

Automatické hodnocení instancí pomocí umělých neuronových sítí

Bc. Ondřej Plšek

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. Ondřej Plšek**
Osobní číslo: **A11410**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Informační technologie**
Forma studia: **prezenční**

Téma práce: **Automatické hodnocení instancí pomocí umělých neuronových sítí**

Zásady pro vypracování:

1. Seznamte se s oblastí neuronových sítí.
2. Navrhněte klasifikační proceduru pro hodnocení instancí v oblasti prodeje aut.
3. Připravte vhodná trénovací a testovací data.
4. Připravte vhodnou neuronovou síť.
5. Implementujte hodnotící systém do webové aplikace.

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. ŠNOREK M., JIŘINA M.: Neuronové sítě a neuropočítače, ČVUT, 1996, ISBN 80-01-01455-X.
3. BÍLA J.: Umělá inteligence a neuronové sítě v aplikacích, ČVUT, 1996, ISBN 80-01-01275-1.
4. 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.
5. GUTMANS, Andi, Stig S?ther BAKKEN a Derick RETHANS: Mistrovství v PHP 5. Vyd. 1. Brno: CP Books, 2005, 655 s., ISBN 802510799x.
6. VRÁNA, Jakub: 1001 tipů a triků pro PHP. Vyd. 1. Brno: Computer Press, 2010, 456 s. ISBN 9788025129401.
7. ŠÍMA J., NERUDA R.: Teoretické otázky neuronových sítí. Vyd. 1. Praha, 1996: Matfyzpress, 390 s., ISBN 80-85863-18-9

Vedoucí diplomové práce:

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

Datum zadání diplomové práce:

21. února 2014

Termín odevzdání diplomové práce:

20. května 2014

Ve Zlíně dne 21. února 2014

prof. Ing. Vladimír Vašek, CSc.
děkan



[Signature]
doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

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/bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

ABSTRAKT

Cílem diplomové práce je přijít na způsob, jak lze vyhodnocovat kvalitu automobilové inzerce pomocí umělých neuronových sítí. Teoretická část obsahuje úvod do problematiky neuronových sítí, obecně popisuje jednotlivé typy, topologie a učící algoritmy. Popisuje také problémy, které při učení mohou nastat.

V praktické části jsou nejprve popsány technologie použité pro řešení problému a způsob, jakým se získávají data z inzertních serverů. Poté je přistoupeno k realizaci samotné neuronové sítě a vyhodnocení výsledků.

Klíčová slova: neuronové sítě, data mining, web, crawling, inzerce

ABSTRACT

The aim of this thesis is to find a solution how to measure a quality of car advertisements using neural networks. The theoretical part contains an introduction to the topic of the neural networks and generally describes their types, topologies and learning rules. It describes also problems that may occur during the learning.

In the practical part used technologies are described. It describes the way how to obtain a data from advertising servers. At the end the neural network is implemented and the results are evaluated.

Keywords: neural networks, data mining, web, crawling, advertising

Chtěl bych poděkovat vedoucí diplomové práce doc. Ing. Zuzaně Oplatkové, Ph.D. za její cenné rady a odbornou pomoc při psaní diplomové práce. V neposlední řadě také mé rodině, která mě během tvorby podporovala.

OBSAH

ÚVOD	8
I TEORETICKÁ ČÁST	9
1 DATA MINING	10
1.1 HISTORIE.....	10
2 NEURONOVÁ SÍŤ	12
2.1 SROVNÁNÍ PRÁCE POČÍTAČŮ A NERVOVÝCH SÍTÍ.....	12
2.2 BIOLOGICKÝ NEURON.....	13
2.3 HISTORIE.....	16
2.4 VÍCE VRSTEV, VÍCE NEURONŮ.....	17
2.5 UMĚLÝ MODEL NEURONU.....	18
2.5.1 Aktivační funkce.....	19
2.6 TOPOLOGIE SÍTÍ.....	20
2.6.1 Dopředné sítě.....	20
2.6.2 Rekurentní sítě.....	20
2.7 UČENÍ.....	20
2.7.1 Učení s učitelem.....	20
2.7.2 Učení bez učitele.....	21
2.7.3 Overfitting a underfitting.....	22
2.8 TYPY NEURONOVÝCH SÍTÍ.....	23
2.8.1 Perceptron.....	23
2.8.2 Jednovrstvá neuronová síť.....	24
2.8.2.1 Hebbovo pravidlo.....	24
2.8.2.2 Delta pravidlo.....	24
2.8.3 Vícevrstvá perceptronová síť.....	26
2.8.3.1 Backpropagation.....	27
2.8.3.2 Levenberg-Marquardt.....	28
2.8.4 Hopfieldova síť.....	29
2.8.5 Kohonenovy mapy.....	30
2.8.6 Rekurentní vícevrstvé neuronové sítě.....	31
II PRAKTICKÁ ČÁST	33
3 POUŽITÉ TECHNOLOGIE A FRAMEWORKY	34
3.1 NODE.JS.....	34
3.1.1 Asynchronní programování.....	34
3.2 MONGODB.....	36
3.2.1 Dotazovací jazyk.....	37
3.2.2 Ukládání dat.....	37
4 HODNOCENÍ A SBĚR DAT	40
4.1 SBĚR DAT.....	40
4.1.1 Možné problémy.....	40
4.1.2 Krawler.....	41
4.1.3 Cheerio.....	42

4.2	METODIKA HODNOCENÍ DAT	44
4.2.1	Nutná omezení	44
4.2.2	Hodnotící webová aplikace	45
5	TVORBA NEURONOVÉ SÍTĚ.....	47
5.1	BRAIN.JS	47
5.1.1	Transformace dat.....	47
5.2	TOPOLOGIE SÍTĚ	49
5.2.1	Výsledná aplikace	50
5.2.2	Učení sítě.....	51
5.3	OVĚŘENÍ VÝSLEDKŮ.....	53
5.4	NEURONOVÁ SÍŤ.....	56
6	PRAKTICKÁ APLIKACE	58
6.1	MONITORING INZERCE.....	58
6.2	SEKCE TOP NABÍDKY NA INZERTNÍCH PORTÁLECH	59
6.3	ONLINE AUKCE VOZIDEL	61
6.4	DALŠÍ APLIKACE.....	61
	ZÁVĚR	62
	SEZNAM POUŽITÉ LITERATURY.....	64
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	67
	SEZNAM OBRÁZKŮ	68
	SEZNAM TABULEK.....	69
	SEZNAM PŘÍLOH.....	70

ÚVOD

Hlavním impulzem ke zvolení tohoto tématu diplomové práce byla zkušenost s dennodenní praxí autoprodávčů. Ti totiž během své pracovní doby stráví několik hodin nudnou opakující se činností, kdy vyhodnocují automobilovou inzerci na inzertních portálech. Diplomová práce si klade za cíl sestrojít funkční model neuronové sítě, který bude schopen tuto činnost automatizovat.

Teoretická část se nejprve věnuje biologickému neuronu. Poté je popsána historie sítí, jejich základní členění a učící algoritmy. Důraz je kladen především na dopředné neuronové sítě.

Následující praktická část se v úvodu zabývá použitými technologiemi a také vysvětluje postupy, využívané při sběru dat z inzertních portálů. Další kapitoly se už věnují samotné implementaci neuronové sítě pomocí knihovny brain.js a to včetně přípravy trénovacích dat a procesu učení. Na závěr je provedeno zhodnocení dosažených výsledků.

I. TEORETICKÁ ČÁST

1 DATA MINING

Data mining je analytická metodologie získávání netriviálních skrytých a potenciálně užitečných informací z dat. Někdy se chápe jako analytická součást dobývání znalostí z databází, jindy se tato dvě označení chápou jako souznačná.

Data mining se používá:

- v komerční sféře (například v marketingu při rozhodování, které klienty oslovit dopisem s nabídkou produktu),
- ve vědeckém výzkumu (například při analýze genetické informace)
- v jiných oblastech (například při monitorování aktivit na internetu s cílem odhalit činnost potenciálních škůdců a teroristů).

1.1 Historie

První náznaky aktivit, které dnes označujeme jako data mining, se objevily v 60. letech 20. století s rozvojem počítačové techniky. Šlo například o využívání regresní analýzy s automatickým výběrem proměnných a prvních rozhodovacích stromů. Většinou však šlo jen o ojedinělé nebo akademické záležitosti.

Rozvoj statistických metod, databázových aplikací a umělé inteligence spolu s rychlým růstem rychlosti a paměti počítačů byly předpoklady, které umožnily v sedmdesátých a osmdesátých letech první systematická využití data miningové metodologie v praxi. Slovní spojení data mining tehdy ovšem stále mělo spíše hanlivý příděch. Označovalo „vzobávání rozinek“ z dat, hledání korelací ve velkých datových souborech, které – jak známo ze statistické teorie – je vystaveno obrovskému nebezpečí, že „objeví“ pouze nahodilé fluktuace v datech bez možnosti zobecnění a praktického využití.

Obrat přišel počátkem devadesátých let. V té době byly již vybudovány metody, umožňující vyhnout se zmíněnému nebezpečí falešných korelací (například kontrola založená na vynechaných datech nebo na metodě Monte Carlo). Navíc zejména v USA rostla poptávka ze strany komerčních organizací, disponujících již velkými objemy dat a neschopných z nich pomocí klasických tabulačních metod získat potřebné podklady pro rozhodování. To napomohlo k rychlému etablování data miningu jako svébytného oboru aplikované vědy a k jeho širokému použití v komerční praxi. Časté aplikace jsou především v oblastech:

- přímého marketingu (výběr klientů pro oslovení),

- finančnictví (odhadování rizika, hledání podvodů)
- maloobchodního prodeje (analýza nákupních košíků)
- telekomunikací (segmentace klientů, prodej programů)
- internetového prodeje (analýza přechodů mezi stránkami, efektivita reklamy apod.).

[29]

Používanými technikami v oblasti data miningu může být například regresní analýza, shluková analýza (např. pro marketingovou segmentaci) nebo neuronové sítě.

2 NEURONOVÁ SÍŤ

Neuronová síť je algoritmus, který si bere za vzor činnost lidského mozku. Již v dřívějších dobách bylo zjištěno, že mozek je tvořen velkým množstvím vzájemně propletených buněk, které nazýváme neurony, jež spolu komunikují pomocí elektrických impulzů. Od vzniku prvních počítačů se programátorské kapacity snaží vytvořit algoritmus, který bude činnost lidského mozku napodobovat. Vzniká tak pojem umělá inteligence, který výsledky takového snažení někdy více a někdy méně reprezentuje. Princip neuronových sítí je v dnešní době implementován v řadě dostupných analytických a rozhodovacích softwarů a v různých oborech lidské činnosti podává extrémně dobré výsledky, tedy ve srovnání se „standardními“ typy rozhodovacích algoritmů.

Neuronové sítě mají (někdy až pozoruhodnou) schopnost extrahovat pravidla a trendy z komplikovaných průběhů v datech. Další vlastností, při správné aplikaci, je schopnost velmi přesně předpovědět údaje, které nebyly součástí trénovacích dat, tedy schopnost zobecňovat. Toto mají samozřejmě i jiné typy algoritmů, které spadají do oblasti Data Miningu.[3]

2.1 Srovnání práce počítačů a nervových sítí

Nervová soustava funguje zcela jinak než klasický počítač dle Von Neumannovy architektury. V následující tabulce je provedeno srovnání klíčových prvků každého počítače a nervové soustavy.

Tabulka 1: Srovnání klíčových funkcí počítače a nervové soustavy

Počítač	Nervová soustava
Existence centrálního procesoru provádějícího veškerou činnost	Centrální procesor nebyl v nervových tkáních dosud objeven a nic nepotvrzuje jeho existenci
Nízká paralelizace, výjimečně více procesorů (< 1000) komunikujících mezi sebou	Velmi vysoký počet jednodušších (z hlediska zpracování informace) procesorů-neuronu (cca 10^{12}), velmi vysoká hustota propojení mezi nimi (10^3) pro každý neuron
Přesná znalost způsobu vyhodnocení a zpracování informace (stroje typu RAM –	Velmi matná představa o činnosti jednotlivých elementů, téměř žádná představa o

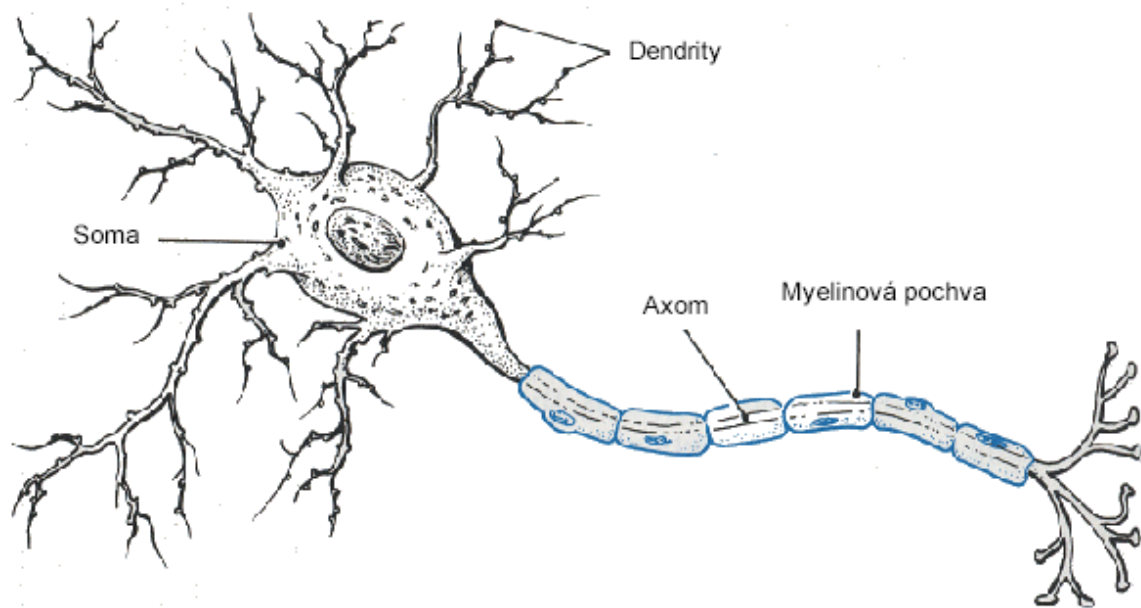
Random Access Machine, RASP – Random Access Stored Program, Turingův stroj)	způsobu komunikace mezi elementy
Nutnost detailní algoritmizace, podložená hlubokým teoretickým pozadím (např. nutnost existence poznatků lineární algebry pro řešení soustav lineárních rovnic).	Zdá se, že schopnost vyhodnocování různých situací je integrální vlastností nervových systémů bez potřeby pochopení a uvědomění si způsobu zpracování informace
Velmi vysoká rychlost početních elementárních operací (dnes cca 3GHz).	Pomalý přenos informace (řádu milisekund)
Přesné definovaná architektura procesoru	Velký počet lokálních elementů vzájemně propojených, s velkou variabilitou hustoty a způsoby spojení.

[4]

2.2 Biologický neuron

Lidský mozek je neuvěřitelně složitý orgán. Je zodpovědný za naše uvažování a řídí funkce celého lidského těla. Jeho základní stavební jednotkou je nervová buňka – neuron. V průměrném lidském mozku se jich nachází asi 1×10^{10} .

Neuron je tvořen sférickým tělem zvaným soma. Signály, které generuje, jsou přenášeny do ostatních neuronů výběžky zvanými axony nebo nervová vlákna. Dalším typem stromovitých výběžků jsou dendrity, které jsou zodpovědné za příjem příchozích signálů generovaných jinými neurony.



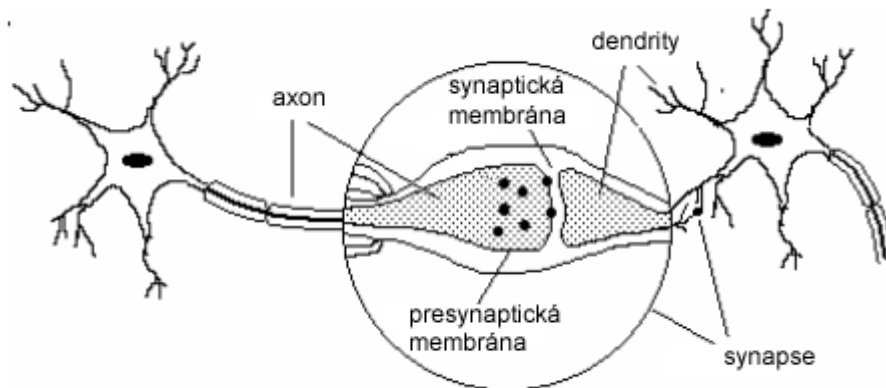
Obrázek 1: Nervová buňka[33]

Délka axonu se pohybuje od zlomku milimetru do jednoho metru. Axon je vždy pouze jeden, je to vlastně prodloužení buněčného těla. Na konci se axon rozšiřuje a je spojen s druhým neuronem pomocí synapsí. V synapsích se neurony přímo nedotýkají, ale je mezi nimi tzv. synaptická štěrbina o šířce asi 20nm. Nejběžnějším spojením je spojení mezi axonem prvního neuronu a dendrity druhého neuronu. I když to není zcela běžné, synaptické spojení může vzniknout i mezi dvěma dendrity nebo axony.

