UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



EXPERIMENTY S NEURÓNOVÝMI SIEŤAMI

DIPLOMOVÁ PRÁCA

2019 Bc. Miloš Kútny

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

EXPERIMENTY S NEURÓNOVÝMI SIEŤAMI

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomicko-finančná matematika a modelovanie
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	Doc. RNDr. Mária Trnovská, PhD.

Bratislava 2019 **Bc. Miloš Kútny**





Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Miloš Kútny
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: aplikovaná matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Experimenty s neurónovými sieťami
Experiments with neural networks

Anotácia: Neurónové siete patria k najvýkonnejším učiacim sa algoritmom, a ich vývoj bol inšpirovaný fungovaním ľudského mozgu. Sú vhodné na riešenie veľkorozmerných úloh z oblastí ako spracovanie signálov, či rozpoznávanie reči a ich vývoj neustále prebieha. Proces natrénovania neurónovej siete je iteratívny, často založený na metóde pokus-omyl, pri ktorej sa pracuje z rôznymi nastaveniami funkcií, či parametrov úlohy. Práca je zameraná na skúmanie problematiky neurónových sietí, vhodnosti použitia alternatívnych optimalizačných metód a experimentovanie s nastaveniami pre daný problém.

Vedúci: doc. RNDr. Mária Trnovská, PhD.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, DrSc.

Dátum zadania: 24.01.2018

Dátum schválenia: 25.01.2018 prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie

Touto cestou sa chcem poďakovať svojej vedúcej diplomovej práce doc. RNDr. Márii Trnovskej PhD. za ochotu, odborné rady a podnetné pripomienky, ktoré mi pomohli pri písaní tejto práce.

Abstrakt

Neurónové siete sa dostali len nedávno do povedomia ako najvýkonnejšie učiace sa algoritmy. Ich myšlienka je založená na princípe fungovania ľudského mozgu a svoje uplatnenie si našli už v mnohých odvetviach. Táto diplomová práca je zameraná na skúmanie problematiky neurónových sietí, vybudovanie vlastnej neurónovej siete a experimentovanie na rozličných reálnych dátach, akými sú napríklad obrázky psov a mačiek alebo číslic. Táto práca pozostáva z 3 kapitol. Prvá časť práce je zameraná na priblíženie si problematiky neurónových sietí, definovanie základných pojmov a vlastností neurónovej siete. Druhá časť práce sa bližšie zaoberá logistickou regresiou, porovnáva ju so základnou neurónovou sieťou a popisuje ich podobnosti. Zvyšok kapitoly predstavuje nami vytvorenú, základnú neurónovú sieť a jej aplikáciu na reálnych dátach. Tretia kapitola sa zaoberá zložitejšou neurónovou sieťou a rôznymi algoritmami zlepšujúcimi fungovanie a úspešnosť klasifikácie neurónovej siete. Záver práce je venovaný experimentovaniu s rôznymi nastaveniami neurónovej siete a porovnaním dosiahnutých výsledkov.

Kľúčové slová: neurón, neurónová sieť, logistická regresia, nákladová funkcia, aktivačná funkcia.

Abstract

Neural networks are relatively new and powerful learning algorithms. The idea is based on the functioning of the human brain and they have found their application in many areas. This master's thesis is focused on the research of neural networks, building our own neural network and experimentation on various real data such aa dog and cat pictures or numbers. This thesis consists of 3 chapters. The first part of the thesis is focused on approaching neural networks, defining basic concepts and characteristics of neural networks. The second part deals with logistic regression, compares it with the basic neural network and describes their similarities. The rest of the chapter is focused on building our own neural network and its application on real data. Third chapter presents more complicated neural network and various algorithms which improve network's classification accuracy. Last part of thesis presents experimenting with parameters of the network and shows results of this experimenting.

Key words: neuron, neural network, logistic regression, cost function, activation function.

Obsah

Úvod **8**

1 Úvod do neurónových sietí **9**

1.1 Základná myšlienka neurónových sietí 9

1.2 Biologická motivácia a spojitosť 9

1.3 Typy aktivačných funkcií 10

1.3.1 Binárna funkcia 11

1.3.2 Sigmoid funkcia 14

1.3.3 Hyperbolický tangens 16

1.3.4 ReLu 17

1.4 Porovnanie aktivačných funkcií 17

1.5 Architektúra umelých neurónových sietí 18

2 Budovanie jednoduchej neurónovej siete **20**

2.1 Binárna klasifikácia 20

2.2 Logistická regresia 21

2.3 Nákladová funkcia 22

2.4 Gradientná metóda 27

2.4.1 Stochastická gradientná metóda 28

2.5 Spustenie jednoduchej neurónovej siete 29

2.6 Aplikácia jednoduchej neurónovej siete na binárnu sadu dát 30

2.7 Aplikácia jednoduchej neurónovej siete na sadu dát s viacerými triedami 34

3 Zložitejšia neurónová sieť **38**

3.1 Spätná propagácia 38

3.1.1 Špeciálny tvar rovníc spätnej propagácie 41
3.2 Vylepšenie neurónovej siete 42
3.2.1 Úvodná voľba parametrov 42
3.2.2 Regularizácia 43
3.3 Spustenie zložitej neurónovej siete 45
3.4 Aplikácia zložitej neurónovej siete na binárnu sadu dát 46
3.5 Aplikácia zložitej neurónovej siete na sadu dát s viacerými triedami . . 48

Záver **53**

Zoznam použitej literatúry **55**

Úvod

Myšlienka, nápad, idea. Všetky vznikajú v najzáhadnejšej časti ľudského tela, v mozgu. Už od nepamäti vedcov fascinoval ľudský mozog a jeho fungovanie. Jedným z hlavných dôvodov bola snaha vytvoriť prístroj alebo program, ktorý by sa správal ako ľudský mozog, takzvanú umelú inteligenciu. Jedným z najznámejších prístupov, ktorý sa snaží napodobniť správanie sa ľudského mozgu, je umelá neurónová sieť.

Cieľom tejto práce je v krátkosti uviesť čitateľov do tematiky neurónových sietí, vysvetliť princíp ich fungovania a následne zostrojiť vlastnú neurónovú sieť, otestovať ju na vybraných dátových sadách a experimentovať s rôznymi nastaveniami siete. Prvá kapitola obsahuje informácie o súvislostiach medzi skutočnými neurónovými sieťami, nachádzajúcimi sa v mozgu a umelými neurónovými sieťami [7]. Následne sa v tejto kapitole rozoberajú základne typy neurónov a ich vzájomné vzťahy. Ďalej je v nej uvedená architektúra umelej neurónovej siete a jej rôzne typy.

Druhá kapitola je venovaná samotnému budovaniu jednoduchej umelej neurónovej siete. Prvá časť tejto kapitoly sa zaoberá logistickou regresiou [1, 13] a jej súvisom s neurónovými sieťami. V tejto časti kapitoly sa čitateľ takisto oboznámi s nákladovou funkciou, ktorá reprezentuje kvalitu neurónovej siete. Druhá časť kapitoly je venovaná spôsobu, ako zlepšovať neurónovú sieť pomocou klasickej gradientnej metódy[15] a jej stochastickou verziou [8]. Zvyšok kapitoly pozostáva z budovania jednoduchej neurónovej siete pomocou nadobudnutých poznatkov a jej aplikácie na zvolené dátové sady.

Tretia kapitola sa zaoberá zložitejšími neurónovými sieťami. Prvá časť je venovaná teórii spätnej propagácie, ktorá slúži na rýchlejšie zlepšovanie sa neurónovej siete [10]. Následne sa v kapitole venujeme rôznym spôsobom, ako vylepšiť neurónovú sieť v zmysle dosiahnutia vyššej úspešnosti klasifikácie. Posledná časť kapitoly je praktická a zaoberá sa aplikáciou zložitej neurónovej siete na rôzne sady dát a viaceré druhy experimentov.

1 Úvod do neurónových sietí

1.1 Základná myšlienka neurónových sietí

Neurónové siete majú v dnešnej dobe široké využitie. Používajú sa hlavne pri problémoch, pri ktorých iné klasifikačné programy nedosahujú dostatočne dobré výsledky. Od rozpoznania ručne napísaných číslic, cez identifikáciu rôznych objektov na obrázkoch, či identifikáciu ľudského hlasu až po porážanie najlepších ľudských hráčov v hrách ako šach, go alebo dokonca Starcraft 2. Neurónové siete našli obrovské uplatnenie v bežnom živote. Dôvodom tohto úspechu je iný prístup k riešeniu problémov. Myšlienka neurónových sietí je založená na riešení problému cestou učenia sa. Základom je obrovské množstvo dát, ktoré sa zvyčajne rozdelí na 2 sady, trénovaciu a testovaciu. Na prvej sade, teda trénovacej, sa neurónová sieť učí a druhá sada dát, testovacia, slúži na kontrolu, či je sieť dostatočne dobrá.

Odkiaľ ale vôbec pochádza názov neurónová sieť a aký to má súvis s biologickými neurónmi nachádzajúcimi sa napríklad aj v ľudskom mozgu?

1.2 Biologická motivácia a spojitosť

Základnou výpočtovou jednotkou mozgu je neurón. Približne 86 miliard neurónov sa nachádza v ľudskom nervovom systéme a sú poprepájané približne $10^{14} - 10^{15}$ synapsiami. Neurón sa skladá z tela bunky (neurocytu) a výbežkov (dendritov a neuritov). Dendrity prijímajú vstupnú informáciu (nervový vzruch), neurocyt informáciu vyhodnotí a neurity vedú informáciu od tela. Kontakty medzi neurónmi navzájom a s inými bunkami zabezpečujú synapsy. Podrobný výklad možno nájsť v [7].

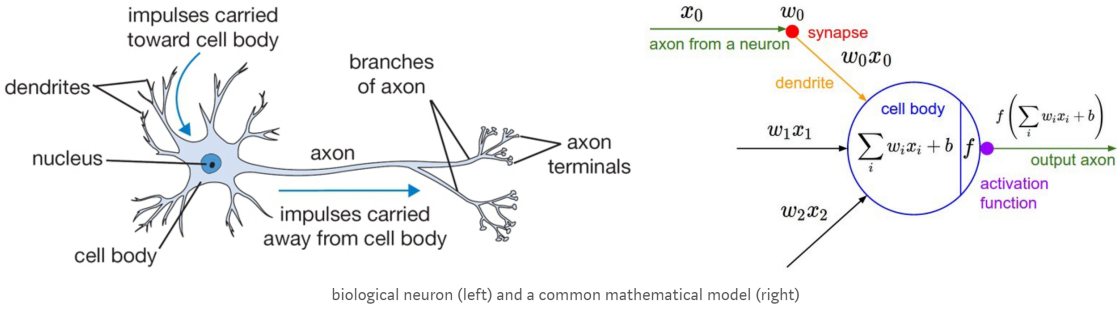
Umelý neurón funguje podobne. Prijíma informáciu z iných neurónov alebo vstupov, vyhodnotí ju a pošle ďalej. Matematicky je možné umelý neurón zapísať nasledovne:

$$y = f((w^\top x) + b),$$

pričom

- n je počet vstupov do neurónu,
- $x \in \mathrm{R}^n$ je vektor vstupov neurónu,
- $w \in \mathrm{R}^n$ je vektor váh,
- $b \in \mathrm{R}$ je bias,
- $f(x)$ je aktivačná funkcia,
- $y \in \mathrm{R}$ je výstup neurónu.

Pod váhami sa dá chápať, ako je ktorý vstup do neurónu dôležitý. Aktivačná funkcia určuje, ako bude daný neurón vyhodnotený. Parameter b sa v anglickej literatúre nazýva *bias*. Dá sa chápať ako istý druh hranice, ktorý musí byť prekonaný aby sa neurón aktivoval. Na Obr. 1 je možné vidieť porovnanie skutočného a umelého neurónu.

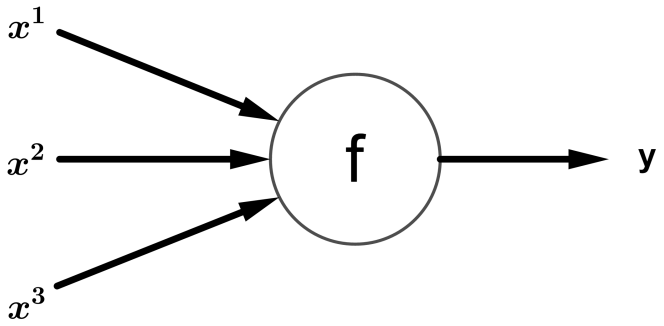


Obr. 1: Porovnanie skutočného (vľavo) a umelého (vpravo) neurónu. Zdroj: [7].

Princíp umelého neurónu je zobrazený aj na Obr. 2.

1.3 Typy aktivačných funkcií

Medzi základné typy aktivačných funkcií neurónov patria napríklad:



Obr. 2: Grafické znázornenie princípu umelého neurónu. Zdroj: vlastné spracovanie.

1. binárna funkcia,
2. sigmoid funkcia,
3. hyperbolický tangens,
4. ReLu funkcia.

V nasledujúcich podkapitolách sú podrobne definované rôzne druhy neurónov využívajúce vyššie vymenované aktivačné funkcie.

