

# **Rozpoznávání dopravních značek pomocí umělých neuronových sítí**

Road Signs Recognition by Means of Artificial Neural Networks

Bc. Tomáš Vinkler

---

Diplomová práce  
2014



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
akademický rok: 2013/2014

## ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: Bc. Tomáš Vinkler  
Osobní číslo: A12429  
Studijní program: N3902 Inženýrská informatika  
Studijní obor: Počítačové a komunikační systémy  
Forma studia: prezenční

Téma práce: Rozpoznávání dopravních značek pomocí umělých neuronových sítí

Téma anglicky: Road Sign Recognition by Means of Artificial Neural Networks

Zásady pro vypracování:

1. Proveďte rešerši v oblasti umělých neuronových sítí se zaměřením na rozpoznávání vzorů.
2. Vyberte vhodné neuronové síť.
3. Použijte připravenou trénovací a testovací množinu s dopravními značkami k naučení neuronovými sítěmi.
4. Srovnajte vhodnost použití vybraných neuronových sítí.
5. Připravte nejvhodnější síť pro simulovaný navigační systém.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: tištěná/elektronická

Seznam odborné literatury:

1. ZELINKA, Ivan. Umělá inteligence I. Volume 1. Zlín: Vutium, Brno, 1998, 126 p. ISBN 80-214-1163-5.
2. RAHMAN, Atiqur. Computer vision and action recognition: a guide for image processing and computer vision community for action understanding. Amsterdam: Atlantis Press, c2011, xxi, 211 s. ISBN 9789491216206.
3. ŠNOREK M., JIŘINA M.: Neuronové sítě a neuropočítače, ČVUT, 1996, ISBN 80-01-01455-X.
4. BÍLA J.: Umělá inteligence a neuronové sítě v aplikacích, ČVUT, 1996, ISBN 80-01-01275-1.
5. BOSE N.K., LIANG P.: Neural Network Fundamentals with Graphs, Algorithms, and Applications, McGraw-Hill Series in Electrical and Computer Engineering, 1996, ISBN 0-07-006618-3.
6. FREEMAN J. A.: Simulating Neural Networks with Mathematica, Addison-Wesley Publishing Company, 1994, ISBN 0-201-56629-X.
7. NOVÁK M., FABER J., KUFUDAKI O.: Neuronové sítě a informační systémy živých organismů, Grada, 1993, ISBN 80-58424-95-9.
8. The Mathematica Book, manuál softwaru Mathematica.

Vedoucí diplomové práce:

doc. Ing. Zuzana Komínková Oplatková, Ph.D.  
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce:

7. února 2014

Termín odevzdání diplomové práce:

27. května 2014

Ve Zlíně dne 7. února 2014

prof. Ing. Vladimír Vašek, CSc.  
děkan



prof. Ing. Karel Vlček, CSc.  
ředitel ústavu

## **ABSTRAKT**

Práce se zabývá návrhem vhodného modelu neuronové sítě, který je schopen rozpoznávat kandidáty dopravního značení. Na základě vzájemné spolupráce s ostatními členy, kteří se na této týmové práci podílí, je cílem vytvoření výsledné aplikace včetně uživatelského rozhraní, do které jsou implementovány veškeré části všech členů.

Klíčová slova: Umělá neuronová síť, dopravní značení, zpracování obrazu

## **ABSTRACT**

This work deals with designing an appropriate neural network model that is able to identify candidates for traffic signs. On the basis of cooperation with other members who are on the teamwork involved, the aim is to create a final application, including the user interface, which is implemented in all parts of all members.

Keywords: Artificial neural network, traffic signs, imageprocessing

Tímto bych chtěl poděkovat Doc. Ing. Zuzaně Komínkové Oplatkové, Ph.D. za cenné připomínky a rady k této práci.

**Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

**OBSAH**

<b>ÚVOD</b> .....	<b>9</b>
<b>I TEORETICKÁ ČÁST</b> .....	<b>10</b>
<b>1 DOPRAVNÍ ZNAČENÍ V ČR</b> .....	<b>11</b>
1.1 DĚLENÍ DOPRAVNÍCH ZNAČEK.....	11
1.2 ROZMĚRY DOPRAVNÍCH ZNAČEK.....	13
1.3 ZÁSADY UŽITÍ DOPRAVNÍCH ZNAČEK .....	14
1.4 UMISŤOVÁNÍ SVISLÝCH DOPRAVNÍCH ZNAČEK.....	15
1.4.1 Umístění podle směru provozu .....	15
1.4.2 Vzdálenost mezi značkami.....	17
1.4.3 Počet značek.....	18
<b>2 UMĚLÉ NEURONOVÉ SÍTĚ</b> .....	<b>19</b>
2.1 HISTORIE.....	19
2.2 MODEL UMĚLÉHO NEURONU .....	19
2.3 NEURONOVÁ SÍŤ.....	22
2.3.1 Dělení neuronových sítí .....	22
2.3.1.1 Z hlediska průchodu informací.....	22
2.3.1.2 Podle počtu vrstev.....	22
2.3.1.3 Podle algoritmu učení .....	22
2.3.1.4 Podle stylu učení.....	23
2.3.1.5 Podle topologie .....	23
2.3.2 Učení neuronové sítě.....	24
2.3.2.1 Organizační dynamika .....	24
2.3.2.2 Aktivní dynamika .....	24
2.3.2.3 Adaptivní dynamika.....	25
2.4 DOPŘEDNÁ NEURONOVÁ SÍŤ.....	25
2.4.1 Algoritmus Backpropagation .....	27
2.4.2 Levenberg-Marquardtovo učení.....	28
2.4.2.1 Metoda největšího spádu .....	29
2.4.2.2 Newtonova metoda .....	29
2.4.2.3 Gauss-Newtonův algoritmus .....	31
2.4.2.4 Levenberg-Marquardtův algoritmus.....	32
<b>3 WOLFRAM MATHEMATICA</b> .....	<b>36</b>
3.1 ZÁKLADNÍ SLOŽKY SOFTWARE.....	36
3.1.1 Dokumentační systém notebook .....	36
3.1.2 Interaktivní nápověda.....	37
3.1.3 Grafika.....	37
3.1.4 Paleta nástrojů .....	38
3.2 BALÍČEK NEURALNETWORKS .....	38
3.2.1 Funkce balíčku .....	38
3.2.2 Možnosti sítě .....	39
3.2.2.1 Formát vstupních dat .....	39
3.2.2.2 Zápis a význam funkcí.....	40
3.2.2.3 Formát sítě .....	40
3.2.3 Dopředná síť FeedForward .....	41

<b>II PRAKTICKÁ ČÁST .....</b>	<b>44</b>
<b>4 PŘÍPRAVA DAT PRO UČENÍ NEURONOVÉ SÍTĚ .....</b>	<b>45</b>
4.1 IMPORT DAT .....	45
4.2 PŘÍPRAVA TRÉNOVACÍ A TESTOVACÍ MNOŽINY .....	45
4.2.1 Úprava barevného režimu .....	46
4.2.2 Úprava rozměrů.....	46
4.2.3 Transformace obrazových dat .....	47
4.2.4 Normalizace jasu .....	49
4.2.5 Formát trénovací a testovací množiny .....	53
<b>5 UČENÍ NEURONOVÉ SÍTĚ.....</b>	<b>54</b>
5.1 ROZPOZNÁVÁNÍ ČERVENÝCH DOPRAVNÍCH ZNAČENÍ .....	56
5.2 ROZPOZNÁVÁNÍ ŽLUTÝCH DOPRAVNÍCH ZNAČENÍ.....	67
5.3 ROZPOZNÁVÁNÍ MODRÝCH DOPRAVNÍCH ZNAČENÍ .....	68
<b>6 POPIS UŽIVATELSKÉHO ROZHRAŇÍ VÝSLEDNÉ APLIKACE .....</b>	<b>72</b>
<b>ZÁVĚR .....</b>	<b>75</b>
<b>ZÁVĚR V ANGLIČTINĚ.....</b>	<b>76</b>
<b>SEZNAM POUŽITÉ LITERATURY.....</b>	<b>77</b>
<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>	<b>80</b>
<b>SEZNAM OBRÁZKŮ .....</b>	<b>81</b>
<b>SEZNAM TABULEK.....</b>	<b>83</b>
<b>SEZNAM PŘÍLOH.....</b>	<b>84</b>

## ÚVOD

V dnešní době existuje mnoho technik zabývajících se problematikou rozpoznávání vzorů, které dosahují velkých úspěšností, avšak stále je nelze porovnávat s kvalitami lidského mozku. To je dáno tím, že člověk stále nedokázal zjistit samotný princip jeho fungování. Představa, že by byl lidský mozek jakožto ohromně složitý systém počítačově simulován, je doposud nemožná. Mezi nejpoužívanější techniku, která má snahu napodobovat činnost lidského mozku, patří umělé neuronové sítě.

Umělá neuronová síť je výpočetní systém, který nepracuje pouze na základě daných algoritmů. Kopírováním schopností živých organismů se dokáže sám učit a analyzovat rozsáhlé množiny dat, které jiné algoritmy jen těžko zvládnou. Proto se dnes neuronové sítě využívají v mnoha vědních oborech a disciplínách, mezi které patří i obor rozpoznávání vzorů.

Cílem této diplomové práce je skupinový vývoj aplikace na rozpoznávání dopravních značení z videozáznamů, kde moje část se zabývá rozpoznáváním vzorů pomocí umělých neuronových sítí, zkoumáním a nastavováním parametrů, jakými lze ovlivnit chování neuronových sítí pomocí vstupních dat získané od Bc. Josefa Hrubého, který se podrobně ve své diplomové práci zabývá metodami zpracování obrazu a přípravě dat pro neuronovou síť. Třetí část, vyvíjena Bc. Janem Dospivou v jazyce C#, se zabývá spojením těchto dvou diplomových prací do výsledné aplikace.

Teoretická část se zabývá nejprve stručným popisem dopravních značení v ČR. Dále podrobným popisem umělých neuronových sítí, kde kladu největší důraz na jednotlivé učící algoritmy, které z velké části ovlivňují správnost učení a rozpoznávání. Naposled v této části popisují prostředí Wolfram Mathematica a využití balíčku NeuralNetworks pro učení neuronových sítí. Praktická část se nejprve zabývá zpracováním vstupních dat do podoby trénovací a testovací množiny. V další části se detailně zabývám učením a testováním neuronových sítí a stručným popisem uživatelského rozhraní výsledné aplikace. V závěru práce jsou vyhodnoceny celkové výsledky rozpoznávání dopravních značení pomocí umělých neuronových sítí.

## **I. TEORETICKÁ ČÁST**

## 1 DOPRAVNÍ ZNAČENÍ V ČR

Na silnicích české republiky se využívá dvou typů dopravních značek – svislé a vodorovné. Svislé značky, zobrazené na tabulích či panelech, jsou umístěny nad úrovní silnice, na rozdíl od vodorovných dopravních značek, které jsou vyznačené zpravidla na silnici. Na silnicích se smí využívat jen určitých dopravních značek (dané vyhláškou č. 30/2001 Sb.) a nesmí se nijak upravovat jejich tvar symbolů (s výjimkou značek se symboly, které mohou být natočeny).[1]

### 1.1 Dělení dopravních značek

Dle vyhlášky č. 30/2001 Sb. dělíme značky podle významu na:

#### 1. Výstražné

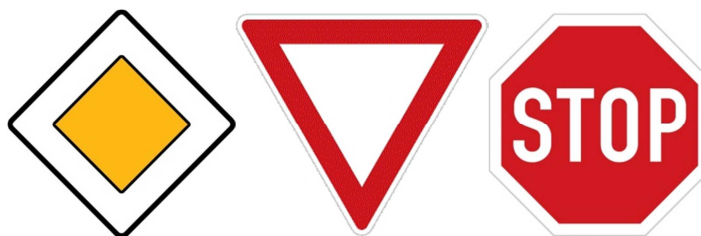
Dopravní značky, které slouží k upozornění řidičů před možným nebezpečím, neboli místa, kde musí řidič dodržovat zvýšenou opatrnost.



Obr. 1 - Ukázka vybraných výstražných značek.[25]

#### 2. Upravující přednost

Mezi nejdůležitější dopravní značky patří značky upravující přednost, které přiřazují řidičům přednost v jízdě. Nejčastější využití je při křížení komunikace s jinou komunikací.



Obr. 2 - Ukázka vybraných značek upravujících přednost.[25]

### 3. Zákazové

Dopravní značky, které udávají omezení nebo zákazy.



Obr. 3 - Ukázka vybraných zákazových značek.[25]

### 4. Příkazové

Dopravní značky, které dávají účastníkovi provozu příkazy.



Obr. 4 - Ukázka vybraných příkazových značek.[25]

### 5. Informativní

Informativní dopravní značky slouží k poskytování informací účastníku provozu nebo se snaží zlepšit jeho orientaci na vozovce.



Obr. 5 - Ukázka vybraných informativních značek.[25]

### 6. Dodatkové tabulky

Speciální typy dopravních značek, které slouží jako dodatky k výše uvedeným typům, značek, neboli slouží k zpřesnění, omezení nebo doplnění významu značky. [1]



Obr. 6 - Ukázka vybraných dodatkových tabulí.[25]

Dopravní značky se dále dělí podle provedení na:

### 1. Stále

Jedná se o dopravní značky pevně umístěny do terénu, které jsou nejčastěji umístěny vedle nebo nad silnicí.

### 2. Přenosné

Přenosné dopravní značky nejsou zabudovány do terénu, ale s určitou stabilitou zakotveny do stojanů. Tyto značky jsou nadřazené ostatním značkám.

### 3. Proměnné

Jedná se o speciální zařízení, které zobrazují účastníkům provozu aktuální informace o dopravní situaci na silnici pomocí piktogramu dopravní značky nebo textu.[1]

## 1.2 Rozměry dopravních značek

Důležitou podmínkou pro dopravní značky jsou jejich rozměry. Rozlišujeme 3 základní typy rozměrů – základní (velikost 2), zmenšené (velikost 1) a zvětšené (velikost 3). Každá z těchto velikostí má své využití, kdy zvětšené značky se využívají hlavně na dálnicích a místních silnicích 1. třídy, základní na silnicích 1. a 2. třídy a dopravně významnějších silnicích 3. třídy a zmenšené na méně významných silnicích 3. třídy.[1]

Tabulka 1 - Rozměry symbolů na vybraných značkách.

Velikost značky	Velikost významového symbolu na značce			
	Trojúhelník	Kruh	Čtverec	Obdélník
zmenšená (1)	75%	70%	-	-
základní (2)	100%	100%	100%	100%
zvětšená (3)	135%	125%	150%	200%

### 1.3 Zásady užití dopravních značek

Před použitím dopravních značek je nutné zjistit, zdali je dodrženo těchto 5 základních zásad:

#### 1. Účelnost

- Dopravní značky musí v kombinaci se světelnými a akustickými zařízeními dodržovat ucelený systém organizace a řízení provozu.
- Veškerá dopravní zařízení musí být používána tak, aby neohrozila bezpečnost a plynulost provozu či jiný veřejný zájem.
- Doba využívání dopravních značek musí být omezena na nezbytné minimum. V případě, že už není důvod dále značku používat, musí být ihned odstraněna.

#### 2. Srozumitelnost a výstižnost

- Srozumitelnost, výstižnost, jednoznačnost a úplnost dopravních značek pro každého účastníka provozu.
- Dopravní značení musí účastníkům provozu podávat co největší množství potřebných informací a vystihnout reálnou situaci následujícího úseku.
- Dopravní značení musí poskytnout účastníku provozu dostatečný čas pro jeho rozhodování.
- Dodržování vzdálenosti mezi značkou a situací, kterou značka vystihuje.

#### 3. Viditelnost

- Viditelnost dopravních značení z dostatečné vzdálenosti (nesmí být ničím překrývány).
- V případě splnutí důležitých dopravních značek s okolím je nutné tyto značky dostatečně upravit, popř. umístit stejnou značku na levý okraj vozovky.

#### 4. Údržba

- Dodržení funkce dopravní značky dostatečnou údržbou (čištění či obnova).
- Zabezpečení dopravních značek vůči deformacím, pootočení nebo posunutí (vlivem povětrnostních podmínek).[1]

## 1.4 Umístování svislých dopravních značek

Při používání dopravních značek je nutné dodržovat určité boční, směrové a výškové umístění, ale také jejich vzdálenosti, kombinace a uspořádání.[1]

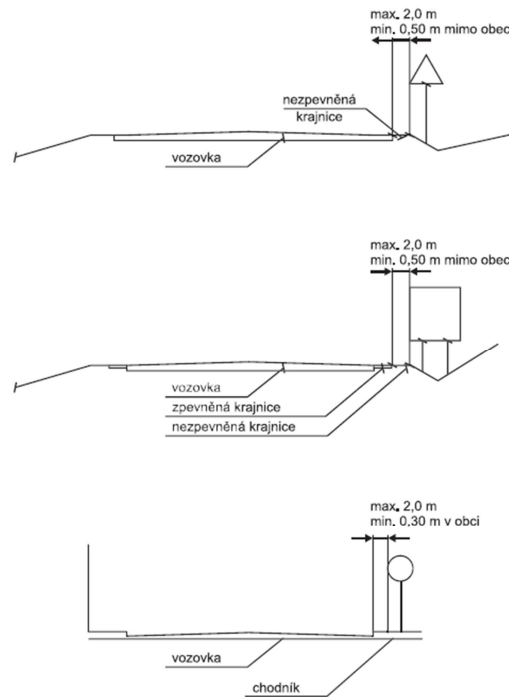
### 1.4.1 Umístění podle směru provozu

Nejčastěji jsou dopravní značky umístěny na pravém okraji vozovky. V případě, že se jedná o dopravní značku, která upozorňuje na důležitou nadcházející dopravní situaci, je možné tuto značku po určitých vzdálenostech opakovat, popřípadě umístit na levou stranu vozovky (přibližně na stejné úrovni se značkou na pravé straně vozovky) nebo nad vozovku.

V případě silnic s více jízdními pruhy se zákazové a příkazové značky umísťují nad jízdni pruh, ke kterému se vztahují. Značky „Zákaz zastavení“ a „Zákaz stání“ bývají z pravidla umístěny na té straně vozovky, ke které se vztahují.

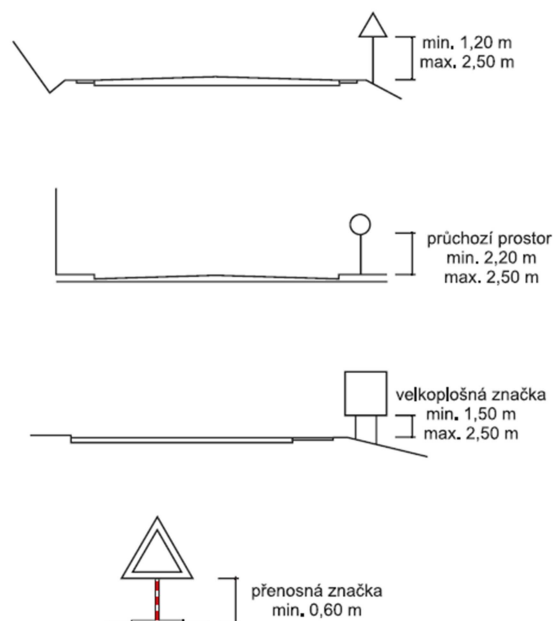
Při umístování stálých dopravních značek (mimo značky označující překážky provozu, pracovní místa, atd.) vedle vozovky je nutné dodržovat normy (ČSN 73 6101, ČSN 73 6110 a ČSN 73 6201), které říkají, že značka, včetně její nosné konstrukce, nesmí zasahovat do vymezené části dopravního prostoru. Maximální vzdálenost od vozovky nebo zpevněné krajnice a okraje dopravní značky je 2 m a minimální vzdálenost 0,5 m, v obci 0,3 m (Obr. 7).

V případě, že jsou vedle vozovky rozmístěny svodidla nebo jiná záchytná bezpečnostní zařízení, umísťují se dopravní značky za jejich deformační zónu.[1]



Obr. 7 – Boční umístění dopravních značek.

Kromě zásad pro boční umístění dopravních značek vedle vozovky existují i zásady pro výškové umístění, které chápeme jako vzdálenost mezi úrovní vozovky a nejnižší umístěnou značkou na nosné konstrukci. Nejmenší tato vzdálenost je 1,2 m, na mostních objektech 2,5 m. Pro značky umístěné v prostoru pro chodce je minimální vzdálenost od vozovky či chodníku stanovena na 2,2 m (v prostorech pro cyklisty 2,5 m).[1]



Obr. 8 – Výškové umístění dopravních značek.

Výše zmíněné zásady pro výškové umístění se nevztahují na všechny dopravní značky. Například pro desky označující železniční přejezd, „Příkazaný směr jízdy“ a „Kilometrovník“ jsou tyto zásady odlišné.

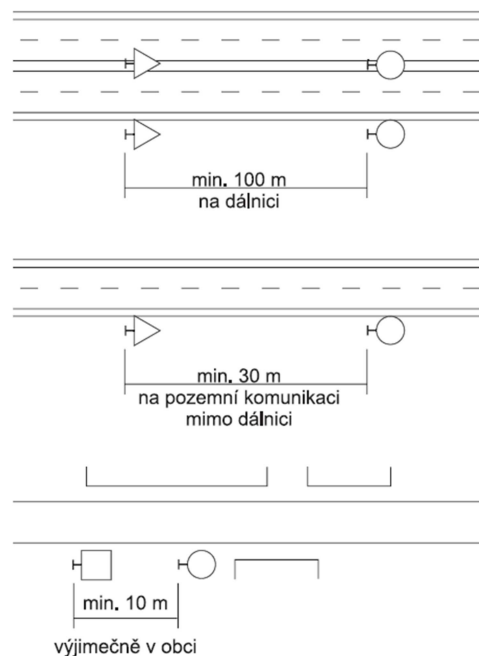
Jedná-li se o dopravní značky nad vozovkou, jsou jejich vzdálenosti od úrovně vozovky dány podle kategorie pozemní komunikace, nad kterou jsou umístěny (viz. Tabulka 2).[1]

Tabulka 2 – Vzdálenosti dopravních značek od vozovek.

	Vzdálenost (m)
Dálnice	5
Silnice I. třídy	5
Silnice II. třídy	5
Silnice III. třídy	4,7
Místní komunikace rychlostní a sběrné	4,7
Místní komunikace obslužné a účelové	4,4

#### 1.4.2 Vzdálenost mezi značkami

Rozmístění dopravních značek v podélném směru je určeno podle kategorie dopravní komunikace a doby, dostatečné pro jejich včasné zaznamenání (Obr. 9).



Obr. 9 – Vzdálenost mezi značkami.

### 1.4.3 Počet značek

V závislosti na typu dopravní značky je omezen počet značek na nosné konstrukci. Kromě směrových tabulí, návěstí před křižovatkou, značek označujících název ulice, značek označujících kulturní, turistický a komunální cíl platí, že na jedné nosné konstrukci smí být maximálně 2 značky, které mohou být ovšem doplněny o dodatkové tabulky.[1]

## 2 UMĚLÉ NEURONOVÉ SÍTĚ

Neuronové sítě jsou tzv. nedeklarativní systémy (nemusíme určovat žádná pravidla, kterými by se měla neuronová síť řídit) a patří mezi jeden z výpočetních modelů umělé inteligence. Schopností učení se získává neuronová síť postupně stále více znalostí pouze díky vstupním (popř. i výstupním) datům, aniž by znala algoritmus řešení.

Neuronové sítě se nejčastěji využívají při řešení problému, kdy není možné matematicky popsat jednotlivé vztahy nebo propojení, ale taky v případech, kdy sice dokážeme tyto vztahy popsat, ale výsledná algoritmizace úlohy je díky složitosti matematického modelu příliš složitá.

