NNUI2

Semestrální práce 1

Michal Struna

1. Instalace a spuštění

Pro spuštění programu je třeba mít nainstalovaný Python (testováno ve verzi 3.6.9) a následující balíčky:

- numpy
- matplotlib
- mpl_toolkits
- tensorflow nebo tensorflow-gpu

Spuštění programu lze provést z terminálu pomocí interpretu python3 (popř. python):

```
python3 main.py

# Na linuxu lze i toto (pokud se python nachází na /usr/bin/python3):
./main.py
```

Zdrojový kód 1 – Spuštění programu

1.1. Parametry

Program bez parametrů pouze vypíše `summary()` modelu neuronové sítě. Pro změnu chování je nutno dosadit některé z parametrů:

Parametr	**Zkratka**	**Význam**
`--help`	`-h`	Zobrazí nápovědu s použitím všech parametrů.
`--run` *`A B C`*	`-r`	Zobrazí výstup sítě pro vstupy A, B a C.
`--test` *`N`*	`-t`	Provede N testování náhodných vstupů a porovnání s očekávanými hodnotami zobrazí v *grafech*. *
`--train` *`E B V`*	`-tr`	Natrénuje síť E epochami, velikostí dávky B a validačního rozdělení V. Síť je ve výchozím stavu **již natrénovaná** a tento příkaz není nutné spouštět. *Může trvat delší dobu.*
`--loss` *`A B N`*	`-l`	Zobrazí v grafu chybu účelové funkce nad trénovací množinou v závislosti na počtu neuronů ve skryté vrstvě. Učiní tak pro každý Ntý počet neuronů od A do B. *Může trvat delší dobu.*

Tabulka 1 – Parametry programu.

** Protože funkce přijímá 3 argumenty, její graf musí být 4D. Z tohoto důvodu jsou zobrazeny 2 grafy - jeden pro argumenty A, B a výstup, druhý pro B, C a výstup.*

```
# Predikuje X a Y pro vstupy A = -0.5, B = 0.2 a C = 1.
python3 main.py --run -0.5 0.2 1

# Provede testování pro 100 náhodných vstupů.
python3 main.py --test 100

# Natrénuje síť s 1000 epochami, dávkou 32 a validační množinou 15 %.
python3 main.py --train 1000 32 0.15

# Zobrazí chybu účelové funkce pro 2, 5, 8 a 11 neuronů ve skryté vrstvě.
python3 main.py --loss 2 12 3
```

Zdrojový kód 3 – Příklady spuštění programu s parametry.

2. Projektová struktura

2.1. main.py

Hlavní soubor programu, který na základě uživatelského vstupu provede zavolá příslušné funkce.

2.1. io_utils.py

Pomocný soubor pro parsování argumentů, formátovaný výpis do konzole nebo tvorbu grafů.

2.1. model.py

Soubor uchovávající logiku neuronové sítě pracující přímo s tensorflow.

2.1. model.h5

Záloha vytvořené a naučené sítě.

3. Postup vypracování

3.1. Návrh neuronové sítě

Neuronová síť má 3 vrstvy a pro učení využívá algoritmus zpětného šíření chyby.

