji možnost pokračovat v projektu s možnými změnami, jak v napočítaných datech, tak i ve struktuře modelu neuronových sítí. Dále se budu zaměřovat na vyladění alpha-beta, aby bylo možné v kratší časové době spočítat větší množství možných kombinací. V neposlední řadě bude možné vytvořit určitý skript, který projde všechny dostupné hry a zjistí nejhranější pozice, které budou sloužit jako potenciální kandidáti pro nová trénovací data neuronových sítí.

Projekt mi dal mnoho zkušeností do praxe a umožnil mi se posunout dále v problematice umělé inteligence v hrách. Rok aktivní práce na projektu se ukázal jako nedostačující doba pro hlubší výzkum, kde složitější experimenty musely být vynechány. Zároveň jsem získal potřebné informace, které mi dodaly jistotu a chuť v projektu pokračovat i v budoucnu.

# 7 Seznam použitých zdrojů


[1] Muzika, Martin. Application of Game Theoretic Algorithms to Gomoku. Praha, 2017. Bakalářská práce. ČVUT.

[2] Čtvrtníková, Katrin. Stolní hra jako stimulace logického myšlení u dětí ve věku 5–7 let. Praha, 2018. Bakalářská práce. Univerzita Karlova v Praze.

[3] Yoav Shoham and Kevin Leyton-Brown. Multiagent systems: Algorithmic, game-theoretic, and logical foundations. Cambridge University Press, 2008.

[4] Mráček, Bc. Lukáš. Vliv hraní akčních a strategických počítačových her na kognitivní percepci hráčů: přehledová studie. Olomouc, 2022. Bakalářská práce. Univerzita Palackého v Olomouci.

[5] Heus, Steven de. Designing and Evaluating an Educational Board Game. Twente, 2020. Student theses. University of Twente.

[6] Korotovskaia, Alina. Board game for participatory design. Lapland, 2020. Master's theses. University of Lapland.

[7] Kopecká, Simona. Didactic Potential of Board Games in Teaching English at Primary Schools. Brno, 2019. Bachelor thesis. Masaryk university.

[8] Dosedělová, Iva. Popularizace teorie her. Brno, 2015. Bakalářská práce. Masaryk university.

[9] Trombiková, Bc. Iveta. Rozvoj logického myšlení u žáků na 2. stupni ZŠ. Brno, 2021. Diplomová práce. Masarykovo Univerzita.

[10] Arjoranta, Jonne. How to define games and why we need to. Jyväskylä, 2019. Student's thesis. University of Jyväskylä.

[11] Gentile, Manuel, Mario ALLEGRA and Heinrich SÖBKE. Games and Learning Alliance: 7th International Conference, GALA 2018 Palermo, Italy, December 5–7, 2018 Proceedings. Springer, 2018.

[12] Lin, Y.-T.; Cheng, C.-T. Effects of Technology-Enhanced Board Game in Primary Mathematics Education on Students' Learning Performance. Appl. Sci. 2022, 12, 11356. https://doi.org/10.3390/app122211356

[13] Chou MJ. Board games play matters: A rethinking on children's aesthetic experience and interpersonal understanding. Eurasia Journal of Mathematics, Science and Technology Education. 2017 Jun 15;13(6):2405-21.

[14] Hrůša, Lukáš. EXPERIMENTÁLNÍ HRY V PSYCHOLOGICKÉ PRAXI. Brno, 2018. Diplomová práce. Masarykovo univerzita.

[15] Pazika, Bc. Jiří. Základní principy umělé inteligence. Praha, 2009. Diplomová práce. Bankovní institut vysoká škola Praha.

[16] Trunda, Otakar. Monte Carlo Techniques in Planning. Praha, 2013. Master thesis. Charles University in Prague.

[17] Viliam Lisy, Vojta Kovarik, Marc Lanctot, and Branislav Bosansky. Convergence of monte carlo tree search in simultaneous move games. In Advances in Neural Information Processing Systems, pages 2112–2120, 2013.

[18] Gomoku. Gomoku czech website. http://www.piskvorky.cz/, 2006 (accessed December 16, 2023).

[19] Gomoku. Gomoku world website. http://gomokuworld.com/, 2013 (accessed December 16, 2023).

[20] Louis Victor Allis et al. Searching for solutions in games and artificial intelligence. Ponsen & Looijen, 1994.

[21] Géron, Aurélien. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and techniques to build intelligent systems. Beijing ; Boston ; Farnham ; Sevastopol ; Tokyo: O'Reilly, 2019. ISBN 978-1-4920-3264-9.