Neurony jsou pokryté semipermeabilní membránou, jejíž tloušťka je pouhých 5nm. Membrána je schopná selektivně absorbovat a odmítat ionty intracelulární tekutiny. Membrána se v zásadě chová jako iontová pumpa, která udržuje rozdílnou koncentraci iontů mezi intracelulární a extracelulární tekutinou. Pro zachování rovnováhy jsou ionty sodíku neustále extrahovány z intracelulární tekutiny, zatímco ionty draslíku jsou absorbovány dovnitř buňky. Právě díky rozdílu koncentrace iontů se membrána polarizuje. V rovnovážném stavu má neuron tzv. klidový potenciál, jehož hodnota je asi -70mV.

Neuron přijímá vstupní signály z velkého počtu ostatních neuronů skrze synaptická spojení. Nervový signál putující přes presynaptickou membránu způsobí, že jsou neurotransmitery vylity do synaptické membrány. Tyto poté putují a vážou se k receptorům na postsynaptické membráně, což způsobí zvýšení nebo snížení jejího potenciálu. Při depolarizaci (zvýšení permeability pro sodné, draselné a chloridové ionty) mluvíme o excitaci synapse,

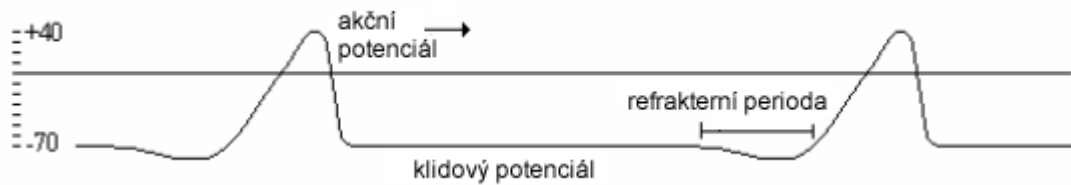
při hyperpolarizaci (zvýšení permeability pro draselné a chloridové ionty) mluvíme o inhibici synapse. Ke vzniku vzruchu musí dojít k místní depolarizaci membrány a náhlému rychlému poklesu membránového potenciálu. Prahová hodnota, při které dojde k otevření elektricky řízených iontových kanálů a ke vzniku vzruchu, je -55mV . Většinou je zapotřebí více signálů z více neuronů, aby bylo možné překonat prahovou hodnotu.



Obrázek 2: Detail synaptické membrány [5]

Excitace neuronu způsobí v axonovém výběžku dočasné přerušení transportu iontů skrze buněčnou membránu, tím se naruší rovnováha mezi vnitřní částí buňky a jejího okolí. Pokud je depolarizace dostatečně velká, membrána zkolabuje a neuron získá nakrátko kladný – akční potenciál. Akční potenciál poté putuje směrem k přilehlému axonovému výběžku a přenáší se tak na ostatní neurony.

Po každém přenosu akčního potenciálu následuje tzv. refrakterní perioda, kdy jsou iontové kanálky v deaktivovaném stavu a nelze je otevřít bez ohledu na membránový potenciál. Díky této skutečnosti je možné přenášet signály pouze jedním směrem. Refrakterní perioda trvá obvykle cca 1ms . Celkový počet impulzů a rychlost, s jakou se šíří do synapsí daného neuronu, určuje, zda bude depolarizace dostatečná k překonání prahové hodnoty a tím pádem k excitaci neuronu a vyslání signálu do axonu.



Obrázek 3: Šíření akčního potenciálu v čase [5]

Axony jsou většinou obaleny myelinovou pochvou, která je izoluje od extracelulární tekutiny a také v závislosti na své šířce ovlivňuje rychlost přenosu signálu. Myelinová pochva je přerušována Ranvierovými zářezy, které působí jako přírodní zesilovač. Nervový signál přeskakuje z jednoho na druhý, při každém skoku dochází k obnovení jeho původní intenzity.

Velmi často je přenos nervového signálu mezi neurony popisován jako přenos signálu digitálního, kde je neuron buď plně aktivní, nebo úplně neaktivní. Nicméně není to tak úplně pravda, protože intenzita nervového signálu je dána jeho frekvencí. Mnohem lepší je proto uvažovat o biologické nervové soustavě jako o formě přenosu informace za pomoci pulzní modulace. Jednotlivé signály mají téměř stejnou amplitudu, ale počet generovaných pulzů a jejich časové rozmezí se liší.[5]

2.3 Historie

I přesto, že jsou neuronové sítě považovány za relativně novou záležitost, první pokusy se datují již do první poloviny 20. století. V roce 1943 Warren McCulloch a Walter Pitts představili první model jednoduchého neuronu. Model měl dva vstupy a jediný výstup. Váhy pro každý vstup byly stejné a výstup byl binární. Dokud součet všech vstupů nepřekonal určitou prahovou hodnotu, výstup zůstal nulový. Později vzešel neuron Warrena McCullocha a Waltera Pittse ve známost jako logický obvod.

První praktické využití neuronové sítě přišlo roku 1958 s vynálezem perceptronu, který je schopen učení, Frankem Rosenblattem. Perceptrony využil k sestavení neuronové sítě k rozpoznávání vzorů. Prvotní úspěch přinesl velké nadšení. Později se ale ukázalo, že základní perceptronová neuronová síť dokáže řešit pouze lineárně separovatelné úlohy a nadšení tak upadlo.

Zhruba ve stejném období přišel Bernard Widrow a Ted Hoff se sítěmi ADALINE (Adaptive Linear Neuron) a MADALINE (Multiple ADALINE) a novým učícím algoritmem, který se používá do dneška, tzv. delta pravidlo.

Rosenblatova i Widrowova síť trpěla stejnými omezeními – dokázala řešit pouze lineárně separovatelné úlohy. Tuto skutečnost publikoval Marvin Minsky a Seymour Papert ve své knize. Rosenblat i Widrow si byli těchto omezení vědomi a tak navrhli nové sítě, které tyhle problémy odstraňovaly. Nebyli však schopni upravit jejich učící algoritmy pro trénování více komplexních sítí.

Tahle skutečnost a také nedostatek výpočetní síly tehdejších počítačů stály za utlumením výzkumu v oblasti neuronových sítí na více než deset let. I v tomto „období temna“ bádání v omezené míře pokračovalo. Nové modely sítí uvedli například pánové Steve Grossberg, Teuvo Kohonen nebo Henry Klopff.

Obnovení zájmu nastalo až v 80. letech. Nejprve John Hopfield prezentoval nové myšlenky ve své seminární práci. Tvrdil, že by se výzkum neuronových sítí neměl zabývat pouze vytvářením umělých modelů, ale měl by zkoumat techniky, které by mohly pomoci při řešení každodenních problémů. Poté byl roku 1986 objeven nezávisle několika výzkumníky backpropagation algoritmus, který umožnil trénování vícevrstvých perceptronových sítí. Řešil tak problém, který koncem 60. let kritizoval Minsky a Papert.[6,7,8]

2.4 Více vrstev, více neuronů

V současnosti se výzkum v oblasti neuronových sítí ubírá k co nejvěrnější simulaci lidského mozku. Cílem je vytvořit síť s mnoha vrstvami složených z velkého počtu neuronů, které budou schopny rozpoznat v datech více charakteristických vlastností. Výzkumníkům se ale dlouhou dobu nedařilo vyvinout algoritmus, který by tyto sítě dokázal spolehlivě naučit. Pozitivní výsledky byly zaznamenány pouze na 2 – 3 vrstvých sítích.

Zlom nastal až v roce 2006, kdy G. E. Hinton a jeho tým představili na univerzitě v Torontu DBN (Deep Belief Networks) a algoritmus pro jejich naučení.

Na vzestupu jsou také sítě, které nepotřebují k prvotnímu naučení označená data. Adaptační proces u takových sítí probíhá následovně. Nejprve síti předložíme velké množství neoznačených dat, síť se mezi nimi snaží najít charakteristické vlastnosti a data správně klasifikovat. Poté ještě síť natrénujeme s označenými daty. Výhoda takového postupu je zřejmá.

Neoznačených dat je v drtivé většině případů daleko více než dat označených. Proces učení tak může být mnohem automatizovanější.

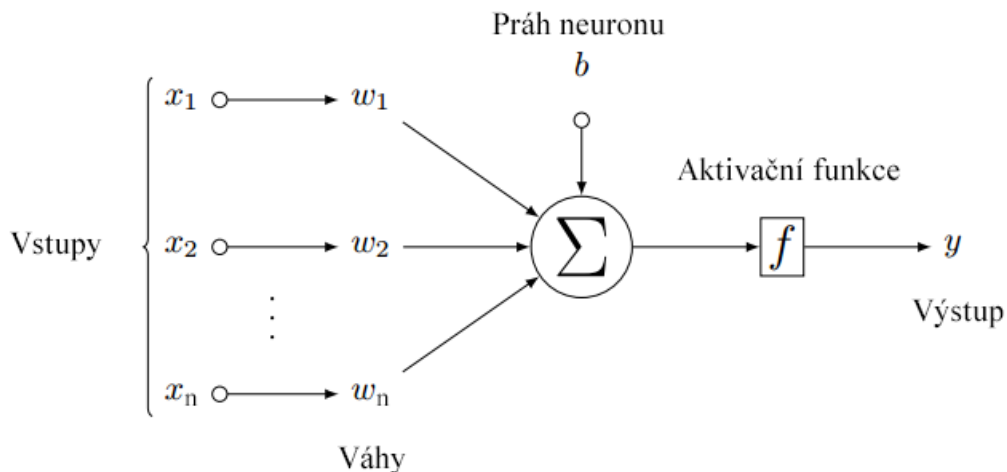
Jedním ze zajímavých projektů současnosti je Human Brain Project. Ten se snaží o detailní zmapování lidského mozku a o kompletní přenesení jeho funkčnosti do virtuálního světa.

Neuronové sítě jsou v současnosti nejvyspělejší algoritmy z oblasti strojového učení, které známe a jejich budoucnost je více než nadějná.[10, 11, 12, 13]

2.5 Umělý model neuronu

Jak bylo popsáno v kapitole 2.1, přenos nervového signálu z jednoho neuronu na ostatní skrze synapse je komplexní chemický proces.

Umělý model neuronu je inspirován svým biologickým vzorem a je jakousi základní výpočetní jednotkou neuronových sítí. Obsahuje několik vstupů (x_1, x_2, \dots, x_n), které jsou ohodnoceny vahami (w_1, w_2, \dots, w_n) a jeden výstup y . Čím je hodnota váhy vyšší, tím je daný vstup důležitější. Dokud hodnota součtu všech vah $\sum_{i=1}^n (w_i x_i)$ nepřekročí práh b , zůstává neuron v pasivním stavu.



Obrázek 4: Umělý model neuronu

Celkový výstup neuronu y je pak dán vztahem

$$y = f\left(\sum_{i=1}^n (w_i x_i) + b\right) \quad (1)$$

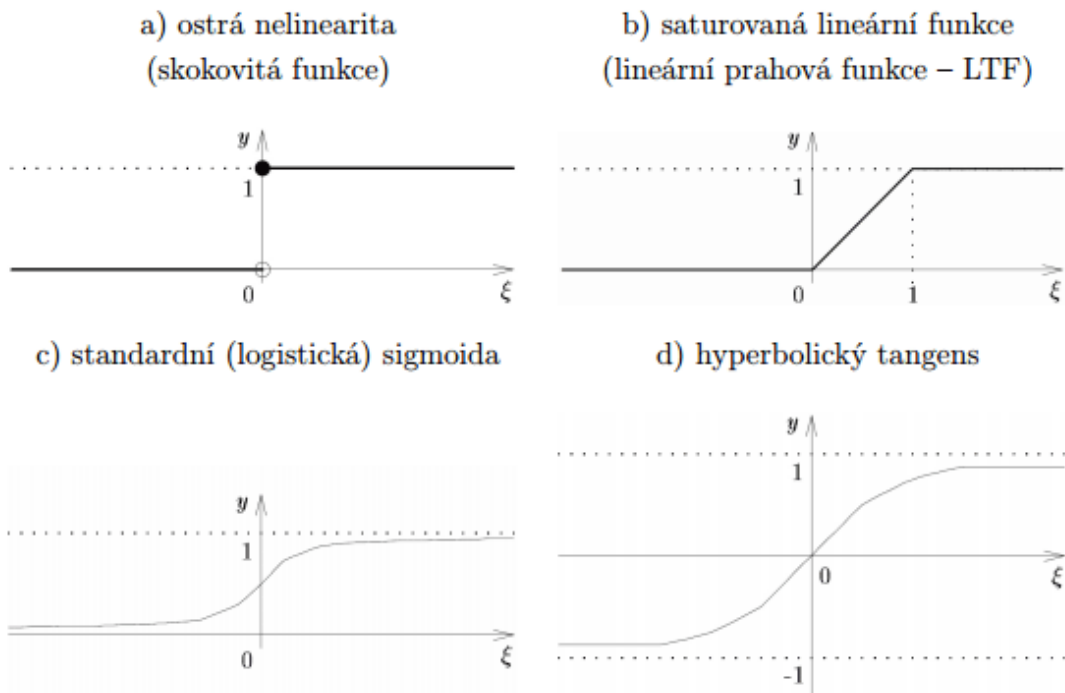
kde f je aktivační funkce neuronu [5]

2.5.1 Aktivační funkce

Aktivační funkce umělého neuronu je v podstatě abstrakce akčního potenciálu jeho biologického vzoru. Převádí vnitřní potenciál neuronu do definovaného oboru výstupních hodnot y . Nejčastěji je obor výstupních hodnot omezen intervalem $[0,1]$ nebo $[-1,1]$. V nejjednodušší formě může být binární. Říká, zda je neuron v excitovaném stavu nebo ne. Mnohem rozšířenější a použitelnější je logistická sigmoida, neboť je diferencovatelná v celém svém definičním oboru. Je dána vztahem

$$y = \frac{1}{1 + e^{-x}} \quad (2)$$

Na následujícím obrázku jsou znázorněny některé z aktivačních funkcí.



Obrázek 5: Aktivační funkce [4]

2.6 Topologie sítí

Neuronové sítě jsou inspirovány svým biologickým vzorem. Mohou být tvořeny jedním nebo i několika stovkami neuronů a množstvím spojení mezi nimi. Dle typu spojení rozlišujeme několik typů základních architektur: dopředné sítě a rekurentní sítě.

2.6.1 Dopředné sítě

V dopředných neuronových sítích jsou neurony organizovány do vrstev. Neurony ve vrstvách dostávají vstupní signály z neuronů v předchozích vrstvách a své výstupy posílají zase neuronům ve vrstvách nižších. V tomto typu sítí nejsou povolena vzájemná spojení neuronů ve stejné vrstvě. Poslední vrstva neuronů se nazývá výstupní a vrstvy mezi vstupní a výstupní jsou nazývány vrstvy skryté. Vstupní vrstvu tvoří pouze neurony, které zpracovávají externí vstup do výstupního signálu. Síť, ve které není přítomná skrytá vrstva, nazýváme jednovrstvé sítě. Pokud je přítomna alespoň jedna skrytá vrstva, mluvíme o vícevrstevných sítích.

2.6.2 Rekurentní sítě

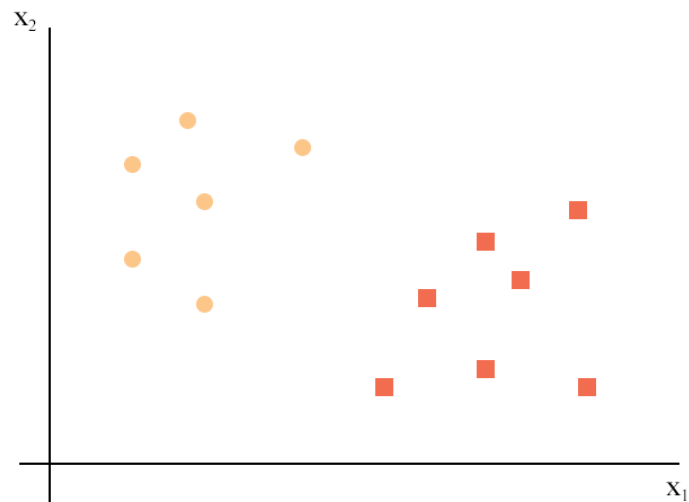
Struktury, kde je díky zpětným vazbám sítě umožněn i tok informací vazbě z jakéhokoliv výstupního bodu zpět na vstup sítě, do určité vrstvy nebo mezi neurony stejné vrstvy, nazýváme rekurentní neuronové sítě. Příkladem rekurentní sítě je například Hopfieldova síť.

2.7 Učení

Proces učení je nedílnou součástí aplikace neuronových sítí. Snažíme se při něm nastavit váhy a prahové hodnoty jednotlivých neuronů tak, aby množina vstupů vytvářela správný výstup. Na začátku učení bývají váhy nejčastěji nastaveny na náhodná čísla. Můžeme také upravovat přenosové funkce, počty neuronů v síti, popřípadě i celkovou topologii sítě. Typy učení můžeme rozdělit do dvou skupin: učení s učitelem a učení bez učitele.