V dnešní době se neuronové sítě využívají v mnoha vědních disciplínách. Jedno z možných využití je rozpoznávání obrazů pomocí umělých neuronových sítí.[3]

### 2.1 Historie

První model neuronu byl publikován ve 40. letech 20. století Američanem Warrenem Sturgisem McCullochem, který vypracoval společně s jeho studentem Walterem Pittsem. Tento model je používán až dodnes a stal se také základem pro vytvoření první perceptronové sítě v roce 1958 Frankem Rosenblattem. Po necelých 10 letech Rosenblattův bývalý spolužák Marvin Minsky využil toho, že jeho síť má nedostatek, který publikoval v knize „Perceptron“. Tato kniha tehdy způsobila, že zájem o neuronové sítě zcela upadl až do 80. let, kdy došlo opět k obnovení zájmu o neuronové sítě díky několika průkopníkům. V těchto letech byla sepsána kniha „Learning Internal Representation by Error Propagation“, která podrobně popisuje vícevrstvé sítě. Postupem času vznikají další typy neuronových sítí, které mají také za následek nabývání významu uplatnění neuronových sítí.[2]

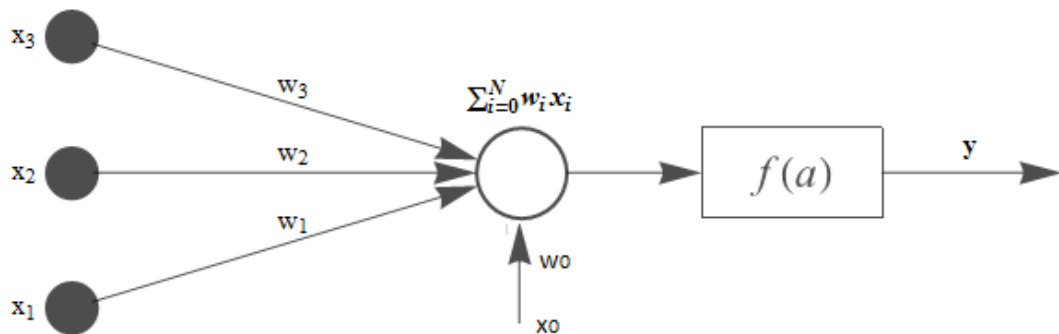
### 2.2 Model umělého neuronu

První matematický model tzv. formálního neuronu, který se využívá dodnes, byl vytvořen už ve 40. letech 20. století McCullochem a Pittsem (**Chyba! Nenalezen zdroj odkazů.**). Model tohoto formálního neuronu (dále jen neuron) se skládá ze tří částí:

- vstupní část se skládá ze vstupů a synaptických vah, jejichž koeficienty napomáhají ke zvýhodňování nebo potlačení vstupů,
- výkonná část zpracovává informace ze vstupu a vytváří výstupní odezvu,

- výstupní část přenáší výstupní informace na vstup dalších neuronů.[4]

**Vstupní vektor**



*Obr. 10 - Matematický model umělého neuronu.[5]*

Neuron má obecně  $n$  vstupů určujících vstupní vektor  $x = (x_1, x_2, x_3, \dots)$ . Každý z těchto vstupů je ohodnocen synaptickými váhami, které tvoří vektor vah  $w = (w_1, w_2, w_3, \dots)$ . Tyto váhy mají formu reálných čísel, kde každá z nich je vynásobena se vstupy, čímž jim přiřazuje důležitost. Během celého procesu dochází k výpočtům a změnám vah, které tak ovlivňují výpočet výstupů.

Součástí modelu je nezávislý vstup nazývaný práh (prahová hodnota)  $\Theta$ , přes který prochází signál k aktivační funkci. Tento práh je dán součinem váhy  $w_0$  a vstupu  $x_0$ , kde  $x_0 = 1$ .

$$\Theta = w_0 \cdot x_0 \quad (2.1)$$

Neuron má různé množství vstupů. Pokud je vstup pouze prahová hodnota  $\Theta$ , pak se jedná o sigma neuron. V případě více vstupů se vypočítá vnitřní suma, která představuje vnitřní potenciál neuronů  $A$ , tedy:

$$A = \sum_{i=1}^N w_i x_i + \Theta. \quad (2.2)$$

Je-li brán práh jako vstup neuronu, lze rovnici pro výpočet vnitřního potenciálu neuronu vyjádřit takto:

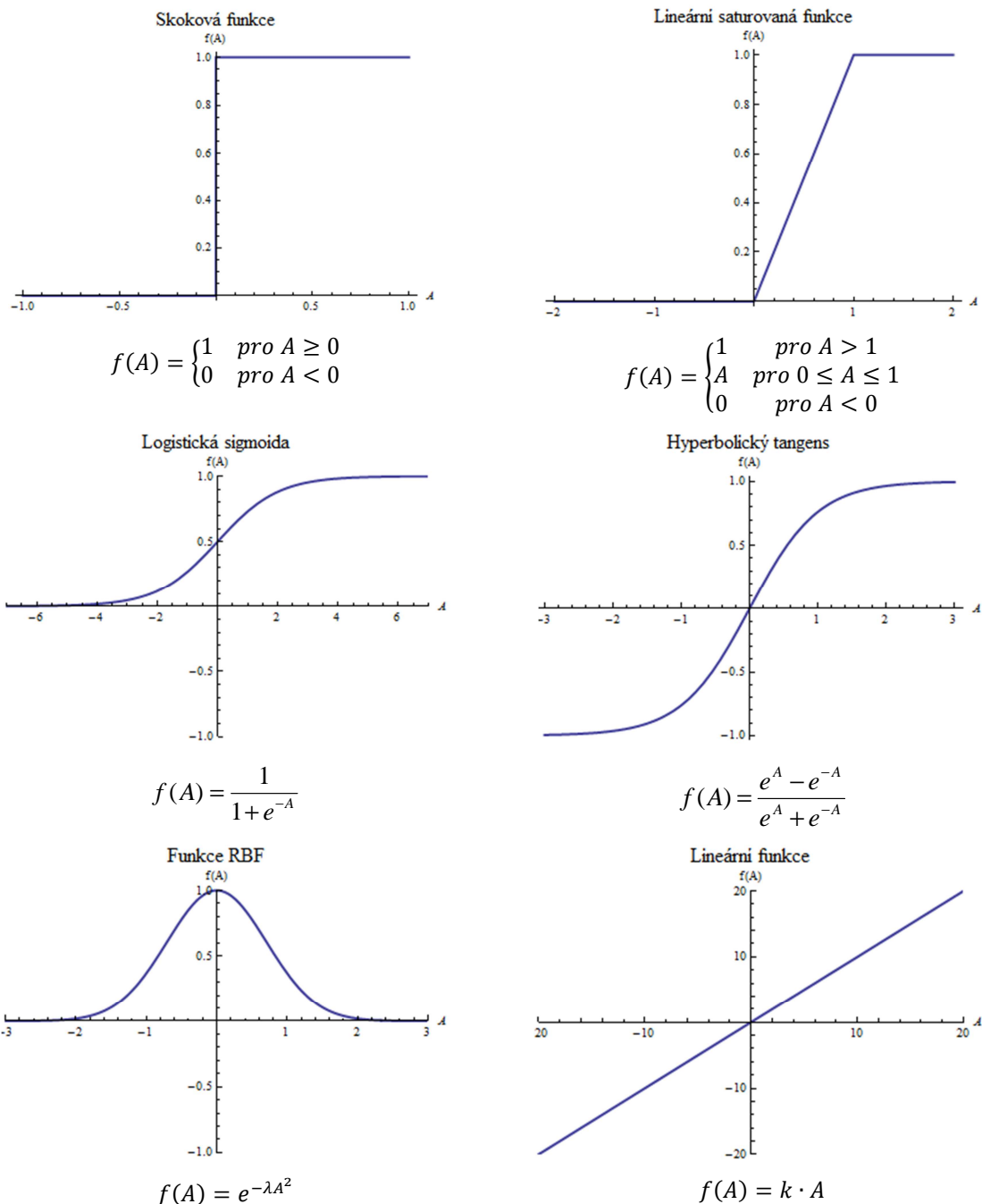
$$A = \sum_{i=0}^N w_i x_i. \quad (2.3)$$

Další fází procesu se vyhodnotí vstupy v aktivační potenciál pomocí aktivační funkce. Tento potenciál je poté přenesen v transformované podobě na výstup neuronu. U formálního neuronu se využívá jednoduchá prahová funkce  $f$ , kde

$$f(A) = \begin{cases} 1 & A \geq \Theta \\ 0 & A < \Theta \end{cases} \quad (2.4)$$

Přenosová funkce, která transformuje agregovaný signál na výstup, neboli určuje odezvu na výstupu na vstupní podnět, je důležitá pro správný chod neuronu.

Existují přenosové funkce spojité nebo s nespojitostí prvního druhu, přičemž pro všechny platí, že výsledná hodnota je vždy v intervalu -1 až 1. Volba této funkce (Obr. 11) závisí na problematice řešené úlohy.[4]



Obr. 11 – Grafy přenosových funkcí včetně jejich zápisu.[5]

Při volbě jakékoli přenosové funkce má neuron omezenou výpočetní sílu. Z tohoto důvodu vznikají spojováním neuronů neuronové sítě.[4]

## **2.3 Neuronová síť**

Neuronová síť se skládá ze vzájemně propojených formálních neuronů, kde jeden neuron je vstupem do jednoho nebo i více dalších neuronů. Topologie neuronové sítě je určena počtem neuronů a jejich propojením. Tok a zpracování informací umožňují změny stavů skrytých neuronů (neurony ležící mezi vstupními a výstupními neurony) a tím určují celkový stav neuronové sítě. Synaptické váhy vyjadřující důležitost neuronů představují konfiguraci neuronové sítě.[6]

### **2.3.1 Dělení neuronových sítí**

Neuronové sítě se dělí podle kritérií, které pak určují, o jakou síť se jedná.

#### **2.3.1.1 Z hlediska průchodu informací**

Dopředná síť (feed forward) je moderní vícevrstvá síť, která má jasně definovaný tok informací. Data jsou přivedeny na vstupy a prochází sítí, vrstvu po vrstvě, dokud nedojdou na výstup. Jedná-li se o operace třídění, neexistuje mezi vrstvami zpětná vazba. Z tohoto důvodu jsou nazývány jako dopředné neuronové sítě.

Rekurentní síť (recurent) nešíří signál pouze od vstupu na výstup, ale umožňuje i zpětnovazební přenos informace od vyšších vrstev k vrstvám nižším (pomocí rekurentních neuronů).[7]

#### **2.3.1.2 Podle počtu vrstev**

Neuronové sítě dělíme podle počtu vrstev na jednovrstvé, dvouvrstvé, třívrstvé a vícevrstvé. Mezi jednovrstvé a dvouvrstvé patří např. Kohonenova nebo Hopfieldova síť. Hlavními odlišnostmi těchto sítí jsou jejich vlastní speciální učící algoritmy a topologie. Mezi třívrstvé a vícevrstvé sítě patří nejčastěji klasická vícevrstvá síť.[4]

#### **2.3.1.3 Podle algoritmu učení**

Učení s učitelem se provádí na základě využívání trénovací množiny. Učící funkce je vždy dvojice – vstup a jemu odpovídající výstup. Po natrénování dat a uplatnění právě naučených znalostí se klasifikátor snaží odhadnout výstup. Aby bylo určeno, do jaké třídy

patří jednotlivé vstupy, musí nejprve klasifikátor porovnat nynější výstup s naučenými výstupy a upravit váhy tak, aby našel co největší shodu.

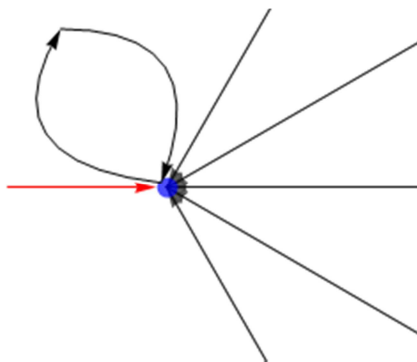
Učení bez učitele je druhým typem učení, kdy klasifikátor nevyužívá trénovací množinu (nenachází se zde žádné rozhodovací kritérium), ale využívá naopak výstupů, ze kterých si dokáže sám určit správnost pomocí zpětné vazby (samoorganizace). Učící množina je zde nahrazena výstupy z klasifikátoru.[8]

#### 2.3.1.4 Podle stylu učení

Stochastické neuronové sítě fungují na základě zavedení náhodné změny do sítě, a to buď zavedením náhodné přenosové funkce do neuronů sítě, nebo nastavením náhodných vah. Deterministické neuronové sítě na rozdíl od stochastických zajišťují váhy výpočtem.[4]

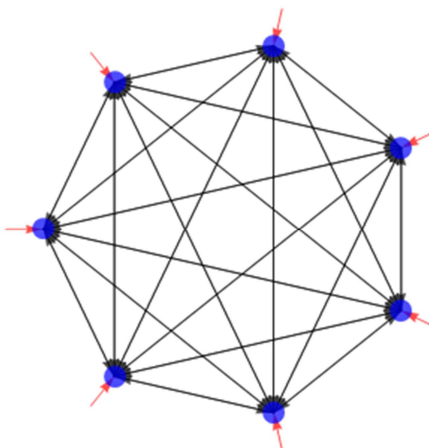
#### 2.3.1.5 Podle topologie

Z hlediska architektury neuronové sítě rozlišujeme dva typy – cyklické a acyklické. Jak již z názvu vyplývá, **cyklická** síť je tvořena neurony spojenými do kruhu, neboli do cyklu, kde výstup prvního neuronu je vstupem druhého neuronu, výstup druhého neuronu je vstupem třetího až k poslednímu neuronu, jehož výstup je vstupem prvního neuronu. V cyklické topologii sítě existuje i speciální vazba – zpětná vazba, kde výstup neuronu je zároveň jeho vstupem (Obr. 12).



Obr. 12 – Zpětná vazba neuronu.

Je-li každý výstup libovolného neuronu v síti vstupem do všech ostatních neuronů, nazýváme takovou architekturu jako úplná topologie cyklické neuronové sítě (Obr. 13).[4]



Obr. 13 – Úplná topologie cyklické neuronové sítě.

U **acyklických** sítí naopak žádné cykly ani zpětné vazby neexistují. Neurony v acyklických sítích lze rozdělit do několika vrstev. Jsou-li tyto vrstvy zobrazeny vedle sebe, pak spoje mezi neurony vedou pouze z levých vrstev do pravých. Typickým představitelem acyklické architektury je vícevrstvá neuronová síť (Obr. 14).[9]

### 2.3.2 Učení neuronové sítě

V procesu vývoje neuronové sítě dochází ke změně stavu neuronů a adaptování vah. Z důvodu těchto změn v čase se rozděluje celková dynamika sítě do tří fází – organizační, aktivní a adaptivní. Každá z těchto fází je popsána pravidly, které stanovují vývoj příslušné charakteristiky sítě.

Každý model neuronové sítě je dán konkrétními dynamikami. Jinými slovy, chceme-li specifikovat model sítě, stačí formulovat jeho organizační, aktivní a adaptivní dynamiku.[10]

#### 2.3.2.1 Organizační dynamika

Organizační dynamika slouží pro upřesnění topologie sítě, kdy nejčastěji předpokládá s neměnicí se architekturou sítě v čase, ale mohou nastat i situace, kdy dojde k případným změnám v topologii (rozšíření o další neurony a k nim příslušné spoje).[11]

#### 2.3.2.2 Aktivní dynamika

Aktivní dynamika upřesňuje počáteční **stav** sítě a sleduje jeho změnu v čase. Jedná se tedy o proces, ve kterém se na začátku předloží vektor informací na vstup, neboli nastaví se vstup sítě (stavový prostor). Jakmile dojde k inicializaci sítě, zahájí se vlastní výpočet. Vstup sítě je přepočítán na výstup přes všechny spoje včetně jejich vah. Výsledkem

výpočtu je odezva sítě na vstupní vektor, který má formu výstupního vektoru. Tento vypočítaný výstupní vektor se nakonec porovná s originálním výstupním vektorem a rozdíl mezi nimi se nazývá lokální odchylka (chyba). [11]

### 2.3.2.3 *Adaptivní dynamika*

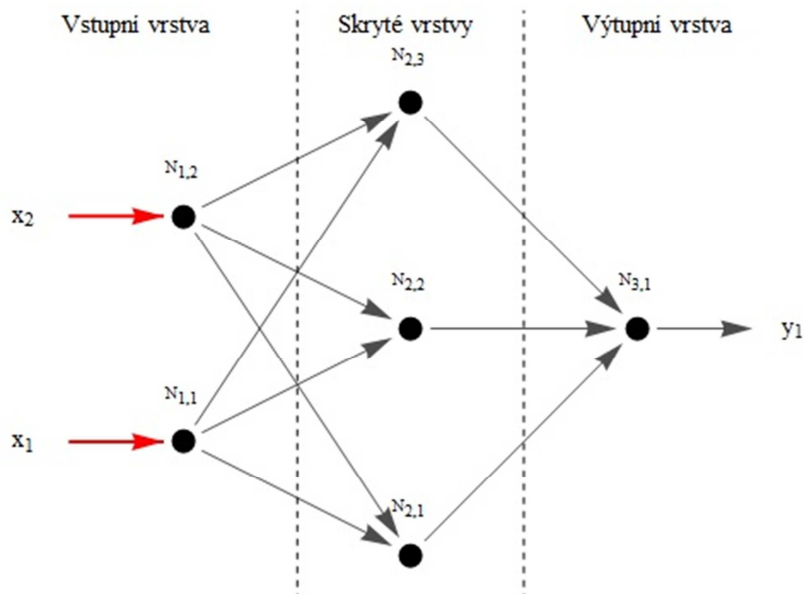
Adaptivní dynamika upřesňuje počáteční **konfiguraci** sítě a sleduje, jakým způsobem dochází ke změnám váhových ohodnocení spojů mezi jednotlivými neurony v čase. Jedná se o proces, ve kterém se nejprve nastaví váhy veškerých spojů v síti – počáteční konfigurace. Po inicializaci konfigurace sítě dojde k přepočítání vah jednotlivých spojů mezi neurony směrem od výstupu ke vstupu a k porovnání výstupní odezvy s originálním vektorem.

Po dokončení adaptivní fáze dojde opět k aktivní fázi, která přepočítá lokální odchylku a přičte k předchozí. Pokud tímto způsobem projdeme celou trénovací množinu, pak součet všech lokálních odchylek nazýváme globální odchylkou (globální chybou). Proces učení je ukončen tehdy, pokud globální odchylka je menší než námi nastavená hodnota (požadovaná chyba).[4]

## 2.4 Dopředná neuronová síť

V dopředné síti (feedforward) jsou neurony vždy rozděleny do za sebou následujících vrstev tak, že spoje mezi neurony vedou z nižší vrstvy do vrstvy vyšší, nikoliv mezi neurony ve stejné vrstvě (Obr. 14). Díky této jednosměrnosti neexistuje u dopředných sítí zpětná vazba. Topologie této sítě se obecně dělí na tři vrstvy:

- Vstupní vrstva je tvořena vstupními neurony, které slouží jako zdrojové uzly. Z této vrstvy se informace přenáší a rozdělují do dalších vrstev až do poslední vrstvy.
  - Výstupní vrstva slouží jako zprostředkovatel celkového výstupu sítě. Je poslední vrstvou v síti a skládá se z výstupních neuronů.
  - Skryté vrstvy se nacházejí mezi vstupní a výstupní vrstvou a skládají se ze skrytých (mezilehlých, vnitřních) neuronů. Slouží k napomáhání výpočtu výstupu.
- [12]



Obr. 14 – Neuronová síť s jednou skrytou vrstvou.[5]

Dopředná neuronová síť může obsahovat různý počet vrstev a neuronů v nich, podle kterých se síť označuje (např. zápis 6-8-4 označuje 6 vstupních neuronů, 8 skrytých a 4 výstupní). Počet neuronů ve vstupní a výstupní vrstvě je předem dán řešenou úlohou a tím, jak jí chceme řešit, proto nejdůležitějším parametrem je počet neuronů ve skryté vrstvě, který se volí podle složitosti aktuální řešené úlohy, přičemž neplatí předpis, že čím více vrstev a neuronů, tím lepší bude výsledek. Naopak, čím větší bude počet vrstev či neuronů, tím horší může výsledek být, jelikož může dojít k přeučení sítě. Počet skrytých neuronů můžeme nastavit dle univerzálních vzorců, které jsou ale pouze obecné a nepřesné.[13]

Pro dvouvrstvou síť platí vztah:

$$n_{skryt.} = \sqrt{n_{vstup} \cdot n_{výstup}} \quad (2.5)$$

Pro třívrstvou

$$n_{1.skryta} = n_{výstup} \cdot \left( \sqrt[3]{\frac{n_{vstup}}{n_{výstup}}} \right)^2 \quad n_{2.skryta} = n_{výstup} \cdot \left( \sqrt[3]{\frac{n_{vstup}}{n_{výstup}}} \right) \quad (2.6)$$

Výpočet dopředné sítě probíhá od vstupní vrstvy k výstupní. Nejprve se nastaví stavy vstupních neuronů a následně vypočítá vnitřní potenciál všech neuronů podle vztahu:

$$\mathcal{E}_j = \sum_{i=0}^n w_{ji} y_i \quad (2.7)$$

kde  $w_{ji}$  je váha neuronu a  $y_i$  výstup neuronu. V tomto případě  $i$  označuje nižší vrstvu a  $j$  vrstvu vyšší. V dalším kroku je na potenciál každého neuronu použita aktivační funkce. Pro ukázkou je zvolena funkce logistická sigmoida, která je dána vztahem:

$$\delta(\varepsilon) = \frac{1}{1 + e^{-\varepsilon}}. \quad (2.8)$$

Výstup neuronu se dále vypočítá podle vztahu:

$$y_j = \delta_j(\varepsilon_j). \quad (2.9)$$

Vstupní vrstva u dopředných sítí slouží pouze k distribuci vektoru vstupních hodnot k první skryté vrstvě, proto zde nemá smysl váhy uvažovat.

Tento postup výpočtu pro každou vrstvu sítě probíhá opakovaně do doby, než budou vypočítány výstupy všech neuronů. Výsledná odezva sítě je sestavena z výstupních neuronů.[12,14]

#### 2.4.1 Algoritmus Backpropagation

Učení pomocí algoritmu Backpropagation je založeno na gradientní metodě (metoda gradientního sestupu). Tento způsob učení má také mnoho vylepšení, z nichž většina učí síť rychleji. V tomto případě se jedná o učení s učitelem, můžeme tedy určit chybovou funkci

$$E = \frac{1}{2} \sum_j (d_j - y_j)^2, \quad (2.10)$$

kteřá představuje kvadrát mezi očekávaným a reálným výstupem, kde  $y_j$  je očekávaný výstup  $j$ -tého vzoru. Dalším krokem je provedení parciální derivace podle vah vyjádřený vztahem:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial \varepsilon_j} \frac{\partial \varepsilon_j}{\partial w_{ij}}, \quad (2.11)$$

díky které zjistíme, jakým směrem je nejbližší množina. Pokud tuto parciální derivaci aplikujeme na logistickou sigmoidu, získáme:

$$\delta_j = f'(\varepsilon_j)(d_j - y_j) = (1 - y_j)y_j(d_j - y_j), \quad (2.12)$$

kde  $\delta$  je velikost změny váhy. Celé učení lze popsat v 5 fázích.