1.3.1 Binárna funkcia

Neurón s binárnou aktivačnou funkciou sa nazýva perceptrón. Tento druh neurónov bol predstavený v 50. a 60. rokoch 20. storočia Frankom Rosenblattom, ktorý sa inšpiroval prácami Warrena McCullocha a Waltera Pittsa [8].

Princíp perceptrónu spočíva v prijatí niekoľkých vstupov $x^1, x^2, \ldots, x^n \in \mathrm{R}$ a výsledkom je jeden binárny výstup y. Na zistenie výstupu má každý vstup priradenú váhu w, reálne číslo, ktoré reprezentuje jeho dôležitosť. Výstup neurónu je daný nasledovným vzťahom:

$$y = \begin{cases} 1, & ak\ w^\top x > \text{excitačná hranica}, \\ 0 & inak. \end{cases}$$

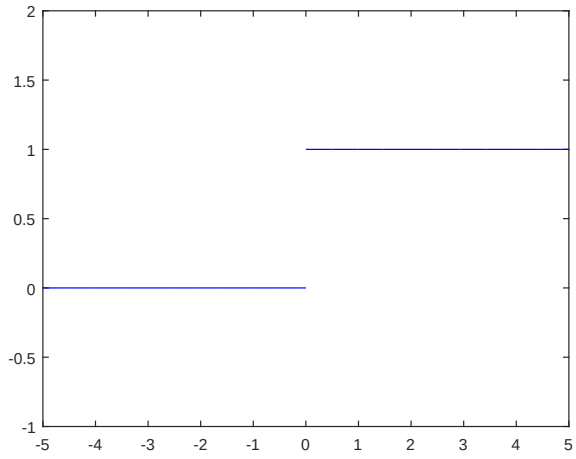
Excitačná hranica predstavuje zvolenú hodnotu, pri ktorej sa určuje, či sa daný neurón aktivuje (excituje) alebo nie, teda či výstup neurónu bude 1 alebo 0. Zmenou váh a excitačnej hranice dostávame iný výstup pre daný neurón. Zvyšovanie excitačnej hranice vedie k častejšiemu výstupu 0 a znižovanie k častejšiemu výstupu 1.

Označením $b = -$ excitačná hranica sa zápis výstupu neurónu zjednoduší nasledovne:

$$y = \begin{cases} 1, & ak\, w^\top x + b > 0, \\ 0 & inak. \end{cases}$$

.

V súčastnosti sa však perceptróny príliš nevyužívajú, čoho dôvodom je, že už aj pri veľmi malej zmene niektorej z váh môže výraz $w^\top x + b$ zmeniť hodnotu zo zápornej na kladnú alebo naopak a tým zmeniť výstup neurónu. Takéto správanie neurónu však nie je žiadúce. Na nasledujúcom obrázku Obr. 3 je znázornená binárna funkcia.

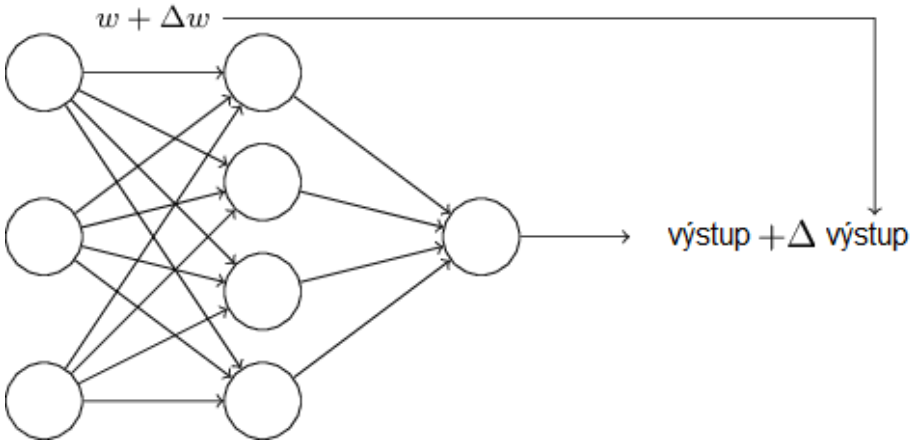


Obr. 3: Grafické znázornenie binárnej funkcie. Zdroj: vlastné spracovanie.

Hlavným cieľom tejto práce je vytvorenie učiacieho sa algoritmu, ktorý funguje

na princípe neurónových sietí. Predstavme si, že máme sieť perceptrónov, ktorú by sme chceli naučiť riešiť určitý problém. Cieľom by bolo, aby si sieť nastavila váhy a bias tak, aby bol výstup správne klasifikovaný. K pochopeniu toho, ako učenie funguje, je vhodné spraviť malú zmenu niektorej váhy alebo excitačnej hranice. Chceme aby táto malá zmena spôsobila iba malú zmenu na výstupe zo siete. Takáto vlastnosť bude kľúčová pri učiacich sa algoritmoch.

Algoritmy s touto vlastnosťou by sme mohli modifikovať tak, aby sa správali podľa našich predstáv. Predstavme si napríklad, že sieť nesprávne identifikovala výstup. Ak by sme prišli na to, ktorou malou zmenou vo váhach alebo v excitačnej hranici sa neurónová sieť priblíži k správnej klasifikácii výstupu tak opakovaním menenia týchto parametrov, by sme mohli dostávať lepšie výsledky. Tým pádom by sa sieť učila. Vhodný algoritmus, pri ktorom malá zmena váh a biasu spôsobí malú zmenu vo výstupe, je zobrazený na Obr. 4.



Obr. 4: Grafické znázornenie vhodného algoritmu. Zdroj: [8].

Avšak takéto správanie neurónovej siete nie je možné pri perceptrónoch, pretože pri malej zmene parametrov sa môže úplne zmeniť výstup perceptrónu, nakoľko je tento výstup binárny. Pri zložitejšej sieti, pozostávajúcej z viacerých vrstiev by mohla aj malá zmena niektorého z parametrov úplne zmeniť správanie neurónovej siete. Vyššie definované správanie perceptrónu by nám neumožnilo sledovať, či malé zmeny vedú k lepším výsledkom. Tento problém sa dá obísť pomocou sigmoid neurónov.

1.3.2 Sigmoid funkcia

Sigmoid funkciu, využívanú v sigmoid neurónoch si definujeme nasledovne.

Definícia 1.1. Nech $z \in \mathrm{R}$. Potom definujeme sidmoid funkciu ako:

$$\sigma(z) = \frac{e^x}{1 + e^z},$$

kde $\sigma : \mathrm{R} \to \mathrm{R}$.

Sigmoid je podobný perceptrónu, teda obsahuje vstupy $x^1, x^2, \ldots, x^n$, váhy, excitačnú hranicu a výstup, ktorý nadobúda hodnotu:

$$y = \sigma(w^\top x + b).$$

Hoci sa to na prvý pohľad nemusí zdať, perceptrón a sigmoid sú si dosť podobné. Ak

- $z = w^\top x + b \gg 0$, tak $\sigma(z) \approx 1$,

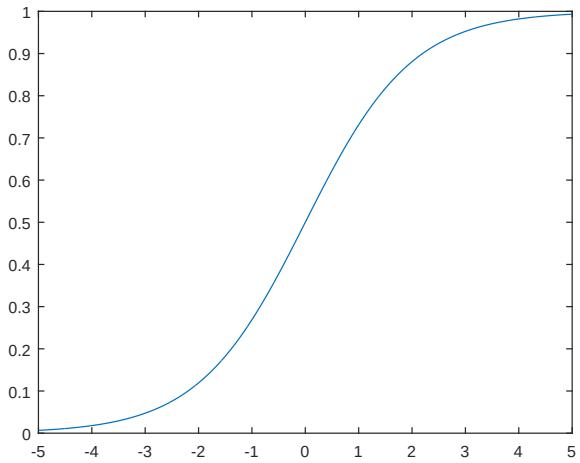
naopak, ak

- $z = w^\top x + b \ll 0$, potom $e^{-z} \longrightarrow \infty$ a $\sigma(z) \approx 0$.

Takže sa sigmoid správa v okrajových bodoch podobne ako perceptrón. Na Obr. 5 je znázornená sigmoid funkcia.

Hlavnou výhodou sigmoid funkcie oproti perceptrónu je jej hladkosť. Táto vlastnosť zaručuje, že pri malej zmene niektorej váhy alebo biasu nastane len malá zmena vo výstupe. Sigmoid funkcia nadobúda hodnotu z intervalu (0,1). Pri binárnej klasifikácii sa berie hodnota 0,5 ako hraničná. Táto hodnota je rozhodujúca pri klasifikácii dát do jednej alebo druhej skupiny. V nasledujúcej leme 1.1 uvádzame základné vlastnosti sigmoid funkie.

Lema 1.1. Vlastnosti sigmoid funkcie
Nech $z \in \mathrm{R}$, potom pre sigmoid funkciu platia nasledujúce vlastnosti:



Obr. 5: Grafické znázornenie sigmoid funkcie. Zdroj: vlastné spracovanie.

1. $0 < \sigma(z) < 1$,
2. $\sigma(z) \leq 0.5$, ak $z \leq 0$,
3. $\sigma(z) > 0.5$, ak $z > 0$,
4. $\sigma(z) = \dfrac{e^z}{e^z + 1} = \dfrac{1}{1 + e^{-z}}$,
5. $\sigma(z)' = \sigma(z)(1 - \sigma(z)) > 0$,
6. $1 - \sigma(z) = \sigma(-z)$.

Dôkaz. Nakoľko prvé 3 vlastnosti uvedené v leme 1.1 sú triviálne, nedokazujeme ich. Štvrtá vlastnosť sa dá dokázať nasledovne

$$\sigma(z) = \frac{e^z}{e^z + 1} = \frac{1}{\dfrac{e^z + 1}{e^z}} = \frac{1}{1 + \dfrac{1}{e^z}} = \frac{1}{1 + e^{-z}}.$$

Využitím štvrtej vlastnosti je možné ukázať, že derivácia $\sigma(z)$ sa dá upraviť na tvar

$$\sigma(z)' = \left[\frac{1}{1 + e^{-z}}\right]' = \frac{e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}} \frac{e^{-z}}{1 + e^{-z}} = \sigma(z)(1 - \sigma(z)).$$

Keďže $\sigma(z) \in (0, 1)$, tak platí $\sigma(z)(1 - \sigma(z)) > 0$. Posledná, šiesta vlastnosť vyplýva zo vzťahu

$$(1 - \sigma(z)) = 1 - \frac{1}{1 + e^{-z}} = \frac{e^{-z}}{1 + e^{-z}} = \sigma(-z).$$

■

1.3.3 Hyperbolický tangens

Ďalšou z aktivačných funkcií, ktorými sa v tejto práci bližšie zaoberáme, je hyperbolický tangens definovaný nasledovne.

Definicia 1.2. Nech $z \in \mathrm{R}$, potom funkcia $h : \mathrm{R} \to \langle -1, 1 \rangle$ v tvare

$$h(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

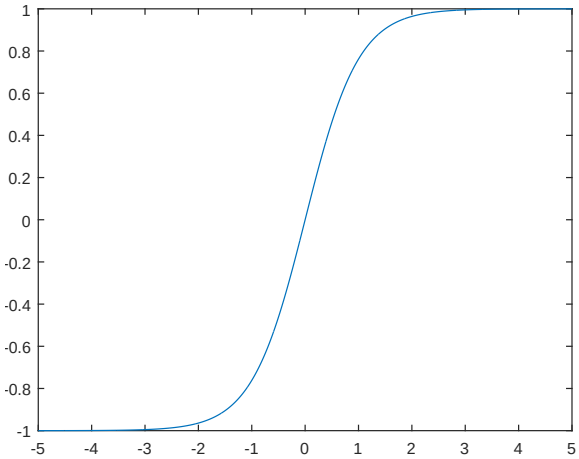
sa nazýva hyperbolický tangens. Označuje sa $h(z) = \tanh(z)$.

Hyperbolický tangens je tvarom podobná funkcia sigmoid funkcii. Na Obr. 6 sa nachádza jej grafické znázornenie.

O vzťahu sigmoid funkcie a hyperbolického tangensu pojednáva nasledujúca lema.

Lema 1.2. Nech $z \in \mathrm{R}$, potom platí nasledujúci vzťah: $2\sigma(2z) - 1 = \tanh(z)$.

Dôkaz. $2\sigma(2z) - 1 = 2\left[\frac{e^{2z}}{e^{2z}+1}\right] - 1 = \frac{2e^{2z} - (e^{2z}+1)}{e^{2z}+1} = \frac{e^{2z}-1}{e^{2z}+1} = \frac{\frac{e^{2z}-1}{e^z}}{\frac{e^{2z}-1}{e^z}} = \frac{e^z - \frac{1}{e^z}}{e^z + \frac{1}{e^z}} = \frac{e^z - e^{-z}}{e^z + e^{-z}} = \tanh(z).$ ■



Obr. 6: Grafické znázornenie funkcie hyperbolický tangens. Zdroj: vlastné spracovanie.

Na využitie aktivačnej funkcie hyperbolický tangens v neurónovej sieti je potrebné poznať jej deriváciu. Táto derivácia je uvedená spolu s dôkazom v nasledujúcej leme 1.3.