2.7.1 Učení s učitelem

Je zřejmě nejběžnějším typem učení. Probíhá na trénovací množině s označenými daty. Neuronová síť upravuje váhy jednotlivých neuronů tak, aby rozdíl mezi vstupem a očekávaným výstupem byl co nejmenší. Velikost změn vah je obvykle malá. Grafická znázornění jsou zobrazena na následujícím obrázku:



Obrázek 6: Znázornění označených dat

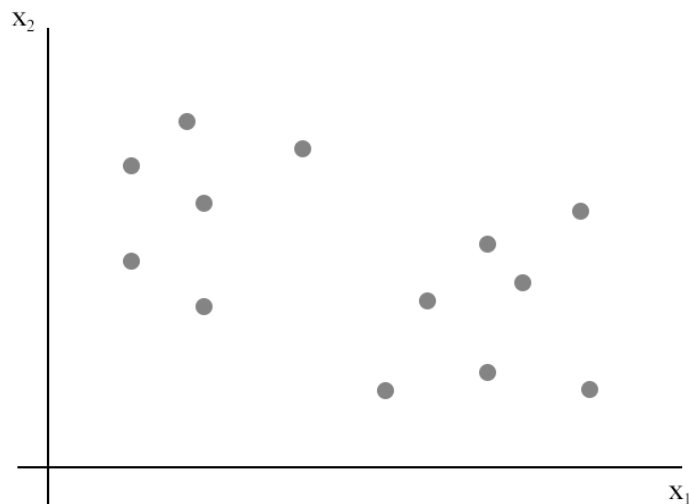
Asi nejznámějším algoritmem pro učení s učitelem je backpropagation. Toto téma a jednotlivé učící algoritmy budou rozebrány v následujících kapitolách podrobněji.

Nevýhodou tohoto typu učení je právě přítomnost označených dat. Ta je totiž mnohdy obtížné sehnat v dostatečném počtu.

2.7.2 Učení bez učitele

Při tomto typu učení je síti taktéž poskytnuta trénovací množina, ale s daty, která nejsou nikterak označená. Síť si musí v datech najít strukturu pro jednotlivé vzory. Algoritmy jsou založené na vzájemné soutěži mezi jednotlivými neurony. Učení bez učitele se také říká samoorganizace.

Jak takové množiny dat můžou vypadat lze vidět na obrázku 7:



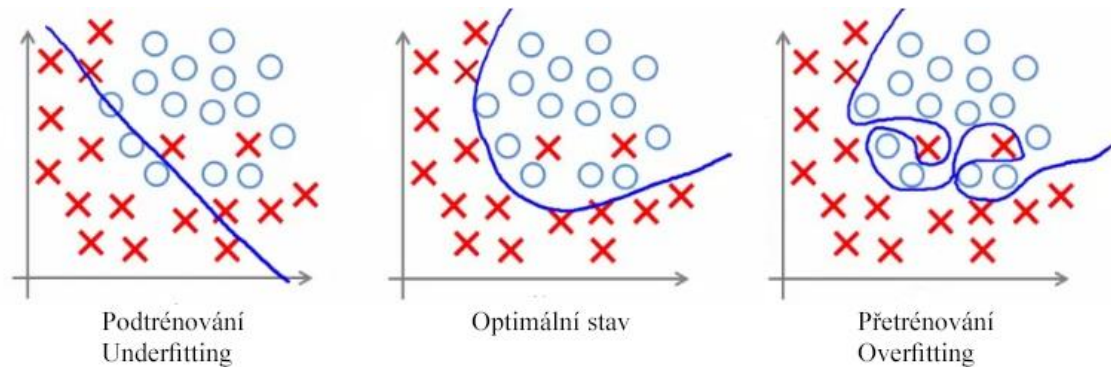
Obrázek 7: Znáznornění neoznačených dat

Velkou výhodou všech algoritmů, které se řadí do kategorie učení bez učitele, je schopnost pracovat s neoznačenými daty. Tyto algoritmy ovšem nelze použít pro všechny typy problému.

2.7.3 Overfitting a underfitting

Jedním z problémů, který může během procesu učení nastat, je tzv. overfitting neboli přetrénování sítě. Při přetrénování je síť výborně adaptována na trénovací data, ale nedokáže zobecňovat.

Opačným problémem je tzv. underfitting, který vzniká, když síť není dostatečně adaptována ani na trénovací data. Snažíme se tedy dosáhnout optimálního stavu, kdy síť dokáže aproximovat trénovací data s přijatelnou chybou a zároveň dokáže dostatečně zobecňovat. Popsané situace jsou znázorněny na následujícím obrázku.



Obrázek 8: Ilustrace možných stavů po naučení sítě [35]

2.8 Typy neuronových sítí

Protože jsou všechny neuronové sítě založené na stejném konceptu vzájemně propojených neuronů a jejich aktivačních funkcích, jsou si také všechny podobné svou strukturou. Nejvíce rozdílů nalezneme v jejich způsobu učení a také v tom, jak učení ovlivňuje jejich typickou topologii.

Obecně aplikace neuronových sítí spadají do několika hlavních kategorií:

- Predikce
- Klasifikace
- Asociace
- Segmentace
- Filtrace

2.8.1 Perceptron

Perceptron je nejjednodušším modelem dopředné neuronové sítě. Sestává pouze z jednoho neuronu. Jeho vnitřní potenciál je definovaný jako vážený součet všech vstupů. Dokud je vnitřní potenciál menší než prahová hodnota výstupem je 0, po překonání dojde k excitaci a výstupem se stává 1.

Učení probíhá na základě tréninku s učitelem. Perceptronu jsou předkládány vstupní vektory spolu s očekávanými výstupy a učicí pravidlo následně upravuje hodnotu jednotlivých vah a prahu tak, aby byl vypočítaný výstup co nejblíže originálu.

Jednoduchý perceptron ale dokáže rozdělit prostor možných řešení pouze na dva poloprostory, proto je jeho použití omezené. Pro řešení komplexnějších úloh je nutné neurony spojit do neuronových sítí.

2.8.2 Jednovrstvá neuronová síť

Jak již bylo zmíněno v předchozí kapitole, jeden neuron není schopen řešit složitější problémy. Tento problém lze překonat, když jednotlivé neurony spojíme do sítě. Jeden neuron tak řeší pouze část celého problému.

Jednovrstvá neuronová síť obsahuje jednu vstupní vrstvu s libovolným počtem neuronů a jednu vrstvu výstupní. Výstupní signály neuronů vstupní vrstvy jsou použity jako vstupní signály pro neurony vrstvy výstupní. Aktivační funkce může být lineární i nelineární. Pro naučení lze použít např. velmi jednoduché Hebbovo pravidlo nebo pravidlo Delta.

2.8.2.1 Hebbovo pravidlo

První učící pravidlo pro neuronové sítě bylo představeno Donaldem Hebbem roku 1949 v jeho knize *The Organization of Behavior*. Spočívá v jednoduché myšlence: Pokud neuron obdrží vstupní signál z jiného neuronu a pokud jsou oba v excitovaném stavu, tak by váha mezi nimi měla být zesílena.

2.8.2.2 Delta pravidlo

Delta pravidlo bylo poprvé použito v 60. letech pány Bernardem Widrowem a Tedem Hoffem v neuronové síti ADALINE. Někdy je také označováno jako Widrow-Hoffovo pravidlo.

Delta pravidlo se pokouší minimalizovat celkovou chybovou funkci výstupu pomocí gradientního sestupu.

Chybová funkce E s počtem výstupů j může být definována jako

$$E = \sum_j \frac{1}{2} (t_j - y_j)^2 \quad (3)$$

kde t je očekávaný výstup a y je skutečný výstup neuronu

Snažíme se najít takové váhy, aby hodnota celkové chybové funkce byla minimální. Hledáme tedy směr poklesu chyby E v závislosti na vahách sítě. Postupujeme následovně: Nejprve spočítáme parciální derivace chybové funkce pro každou váhu s indexem i jako

$$\frac{\partial E}{\partial w_{ji}} \quad (4)$$

Protože počítáme s počtem j neuronů, můžeme přepsat chybovou funkci E s vynecháním sumace

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial \left(\frac{1}{2} (t_j - y_j)^2 \right)}{\partial w_{ji}} \quad (5)$$

Zjednodušíme a rozložíme jako dvě derivace

$$\frac{\partial \left(\frac{1}{2} (t_j - y_j)^2 \right)}{\partial y_j} \frac{\partial y_j}{\partial w_{ji}} \quad (6)$$

Levou derivaci spočítáme jako

$$-(t_j - y_j) \frac{\partial y_j}{\partial w_{ji}} \quad (7)$$

$$-(t_j - y_j) \frac{\partial y_j}{\partial h_j} \frac{\partial h_j}{\partial w_{ji}} \quad (8)$$

A pravou

Protože výstup j neuronu y_j je vlastně aktivační funkcí g vstupu neuronu h_j , můžeme přepsat derivaci y_j podle h_j jako první derivaci g

$$-(t_j - y_j) g'(h_j) \frac{\partial h_j}{\partial w_{ji}} \quad (9)$$

Dále přepíšeme h_j jako součet všech k vah w_{jk} vynásobený odpovídajícím vstupem x_k

$$-(t_j - y_j) g'(h_j) \frac{\partial \sum_k x_k w_{jk}}{\partial w_{ji}} \quad (10)$$

Protože nás zajímá pouze váha s indexem i je jediným relevantním výstupem sumace $x_i w_{ji}$, tedy

$$\frac{\partial x_i w_{ji}}{\partial w_{ji}} = x_i \quad (11)$$

Dostáváme finální rovnici pro gradient

$$\frac{\partial E}{\partial w_{ji}} = -(t_j - y_j)g'(h_j)x_i \quad (12)$$

Zvolením učící konstanty α (learning rate) a eliminací znaménka mínus dostáváme konečnou podobu Delta pravidla

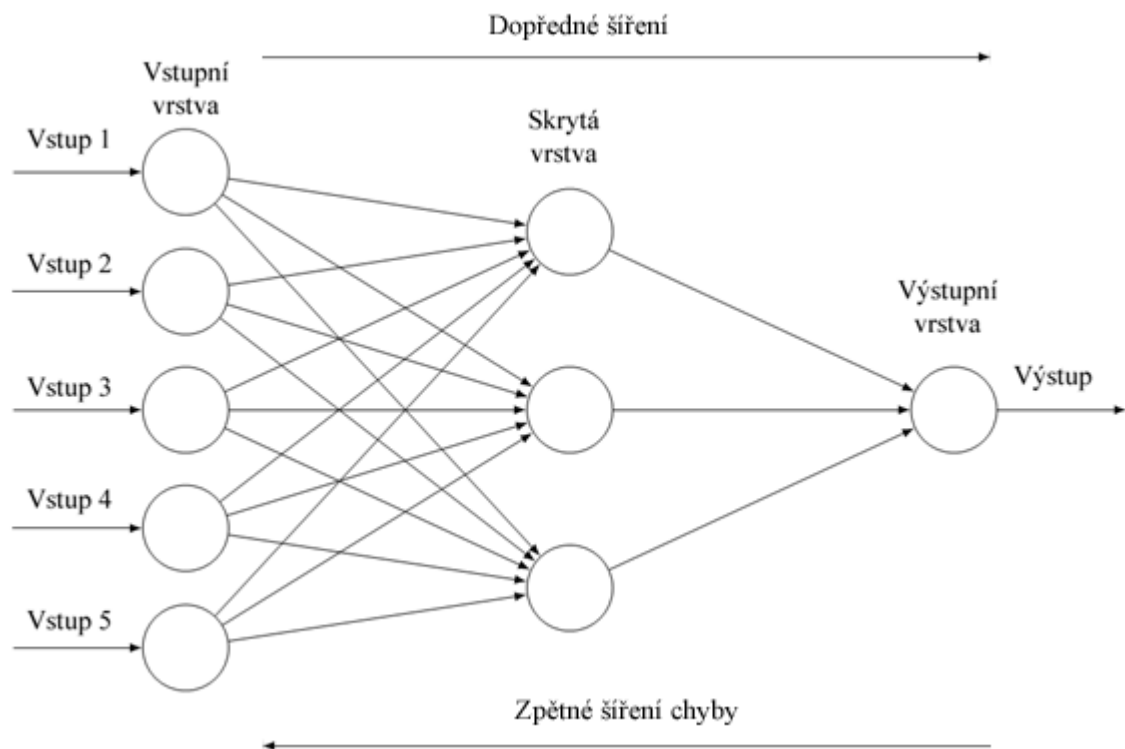
$$\Delta w_{ji} = \alpha(t_j - y_j)g'(h_j)x_i \quad (13)$$

[31, 32]

2.8.3 Vícevrstvá perceptronová síť

Jedná se patrně o nejpoblárnější typ sítě, který má využití v nejrůznějších odvětvích. Její největší silou je schopnost řešit nelineární problémy. Aktivační funkce u vícevrstevných sítí je nelineární, nejčastěji to bývá logistická sigmoida.

Typická vícevrstvá síť má jednu vstupní vrstvu, jednu výstupní a alespoň jednu skrytou. Zatímco počet vstupních a výstupních neuronů je pevně velikostí dimenze vstupního a výstupního vektor, pro určení počtu skrytých vrstev a počtu jejich neuronů neexistuje jednoznačné pravidlo. Nejrozšířenější metodou je tzv. postupná adaptace, kdy začneme s jednou skrytou vrstvou a malým počtem jejich neuronů. Síť natrénujeme a otestujeme její funkčnost. Pokud má tato síť velkou chybu, jednoduše přidáme neurony nebo další skrytou vrstvu. Takto pokračujeme až do bodu, když jsme spokojeni s výsledky. Nedá se ovšem říct, že velký počet neuronů a skrytých vrstev zaručí optimální parametry sítě. Proto musíme hledat optimální řešení, jehož nalezení není pokaždé jednoduché. [14]



Obrázek 9: Schéma dopředné neuronové sítě se zpětným šířením chyby

2.8.3.1 Backpropagation

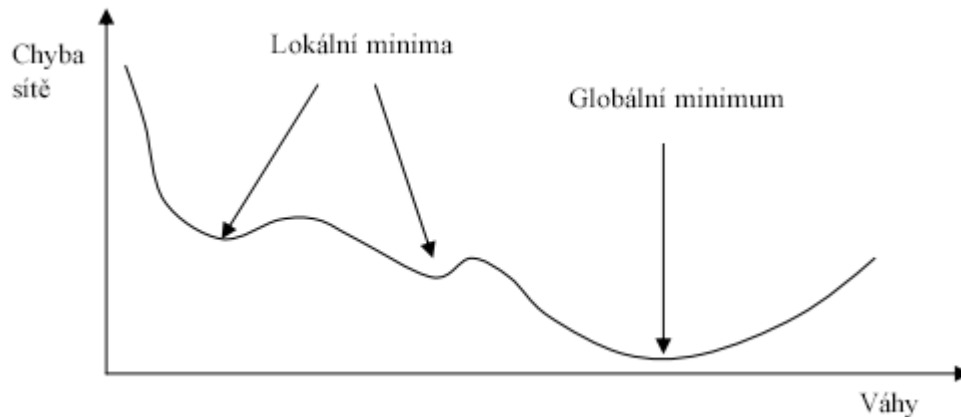
Backpropagation algoritmus je vlastně zobecněním Delta pravidla pro jednovrstvé neuronové sítě. Algoritmus je možné použít pouze pro aktivační funkce, jež jsou diferencovatelné, nejčastěji se jako aktivační funkce používá logistická sigmoida.

Algoritmus funguje následovně. Nejprve inicializujeme všechny váhy jednotlivých neuronů na náhodná čísla. Provedeme dopředné šíření signálu přes vrstvy l a neurony i , při němž spočítáme hodnoty aktivačních funkcí.

Poté provedeme výpočet chyby $\delta_i^{(l)}$, která nám říká, jak moc byl daný neuron „zodpovědný“ za chybu na výstupu. Pro neurony ve výstupních vrstvách spočítáme rozdíl mezi očekávaným a vypočítaným vstupem a použijeme jej přímo k definici jeho chyby $\delta_i^{(n_l)}$, kde n_l je výstupní vrstva.

Pro neurony $a_i^{(l)}$ ve skrytých vrstvách spočítáme jejich chybu $\delta_i^{(l)}$ jako vážený průměr chyby neuronů z následující vrstvy.[15]

Velkou nevýhodou backpropagation algoritmu je použití gradientního sestupu pro minimalizace chybové funkce. Tato metoda totiž konverguje do lokálního minima relativně pomalu a může dokonce uvíznout v některém z lokálních minim jak je zobrazeno na následujícím obrázku.



Obrázek 10: Lokální a globální minimum

Algoritmus backpropagation je i přes své zjevné nedostatky používán dodnes. Nicméně je velmi vhodné použít jinou metodu pro nalezení globálního minima chybové funkce. Jednou takovou může být například Levenberg-Marquardt.