1. fáze: Inicializace vah náhodně malými hodnotami.
2. fáze: Předložení vstupního vektoru  $x = x_1, x_2, \dots, x_n$  a odpovídajícího požadovaného výstupu  $d = d_1, d_2, \dots, d_m$ .
3. fáze: Výpočet aktuálních výstupů sítě.
4. fáze: Adaptace vah – nová hodnota váhy je vypočítána následujícím vztahem:

$$w_{ij}^s(t+1) = w_{ij}^s(t) + \eta \delta_j^{s+1} y_i^s, \quad (2.13)$$

kde  $w_{ij}^s(t)$  je váha mezi  $i$ -tým a  $j$ -tým neuronem v  $s$ -té vrstvě  $t$ -ho kroku adaptace,  $\eta$  parametr učení,  $\delta_j^s$  chyba neuronů v aktuální vrstvě a  $y_i^s$  výstup neuronu aktuální vrstvy.

V případě poslední vrstvy  $s = M$  a

$$\delta_j^M = y_j(1 - y_j)(d - y_j). \quad (2.14)$$

Nejedná-li se o poslední vrstvu, pak ve zbylých vrstvách platí:

$$\delta_j^s = y_j^{s+1}(1 - y_j^{s+1}) \sum_{k=1}^{p^{s+1}} \delta_k^{s+1} \cdot w_{jk}^{s+1}, \quad (2.15)$$

kde  $s = M - 1, \dots, 1$  a  $k$  prochází přes všechny neurony v  $s + 1$  vrstvě.

5. fáze: Podmínka ukončení – pokud se hodnoty vah nemění nebo došlo k dosažení maximálního povoleného počtu změn vah, učení končí. Jinak pokračuj fází 2.[14]

#### 2.4.2 Levenberg-Marquardtovo učení

V této části bude odvození algoritmu Levenberg-Marquardt uvedeno do tří částí:

- Metoda největšího spádu
- Newtonova metoda
- Gauss-Newtonův algoritmus
- Levenberg-Marquardtův algoritmus

Před odvozením této metody je třeba se seznámit s významy používaných indexů:

- $p$  je index vzorů, od 1 do  $P$ , kde  $P$  je počet vzorů,
- $m$  je index výstupu, od 1 do  $M$ , kde  $M$  je počet výstupů,
- $i$  a  $j$  jsou indexy vah, od 1 do  $N$ , kde  $N$  je počet vah.
- $k$  je index iterací.

### 2.4.2.1 Metoda největšího spádu

Metoda největšího spádu je algoritmus prvního řádu, jelikož používá pouze derivace prvního řádu chybové funkce pro nalezení minima v chybovém prostoru.

Za normálních okolností je gradient  $g$  definován jako derivace prvního řádu chybové funkce (2.10):

$$g = \frac{\partial E}{\partial w} = \left[ \frac{\partial E}{\partial w_1} \quad \frac{\partial E}{\partial w_2} \quad \dots \quad \frac{\partial E}{\partial w_N} \right]^T. \quad (2.16)$$

S definicí gradientu  $g$  z rovnice (2.16), pravidlo aktualizace algoritmu nejstrmějšího spádu může být zapsáno jako

$$w_{k+1} = w_k - \alpha g_k, \quad (2.17)$$

kde  $\alpha$  je učicí konstanta (počet kroků učení).

Učící proces algoritmu největšího spádu je asymptotická konvergence. Všechny prvky vektoru gradientu mohou být velmi malé, což může způsobit pouze nepatrnou změnu vah.[15]

### 2.4.2.2 Newtonova metoda

Newtonova metoda předpokládá, že veškeré složky gradientu  $g_1, g_2, \dots, g_N$  jsou funkcemi vah a všechny váhy jsou lineárně nezávislé:

$$\begin{cases} g_1 = F_1(w_1, w_2 \dots w_N) \\ g_2 = F_2(w_1, w_2 \dots w_N), \\ \dots \\ g_N = F_N(w_1, w_2 \dots w_N) \end{cases}, \quad (2.18)$$

kde  $F_1, F_2, \dots, F_N$  jsou nelineární vztahy mezi váhami a souvisejícími gradienty.

Rozložme jednotlivé  $g_i$  ( $i = 1, 2, \dots, N$ ) ze vztahu (2.18) podle Taylorovy řady a vezměme první řád aproximace:

$$\begin{cases} g_1 \approx g_{1,0} + \frac{\partial g_1}{\partial w_1} \Delta w_1 + \frac{\partial g_1}{\partial w_2} \Delta w_2 + \dots + \frac{\partial g_1}{\partial w_N} \Delta w_N \\ g_2 \approx g_{2,0} + \frac{\partial g_2}{\partial w_1} \Delta w_1 + \frac{\partial g_2}{\partial w_2} \Delta w_2 + \dots + \frac{\partial g_2}{\partial w_N} \Delta w_N. \\ g_N \approx g_{N,0} + \frac{\partial g_N}{\partial w_1} \Delta w_1 + \frac{\partial g_N}{\partial w_2} \Delta w_2 + \dots + \frac{\partial g_N}{\partial w_N} \Delta w_N \end{cases} \quad (2.19)$$

Kombinací definic gradientu vektoru  $g$  v (2.16) lze určit, že

$$\frac{\partial g_i}{\partial w_j} = \frac{\partial \left( \frac{\partial E}{\partial w_j} \right)}{\partial w_j} = \frac{\partial^2 E}{\partial w_i \partial w_j}. \quad (2.20)$$

Vložením rovnice (2.20) do (2.19):

$$\begin{cases} g_1 \approx g_{1,0} + \frac{\partial^2 E}{\partial w_1^2} \Delta w_1 + \frac{\partial^2 E}{\partial w_1 \partial w_2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_1 \partial w_N} \Delta w_N \\ g_2 \approx g_{2,0} + \frac{\partial^2 E}{\partial w_2 \partial w_1} \Delta w_1 + \frac{\partial^2 E}{\partial w_2^2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_2 \partial w_N} \Delta w_N \\ \dots \\ g_N \approx g_{N,0} + \frac{\partial^2 E}{\partial w_N \partial w_1} \Delta w_1 + \frac{\partial^2 E}{\partial w_N \partial w_2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_N^2} \Delta w_N \end{cases} \quad (2.21)$$

Ve srovnání s metodou největšího spádu, derivaci druhého řádu chybové funkce je třeba vypočítat pro každou složku gradientu.

Za účelem získání minima chybové funkce  $E$  by měl být každý prvek vektoru gradientu nulový. Jestliže jsou tedy levé strany rovnice (2.21) všechny nulové, pak:

$$\begin{cases} 0 \approx g_{1,0} + \frac{\partial^2 E}{\partial w_1^2} \Delta w_1 + \frac{\partial^2 E}{\partial w_1 \partial w_2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_1 \partial w_N} \Delta w_N \\ 0 \approx g_{2,0} + \frac{\partial^2 E}{\partial w_2 \partial w_1} \Delta w_1 + \frac{\partial^2 E}{\partial w_2^2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_2 \partial w_N} \Delta w_N \\ \dots \\ 0 \approx g_{N,0} + \frac{\partial^2 E}{\partial w_N \partial w_1} \Delta w_1 + \frac{\partial^2 E}{\partial w_N \partial w_2} \Delta w_2 + \dots + \frac{\partial^2 E}{\partial w_N^2} \Delta w_N \end{cases} \quad (2.22)$$

V rámci řešení pak může být váhový prostor opakovaně aktualizován. Výsledek kombinace rovnice (2.16) a (2.22) může být zapsán v maticovém tvaru:

$$\begin{bmatrix} -g_1 \\ -g_2 \\ \dots \\ -g_N \end{bmatrix} = \begin{bmatrix} -\frac{\partial E}{\partial w_1} \\ \frac{\partial E}{\partial w_2} \\ \dots \\ \frac{\partial E}{\partial w_N} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 E}{\partial w_1^2} & \frac{\partial^2 E}{\partial w_1 \partial w_2} & \dots & \frac{\partial^2 E}{\partial w_1 \partial w_N} \\ \frac{\partial^2 E}{\partial w_2 \partial w_1} & \frac{\partial^2 E}{\partial w_2^2} & \dots & \frac{\partial^2 E}{\partial w_2 \partial w_N} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 E}{\partial w_N \partial w_1} & \frac{\partial^2 E}{\partial w_N \partial w_2} & \dots & \frac{\partial^2 E}{\partial w_N^2} \end{bmatrix} \times \begin{bmatrix} \Delta w_1 \\ \Delta w_2 \\ \dots \\ \Delta w_N \end{bmatrix}. \quad (2.23)$$

kde čtvercová matice je hessian:

$$H = \begin{bmatrix} \frac{\partial^2 E}{\partial w_1^2} & \frac{\partial^2 E}{\partial w_1 \partial w_2} & \dots & \frac{\partial^2 E}{\partial w_1 \partial w_N} \\ \frac{\partial^2 E}{\partial w_2 \partial w_1} & \frac{\partial^2 E}{\partial w_2^2} & \dots & \frac{\partial^2 E}{\partial w_2 \partial w_N} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 E}{\partial w_N \partial w_1} & \frac{\partial^2 E}{\partial w_N \partial w_2} & \dots & \frac{\partial^2 E}{\partial w_N^2} \end{bmatrix}. \quad (2.24)$$

Kombinací rovnic (2.16) a (2.24) s rovnicí (2.23) získáme:

$$-g = H\Delta w, \quad (2.25)$$

kde

$$\Delta w = -H^{-1}g. \quad (2.26)$$

Pravidlo aktualizace pro Newtonovu metodu je

$$w_{k+1} = w_k - H_k^{-1}g_k. \quad (2.27)$$

Porovnáním rovnic (2.17) a (2.27) si můžeme všimnout, že zvolený počet kroků je dán invertovaným hesianem.[16,17]

### 2.4.2.3 Gauss-Newtonův algoritmus

Je-li použita Newtonova metoda pro úpravu vah za účelem získání hessianu  $H$ , výpočet derivace druhého řádu chybové funkce může být velice komplikovaný. Za účelem zjednodušení výpočetního procesu se využívá Jacobiho matice  $J$ , zapsána jako:

$$J = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w_1} & \frac{\partial e_{1,1}}{\partial w_2} & \dots & \frac{\partial e_{1,1}}{\partial w_N} \\ \frac{\partial e_{1,2}}{\partial w_1} & \frac{\partial e_{1,2}}{\partial w_2} & \dots & \frac{\partial e_{1,2}}{\partial w_N} \\ \dots & \dots & \dots & \dots \\ \frac{\partial e_{1,M}}{\partial w_1} & \frac{\partial e_{1,M}}{\partial w_2} & \dots & \frac{\partial e_{1,M}}{\partial w_N} \\ \dots & \dots & \dots & \dots \\ \frac{\partial e_{p,1}}{\partial w_1} & \frac{\partial e_{p,1}}{\partial w_2} & \dots & \frac{\partial e_{p,1}}{\partial w_N} \\ \frac{\partial e_{p,2}}{\partial w_1} & \frac{\partial e_{p,2}}{\partial w_2} & \dots & \frac{\partial e_{p,2}}{\partial w_N} \\ \dots & \dots & \dots & \dots \\ \frac{\partial e_{p,M}}{\partial w_1} & \frac{\partial e_{p,M}}{\partial w_2} & \dots & \frac{\partial e_{p,M}}{\partial w_N} \end{bmatrix}. \quad (2.28)$$

Integrovaní rovnice (2.10) a (2.16) mohou být prvky vektoru gradientu vypočítány jako:

$$g_i = \frac{\partial E}{\partial w_i} = \frac{\partial \left( \frac{1}{2} \sum_{p=1}^P \sum_{m=1}^M e_{p,m}^2 \right)}{\partial w_i} = \sum_{p=1}^P \sum_{m=1}^M \left( \frac{\partial e_{p,m}}{\partial w_i} e_{p,m} \right). \quad (2.29)$$

Kombinací rovnic (2.28) a (2.29), vztah mezi Jacobiho maticí  $J$  a vektorem gradientů  $g$  bude vyjádřen:

$$g = Je, \quad (2.30)$$

kde vektor chyb  $e$  má tvar:

$$e = \begin{bmatrix} e_{1,1} \\ e_{1,2} \\ \dots \\ e_{1,M} \\ \dots \\ e_{P,1} \\ e_{P,2} \\ \dots \\ e_{P,M} \end{bmatrix}. \quad (2.31)$$

Vložením rovnice (2.10) do (2.24), prvek  $i$ -tého řádku a  $j$ -tého sloupce matice hessianu lze vypočítat jako:

$$h_{i,j} = \frac{\partial^2 E}{\partial w_i \partial w_j} = \frac{\partial^2 \left( \frac{1}{2} \sum_{p=1}^P \sum_{m=1}^M e_{p,m}^2 \right)}{\partial w_i \partial w_j} = \sum_{p=1}^P \sum_{m=1}^M \frac{\partial e_{p,m}}{\partial w_i} \frac{\partial e_{p,m}}{\partial w_j} + S_{i,j}, \quad (2.32)$$

kde  $S_{i,j}$  je roven:

$$S_{i,j} = \sum_{p=1}^P \sum_{m=1}^M \frac{\partial^2 e_{p,m}}{\partial w_i \partial w_j} e_{p,m}. \quad (2.33)$$

Vztah mezi hesianem a Jacobiho maticí lze přepsat jako:

$$H \approx J^T J. \quad (2.34)$$

Kombinací rovnic (2.27), (2.30) a (2.34) je pravidlo pro úpravu vah Gauss-Newtonovým algoritmem prezentováno jako:

$$w_{k+1} = w_k - (J_k^T J_k)^{-1} J_k e_k. \quad (2.35)$$

Výhodou gauss-Newtonova algoritmu oproti Newtonově metodě (2.27) je, že nevyžaduje výpočet derivací druhého řádu chybové funkce nahrazením za Jacobiho matici  $J$ . Nicméně, Gauss-Newtonův algoritmus stále čelí stejnému konvergentnímu problému jako Newtonův algoritmus pro komplexní optimalizaci chybového prostoru.[18]

#### 2.4.2.4 Levenberg-Marquardtův algoritmus

Ve snaze se ujistit, že aproximovaný hessian  $J^T J$  je invertovatelný, Levenberg-Marquardtův algoritmus zavádí další aproximace hessianu:

$$H \approx J^T J + \mu I, \quad (2.36)$$

kde  $\mu$  je součinitel kombinace (je vždy kladný) a  $I$  je jednotková matice.

V rovnici (2.36) si můžeme všimnout, že prvky na hlavní diagonále aproximovaného hessianu budou větší jak nula. Proto si u této aproximace můžeme být jisti, že matice  $H$  je vždy invertovatelná.

Kombinací rovnic (2.35) a (2.36) a je pravidlo pro úpravu vah Levenberg-Marquardtovým algoritmem prezentováno jako:

$$w_{k+1} = w_k - (J_k^T J_k + \mu I)^{-1} J_k e_k. \quad (2.37)$$

Levenberg-Marquardtův algoritmus využívá kombinaci metody největšího spádu a gauss-Newtonova algoritmu (přepíná mezi nimi během trénovacího procesu). Je-li koeficient kombinace  $\mu$  velmi malý, rovnice (2.37) se přibližuje rovnici (2.35), je použit gauss-Newtonův algoritmus. V opačném případě, pokud je  $\mu$  velmi velký, rovnice 2.37 se přibližuje rovnici (2.17), pak je použita metoda největšího spádu.

Je-li koeficient kombinace  $\mu$  v rovnici (2.37) velmi velký, může být interpretován jako učící koeficient v metodě největšího spádu ((2.17)):

$$\alpha = \frac{1}{\mu}. \quad (2.38)$$

Celý proces učení pomocí Levenberg-Marquardtovy metody lze popsat v několika fázích:

**1. fáze** Výpočet hodnot  $net$ , sklonů a výstupů ze všech neuronů první vrstvy:

$$net_j^1 = \sum_{i=1}^{n_i} I_i w_{j,i}^1 + w_{j,0}^1, \quad (2.39)$$

$$y_j^1 = f_j^1(net_j^1), \quad (2.40)$$

$$s_j^1 = \frac{\partial f_j^1}{\partial net_j^1}, \quad (2.41)$$

kde  $I_i$  jsou vstupy sítě, index „1“ představuje první vrstvu,  $j$  je index neuronů v první vrstvě,  $net$  je součet vážených hodnot a  $s$  představuje sklon aktivační funkce.

**2. fáze** Použít výstupy neuronů první vrstvy jako vstupy všech neuronů v druhé vrstvě, provést podobný výpočet  $net$  hodnot, sklonů a výstupů:

$$net_j^2 = \sum_{i=1}^{n_1} y_i^1 w_{j,i}^2 + w_{j,0}^2, \quad (2.42)$$

$$y_j^2 = f_j^2(net_j^2), \quad (2.43)$$

$$s_j^2 = \frac{\partial f_j^2}{\partial net_j^2}. \quad (2.44)$$

**3. fáze** Použít výstupy neuronů druhé vrstvy jako vstupy všech neuronů ve výstupní (třetí) vrstvě, provést podobný výpočet  $net$  hodnot, sklonů a výstupů:

$$net_j^3 = \sum_{i=1}^{n_2} y_i^2 w_{j,i}^3 + w_{j,0}^3, \quad (2.45)$$

$$o_j = f_j^3(net_j^3), \quad (2.46)$$

$$s_j^3 = \frac{\partial f_j^3}{\partial net_j^3}. \quad (2.47)$$

S výsledky dopředného výpočtu pro daný výstup  $j$  může být zpětné šíření provedeno následovně:

**4. fáze** Vypočítej chybu na výstupu  $j$  a počáteční  $\delta$  jako sklon výstupu  $j$ :

$$e_j = d_j - o_j, \quad (2.48)$$

$$\delta_{j,j}^3 = s_j^3, \quad (2.49)$$

$$\delta_{j,k}^3 = 0, \quad (2.50)$$

kde  $d_j$  je požadovaný výstup na výstupu  $j$ ,  $o_j$  je skutečný výstup na výstupu  $j$ ,  $\delta_{j,j}^3$  je vlastní zpětné šíření,  $\delta_{j,k}^3$  je zpětné šíření neuronů ve stejné vrstvě (výstupní vrstvě).

**5. fáze** Zpětné šíření  $\delta$  ze vstupů třetí vrstvy na výstupy druhé vrstvy.

$$\delta_{j,k}^2 = w_{j,k}^3 \delta_{j,j}^3, \quad (2.51)$$

kde  $k$  je index neuronů v druhé vrstvě.

**6. fáze** Zpětné šíření  $\delta$  z výstupů druhé vrstvy na vstupy druhé vrstvy.

$$\delta_{j,k}^2 = \delta_{j,k}^2 s_k^2. \quad (2.52)$$

**7. fáze** Zpětné šíření  $\delta$  ze vstupů druhé vrstvy na výstupy první vrstvy.

$$\delta_{j,k}^1 = \sum_{i=1}^{n_2} w_{j,i}^2 \delta_{j,i}^2. \quad (2.53)$$

**8. fáze** Zpětné šíření  $\delta$  z výstupů první vrstvy na vstupy první vrstvy.

$$\delta_{j,k}^1 = \delta_{j,k}^1 s_k^1. \quad (2.54)$$

Fáze 4 až 8 (neboli proces zpětného šíření) se pro další výstupy opakuje.[19]

### 3 WOLFRAM MATHEMATICA

Wolfram Mathematica je programovací prostředí, které sjednocuje nástroje numeriky a symboliky, grafiku a dokumentaci a umožňuje propojení s jinými aplikacemi. Byla založena již v roce 1987 Stefenem Wolframem společností Wolfram Research Inc. Mathematica se považuje za sjednotitele, jelikož dříve existovaly samostatné sady, které řešily právě jen určité úlohy (numerické, grafické, atd.). Wolfram Mathematica veškeré tyto sady integroval do jednoho produktu.

Velkou předností tohoto softwaru je symbolické programování, to znamená, že grafy, funkce, data i programy lze vyjádřit jako symbolický výraz, díky čemuž přidává softwaru na jednoduchosti.

Mathematica využívá nezávislého dokumentu tzv. notebooku. Tento dokument je složen z buněk seřazených za sebou. Pomocí rozbalovacích oken lze tyto buňky otevírat či zavírat. Oproti jiným programovacím softwarům, Mathematica pomocí buněk dokáže oddělovat vstupy, výstupy, nadpisy, texty, grafiku atd.[21]



*Obr. 15 – Logo Wolfram Mathematica 9.*

#### 3.1 Základní složky softwaru

##### 3.1.1 Dokumentační systém notebook

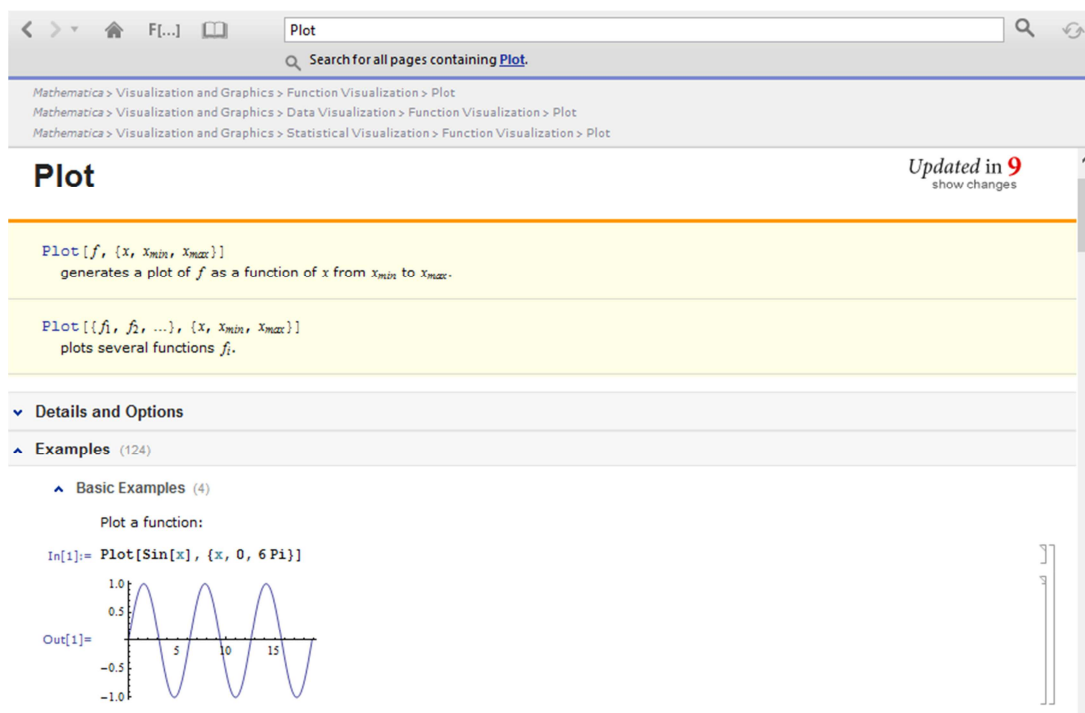
Od jednoduchých výpočtů až po kompletní propracované dynamické rozhraní, tohle vše lze provádět se standardním interaktivním rozhraním Mathematica pomocí notebooku. Notebooky jsou unikátní výpočetní dokumenty, které podporují dynamické rozhraní, různé typy vstupů, grafické vstupy, automatické napovídání příkazů, kompletní rozhraní na vysoké úrovni a mnoho dalších funkcí a možností.

Notebook se skládá z buněk, kde každé buňce je přiřazen určitý styl, který určuje jeho úlohu v notebooku. Mathematica nabízí také menu a klávesové zkratky pro pohodlnější tvorbu buněk s různými styly nebo změnu existujících buněk.

