Projekt MapAwareness

Starcraft 2 je strategická hra od společnosti Blizzard Entertainment. Jedním z herních prvků je i minimapa, která zobrazuje zobrazuje clou herní mapu ve zmenšené dvourozměrné podobě a slouží ktomu, aby měli hráči přehled i o dění mimo oblast na hlavní obrazovce, např. že nepřítel útočí na jejich základnu. Já jsem se pokusil zrovna tento případ detekovat automaticky. Cílem tedy bylo na základě obrázku minimapy detekovat, zda je ohrožena nějaká základna.

Obrázky map použité pro trénování a testování sítě byly získány z komentovaných záznamů zápasů ve Starcraft 2 (pochazejících ze serveru www.youtube.com). Záznamy byly vybírány tak, aby jeden hráč byl modrý a druhý červený, toto barevné rozdělení jsem interpretoval tak, že modré jednotky jsou „mé“ a červené jednotky jsou „nepřátelské“.

Algoritmus a implementace

Tuto automatickou detekci jsem se pokusil naučit neuronovou síť. Inspiraci jsem si vzal z projektu ALWIN (http://ftp.utcluj.ro/pub/docs/imaging/Autonomous_driving/Articole%20sortate/CThorpe/ALVINN%20Project%20Home%20Page.htm) jehož autorům se podařilo využít schopnosti neuronové sítě rozpoznat vzory k tomu, aby naučily jednoduchou třívrstvou neuronovou síť řídit auto podle obrazu z kamery. Já jsem chtěl využít stejnou vlastnost abych síť naučil detekovat hrozbu základně z obrazu minimapy. Modelem sítě je vícevrstevý perceptron (podobně jako v projektu ALWIN) se sigmoidální aktivační funkcí na všech neuronech. Obrázky jsou mapovány na vstupní neurony tak, že každému pixelu odpovýdají tři neurony, a každému neuronu je přiřazena hodnota jedné ze základních barev daného pixelu v barevném schématu RGB tak, že hodnoty barev jsou z celočíselné reprezentace v rozsahu 0 až 255 transformovány do reálných čísel rozsahu 0,0 až 1,0 a takto vzniklá hodnota je interpretována jako výstupní hodnota daného neuronu.

Pro učení sítě používám učení s učitelem pomocí algoritmu Resilientní zpětné propagace, což je upravená verze klasického algoritmu zpětné propagace.

Síť je implementována v jazyce Java frameworkem Encog, pro vytvoření a obsluhu sítě jsem vytvořil také jednoduché GUI, které umožňuje načíst nebo vytvořit novou síť, trénovat síť a vyhodnotit jednotlivé obrázky nebo celou datovou sadu.

Všechny vstupní obrázky jsou po načtení sady automaticky zmenšeny na 100x100 pixelů. Jako nejlepší konfigurace se ukázala třívrstvá síť 100x100x3 vstupních neuronů, 40 ve stkryté vrstvě a 10 výstupních. Jako limit pro zastavení tréninku byla zvolena hranice pod 2% MSE (soubor best_40n_MSE2.net).

Návod

1) Spuštění

Aplikace vyžaduje javu 1.7. Aplikeaci je vhodné spouštět z konzole příkazem „java -jar -Xmx1000M MapAwarenessTEST-1.0-SNAPSHOT.jar“.
Jako datovou sadu aplikace očekává obrázky roztřízené do složek podle kategorií ve formátu: *názevSady/názevKategorie/Obrázek*

Tedy **datová sada** je složka obsahující složky pojmenované podle kategorií, každá obsahující obrázky minimapy reprezentující danou kategorii.
(mnou použité sady jsou ve složce *datasets*, složky v ní tedy reprezentují jednotlivé sady; např. cesta *trenink/0/img104801.png* znamená, že obrázek *img104801.png* reprezentuje kategorii 0 v datové sadě *trenink*)

Aplikace vypisuje svůj textový výstup do konzole.