2.8.3.2 Levenberg-Marquardt

Levenberg-Marquardt je komplexní metoda druhého řádu pro řešení úloh vznikající při nelineární metodě nejmenších čtverců. U metod druhého řádu nevyužíváme pouze derivace funkce (sklonu), ale i druhou derivaci v aktuálním bodě. Je to metoda relativně jednoduchá, ale zároveň robustní. V podstatě spočívá v řešení rovnice

$$(J^t J + \lambda I) \delta = J^t E \quad (14)$$

kde

- J je Jacobián matice systému,
- λ je tzv. Lavenbergův stabilizační faktor
- δ je vektor změny vah, který hledáme.
- E je vektor chyb, výstupu

Vektor změny vah δ nám říká, jak moc máme změnit váhy sítě, abychom dosáhli lepšího řešení. Matice $J^t J$ může být aproximována Hesiánem. Stabilizační faktor λ je upravován při

každé iteraci a řídí optimalizační proces. Pokud E konverguje příliš rychle, můžeme jako λ zvolit menší číslo. Tím pádem se algoritmus bude chovat podobně jako Gauss–Newtonův algoritmus. Naopak pokud je změna E nedostatečná, můžeme parametr λ zvětšit.

1. Vypočteme Jacobián.
2. Spočteme gradient chybové funkce $g = J^t E$.
3. Provedeme aproximace Hessiánem $H = J^t J$.
4. Vyřešíme rovnici $(H + \lambda I)\delta = g$ a nalezneme δ .
5. Za pomoci vypočteného δ provedeme úpravu jednotlivých vah sítě.
6. Spočítáme celkovou chybu.
7. Pokud se chyba nezmenšila, vrátíme váhy do původního stavu, zvětšíme parametr λ a pokračujeme znovu v bodě 2.
8. Pokud se chyba zmenšila, λ zvětšíme a pokračujeme v bodě 2.

Algoritmus končí dosažením zvolené chybové hodnoty nebo maximálního počtu iterací.

Levenberg-Marquardt je velice citlivý na počáteční nastavení vah sítě a často u něj také můžeme pozorovat overfitting.[16, 17]

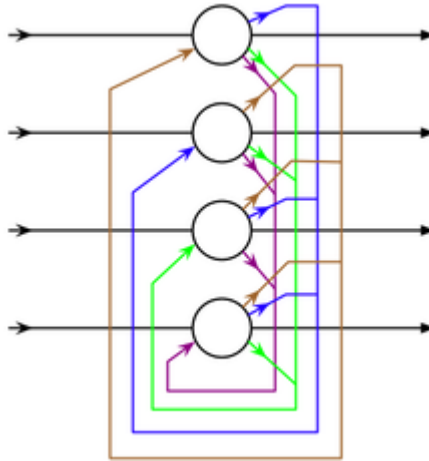
2.8.4 Hopfieldova síť

Hopfieldova síť byla poprvé představena Johnem Hopfieldem roku 1982.

Její hlavní rozdíl oproti ostatním neuronovým sítím je, že vstupní vektor je aplikován současně na všechny neurony v síti, ty jsou totiž vzájemně propojené. Po aplikaci následuje cyklus postupných změn excitací neuronů až do dosažení stabilního stavu – výstupy předchozích kroků se stávají novými vstupy současného kroku.

Inicializační stav reprezentuje různorodost excitací neuronů, které se vzhledem k tomu, že jsou všechny propojeny, začnou navzájem ovlivňovat. To může znamenat, že jeden neuron se snaží neurony excitovat na rozdíl od jiného, který se snaží o opačné. Výsledkem je nalezení kompromisu - síť relaxovala do stabilního stavu.[18]

Hopfieldova síť díky své struktuře také poskytuje model pro porozumění fungování paměti člověka.

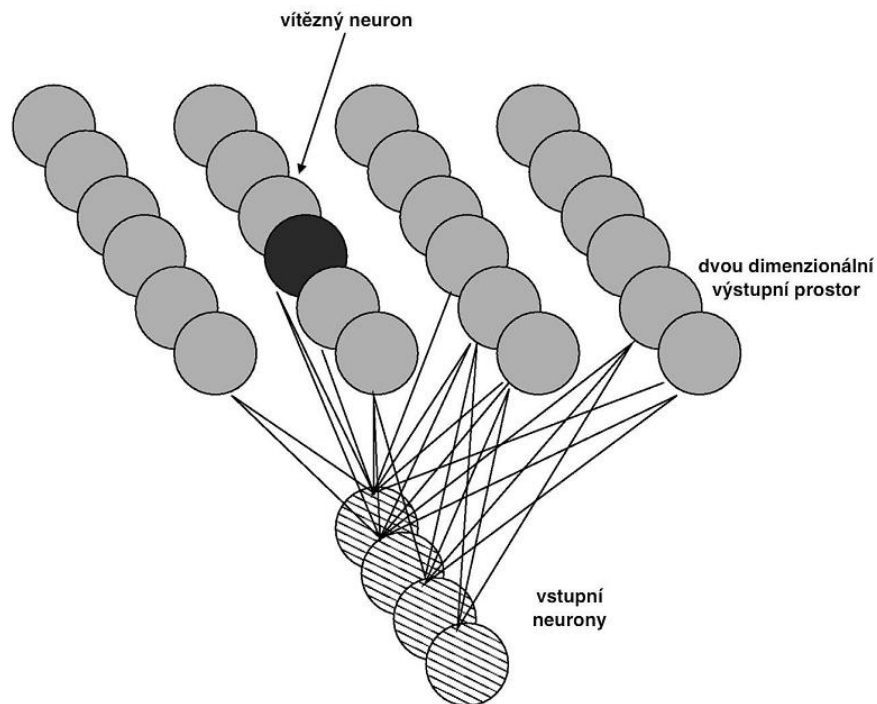


Obrázek 11: Hopfieldova síť [34]

2.8.5 Kohonenovy mapy

Kohonenovy mapy řadíme do kategorie samoučících se neuronových sítí. Jejich typickým úkolem je prostorová reprezentace složitých datových struktur. Nejčastěji se využívají pro redukci dimenze do dvou nebo třírozměrného prostoru tak, aby bylo možno tyto složité struktury názorně vizualizovat.

Dále je lze využívat k nalezení vzoru ve vstupních datech například rozpoznání hlasu, analýza signálů nebo rozeznání tváře. Využívají se taktéž pro vizualizaci rozsáhlých databází (mapy kriminality, hustota osídlení), rozpoznání textu, automatické třídění a mnohé další.



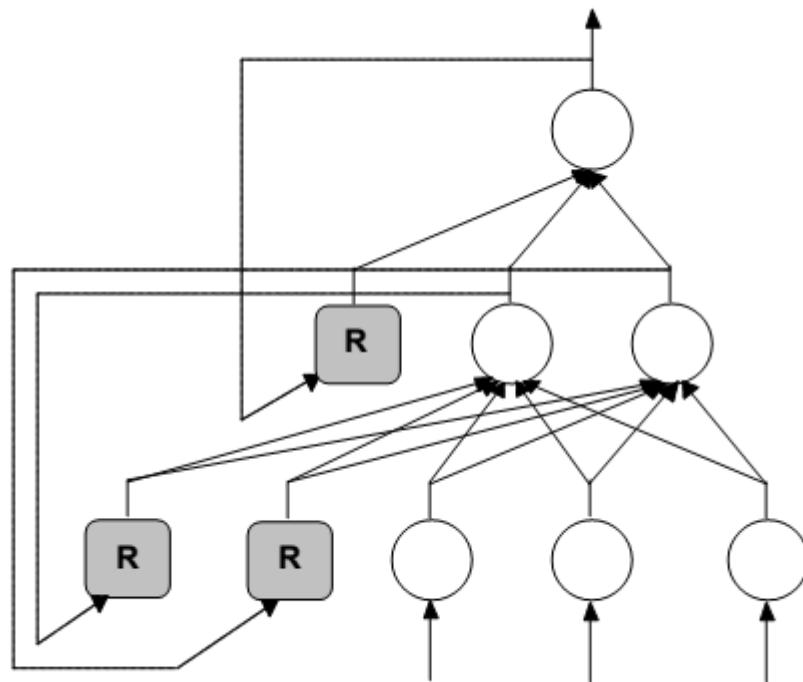
Obrázek 12: Kohonenovy mapy [34]

Učící algoritmus vychází ze strategie soutěžního učení. Při učení postupně síti předkládáme jednotlivé tréninkové vzory. Po předložení tréninkového vzoru proběhne mezi neurony soutěž, vítězným neuronem je ten, jehož Eukleidovská vzdálenost od předloženého vzoru je nejmenší. [18, 21]

2.8.6 Rekurentní vícevrstvé neuronové sítě

Všechny uvedené předchozí neuronové sítě byly nezávislé na časovém kontextu. To znamená, že stejný vstupní vektor vedl vždy na stejnou odezvu sítě. Rekurentní neuronové sítě se snaží vnést do procesu učení právě časový kontext. To znamená, že odezva sítě nebude záviset pouze na vstupním vektoru, ale bude brát v potaz i vliv vektorů, které mu předcházely. Jako učící algoritmus je často použit backpropagation.

U rekurentních sítí se signál nešíří pouze od vstupu směrem k výstupu, ale dochází i ke zpětnému přenosu signálu z vyšších vrstev do vrstev nižších pomocí tzv. rekurentních neuronů.



Obrázek 13: Rekurentní neuronová síť [18]

Rekurentní neurony jsou ve stejném počtu jako klasické neurony ve vrstvě o stupeň níže. Každý z těchto neuronů přijímá vstupní signál od příslušného klasického neuronu ve vrstvě nižší. Vyvolání informace spočívá v postupném předkládání vstupních vektorů s následnou odezvou, která bere v potaz časový kontext.[18]

II. PRAKTICKÁ ČÁST

3 POUŽITÉ TECHNOLOGIE A FRAMEWORKY

Klíčem k úspěchu jsou správně zvolené nástroje. V následující kapitole jsou obecně popsány použité technologie spolu s jejich přednostmi a omezeními.

3.1 Node.js

Jedná se o poměrně mladou technologii. První verze spatřila světlo světa v roce 2009 a jejím autorem je Ryan Dahl.

Node.js je vysoce výkonné, událostmi řízené prostředí pro Javascript. Základem Node.js je javascriptový interpret V8 z Google Chrome. Nad ním je tenká vrstva kódu v C++ poskytující minimální nutné zázemí (práce se souborovým systémem, event-loop vyhodnocující příchozí události, obsluha I/O bufferů a jiné). [19, 20]

Node.js je velmi vhodný pro realtime aplikace díky svému asynchronnímu neblokujícímu I/O modelu, naopak se nehodí pro aplikace, které jsou náročné na výkon CPU, protože pracuje pouze s jediným vláknem.

3.1.1 Asynchronní programování

Jedním ze základních pilířů Node.js je jeho asynchronní programovací model. Při volání asynchronní funkce není blokováno hlavní vlákno po celou dobu, kdy probíhá nějaká časově náročná operace. Funkce může tedy „skončit“ ještě předtím, než je znám výsledek celé operace a v okamžiku, kdy je operace dokončena, je volána tzv. *callback* funkce, která se předává jako parametr.

V případě Node.js je nepsaným pravidlem, že asynchronní funkce přebírá parametry dva. Prvním parametrem je *error callback*. Funkce, která je volána v případě chyby. Druhým parametrem je už poté klasický *success callback*, funkce, která se zavolá po úspěšném dokončení asynchronní operace.

Koncept asynchronního programování je samozřejmě přítomen i v jiných programovacích jazycích, běžně se používá například při návrhu uživatelských rozhraní. Ale teprve až Node.js jej dokázal zpopularizovat a prosadit i v takzvaném „programátorském mainstreamu“.

Asynchronní náтура celého programování v Node.js může být pro mnohé vývojáře problémem. Je nutné dbát na správnou kompozici všech funkcí a velmi pečlivě ošetřovat chybové stavy. Pokud dojde k vyhození a nezachycení výjimky v synchronním kódu, program většinou ukončí svůj běh, ale chyba bude přinejmenším zaznamenána. Pokud ovšem nastane

stejná situace někde v průběhu asynchronní funkce, vývojář se o ní nemusí vůbec dozvědět. Často pak dochází ke skrytým memory leakům, které se zpětně velmi těžko dohledávají.

Pokud vývojář nemá zkušenost se správnou kompozicí asynchronního kódu, může snadno dojít k tzv. „*callback hell*“, kód se poté snadno stane nepřehledným a neudržovatelným.

Pro ukázkou jsou uvedeny 2 příklady pseudokódu napsaného v blokujícím modelu a poté v modelu neblokujícím.

Kód napsaný v blokujícím modelu

```
var connection = connectToDb();
var user = connection.selectUser();
var emailResult = sendEmailTo(user);
connection.updateUser(user.id, emailResult);
```

Kód napsaný v neblokujícím modelu a vznik „*callback hell*“

```
connectToDb(function(err, connection) {
  connection.selectUser(function(err, user) {
    sendEmailTo(user, function(err, result) {
      connection.updateUser(user.id, result, function(err, result) {
        });
      });
    });
  });
});
```

Jak je z příkladu patrné, kód napsaný v neblokujícím modelu je mnohem hůře čitelný, nemluvě o zpracování chyb, které je v příkladu zcela ignorováno. Můžeme naštěstí využít knihoven nebo technik z oblasti funkcionálního programování (např. promises), které nám s organizací kódu pomůžou. V nově vyšlé specifikaci ECMAScript 6 lze také využít konceptu tzv. generátorů, pomocí kterých lze asynchronní kód zapsat téměř stejně jako kód synchronní.

Podrobnější zpracování tohoto tématu je již ale nad rámec diplomové práce, proto jsou v následujícím seznamu jen uvedeny názvy knihoven, které tyto problémy řeší:

- `async` (<https://github.com/caolan/async>)
- `q` (<https://github.com/kriszowal/q>)
- `promise` (<https://www.npmjs.org/package/promise>)

V základní instalaci je Node.js dodáván s promyšleným systémem pro správu externích závislostí - npm (Node Packaged Modules). V době psaní diplomové práce bylo v jeho repozitáři k dispozici přes 68 000 nejrůznějších nástrojů a knihoven.

Node.js nelze považovat za mladou technologii s nejistou budoucností. V současnosti jej používají i obři jako PayPal, Yahoo!, LinkedIn, či Heroku. [22]

3.2 MongoDB

MongoDB je multiplatformní dokumentově orientovaná databáze. Řadíme ji také mezi NoSQL databáze. Oproti tradičním relačním databázím využívají NoSQL databáze jiné prostředky pro ukládání a zpracování dat než tabulky s relačním schématem. Výhodou tohoto přístupu může být jednoduchost designu, horizontální a vertikální škálovatelnost a jemnější kontrola dostupnosti. Databáze bez SQL jsou často vysoce optimalizovaná úložiště typu klíč-hodnota (ne vždy).

Segment NoSQL databází v současnosti významně roste a prospívá především v oblasti big data a real-time webu. NoSQL systémům se také občas říká „nejen SQL“ pro zdůraznění faktu, že často umožňují dotazy v SQL (či podobném) jazyce.

Bariéry k rozsáhlejšímu nasazení těchto úložišť do praxe jsou např. nepřítomnost plnohodnotné podpory transakčního modelu ACID, použití (různých) nízkoúrovňových dotazovacích jazyků, nedostatečná standardizace rozhraní a vysoké realizované investice podniků do SQL v minulosti.[30]

Základní jednotkou je dokument - objekt ve formátu JSON. Ten může být libovolného schématu s libovolně zanořenými daty, jediným omezením je jeho maximální velikost - 16MB. Dokumenty jsou poté sdruženy do kolekcí. Pro představu si můžeme pomoci analogií s relačním databázovým schématem: tabulka \approx kolekce, dokument \approx řádek v tabulce. Každý dokument musí mít unikátní klíč `_id`, který může být libovolného typu. Pokud není uveden, je při ukládání dokumentu automaticky vytvořen jako typ `ObjectID`. V databázi je poté uložen jako hexadecimální řetězec, který je složen z aktuálního timestampu, hashe pracovního stroje, ID procesu a čísla čítače počtu dokumentů v kolekci.