Lema 1.3. Nech $z \in R$, potom platí $[\tanh(z)]' = 1 - \tanh^2(z)$.

Dôkaz. Využitím vlastností $[\sinh(z)]' = \cosh(z)$ a $[\cosh(z)]' = \sinh(z)$ dostávame:

$$[\tanh(z)]' = \left[\frac{\sinh(z)}{\cosh(z)}\right]' = \frac{[\sinh(z)]' \cdot \cosh(z) - \sinh(z) \cdot [\cosh(z)]'}{\cosh(z)^2} =$$

$$= \frac{\cosh^2(z) - \sinh^2(z)}{\cosh^2(z)} = 1 - \frac{\sinh^2(z)}{\cosh^2(z)} = 1 - \tanh^2(z).$$

■

1.3.4 ReLu

Za krkolomným názvom *Retifikovaná lineárna jednotka* (z anglického *Rectified linear unit*) sa skrýva pomerne jednoduchý predpis nasledovnej funkcie

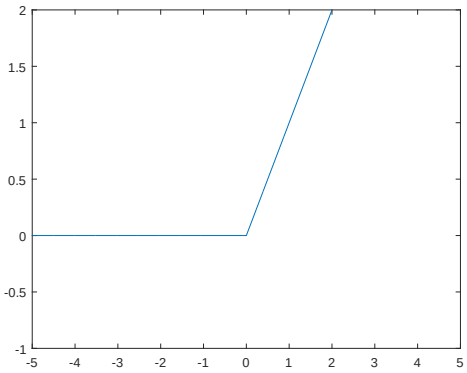
$$h(z) = max\{z, 0\}.$$

ReLu je v súčasnosti najpoužívanejšou aktivačnou funkciou hlavne v hlbokých učiacich sa sieťach. Jedným z hlavných nevýhod sidmoid funkcie a hyperbolického tangensu je, že sú relatívne ploché na väčšine svojho definičného oboru. Relatívna plochosť znamená malé hodnoty derivácií, čo spôsobuje ťažkosti pri vylepšovaní váh hlavne v sietiach s mnohými vrstvami. ReLu týmto problémom netrpí, keďže jej derivácia na intervale $(0, \infty)$ je 1.

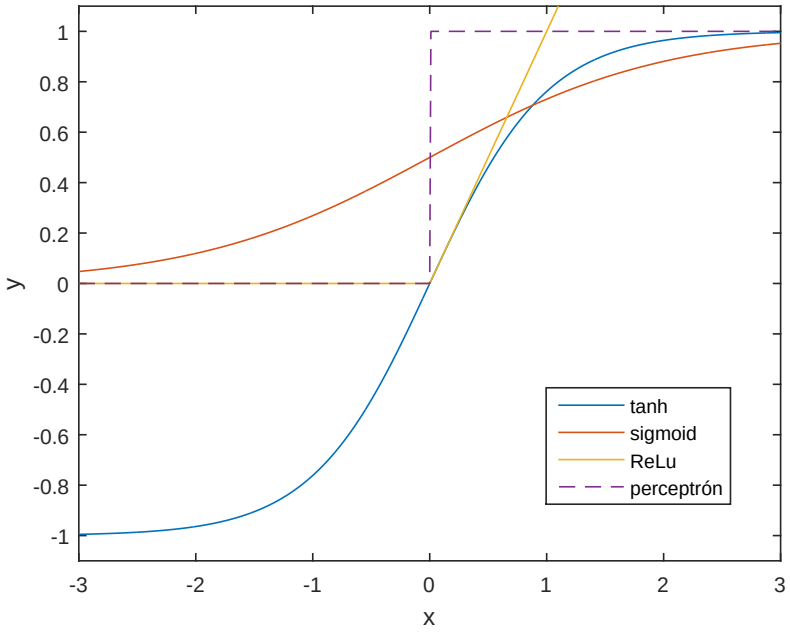
Na Obr. 7 uvádzame tvar ReLu funkcie.

1.4 Porovnanie aktivačných funkcií

Na nasledujúcom obrázku Obr. 8 je zobrazené porovnanie všetkých vyššie spomenutých aktivačných funkcií.



Obr. 7: Grafické znázornenie ReLu funkcie. Zdroj: vlastné spracovanie.



Obr. 8: Porovnanie rôznych aktivačných funkcií. Zdroj: vlastné spracovanie.

1.5 Architektúra umelých neurónových sietí

Umelá neurónová sieť pozostáva z neurónov, ktoré sú medzi sebou poprepájané. Každá skupina neurónov je usporiadaná v jednotlivých vrstvách. Pri klasifikácii sme vychádzali hlavne z teórie definovanej v [7].

Neurónová sieť pozostáva z nasledujúcich vrstiev:

- vstupná vrstva,

- skryté vrstvy,
- výstupná vrstva.

Vstupná vrstva obsahuje vstupné neuróny, inak povedané vstupné údaje a na tejto vrstve neprebiehajú ešte žiadne výpočty. Ak by účelom neurónovej siete bolo rozpoznávanie obrázkov, tak na vstupnej vrstve by sa mohli nachádzať údaje o jednotlivých pixeloch skúmaného obrázka, napríklad hodnota zložky z RGB (z anglického *Red, Green, Blue*).

Skrytých vrstiev v neurónovej sieti môže byť viacero, ale niekedy nemusí byť žiadna. V týchto vrstvách prebieha výpočet, následná aktivácia niektorých neurónov a presun informácií do ďalšej vrstvy.

Vo výstupnej vrstve dochádza k určeniu výsledku klasifikácie. V prípade klasifikácie obrázkov posledná vrstva určuje, čo daný obrázok predstavuje.

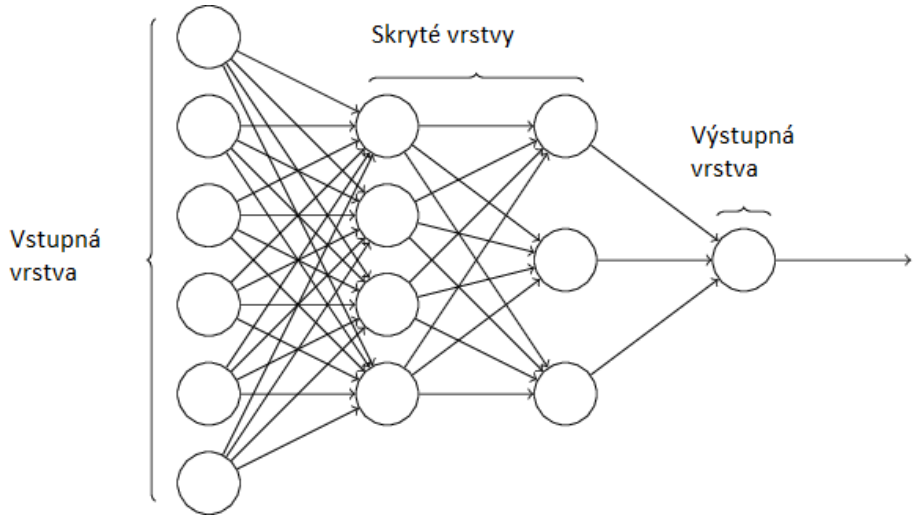
Z hľadiska architektúry sa neurónové siete členia na nasledujúce typy:

1. dopredná neurónová sieť,
2. rekurentná neurónová sieť,
3. konvolučná neurónová sieť.

V tejto práci sa zaoberáme doprednou plnou neurónovou sieťou. Dopredná neurónová sieť znamená, že informácie z i-tej vrstvy prechádzajú do $(i+1)$ vrstvy a nie naopak, teda postupujú dopredu. Plná neurónová sieť značí, že každý neurón z $(i+1)$ vrstvy je prepojený so všetkými neurónmi z i-tej vrstvy.

Rekurentné neurónové siete umožňujú posielanie informácie aj opačným smerom. Konvolučné neurónové siete obsahujú jednu alebo viacero konvolučných vrstiev. Konvolučná vrstva môže slúžiť na znižovanie dimenzie vstupných dát.

Na Obr. 9 je graficky znázornená štruktúra umelej neurónovej siete so skrytými vrstvami.



Obr. 9: Umelá neurónová sieť. Zdroj: vlastné spracovanie.

2 Budovanie jednoduchej neurónovej siete

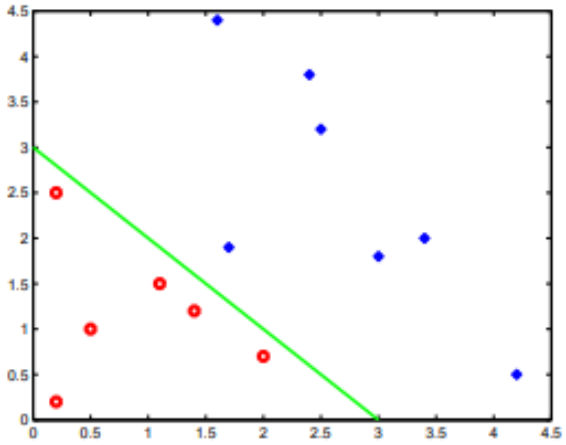
V prvej kapitole bolo objasnené, čo je neurón, v čom sa líšia rôzne druhy neurónov a že ich spojením vzniká neurónová sieť. Cieľom tejto kapitoly je navrhnúť detailný spôsob, ako by mala vyzerať a fungovať vlastná neurónová sieť. Jedným z cieľov tejto práce je vytvorenie neurónovej siete zameranej na binárnu klasifikáciu. To znamená nájdenie vhodného spôsobu, ako oddeliť skupinu dát na 2 časti vzhľadom na to, do ktorej skupiny patria. Skupina dát $x^1, x^2, ..x^m \in R^n$ patrí do jednej z dvoch skupín. Toto zaradenie nech je reprezentované príslušnou premennou $y^i \in \{0, 1\}$,

2.1 Binárna klasifikácia

Ak máme m vstupných dátových bodov s n črtami, kde každý jeden bod patrí do jednej z dvoch skupín, tak cieľom binárnej klasifikácie je nájdenie takej nadroviny, ktorá bude čo najlepšie separovať 2 skupiny bodov v priestore. Táto nadrovina môže byť reprezentovaná vektorom w. Grafické znázornenie takejto situácie v R^2 je možné vidieť na Obr. 10.

Separovateľnosť je možno vyjadriť nasledovne:

$$w^\top x^i + b < 0, \text{ ak } y^i = 0,$$



Obr. 10: Grafické znázornenie binárnej separácie v R^2, zdroj [16].

$$w^\top x^i + b > 0, \text{ ak } y^i = 1.$$

Jedným zo spôsobov ako určiť separujúcu nadrovinu je logistická regresia.

2.2 Logistická regresia

V množstve dát vystupuje ako odhadovaná premenná veličina, ktorá nie je kvantitatívna ale kvalitatívna. Pri predikcii takýchto veličín nie sú klasické metódy ako napríklad lineárna regresia vhodné. Jednou z vhodných metód je logistická regresia, ktorá odhaduje pravdepodobnosť, že pozorovanie patrí do jednej z dvoch kategórií. Pri definícii logistickej regresie v tejto podkapitole sme vychádzali z [1].

Definícia 1.2. [1] Nech y $\in \{0, 1\}$ je náhodná premenná s nasledujúcimi vlastnosťami:

$P(y = 1) = p$, $P(y = 0) = 1 - p$, kde $p \in [0, 1]$. Pravdepodobnosť v logistickom modeli má tvar

$$p = \frac{e^{(w^\top x+b)}}{1 + e^{(w^\top x+b)}} = \sigma(w^\top x + b),$$

kde $w \in R^n$ a $b \in R$ sú parametre modelu určujúce ako sa bude pravdepodobnosť p meniť, vzhľadom na premennú x.

Ako je vidieť logistická funkcia je vlastne sigmoid, takže ich možno považovať za totožné funkcie.

Nech $x^1, .., x^m \in R^n$ sú vstupné premenné a $y^1, ..y^m \in \{0, 1\}$ korešpondujúce výstupné premenné. Logistická regresia spočíva v nájdení odhadu maximálnej vierohodnosti pre parametre w a b. Zoskupením dát do 2 skupín tak, že $I_0 = \{i; y^i = 0\}$ a $I_1 = \{i; y^i = 1\}$ sa dá funkcia vierohodnosti zapísať v tvare:

$$\prod_{i \in I_1} p_i \prod_{i \in I_0} (1 - p_i).$$

Logaritmus funkcie vierohodnosti má následne tvar:

$$\ln \left[\prod_{i \in I_1} p_i \prod_{i \in I_0} (1 - p_i) \right] = \sum_{i \in I_1} \ln(p_i) + \sum_{i \in I_0} \ln(1 - p_i).$$

Dosadením sigmoid funkcie za p dostaneme funkciu v tvare:

$$l(w, b) = \sum_{i \in I_1} \ln \left(\frac{e^{(w^\top x^i + b)}}{1 + e^{(w^\top x^i + b)}} \right) + \sum_{i \in I_0} \ln \left(\frac{1}{1 + e^{(w^\top x^i + b)}} \right) =$$

$$= \sum_{i \in I_1} \ln \left(\frac{e^{(w^\top x^i + b)}}{1 + e^{(w^\top x^i + b)}} \right) + \sum_{i \in I_0} \ln \left(\frac{1}{1 + e^{(w^\top x^i + b)}} \right) =$$

$$\sum_{i \in I_1} w^\top x^i + b - \sum_{i \in I} \ln \left(1 + e^{(w^\top x^i + b)} \right).$$

Využitím nasledovnej rovnosti, môžeme výraz spojiť do jednej sumy

$$\sum_{i \in I_1} w^\top x^i + b = \sum_{i \in I} y^i \cdot (w^\top x^i + b).$$

Táto rovnosť platí, keďže y^i nadobúda nulové hodnoty v intervale I_0.

$$l(w, b) = \sum_{i \in I} \left[y^i \cdot (w^\top x^i + b) - \ln(1 + e^{(w^\top x^i + b)}) \right].$$

Ak by bola funkcia l konkávna vo w a b, tak jej maximalizáciou cez tieto premenné by bolo možné získať optimálne hodnoty pre w a b.