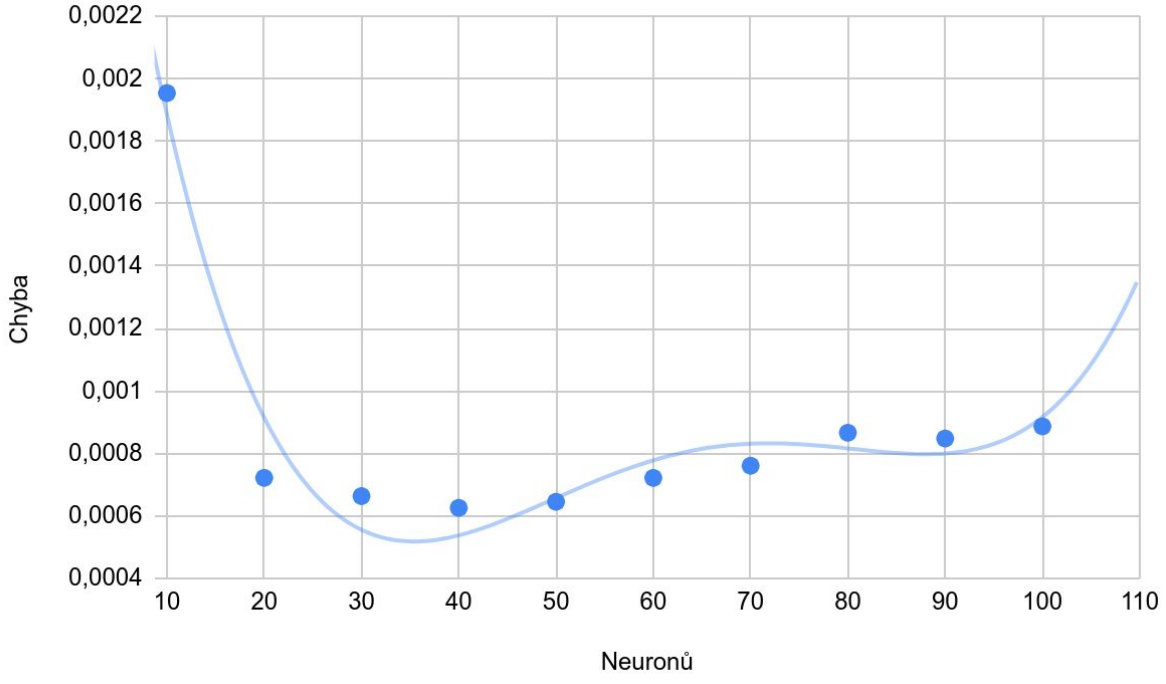
Vrstva	Neuronů	Poznámka	Aktivač. fce.
Vstupní	3	Na vstupu se nachází 3 parametry: a, b, c.	
1. skrytá	38	Chyba účelové funkce pro 38 neuronů je nejmenší.	tanh
Výstupní	2	Na výstupu se nachází 2 parametry x, y.	linear

Tabulka 2 – Vrstvy neuronové sítě.

3.2. Určení počtu neuronů ve skryté vrstvě

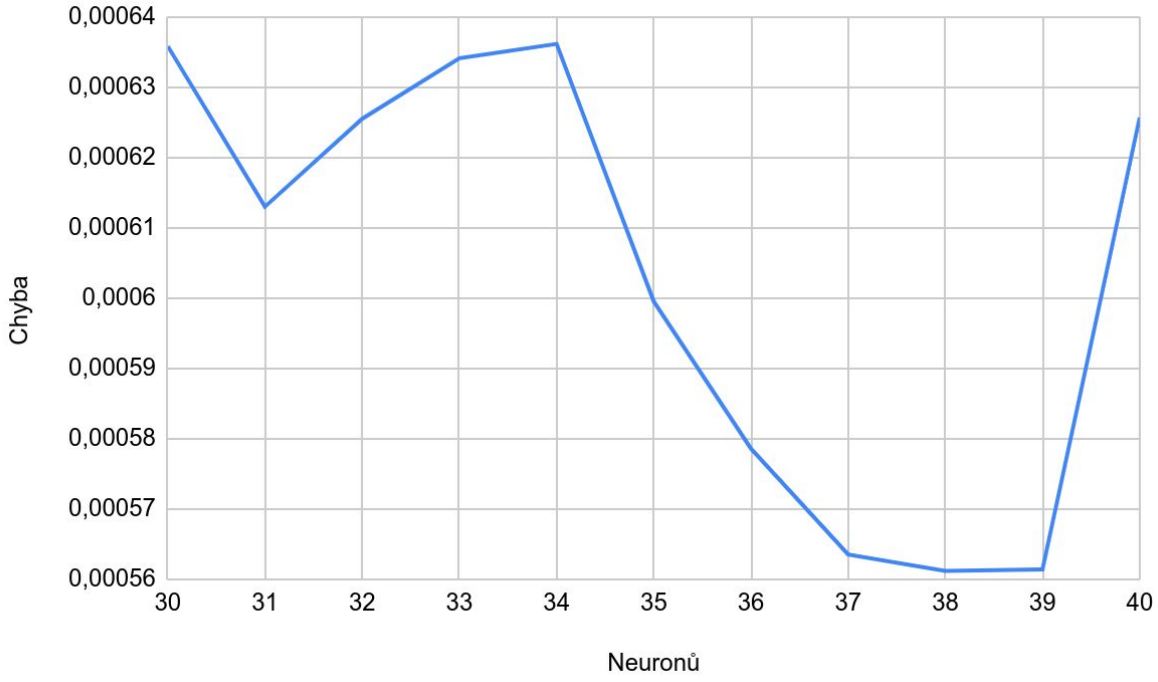
Kvůli časové náročností trénování (a to i v případě využití CUDA) bylo určení počtu neuronů rozděleno do dvou částí. Vždy ovšem platí, že trénování pro každý počet neuronů probíhá 30krát a má 1000 epoch. Hodnota chyby účelové funkce nad trénovacími daty pro každý počet neuronů je pak průměrem chyb z předešlých 30 trénování.