Data si MongoDB ukládá data ve vlastním formátu BSON (binární JSON). [23]

3.2.1 Dotazovací jazyk

Dotazovacím jazykem je Javascript. Podmínky a omezení se zapisují jako JSON objekt. V následující tabulce jsou uvedeny ukázky dotazů pro MongoDB a jejich ekvivalenty v SQL:

Tabulka 2: Porovnání SQL dotazů s MongoDB dotazy

SQL dotaz	MongoDB dotaz
<code>SELECT* FROM users</code>	<code>db.users.find()</code>
<code>SELECT * FROM users WHERE status = "A" AND age = 50</code>	<code>db.users.find({ status: "A", age: 50 })</code>
<code>SELECT * FROM users WHERE status = "A" OR age = 50</code>	<code>db.users.find({ \$or: [{ status: "A" } , { age: 50 }] })</code>
<code>SELECT * FROM users LIMIT 5 SKIP 10</code>	<code>db.users.find().limit(5).skip(10)</code>

[24]

3.2.2 Ukládání dat

Protože v MongoDB neexistují žádné relace a vazby, je nutné tuto skutečnost zohlednit při ukládání dat. Vše by mělo směřovat k tomu, aby bylo možné data získat jedním dotazem. Nutně tak bude docházet k jejich duplikaci. Při aktualizaci některého ze sdílených záznamů bude nutné projít všechny dokumenty a v každém jej upravit. Jednoduchý způsob získávání dat je zaplacen jejich složitější aktualizací. [25]

Pro ukázkou je uveden dokument inzerátu vozidla, který obsahuje vnořené, „relační“ dokumenty pro prodejce – *vendor*, značku a model vozidla – *brand*, *model*.

```
{
  "_id": {
    "$oid": "5323aa05cbb06214c10000d5"
  },
  "title": "Prodej osobního vozu Škoda Octavia",
  "url": "http://www.aktualnivozy.cz/osobni/skoda-octavia-2-0-tdi-103kw-4513966/",
  "domain": "http://www.aktualnivozy.cz",
  "domain_id": "4513966",
  "price": 230000,
  "image": "http://www.aktualnivozy.cz/images/4514000/4513966/300x225/4513966-001-skoda-octavia-2-0-tdi-103kw.jpg",
  "vat_deductible": false,
  "description": "6000 kvalitních aut za nejlepší ceny v ČR. Největší výběr – nejlepší ceny – nejširší garance. 4500 vozů po 1.-2.majiteli. 24 měsíců garance na mechanický stav vozu, kontrola stavu najetých km. 19 poboček po celé ČR - auto vám zdarma převezeme
```

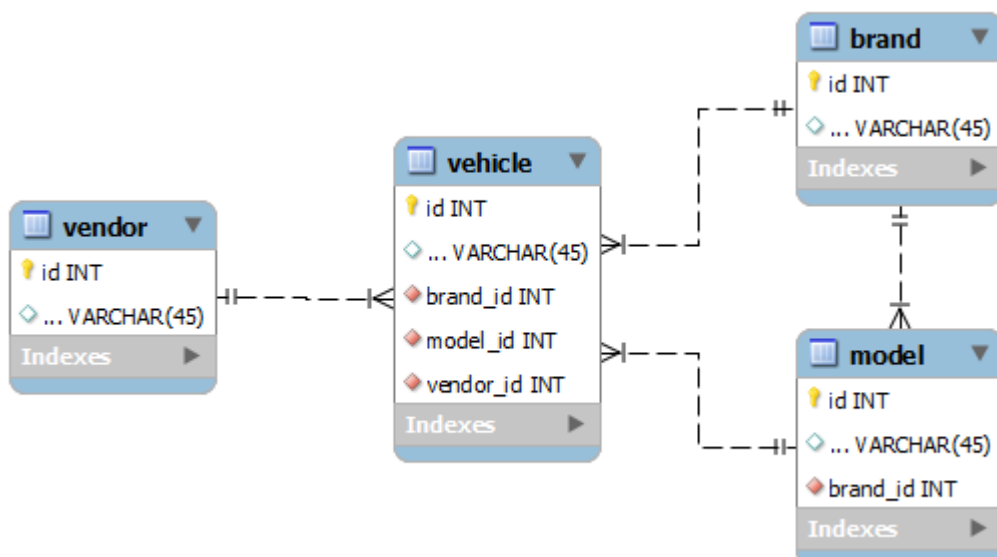
na vaši nejbližší pobočku. Poznámka: Všechny naše vozy procházejí důkladnou kontrolou 107 technických parametrů. Získejte slevu 20 000 - 50 000 při koupi vozu na úvěr! Cena při financování: 185000 Kč.",

```

    "date_crawled": {
      "$date": 1394846213413
    },
    "vendor": {
      "name": "AAA AUTO - 6000 kvalitních a prověřených vozů",
      "region": "Praha"
    },
    "fuel": "nafta",
    "color": "šedá",
    "gearbox": "manuální",
    "date_made_year": 2010,
    "km": 133617,
    "kw": 103,
    "ccm": 2000,
    "brand": {
      "_id": 93,
      "name": "Škoda",
      "kind_id": 1,
      "kind_name": "osobní"
    },
    "model": {
      "_id": 705,
      "name": "Octavia",
      "brand_id": 93,
      "kind_id": 1,
      "kind_name": "osobní"
    }
  }
}

```

V relačním databázovém modelu by struktura tabulek s relacemi vypadala zhruba takto:



Obrázek 14: Ukázkové relační databázové schéma

Z uvedených příkladů by se mohlo zdát, že strukturovaná data automobilových inzerátů by bylo vhodnější ukládat do relační databáze namísto databáze dokumentově orientované. Nicméně inzeráty jsou z různých zdrojů a data často nejsou kompletní. Jejich struktura je tak mnohdy různorodá. Někdy se také nepodaří přiřadit značku či model vozidla, ale inzerát by i přesto měl být v databázi uložen. MongoDB také poráží tradiční relační databáze co se týče rychlosti při vkládání nových dat.

4 HODNOCENÍ A SBĚR DAT

Data automobilových inzerátů byla získána z veřejně dostupných webových stránek inzertních portálů.

4.1 Sběr dat

Sběr dat je prováděn pomocí specializovaných programů, tzv. *crawlerů*. Obecně je *crawler* chápán jako program, který systematicky prochází hierarchickou strukturu internetových stránek a ukládá je do databáze pro následnou indexaci a zpracování. Nejčastěji tyto programy provozují fulltextové vyhledávače jako např. Seznam či Google.[27]

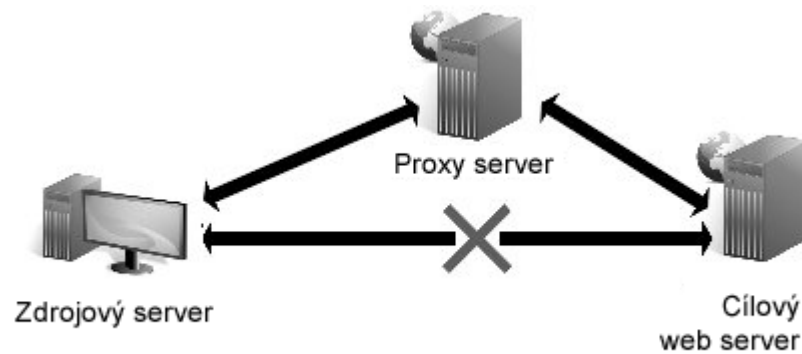
Crawler, který je použit pro potřeby diplomové práce, je více specifický. Prochází pouze vybrané servery a vyhledává pouze konkrétní informace – detaily automobilových inzerátů. Postupuje dle následujícího schématu:

1. Načte první předdefinovanou stránku (nejčastěji úvodní stránku, tzv. landing page).
2. Vyhledá na stránce odkazy na detaily automobilových inzerátů.
3. Paralelně vyhledané odkazy načte a zpracuje jejich detail.
4. Vyhledá na stránce odkazy se stránkováním a přejde na další stranu.
5. Opakuje postup od bodu 2.

4.1.1 Možné problémy

Při sběru dat ve velkých objemech je potřeba počítat s řadou překážek. Servery se pochopitelně této činnosti brání a tak mnohdy limitují počty povolených přístupů z jedné IP adresy. Jedna možnost, jak tenhle limitující faktor obejít, je využití proxy serveru. TCP/IP spojení pak neprobíhá napřímo mezi dvěma servery, ale putuje skrze prostředníka – proxy server. Pro cílový server se potom jako zdroj jeví právě proxy server. Každý požadavek na stažení webové stránky může být poté teoreticky směřován z jiné IP adresy. Jak je zobrazeno na obr 15.

Další možností je využít VPN. Zdrojový server se připojí na určitý časový úsek do privátní sítě – změní tak svou veřejnou IP adresu a může vůči cílovému serveru vystupovat s novou identitou.



Obrázek 15: Znárodnění komunikace skrze proxy server [36]

Dalším problémem, se kterým je nutno počítat, je nejednoznačnost URL. Jedna URL adresa může pod sebou skrývat rozdílný obsah a naopak, stejný obsah se může skrývat pod více URL adresami. Jako jednoznačný identifikátor stránky je proto nutné zvolit něco jiného. Lze využít například toho, že vesměs všechny weby někde zobrazují ID inzerátu. V případě serveru tipcars je přímo součástí část samotné URL.



Obrázek 16: Zobrazení ID inzerátu v adresním řádku prohlížeče

K jeho získání pak stačí použít jednoduchý regulérní výraz:

```
domain_id: url.match(/-(\d*).aspx/)[1]
```

Získané ID pak spolu s názvem serveru tvoří unikátní klíč v databázi. Tím je zabráněno vytváření duplicit v sesbíraných inzerátech.

4.1.2 Krawler

Pro potřeby sběru dat automobilových inzerátů byl napsán *crawler*, který je k dispozici na balíčkovacím systému npm pod jménem Krawler.

Díky asynchronnímu zpracování kódu v Node.js je velmi jednoduché stahovat a zpracovávat několik stránek najednou. Krawler to vše ještě zapouzdřuje do jednoduchého API.

```
var Krawler = require('krawler')
```

```
var urls = [
  'http://www.annonce.cz/inzerat/audi-a6-3-0-quattro-25024792-wpdydf.html',
  'http://www.annonce.cz/inzerat/vw-passat-variant-1-9tdi-25304112-was8mb.html',
```

```
'http://www.annonce.cz/inzerat/mercede-benz-m1-350-25328941-w71g8t.html'  
];  
  
var krawler = new Krawler({  
  maxConnections: 10, // max. počet současně otevřených spojení  
  parser: 'cheerio', // typ parseru  
  forceUTF8: false, // zda převést data do kódování UTF8  
});  
  
krawler  
  .queue(urls)  
  .on('data', function($, url, response) {  
    // $ - cheerio instance  
    // url adresa stránky  
    // response objekt  
  })  
  .on('error', function(err, url) {  
    // nastala chyba 'err' na stránce 'url'  
  })  
  .on('end', function() {  
    // všechny stránky byly staženy  
  });
```

Při vytváření nové instance lze v konstruktoru předat objekt s nastavením. Je možné omezit maximální počet otevřených spojení, jaký použít parser a zda data převést do kódování UTF-8.

Krawler se snaží simulovat chování webového prohlížeče a tak při dotazování na server odesílá i příslušné HTTP hlavičky. Bere také v potaz nastavení parseru a kódování a hlavičky dle toho upraví.

Stahování zahájí metoda *queue*, která přebírá jediný parametr a tím je pole s URL adresami. Krawler generuje události po stažení stránky, při chybě a po dokončení stahování. Na události je možné zaregistrovat nějaký *listener* pomocí metody *on*. Je to vlastně obdoba *callback* funkce.

4.1.3 Cheerio

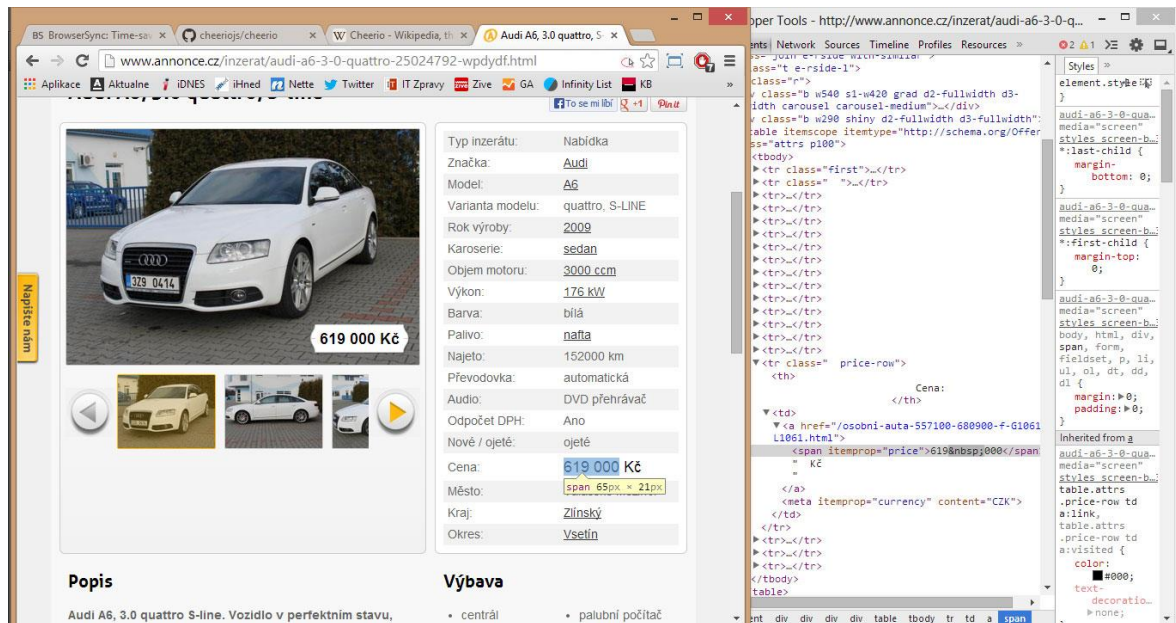
Aby bylo možné stažená data dále využívat, je potřeba je nejdříve nějakým způsobem zpracovat neboli rozparsovat. Krawler má v sobě implementován parser nejen pro HTML, ale i JSON a XML.

Parser pro HTML se nazývá *cheerio*. Implementuje klíčovou funkcionalitu velmi rozšířeného javascriptového frameworku jQuery. Pohyb v DOMu webové stránky a selekce HTML elementů se pak provádí pomocí jednoduché intuitivní syntaxe.

Na obr. 17 je ukázáno, jak lze získat titulek stránky a seznam všech HTML tagů H1.

```
var pageTitle = $('title').text();
var headings = $('h1');
```

Pro ilustraci, jak takové parsování probíhá, je na následujícím obrázku zobrazen snímek obrazovky s webovou stránkou ze serveru Annonce.



Obrázek 17: Zdrojový kód detailu inzerátu

Na obr.17 je zdrojový kód, který za použití *cheerio* extrahuje ze stránky klíčová data. Data jsou vlastnostmi JSON objektu *advert*. Před uložením do databáze jsou ještě normalizována. To znamená, že jsou odstraněny uvozující bílé znaky v textu, čísla jsou přeformátována na správný datový typ apod.

```
var $table = $("table[itemtype='http://schema.org/Offer']");

var advert = {
  title: $('h1').text().trim(),
  url: url,
  domain: self.config_domain,
  domain_id: url.match(/-(\d{5,})-/)[1],
  image: self.config_domain + $('img').attr('src'),

  price: $table.find('th:contains(Cena)').next().text().replace(/[\^0-9]/g, ''),
  km: $table.find('th:contains(Najeto)').next().text().replace(/[\^0-9]/g, ''),
  kw: $table.find('th:contains(Výkon)').next().text().replace(/[\^0-9]/g, ''),
  ccm: $table.find('th:contains(Objem)').next().text().replace(/[\^0-9]/g, ''),
  fuel: $table.find('th:contains(Palivo)').next().text().trim().toLowerCase(),
  gearbox: $table.find('th:contains(Převodovka)').next().text().trim().toLowerCase(),

  date_crawled: new Date(),
  date_made_year: $table.find('th:contains(Rok výroby)').next().text().replace(/[\^0-9]/g, ''),

  description: $('div[itemprop=description]').text().trim(),
```

```
vendor: {
  name: $("span[itemprop='seller']").text().trim(),
  region: $table.find('th:contains(Kraj)').next().text().trim(),
  locality: $table.find('th:contains(Město)').next().text().trim(),
  telephone: $(' .phone-link').text().replace(/[^0-9]/g, ''),
}
};
```

4.2 Metodika hodnocení dat

Automobilové inzeráty byly hodnoceny na základě toho, zda se jedná o „dobrou koupí“ či nikoliv. „Dobrou koupí“ můžeme zhruba charakterizovat jako vozidlo s následujícími vlastnostmi:

- Málo najetých km
- Velký objem
- Velký výkon motoru
- Nízké stáří
- Nízká cena

Musíme ale vzít také v potaz, že předchozí vymezení nejsou stejná pro všechny modelové řady případně kategorie vozidel. Do hry pak vstupuje také typ převodovky, výbava, barva, typ karoserie. Cena také nesmí být příliš nízká – s největší pravděpodobností se bude jednat o falešný inzerát nebo poškozené vozidlo.

Rozhodnout, zda je vozidlo prezentované na inzerátu „dobrá koupě“, je proto poměrně obtížné a pro potřeby diplomové práce byla přijata určitá omezení.

4.2.1 Nutná omezení

U vozidel je velké množství vlastností a parametrů, které můžeme při hodnocení brát v potaz. Nicméně všechny nemají stejnou váhu a některé lze proto úplně vynechat. Díky rozdílným číselníkům inzertních serverů je také mnohdy problém některé parametry správně klasifikovat. Bylo proto zvoleno pouze 9 klíčových vlastností:

- Značka
- Model
- Rok výroby
- Počet najetých km
- Výkon motoru

- Objem motoru
- Typ převodovky
- Typ paliva
- Cena

Ostatní parametry jsou z již zmíněných důvodů při hodnocení zcela ignorovány.