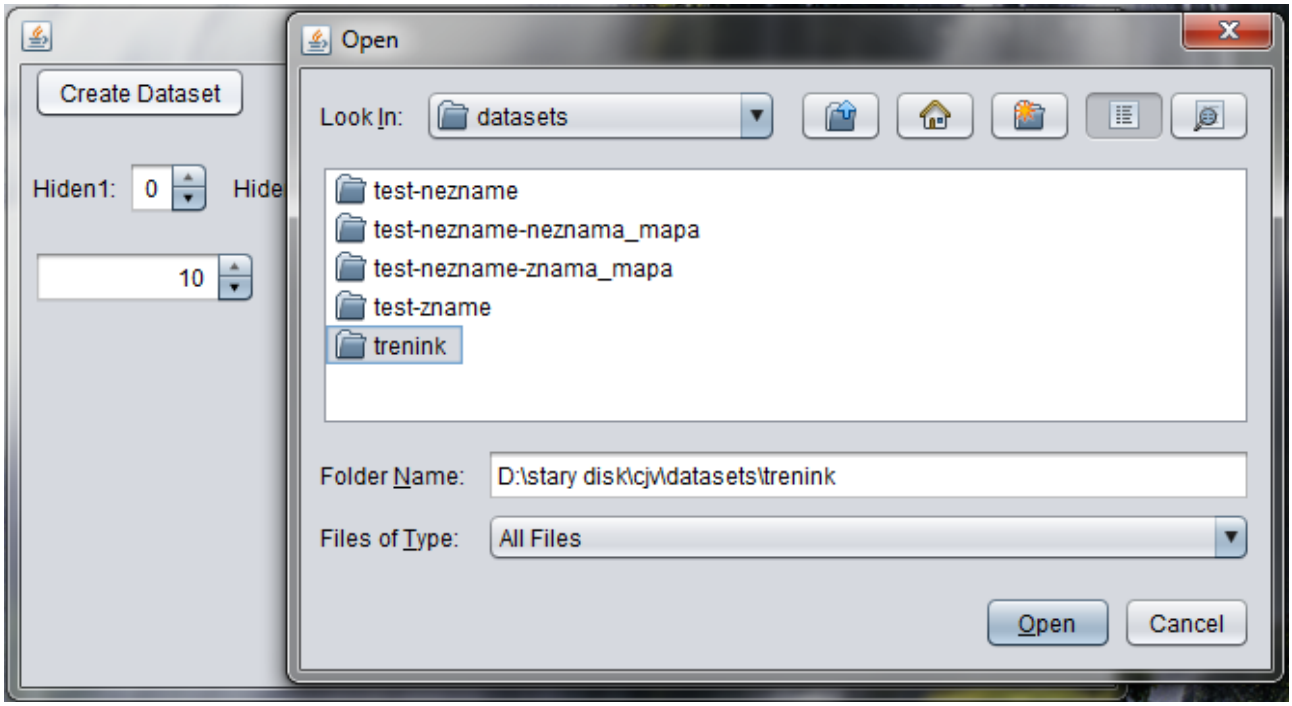
```
File: 7_8_img1018461.png detected:4 expected: 8
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,2490 5 = 0,0000 6 = 0,0000 7 = 0,0000 8 = 0,0027 9 = 0,0000
File: 8_9_6img385100.png detected:9 expected: 8
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,0000 5 = 0,0000 6 = 0,0000 7 = 0,0000 8 = 0,0056 9 = 0,1738
File: 8_9_img1297400.png detected:5 expected: 8
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,0000 5 = 0,3864 6 = 0,0000 7 = 0,0000 8 = 0,0059 9 = 0,0000
File: 8_9_img1299200.png detected:9 expected: 8
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,0000 5 = 0,0000 6 = 0,0000 7 = 0,0000 8 = 0,0001 9 = 0,9999
File: 8_9_6_img389904.png detected:8 expected: 9
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,0000 5 = 0,0000 6 = 0,0001 7 = 0,0000 8 = 0,9940 9 = 0,0084
File: 9_8_img287202.png detected:8 expected: 9
Values:  0 = 0,0000 1 = 0,0000 2 = 0,0000 3 = 0,0000 4 = 0,0000 5 = 0,0000 6 = 0,0000 7 = 0,0000 8 = 0,0022 9 = 0,0000
Succesfuly detected - strictly 76,063830%:
3: 53,333333%
2: 72,619048%
1: 70,000000%
0: 75,510204%
7: 72,727273%
6: 66,666667%
5: 86,666667%
4: 93,846154%
9: 91,304348%
8: 50,000000%
Succesfuly detected - loosely 85,372340%:
3: 60,000000%
2: 82,142857%
1: 90,000000%
0: 75,510204%
7: 100,000000%
6: 100,000000%
```

*Výstup: test sady **test-zname***

Ukázka výstupu výše ukazuje část výstupu testování (tlačítko test) sady *test-zname*, Sucsesfuly detected – strictly/loosely udává úspěšnost při strikním/volném vyhodnocení (viz. kapitola Vyhodnocení) nejprve celkově a poté pro jednotlivé složky.