Další jedinečností softwaru Mathematica je schopnost exportu notebooku do jiných formátů, jakými jsou HTML a TeX, lze je odeslat emailem nebo přímo vložit na webovou stránku, FTP či server.[21]

### 3.1.2 Interaktivní nápověda

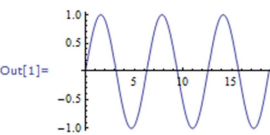
Nápověda softwaru Mathematica poskytuje uživateli podrobnou dokumentaci pro veškeré funkce. Na rozdíl od jiných softwarů, Mathematica nabízí ke každé funkci hned několik příkladů použití, včetně příkladů nastavení jednotlivých parametrů (Obr. 16). Tyto příklady lze v nápovědě upravovat a provádět, aniž by výsledky zasahovaly do vlastního kódu. Změny v nápovědě nejsou trvalé. Potřebuje-li uživatel obnovit předchozí příklady, stačí obnovit aktuální stránku, čímž se obnoví i uvedené příklady použití funkce.



The screenshot displays the Mathematica help page for the `Plot` function. At the top, there is a search bar and navigation links. The main heading is **Plot**, with a note 'Updated in 9 show changes'. Below the heading, two function signatures are listed:

- `Plot[f, {x, xmin, xmax}]` generates a plot of  $f$  as a function of  $x$  from  $x_{\min}$  to  $x_{\max}$ .
- `Plot[{f1, f2, ...}, {x, xmin, xmax}]` plots several functions  $f_i$ .

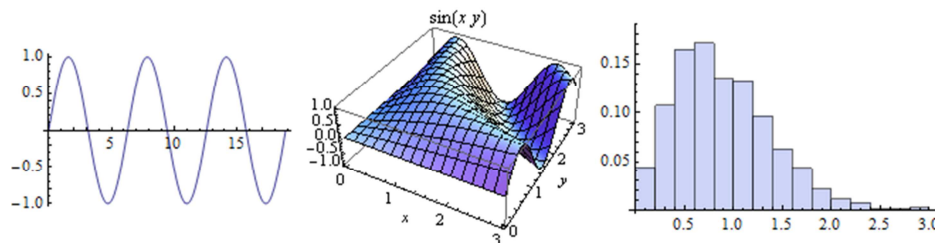
Under the heading **Details and Options**, there is a section for **Examples (124)**, which includes a sub-section **Basic Examples (4)**. One example is shown:

Plot a function:  
In[1]:= `Plot[Sin[x], {x, 0, 6 Pi}]`  
Out[1]= 

Obr. 16 – Nápověda softwaru Mathematica.

### 3.1.3 Grafika

Wolfram Mathematica poskytuje uživateli velké množství grafických vzorů pro vykreslení výsledků, jako jsou vrstevnicové grafy, 2D a 3D grafy, grafy hustoty a další specializované grafy.



Obr. 17 – Ukázka grafických vzorů softwaru Mathematica.

Mathematica také využívá své symbolické architektury a dynamického rozhraní k jednoduchému a pohodlnému vytváření různých grafik a vizualizace informací. S daty pracuje ihned, vytváří dynamické vizualizace nejrůznějších formátů.

### 3.1.4 Paleta nástrojů

Nástrojové palety umožňují okamžitý přístup k mnoha funkcím (vkládání speciálních znaků, matematických funkcí, 2D a 3D grafů, atd.), které jsou zabudované v softwaru Mathematica. Mathematica dále nabízí vytvoření vlastní palety, do které si může uživatel vložit např. nejpoužívanější symboly či funkce.[21]

## 3.2 Balíček NeuralNetworks

NeuralNetworks je tzv. přídavek softwaru Mathematica navržen pro trénování, vizualizaci a validaci modelů neuronové sítě. Tento model je nastaven nebo naučen používat data z daného zdroje jako vstup, neboli jako trénovací množinu. Po úspěšném naučení je neuronová síť schopna provádět klasifikaci, odhad, predikci nebo simulaci na nových datech ze stejného nebo podobného zdroje. Balíček NeuralNetworks podporuje různé typy učení a učících algoritmů.

Většina funkcí v balíčku NeuralNetworks podporuje celou řadu různých možností, které lze použít k úpravě algoritmů. Nicméně, výchozí hodnoty jsou nastaveny tak, aby byly výsledky přijatelné pro širokou škálu problémů, což umožňuje rychlé použití jen několika příkazů pro práci s neuronovou sítí.[12]

### 3.2.1 Funkce balíčku

Tento balíček podporuje hned několik typů neuronových sítí:

- Funkce Radiální báze (RBF)
- Dopředná síť
- Dynamické sítě

- Hopfieldova síť
- Perceptron
- Vektorové kvantování
- Síť s učením bez učitele

Funkce v balíčku jsou konstruovány tak, že minimální množství informací, jenž má být specifikováno uživatelem (např. počet vstupů a výstupů v síti) jsou automaticky rozpoznány z rozměrů dat, takže je není třeba jednoznačně zadávat.

Naučené síť jsou obsaženy ve zvláštních objektech s hlavičkou, která popisuje typ sítě. Nemusí se tak sledovat všechny parametry a další informace obsažené v modelu neuronové sítě, jelikož je vše obsaženo v tomto objektu sítě.

Důležité informace se během trénování sítě ukládají a na konci trénování jsou vypsány do speciálního trénovacího záznamu. Tento záznam může být použit pro analýzu úspěšnosti učení, popřípadě k získání mezilehlých hodnot učení.

Struktura dopředné sítě a radiální bázové funkce může být upravena tak, aby se přizpůsobila na konkrétní problém (např. aktivační funkce neuronu může být změněna na jinou vhodnou funkci).

Pro dopředné, radiální bázové funkce a dynamické neuronové síť jsou váhy neustále upravovány na základě použití gradientní metody. Levenberg-Marquardt algoritmus se používá ve výchozím nastavení, protože je považován za nejlepší volbu pro většinu řešených problémů.[12]

### 3.2.2 Možnosti sítě

Tato část popisuje konvence společné pro všechny typy neuronových sítí podporovaných balíčkem NeuralNetworks.[22]

#### 3.2.2.1 Formát vstupních dat

Aby byly veškeré funkce k dispozici, musí být nejprve načten balíček NeuralNetworks pomocí příkazu:

```
<<NeuralNetworks`.
```

Před učením sítě je třeba nejprve načíst data  $\{x_i, y_i\}_{i=1}^N$  obsahující  $N$  vstupně-výstupních párů (všechny funkce balíčku vyžadují stejný formát vstupních dat). Veškeré

vstupní a výstupní data jsou uspořádány ve formě matice softwaru Mathematica (každý vstupní vektor  $x_i$  je  $i$ -tým řádkem vstupní matice a každý výstupní vektor  $y_i$  je  $i$ -tým řádkem výstupní matice). V případě jednoho vstupního a výstupního vektoru se nepoužívá zápis do matice.[22]

### 3.2.2.2 Zápisy a význam funkcí

Většina typů neuronových sítí se spoléhají na následujících pět příkazů, kde \* je nahrazena názvem typu sítě.

Tabulka 3 – Popis hlavních příkazů balíčku NeuralNetworks.

Název	význam
Initialize*	Inicializuje neuronovou síť daného typu
*Fit	Trénování sítě daného typu
NetPlot	Zobrazení neuronové sítě
NetInformation	Vrací řetězec informací o neuronové síti
NeuronDelete	Vymaže neuron z existující sítě

Příkaz `Initialize*` vytvoří objekt neuronové sítě s hlavičkou, která se rovná názvu sítě. `*Fit` je list se dvěma prvky. První prvek je trénovací verze sítě, druhý je objekt s hlavičkou `Record*`, který obsahuje zaznamenané informace o trénování. Příkaz `NetInformation` aplikovaný na síť vrací řetězec informací o síti.[12,22]

### 3.2.2.3 Formát sítě

Naučená síť je identifikována hlavičkou a listem prvků následujícím způsobem:

- Hlavička listu identifikuje typ neuronové sítě.
- První položka listu obsahuje parametry sítě.
- Druhá položka listu obsahuje seznam pravidel, označující různé vlastnosti sítě.

Jako příklad uvedeme vytvoření sítě Perceptron (Obr. 18).

```
In[1]= << NeuralNetworks`
In[2]= InitializePerceptron[{{1., -1.}, {-1., 1.}}, {1, 0}]
Out[2]= Perceptron[{w, b}, {CreationDate -> {2014, 5, 12, 19, 41, 50.5376200}, AccumulatedIterations -> 0}]
```

Obr. 18 – Vytvoření sítě Perceptron.

Hlavička sítě je Perceptron, první položka obsahuje parametry modelu neuronové sítě, které jsou zapsány pro perceptron symbolem  $\{w, b\}$  a druhá položka je tvořena dvěma parametry – `CreationDate` indikuje, kdy byla síť vytvořena a `AccumulatedIterations` udává počet iterací učení (v tomto případě učení neproběhlo).[22]

### 3.2.3 Dopředná síť FeedForward

Inicializace sítě se provádí příkazem

```
InitializeFeedForwardNet[x, y, nh, opts],
```

kde  $x$  jsou vstupní data,  $y$  výstupní data,  $nh$  počet vrstev a neuronů v nich a  $opts$  další parametry sítě (Tabulka 4).

Tabulka 4 – Parametry inicializace FeedForward sítě včetně jejich významu.

parametr	Výchozí hodnota	význam
<code>LinearPart</code>	False	lineární model umístěn paralelně v síti
<code>Neuron</code>	Sigmoid	aktivační funkce neuronů
<code>BiasParameters</code>	True	zahrnutí parametrů biasu do sítě
<code>RandomInitialization</code>	False	Náhodná inicializace parametrů
<code>NetRegularization</code>	None	regularizace v metodě nejmenších čtverců lineárních parametrů
<code>FixedParameters</code>	None	stanovení některých parametrů a tudíž jejich vyloučení z trénování
<code>InitialRange</code>	1	rozsah jednotných pravděpodobnostní funkce v případě, že parametry jsou inicializovány náhodně
<code>OutputNonlinearity</code>	None	v případě, že výstupní neuron by měl být nelineární, doporučuje se <code>OutputNonlinearity-&gt;Sigmoid</code>
<code>Compiled</code>	True	použití compiled verze

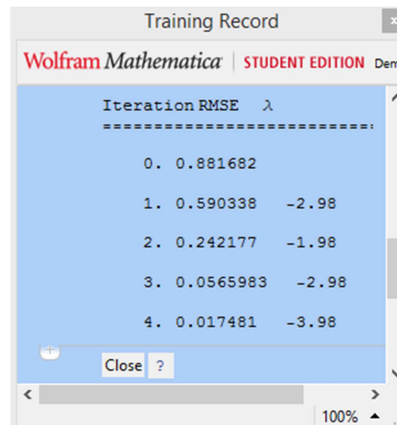
Dalším krokem je samotné trénování sítě příkazem:

```
NeuralFit[net, x, y, xv, yv, iterations],
```

kde `net` je trénovací model,  $x$  jsou vstupní data,  $y$  výstupní data,  $xv$  a  $yv$  předložená validační data a `iterations` je počet iterací.[12,22]

Během učení jsou průběžné výsledky zobrazeny v automaticky vytvořeném notebooku (Obr. 19). Po každém učení se zobrazí následující informace:

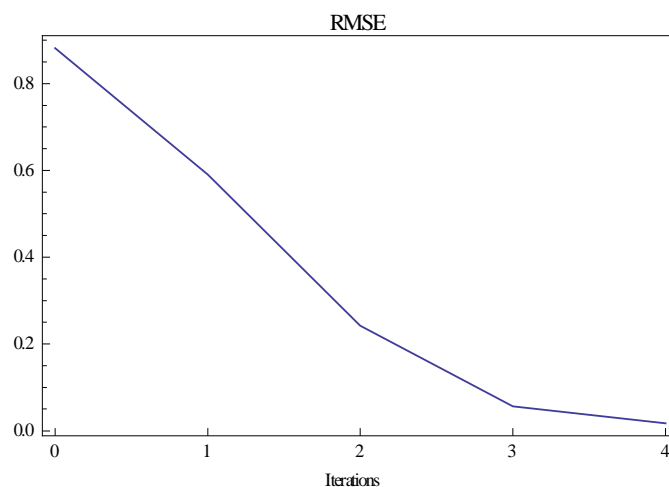
- Počet iterací učení a hodnota root-mean-square chyby RMSE.
- Pokud jsou k učení předloženy i validační data, pak se zobrazí také hodnota RMSE vypočítána z validace.
- Velikost kontrolního parametru minimalizační funkce.



Iteration	RMSE	$\lambda$
0.	0.881682	
1.	0.590338	-2.98
2.	0.242177	-1.98
3.	0.0565983	-2.98
4.	0.017481	-3.98

Obr. 19 – Zobrazení počtu iterací,  
RMSE a kontrolního parametru.

Na konci trénování je pokles RMSE zobrazen v grafu jako funkce počtu iterací. Často se křivka chyby RMSE ke konci učení zplošťuje, i když už je dosaženo přesného minima. Podle tohoto grafu pak můžeme usoudit, zda je potřeba přidat více iterací trénování. To docílíme opětovným předložením natrénované sítě (nebo jejím záznamem) funkci `NeuralFit` a nemusíme tak provádět trénování od začátku.[12,22]



Obr. 20 – Graf poklesu RMSE.

Další možnosti příkazu `NeuralFit`:

Tabulka 5 – Parametry příkazu `NeuralFit` včetně jejich významu.

parametr	Výchozí hodnota	význam
<code>Method</code>	Automatic	trénovací algoritmus
<code>Momentum</code>	0	momentum u učení Backpropagation
<code>StepLength</code>	Automatic	délka kroku u algoritmů Největšího spádu a Backpropagation
<code>FixedParameters</code>	None	parametry vyloučené z trénování
<code>Separable</code>	Automatic	použít oddělený algoritmus
<code>PrecisionGoal</code>	6	počet cifer v kritériu zastavení
<code>CriterionPlot</code>	Automatic	graf kritéria na konci trénování
<code>MinIterations</code>	3	minimální počet iterací trénování
<code>MoreTrainingPrompt</code>	False	vyzve k zadání více iterací, je-li nastaven na hodnotu True

Nejdůležitějším parametrem funkce `NeuralFit` je parametr `Method`, pomocí kterého si můžeme zvolit trénovací algoritmus. Balíček `NeuralNetworks` podporuje tyto algoritmy:

- Levenberg-Marquardt
- Gauss-Newton
- Steepest-descent (metoda největšího spádu)
- Backpropagation
- FindMinimum

Příklad trénování dopředné sítě s využitím algoritmu `Backpropagation` je znázorněn na obrázku (Obr. 21).[12,22]

```
In[19]:= << NeuralNetworks`
In[20]:= fdfwrdr = InitializeFeedForwardNet[{{1}}, {{1}}, {}, RandomInitialization -> True,
OutputNonlinearity -> Sigmoid];
In[21]:= {fdfwrdr2, fitrecord} = NeuralFit[fdfwrdr2, x, y, 200, Method -> BackPropagation,
StepLength -> 0.1, Momentum -> 0.9];
```

Obr. 21 – Příklad učení dopředné sítě s algoritmem `backpropagation`.

## **II. PRAKTICKÁ ČÁST**

## 4 PŘÍPRAVA DAT PRO UČENÍ NEURONOVÉ SÍTĚ

Cílem této skupinové práce je vývoj softwaru zabývající se rozpoznáváním vybraných dopravních značení. Tato diplomová práce se zabývá jádrem celého softwaru, neboli učením neuronových sítí, zkoumáním a testováním všech možných parametrů, jakými lze ovlivnit chování sítě a následné rozpoznávání kandidátů, přičemž veškeré trénovací a testovací množiny byly zpracovány kolegou zabývajícím se metodami zpracování obrazu [24]. Výstupem této diplomové práce byly vztahy z jednotlivých výstupních neuronů v závislosti na vstupních údajích, které byly dále předány ve formě textového dokumentu kolegovi zabývajícím se vývojem výsledné aplikace a kompletací všech tří částí [23].

Tato kapitola podrobně popisuje načtení vstupních dat a následně jejich zpracování v softwaru Mathematica tak, aby jejich použití při samotném učení neuronové sítě (využitím balíčku NeuralNetworks) bylo co nejefektivnější.

### 4.1 Import dat

Veškerá (obrazová) data, která jsem použil pro trénování, testování a validaci neuronové sítě, byla nejprve zpracována dle mých požadavků v diplomové práci kolegy [24].

Import dat do softwaru Mathematica byl proveden příkazem `ReadList`, který na rozdíl od příkazu `Import` automaticky uloží načtená data do listu (v tomto případě matice).

### 4.2 Příprava trénovací a testovací množiny

Trénovací množina se skládá z trénovacího vektoru (všechny prvky, které zastupují jednotlivé pixely obrázku tohoto vektoru, jsou vstupy každého vstupního neuronu sítě) a z vektoru požadovaného výstupu (v případě učení s učitelem).

Jedním z parametrů, které určují časovou náročnost učení sítě, je počet vstupů do neuronu. Jinak řečeno, čím větší bude rozměr prvků trénovacího vektoru (obrazových dat), tím déle se síť bude učit. Na druhou stranu, čím větší bude rozdíl mezi jednotlivými prvky trénovací množiny, tím přesněji bude síť naučena. Teoreticky největšího rozdílu lze docílit předložením všech pixelů obrázku na vstup, což má za následek časovou a výpočetní náročnost učení. Proto je třeba nalézt kompromis v podobě optimalizace obrazových dat tak, aby byl jejich počet jako vstup do neuronů dostatečný pro správné naučení sítě.

### 4.2.1 Úprava barevného režimu

Jedna z možností, jak optimalizovat počet obrazových dat, je převedením do jiného barevného režimu. V případě barevného modelu RGB je každý pixel zastoupen vektorem o třech prvcích  $\{r, g, b\}$ , kde  $r$  je hodnota červené,  $g$  hodnota zelené a  $b$  hodnota modré barvy v rozsahu 0 až 255. Převedením tohoto modelu do stupně šedi nebo binárního obrazu (Obr. 22) je každý pixel zastoupen pouze jednou hodnotou (stupeň šedi hodnota 0 až 1, binární obraz 0 nebo 1). Tímto způsobem snížíme počet vstupních dat na jednu třetinu.

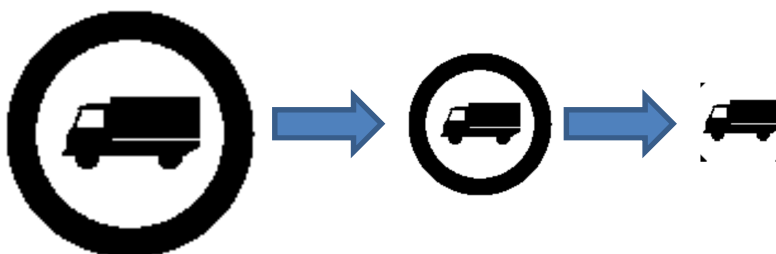


Obr. 22 – Převedení RGB modelu (vlevo) do stupně šedi (uprostřed) a binárního obrazu (vpravo).

Převedením obrázku do stupně šedi i binární podoby má své nevýhody. Např. značky typu „Zákaz stání“ a „Zákaz zastavení“ jsou reprezentovány kombinací červené a modré barvy, kdy převedením do stupně šedi či binární podoby dojde ke splynutí těchto dvou barev a tím i téměř ke ztrátě vzájemné odlišnosti.

### 4.2.2 Úprava rozměrů

Rozměry obrázku lze upravit dvěma způsoby – zmenšením (škálováním) nebo ořezáním na potřebné rozměry (těmito způsoby se podrobně zabýval kolega [24]).



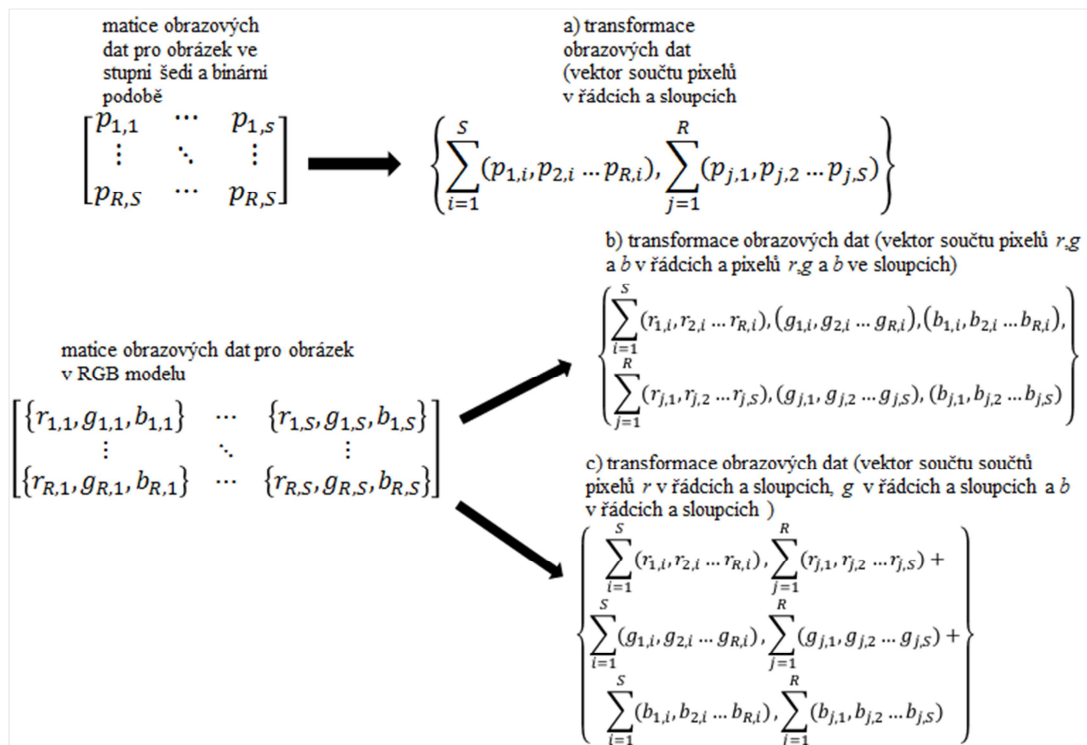
Obr. 23 – Zmenšení originálního obrázku a následné ořezání.

Zmenšením obrázku pomocí algoritmu bipolární interpolace dochází ke ztrátě ostrosti. Zmenšíme-li obrázek na 50% velikosti původního obrázku, je tato ztráta minimální a neovlivní tak učení sítě.

### 4.2.3 Transformace obrazových dat

Dalším způsobem, jak optimalizovat počet vstupních hodnot neuronů, je transformace obrazových dat metodou součtů jednotlivých řádků a sloupců.

Po načtení obrazových dat jsem pomocí vnořených cyklů v prostředí Mathematica sečetl jednotlivé hodnoty pixelů v každém řádku a sloupci matice obrazových dat a následně vykreslil do grafu tak, abych mohl zhodnotit, která z kombinací barevného modelu a typu transformace bude pro učení sítě nejvhodnější. Příklad různých typů transformací pro jednotlivé barevné modely jsou zobrazeny na následujícím obrázku (Obr. 24).