Jelikož rozptýl víceméně všech vlastností vozidla je značný a správně ohodnocených dat je nedostatek, bylo potřeba omezit celkovou variabilitu dat. Pokud by data nebyla omezena, bylo by obtížné neuronovou síť správně naučit.

Pro hodnocení byla proto vybrána pouze vozidla s následujícími omezeními:

- Značky: Audi, Alfa Romeo, BMW, Škoda
- Cena větší než 500 000 Kč
- Maximální stáří 2 roky

Zvolené značky jsou v České republice populární a díky omezením ceny a roku výroby byly odfiltrovány inzeráty, které by bylo obtížné hodnotit.

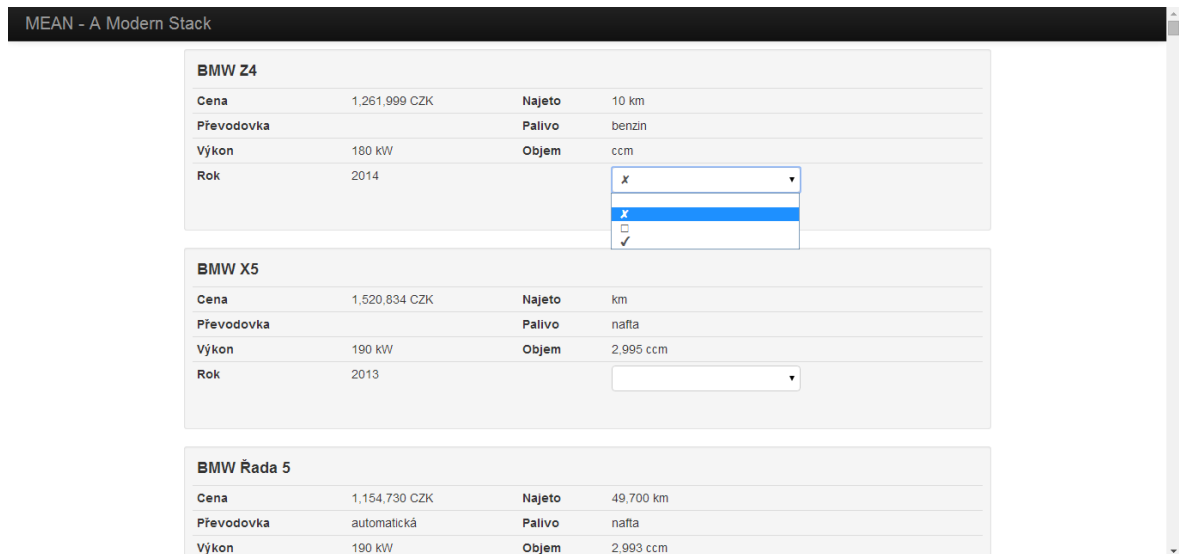
Z cca 2 mil. nasbíraných inzerátů bylo poté náhodně vybráno 200 inzerátů splňující předemšlé omezující podmínky.

4.2.2 Hodnotící webová aplikace

Největším problémem, který při vyhodnocování vybraných automobilových inzerátů nastal, byla neochota prodejců aut. I přesto, že celá myšlenka strojově vyhodnocovat kvalitu inzerátů vzešla z jejich potřeb a přání, nebyli se bohužel schopni odpovědně zapojit do procesu hodnocení a většinu dat proto musel vyhodnotit autor práce. Navzdory mé snaze můžou být proto zdrojová data ohodnocená nepřesně. Chybí mi totiž několikaletá zkušenost, kterou autoprodejci bezpochyby mají.

Webová aplikace je velmi jednoduchá. Jedná se o jednu stránku s výpisem inzerátů. U každého inzerátu může prodejce pomocí selectboxu zvolit, zda se mu líbí - ✓ či ne - ✗ nebo zda se jedná o běžný inzerát - □. Jakmile prodejce provede ohodnocení, známka se odešle na server, kde se uloží a inzerát z výpisu zmizí.

Screenshot aplikace je na obr. 18.



Obrázek 18: Hodnotící webová aplikace

Inzeráty jsou záměrně vypsány bez jakýchkoliv dalších parametrů, zpětných odkazů na původní inzertní server nebo obrázků. Prodejce se tak musí rozhodovat jen na základě stejných informací, jaké bude mít k dispozici neuronová síť.

Aplikace byla napsána tak, aby data mohlo hodnotit více prodejců. Do databáze jsou známky ukládány jako pole JSON objektů mající dvě vlastnosti: ID uživatele a známku.

```
marks: [  
  {  
    "userId": "milan" ,  
    "mark" : "1"  
  },  
  {  
    "userId": "ondra" ,  
    "mark" : "1"  
  }  
];
```

Výsledná známka vozidla je poté průměrem ze všech získaných známek.

Frontend webové aplikace je napsán v javascriptovém frameworku AngularJS. Na backendu běží Nodejs s Express frameworkem a MongoDB. Tato kombinace technologií je označována zkratkou MEAN - dle počátečních písmen jednotlivých názvů.[28]

5 TVORBA NEURONOVÉ SÍTĚ

Pro tvorbu neuronové sítě byla použita javascriptová knihovna brain.js, kterou lze spustit jak na serveru pomocí Node.js, tak i v běžném internetovém prohlížeči.

5.1 Brain.js

Brain.js pracuje s dopřednými neuronovými sítěmi a jako učící algoritmus používá back-propagation.

V kapitole 3.1 bylo zmíněno, že Node.js není vhodný pro výpočetně náročné operace, protože pracuje pouze s jediným vláknem. To je pravda v případě, kdy potřebujeme, aby instance aplikace byla schopná reagovat na další požadavky i během náročného výpočtu. Typicky by se jednalo o případ webového serveru, který musí být schopný zpracovat a vyřídit několik požadavků zároveň a není možné, aby po nějaký čas neodpovídal. Při učení neuronové sítě nás tento problém netrápí. Můžeme vytvořit několik instancí, které nedělají nic jiného, než že zpracovávají algoritmus potřebný pro naučení neuronové sítě.

Je možné polemizovat nad rychlostí javascriptového interpretu při učení sítě. Interpretovaný javascriptový kód bude nepochybně pomalejší než stejný kód například v C++. Nicméně V8 dokáže provést mnohé optimalizace a výsledná ztráta rychlosti poté není až takový problém.

5.1.1 Transformace dat

Brain.js používá jako aktivační funkci logistickou sigmoidu a pracuje pouze s vstupy v intervalu $\langle 0;1 \rangle$, proto je nutné veškerá vstupní data transformovat. U číselných datových typů je taková operace velmi snadná - podělíme je příslušným koeficientem. U textu jako jsou značky, modely, musíme nejdříve provést jejich substituci nějakým číslem. Využijeme toho, že tato data jsou uložena v databázi vždy pod unikátním číselným identifikátorem – ID. Toto ID je celé číslo, které opět podělíme koeficientem, abychom je transformovali do požadovaného intervalu.

V tabulce 3 jsou vypsány značky a jejich číselné identifikátory.

Tabulka 3: Značky a jejich číselné ID

Značka	Audi	Alfa Romeo	BMW	Škoda
Číslo/ID	2	1	5	93

Typ paliva a převodovky je také nutné nahradit číselným ekvivalentem. Vzhledem k tomu, že těchto dat ale není mnoho, je efektivnější je definovat přímo v hashmapě. Její klíče jsou textové řetězce – typy paliva, hodnoty jsou jejich číselné substituenty.

Tabulka 4: Typy paliva a jejich číselné substituce

Typ paliva	Benzín	Nafta	LPG	Elektro	Hybridní	CNG	Ethanol	jiné
Číslo/ID	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8

U převodovek jsou brány v potaz pouze dvě možnosti. Pokud má vozidlo automatickou převodovku, je provedeno nahrazení číslem 1, všechny ostatní možnosti jsou interpretovány jako 0.

Výslednou transformaci poté obstarává funkce *scale* s jedním vstupním parametrem, kterým není nic jiného než samotné vozidlo. Její kód vypadá takto:

```
/**
 * @param {Object} vehicle
 * @returns {*}
 */
scale: function(vehicle) {
  var fuels = {
    'benzín': 0.1,
    'nafta': 0.2,
    'LPG': 0.3,
    'elektro': 0.4,
    'hybridní': 0.5,
    'CNG': 0.6,
    'ethanol': 0.7,
    'jiné': 0.8
  };

  return {
    brand: vehicle.brand._id / 1000,
    model: vehicle.model._id / 10000,
    year: vehicle.date_made_year / 10000,
    km: vehicle.km / 10000000 || 0,
    kw: vehicle.kw / 1000,
    ccm: vehicle.kw / 10000,
    gearbox: vehicle.gearbox == 'automatická' ? 1 : 0,
  };
}
```

```
    fuel: fuels[vehicle.fuel] || 0,  
    price: vehicle.price / 100000000  
  };  
},
```

Brain.js přebírá vstupní data jako JSON objekt s dvěma vlastnostmi (*input*, *output*), které mohou být buď polem nebo opět objektem. Knihovna poté na základě počtu vlastností pro vstup a vlastností pro výstup sama určí odpovídající počet neuronů. Vstupem je vozidlo s transformovanými vlastnostmi, výstupem je aritmetický průměr nasbíraných známek. Transformaci provádí funkce *transformInput*

```
/**  
 *  
 * @param {Object} vehicle  
 * @returns {*}  
 */  
transformInput: function(vehicle) {  
  
  var sum = function(a, b) {  
    return a + b;  
  };  
  
  var getMark = function(mark) {  
    return parseFloat(mark.mark);  
  };  
  
  return {  
    input: Helpers.scale(vehicle),  
    output: {  
      mark: vehicle.marks.map(getMark).reduce(sum) / vehicle.marks.length  
    }  
  }  
}  
};
```

5.2 Topologie sítě

Jelikož počet neuronů ve vstupní a výstupní vrstvě je pevně dán strukturou dat, jediným parametrem, který lze na topologii sítě upravit, je počet skrytých vrstev a jejich neuronů. Otázkou je, jak tento počet zvolit. Neexistuje totiž jednoznačné pravidlo, pomocí kterého bychom jej zjistili a tak je mnohdy nejlepším řešením prostě vytvořit několik sítí s různými variantami skrytých vrstev. Ty potom naučit, otestovat a vybrat tu s nejmenší chybou.

Bylo zvoleno celkem 8 variant, jejichž struktura je popsána v následující tabulce.

Tabulka 5: Varianty sítě

Varianta sítě	Počet skrytých vrstev	Počet neuronů ve skrytých vrstvách
9	1	9
12	1	12
18	1	18
27	1	27
9 - 9	2	9
12 - 12	2	12
18 - 18	2	18
27 - 27	2	27

5.2.1 Výsledná aplikace

Aplikace pomocí, které je možné neuronovou síť ovládat a trénovat byla napsána v Javascriptu jako knihovna pro Node.js. Aplikace se spouští přes příkazový řádek a je ji možné ovládat pomocí několika parametrů.

Parametr **train** spouští samotný proces učení. Pokud není stanoveno jinak, aplikace použije své základní nastavení. Pakliže chceme učení sítě přizpůsobit našim požadavkům, je tak možné učinit pomocí následující sady argumentů:

- -h <číslo> udává počet neuronů ve skryté vrstvě, pokud chceme vytvořit více skrytých vrstev, zadáme parametr -h <číslo> vícekrát
 - Příklad pro vytvoření 2 skrytých vrstev s 6 a 3 neurony: -h 6 -h 3
- -e <číslo> určuje požadovanou chybu učení
 - Příklad pro nastavení chyby 5%: -e 0.05
- -i <číslo> určuje maximální počet iterací
- -l <číslo> nastavuje learning rate
- -f <řetězec> jméno souboru, do kterého se síť po naučení uloží

Během učení jsou na konzoli vypisovány počty iterací a průběžná chyba. Jakmile proces skončí, je vypsána dosažená chyba spolu s počtem iterací a časem potřebným k natrénování sítě.

Následující kód spouští aplikaci pro trénování neuronové sítě s dvěma skrytými vrstvami každá s devíti neurony. Maximální počet iterací je stanoven na 100000, learning rate na 0.3 a požadovaná chyba na 5%. Neuronová síť bude uložena do souboru s názvem 99.json a výpis průběhu učení je z konzole přesměrován do souboru 99.log.

```
node app/run --train -h 9 -h 9 -e 0.05 -l 0.3 -i 100000 -f 99 > 99.log
```

Po celkovém dokončení procesu učení je správná funkčnost sítě ověřena na testovací množině dat. Pro vozidla je vypočtena známka a ta jsou následně vypsána jako pole JSON objektů.

```
{ error: 0.08457136698037446, iterations: 100000 }
train time: 607128ms
[ { url: 'http://www.rajaut.cz/85495-audi-a6-allroad-30-tdi',
  mark: 0.5,
  evaluatedMark: 0.9434402550799584 },
  { url: 'http://www.sauto.cz/osobni/detail/audi/a8/14825288',
  mark: 0.25,
  evaluatedMark: 0.21412140975473043 },
  { url: 'http://www.sauto.cz/osobni/detail/audi/a8/15017895',
  mark: 1,
  evaluatedMark: 0.5879737407081109 },
  { url: 'http://www.sauto.cz/osobni/detail/bmw/rada-3/14997196',
  mark: 0.25,
  evaluatedMark: 0.3658594858825954 },
  { url: 'http://www.sauto.cz/osobni/detail/bmw/rada-5/15178599',
  mark: 0.5,
  evaluatedMark: 0.24910757041484707 },
  { url: 'http://www.cars.cz/inzerce/osobni-auto-0/BMW/X3/X3-2.5-i-Hatchback-13949048234541652.html?c1ksrc=Y2Fyc2hwX2NsaWNrWzhd',
  mark: 0.25,
  evaluatedMark: 0.029789430278072068 },
  { url: 'http://www.sauto.cz/osobni/detail/skoda/superb/15336144',
  mark: 0,
  evaluatedMark: 0.00006064410184827716 },
  .
  .
  .
]
```

5.2.2 Učení sítě

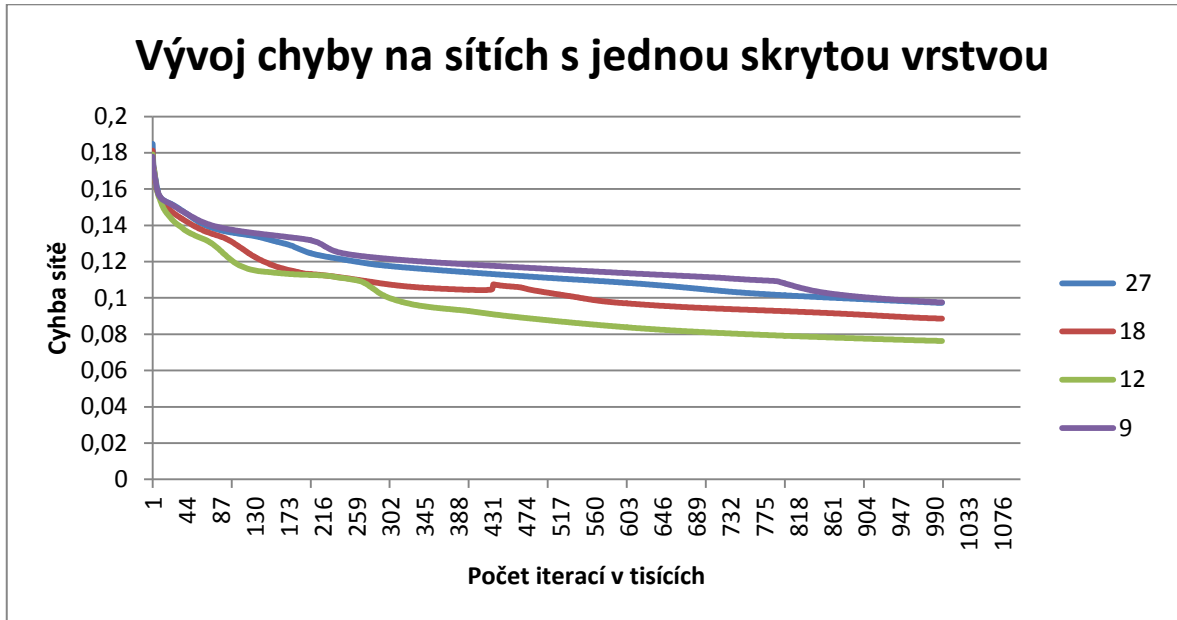
Celková množina ohodnocených zdrojových dat byla rozdělena na dvě části. Trénovací část obsahuje 70% dat a testovací zbylých 30%.