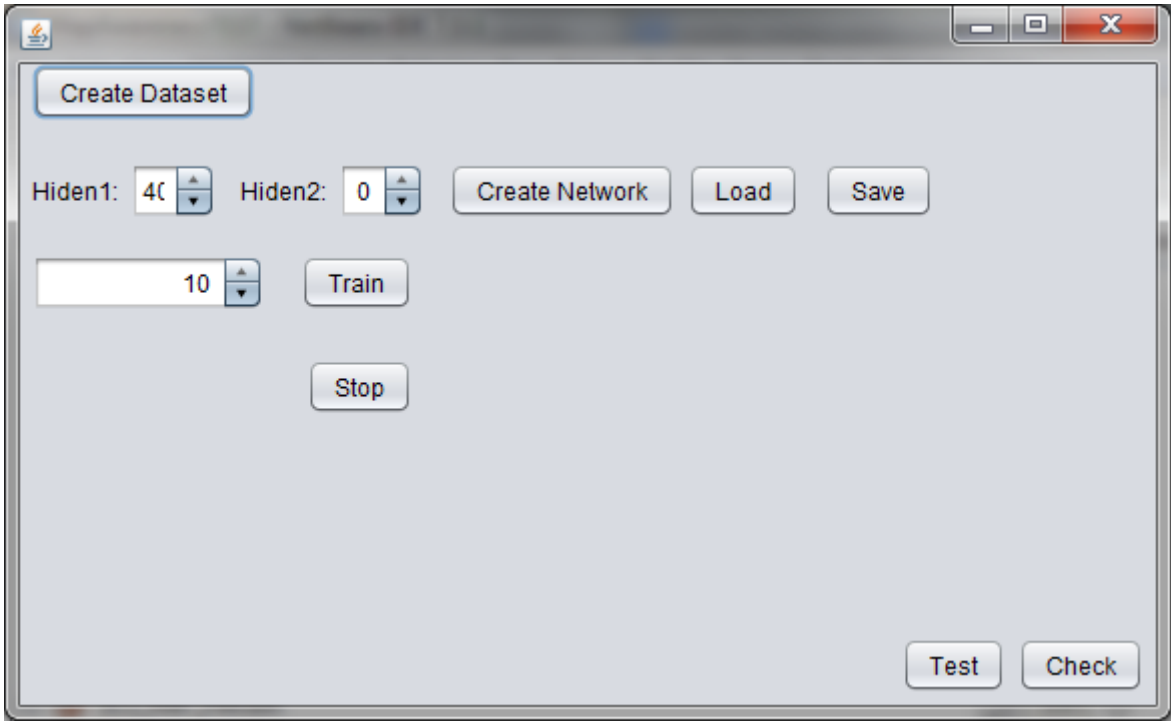
```
Added input image:img986000.png
Added input image:img916281.png
Added input image:img997802.png
Added input image:img93404.png
Added input image:img998504.png
Added input image:img979903.png
Downsampling images...
Done.
Created network
Beginning training...
Iteration #1 Error:49,040404% elapsed time = 00:00:03 time left = 00:10:00
Iteration #2 Error:9,965849% elapsed time = 00:00:07 time left = 00:10:00
Iteration #3 Error:9,388570% elapsed time = 00:00:11 time left = 00:10:00
Iteration #4 Error:9,121968% elapsed time = 00:00:15 time left = 00:10:00
Iteration #5 Error:13,072601% elapsed time = 00:00:18 time left = 00:10:00
Iteration #6 Error:10,079350% elapsed time = 00:00:22 time left = 00:10:00
Iteration #7 Error:9,369121% elapsed time = 00:00:26 time left = 00:10:00
```