2.3 Nákladová funkcia

Keďže sigmoid funkcia nadobúda hodnoty z intervalu $(0, 1)$ je veľmi vhodná ako aktivačná funkcia do výstupnej vrstvy, ktorá bude určovať s akou pravdepodobnosťou

patrí vstupný prvok do jednej z dvoch tried. Naším cieľom teda je nastavenie váh tak, aby funkcia $\sigma(z)$ nadobúdala hodnoty blízke 1 pre jednu skupinu dát a blízke 0 pre druhú. Jedným zo spôsobov ako to docieliť, je meniť váhy a sledovať či sa hodnota $\sigma(z)$ približuje k správnej triede a tým je chyba odhadu menšia. Na meranie celkovej hodnoty chyby pre všetky vstupné dátové body sa používa nákladová funkcia. Pomocou nákladovej funkcie vieme merať ako pôsobí zmena váh na klasifikáciu. Minimalizovaním nákladovej funkcie cez (w, b) vieme docieliť nájdenie optimálnych váh a biasu. Zvyčajne sa používa optimalizácia pomocou metódy najmenších štvorcov a minimalizačnej úlohy.

$$\hat{J}(X, Y, w, b) = \frac{1}{2m} \sum_{i=1}^{m} (\sigma(w^\top x^i + b) - y^i)^2, = \frac{1}{m} \sum_{i=1}^{m} C_{quad}(y^i, w^\top x^i + b),$$

kde

- m je počet vstupných dát,
- n je rozmer dátových bodov,
- $X = (x^1, ..., x^m) \in \mathrm{R}^{n \times m}$ je vstupná matica dátových bodov rozmeru $n \times m$,
- $Y = (y^1, ..., y^m) \in \mathrm{R}^m$ predstavuje vektor výstupov prislúchajúcich jednotlivým vstupom,
- $C_{quad}(y, z) = \frac{1}{2}(\sigma(z) - y)^2$.

Dá sa ukázať, že funkcia C_{quad} nie je konvexná, a teda nie je možné využiť klasické optimalizačné metódy, avšak použitím nasledujúcej konvexnej alternatívy sa tento problém dá vyriešiť.

$$C_{\ln}(y, z) = -\Big[y \cdot \ln(\sigma(z)) + (1 - y) \cdot \ln(1 - \sigma(z))\Big].$$

Hoci sa to na prvý pohľad nemusí zdať, $C_{\ln}$ je vhodná funkcia. Čím sú váhy lepšie nastavené, tým má $\sigma(z)$ hodnotu bližšiu k príslušnému y a celková hodnota nákladovej funkcie klesá k 0.

Ak $y = 0$, tak platí:

$$\lim_{\sigma(z)\to 0} C_{\ln}(0, z) = \lim_{\sigma(z)\to 0} \Big[- (0 \cdot \ln(\sigma(z)) + (1 - 0) \cdot \ln(1 - \sigma(z))) \Big] =$$
$$= \lim_{\sigma(z)\to 0} \Big[- \ln(1 - \sigma(z)) \Big] = 0.$$

Ak $y = 1$, tak platí:

$$\lim_{\sigma(z)\to 1} C_{\ln}(1, z) = \lim_{\sigma(z)\to 1} \Big[- (1 \cdot \ln(\sigma(z)) + (1 - 1) \cdot \ln(1 - \sigma(z))) \Big] =$$
$$= \lim_{\sigma(z)\to 1} \Big[- \ln(\sigma(z)) \Big] = 0.$$

Funkcia $C_{\ln}$ sa dá zjednodušiť na nasledujúci tvar:

$$\begin{aligned} C_{\ln}(y, z) &= -\Big[y \cdot \ln(\sigma(z)) + (1 - y) \cdot \ln(1 - \sigma(z)) \Big] = \\ &= -\Big[y \cdot \ln(\frac{e^z}{1 + e^z}) + (1 - y) \cdot \ln(1 - \frac{e^z}{1 + e^z}) \Big] = \\ &= -\Big[y \cdot z - y \cdot \ln(1 + e^z) + (1 - y) \cdot \ln(\frac{1}{1 + e^z}) \Big] = \\ &= -\Big[y \cdot z - y \cdot \ln(1 + e^z) - (1 - y) \cdot \ln(1 + e^z) \Big] = \\ &= -\Big[y \cdot z - \ln(1 + e^z) \Big]. \end{aligned}$$

Takže pre dáta reprezentované X a Y, a parametre w a b, bude nákladová funkcia vyzerať nasledovne:

$$\begin{aligned} J(X, Y; w, b) &= \frac{1}{m} \sum_{i=1}^{m} C_{ln}(y^i, w^\top x^i + b) = \\ &= \frac{1}{m} \sum_{i=1}^{m} \Big[y^i \cdot \ln(\sigma(w^\top x^i + b)) + (1 - y^i) \cdot \ln(1 - \sigma(w^\top x^i + b)) \Big] = \\ &\quad -\frac{1}{m} \sum_{i=1}^{m} \Big[y^i \cdot (w^\top x^i + b) - \ln(1 - e^{(w^\top x^i + b)} \Big]. \end{aligned} \tag{1}$$

Z tohto tvaru nákladovej funkcie je už pomerne jednoduché vidieť, že platí

$$J(X, Y, w, b) = -\frac{1}{m} l(w, b).$$

V nasledujúcej časti práce je dokázaná konvexnosť nákladovej funkcie J.

Tvrdenie 1.1.: Funkcia $J(X, Y; w, b)$ je konvexná vo (w, b).

Dôkaz. Keďže funkcia J je lineárnou kombináciou funkcií $C_{\ln}$, stačí ukázať, že $C_{\ln}$ je konvexná vo (w, b).

Nech $g : R^n \to R$ má spojité druhé parciálne derivácie. Potom g je konvexná, práve vtedy keď, $\forall x \in R^n$ je Hessova matica funkcie g kladne semidefinitná, takže postačujúcou podmienkou je, že Hessova matica funkcie $C_{\ln}$ vo (w, b) je kladne semidefinitná.

Hessova matica bude mať nasledujúcu štruktúru:

$$\nabla^2 C_{\ln} = \left[\begin{array}{c|c} \frac{d^2}{dw^2} C_{\ln} & \frac{d^2}{dbdw} C_{\ln} \\ \hline \frac{d^2}{dwdb} C_{\ln} & \frac{d^2}{db^2} C_{\ln} \end{array}\right].$$

Najskôr si, kvôli prehľadnosti, odvodíme niektoré čiastočné derivácie:

$$\frac{\partial(w^\top x + b)}{\partial w_j} = \frac{\partial(w_1 x_1 + ... + w_j x_j + ... + w_n x_n + b)}{\partial w_j} = x_j.$$

Využitím lemy 1.1 vlastnosti 4 dostaneme

$$\frac{\partial(\sigma(w^\top x + b))}{\partial w_j} = \sigma(w^\top x{+}b)(1{-}(\sigma(w^\top x{+}b))) \frac{\partial(w^\top x + b)}{\partial w_j} = \sigma(w^\top x{+}b)(1{-}(\sigma(w^\top x{+}b))x_j$$

a potom platí

$$\frac{\partial(\ln(\sigma(w^\top x + b)}{\partial w_j} = \frac{1}{\sigma(w^\top x + b)} \frac{\partial(\sigma(w^\top x + b))}{\partial w_j} =$$

$$= \frac{1}{\sigma(w^\top x + b)} \sigma(w^\top x + b)(1 - (\sigma(w^\top x + b))x_j = (1 - (\sigma(w^\top x + b))x_j.$$

Využitím týchto informácií je už celkom jednoduché vypočítať jednotlivé parciálne derivácie:

$$\begin{aligned}
\frac{\partial C_{\ln}}{\partial w_j} &= \frac{\partial}{\partial w_j} \left[-y \cdot \ln(\sigma(w^\top x + b)) - (1 - y) \cdot \ln(1 - \sigma(w^\top x + b))\right] = \\
&= \frac{\partial}{\partial w_j} \left[-1 \left(y \cdot \ln(\sigma(w^\top x + b)) + (1 - y) \cdot \ln(1 - \sigma(w^\top x + b))\right)\right] = \\
&= -1 \cdot \left[y \cdot (1 - \sigma(w^\top x + b)) \cdot x_j - (1 - y) \cdot \sigma(w^\top x + b) \cdot x_j\right] = \\
&= -1 \cdot \left[y \cdot x_j - y \cdot \sigma(w^\top x + b) \cdot x_j - 1 \cdot \sigma(w^\top x + b) \cdot x_j + y \cdot \sigma(w^\top x + b) \cdot x_j\right] = \\
&= x_j \cdot (\sigma(w^\top x + b) - y),
\end{aligned}$$

$$\frac{\partial C_{\ln}}{\partial b} = \frac{\partial}{\partial b}[-y \cdot \ln(\sigma(w^\top x + b)) - (1-y) \cdot \ln(1 - \sigma(w^\top x + b))] =$$
$$= -1 \cdot \Big[y \cdot (1 - \sigma(w^\top x + b)) \cdot -(1-y) \cdot \sigma(w^\top x + b) \cdot \Big] = (\sigma(w^\top x + b) - y).$$

Následne pomocou prvých parciálnych derivácií môžeme odvodiť druhé parciálne derivácie:

$$\frac{\partial^2 C_{\ln}}{(\partial w_j)^2} = \frac{\partial}{\partial w_j}[x_j \cdot (\sigma(w^\top x + b) - y)] = (x_j)^2 \cdot \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)),$$

$$\frac{\partial^2 C_{\ln}}{\partial w_j \partial w_k} = \frac{\partial}{\partial w_k}[x_j \cdot (\sigma(w^\top x + b) - y)] = (x_j \cdot x_k \cdot \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)),$$

$$\frac{\partial^2 C_{\ln}}{\partial w_j \partial b} = \frac{\partial}{\partial b}[x_j \cdot (\sigma(w^\top x + b) - y)] = (x_j \cdot \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)),$$

$$\frac{\partial^2 C_{\ln}}{(\partial b)^2} = \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)).$$

Z lemy 1.1 vlastnosť 4 vieme, že výraz $\sigma(w^\top x + b)(1 - \sigma(w^\top x + b))$ je kladný. Keďže sa nachádza v každej parciálnej derivácii, tak môžeme Hessovu maticu upraviť do nasledujúceho tvaru:

$$\nabla^2 C_{\ln} = \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)) \begin{bmatrix} (x_1)^2 & \dots & x_1 \cdot x_n & x_1 \\ \vdots & \ddots & \vdots & \\ x_1 \cdot x_n & \dots & (x_n)^2 & x_n \\ x_1 & \dots & x_n & 1 \end{bmatrix} =$$
$$= \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)) \cdot (x_1, x_2, ..., x_n, 1)^\top \cdot (x_1, x_2, ..., x_n, 1) =$$
$$= \sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b)) \cdot \tilde{x}^\top \tilde{x},$$

kde $\tilde{x} = (x_1, x_2, ..., x_n, 1)$.

Súčin $\sigma(w^\top x + b) \cdot (1 - \sigma(w^\top x + b))$ je kladný a ľubovoľná matica v tvare $\tilde{x}^\top \tilde{x}$ je kladne semidefinitná. Z toho vyplýva, že aj Hessova matica funckie $C_{\ln}$ je kladne semidefinitná a teda $C_{\ln}$ je konvexná funkcia vo (w, b), z čoho vyplýva, že aj funkcia J je konvexná vo (w, b).

■

Keďže sme dokázali, že nákladová funkcia J je konvexná vo (w, b), tak pre hľadanie optimálnych hodnôt w a b je možné použiť klasické optimalizačné metódy ako napríklad gradientnú metódu definovanú v nasledujúcej podkapitole.

2.4 Gradientná metóda

Zistenie optimánych váh spočíva v hľadaní minima funkcie J. Možnosťou ako získať optimálne váhy je postupné minimalizovanie nákladovej funkcie J. Ako však meniť váhy tak, aby nákladová funkcia zmenšovala svoju hodnotu? Vhodným riešením je robiť zmenu váh v opačnom smere najväčšieho gradientu. Táto myšlienka je základom gradientnej metódy [15].