Pro učení sítě bylo zvoleno následující nastavení

- Počet iterací: 100 000

- Chyba sítě: 5%
- Learning rate: 0.3

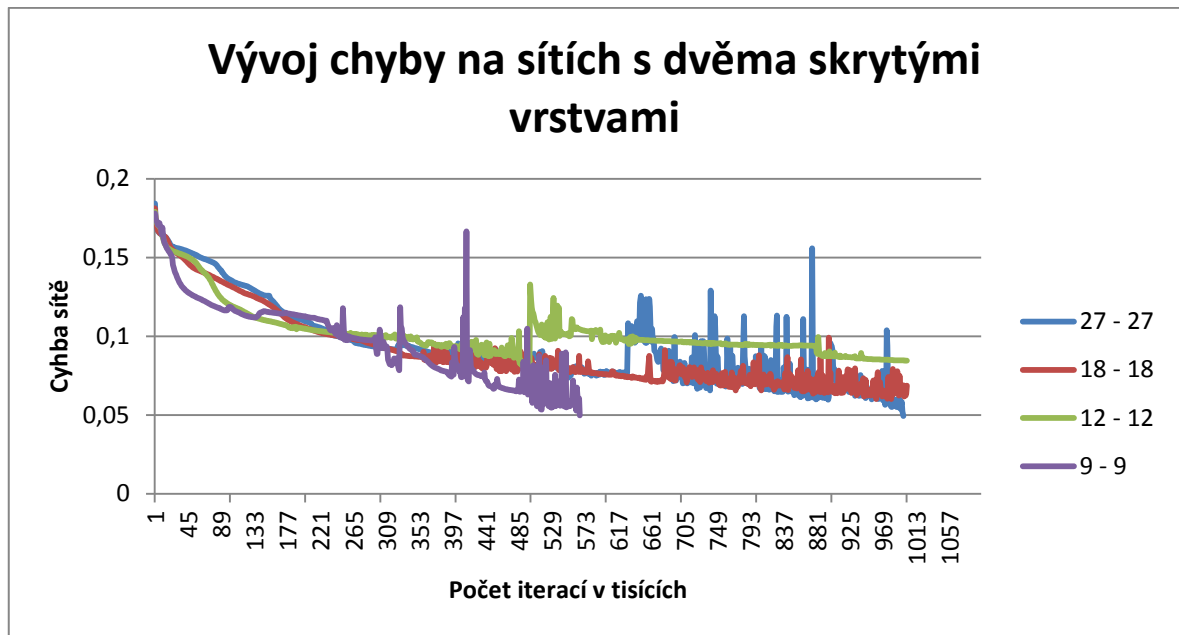
Poté proběhlo naučení pro každou variantu dle tabulky. Vývoj chyby je zobrazen v grafech.



Obrázek 19: Vývoj chyby na sítích s jednou skrytou vrstvou

Jak je z grafu patrné, chyba na sítích s jednou skrytou vrstvou vždy konvergovala k minimu, ale nikdy se nedokázala dostat pod předem stanovenou hranici 5% ani po dokončení stanoveného maximálního počtu iterací. Průběh křivky je ve všech případech plynulý.

Nejlépe si vedla síť s 12 skrytými neurony.



Obrázek 20: Vývoj chyby na sítích s dvěma skrytými vrstvami

Oproti sítím s jednou skrytou vrstvou je zde patrný velký šum při průběhu všech křivek, nicméně všechny opět konvergovali k minimu.

Nejlépe si vedla síť s označením 9 – 9, která se byla schopná naučit pod stanovenou minimální chybu 5% během poloviny z maximálního nastaveného počtu iterací. Pod hranici 5% se dokázala dostat ještě robustní síť 27 – 27. Ostatní sítě skončili během procesu učení vždy nad touto hranicí.

5.3 Ověření výsledků

Správná funkce neuronové sítě byla ověřena na testovacích datech.

V následující tabulce jsou vypsána všechna vozidla z testovací množiny dat a jejich reálné známky. Ke každému vozidlu jsou zobrazeny i vypočítané známky od všech variant neuronových sítí.

Tabulka 6: Známky neuronových sítí pro testovací data

		Známka dle neuronové sítě							
Vůz	Známka	9	12	18	27	9 - 9	12 - 12	18 - 18	27 - 27
1	0,5	0,76	0,70	0,73	0,71	0,88	0,94	0,74	0,60
2	0,25	0,14	0,18	0,27	0,27	0,02	0,21	0,46	0,23

3	1	0,55	0,51	0,54	0,51	0,38	0,59	0,73	0,64
4	0,25	0,44	0,38	0,48	0,47	0,32	0,37	0,48	0,31
5	0,5	0,34	0,26	0,43	0,42	0,27	0,25	0,36	0,15
6	0,25	0,06	0,08	0,05	0,09	0,06	0,03	0,24	0,27
7	0	0,00	0,00	0,00	0,05	0,03	0,00	0,00	0,00
8	1	0,81	0,95	0,91	0,82	1,00	1,00	0,98	0,98
9	0,5	0,33	0,05	0,56	0,39	0,31	0,58	0,53	0,75
10	0	0,21	0,30	0,32	0,24	0,36	0,32	0,40	0,35
11	1	0,81	0,73	0,80	0,99	0,98	0,90	1,00	1,00
12	1	0,94	1,00	1,00	0,92	0,98	1,00	1,00	1,00
13	0,5	0,30	0,20	0,29	0,31	0,18	0,28	0,17	0,18
14	0	0,72	0,88	0,76	0,69	0,29	0,67	0,52	0,52
15	1	0,75	0,93	0,79	0,70	0,50	0,64	0,50	0,50
16	0	0,18	0,09	0,19	0,24	0,13	0,24	0,16	0,10
17	0	0,14	0,05	0,37	0,19	0,25	0,20	0,20	0,23
18	0,5	0,09	0,03	0,26	0,15	0,28	0,19	0,19	0,21
19	0,5	0,24	0,32	0,24	0,26	0,38	0,21	0,28	0,35
20	0,5	0,34	0,56	0,55	0,54	0,66	0,74	0,82	0,54
21	0	0,15	0,24	0,11	0,17	0,33	0,28	0,28	0,25
22	0,5	0,16	0,25	0,12	0,18	0,34	0,29	0,28	0,30
23	0	0,00	0,00	0,05	0,01	0,00	0,00	0,00	0,00
24	1	0,69	0,43	0,62	0,88	1,00	0,91	1,00	0,99
25	0,75	0,37	0,60	0,60	0,58	0,82	0,72	0,97	0,54
26	0,5	0,15	0,36	0,37	0,45	0,46	0,50	0,82	0,53
27	0	0,12	0,05	0,08	0,10	0,07	0,01	0,07	0,11

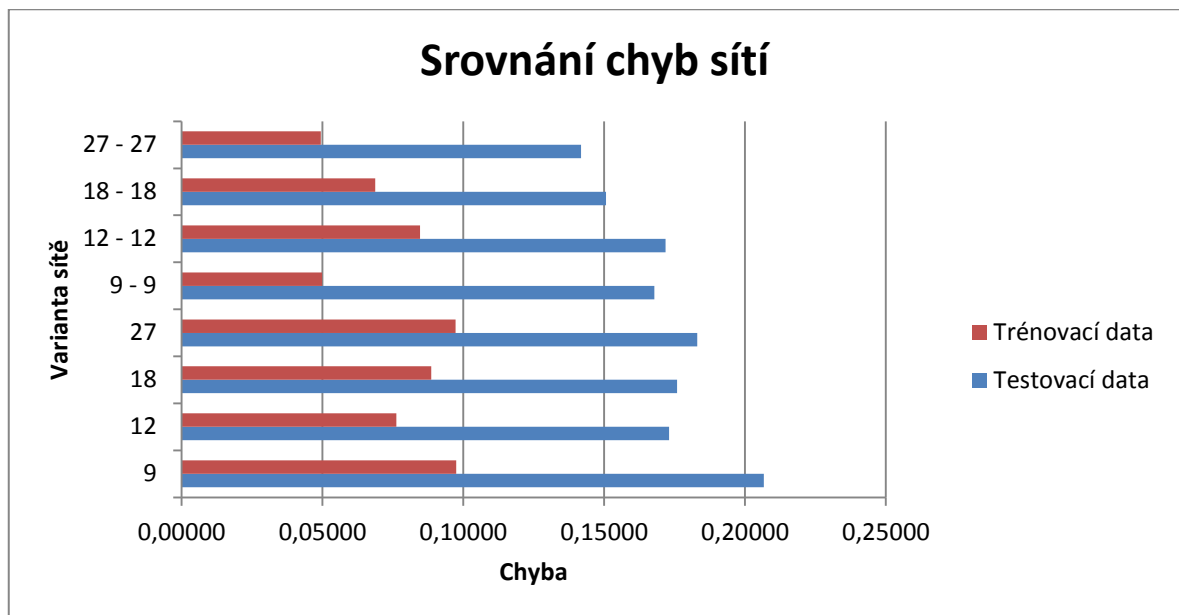
28	1	0,77	0,96	0,91	0,72	1,00	0,81	1,00	0,94
29	0	0,00	0,00	0,06	0,01	0,00	0,01	0,00	0,00
30	0	0,18	0,05	0,03	0,00	0,00	0,22	0,00	0,00
31	0	0,00	0,00	0,01	0,05	0,08	0,09	0,00	0,02
32	0	0,29	0,06	0,08	0,26	0,11	0,08	0,10	0,11
33	0,5	0,47	0,31	0,57	0,52	0,42	0,50	0,48	0,83
34	1	1,00	0,88	0,96	0,97	0,89	1,00	1,00	0,97
35	0,75	0,61	0,79	0,58	0,53	1,00	0,74	0,64	0,80
36	1	0,53	0,92	0,54	0,40	0,92	0,47	0,92	0,96
37	1	0,54	0,50	0,53	0,50	0,38	0,57	0,71	0,61
38	0,5	0,24	0,44	0,42	0,43	0,46	0,48	0,48	0,54
39	0,25	0,11	0,25	0,23	0,01	0,00	0,11	0,20	0,23
40	0	0,26	0,33	0,26	0,28	0,38	0,21	0,28	0,35
41	0	0,19	0,08	0,20	0,25	0,12	0,24	0,16	0,09
42	0,5	0,54	0,50	0,54	0,53	0,56	0,54	0,78	0,69
43	0	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
44	0,25	0,46	0,47	0,47	0,41	0,45	0,47	0,23	0,29

Rozdíly všech vypočtených známek každé varianty sítě byly zprůměrovány a jsou zobrazeny v tabulce.

Tabulka 7: Průměrná chyba sítí na testovacích datech

Varianta sítě	9	12	18	27	9 - 9	12 - 12	18 - 18	27 - 27
Průměrná chyba	0,21	0,17	0,18	0,18	0,17	0,17	0,15	0,14

V grafu je poté přehledně zobrazena dosažená chyba jednotlivých sítí, pro trénovací a testovací množinu dat.



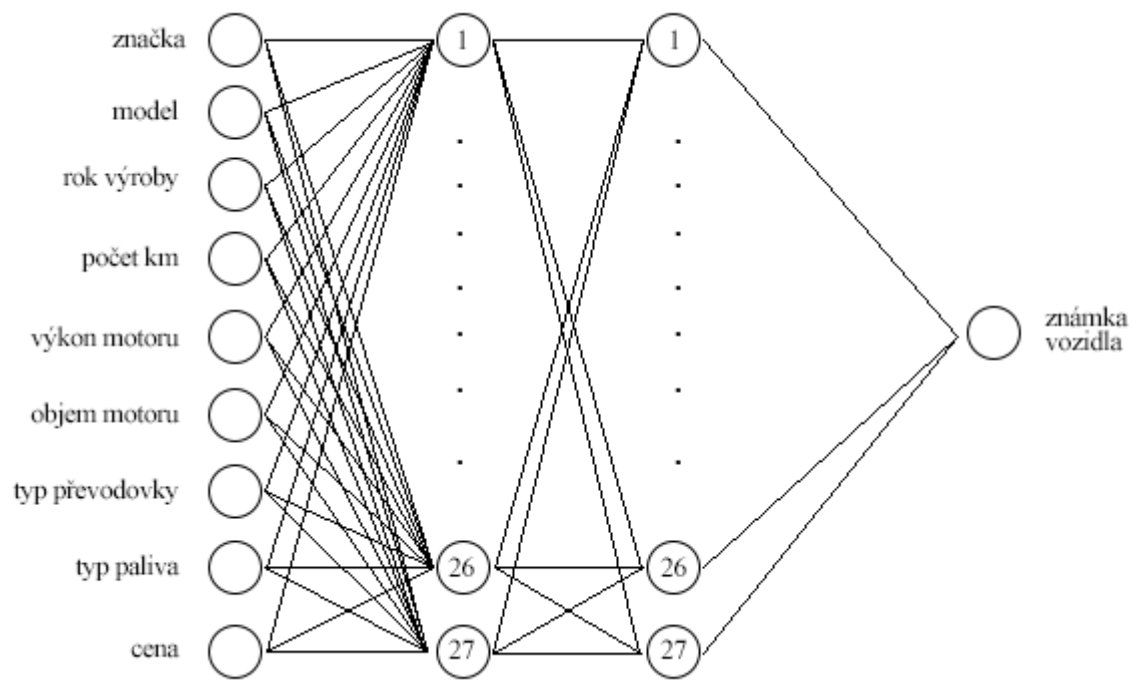
Obrázek 21: Srovnání chyb sítí

Přestože rozdíly nejsou nijak markantní, z výsledků vyplývá, že si při ohodnocování automobilových inzerátů lépe vedou neuronové sítě se dvěma skrytými vrstvami, nejlépe potom síť 27 – 27.

5.4 Neuronová síť

Na základě měření byla vybrána jako nejlepší varianta síť 27 – 27.

Výsledná neuronová síť má tedy 9 neuronů ve vstupní vrstvě, jeden neuron ve výstupní vrstvě a dvě skryté vrstvy, každá s 27 neurony. Její schématický náčrt i spolu se vstupními vlastnostmi je zobrazen na následujícím obrázku.



Obrázek 22: Schéma neuronové sítě 27 - 27

6 PRAKTICKÁ APLIKACE

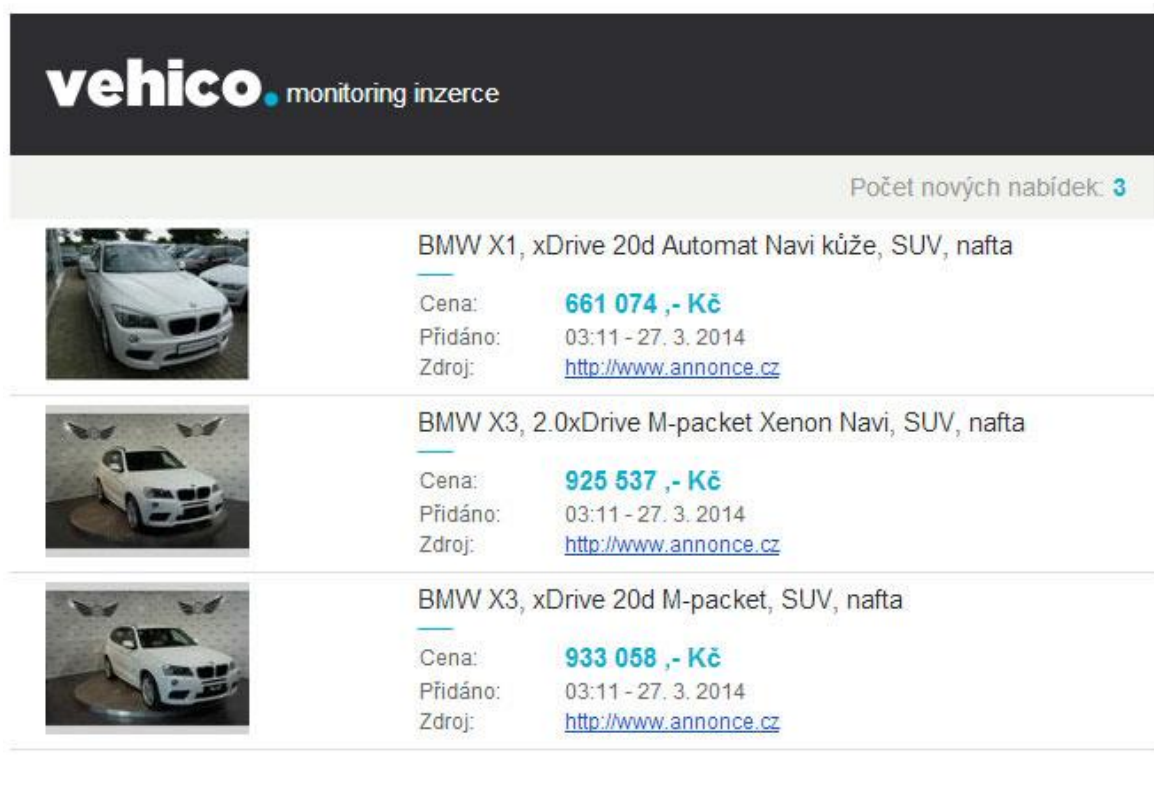
Zadání diplomové práce bylo vytvořeno na základě praktických potřeb autoprodávajících. V této kapitole jsou proto uvedeny některé aplikace neuronové sítě, které by jejich potřeby pomohly řešit.

6.1 Monitoring inzerce




Jednou z možností využití neuronových sítí může být například monitoring inzerce. V současné době monitoring inzerce funguje jako služba pro autoprodávající, kdy jsou emailem upozorňováni na nejnovější inzeráty podle předem stanoveného filtru.