Výstup: načtení tréninkové sady a trénink nové sítě



Ukázka GUI: načtení trénikové sady

2) Obsluha



Create Dataset – Otevře dialogové okno pro výběr složky s tréninkovou datovou sadou.
Create Network – Vytvoří novou 3 nebo 4 vrstvou síť, počet neuronů první skryté vrstvy udává pole **Hiden1**, počet neuronů případné druhé skryté vrstvy pak pole **Hiden2** (pokud je **Hiden2**=0 tak bude síť mít jen jednu skrytou vrstvu, tedy v situaci na obrázku výše by **Create Network** vytvořilo novou tří vrstvou síť se jednou skrytou vrstvou o 40 neuronech).
Poznámka: pole **Hiden1/2** se týkají pouze vytvoření nové sítě nikoliv např. načtení dříve uložené sítě.
Load – načte dříve uloženou síť
Save – uloží právě používanou síť
Train – Spustí trénink sítě, políčko předd tlačítkem **Train** udává časový limit na trénink v

minutách, po uplinutí limitu se trénink zastaví. Aplikace také obsahuje pevný limit na 2% tréninkovou chybu, pukud síť dosáhne menší chyby, tak je trénink zastaven. Opětovným stisknutím **Train** lze trénink znovu spustit s novým časovým limitem.

Stop – zastaví trénink.
Test – Otevře dialog výběru složky datové sady a otestuje zvolenou datovou sadu.
Check – Otevře dialog výběru souboru a pokusí se klasifikovat vybraný obrázek.

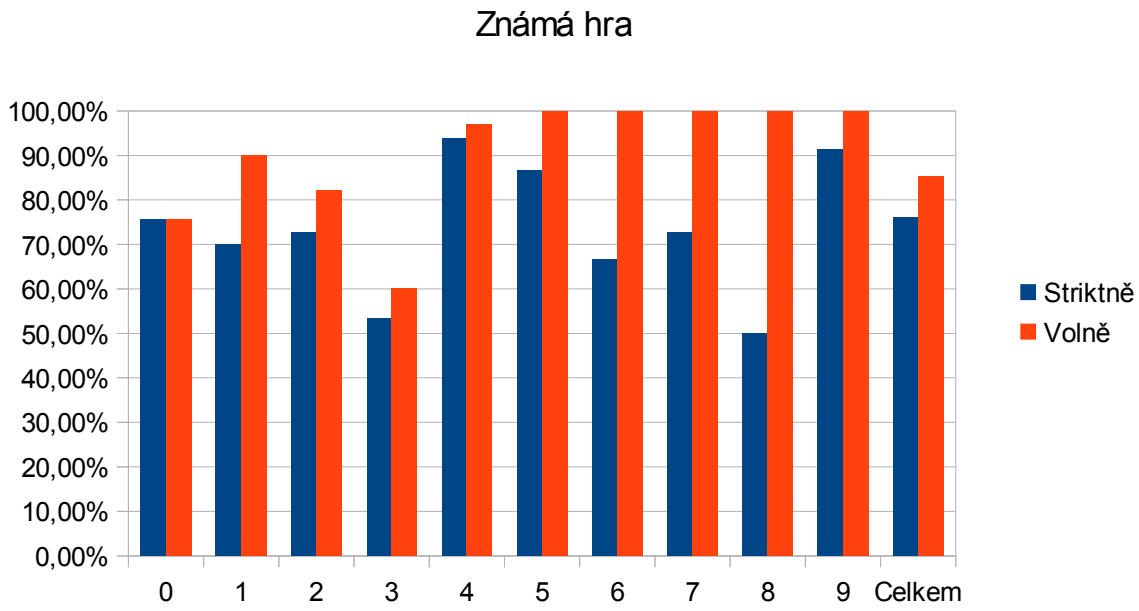
Vyhodnocení

Mapy jsem získával a třídil ručně a kromě správného rozdělení do kategorií jsem se snažil, aby mezi jednotlivými obrázky byl rozumný časový rozestup (aby se příliš nevyskytovaly prakticky identické mapy).

Správné vyhodnocení síti stěžoval jednak terén mapy, který nebyl pro klasifikaci podstatný a působil jako šum, a také různá velikostí map.