Algoritmus gradientnej metódy je nasledovný.
Pre vopred zvolený počet iterácií vykonajú nasledovné kroky:

1. zvolenie smeru pomocou gradientu nákladovej funkcie,

2. určenie nových váh pomocou nasledujúceho iteračného vzťahu:

$$\begin{aligned} w(k+1) &= w(k) - \alpha \frac{dJ}{dw}(k), \\ b(k+1) &= b(k) - \alpha \frac{\partial J}{\partial b}(k), \end{aligned} \tag{2}$$

   kde α je malé kladné číslo určujúce dĺžku kroku, $\frac{dJ}{dw}$ a $\frac{\partial J}{\partial b}$ sú vektory parciálnych derivácií funkcie J.

Po zbehnutí istého počtu iterácií sa posledná hodnota w a b berie ako optimálna. Pre nákladovú funkciu $J(w, b)$ a aktivačnú funkciu sigmoid zvolené v tejto práci nado-

búda gradient nasledujúci tvar:

$$\nabla J = (\frac{dJ}{dw}, \frac{\partial J}{\partial b})^\top,$$

pričom jednotlivé zložky vektora ∇J si vieme vyjadriť ako

$$\frac{dJ}{dw} = (\frac{\partial J}{\partial w_1}, ..., \frac{\partial J}{\partial w_m})^\top; \qquad \frac{\partial J}{\partial b} = \frac{1}{m}\sum_{i=1}^{m}(\sigma(w^\top x^i + b) - y^i),$$

kde

$$\frac{\partial J}{\partial w_1} = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial C_{\ln}(y^i, w^\top x^i + b)}{\partial w_1} = \frac{1}{m}\sum_{i=1}^{m} x_1^i \cdot (y^i - \sigma(w^\top x^i + b)),$$

$$\begin{aligned}\frac{\partial J}{\partial w} &= \Big(\frac{1}{m}\sum_{i=1}^{m} x_1^i \cdot (y^i - \sigma(w^\top x^i + b)), ..., \frac{1}{m}\sum_{i=1}^{m} x_m^i \cdot (y^i - \sigma(w^\top x^i + b))\Big)^\top \\ &= \frac{1}{m}X(\sigma(Z) - Y),\end{aligned}$$

kde $Z = (z^1, \ldots, z^m)^\top$, pričom $z^i = w^\top x^i + b$ nazývame vážený vstup pre x^i

a na záver dostávame

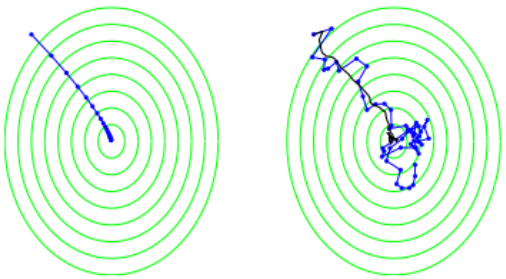
$$\nabla J = \Big[\frac{1}{m}X(\sigma(Z) - Y), \frac{1}{m}\sum_{i=1}^{m}(\sigma(w^\top x^i + b) - y^i)\Big]^\top. \tag{3}$$

2.4.1 Stochastická gradientná metóda

Pre veľké sady dát, ktoré sú potrebné pre trénovanie neurónových sietí, klasická gradientná metóda nie je dostačujúca z dôvodu časovej a výpočtovej náročnosti. Preto sa pri pri veľkých dátových sadách využíva takzvaná stochastická gradientná metóda označovaná SGD (z anglického *Stochastic Gradient Descent*). Jej princíp spočíva v spočítaní gradientu nákladovej funkcie J pre náhodné zvolenú jednu vstupnú hodnotu alebo malú vzorku dát a následnej úprave váh ná základe vypočítaného gradientu. Toto sa opakuje k-krát, kde

$$k = \frac{\text{počet vstupných dát}}{\text{veľkosť vzorky}}.$$

Takýmto spôsobom sa počas jednej iterácie využitím celej sady dát upravia váhy k-krát miesto 1 úpravy, ako by tomu bolo pri klasickej gradientnej metóde. Keďže sa gradient počíta iba z malého množstva dát, tak gradient nie je optimálny pre celú sadu, čo môže spôsobiť nespádovosť pri poklese nákladovej funkcie.



Obr. 11: Ilustrácia obyčajnej gradientnej metódy (vľavo) a stochastickej gradientnej metódy (vpravo). Zdroj [11].

2.5 Spustenie jednoduchej neurónovej siete

Teraz je už všetko pripravené na spustenie jednoduchej neurónovej siete. Pod jednoduchou neurónovou sieťou je možné rozumieť iba toľko, že nemá žiadnu skrytú vrstvu.

Algoritmus fungovania jednoduchej neurónovej siete:

1. načítanie vstupných dát a prvotná náhodná inicializácia váh a biasu,
2. pre vopred zvolený počet iterácií, sa vykonajú nasledovné kroky:
   - výpočet $\sigma(Z)$,
   - výpočet nákladovej funkcie $J(X, Y; w, b)$ pomocou vzťahu (1),
   - výpočet gradientu nákladovej funkcie na základe vzťahu (3),
   - úprava w a b využitím (2),
3. uloženie váh získaných pri poslednej iterácii a ich využívanie na klasifikáciu testovacích dát.

2.6 Aplikácia jednoduchej neurónovej siete na binárnu sadu dát

Neurónovú sieť sme programovali v prostredí Matlab, konkrétne vo verzii R2015a. Všetky kódy sú voľne dostupné na internetovej stránke `https://github.com/mile3ko/diplomovka`.

Za testovacie dáta, pre nami vytvorenú neurónovú sieť, sme si zvolili vzorku fotiek psov a mačiek z dátového zdroja [6]. V súbore sa nachádza 25 000 obrázkov, pričom polovicu tvoria obrázky psov a polovicu obrázky mačiek. Túto sadu sme rozdelili na 2 časti v pomere 4:1. Trénovacia sada slúži na nájdenie optimálnych váh a biasov. Druhá časť, testovacia sada, slúži na vyhodnotenie úspešnosti klasifikácie neurónovej siete.

Obrázky boli skonvertované do rozmeru 64×64 pixelov a každý pixel je reprezentovaný 3 hodnotami farebného spektra RGB (z anglického *Red, Green, Blue*). Každý obrázok je teda reprezentovaný $64 \cdot 64 \cdot 3 = 12\ 288$ hodnotami (od 0 do 255) vo forme vektora. Dáta boli následne normalizované prenásobením každej hodnoty číslom $\frac{1}{255}$. Tieto hodnoty prestavujú vstupnú neurónovú vrstvu. Výstupná vrstva pozostáva iba z jedného neurónu, ktorý bude reprezentovať, či je daný obrázok pes alebo mačka s istou pravdepodobnosťou. Čím bude toto číslo bližšie k 1, tým je podľa neurónovej siete väčšia pravdepodobnosť, že na danom obrázku je mačka.

Kvôli prehľadnosti tabuliek sme zaviedli nasledujúce označenia:

- J_{tren} - hodnota nákladovej funkcie pre trénovaciu sadu dát,
- J_{test} - hodnota nákladovej funkcie pre testovaciu sadu dát,
- ÚK$_{tren}$ - úspešnosť klasifikácie trénovacej sady dát,
- ÚK$_{test}$ - úspešnosť klasifikácie testovacej sady dát.

Dôležitým nastavením pri gradientnej metóde je voľba parametra α, ktorý určuje dĺžku kroku. Pre tento parameter neexistuje exaktná voľba, pretože pri rôznych dátach je potrebné ho zvoliť inak. Jedným z odporúčaní je voliť čo čo najväčšie α, pri ktorom

nebude hodnota nákladovej funkcie divergovať. V nasledujúcej Tabuľke 1 sú uvedené hodnoty nákladovej funkcie a úspešnosť klasifikácie trénovacej a testovacej sady dát po 20000 iteráciách pre rôzne nastavenia parametra α.

Tabuľka 1: Hodnoty nákladovej funkcie a úspešnosť klasifikácie dát pre rôzne hodnoty parametra α po 10 000 iteráciách a veľkosti trénovacej sady 200 dát.

α	0.05	0.03	0.01	0.001
J_{tren}	NA	0.00124420	0.0072719	0.0753245
ÚK$_{tren}$	NA	1	1	1
ÚK$_{test}$	NA	0.61	0.60	0.60

Zdroj: vlastné spracovanie.

Pre hodnotu $\alpha = 0.05$ hodnota nákladovej funkcie začala rásť, čo znamená, že daný krok je príliš veľký. Ostatné testované hodnoty α boli dostatočne malé, aby hodnota nákladovej funkcie klesala. Zároveň je dôležité, aby parameter nebol príliš malý, pretože by konvergencia trvala príliš dlho. Pre túto testovaciu sadu dát bude vhodné voliť parameter $\alpha \in \langle 0.01, 0.03 \rangle$

Pre dosiahnutie dobrých výsledkov potrebujú neurónové siete veľké množstvo dát, avšak čím je trénovacia vzorka väčšia, tým dlhšie trvajú jednotlivé iterácie a narastá celková časová náročnosť. V nasledujúcej Tabuľke 2 sú uvedené výsledky pre rôznu veľkosť trénovacej vzorky.

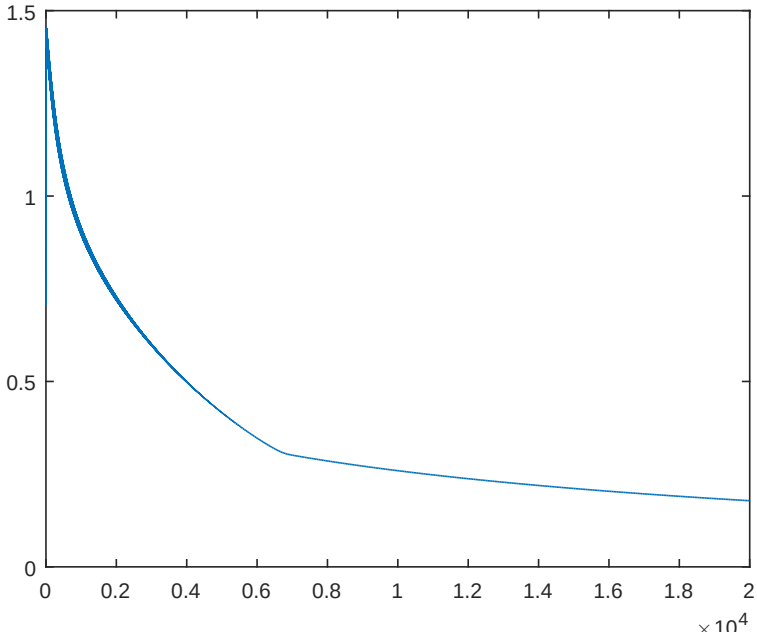
Zvyšovaním veľkosti trénovacej sady nielen že narastá čas, ktorý je potrebný na natrénovanie neurónovej siete, ale ani neprišlo k zlepšeniu úspešnosti klasifikácie testovacích dát. Takáto jednoduchá neurónová sieť nebude postačujúca na binárnu klasifikáciu obrázkov. Túto úvahu je možné potvrdiť nasledujúcimi obrázkami. Na Obr. 12 je vidieť vývoj nákladovej funkcie. Hodnota nákladovej funkcie neustále klesala a úspešnosť klasifikácie trénovacej sady narastala. Existuje teda predpoklad, že by sa zlepšovala aj naďalej. Vývoj úspešnosti klasifikácie trénovacej sady dát je zobrazený na Obr. 13. Napriek tomu k zlepšovaniu klasifikácie testovacej sady neprichádzalo. Najvyššia úspešnosť klasifikácie testovacej sady bola dosiahnutá okolo 6 000 iterácii a

Tabuľka 2: Hodnoty nákladovej funkcie a úspešnosť klasifikácie pre rôznu veľkosť trénovacej sady po 20 000 iteráciách pre $\alpha = 0.01$.

Veľkosť trénovacej sady	200	500	1 000	2 000
J_{tren}	0.007268	0.03101	0.090358	0.6191869
ÚK_{tren}	1	1	1	0.754
ÚK_{test}	0.60	0.59	0.58	0.60
Čas trénovania NN v sekundách	108	257	570	1 037

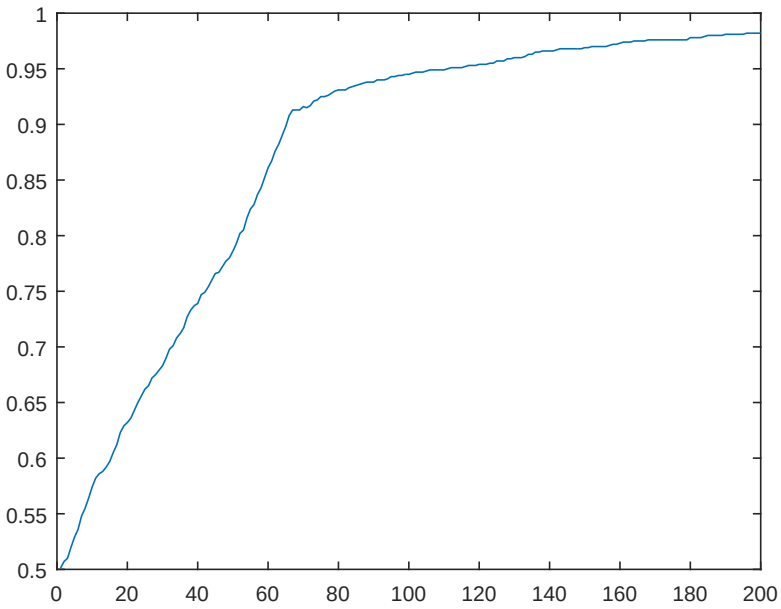
Zdroj: vlastné spracovanie.

potom nastal mierny pokles úspešnosti (viď Obr. 14).