V první části se budou počítat chyby účelové funkce pro 10, 20, 30, …, 100 neuronů. Zjistí se tak přibližná poloha optimálního počtu neuronů.



Graf 1 – Chyba účelové funkce nad trénovacími daty v závislosti na počtu neuronů ve skryté vrstvě.

Z výsledku lze odvodit závěr, že optimální počet neuronů leží někde mezi 30 a 40. Proto je v druhé části celý proces opakován, tentokrát však pro 30, 31, 32, …, 40 neuronů.



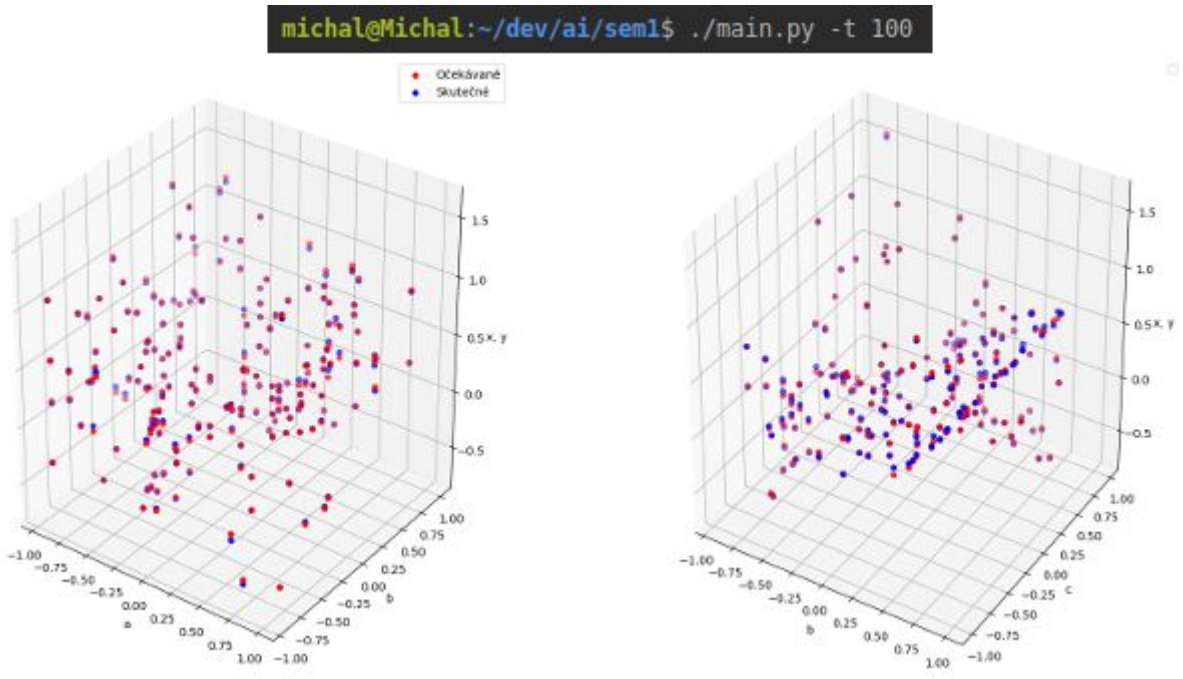
Graf 2 – Chyba účelové funkce nad trénovacími daty v závislosti na počtu neuronů ve skryté vrstvě.