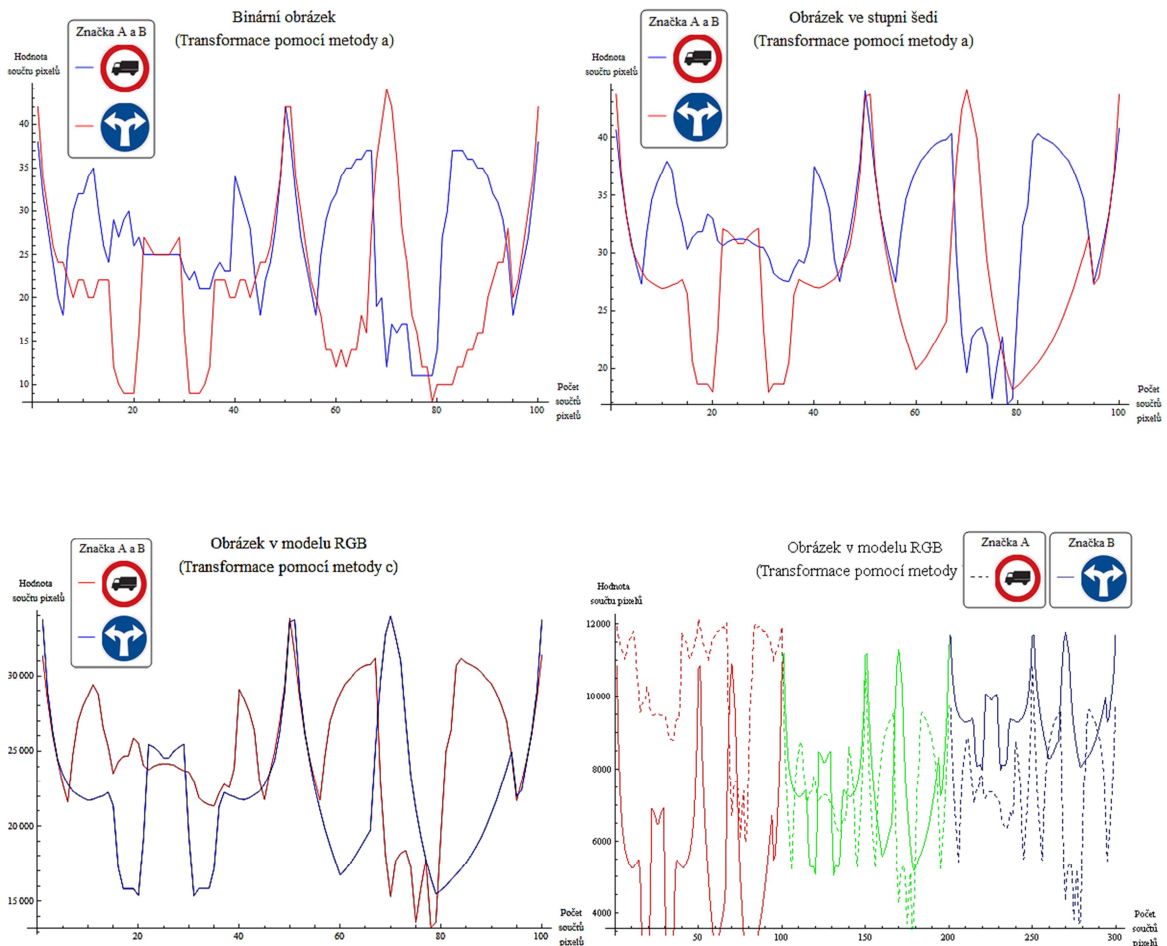


Obr. 24 – Metody transformací obrazových dat.

Typ transformace jsem volil vždy na základě rozměrů či charakteristických vlastností snímků trénovacího vektoru jednotlivých fází učení. Pro rychlé vyhodnocení nejvhodnější transformační metody jsem si vytvořil funkci `Grafy1[]`. Vstupem této funkce jsou dva obrázky o stejných rozměrech v barevném modelu RGB a výstupem grafy jednotlivých transformací vstupů v příslušném barevném provedení.

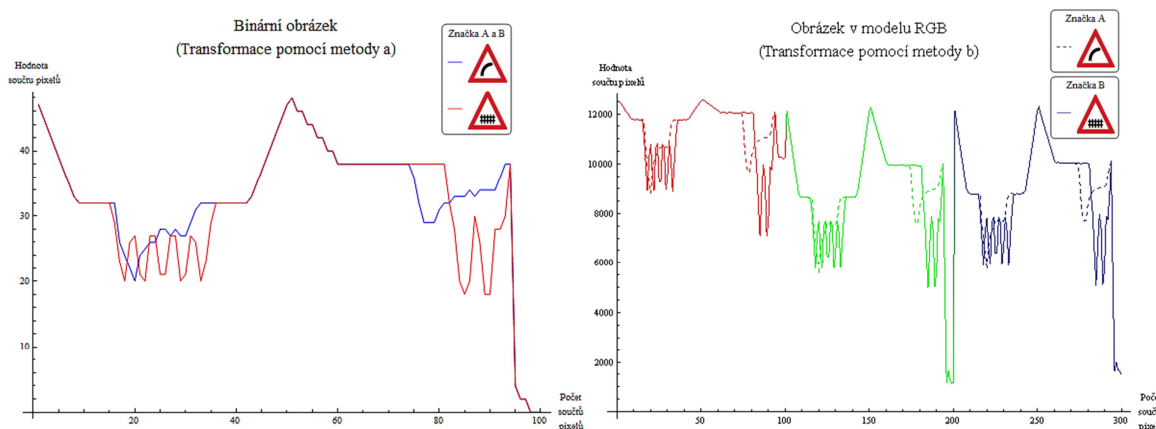
Jako příklad jsem přivedl na vstup funkce `Grafy1[]` dva konkrétní snímky příkazové a zákazové značky. Z grafů zobrazených na obrázku (Obr. 25) jsem jednoznačně

usoudil, že v případě učení dopravních značek, které jsou jasně odlišné svou barevností, použiji transformační metodu  $b$  v RGB modelu.



Obr. 25 – Grafy aplikovaných transformací na různé barevné modely.

V případě dvou podobných značek stejného typu jsou tyto metody transformace téměř nepoužitelné, jelikož při sčítání hodnot pixelů v řádcích a sloupcích logicky dojde ke zkreslení informace o poloze důležitých pixelů v obrázku (Obr. 26).



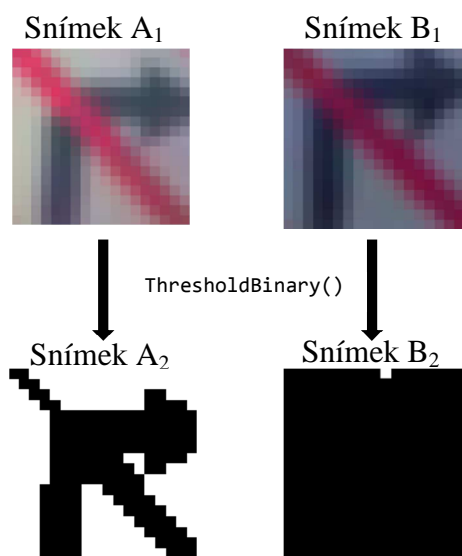
Obr. 26 - Grafy aplikovaných transformací na různé barevné modely.

#### 4.2.4 Normalizace jasu

Obecně existuje mnoho vnějších vlivů, které ovlivní jas dopravních značení (počasí, částí dne, natočení značky vůči slunci, ztráta sytosti barev, atd.). Na základě těchto skutečností jsem musel přijít na způsob, jakým lze normalizovat jas obrázků tak, aby byl pro každé dopravní značení přibližně stejný a tím zvýšit úspěšnost rozpoznávání.

Ve výsledné aplikaci na rozpoznávání dopravního značení v prostředí C# [23] je obrázek nejprve převeden pomocí knihovny Emgu CV do stupně šedi a následně do binárního obrazu pomocí metody `ThresholdBinary()`. Tato metoda obsahuje dva parametry – `thresh` a `maxValue`. Metoda je založena na porovnávání intenzity jednotlivých pixelů (je-li aktuální pixel větší než nastavená hodnota `thresh`, pak je hodnota intenzity nového pixelu nastavena na 0, v opačném případě na hodnotu `maxValue`).

Mějme dva snímky A a B s kandidáty na stejnou dopravní značku s odlišnou hodnotou jasu. Pomocí metody `ThresholdBinary()` jsem tyto snímky převedl do binární podoby a u obou nastavil stejné hodnoty parametrů `thresh: new Gray(120)` a `maxValue: new Gray(255)` (Obr. 27).

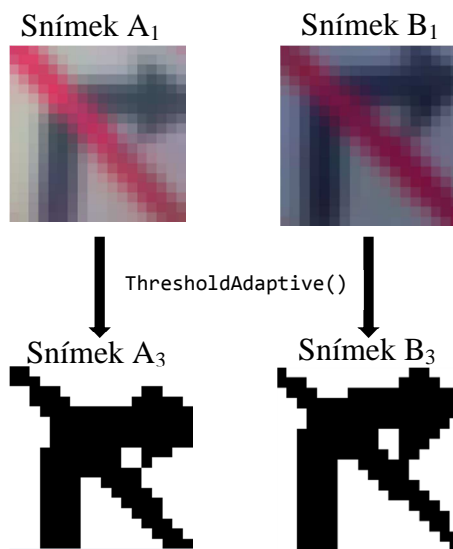


Obr. 27 – Převedení obrázku do binární podoby metodou *ThresholdBinary()*.

Pro snímek  $A_2$  je nastavení parametru `thresh` ideální. Naopak, u snímku  $B_2$  je po převedení do binárního obrazu hodnota téměř všech pixelů nastavena na 0. Předložením kandidáta na dopravní značení (obrazová data snímku  $B_2$ ) naučené neuronové síti dojde logicky ke špatnému rozpoznání.

Z toho vyplývá, že nevýhodou metody `ThresholdBinary()` je převádění snímku do binárního obrazu nezávisle na hodnotě jasu, jelikož je založena na principu globálního prahování (tzn. hodnota prahu je neměnná). Na základě těchto skutečností jsem zavedl alternativní metodu `ThresholdAdaptive()`, která je založena na počítání průměrné hodnoty jasu v malých regionech snímku. Parametry této funkce jsou `maxValue` (maximální hodnota prahu), `adaptiveType` (typ adaptivní metody), `thresholdType` (typ prahování), `blockSize` (velikost regionu sousedících pixelů) a `param1` (konstanta odečtená od průměru jasu regionu).

Tuto metodu jsem aplikoval na stejné snímky jako v předchozím případě a nastavil parametry `maxValue`: `new Gray(255)`, `adaptiveType`: `CV_ADAPTIVE_THRESH_MEAN_C`, `thresholdType`: `CV_THRESH_BINARY`, `blockSize`: 31 a `param1`: `new Gray(15)`. Poslední dva parametry byly zvoleny experimentálně. Výsledek převodů snímků je zobrazen na obrázku (Obr. 28).



Obr. 28 - Převedení obrázku do binární podoby metodou *ThresholdAdaptive()*.

Na rozdíl od předchozí metody `ThresholdBinary()`, funkce `ThresholdAdaptive()` je založena na principu lokálního prahování (hodnota prahu se dynamicky mění podle vypočítaného průměru jasu v jednotlivých regionech).

Další problém se vyskytl při rozpoznávání snímků dopravních značek „Zákaz stání“ a „Zákaz zastavení“, kde výstupem převodu do binárního obrazu pomocí poslední zmíněné metody byl pouze bílý snímek. To je způsobeno už převodem do stupně šedi, kdy červená i modrá barva je zastoupena podobným barevným odstínem.

Tento problém jsem vyřešil na základě myšlenky normalizací jasu snímků, separace jednotlivých barevných kanálů a vytvořením masky snímku pouze z červeného barevného kanálu.

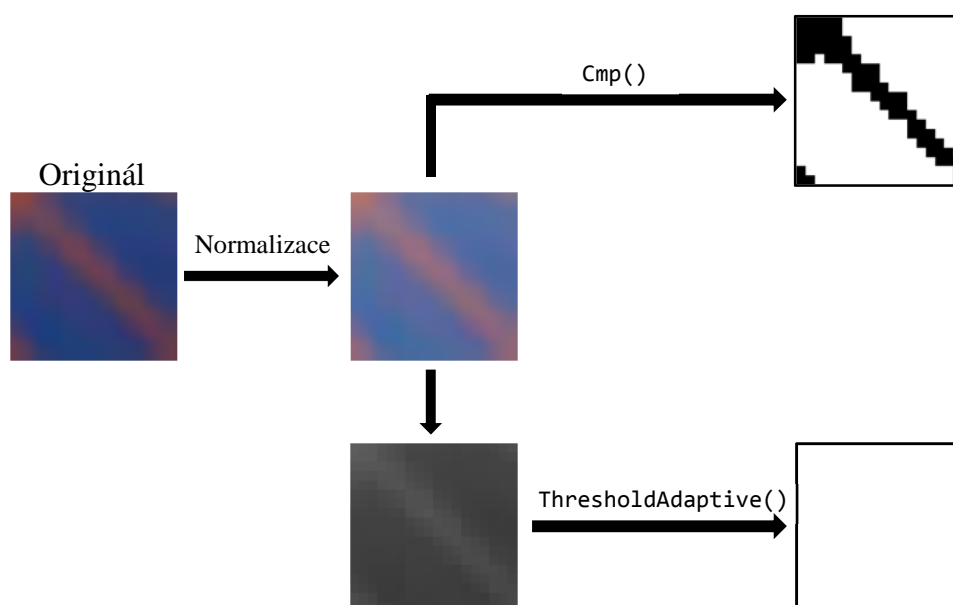
V prostředí C# jsem se doposud neseťkal s žádnou jednoduchou metodou či funkcí, která by jednoduše vypočítala průměrnou hodnotu jasu snímku. Proto jsem navrhl alternativní způsob zjištění této hodnoty založený na faktu, že pokud jsou na snímku zastoupeny pouze odstíny červené a modré barvy, pak odstín zelené barvy může teoreticky vyjadřovat právě hodnotu jasu.

Hodnotu odstínu zelené barvy jsem zjistil jednoduchým příkazem `Green`, který jsem aplikoval hned na několik pixelů a následně vypočítal jejich průměr, ze kterého jsem vycházel při úpravě jasu snímku příkazem `gammaCorection`. Cyklem `while` jsem

upravoval jas obrázku do té doby, než splňoval podmínku minimální a maximální hodnoty odstínu zelené barvy, kterou jsem si experimentálně zvolil, a tím tak normalizoval jas obrázku.

V dalším kroku jsem příkazem `split` separoval jednotlivé barevné kanály obrázku a metodou `Cmp()` (aplikovanou na třetí barevný kanál modelu BGR), která vytvoří masku původního snímku podle nastavených parametrů `value` (porovnávací hodnota) a `comparisonType` (typ porovnání). Aplikací metody `Cmp()` na několika desítkách normalizovaných snímků zmíněných typů dopravních značek jsem zjistil, že nastavením parametru `value=130` a `comparisonType: CV_CMP_GT` (porovnání „větší než“) dojde k optimálnímu vytvoření masky pouze z červeného odstínu barvy.

Na obrázku (Obr. 29) je znázorněn rozdíl mezi použitím funkce `ThresholdAdaptive()` a metody separace barevného kanálu a následné vytvoření jeho masky funkcí `Cmp()`.



Obr. 29 – Převedení snímku do binární podoby pomocí dvou odlišných metod.

Na obrázku (Obr. 29) je jasně pozorovatelné, že pro rozpoznání či odlišení dopravních značek „Zákaz stání“ a „Zákaz zastavení“ je metoda převedení obrazu do binární podoby nepoužitelná, ale i přesto má své uplatnění v jiném učení (viz. kapitola 5.1).

#### 4.2.5 Formát trénovací a testovací množiny

Vstupy neuronové sítě vyžadují určitou formu trénovací množiny, která se skládá z trénovacího vektoru a vektoru požadovaného výstupu, jak už bylo zmíněno výše. Proto jsme se s kolegou [24] dohodli na takovém formátu předávaných dat v podobě trénovacího vektoru, abych z něj mohl univerzálně vytvořit vektor požadovaného výstupu.

Importovaná data do softwaru Mathematica jsem si uložil do proměnné `dataImport`. Budeme-li zvyšovat počet indexů této proměnné v kombinaci s příkazem `Dimensions[]` a `Length[]`, zjistíme rozměry vnitřních vektorů, jejichž význam je popsán v tabulce (Tabulka 6).

Tabulka 6 – Rozměry vektoru importovaných dat.

Příkaz	hodnota	význam
<code>Dimensions[dataImport]</code>	{6}	Počet kategorií
<code>Length[#] &amp; /@ dataImport</code>	{2, 6, 2, 2, 2, 6}	Počet obrázků v jednotlivých kategoriích
<code>Length[dataImport[[1, 1]]]</code>	35	Počet pixelů v řádku
<code>Length[dataImport[[1, 1, 1]]]</code>	35	Počet pixelů ve sloupci
<code>dataImport[[1, 1, 1, 1]]</code>	{0., 0.286275, 0.568627}	Hodnota pixelu

Délka trénovacího vektoru by měla být rovna celkovému počtu obrázků v podobě obrazových dat v importovaném souboru. Proto jsem odstranil dělení do jednotlivých kategorií, které je pro učení sítě nepotřebné, příkazem `Flatten[]` (odstranění závorek ve vektoru). Dále jsem obrazová data představující vstup do neuronů sítě každého obrázku zpracoval buď opět odstraněním závorek (v případě předložení hodnot všech pixelů na vstup neuronů) nebo přepočtem podle zvolené metody transformace dat (kapitola 4.2.3). Tyto úpravy jsem provedl pomocí několika vnořených cyklů.

Délka vektoru požadovaného výstupu je rovna délce trénovacího vektoru s tím rozdílem, že jeho jednotlivé prvky neobsahují obrazová data obrázku, ale pouze množinu hodnot 1 a 0 (popř. -1), kde pozice hodnoty 1 určuje, do které kategorie daná obrazová data patří. Vektor požadovaného výstupu jsem vytvořil univerzálně na základě zjištěných informací o struktuře importovaného vektoru dat (Tabulka 6).

Testovací množina mi posloužila ke zjištění úspěšnosti rozpoznávání. Stejně jako trénovací množina se skládá ze dvou vektorů – testovací vektor a vektor požadovaného výstupu. Úprava testovacího vektoru byla provedena stejným způsobem jako u trénovacího vektoru. Hodnoty vektoru požadovaného výstupu byly zadány ručně. Do učení lze zahrnout i tzv. validační množinu, která slouží k testování chyby během učení.

## 5 UČENÍ NEURONOVÉ SÍTĚ

V oblasti klasifikace a rozpoznávání vzorů lze využít hned několika typů neuronových sítí. V této diplomové práci jsem si zvolil vícevrstvou dopřednou neuronovou síť jako řešení problematiky rozpoznávání dopravních značek, která je praxí ověřená jako spolehlivý základní klasifikační mechanismus a navíc nabízí velké množství parametrů, které vhodným nastavením výrazně ovlivní spolehlivost učení sítě.

Nejprve jsem se musel zamyslet nad tím, jak se od sebe jednotlivá dopravní značení odlišují, nalézt jejich hlavní charakteristickou vlastnost a na základě těchto faktů rozložit celkové učení do několika fází (Příloha PI).

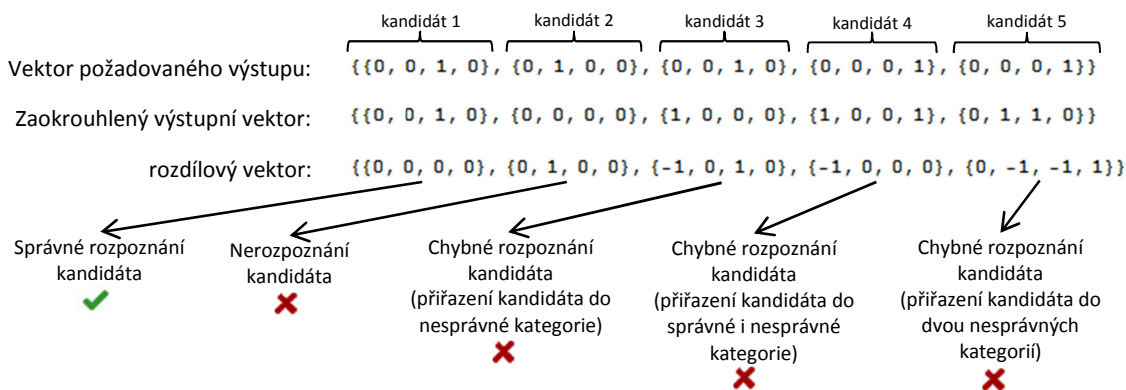
Každé učení proběhlo předložením trénovací množiny na vstup a nastavením parametrů sítě. Po učícím procesu jsem na vstup naučené sítě opět předložil trénovací vektor a tím získal hodnoty jednotlivých výstupních neuronů (výstupní vektor), jejichž interval je dán přenosovou funkcí. Výsledný vektor jsem následně zaokrouhlil a odečetl od vektoru požadovaného výstupu, čímž jsem zjistil úspěšnost naučení sítě.

Stejným postupem jsem zjistil i úspěšnost rozpoznávání kandidátů s tím rozdílem, že naučené síti jsem předložil testovací vektor, který se skládá z obrazových dat snímků dopravních značek, se kterými se naučená neuronová síť doposud „nesetkala“. Po odečtení zaokrouhleného výstupního vektoru od vektoru požadovaného výstupu (rozdílový vektor) může nastat několik situací (uvažujme pouze testování jednoho kandidáta):

- Správné rozpoznání – přiřazení kandidáta do správné kategorie (hodnoty rozdílového vektoru jsou nulové).
- Nerozpoznání – nepřičtení kandidáta do žádné kategorie (rozdílový vektor je roven vektoru požadovaného výstupu).
- Chybné rozpoznání – přiřazení kandidáta do nesprávné kategorie nebo do dvou a více kategorií.

Na základě těchto situací jsem v jednotlivých fázích učení vypočítal celkovou úspěšnost rozpoznávání ze správně a chybně rozpoznávaných kandidátů (nerozpoznané kandidáty jsem ignoroval).

Na následujícím obrázku (Obr. 30) je uveden případ, kdy nastaly všechny tři situace.



Obr. 30 – Příklady správného a chybného rozpoznání kandidáta.

Na závěr každé fáze učení jsem po úspěšném naučení sítě vytvořil jako formát textu pole proměnných  $x[0]$  až  $x[i]$ , kde  $i$  je počet (transformovaných) obrazových dat snížen o 1 z důvodu odlišné indexace v prostředí C#, které začíná od 0. Toto pole jsem předložil naučené síti a tím získal vztahy z jednotlivých výstupních neuronů v závislosti na vstupních údajích. Příkazem `Export` jsem výsledné vztahy uložil do textového dokumentu a předal kolegovi [23], který je implementoval do výsledné aplikace.

Každá fáze učení neuronové sítě (v příloze PI označená čísly 1 až 9) proběhla v softwaru Mathematica pomocí balíčku `NeuralNetworks`, se kterým jsem se v první řadě musel seznámit. Součástí tohoto balíčku je jednoduchá nápověda, ze které jsem čerpal, obsahující informace o neuronových sítích, učících algoritmech a podrobné příklady použití jednotlivých příkazů včetně možných nastavení jejich parametrů.

Výsledná softwarová aplikace vytvořená kolegou [23] je založena na hledání kandidátů dopravních značek v každém snímku videozáznamu. Vzhledem k tomu, že nalezení kandidáti vždy vyjadřují vzájemné odlišnosti (i v případě, že se jedná o kandidáty na stejnou dopravní značku), ještě před učením jsme s kolegy prozkoumali všechna možná řešení, jakými lze tyto odlišnosti potlačit a zvýraznit pouze hlavní rysy, kterými jsou jednotlivé dopravní značení charakteristické (kapitola 4.2). Jednotlivé fáze učení byly rozděleny do tří bloků (červený, modrý a žlutý), které reprezentují zbarvení dopravních značek.

## 5.1 Rozpoznávání červených dopravních značení

Stejně jako u ostatních dílčích učení jsem zvolil vícevrstvou dopřednou neuronovou síť. Tato fáze učení mi posloužila hlavně jako zkoušení všech možností, jakými lze síť naučit s co největší úspěšností rozpoznávání. Kromě různých úprav vstupních obrázků dopravních značek jsem se snažil využít i veškerých učících algoritmů, přenosových funkcí a dalších možností, které balíček NeuralNetworks nabízí.