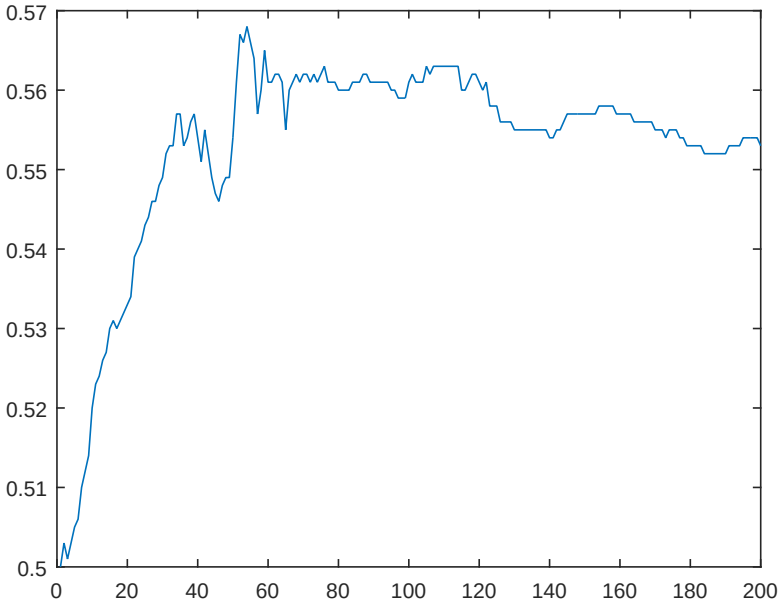


Obr. 12: Vývoj hodnoty nákladovej funkcie trénovacej sady počas iteračného procesu. Zdroj: vlastné spracovanie.

Jednoduchá neurónová sieť iba so vstupnou a výstupnou vrstvou sa ukazuje ako nedostatočná pre binárnu klasifikáciu, hoci úspešnosť klasifikácie stúpla na 60%. Preto je potrebné rozšíriť neurónovú sieť o ďalšie, skryté vrstvy.



Obr. 13: Úspešnosť klasifikácie trénovacej sady dát zaznamenaná pre každú 100. iteráciu. Zdroj: vlastné spracovanie.



Obr. 14: Úspešnosť klasifikácie testovacej sady dát zaznamenaná pre každú 100. iteráciu. Zdroj: vlastné spracovanie.

2.7 Aplikácia jednoduchej neurónovej siete na sadu dát s viacerými triedami

Ako druhú sadu dát na klasifikáciu pomocou neurónovej siete sme použili dátovú sadu SVHN (z anglického *The Street View House Numbers*) zo zdroja [17]. Táto dátova sada predstavuje farebné fotky čísel domov, získaných z Google street view. Použili sme už predpripravenú sadu dát, všetky fotky majú rozmer 32×32 pixelov a sú vycentrované na číslicu, ktorá má byť klasifikovaná. Sada pozostáva z približne 70 000 trénovacích dát a 20 000 testovacích. V dátovej sade sa nachádza 10 rôznych číslic $(0, \ldots, 9)$, každá reprezentovaná jednou triedou. Na nasledujúcom obrázku Obr. 15 je ukážka fotiek z dátovej sady, s ktorou pracujeme.



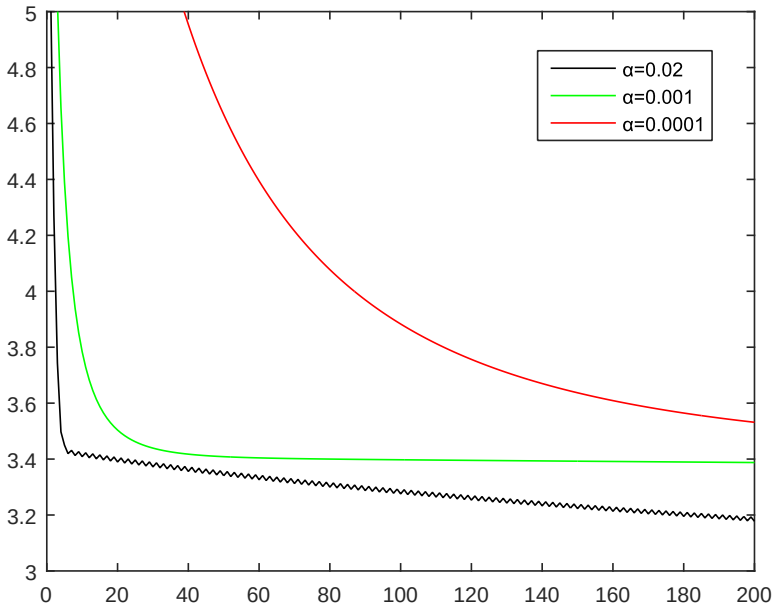
Obr. 15: Ukážka obrázkov z dátovej sady SVHN. Zdroj: [18].

Poznatky získané pri binárnej klasifikácii je možné využiť aj pri klasifikácii do viacerých tried. Vo výstupnej vrstve neurónovej siete sa bude nachádzať 10 neurónov, každý bude vyjadrovať s akou pravdepodobnosťou patrí daný vstup do konkrétnej

triedy. Teda výstupom neurónovej siete pre jeden vstup $x^i \in \mathrm{R}^m$ bude vektor $\sigma(z^i) \in \mathrm{R}^{10}$, kde hodnota na j-tej pozícii bude vyjadrovať pravdepodobnosť, že vstup patrí do triedy j.

Pre ktorú triedu bude pravdepodobnosť najvyššia, tá trieda sa bude brať za odhadovanú triedu. Každému obrázku reprezentovanému vektorom x^i bude prislúchať vektor $y^i \in R^{10}$, kde číslo 1 na j-tej pozícii, bude znamenať, že obrázok patrí do triedy j. Číslici 0 bude patriť 10 pozícia. Na ostatných pozíciách vektora sa nachádzajú 0. Napríklad pre obrázok, reprezentovaný vektorom x^i na ktorom sa nachádza číslica 7, bude príslušný vektor y^i mať tvar $y^i = (0, 0, 0, 0, 0, 0, 1, 0, 0, 0)^\top$.

Najpr sme sa pozreli ako sa správa nákladová funkcia pre rôzne hodnoty parametra α. Vývoj je zobrazený na Obr. 16.



Obr. 16: Grafické znázornenie vývoja nákladovej funkcie pre rôzne hodnoty parametra α. Zdroj: vlastné spracovanie.

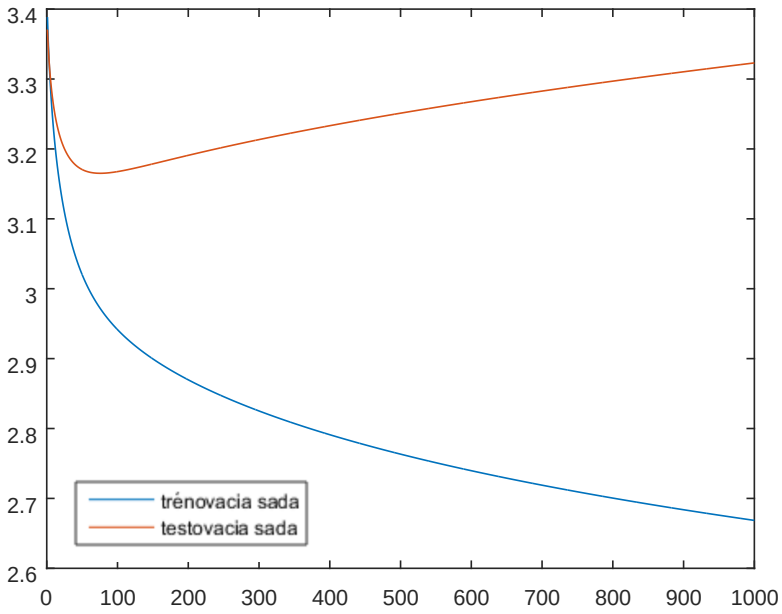
Hodnota 0,02 sa ukázala ako vhodná, pretože zaručovala dostatočne rýchly pokles, tak sme pomocou nej otestovali úspešnosť klasifikácie pre rôzne veľkosti sady dát. Výsledky sú uvedené v nasledujúcej Tabuľke 3. Úspešnosť klasifikácie stúpla z 10% približne na 26% čo nie je dostatočne dobrá klasifikácia.

Tabuľka 3: Hodnota nákladovej funkcie a úspešnosť klasifikácie pre rôznu veľkosť trénovacej sady dát.

Počet iterácií = 10 000, alfa=0.02				
Veľkosť trénovacej sady	200	500	1 000	5 000
J_{tren}	0.482	1.562	2.377	3.047
J_{test}	4.086	3.847	3.882	3.279
ÚK$_{tren}$	0.995	0.764	0.529	0.304
ÚK$_{test}$	0.215	0.216	0.214	0.263

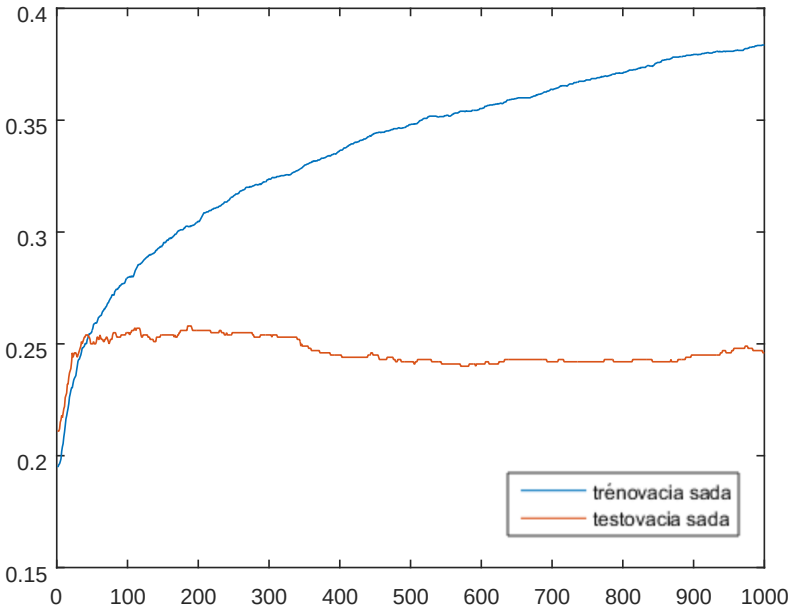
Zdroj: vlastné spracovanie.

Na nasledujúcich obrázkoch je znázornený vývoj nákladových funkcií a úspešnosť klasifikácie pre trénovaciu a testovaciu sadu počas 100 000 iterácií využitím klasického gradientu.



Obr. 17: Grafické znázornenie vývoja nákladových funkcií zaznamenaných každú 100. iteráciu. Zdroj: vlastné spracovanie.

Ako vidieť z obrázkov Obr. 17 a Obr. 18 ani dlhšie trénovanie takejto jednoduchej



Obr. 18: Grafické znázornenie vývoja úspešnosti klasifikácie zaznamenaných každú 100. iteráciu. Zdroj: vlastné spracovanie.

neurónovej siete nepomohlo zlepšiť klasifikáciu testovacej sady.

3 Zložitejšia neurónová sieť

V predchádzajúcej kapitole sa ukázala jednoduchá neurónová sieť ako nedostatočná na binárnu klasifikáciu obrázkov. Táto kapitola sa preto zaoberá rozšírením neurónovej siete o ďalšie vrstvy. Pri jednoduchej neurónovej sieti bol potrebný jeden vektor váh $w \in R^n$ a jedna excitačná hranica $b \in R$, ktoré prepájali n vstupných neurónov s 1 výstupným neurónom. Pre pridanie 1 skrytej vrstvy s N_h neurónmi bude potrebné pre každý neurón v skrytej vrstve vytvoriť zodpovedajúci vektor váh a bias. V skrytých vrstvách sme používali hyperbolický tangens ako aktivačnú funkciu.