1. Známá hra

(sada: test-zname)

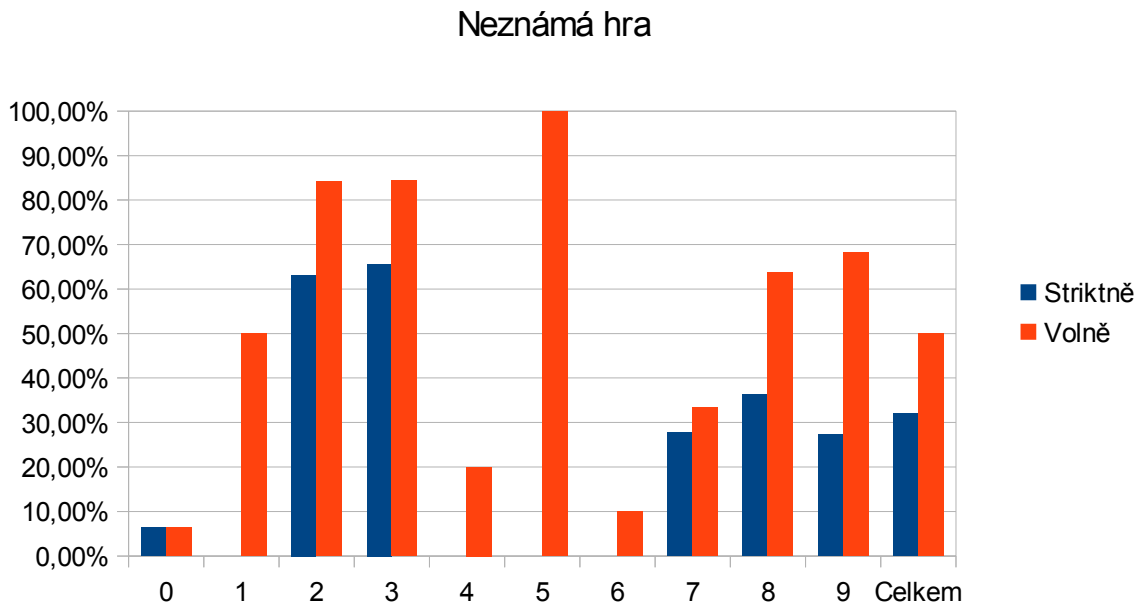


Graf 1

Pro první test jsem použil obrázky ze zápasů ze kterých pocházely i obrázky, na kterých síť trénovala. Graf 1 zobrazuje procentuální úspěšnost klasifikace obrázků z jednotlivých složek (kategorií) testovací sady, modré sloupce představují podíl úspěšně klasifikovaných obrázků z dané kategorie, pokud povoluji jen jediný možný výsledek, takto byla síť trénována, to ale může být zbytečně přísné, pokud je útočník poblíž hranice regionů a síť ho detekuje v opačném regionu nebo probíhá útok na více místech zároveň, proto jsem přidal volnější vyhodnocení znázorněné červeně, které připouští jako správnou odpověď i tyto případy.

2. Neznámá hra

(sada: test-nezname)