Na pozadí *crawleři* v krátkých časových intervalech opakovaně sbírají inzeráty z internetních serverů a veškerá data ukládají do databáze. Periodicky se poté spouští skript, který z databáze nové inzeráty vybírá a kontroluje, zda některý odpovídá nastaveným filtrům. Pokud ano, je na něj prodejce upozorněn emailem.

Na obr. 23 je zobrazen email s novými nabídkami vozidel BMW.



The image shows an email interface for 'vehico monitoring inzerce'. At the top right, it indicates 'Počet nových nabídek: 3'. Below this, there are three listings for BMW vehicles, each with a small image of the car and its details:

Image	Model	Cena	Přidáno	Zdroj
	BMW X1, xDrive 20d Automat Navi kůže, SUV, nafta	661 074,- Kč	03:11 - 27. 3. 2014	http://www.annonce.cz
	BMW X3, 2.0xDrive M-packet Xenon Navi, SUV, nafta	925 537,- Kč	03:11 - 27. 3. 2014	http://www.annonce.cz
	BMW X3, xDrive 20d M-packet, SUV, nafta	933 058,- Kč	03:11 - 27. 3. 2014	http://www.annonce.cz

Obrázek 23: Monitoring inzerce

Aby se prodejce vyhnul zahlcení informacemi, musí filtr nastavit velmi podrobně. I přesto jsou mu často zasílány inzeráty, které nejsou atraktivní.

Neuronová síť by zde tedy fungovala jako další filtr, který vybere jen ty opravdu zajímavé nabídky. Pokud by se jednalo o velmi výhodnou nabídku, může ji nabídnout i nad rámec základního filtru. Dobrý prodejce si totiž nenechá ujít žádnou zajímavou nabídku.

6.2 Sekce TOP nabídky na inzertních portálech

Inzertní portál SEA24 nabízí vozy italských prodejců pro český trh. Oproti jiným inzertním portálům není jeho hlavním zdrojem příjmu reklama a placené inzeráty, které se zobrazují na úvodní stránce. Portál pouze zobrazuje informace, zdrojem příjmů je dovoz vozů do ČR a jejich následný prodej.

Na základě tohoto modelu je proto výhodné zobrazovat na úvodní straně opravdu ty nejlepší nabídky, tak aby byly co nejvíce na očích a potenciální zákazník by o nich co nejdříve věděl.

SEA24 SDRUŽENÍ EVROPSKÝCH AUTOPRODEJČŮ Nová auta z evropy na 1 klik!







Vyhledat vůz

Fulltext: Např. Audi A8 2013 quattro

Značka: Nerozhoduje Model: Nerozhoduje Palivo: Nerozhoduje Cena do (Kč): Nerozhoduje

Zobrazit Podrobný filtr

TOP nabídka

		Immagine non disponibile	Immagine non disponibile
339 900 CZK	1 579 900 CZK	727 400 CZK	1 329 900 CZK
Fiat Fiorino QJSD 1.3 MJT 75CV R1 (N1)	Ford Ostatní 6.7 TD V8 XLT 4x4 POWER STROKE R1 (R101)	Mitsubishi Ostatní 2.5 DI D 178CV DC INTENSE R1 (R1)	Dodge Ram 1500 4x4 CREW CAR & RAM R1 (R1)
Rok : 2014	Rok : 2015	Rok : 2015	Rok : 2015
Najeto : 0 km	Najeto : 0 km	Najeto : 0 km	Najeto : 0 km
Výkon : 55 kW	Výkon : 287 kW	Výkon : 121 kW	Výkon : 200 kW
			
1 101 700 CZK	412 800 CZK	693 200 CZK	761 700 CZK
Dodge Ram 1500 5.7L RAM BOX RAM R1 (R101) R1 (R101)	Citroën Jumpy 27 1.6 HDI/90 R1 (R1) R1 (R1)	Isuzu Ostatní NKR L35 ADAPTOR NISSAN KM R1 (R1)	Isuzu Ostatní 2.5 Crew Cab Duster 4/7 4WD

Obrázek 24: Screenshot úvodní strany portálu SEA24

Jednou z možností jak toto provést, je ručně inzeráty procházet a vybírat ty nejzajímavější. Na portálu je ale cca 40 000 inzerátů, které se denně aktualizují a není to proto v lidských silách. Je nutné tento proces zautomatizovat, nejlépe pomocí neuronové sítě. Ta může poté pomoci i v boji proti falešným inzerátům, které jsou zobrazeny se smyšlenou cenou.

6.3 Online aukce vozidel

Neuronová síť může sloužit jako pomocník při online aukcích vozidel, které probíhají na webových stránkách. Díky tomu, že knihovna Brain.js umožňuje jednoduše vyexportovat neuronovou síť jako javascriptový kód, je velmi snadné z něj poté vytvořit doplněk do prohlížeče, který by prodejci orientaci v aukční místnosti usnadnil, či zcela automatizoval.

6.4 Další aplikace

Předchozí ukázky demonstrují praktické využití neuronových sítí pouze z oblasti automobilového průmyslu. Nicméně možnosti jejich použití jsou velmi široké, prakticky z každého odvětví lidské činnosti.

Pokud pomineme jejich užití už dříve popsané v teoretické části diplomové práce a soustředíme se na webové či mobilní aplikace, neuronová síť může být dobrým pomocníkem při hledání nového bytu, dovolené či třeba pračky.

ZÁVĚR

Hlavním cílem diplomové práce bylo přijít na způsob, jakým bude možné automaticky vyhodnocovat kvalitu automobilových inzerátů pomocí neuronových sítí.

Teoretická část se nejprve zabývá poměrně podrobným popisem biologického neuronu. Je totiž nezbytné poskytnout čtenáři základní informace, na kterých stojí celý koncept umělých neuronových sítí. Dále je rozebírána jejich historie, od prvních krůčků v polovině 20. století až po současnost, kdy se vědci snaží nasimulovat celý lidský mozek s jeho miliardami neuronů a synapsí. V neposlední řadě je v teoretické části popsáno základní členění neuronových sítí s popisem učících algoritmů. Pozornost je soustředěna na neuronové sítě s učitelem, s nimiž se poté pracuje i v praktické části.

Praktická část se již zabývá konkrétním problémem a to jak lze strojově zpracovat a vyhodnotit zajímavost automobilových inzerátů pomocí neuronové sítě. Operuje se s termínem „dobrá koupě“.

V první části jsou popsány nástroje a techniky, díky kterým lze automobilové inzeráty z webových stránek inzertních portálů získat. Poté je popsán proces, jakým byla data hodnocena. Jako omezující prvek jsou označeni autoprodávající, kteří nebyli schopni, i přes jejich příslib, ohodnotit automobilové inzeráty. Získání označených dat pro sítě, které potřebují ke svému naučení učitele, je obecně problémem. Neoznačených dat je totiž v reálných situacích daleko více.

Poté bylo přistoupeno k realizaci samotné neuronové sítě. Proběhlo několik měření s různými variantami skrytých vrstev a počtem jejich neuronů. Ve výsledku se jako nejlepší volba ukázala neuronová síť s dvěma skrytými vrstvami s 27 neurony v každé skryté vrstvě. Síť dosahovala průměrné 14% chyby na testovací množině dat.

V poslední části se diplomová práce zabývá praktickou aplikací neuronových sítí v oblasti automobilové inzerce. Popisuje situace, se kterými se prodejci potýkají každý den a nabízí řešení.

Cíl diplomové práce lze tedy považovat za splněný. Bylo prokázáno, že neuronové sítě k vyhodnocování kvality automobilových inzerátů použít lze. Je ale nutné zmínit, že pro jejich praktickou aplikaci by síť měla mít větší počet vstupních parametrů. Vzhledem k tomu, že prodejci nebyli schopni pomoci s hodnocením inzerátů, musely být přijaty ome-

zující podmínky pro všechny inzeráty. Neuronová síť se tedy naučila hodnotit inzeráty jen na této omezené množině dat. V praxi by nejspíše neobstála.

Pro úspěšnou aplikaci neuronových sítí do každodenního života prodejců aut bude nutné nejdříve obstarat dostatek kvalitně ohodnocených dat. Jestli se tak podaří, zůstává otázkou.

SEZNAM POUŽITÉ LITERATURY

- [1] OLAH, Christopher. *Neural Networks, Manifolds, and Topology*. [online]. [cit. 2014-05-14]. Dostupné z: <http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/>
- [2] CHALUPNÍK, Vitalij. *Biologicky inspirované algoritmy*. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.root.cz/clanky/biologicke-algoritmy-4-neuronove-site/>
- [3] *Úvod do neuronových sítí*. [online]. [cit. 2014-05-14]. Dostupné z: http://www.statsoft.cz/file1/PDF/newsletter/2013_02_05_StatSoft_Neuronove_site_linky.pdf
- [4] BISKUP, Roman. *Možnosti neuronových sítí*. Praha, 2009. Disertační práce. Česká zemědělská Univerzita v Praze.
- [5] HALICI, Ugur. *ARTIFICIAL NEURAL NETWORKS: From Biological to Artificial Neuron Mode* [online]. Ankara, 2004 [cit. 2014-05-14].
- [6] *History and Development of Neural Networks*. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.neuralnetworksolutions.com/nn/intro2.php>
- [7] *History Of Neural Networks*. [online]. [cit. 2014-05-14]. Dostupné z: <http://scholar.lib.vt.edu/theses/available/etd-111597-81423/unrestricted/chap2.pdf>
- [8] B. DEMUTH, Howard, Martin T. HAGAN a Mark BEALE. *Neural Network Design*. Bei jing, 2002. ISBN 71-111-0841-8.
- [9] ZELINKA, Ivan. *Umělá inteligence I: Neuronové sítě a genetické algoritmy*. 1. vyd. Brno: VUT v Brně, 1998, 126 s. ISBN 80-214-1163-5.
- [10] BENGIO, Yoshua. *Learning Deep Architectures for AI*. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.iro.umontreal.ca/~bengioy/papers/ftml.pdf>
- [11] *Introduction to Deep Learning Algorithms* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.iro.umontreal.ca/~pift6266/H10/notes/deepintro.html>
- [12] *Deep Learning for Data Scientist* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.slideshare.net/andrewgardner5811/deep-learning-for-data-scientists-dsatl-talk-alpharetta-20140108>
- [13] *Deep Learning and Applications in Neural Networks* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.slideshare.net/hammawan/deep-neural-networks>

- [14] *Neural Network Structures* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.ieee.cz/knihovna/Zhang/Zhang100-ch03.pdf>
- [15] *Backpropagation Algorithm* [online]. [cit. 2014-05-14]. Dostupné z: http://ufldl.stanford.edu/wiki/index.php/Backpropagation_Algorithm
- [16] YU, Hao a Bogdan M. WILAMOWSK. Levenberg–Marquardt Training. [online]. 2010 [cit. 2014-05-14]. Dostupné z: http://www.eng.auburn.edu/~wilambm/pap/2011/K10149_C012.pdf
- [17] STŘELEČEK, Martin. *Využití metody nelineárních nejmenších čtverců pro rekonstrukci přechodové charakteristiky* [online]. Plzeň [cit. 2014-05-14]. Dostupné z: http://strela.wz.cz/Clanky/StepRecon_CZ.pdf
- [18] VONDRÁK, Ivo. *Umělá inteligence a neuronové sítě*. Ostrava: VŠB, 1995, 139 s. ISBN 80-707-8259-5.
- [19] MALÝ, Martin. *Node.js – s JavaScriptem na server* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.zdrojak.cz/clanky/node-js-s-javascriptem-na-server/>
- [20] NEŠETŘIL, Jakub. *JavaScript na serveru: Začínáme s Node.js* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.zdrojak.cz/clanky/javascript-na-serveru-zaciname-s-node-js/>
- [21] ŽÁČEK, Viktor. *Kohonenova samoorganizační mapa*. Brno, 2012. Diplomová práce. Vysoké učení technické v Brně.
- [22] *Projects, Applications, and Companies Using Node*. [online]. [cit. 2014-05-14]. Dostupné z: <https://github.com/joyent/node/wiki/Projects,-Applications,-and-Companies-Using-Node>
- [23] *Introduction to MongoDB*. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.mongodb.org/about/introduction/>
- [24] *SQL to MongoDB Mapping Chart*. [online]. [cit. 2014-05-14]. Dostupné z: <http://docs.mongodb.org/manual/reference/sql-comparison/>
- [25] *JavaScript na serveru: MongoDB, Mongoose a AngularJS*. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.zdrojak.cz/clanky/javascript-na-serveru-mongodb-mongoose-angularjs/>
- [26] ŠNOREK, Miroslav. *Neuronové sítě a neuropočítače*. Vyd. 1. Praha: ČVUT, 1996, 124 s. ISBN 80-010-1455-X.

- [27] PANT, Gautam, Padmini SRINIVASAN a Filippo MENCZER. Crawling the Web. [online]. [cit. 2014-05-14]. Dostupné z: http://pdf.aminer.org/000/003/078/crawling_the_web.pdf
- [28] MEAN. [online]. [cit. 2014-05-14]. Dostupné z: <http://mean.io/#/>
- [29] Data Mining. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.laits.utexas.edu/~anorman/BUS.FOR/course.mat/Alex/>
- [30] NoSQL Databases Explained. [online]. [cit. 2014-05-14]. Dostupné z: <http://www.mongodb.com/nosql-explained>
- [31] A. BULLINARIA, John. The Generalized Delta Rule and Practical Considerations. In: [online]. [cit. 2014-05-14]. Dostupné z: <http://www.cs.bham.ac.uk/~jxb/INC/16.pdf>
- [32] Training Hidden Units: The Generalized Delta Rule. In: [online]. [cit. 2014-05-14]. Dostupné z: <http://www.stanford.edu/group/pdplab/originalpdphandbook/Chapter%205.pdf>
- [33] *Neuron Basics* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.mindcreators.com/neuronbasics.htm>
- [34] ŠŤASTNÝ, Petr. *Rozpoznávání objektů pomocí neuronových sítí*. Brno, 2014. Diplomová práce. Masarykova univerzita Fakulta informatiky.
- [35] ROUBAL, Michal. *Neuronové sítě a jejich aplikace*. Olomouc, 2012. Diplomová práce. Univerzita Palackého v Olomouci.
- [36] *What is a proxy server?* [online]. [cit. 2014-05-14]. Dostupné z: <http://www.publicproxyservers.com/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

I/O	Input/Output.
ID	Jednoznačný indentifikátor.
HTML	HyperText Markup Language.
URL	Uniform Resource Locator
API	Application Programming Interface
DOM	Document Object Model
JSON	JavaScript Object Notation
XML	Extensible Markup Language
SQL	Structured Query Language
ACID	Atomicity, Consistency, Isolation, Durability
VPN	Virtual Private Network
IP	Internet protocol
UTF	UCS Transformation Format
BSON	Binary JSON
NoSQL	Not Only SQL
TCP	Transmission Control Protocol
npm	Node Packaged Modules

SEZNAM OBRÁZKŮ

Obrázek 1: Nervová buňka[33].....	14
Obrázek 2: Detail synaptické membrány [5]	15
Obrázek 3: Šíření akčního potenciálu v čase [5]	16
Obrázek 4: Umělý model neuronu.....	18
Obrázek 5: Aktivační funkce [4]	19
Obrázek 6: Znáznornění označených dat.....	21
Obrázek 7: Znáznornění neoznačených dat.....	22
Obrázek 8: Ilustrace možných stavů po naučení sítě [35]	23
Obrázek 9: Schéma dopředné neuronové sítě se zpětným šířením chyby	27
Obrázek 10: Lokální a globální minimum.....	28
Obrázek 11: Hopfieldova síť [34].....	30
Obrázek 12: Kohonenovy mapy [34].....	31
Obrázek 13: Rekurentní neuronová síť [18]	32
Obrázek 14: Ukázkové relační databázové schéma.....	38
Obrázek 15: Znáznornění komunikace skrze proxy server [36].....	41
Obrázek 16: Zobrazení ID inzerátu v adresním řádku prohlížeče	41
Obrázek 17: Zdrojový kód detailu inzerátu	43
Obrázek 18: Hodnotící webová aplikace	46
Obrázek 19: Vývoj chyby na sítích s jednou skrytou vrstvou	52
Obrázek 20: Vývoj chyby na sítích s dvěma skrytými vrstvami	53
Obrázek 21: Srovnání chyb sítí.....	56
Obrázek 22: Schéma neuronové sítě 27 - 27	57
Obrázek 23: Monitoring inzerce	58
Obrázek 24: Screenshot úvodní strany portálu SEA24	60

SEZNAM TABULEK

Tabulka 1: Srovnání klíčových funkcí počítače a nervové soustavy	12
Tabulka 2: Porovnání SQL dotazů s MongoDB dotazy	37
Tabulka 3: Značky a jejich číselné ID	48
Tabulka 4: Typy paliva a jejich číselné substituce	48
Tabulka 5: Varianty sítě.....	50
Tabulka 6: Znamky neuronových sítí pro testovací data.....	53
Tabulka 7: Průměrná chyba sítí na testovacích datech.....	55

SEZNAM PŘÍLOH

P I CD-ROM s příloženými zdrojovými kódy a diplomovou prací.