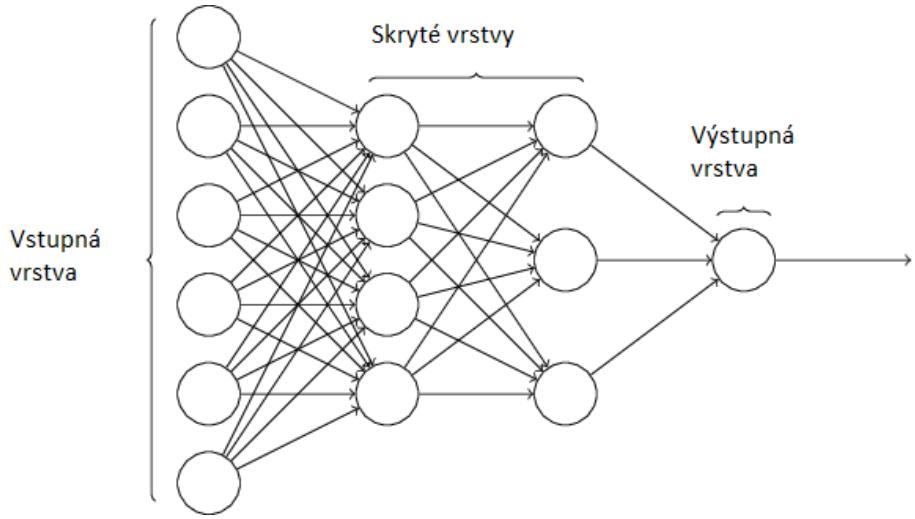
Zavedenie nového značenia pre neurónovú sieť so skrytými vrstvami, ak $(l-1)$-vrstva pozostáva z u neurónov a l-tá vrstva pozostáva z v neurónov :

- w^l_{jk} = váha medzi k-tym neurónom v $(l-1)$ vrstve a j-tym neurónom v l-tej vrstve,
- $w^l \in R^{v \times u}$ = matica váh v l-tej vrstve,
- b^l_j = bias pre j-ty neurón v l-tej vrstve,
- $b^l \in R^v$ = vektor biasov v l-tej vrstve,
- z^l_i = vážený vstup pre x^i v l-tej vrstve,
- Z^l = matica vážených vstupov v l-tej vrstve,
- a^l_i = aktivačný vektor pre x^i v l-tej vrstve, pričom $a^l_i = f(z^l_i)$,
- a^l = matica aktivačných vektorov v l-tej vrstve.

Na Obr. 19 je graficky znázornená štruktúra viacvrstvovej neurónovej siete.

3.1 Spätná propagácia

Aby bolo možné minimalizovať nákladovú funkciu a optimalizovať váhy k čo najpresnejšej klasifikácii, je potrebné spočítať parciálne derivácie nákladovej funkcie.



Obr. 19: Viacvrstvová neurónová sieť. Zdroj: vlastné spracovanie.

Vhodným algoritmom je takzvaná spätná propagácia. Tento algoritmus bol prvý krát spomenutý v [10]. V tejto podkapitole sme sa opierali predovšetkým o teóriu definovanú v [8].

Cieľom spätnej propagácie je rýchle vypočítanie $\frac{dJ}{dw}$ a $\frac{\partial J}{\partial b}$.

K pochopeniu princípu spätnej propagácie je vhodné zaviesť pomocnú premennú δ. Túto premennú je možné interpretovať ako chybu. Konkrétnejšie, δ_j^l je chyba na j-tom neuróne na l-tej vrstve, ktorá je definovaná nasledovne:

$$\delta_j^l = \frac{\partial J}{\partial z_j^l}.$$

Teda ak $\frac{\partial J}{\partial z_j^l}$ je malá, tak jej zmenou v smere gradientu sa iba málo zmení nákladová funkcia a preto chyba na tomto mieste je malá. Naopak ak je $\frac{\partial J}{\partial z_j^l}$ veľká, je možné zmenou signifikantnejšie ovplyvniť nákladovú funkciu. Na základe tento chyby je možné odvodiť nasledujúce rovnice spätnej propagácie.

Základné 4 rovnice spätnej propagácie:

$$
\begin{aligned}
SP1 &: \delta^L = \nabla_a J \odot \sigma'(z^L), \\
&\quad \delta_i^L = \frac{\partial J}{\partial a_i^L} \sigma'(z_i^L), \\
SP2 &: \delta^l = (w^{l+1})^\top \delta^{l+1} \odot \sigma'(z^l), \\
SP3 &: \frac{\partial J}{\partial b_j^l} = \delta_j^l, \\
SP4 &: \frac{\partial J}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l,
\end{aligned}
$$

kde znak $\odot$ predstavuje Hadamardov súčin.

Definícia 1.4. Hadamardov súčin [4]
Nech $A, B \in \mathrm{R}^{n \times m}$ sú matice. Potom je Hadamardov súčin definovaný nasledovne:

$$[A \odot B]_{ij} = A_{ij} B_{ij}, \forall i \in \langle 1, m \rangle, \forall j \in \langle 1, n \rangle.$$

Dôkaz: Odvodenie týchto 4 rovníc je možné nájsť napríklad v [10]. Tu uvádzame iba náčrt dôkazu.

Pre rovnicu SP1 využitím derivácie zloženej funkcie platí nasledovné:

$$\delta_j^L = \frac{\partial J}{\partial z_j^l} = \sum_k \frac{\partial J}{\partial a_k^L} \frac{\partial a_k^L}{\partial z_j^L},$$

kde k je počet neurónov vo výstupnej vrstve siete. Pre $k \neq j$ aktivačná funkcia a_k^l nezávisí od váženého vstupu z_j^L, preto rovnosť môže byť zjednodušená ako

$$
\begin{aligned}
\delta_j^L &= \frac{\partial J}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial J}{\partial a_j^L} \sigma'(z_j^L), \\
\delta^L &= \nabla_a J \odot \sigma'(z^L).
\end{aligned}
$$

Druhá rovnica spätnej propagácie SP2 vyjadruje ako závisí vektor chýb v l-tej vrstve od vektora chýb v $(l+1)$-vrstve. Pri odvodení sa znova vychádza z definície chyby a rovnosť sa upraví na požadovaný tvar

$$\delta_j^l = \frac{\partial J}{\partial z_j^l} = \sum_k \frac{\partial J}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1}.$$

K vyjadreniu $\frac{\partial z_k^{l+1}}{\partial z_j^l}$ je vhodné sa pozrieť na z_k^{l+1} :

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} \sigma(z_j^l) + b_k^{l+1}.$$

Zderivovaním podľa z_j^l dostaneme

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} \sigma'(z_j^l).$$

Následnou spätnou substitúciou dostávame požadovaný tvar

$$\delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} \sigma'(z_j^l),$$
$$\delta^l = (w^{l+1})^\top \delta^{l+1} \odot \sigma'(z^l).$$

Tretia rovnica spätnej propagácie definuje vzťah medzi chybou a zmenou nákladovej funkcie pri zmene excitačnej hranice

$$\frac{\partial J}{\partial b_j^l} = \frac{\partial J}{\partial z_j^l} \frac{\partial z_j^l}{\partial b_j^l} = \delta_j^l \cdot 1 = \delta_j^l.$$

Posledná rovnica vyjadruje závislosť nákladovej funkcie of zmeny váh

$$\frac{\partial J}{\partial w_{jk}^l} = \frac{\partial J}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} = \frac{\partial J}{\partial a_j^l} \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_j^l} =$$

$$= \sum_i \left[\frac{\partial J}{\partial z_i^{l+1}} \frac{\partial z_i^{l+1}}{\partial a_j^l} \right] \frac{\partial a_j^l}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} =$$

$$= \sum_i \left[\delta_i^{l+1} w_{ij}^{l+1} \right] \sigma'(z_j^l) a_k^{l-1} = a_k^{l-1} \delta_j^l.$$

3.1.1 Špeciálny tvar rovníc spätnej propagácie

Pre neurónovú sieť so zvolenou nákladovou funkciou J a aktivačnou funkciou σ vo výstupnej vrstve nadobúda prvá rovnica spätnej propagácie nasledovný tvar:

$$SP1 : \delta_i^L = \left[\frac{y^i}{a^i} - \frac{1 - y^i}{1 - a^i} \right] a^i (1 - a^i) = y^i - a^i,$$

$$\delta^L = Y - a.$$

Zvolením $tanh(x)$ ako aktivačnú funkciu vo skrytých vrstvách nadobúda druhá rovnica spätnej propagácie nasledujúci tvar:

$$SP2 : \delta^l = (w^{l+1})^\top \cdot \delta^{l+1} \odot (1 - a(z^l)^2) = (w^{l+1})^\top \cdot \delta^{l+1} \odot (1 - \tanh(z^l)^2),$$

pričom sme využili vzťahy $a(x) = \tanh(x)$ a $[\tanh(x)]' = 1 - \tanh(x)^2$ z lemy 1.3.

Rovnice $SP3$ a $SP4$ ostávajú nezmenené.

$$SP3 : \frac{\partial J}{\partial b^l} = \sum_{i=1}^{m} \delta^l,$$
$$SP4 : \frac{\partial J}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j.$$

3.2 Vylepšenie neurónovej siete

3.2.1 Úvodná voľba parametrov

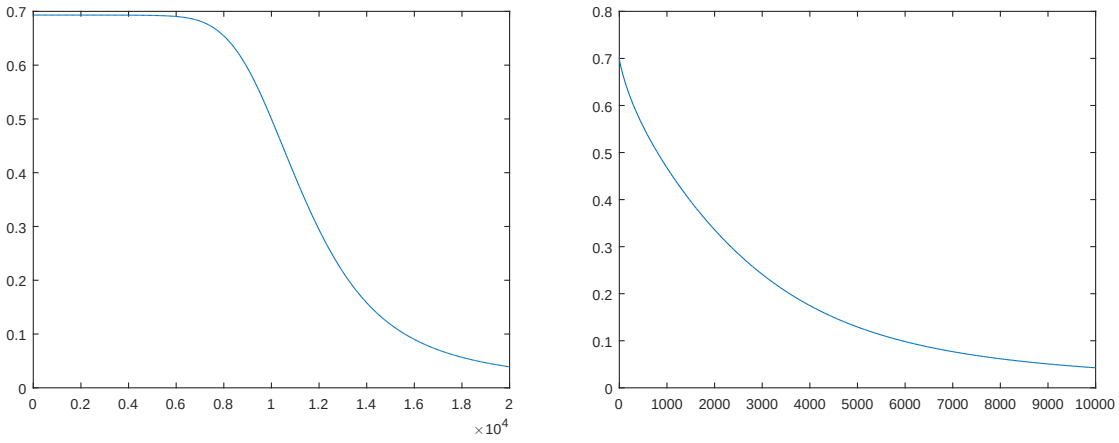
Na spustenie neurónovej siete je potrebné inicializovať počiatočné váhy takým spôsobom, aby následné učenie sa váh nebolo príliš pomalé. Ak by boli váhy zle zvolené, ľahko by sa mohlo stať, že vážený vstup $z = w^\top x + b$ by dosahoval hodnotu príliš vzdialenú od intervalu $\langle -1, 1 \rangle$, čo by spôsobilo, že funkcia $a = f(z)$ by v prípade použitia aktivačnej funkcie typu *sigmoid* alebo *hyperbolický tangens* nadobúdala hodnotu veľmi blízku asymptotickým hodnotám. To následné spôsobí veľmi pomalé učenia sa váh.

Jednou z možností ako tento problém vyriešiť, je voliť každú jednu váhu náhodne z nasledujúceho rozdelenia [8]

$$w^i \sim N(0, \frac{1}{n}), \tag{4}$$

kde n prestavuje počet váh v danej vrstve.

Na nasledujúcom obrázku Obr. 20 je znázornené porovnanie vývoja nákladovej funkcie pre rôzne zvolené počiatočné váhy.



Obr. 20: Vývoj nákladovej funkcie J pre náhodne zvolené počiatočné váhy (vľavo) a vhodne zvolené počiatočné váhy (vpravo). Zdroj: vlastné spracovanie.

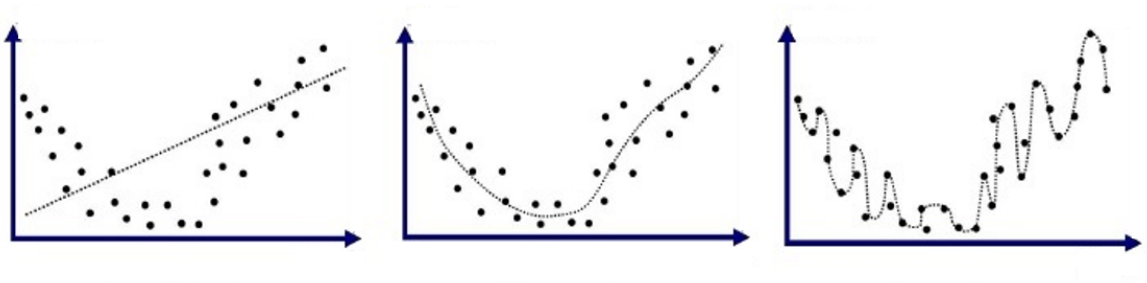
3.2.2 Regularizácia

Úspešne natrénovaná neurónová sieť by mala čo najlepšie klasifikovať trénovacie aj testovacie dáta. Správny model by sa mal naučiť z dát, ktoré mu boli poskytnuté tak aby vedel následne správne klasifikovať aj nové dáta, s ktorými sa predtým ešte nestretol. Na základe toho môžeme modely rozdeliť do 3 kategórií:

1. nedostatočný model,
2. prekomplikovaný model,
3. dobrý model.