[22] CHollet, François. Deep learning with Python. Shelter Island: Manning, 2021. ISBN 978-1-61729-686-4.

[23] Graupe, Daniel. Principles of artificial neural networks. 4th ed. New Jersey: World Scientific, c2019. Advanced Series in Circuits and Systems. ISBN 978-981-120-122-6.

[24] David Kriesel. A brief introduction on neural networks. 2007.

[25] Jun Hwan Kang and Hang Joon Kim. Effective monte-carlo tree search strategies for Gomoku ai. 2016.

[26] Convolutional Neural Network (CNN). Convolutional Neural Network (CNN) [online]. [cit. 2023-12-17]. Dostupné z: https://www.tensorflow.org/tutorials/images/cnn

[27] Machine Learning Glossary. Machine Learning Glossary [online]. [cit. 2023-12-17]. Dostupné z: https://developers.google.com/machine-learning/glossary/#convolutional_neural_network

[28] Cueva Carrión, Henry. "The use of Minecraft to foster creativity, collaboration and motivation through game-based learning and gamification." Master's thesis, H. Cueva Carrión, 2018.

[29] Rapfi. Rapfi [online]. 2022 [cit. 2024-03-24]. Dostupné z: https://github.com/dhbloo/rapfi

[30] Antebi, Liran. “What Is Artificial Intelligence?” Artificial Intelligence and National Security in Israel, Institute for National Security Studies, 2021, pp. 31–40. JSTOR, http://www.jstor.org/stable/resrep30590.7. Accessed 28 Mar. 2024.

[31] Levin, Oscar. Discrete Mathematics: An Open Introduction. United States, INDEPENDENTLY PUBLISHED, 2018.

## 7.1 Seznam obrázků

## 7.2 Seznam grafů

# Přílohy

Ukázka jedné pozice v textové variantě

Pozice

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 2 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 2 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 1 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Ohodnocení:

```
    a    b    c    d    e    f    g    h    i    j    k    l    m    n    o
  |- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - |
 0|-M13 -M15 -M13 -M15 -M13 -M15 -M19 -M15 -M13 -M17 -M15 -M15 -M15 -M15 -M15 |
 1|-M15 -M17 -M15 -M17 -M15 -M19 -M15 -M17 -M17 -M15 -M19 -M17 -M15 -M13 -M15 |
 2|-M13 -M13 -M15 -M13 -M13 -M19 -M19 -M21 -M23 -M15 -M21 -M21 -M17 -M15 -M13 |
 3|-M13 -M13 -M13 -M13 -M15 -M13 -M21 -102  -60 -M21 -163 -M25   X  -M15 -M15 |
 4|-M17 -M13 -M15 -M15 -M15 -M15 -M15 -205 -151   O  -139 -517 -M25 -M17 -M13 |
 5|-M15 -M13 -M13 -M15 -M15 -M19 -M23   X  -262 -192   O  -M17 -M21 -M15 -M13 |
 6|-M15 -M15 -M13 -M13 -M13 -M13 -M17 -M25   O    X  -286  -98 -M15 -M15 -M13 |
 7|-M13 -M19 -M15 -M13 -M13 -M15 -M15 -M19 -M15 -M15 -M23 -M19 -M17 -M13 -M15 |
 8|-M13 -M15 -M15 -M15 -M15 -M13 -M15 -M17 -M13 -M15 -M17 -M15 -M19 -M15 -M13 |
 9|-M15 -M17 -M13 -M15 -M15 -M19 -M17 -M19 -M15 -M15 -M15 -M13 -M15 -M13 -M13 |
10|-M13 -M15 -M15 -M15 -M17 -M15 -M15 -M13 -M13 -M15 -M13 -M13 -M17 -M15 -M13 |
11|-M15 -M15 -M15 -M13 -M15 -M15 -M15 -M13 -M13 -M17 -M19 -M15 -M13 -M15 -M15 |
12|-M13 -M13 -M15 -M15 -M17 -M15 -M15 -M13 -M13 -M15 -M15 -M15 -M15 -M15 -M13 |
13|-M15 -M13 -M17 -M15 -M13 -M15 -M15 -M15 -M13 -M15 -M15 -M13 -M15 -M13 -M15 |
14|-M13 -M13 -M13 -M15 -M13 -M15 -M15 -M15 -M13 -M15 -M15 -M15 -M13 -M15 -M15 |
```

Vysvětlivky:

Kde X = 2, O = 1, M__ značí počet kamenů do výhry/prohry.

Nebo-li, když X(je na tahu) zahraje na h6 tak prohraje nejpozději za 25 kamenů. Pokud zahraje na i3, tak bude mít mírnou nevýhodu.