Minimální průměrná chyba účelové funkce na konci trénování byla zaznamenána pro počet neuronů 38, a proto je tento počet napevno nastaven v programu.

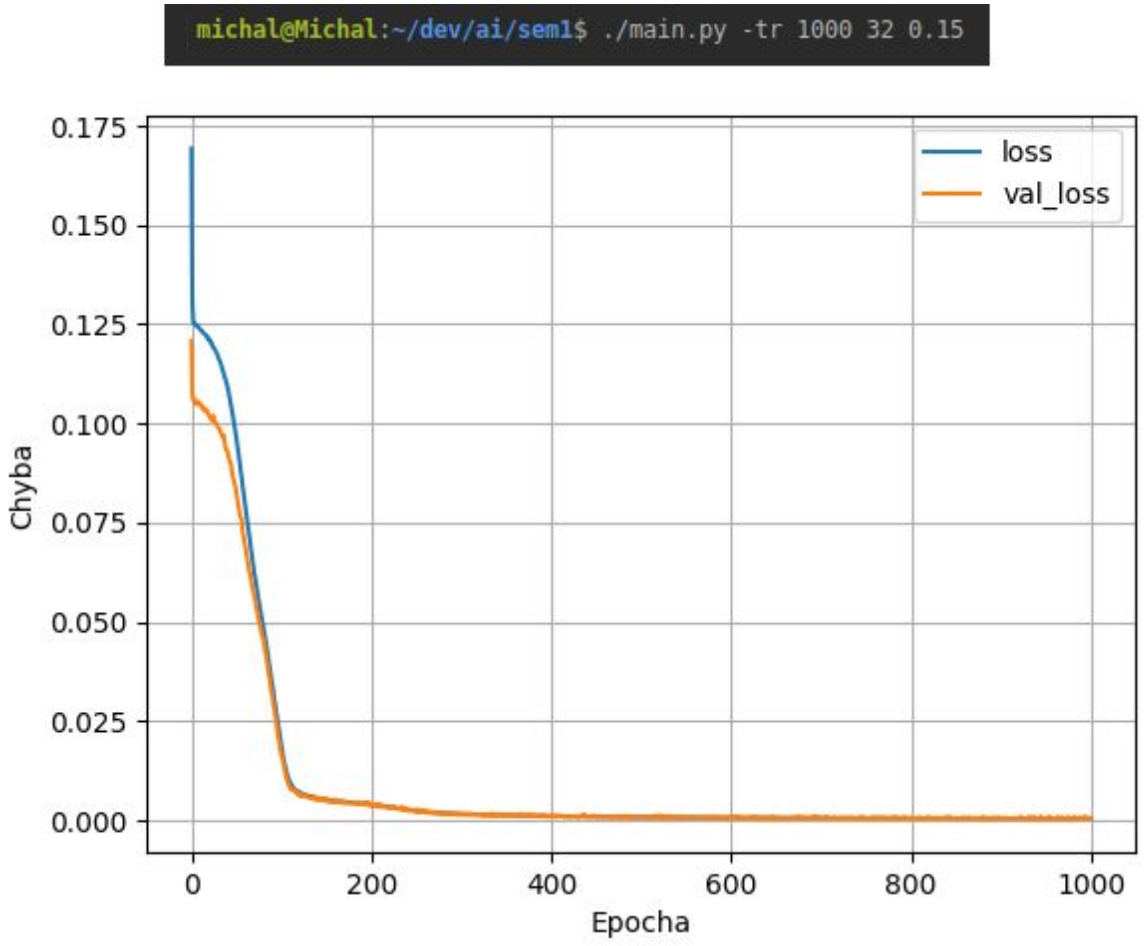
4. Výsledek

```
michal@Michal:~/dev/ai/sem1$ ./main.py -r 1 1 0
a = 1.0
b = 1.0
c = 0.0
x = max(a, b, c) * b => 0.994
y = a^2 - b * c => 0.977
```

Obrázek 1 – Predikce x, y pro vstupy a = 1, b = 1, c = 0



Obrázek 2 – Testování sítě 100 náhodnými vstupy (vlevo (a, b) -> (x, y), vpravo (b, c) -> (x, y))



Obrázek 3 – Trénování sítě a zobrazení průběhu chyby