V prvním řadě jsem testoval jednotlivé algoritmy učení a na základě globální chyby a úspěšnosti rozpoznávání zvolil, který z nich bude nejvhodnější pro řešení této problematiky. Jako vstupní data jsem použil obrázky dopravních značek v barevném modelu RGB o rozměrech 50x50 pixelů s využitím metody transformace dat  $b$  (Obr. 24). U každého učení jsem použil přenosovou funkci logistické sigmoidy (Obr. 11) a další parametry (počet neuronů skryté vrstvy, počet iterací, popř. délka kroku a momentum) jsem nastavil experimentálně tak, aby výsledná globální chyba byla co nejnižší.

Tabulka 7 – Výsledné hodnoty testování učících algoritmů.

Učící algoritmus	Počet skrytých neuronů	Počet iterací	Délka kroku	Momentum	Globální chyba (RMSE)	Úspěšnost rozpoznávání [%]
<b>Gauss-Newton</b>	30	30	x	x	$4,234 \cdot 10^{-10}$	81
<b>Levenberg-Marquardt</b>	3	188	x	x	$2,9 \cdot 10^{-17}$	89
<b>největšího spádu</b>	30	60	x	x	0,325	x
<b>Backpropagation</b>	4	50	0,1	0,9	0,397	x

Srovnáme-li výsledné hodnoty z tabulky (Tabulka 7), zjistíme, že nejnižších globálních chyb (rozdíl mezi nimi je zanedbatelný) dosáhlo učení Gauss-Newtonovým (dále jen GN) a Levenberg-Marquardtovým (dále jen LM) algoritmem. Úspěšnost rozpoznávání jsem zjistil na základě testování přímo ve výsledné aplikaci z celkového počtu 169 rozpoznávaných značek. U ostatních algoritmů jsem nedokázal nastavit jejich parametry do takové míry, aby globální chyba byla dostatečná pro úspěšné naučení sítě (proto jsem je ani nezapojil do testování). Na neuronové síti se stejnými vstupy a využitím LM algoritmu jsem vyzkoušel dostupné přenosové funkce, abych zjistil, která dokáže celkovou úspěšnost rozpoznávání zvýšit.

Tabulka 8 – Výsledné hodnoty testování přenosových funkcí.

Přenosová funkce	Počet skrytých neuronů	Počet iterací	Globální chyba (RMSE)	Úspěšnost rozpoznávání [%]
<b>Logistická sigmoida</b>	4	100	0,0014	100
<b>Hyperbolický tangens</b>	6	40	$1,48 \cdot 10^{-6}$	85,7
<b>Lineární saturaovaná</b>	7	40	$5,93 \cdot 10^{-16}$	57,1

Touto analýzou jsem potvrdil fakt, že přenosová funkce logistická sigmoida je nejvhodnější pro řešení problematiky rozpoznávání vzorů. Jak je vidět v tabulce (Tabulka 8), neuronová síť byla pomocí této přenosové funkce naučena už při čtyřech neuronech, což přináší výhodu pro výslednou aplikaci, jelikož na čím méně neuronů ve skryté vrstvě je neuronová síť naučena, tím je výpočetní náročnost výstupu nižší. Úspěšnost rozpoznávání byla zjištěna předložením testovací množiny na vstup naučené sítě. V případě, kdy jsem měl přebytek trénovacích dat (snímků), jsem do učení zahrnul validaci sítě prostřednictvím validační množiny (menší část trénovací množiny jsem použil k validaci sítě). Po testování úspěšnosti rozpoznávání na nejlépe naučené síti s využitím validace jsem zjistil, že došlo ke stejným výsledkům, jako když byla validační množina součástí trénovací množiny.

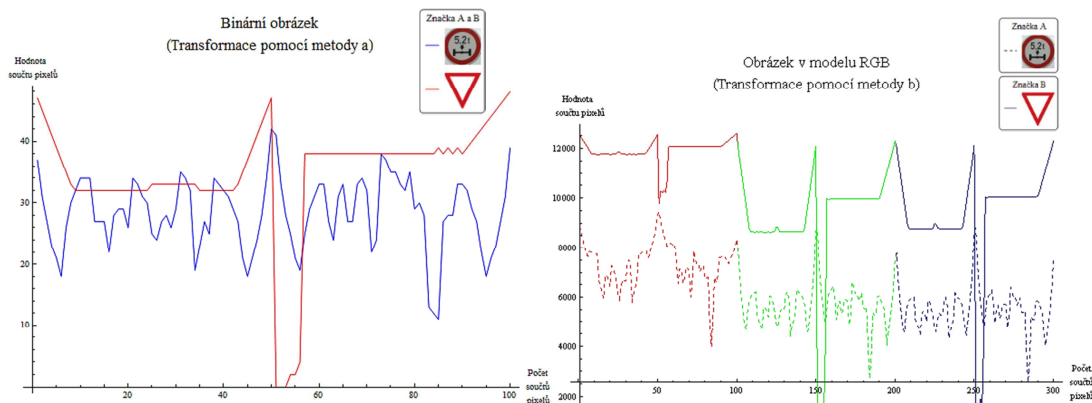
Dalším důležitým prvkem, který ve velké míře ovlivní učení sítě i následné rozpoznávání kandidátů, je barevný režim a rozměr obrázků. Tento prvek byl v každé fázi učení volen individuálně. Po konzultaci s kolegy jsme se dohodli na základním rozměru obrázku 50x50 pixelů, který je optimální jak z hlediska počtu obrazových dat, které představují vstup do každého neuronu sítě, tak i hlediska dostatečného zdůraznění vzájemných odlišností mezi různými typy dopravního značení. Z tohoto rozměru se vycházelo při další úpravě rozměrů (škálováním) či ořezání obrázku.

**První fáze** učení sítě (v příloze PI označená číslem 1) a rozeznávání červených dopravních značek byla založena na vzájemných odlišnostech, nikoli na významu dopravních značek a zároveň rozdělení do dalších skupin či rozpoznání konkrétního typu značky. Porovnáním dopravních značek jednotlivých kategorií (Obr. 31), do kterých by měly být neuronovou sítí následně přiřazeny, jsem usoudil, že největší odlišností těchto obrázků dopravních značek není vnitřní piktogram, ale tvar neboli plocha pixelů zastupujících odstín červené barvy.



Obr. 31 – Zástupci dopravních značek každé kategorie.

Před samotným učením sítě jsem si pomocí vytvořené funkce `Grafy1 [ ]` zjistil, který z barevných režimů vstupních obrázků (popř. metody transformace dat) bude teoreticky nejvhodnější pro učení sítě.



Obr. 32 - Grafy aplikovaných transformací na barevný model RGB a stupeň šedi.

V grafech na obrázku (Obr. 32) je viditelné, že většího rozdílu obrazových dat obrázků dvou odlišných typů dopravních značek dosáhla metoda transformace dat *b* barevného modelu RGB. Z těchto důvodů jsem do učení nezahrnul metodu transformace dat *a* obrázků ve stupni šedi. Učící algoritmus a přenosovou funkci jsem volil v této fázi učení na základě předchozích analýz a rozměry snímků ponechal ve výchozím nastavení 50x50 pixelů. Počet neuronů ve výstupní vrstvě je roven počtu kategorií, do kterých mají být snímky kandidátů dopravních značek přiřazeny (v tomto případě čtyři výstupní neurony). Další parametry, jako je počet skrytých neuronů a počet iterací jsem nastavoval tak, aby globální chyba učení byla co nejnižší. V následující tabulce jsou vypsány informace o nastavení neuronové sítě v jednotlivých učeních (snímky, ze kterých je složena trénovací a testovací množina, jsou uvedeny v diplomové práci kolegy [24]).

Tabulka 9 – Nastavené parametry sítě v jednotlivých krocích učení.

číslo učení	1	2	3	4	5	6	7	8
<b>Barevný režim</b>	RGB	RGB	RGB	RGB	RGB	RGB	RGB	RGB
<b>Metoda transformace dat</b>	c	c	c	c	b	b	b	
<b>Počet vstupních dat (neuronů)</b>	100	100	100	100	300	300	300	7500
<b>Počet neuronů ve skryté vrstvě</b>	4	5	6	20	4	5	8	4
<b>Počet iterací</b>	40	100	100	80	100	30	35	350
<b>Globální chyba (RMSE)</b>	0,0024	0,111011	0,175013	0,108038	0,00143	0,179136	0,269479	0,0001
<b>Úspěšnost rozpoznávání [%]</b>	92,5	92,5	88,2	100	100	76,5	64,3	84,2

Nejdeálnější nastavení sítě a formát vstupní množiny pro první fázi učení jsem volil na základě úspěšnosti rozpoznávání v poměru k počtu neuronů ve skryté vrstvě (čím méně neuronů, tím je výpočetní náročnost rozpoznávání kandidátů menší). V následujícím obrázku (Obr. 33) je uvedena inicializace neuronové sítě s nastavením ideálních parametrů podle tabulky (Tabulka 9) a zavedení sítě do fáze učení.

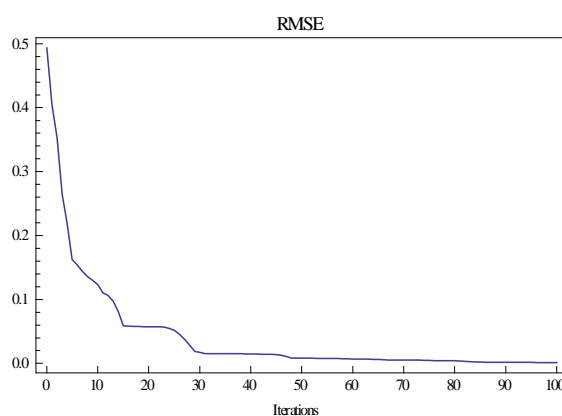
```

sit = InitializeFeedForwardNet[x, y, {4}, Neuron -> Sigmoid, OutputNonlinearity -> Sigmoid]
FeedForwardNet[{{w1, w2}}, {Neuron -> Sigmoid, FixedParameters -> None,
  AccumulatedIterations -> 0, CreationDate -> {2014, 5, 22, 4, 16, 42.7022770},
  OutputNonlinearity -> Sigmoid, NumberOfInputs -> 300}]
{sit2, fitrecord} = NeuralFit[sit, x, y, 100];

```

Obr. 33 – Inicializace a příprava první fáze učení dopředné neuronové sítě.

Doba učení sítě je závislá na počtu neuronů ve všech vrstvách sítě a počtu iterací. Učení bylo automaticky ukončeno buď dosažením maximálního počtu iterací, nebo pokud bylo dosaženo minimální globální chyby (globální chyba dále neklesá). Po ukončení učení byl automaticky vygenerován graf globální chyby (RMSE) v jednotlivých krocích učení zobrazen na obrázku (Obr. 34).



Obr. 34 – Graf průběhu klesání globální chyby (RMSE).

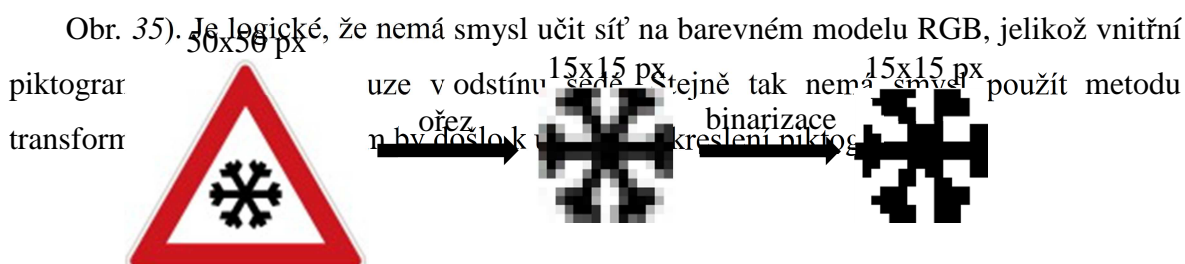
V následující tabulce (Tabulka 10) jsou uvedeny veškeré informace o první fázi výsledného učení včetně jeho úspěšnosti rozpoznávání.

Tabulka 10 – Podrobné informace o první fázi učení.

Neuronová síť	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
Přenosová funkce	Logistická sigmoida	
Počet iterací učení	100	
Označení	300-4-4	
Vstupní vrstva	Počet pixelů v řádcích	50
	Počet pixelů ve sloupcích	50
	Barevný režim	RGB
	Metoda transformace dat	b
Skrytá vrstva	Počet neuronů	300
	Počet skrytých vrstev	1
Výstupní vrstva	Počet neuronů	4
	Počet neuronů	4
Úspěšnost rozpoznávání	Celkový počet testovaných kandidátů	169
	Počet rozpoznávaných kandidátů	120
	Počet nerozpoznaných kandidátů	42
	Počet špatně rozpoznávaných kandidátů	7
	Úspěšnost rozpoznání kandidátů [%]	94,4

Výstupem první fáze učení jsou čtyři kategorie (čtyři výstupní neurony naučené dopředné sítě), kde dvě z nich jsou konkrétní dopravní značky „Dej přednost v jízdě“ a „Stůj, dej přednost v jízdě“, jelikož mají jedinečný tvar. Zbylé dvě kategorie jsou skupiny výstražných a zákazových dopravních značek. Obě tyto skupiny se skládají z konkrétních typů značek se stejným tvarem. Na základě tohoto faktu jsem zavedl dílčí fáze učení.

**Druhá fáze** učení (v příloze PI označená číslem 2) je založena na vzájemných odlišnostech vnitřních piktogramů dopravních značek, jelikož celý tvar dopravní značky je vždy stejný. Na základě této myšlenky bylo nejprve provedeno zmenšení snímku na základní rozměry 50x50 pixelů a následně ořezání, takové, aby výsledný ořez zachytil pouze oblast, na které se vyskytuje piktogram dopravní značky, což odpovídá ořezu o rozměru 15x15 pixelů. Detailní popis ořezání snímku je uveden v diplomové práci kolegy [24]. Po úpravě rozměrů jsem si zvolil typ barevného režimu snímku v binární podobě, na kterém jsem následně učil neuronovou síť (



Obr. 35 – Ořezání a binarizace snímku.

V této fázi učení jsem opět zvolil již otestovanou přenosovou funkci logistická sigmoida a na základě předchozí analýzy učících algoritmů (Tabulka 7) jsem učil síť GN i LM algoritmem. Úspěšnost rozpoznávání jsem určil na základě testování dvaceti dvou kandidátů, kde počet nerozpoznaných kandidátů nebyl do tohoto výpočtu zahrnut.

Tabulka 11 – Druhá fáze učení LM algoritmem.

krok učení	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
Počet neuronů ve skryté vrstvě	3	7	9	11	12	13	15	18	19	30
Počet iterací	100	100	100	50	30	50	30	30	30	20

Globální chyba (RMSE)	$5.9 \cdot 10^{-13}$	$4.5 \cdot 10^{-17}$	$2.1 \cdot 10^{-17}$	$9.2 \cdot 10^{-12}$	0,0001	$2.2 \cdot 10^{-7}$	$4.3 \cdot 10^{-7}$	0,001	0,00003	$3.8 \cdot 10^{-9}$
Nerozpoznaných kandidátů	0	2	1	0	3	1	3	4	4	1
Úspěšnost rozpoznávání [%]	45,5	85	85,7	81,8	89,5	90,5	84,2	100	100	90,5

Tabulka 12 – Učení GN algoritmem

krok učení	1.	2.	3.	4.	5.
Počet neuronů ve skryté vrstvě	12	16	17	19	20
Globální chyba (RMSE)	0,502	0,0004	$1 \cdot 10^{-7}$	$1,2 \cdot 10^{-7}$	$5,3 \cdot 10^{-10}$
Nerozpoznaných kandidátů	0	1	2	1	1
Úspěšnost rozpoznávání [%]	0	90	90	85,7	81

Stejně jako v předchozí fázi učení se i zde osvědčil LM učící algoritmus, který měl s osmnácti a devatenácti neurony ve skryté vrstvě úspěšnost rozpoznávání stoprocentní (Tabulka 11). Inicializace a zavedení sítě s nastavenými ideálními parametry do druhé fáze učení jsou zobrazeny na následujícím obrázku (Obr. 36).

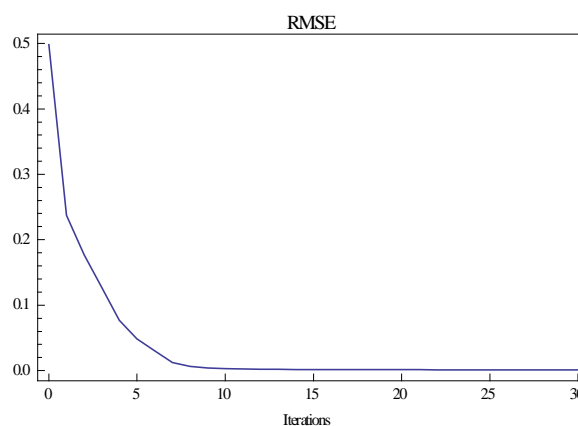
```

sit = InitializeFeedForwardNet[x, y, {18}, Neuron → Sigmoid, OutputNonlinearity → Sigmoid]
FeedForwardNet[{{w1, w2}}, {Neuron → Sigmoid, FixedParameters → None,
  AccumulatedIterations → 0, CreationDate → {2014, 5, 22, 12, 53, 13.8544043},
  OutputNonlinearity → Sigmoid, NumberOfInputs → 225}]
AbsoluteTiming[{{sit2, fitrecord} = NeuralFit[sit, x, y, 30]}

```

Obr. 36 - Inicializace a příprava druhé fáze učení dopředné neuronové sítě.

Příkazem `AbsoluteTiming` jsem si vypsál dobu učení sítě v sekundách. V tomto případě druhá fáze učení trvala 1920 sekund (32 minut).



Obr. 37 - Graf průběhu klesání globální chyby (RMSE).

Na obrázku (Obr. 37) je patrné, že neuronová síť byla úspěšně naučena už při desáté iteraci. I když globální chyba dosáhla téměř nulové hodnoty a úspěšnost rozpoznávání

v softwaru Mathematica byla stoprocentní, úspěšnost rozpoznávání této sítě ve výsledné aplikaci na pořízených videosekvencích byla velice nízká (viz. Tabulka 13).

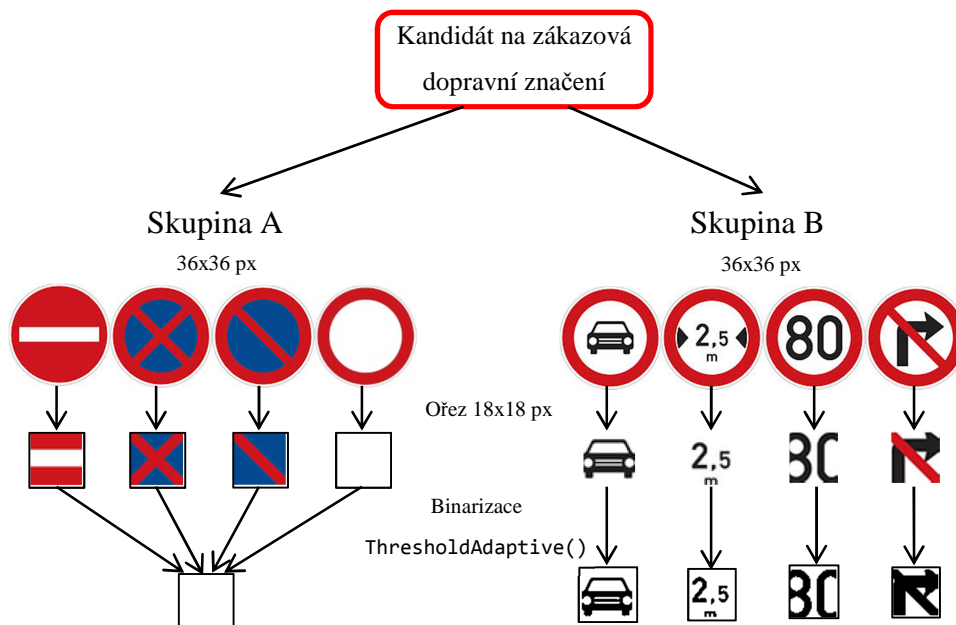
Tabulka 13 - Podrobné informace o druhé fázi učení.

Neuronová síť	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
Přenosová funkce	Logistická sigmoida	
Počet iterací učení	30	
Označení	225-18-10	
Vstupní vrstva	Počet pixelů v řádcích	15
	Počet pixelů ve sloupcích	15
	Barevný režim	Binární obraz
	Metoda transformace dat	x
	Počet neuronů	225
Skrytá vrstva	Počet skrytých vrstev	1
	Počet neuronů	18
Výstupní vrstva	Počet neuronů	10
Úspěšnost rozpoznávání	Celkový počet testovaných kandidátů	46
	Počet rozpoznávaných kandidátů	16
	Počet nerozpoznaných kandidátů	17
	Počet špatně rozpoznávaných kandidátů	13
	Úspěšnost rozpoznání kandidátů [%]	55

V každé tabulce s podrobnými informacemi jednotlivých fází jsou uvedeny procenta úspěšnosti rozpoznávání, které byly vypočítány pouze ze správně rozeznávaných kandidátů (nerozeznání kandidátů nejsou do výpočtu úspěšnosti zahrnuti).

V této fázi učení je počet výstupních neuronů roven 10, tedy 10 kategorií, do kterých jsou kandidáti na výstražné dopravní značení zařazeni. Tyto kategorie jsou tvořeny jednou konkrétní dopravní značkou, nebo skupinou dopravních značení, jejichž význam je podobný.

Cílem **třetí fáze** učení (v příloze PI označená číslem 3) je rozpoznávání zákazových dopravních značení, které jsem rozdělil do dvou skupin (Obr. 38). Skupina A je reprezentována značkami, které jsou charakteristické vnitřním zbarvením, zatímco skupina B představuje značky, které jsou od sebe odlišné vnitřním piktogramem. Ještě před samotným učením sítě jsem zjistil, jakou budou mít podobu snímky kandidátů těchto skupin po úpravě rozměrů a následném převedení do binárního obrazu funkcí `ThresholdAdaptive()`.



Obr. 38 – Rozdělení, ořez a binarizace snímků.

Jak je vidět na obrázku (Obr. 38), výsledkem převodů výřezů snímků zákazových dopravních značek skupiny A do binární podoby funkcí `ThresholdAdaptive()` v prostředí C# je vždy bílý snímek, kterým byla tato skupina reprezentována v trénovacím vektoru. Jinými slovy jsem na základě nedokonalostí funkce `ThresholdAdaptive()` zjednodušil učení sítě a zvýšil úspěšnost rozpoznávání. Tímto způsobem jsme s kolegou zabývajícím se zpracováním obrazu [24] připravil trénovací a testovací množinu.

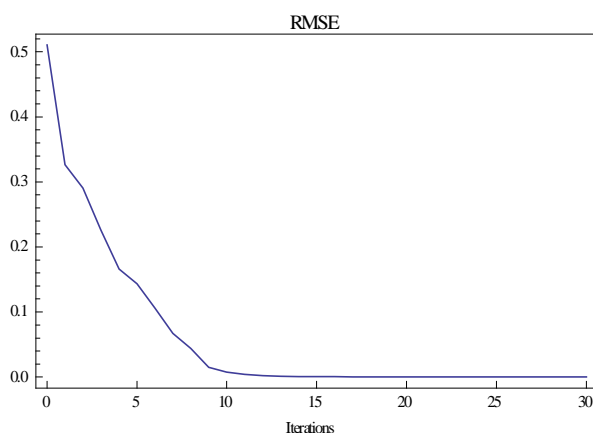
Jako přenosovou funkci jsem zvolil opět logistickou sigmoidu a učící algoritmus LM. Nastavováním počtu skrytých neuronů a dostatečného počtu iterací jsem dopřednou neuronovou síť opakovaně učil, dokud nebyla úspěšnost rozpoznávání kandidátů testovací množiny maximální (Tabulka 14).