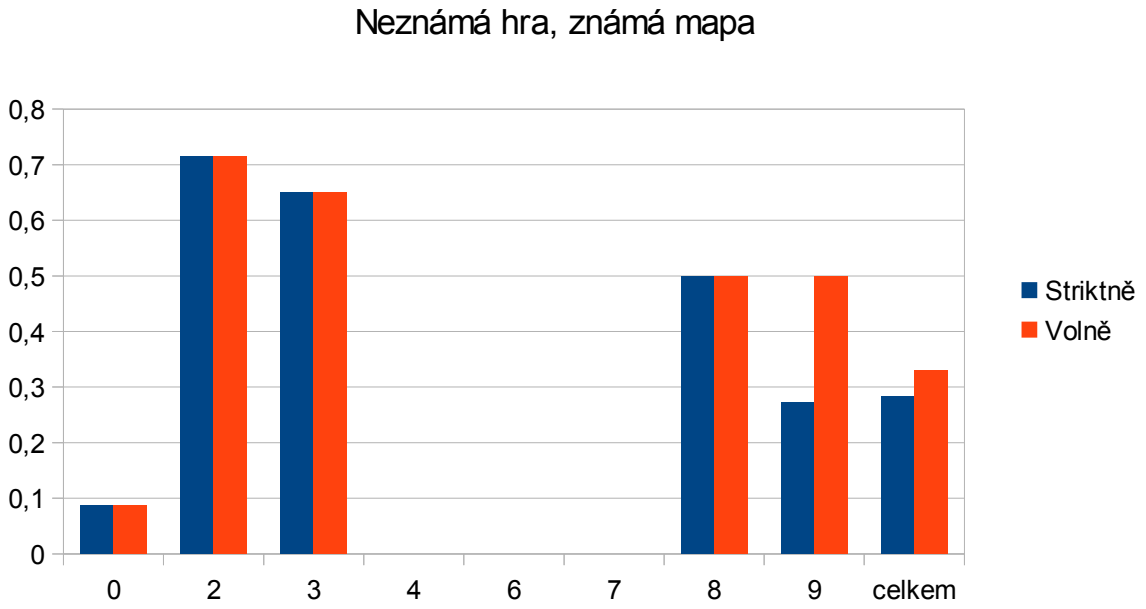


Graf 2

Druhý graf zobrazuje výsledky testu obrázků pocházejících z her, na kterých síť netrénovala.

2.1. Neznámá hra, známá mapa

(sada: test-nezname-znama_mapa)



Graf 3

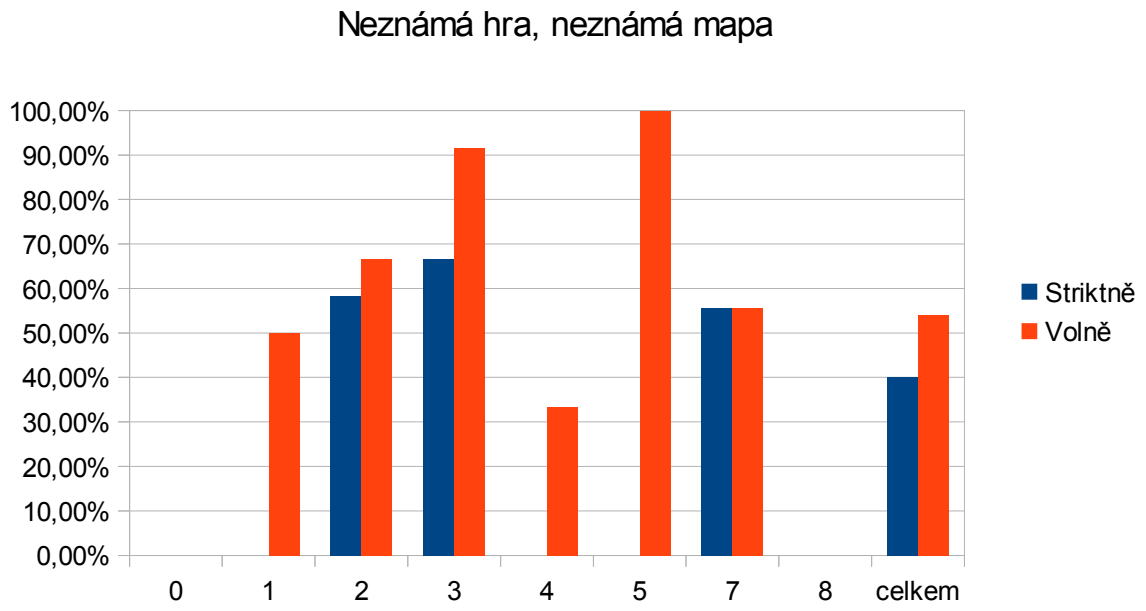
V dalším grafu jsem vybral ze sady neznámých her ty hry, jež se odehrávají na stejných mapách na nichž síť trénovala.

2.2. Neznámá hra, neznámá mapa

(sada: test-nezname-neznama_mapa)

V posledním grafu jsem vybral ze sady neznámých her ty hry, jež se odehrávají na mapách na nichž

síť nikdy netrénovala.



Graf 4

2.3. Alternativní konfigurace

Zde ještě uvedu výsledky testů některých alternativních sítí na neznámých zápasech.

Celková chyba při stejné konfiguraci, ale hranici zastavení pod 1% MSE (40n_MSE1.net): 23,7% striktně, 41,0% volně.