Tabulka 14 - Třetí fáze učení LM algoritmem.

krok učení	1.	2.	3.	4.	5.	6.
Počet neuronů ve skryté vrstvě	7	8	10	13	16	17
Počet iterací	100	40	40	100	12	10
Globální chyba (RMSE)	$3.9 \cdot 10^{-22}$	0,07	$1.6 \cdot 10^{-8}$	$9.7 \cdot 10^{-12}$	0,04	0,001
Nerozpoznaných kandidátů	1	4	2	2	1	4
Úspěšnost rozpoznávání [%]	72,7	94,7	85,7	95,2	91	94,2

Stoprocentní úspěšnost rozpoznávání nastala hned u několika počtů skrytých neuronů. V této situaci jsem z důvodu výpočetní náročnosti zvolil opět nejnižší počet. Inicializace a zavedení sítě do fáze učení proběhla stejně jako v předchozích případech (Obr. 36), pouze

s jiným počtem skrytých neuronů a počtem iterací. Po úspěšném naučení byl opět automaticky vygenerován graf průběhu klesání globální chyby (Obr. 39).



Obr. 39 – Hodnota globální chyby (RMSE) v každém kroku učení.

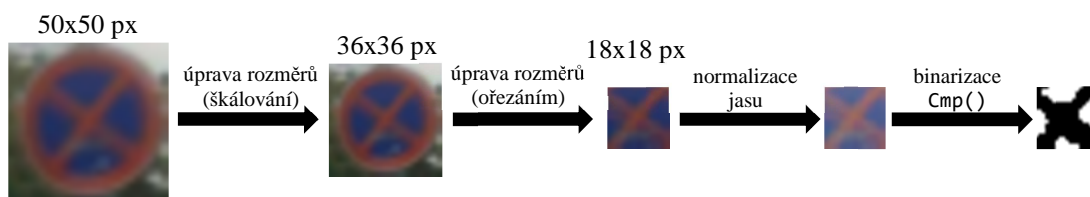
Výsledná globální chyba dosáhla velmi nízké hodnoty, což odpovídá maximální úspěšnosti naučení dopředné sítě. Úspěšnost rozpoznávání ve výše uvedené tabulce (Tabulka 12) mi sloužila pouze k okamžitému zjištění, zda je potřeba dále měnit počet neuronů ve skryté vrstvě a opakovat tak učení sítě. Reálná úspěšnost rozpoznávání byla zjištěna až ve výsledné aplikaci na pořizovaných videosekvencích (Tabulka 15).

Tabulka 15 - Podrobné informace o třetí fázi učení.

<b>Neuronová síť</b>	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
	Přenosová funkce	Logistická sigmoida
	Počet iterací učení	100
	Označení	324-13-7
<b>Vstupní vrstva</b>	Počet pixelů v řádcích	18
	Počet pixelů ve sloupcích	18
	Barevný režim	Binární obraz
	Metoda transformace dat	x
	Počet neuronů	324
<b>Skrytá vrstva</b>	Počet skrytých vrstev	1
	Počet neuronů	13
<b>Výstupní vrstva</b>	Počet neuronů	7
<b>Úspěšnost rozpoznávání</b>	Celkový počet testovaných kandidátů	91
	Počet rozpoznávaných kandidátů	67
	Počet nerozpoznaných kandidátů	16
	Počet špatně rozpoznávaných kandidátů	8
	Úspěšnost rozpoznání kandidátů [%]	89,3

Po domluvě s kolegy jsme se společně shodli, že dopravní značky „Zákaz stání“, „Zákaz zastavení“ a „Zákaz vjezdu“ patří mezi nejdůležitější a nejfrekventovanější dopravní značení v ČR a na základě tohoto faktu jsem se snažil tyto dopravní značení rozpoznávat s co největší přesností zavedením **čtvrté fáze** učení (v příloze PI označená číslem 4).

V předchozí fázi učení bylo použito pro převod snímku do binární podoby příkazu `ThresholdAdaptive()`, díky kterému neuronová síť jednoznačně zařadila naposledy zmíněné konkrétní dopravní značky do jedné kategorie. Čtvrtá fáze se zabývá přesným rozpoznáním těchto značek využitím normalizace jasu a následného převedení do binární podoby metodou separace jednotlivých barevných kanálů (tyto metody byly podrobně popsány v kapitole 4.2.4).



Obr. 40 – Úprava rozměrů, normalizace jasu a binarizace snímku.

Tato metoda (Obr. 40) dokázala dobře zvýraznit charakteristické rysy jednotlivých značek, avšak v některých případech na úkor nepatrného zvýšení časové a výpočetní náročnosti. Na základě velkých odlišností mezi značkami této kategorie nebylo náročné nalézt optimální parametry neuronové sítě takové, aby byla úspěšnost rozpoznávání jednotlivých kandidátů maximální. Z tohoto důvodu zde uvádím pouze tabulku s informacemi o výsledné dopředné síti čtvrté fáze učení (Tabulka 16).

Tabulka 16 - Podrobné informace o čtvrté fázi učení.

Neuronová síť	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
	Přenosová funkce	Logistická sigmoida
	Počet iterací učení	50
	Označení	324-3-3
Vstupní vrstva	Počet pixelů v řádcích	18
	Počet pixelů ve sloupcích	18
	Barevný režim	Binární obraz
	Metoda transformace dat	x
	Počet neuronů	324
Skrytá vrstva	Počet skrytých vrstev	1
	Počet neuronů	3
Výstupní vrstva	Počet neuronů	3
Úspěšnost rozpoznávání	Celkový počet testovaných kandidátů	62
	Počet rozpoznávaných kandidátů	43
	Počet nerozpoznaných kandidátů	11
	Počet špatně rozpoznávaných kandidátů	8
	Úspěšnost rozpoznání kandidátů [%]	84,3

Zákazové dopravní značky skupiny B (Obr. 38), jejichž vzájemná odlišnost je zastoupena vnitřním piktogramem, jsou také rozděleny do několika skupin. Se souhlasem kolegů jsem ty nejméně významné dopravní značky zařadil do kategorie „zákazové

ostatní“. Důležitější značky podobného významu jsem sjednotil do jednotlivých skupin a pro ty nejvýznamnější jsem zavedl dílčí fáze učení.

**Pátá a šestá fáze** učení (v příloze PI označené čísla 5 a 6) se zabývá právě rozpoznáním konkrétních typů dopravních značek z kategorie „Omezení rychlosti, váhy a rozměrů“ a „Zákaz odbočení, otáčení“. U obou fází byla trénovací množina tvořena snímky o rozměrech 18x18 pixelů, převedených funkcí `ThresholdAdaptive()` do binární podoby. Vzhledem k tomu, že rozdíly mezi jednotlivými piktogramy této skupiny dopravních značek jsou minimální, bylo zapotřebí klást větší důraz na učení neuronové sítě. V následujících tabulkách (Tabulka 17, Tabulka 18) jsou uvedeny jednotlivé kroky učení dopředné neuronové sítě s přenosovou funkcí logistická sigmoida a LM i GN učícím algoritmem.

Tabulka 17 - Pátá fáze učení LM algoritmem.

krok učení	1.	2.	3.	4.	5.	6.	7.	8.	9.
Počet neuronů ve skryté vrstvě	6	7	8	9	10	11	12	13	14
Počet iterací	50	50	50	50	50	50	50	50	20
Globální chyba (RMSE)	$2.2 \cdot 10^{-9}$	$4.4 \cdot 10^{-9}$	$1.1 \cdot 10^{-8}$	$1.2 \cdot 10^{-8}$	$2.1 \cdot 10^{-6}$	$6.5 \cdot 10^{-12}$	$1.7 \cdot 10^{-11}$	$3.4 \cdot 10^{-7}$	0,0009
Nerozpoznaných kandidátů	1	3	1	1	0	3	1	0	0
Úspěšnost rozpoznávání [%]	90	90,5	95	90	94,4	94,4	95	95,2	90,5

Tabulka 18 - Šestá fáze učení LM a GN algoritmem.

krok učení	1.	2.	3.	4.	5.	6.	7.	8.
Učící algoritmus	LM	LM	LM	LM	GN	GN	GN	GN
Počet neuronů ve skryté vrstvě	4	6	8	10	4	6	7	8
Počet iterací	20	50	50	20	30	30	30	30
Globální chyba (RMSE)	$1.8 \cdot 10^{-7}$	$5.3 \cdot 10^{-20}$	$1.1 \cdot 10^{-14}$	$9.5 \cdot 10^{-9}$	0,5	$2.8 \cdot 10^{-8}$	$8.2 \cdot 10^{-13}$	$9.1 \cdot 10^{-14}$
Nerozpoznaných kandidátů	0	1	0	0	0	1	1	0
Úspěšnost rozpoznávání [%]	87,5	100	100	100	12,5	87,5	100	100

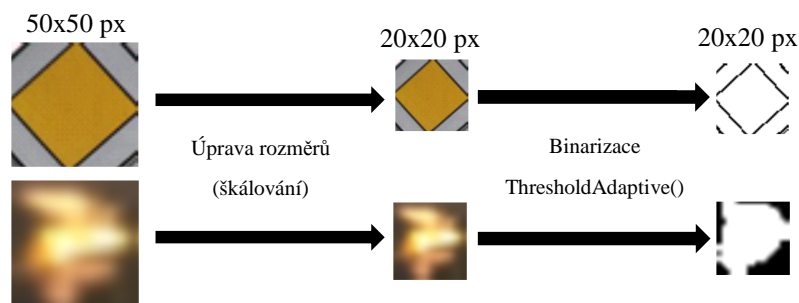
V těchto fázích učení jsem se snažil naučit dopřednou síť i na větším počtu skrytých neuronů. Jelikož výpočetní náročnost učení sítě je přímo úměrná počtu neuronů ve skryté vrstvě, docházelo často k ukončení učení chybovou hláškou `Failed - No memory available`.

Tabulka 19 - Podrobné informace o páté a šesté fázi učení.

Fáze učení	5.	6.
Počet iterací učení	50	50
Označení sítě	324-13-11	324-8-3
Rozměry trénovacích snímků	18x18 px	18x18 px
Barevný režim	Binární obraz	Binární obraz
Počet vstupních neuronů	324	324
Počet skrytých neuronů	13	8
Počet výstupních neuronů	11	3
Počet testovaných kandidátů	17	26
Počet rozpoznaných kandidátů	10	18
Počet nerozpoznaných kandidátů	4	4
Počet špatně rozpoznaných kandidátů	3	4
Úspěšnost rozpoznání kandidátů [%]	80	82

## 5.2 Rozpoznávání žlutých dopravních značení

Do kategorie dopravních značek s odstínem žluté spadá pouze jedna značka – „Hlavní pozemní komunikace“. V tomto případě by měla neuronová síť pouze jeden výstupní neuron. Ovšem jak bylo zjištěno, algoritmy vyhledávající kandidáty na žlutá dopravní značení velmi často označovaly za dopravní značku nežádoucí šum (listy stromů, části budov, atd.). Z tohoto důvodu jsem do učení zahrnul další kategorii, která je tvořena právě těmito nežádoucími šumy.



Obr. 41 – Úprava rozměrů a binarizace snímku.

V této **sedmé fázi** učení (v příloze PI označená číslem 7) byly snímky s kandidáty na žlutou dopravní značku nejprve zmenšeny (škálováním) na rozměry 20x20 a následně převedeny do binárního obrazu (Obr. 41). Jako učící algoritmus jsem zde vyzkoušel LM i GN algoritmus.

Tabulka 20 - Sedmá fáze učení LM a GN algoritmem.

krok učení	1.	2.	3.	4.	5.	7.	8.
<b>Učící algoritmus</b>	LM	LM	LM	LM	GN	GN	GN
<b>Počet skrytých neuronů</b>	4	7	12,8	14	4	14	20
<b>Počet iterací</b>	30	30	30	30	30	30	30
<b>Globální chyba (RMSE)</b>	$2.8 \cdot 10^{-6}$	0.006	$9.8 \cdot 10^{-6}$	$4.2 \cdot 10^{-11}$	0,3	$8.3 \cdot 10^{-14}$	$1.2 \cdot 10^{-13}$
<b>Počet nerozpoznaných kandidátů</b>	0	1	0	1	0	0	1
<b>Úspěšnost rozpoznávání [%]</b>	71,1	84,2	78,9	73	71	81,6	83,8

Kandidáti, na kterých byla testována úspěšnost rozpoznávání, byli získáni výslednou aplikací. Jak je vidět v tabulce (Tabulka 20), úspěšnost nebyla nikdy stoprocentní. To je zaviněno šumem, který má po převodu do binárního obrazu často stejný tvar jako dopravní značka „Hlavní pozemní komunikace“. Tomuto problému jsem však nedokázal čelit ani „posílením“ rozpoznávání podmínkami pro pravoúhlost, podélnost, kruhovost a úhlu natočení kontury (nejspecifičtější příznak pro tento typ dopravní značky), kterými se zabýval kolega [23]. V následující tabulce jsou uvedeny informace o zvolené ideální dopředné síti včetně úspěšnosti rozpoznávání.

Tabulka 21 - Podrobné informace o sedmé fázi učení.

Neuronová síť	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
Přenosová funkce	Logistická sigmoida	
Počet iterací učení	30	
Označení	400-7-2	
Vstupní vrstva	Počet pixelů v řádcích	20
	Počet pixelů ve sloupcích	20
	Barevný režim	Binární obraz
	Metoda transformace dat	x
Skrytá vrstva	Počet neuronů	400
	Počet skrytých vrstev	1
Výstupní vrstva	Počet neuronů	7
	Počet neuronů	2
Úspěšnost rozpoznávání	Celkový počet testovaných kandidátů	58
	Počet rozpoznávaných kandidátů	28
	Počet nerozpoznaných kandidátů	12
	Počet špatně rozpoznávaných kandidátů	18
	Úspěšnost rozpoznání kandidátů [%]	61

### 5.3 Rozpoznávání modrých dopravních značení

Dopravní značky v odstínu modré (příkazová dopravní značení) jsou charakteristické svým tvarem (kruhem) a jejich jedinou vzájemnou odlišností je pouze vnitřní bílý piktogram. Úprava rozměrů snímků byla provedena tak, aby byl zachycen pouze vnitřní piktogram, který je u této kategorie větší než u předešlých zákazových a výstražných skupin. Ořezání bylo realizováno z výchozího rozměru 50x50 pixelů na 34x34 pixelů.

Barevný režim byl použit binární obraz. Pokud je dopravní značka charakteristická pouze jednou či dvěma barvami, považuji učení sítě na obrázcích v RGB režimu za zbytečnost. Učení sítě by v tomto případě dosáhlo teoreticky podobných výsledků, ale s třikrát vyšším počtem vstupních neuronů, což má za následek vyšší výpočetní a časovou náročnost učení. Tato **osmá fáze** učení je v příloze (příloha PI) označena číslem 8.

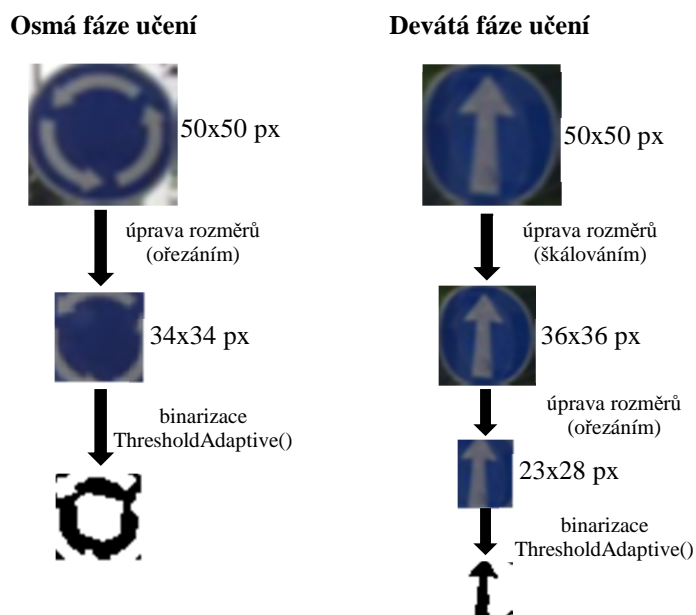
Dopředná síť byla učena pomocí LM učícího algoritmu a přenosové funkce logistická sigmoida. Učení proběhlo ve třech krocích, kdy jsem postupně nastavoval počet skrytých neuronů 3, 5 a 7. Maximální úspěšnosti nakonec dosáhlo učení s 5 neurony ve skryté vrstvě. Následující tabulka popisuje podrobné informace o nastavení parametrů, počtu neuronů v jednotlivých vrstvách a úspěšnosti rozpoznávání (*Tabulka 22*).

*Tabulka 22 - Podrobné informace o osmé fázi učení.*

<b>Neuronová síť</b>	Typ	Dopředná síť (FeedForward)
	Učící algoritmus	Levenberg-Marquardt
	Přenosová funkce	Logistická sigmoida
	Počet iterací učení	25
	Označení	1156-5-4
<b>Vstupní vrstva</b>	Počet pixelů v řádcích	34
	Počet pixelů ve sloupcích	34
	Barevný režim	Binární obraz
	Metoda transformace dat	x
	Počet neuronů	1156
<b>Skrytá vrstva</b>	Počet skrytých vrstev	1
	Počet neuronů	5
<b>Výstupní vrstva</b>	Počet neuronů	4
<b>Úspěšnost rozpoznávání</b>	Celkový počet testovaných kandidátů	68
	Počet rozpoznávaných kandidátů	47
	Počet nerozpoznaných kandidátů	12
	Počet špatně rozpoznávaných kandidátů	9
	Úspěšnost rozpoznání kandidátů [%]	84

Tato fáze učení dělí příkazové dopravní značky do pěti kategorií, kde tři jsou reprezentovány jednou konkrétní značkou („Kruhový objezd“, „Zimní výbava“ a „Příkazovaný jízdní pruh“), čtvrtá kategorie je tvořena několika méně důležitými značkami, které nejsou dále rozpoznávány, a poslední a nejdůležitější kategorie je tvořena dopravními značkami „Příkazovaný směr jízdy“. Po konzultaci s kolegy jsme se dohodli, že je zapotřebí zavést dílčí **devátou fázi** učení (v příloze PI označena číslem 9), která bude rozpoznávat každou dopravní značku poslední zmíněné kategorie.

I když se zdá být zbytečné zavádět dílčí učení pro dopravní značky typu „Příkazovaný směr jízdy“, opak je pravdou. Kromě odlišných rozměrů snímků bylo zapotřebí nastavit i odlišné parametry funkce ThresholdAdaptive(), aby byl vnitřní piktogram správně převeden do binárního obrazu (Obr. 42).



Obr. 42 – Úprava snímků v osmé a deváté fázi učení.

Učení sítě proběhlo pouze s osmi neurony ve skryté vrstvě, kdy rozpoznávání jednotlivých kandidátů dosáhlo maximální úspěšnosti (Tabulka 23).

Tabulka 23 - Podrobné informace o deváté fázi učení.

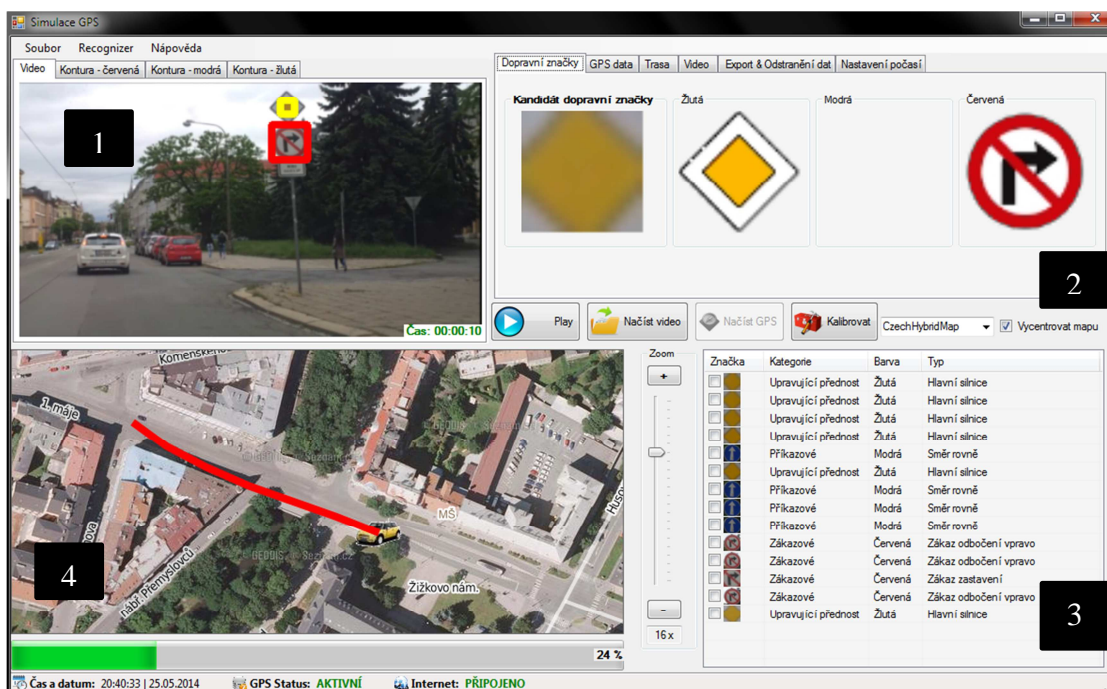
Neuronová síť	Typ	Dopředná síť (FeedForward)
		Učící algoritmus
	Přenosová funkce	Logistická sigmoida
	Počet iterací učení	30
	Označení	644-8-11
Vstupní vrstva	Počet pixelů v řádcích	23
	Počet pixelů ve sloupcích	28
	Barevný režim	Binární obraz
	Metoda transformace dat	x
Skrytá vrstva	Počet neuronů	644
	Počet skrytých vrstev	1
Výstupní vrstva	Počet neuronů	8
	Počet neuronů	11
Úspěšnost rozpoznávání	Celkový počet testovaných kandidátů	53
	Počet rozpoznávaných kandidátů	40
	Počet nerozpoznaných kandidátů	7
	Počet špatně rozpoznávaných kandidátů	6
	Úspěšnost rozpoznání kandidátů [%]	87

K celkovému rozpoznání vybraných dopravních značení bylo použito devět neuronových sítí, na kterých jsem otestoval všechna dostupná nastavení parametrů, tak, aby úspěšnost rozpoznávání byla co nejvyšší. Během procesu učení jsem také zkoumal chování sítě na základě změny parametrů či úpravy vstupních dat (trénovací množiny). Na začátku této kapitoly jsem prostřednictvím testování různých přenosových funkcí označil logistickou sigmoidu jako nejvhodnější pro řešení problematiky této diplomové práce.

Stejně tak jsem testoval i dostupné učící algoritmy. Zde jsem bohužel nedokázal u algoritmu Backpropagation a metody největšího spádu nastavit parametry tak, aby výsledná úspěšnost rozpoznávání byla dostačující. Proto byly v každé fázi učení použity pouze algoritmy LM a GN a podrobné informace o učení sítě byly uvedeny do tabulek.

## 6 POPIS UŽIVATELSKÉHO ROZHRAŇÍ VÝSLEDNÉ APLIKACE

Hlavní okno výsledné aplikace (Obr. 43), jejímž vývojem se zabýval kolega [43], je tvořeno čtyřmi základními částmi.



Obr. 43 – Hlavní okno výsledné aplikace.

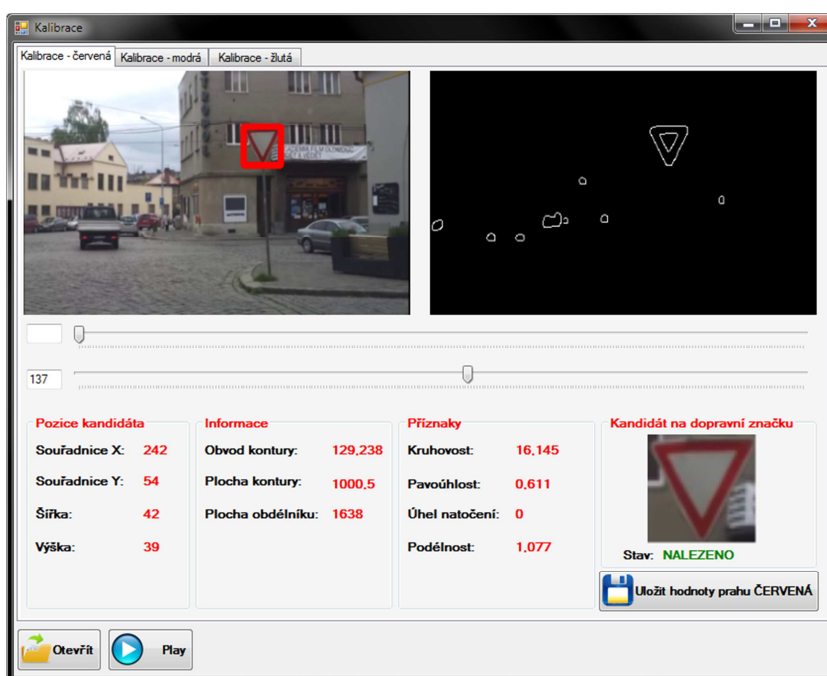
První část (1) slouží k zobrazení přehrávaného videa a upozornění na nalezenou a rozpoznanou dopravní značku prostřednictvím opsaného obdélníků, jehož barva odpovídá barvě dopravního značení. Součástí jsou také záložky, díky kterým má uživatel možnost zobrazit si nalezené kontury pro tři odstíny barev (červená, modrá a žlutá).

Druhá část (2) se skládá ze čtyř ImageBoxů, přičemž do prvního je vždy zobrazen originál nalezené značky z videozáznamu a do zbylých tří jsou podle jejich zbarvení zobrazeny dopravní značky z databáze této aplikace, které odpovídají nalezenému originálu. Pomocí záložek si může uživatel zobrazit informace o GPS souřadnicích, trase a videozáznamu, ale také exportovat nalezené originální snímky či vyčistit list snímků nalezených dopravních značení. Poslední a velmi důležitá záložka slouží k nastavení parametrů vyhledávání kontur pro jednotlivé scénáře.

Třetí část (3) hlavního okna vzestupně vypisuje nalezené dopravní značky v pořadí, v jakém byly nalezeny ve videozáznamu, kde v prvním sloupci jsou zobrazeny originální snímky a v dalších informace o nalezených dopravních značkách (kategorie, barevný odstín a konkrétní typ).

Čtvrtá část (4) vykresluje aktuální polohu řidiče včetně zaznačení ujeté trasy. Kromě možností výběru z 59 druhů map, které jsou součástí knihovny GMaps.NET, má uživatel možnost s těmito mapami různým způsobem manipulovat (posouvat, přibližovat, vycentrovat, atd.).

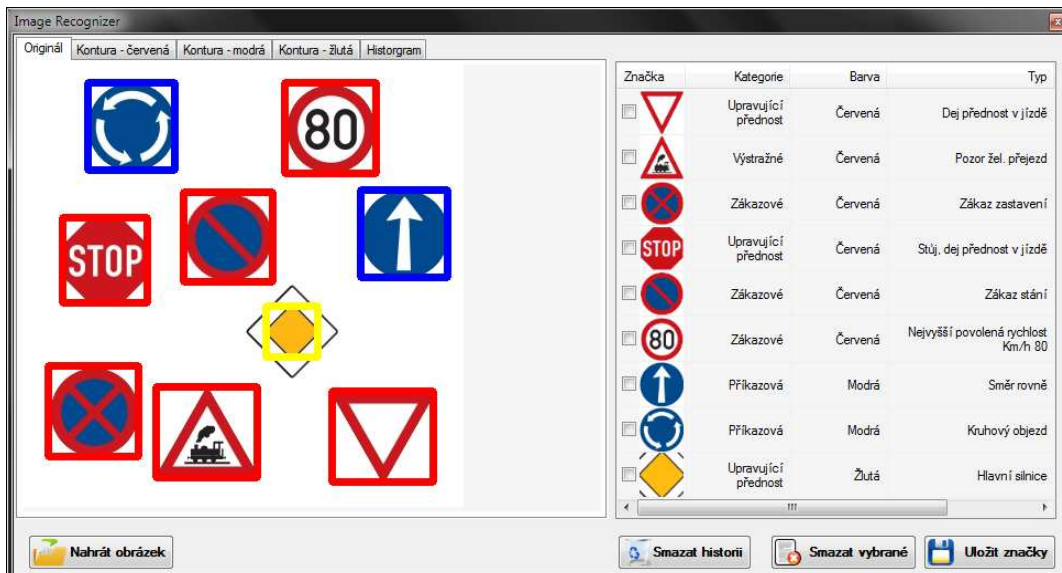
Tlačítkem „Načíst video“ je uživatel ihned vyzván k výběru videozáznamu v mnoha formátech. Po načtení videa je znovu uživatel vyzván tentokrát k výběru GPS dat. Po úspěšném načtení (i v případě, že nedošlo k výběru GPS dat) se automaticky spustí přehrávání vybraného videozáznamu. Tlačítkem „Play/Pause“, jak už název napovídá, lze videozáznam pozastavit nebo opět spustit. Tlačítkem „kalibrovat“ se automaticky zobrazí nové okno (Obr. 44).



Obr. 44 – Okno kalibrace obrazu.

Okno kalibrace obrazu slouží k nastavování prahových hodnot jednotlivých kanálů barevného modelu LAB. V horní části tohoto okna je synchronizovaně přehráván načtený videozáznam, přičemž v pravé části je přehráván v binárním obrazu s nalezenými konturami zvoleného odstínu barvy prostřednictvím záložek. Pomocí posuvníků lze nastavovat prahové hodnoty kanálů. Ve spodní části okna kalibrace jsou zobrazeny podrobné informace o nalezeném kandidátovi na dopravní značení včetně jeho výřezu z videozáznamu. Tlačítkem „Uložit hodnoty prahu ČERVENÁ“ dojde k uložení prahových hodnot do hlavního programu.

Další důležitou součástí programu je okno ImageRecognizer (Obr. 45), které je jádrem celé aplikace, sloužící k rozpoznávání dopravních značek z obrázků. V levé části je zobrazen načtený snímek s dopravní značkou nebo značkami a stejně jako v hlavním okně, nalezený kandidát je označen opsaným obdélníkem. V levé části jsou vypsaní nalezení kandidáti včetně originálního výřezu a podrobných informací o dané dopravní značce.



Obr. 45 – Okno ImageRecognizer.

## ZÁVĚR

Cílem této diplomové práce byla na základě vstupních dat, získaných od Bc. Josefa hrubého, vymodelovat takovou neuronovou síť, která by dokázala rozpoznávat s co největší úspěšností jednotlivé typy dopravních značení, a předání výsledných závislostí jednotlivých výstupních neuronů na vstupních údajích kolegovi Bc. Janu Dospivovi.

První myšlenka modelu sítě byla jednoduchá – předložit na vstup všechny hodnoty pixelů snímku s kandidátem dopravní značky na vstup naučené neuronové sítě a ihned jej přiřadit ke konkrétnímu typu dopravní značky. Tato myšlenka byla ovšem při pokusu o naučení sítě ihned zavrhnuta, z důsledku velkého rozměru vstupních dat (což mělo za následek velkou výpočetní a časovou náročnost učení) a nevhodně navržené topologii neuronové sítě. Z tohoto důvodu jsem zavedl učení do jednotlivých dílčích učení, neboli z jedné neuronové sítě jsem jich vytvořil hned několik, kde každá se zabývá rozpoznáváním kategorií, skupin či konkrétních značek na základě charakteristických vlastností jejich tvaru. Po rozvržení jednotlivých fází jsem se pokusil zpracovat vstupní data (trénovací a testovací množinu) do takové podoby, aby počet pixelů jednotlivých snímků byl co nejmenší a přitom zůstal zachován, popř. ještě více zvýrazněn jeho charakteristická odlišnost od ostatních snímků dopravních značek. Nejprve jsem otestoval, která z přenosových funkcí a učících algoritmů bude nejvhodnější pro řešení problematiky rozpoznávání vzorů a následně jsem učil síť na různých počtech neuronů. Z mé strany byla i snaha naučit neuronovou síť na co největším i nejmenším počtu neuronů ve skryté vrstvě. Avšak z důvodu vysoké výpočetní náročnosti softwaru Mathematica v případě velkého počtu skrytých neuronů docházelo k zastavení procesu učení z důsledku zaplnění operační paměti. I přesto jsem dokázal dosáhnout poměrně velké úspěšnosti rozpoznávání 79,6%, které jsem vypočítal jako průměr dílčích úspěšností rozpoznávání jednotlivých fází učení.

## ZÁVĚR V ANGLIČTINĚ

The aim of this thesis was based on the input data obtained from Bc Josef hrubý , to model a neural network which is able to identify with the greatest success of the different types of traffic signs, and transfer the resulting dependence of the output of individual neurons to input data to a colleague Bc Jan Dospiva.

The first idea of the network model was simple - to submit all input values of pixels in the image with the candidate traffic signs to enter the learned neural network and immediately assign it to a specific type of traffic signs. This idea , however, was when trying to learn the network immediately repudiated , due to the large size of the input data ( resulting in high computational and time consuming learning) and improperly designed topology of neural networks. For this reason, I introduced learning to individual learning sub, or from one neural network I've created several, each dealing with recognition categories, brands or specific groups based on characteristics of their shape. After the layout of each stage, I tried to process the input data ( training and test sets) into a form that the number of pixels of each frame as small as possible and still remained, eventually . further highlighted its characteristic difference from other images of road signs. First, I have tested that the transfer functions and learning algorithms will be most appropriate to address the issue of pattern recognition and then I taught the net on different numbers of neurons. For my part was an effort to teach a neural network to the largest and smallest number of neurons in the hidden layer. However, due to the high computational cost of software Mathematica if a large number of hidden neurons there to stop the learning process as a result of full memory. Still, I managed to achieve relatively high success rate of 79.6 % recognition, which I calculated as the average of the success of the recognition of individual learning phase.

**SEZNAM POUŽITÉ LITERATURY**

- [1] Revize TP 65 Zásady pro dopravní značení na pozemních komunikacích. 2013.
- [2] ŠÍMA, J. *Teoretické otázky neuronových sítí*. Praha, 1996, 390 s. ISBN 80-858-6318-9.
- [3] Úvod do neuronových sítí. *Automa: Časopis pro automatizační techniku* [online]. 2013 [cit. 2014-05-23]. Dostupné z:[http://www.odbornecasopisy.cz/index.php?id\\_document=30255](http://www.odbornecasopisy.cz/index.php?id_document=30255)
- [4] ZELINKA, Ivan. *Umělá inteligence, aneb, Úvod do neuronových sítí, evolučních algoritmů--*. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2005. 127 s. Učební texty vysokých škol (Univerzita Tomáše Bati ve Zlíně. Technologická fakulta). ISBN 80-731-8277-7. Skripta. UTB Zlín.
- [5] VINKLER, Bc. Tomáš. *Animace činnosti umělé neuronové sítě v prostředí Mathematica*. Zlín, 2012. Bakalářská. Univerzita Tomáše Bati. Vedoucí práce Ing. Zuzana Oplatková, Ph.D.
- [6] Neuronové sítě – vývoj a testování. *Shrimphood* [online]. 2014 [cit. 2014-05-12]. Dostupné z: <http://www.shrimphood.net/neuronove-site-vyvoj-a-testovani.html>
- [7] Topologie umělých neuronových sítí. *Pepiino.cz* [online]. 2010 [cit. 2014-05-13]. Dostupné z: [http://sofe2.pepiino.cz/wiki/doku.php?id=topologie\\_umelych\\_neuronovych\\_siti](http://sofe2.pepiino.cz/wiki/doku.php?id=topologie_umelych_neuronovych_siti)
- [8] BÍLA J.: *Umělá inteligence a neuronové sítě v aplikacích*, ČVUT, 1996, ISBN 80-01-01275-1.
- [9] VONDRÁK, Ivo. *Umělá inteligence a neuronové sítě*. Ostrava: VŠB, 1995, 139 s. ISBN 80-707-8259-5.
- [10] Neuronové sítě: Úvod do neuronových sítí. [Http://cyber.felk.cvut.cz/](http://cyber.felk.cvut.cz/) [online]. 2009 [cit. 2014-05-14]. Dostupné z:[http://cyber.felk.cvut.cz/gerstner/biolab/bio\\_web/teach/FunBio/neuron/neursite.html](http://cyber.felk.cvut.cz/gerstner/biolab/bio_web/teach/FunBio/neuron/neursite.html)
- [11] Úvod do problematiky umělých neuronových sítí. *Elektrorevue.cz* [online]. 2008 [cit. 2014-05-18]. Dostupné z:<http://www.elektrorevue.cz/clanky/00013/index.html>

- [12] Wolfram Mathematica Documentation Center. Wolfram Mathematica Documentation Center [online]. 2014 [cit. 2014-05-22]. Dostupné z: <http://reference.wolfram.com/mathematica/guide/Mathematica.html> Wolfram Mathematica—Wolfram Mathematica 9 Documentation reference.wolfram.com
- [13] Feed-Forward Networks. *Biochem.ucl.ac.uk* [online]. 2009 [cit. 2014-05-14]. Dostupné z: [http://www.biochem.ucl.ac.uk/~shepherd/sspred\\_tutorial/ss-nn.html](http://www.biochem.ucl.ac.uk/~shepherd/sspred_tutorial/ss-nn.html)
- [14] NIELSEN, Michael A. How the backpropagation algorithm works. *Http://neuralnetworksanddeeplearning.com* [online]. 2014 [cit. 2014-05-26]. Dostupné z: <http://neuralnetworksanddeeplearning.com/chap2.html>
- [15] Linear Neural Networks. *Willamette University* [online]. 2010 [cit. 2014-05-15]. Dostupné z: <http://www.willamette.edu/~gorr/classes/cs449/linear2.html>
- [16] Neural Network Learning by the Levenberg-Marquardt Algorithm with Bayesian Regularization. */cesarsouza/blog: Science, computing and machine learning*. [online]. 2013 [cit. 2014-05-26]. Dostupné z: [http://crsouza.blogspot.cz/2009/11/neural-network-learning-by-levenberg\\_18.html](http://crsouza.blogspot.cz/2009/11/neural-network-learning-by-levenberg_18.html)
- [17] Levenberg-Marquardt algorithm for multivariate optimization. *ALGLIB* [online]. 2013 [cit. 2014-05-26]. Dostupné z: <http://www.alglib.net/optimization/levenbergmarquardt.php>
- [18] Gauss-Newton Method. *Neos* [online]. 2012 [cit. 2014-05-14]. Dostupné z: <http://neos-guide.org/content/gauss-newton-method>
- [19] Learning and neural networks. *Wikiiversity* [online]. 2012 [cit. 2014-05-12]. Dostupné z: [http://en.wikiversity.org/wiki/Learning\\_and\\_neural\\_networks](http://en.wikiversity.org/wiki/Learning_and_neural_networks)
- [20] CHURÝ, Lukáš a Martin ŠIMEČEK. Umělá inteligence - 4. *Programujte.com* [online]. 2009 [cit. 2014-05-13]. Dostupné z: <http://programujte.com/clanek/2009031100-umela-inteligence-4/>
- [21] CHRAMCOV, Bronislav. Základy práce v prostředí Mathematica. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2006, 122 s. ISBN 80-731-8510-5.
- [22] SJÖBERG, Jonas. *Mathematica neural networks: TRAIN AND ANALYZE NEURAL NETWORKS TO FIT YOUR DATA*. 2004.

- [23] DOSPIVA, Jan. Systém využívající rozpoznávání dopravních značek pomocí umělých neuronových sítí pro hlášení ve značení. Zlín, 2014. Diplomová. UTB Zlín. Obhajoba Červen 2014.
- [24] HRUBÝ, Bc. Josef. *Detekce a zpracování obrazových dat dopravních značek pro rozpoznávání umělými neuronovými sítěmi*. Zlín, 2014. Diplomová. UTB. Obhajoba Červen 2014.
- [25] Dopravní značky. *Vsechny-autoskoly.cz* [online]. 2009 [cit. 2014-05-20]. Dostupné z: [http://www.vsechny-autoskoly.cz/dopravni\\_znacky/](http://www.vsechny-autoskoly.cz/dopravni_znacky/)

## SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

RGB Red Green Blue.

RMSR Root Mean Square Error

## SEZNAM OBRÁZKŮ

<i>Obr. 1 - Ukázka vybraných výstražných značek.[25]</i> .....	11
<i>Obr. 2 - Ukázka vybraných značek upravujících přednost.[25]</i> .....	11
<i>Obr. 3 - Ukázka vybraných zákazových značek.[25]</i> .....	12
<i>Obr. 4 - Ukázka vybraných příkazových značek.[25]</i> .....	12
<i>Obr. 5 - Ukázka vybraných informativních značek.[25]</i> .....	12
<i>Obr. 6 - Ukázka vybraných dodatkových tabulí.[25]</i> .....	13
<i>Obr. 7 – Boční umístění dopravních značek.</i> .....	16
<i>Obr. 8 – Výškové umístění dopravních značek.</i> .....	16
<i>Obr. 9 – Vzdálenost mezi značkami.</i> .....	17
<i>Obr. 10 - Matematický model umělého neuronu.[5]</i> .....	20
<i>Obr. 11 – Grafy přenosových funkcí včetně jejich zápisu.[5]</i> .....	21
<i>Obr. 12 – Zpětná vazba neuronu.</i> .....	23
<i>Obr. 13 – Úplná topologie cyklické neuronové sítě.</i> .....	24
<i>Obr. 14 – Neuronová síť s jednou skrytou vrstvou.[5]</i> .....	26
<i>Obr. 15 – Logo Wolfram Mathematica 9.</i> .....	36
<i>Obr. 16 – Nápopěda softwaru Mathematica.</i> .....	37
<i>Obr. 17 – Ukázka grafických vzorů softwaru Mathematica.</i> .....	38
<i>Obr. 18 – Vytvoření sítě Perceptron.</i> .....	40
<i>Obr. 19 – Zobrazení počtu iterací,</i> .....	42
<i>Obr. 20 – Graf poklesu RMSE.</i> .....	42
<i>Obr. 21 – Příklad učení dopředné sítě s algoritmem backpropagation.</i> .....	43
<i>Obr. 22 – Převedení RGB modelu (vlevo) do stupně šedi</i> .....	46
<i>Obr. 23 – Zmenšení originálního obrázku a následné ořezání.</i> .....	46
<i>Obr. 24 – Metody transformací obrazových dat.</i> .....	47
<i>Obr. 25 – Grafy aplikovaných transformací na různé barevné modely.</i> .....	48
<i>Obr. 26 - Grafy aplikovaných transformací na různé barevné modely.</i> .....	49
<i>Obr. 27 – Převedení obrázku do binární</i> .....	50
<i>Obr. 28 - Převedení obrázku do binární podoby metodou ThresholdAdaptive().</i> .....	51
<i>Obr. 29 – Převedení snímku do binární podoby pomocí dvou odlišných metod.</i> .....	52
<i>Obr. 30 – Příklady správného a chybného rozpoznání kandidáta.</i> .....	55
<i>Obr. 31 – Zástupci dopravních značek každé kategorie.</i> .....	57
<i>Obr. 32 - Grafy aplikovaných transformací na barevný model RGB a stupeň šedi.</i> .....	58

<i>Obr. 33 – Inicializace a příprava první fáze učení dopředné neuronové sítě. ....</i>	<i>59</i>
<i>Obr. 34 – Graf průběhu klesání globální chyby (RMSE). ....</i>	<i>59</i>
<i>Obr. 35 – Ořezání a binarizace snímku. ....</i>	<i>60</i>
<i>Obr. 36 - Inicializace a příprava druhé fáze učení dopředné neuronové sítě. ....</i>	<i>61</i>
<i>Obr. 37 - Graf průběhu klesání globální chyby (RMSE). ....</i>	<i>61</i>
<i>Obr. 38 – Rozdělení, ořez a binarizace snímků. ....</i>	<i>63</i>
<i>Obr. 39 – Hodnota globální chyby (RMSE) v každém kroku učení. ....</i>	<i>64</i>
<i>Obr. 40 – Úprava rozměrů, normalizace jasu a binarizace snímku. ....</i>	<i>65</i>
<i>Obr. 41 – Úprava rozměrů a binarizace snímku. ....</i>	<i>67</i>
<i>Obr. 42 – Úprava snímků v osmé a deváté fázi učení. ....</i>	<i>70</i>
<i>Obr. 43 – Hlavní oko výsledné aplikace. ....</i>	<i>72</i>
<i>Obr. 44 – Okno kalibrace obrazu. ....</i>	<i>73</i>
<i>Obr. 45 – Okno ImageRecognizer. ....</i>	<i>74</i>

**SEZNAM TABULEK**

<i>Tabulka 1 - Rozměry symbolů na vybraných značkách. ....</i>	13
<i>Tabulka 2 – Vzdálenosti dopravních značek od vozovek. ....</i>	17
<i>Tabulka 3 – Popis hlavních příkazů balíčku NeuralNetworks. ....</i>	40
<i>Tabulka 4 – Parametry inicializace FeedForward sítě včetně jejich významu. ....</i>	41
<i>Tabulka 5 – Parametry příkazu NeuralFit včetně jejich významu. ....</i>	43
<i>Tabulka 6 – Rozměry vektoru importovaných dat. ....</i>	53
<i>Tabulka 7 – Výsledné hodnoty testování učících algoritmů. ....</i>	56
<i>Tabulka 8 – Výsledné hodnoty testování přenosových funkcí. ....</i>	56
<i>Tabulka 9 – Nastavené parametry sítě v jednotlivých krocích učení. ....</i>	58
<i>Tabulka 10 – Podrobné informace o první fázi učení. ....</i>	59
<i>Tabulka 11 – Druhá fáze učení LM algoritmem. ....</i>	60
<i>Tabulka 12 – Učení GN algoritmem. ....</i>	61
<i>Tabulka 13 - Podrobné informace o druhé fázi učení. ....</i>	62
<i>Tabulka 14 - Třetí fáze učení LM algoritmem. ....</i>	63
<i>Tabulka 15 - Podrobné informace o třetí fázi učení. ....</i>	64
<i>Tabulka 16 - Podrobné informace o čtvrté fázi učení. ....</i>	65
<i>Tabulka 17 - Pátá fáze učení LM algoritmem. ....</i>	66
<i>Tabulka 18 - Šestá fáze učení LM a GN algoritmem. ....</i>	66
<i>Tabulka 19 - Podrobné informace o páté a šesté fázi učení. ....</i>	67
<i>Tabulka 20 - Sedmá fáze učení LM a GN algoritmem. ....</i>	68
<i>Tabulka 21 - Podrobné informace o sedmé fázi učení. ....</i>	68
<i>Tabulka 22 - Podrobné informace o osmé fázi učení. ....</i>	69
<i>Tabulka 23 - Podrobné informace o deváté fázi učení. ....</i>	70

## SEZNAM PŘÍLOH

PI: Fáze učení

PII: CD s ukázkovými příklady

# PŘÍLOHA P I: FÁZE UČENÍ