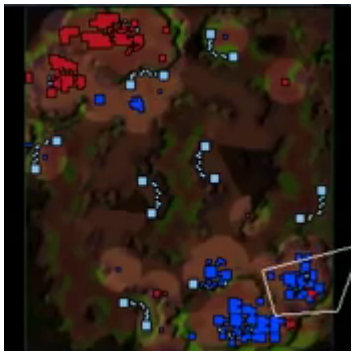
Celková chyba při stejné konfiguraci, ale hranici zastavení pod 3% MSE (40n_MSE3.net): 19,2% striktně, 37,2% volně.

Celková chyba sítě se dvěmi skrytými vrstvami 40 neuronů v horní a 20 v dolní vrstvě MSE cca 2,5% (po více jak 12 hodinách tréninku se nedostala pod 2% hranici, proto jsem se rozhodl použít tuto) (40n_20n_MSE2,5.net): 17,3% striktně, 38,5% volně.

Celková chyba sítě se jednou skrytou vrstvou s 50 neurony a hranicí 2% MSE (50n_MSE2.net): 19,9% striktně, 34,6% volně.

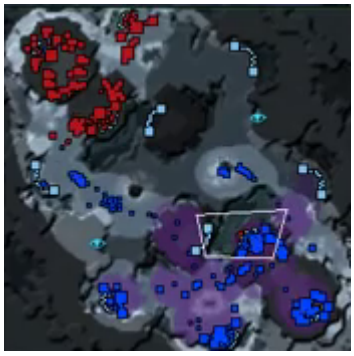
3. Problémy

Občas má síť zjevně problémy s terénem, toto se nejvíce projevje na mapě „Heavy Rain“, kde je většina terénu hnědá (obr. 1) a síť velmy často detekuje útok nepřítele tam, kde žádný není a naopak ignoruje existujícího útočníka.



Obr. 1

Podobný problém, ikdyž výrazně méně častý, má síť i pokud hráč hraje za rasu zergů, která kolem své budovy staví na tzv. „creep“, který je na mapě vyznačen světle fialovou plochou (obr. 2).



Obr. 2

Z grafu 2 je vidět, že síť má také výrazný problém s klasifikací stavu kdy nikdo neútočí (kategorie 0).

Podrobnější prohlídkou chybně klasifikovaných obrázků jsem navíc zjistil, že síť některé obrázky klasifikuje spíše (ikdyž zjevně ne výhradně) na základě použité mapy na níž se hra odehrávala a nikoli na pozici útočníků, nejvýraznější je to na „King Sejong Station“ (obr. 2), kterou velmy často chybně detekuje jako kategorii 5 důvodem je zřejmě fakt, že tato mapa byla pro tuto kategorii v tréninkoví sadě velmi často, neboť kategorie 5 je střed mapy a mnoho map má střed prázdný a špatně bránitelný a tedy pro hráče neatraktivní, takže bylo poměrně obtížné najít hru, ze které by pro tuto kategorii bylo možné získat reprezentanty. Obdobný problém je u mapy „Frost“ která je často mylně detekována jako katergorie 9 (pravý horní roh). Toto se projevuje zejména na neznámých hrách, neboť obrázky ze známých her jsou samozřejmě velmi podobné těm, na nichž síť trénovala. Toto vysvětluje výrazný rozdíl v úspěšnosti klasifikace známých a neznámých her.

Volné vyhodnocení dává lepší výsledky, ale protože je u volného vyhodnocení připouštěno více správných odpovědí, tak je třeba také vzít v úvahu, že síť má větší pravděpodobnost náhodou trefit správnou odpověď, toto vyhodnocení tedy dává spíše horní odhad korektnosti sítě na dané sadě.

Závěr

Původně jsem chtěl napřed síť naučit rozpoznávat napřed libovolné ohrožení základny a poté (v další fázi) i konkrétní hrozbu. Musím konstatovat, že se mi nepodařilo úplně uspokojivě dosáhnout ani prvního cíle. Nadruhou stranu se ale domnívám, že se potvrdilo, že neuronová síť má v tomto značný potenciál. Hlavní zdroj problémů vydím jednak v terénu mapy, který představuje zvláště po zmenšení na rozumnou velikost příliš velký šum a hlavně ve stále relativně malé velikosti datových sad vzhledem počtu použitých kategorií.