Nedostatočný model je príliš jednoduchý na to, aby bol schopný modelovať hoci len trénovacie dáta. Ak model nie je schopný dostatočne dobre reprezentovať trénovacie dáta, tým pádom nebude schopný reprezentovať ani testovacie dáta. Príkladom je klasifikácia nelineárnych dát pomocou lineárneho modelu.

Model, ktorý síce už dobre klasifikuje trénovaciu sadu dát, ale testovaciu už nie, je pravdepodobne príliš komplikovaný. Miesto toho, aby mal nastavené váhy vhodné aj pre iné sady dát, zachytil v testovacích sadách šum. Situácia pri ktorej sa po určitom množstve iterácií prestane zlepšovať klasifikácia testovacích dát, ale klasifikácia tréno-



Obr. 21: Grafické znázornenie nedostatočného (vľavo), vhodného (v strede) a prekomplikovaného (vpravo) modelu. Zdroj: vlastné spracovanie.

vacích dát sa zlepšuje, sa nazýva pretrénovanie (z anglického *overfitting*). Pretrénovanie je častým problémom pri trénovaní neurónových sietí. Naštastie existuje veľa spôsobov ako pretrénovanie obmedziť, ktorých podrobný výklad možno nájsť napríklad v [12].

V tejto práci sú spomenuté iba tie najznámejšie z nich [8, 12].

Najjednoduchším riešením je zväčšenie trénovacej sady dát. Ak neurónová sieť pracuje s väčším množstvom dát, je pre ňu náročnejšie sa pretrénovať. Veľakrát však nie je jednoduché zväčšiť trénovaciu sadu. Získanie nových dát môže byť finančne náročné, dokonca niekedy to nie je ani možné. V takýchto prípadoch sa používajú iné metódy.

Ďalšou metódou je takzvané predčasné ukončenie (z anglického *early stopping*). Počas trénovania váh na trénovacej sade sa vyhodnocuje úspešnosť klasifikácie na testovacej sade. Keď prestane táto úspešnosť narastať, trénovanie neurónovej siete sa predčasne ukončí. Pri tomto prístupe si však treba dať pozor na to, aby trénovanie nebolo zastavené príliš skoro. Niekedy úspešnosť klasifikácie chvíľu stagnuje kým začne znova stúpať. Preto je vhodné zároveň sledovať ako sa vyvíja hodnota nákladovej funkcie pre testovaciu sadu. Ak stále klesá, je vhodnejšie pár iterácií počkať, či nedochádza k zlepšeniu pri klasifikácii.

Ďalšou možnosťou ako zabrániť neurónovej sieti aby sa pretrénovala sú $L1$ a $L2$ regularizácie. Myšlienka spočíva v penalizovaní nákladovej funkcie v prípade veľkých váh. K pôvodnej nákladovej funkcii sa pridá ďalší člen, ktorý bude penalizovať veľ-

kosť jednotlivých váh. Úprava nákladovej funkcie pre jednotlivé druhy regularizácie je nasledovná:

$$L1 : J_{L1} = J + \frac{\lambda}{m} \sum_{w} |w|,$$
$$L2 : J_{L2} = J + \frac{\lambda}{2m} \sum_{w} w^2,$$

kde $\lambda \in R$ je regularizačný parameter. Čím bude mať λ väčšiu hodnotu, tým bude veľkosť váh viacej vplývať na celkovú hodnotu nákladovej funkcie. Ako to však vplýva na nákladovú funkciu? Keďže cieľom je jej minimalizácia, tak veľké hodnoty jednotlivých váh budú pôsobiť penalizačne na hodnotu nákladovej funkcie.

Aký vplyv bude mať zmena nákladovej funkcie na optimalizáciu váh a excitačnej hranice?

$$L1 : \frac{dJ_{L1}}{dw} = \frac{dJ}{dw} + \frac{\lambda}{m} sgn(w), \qquad \frac{dJ_{L1}}{db} = \frac{dJ}{db},$$
$$L2 : \frac{d_{L2}}{dw} = \frac{dJ}{dw} + \frac{\lambda}{m} w, \qquad \frac{dJ_{L2}}{db} = \frac{dJ}{db}.$$

Čo ovplyvní zmenu váh nasledovne:

$$L1 : w \to w - \alpha \frac{dJ}{dw} - \frac{\alpha\lambda}{m} sgn(w),$$
$$L2 : w \to w - \alpha \frac{dJ}{dw} - \frac{\alpha\lambda}{m} w = \left(1 - \frac{\alpha\lambda}{m}\right) w - \alpha \frac{dJ}{dw},$$

miesto pôvodnej zmeny:

$$w \to w - \alpha \frac{dJ}{dw}.$$

3.3 Spustenie zložitej neurónovej siete

Algoritmus fungovania viacvrstvovej neurónovej siete:

1. načítanie vstupných dát a prvotná inicializácia váh pomocou vzťahu (4),
2. pre vopred zvolený počet iterácií sa vykonajú nasledovné kroky:
   - dopredná propagácia,

- spätná propagácia,
- úprava váh na základe gradientnej metódy,

3. otestovanie úspešnosti klasifikácie na trénovacích a testovacích dátach pomocou finálnych váh.

Pod doprednou propagáciou rozumieme postupný výpočet aktivačných vektorov naprieč vrstvami pomocou jednotlivých váh a biasov

$$X \to Z^1 \to a^1 \to Z^2 \to a^2 \cdots \to a^L.$$

3.4 Aplikácia zložitej neurónovej siete na binárnu sadu dát

Keďže sa jednoduchá neurónová sieť (bez skrytej vrstvy) ukázala ako nedostatočná, rozhodli sme sa pridať skryté vrstvy a pozorovať, či dôjde k zlepšeniu klasifikácie. Začali sme jednou skytou vrstvou a optimalizáciou klasickým gradientom. Výsledky uvádzame v nasledujúcej Tabuľke 4, pričom m predstavuje veľkosť trénovacej sady dát. Neurónovú sieť označujeme skratkou NN (z anglického *Neural Network*).

Tabuľka 4: Trénovanie NN s 1 skrytou vrstvou a 1 000 neurónmi v skrytej vrstve

α	m	J_{tren}	ÚK$_{tren}$	ÚK$_{test}$
0.001	2 000	0.32147	0.91150	0.605
0.005	5 000	0.215208	0.934600	0.6230

Zdroj: vlastné spracovanie.

Zväčšovanie trénovacej sady spôsobovalo predlžovanie času potrebného na dostatočné natrénovanie dát pomocou klasickej gradientnej metódy, preto sme sa rozhodli ďalej trénovať neurónovú sieť pomocou stochastickej gradientnej metódy. Počet iterácií sme označili MPI. Hoci je počet iterácií nastavený na oveľa nižšiu hodnotu, keďže využívame metódu stochastického gradientu, tak počas každej iterácie sa váhy optimalizujú niekoľko krát. Tento počet závisí od veľkosti trénovacej sady a veľkosti vzorky, s ktorou sa pracuje pri metóde stochastického gradientu. Výsledky takéhoto trénovania uvádzame v nasledujúcej Tabuľke 5. Ako môžeme z Tabuľky 5 vidieť, úspešnosť

Tabuľka 5: Trénovanie NN s 1 skrytou vrstvou, 200 neurónmi v skrytej vrstve a veľkosťou vzorky 50.

α	m	MPI	J_{tren}	ÚK$_{tren}$	ÚK$_{test}$
0.005	8 000	200	0.17353	0.9631250	0.6620
0.001	8 000	400	0.35795	0.871250	0.6490

Zdroj: vlastné spracovanie.

klasifikácie stúpla na 66 %.

Doteraz sme pracovali s obrázkami rozmerov 64×64 pixelov. Keďže každý pixel pozostával z 3 hodnôt farebného spektra, tak vstupná vrstva pozostávala z $64 \cdot 64 \cdot 3 = 12\ 288$ neurónov K zrýchleniu trénovania neurónovej siete a možnosti trénovať neurónovú sieť v rozumnom čase aj s väčším množstvom dát sme sa rozhodli zmeniť rozmery obrázkov na 32×32 pixelov. Pri takýchto rozmeroch mala vstupná vrstva iba $32 \cdot 32 \cdot 3 = 3\ 072$ neurónov. Zmenšením rozmerov sme mohli zväčšiť trénovaciu sadu, pri zachovaní rozumnej časovej náročnosti. V nasledujúcej Tabuľke 6 sme uviedli nadobudnuté výsledky, kde N_i značí počet neurónov v i-tej vrstve. Úspešnosť klasifikácie stúpla

Tabuľka 6: Trénovanie NN s 1 skrytou vrstvou, veľkosťou trénovacej sady 20 000 a veľkosťou vzorky 100.

α	MPI	N_2	J_{tren}	ÚK$_{tren}$	ÚK$_{test}$
0.01	200	1 000	0.4765	0.7740	0.6840
0.001	500	500	0.5652	0.7108	0.6780

Zdroj: vlastné spracovanie.

na 68%, takže zmenšenie rozmerov nielen že neuškodilo ale vďaka možnosti pracovať s väčšou sadou dát sme boli schopní dosiahnuť lepšie výsledky. Následne sme sa rozhodli ešte otestovať pridanie ďalších skrytých vrstiev. Výsledky sú uvedené v Tabuľke 7 a Tabuľke 8. Najlepšiu úspešnosť klasifikácie sa nám podarilo dosiahnuť s neurónovou sieťou s 2 skrytými vrstvami. Úspešnosť bola na úrovni 69%. Hoci trénovanie bolo na-

Tabuľka 7: Trénovanie NN s 2 skrytými vrstvami, veľkosťou trénovacej sady 20 000, veľkosťou vzorky 50 a rozmermi obrázkov 32×32 pixelov.

α	MPI	$N_2 \times N_3$	J_{tren}	ÚK$_{tren}$	ÚK$_{test}$
0.0001	500,	800×64	0.5110	0.6549	0.6390
0.01	200	800×64	0.0535	0.9680	0.6900

Zdroj: vlastné spracovanie.

Tabuľka 8: Trénovanie NN s 3 skrytými vrstvami,veľkosťou trénovacej sady 20 000,veľkosťou vzorky 40 a rozmermi obrázkov 32×32.

α	MPI	$N_2 \times N_3 \times N_4$	J_{tren}	ÚK$_{tren}$	ÚK$_{tren}$
0.004	500	$800 \times 500 \times 64$	0.0014	1	0.6840

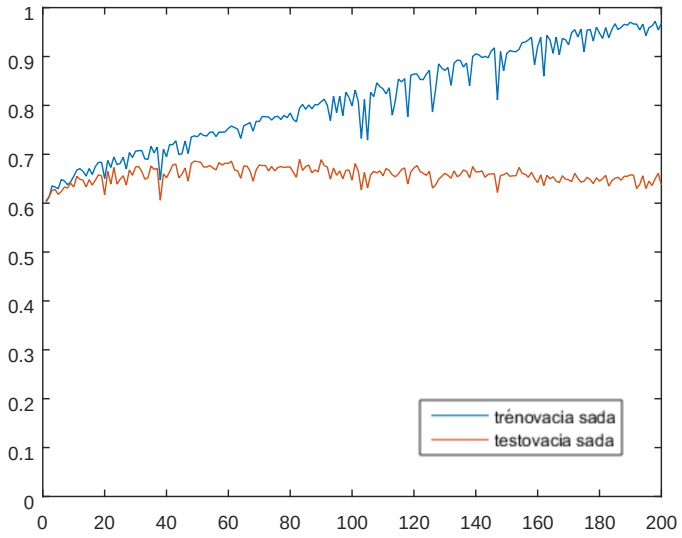
Zdroj: vlastné spracovanie.

stavené na 500 iterácií maximálna úspešnosť testovacej sady bola zaznamenaná po 83 iteráciách. Priebeh tohto iteračného procesu je možné vidieť aj na Obr. 22. Pridanie 3. skrytej vrstvy už zlepšenie neprinieslo. Ako je možné vidieť z Obr. 22 maximálna úspešnosť bola nadobudnutá po iterácii 83 a následne sa veľmi pomaly zhoršovala, hoci úspešnosť klasifikácie trénovacej sady neustále stúpala.

Pomocou klasických neurónových sietí s niekolkými skytými vrstvami sa nám podarilo zlepšiť klasifikáciu z 50% na 69%.

3.5 Aplikácia zložitej neurónovej siete na sadu dát s viacerými triedami

Pre druhú sadu dát sa nám podarilo pomocou neurónovej siete dosiahnuť maximálnu úspešnosť klasifikácie 26%. Keďže bola táto úspešnosť pomerne nízka, rozhodli sme sa otestovať dáta aj na neurónovej sieti so skrytými vrstvami. Takisto sme sa rozhodli pridať $L2$ regularizáciu reprezetovanú parametrom λ a pozorovať, či spôsobí zlepšenie klasifikácie. Trénovanie priebiehalo pomocou metódy stochastického gra-



Obr. 22: Úspešnosť klasifikácie trénovacej a testovacej sady dát. Zdroj: vlastné spracovanie.

dientu pričom veľkosť vzorky bola 50. Maximálny počet iterácií bol nastavený na 500, ale ak hodnoty nákladovej funkcie a úspešnosti klasifikácie pre testovaciu sadu prestali ukazovať zlepšenie, tak bolo trénovanie siete predčasne ukončené.

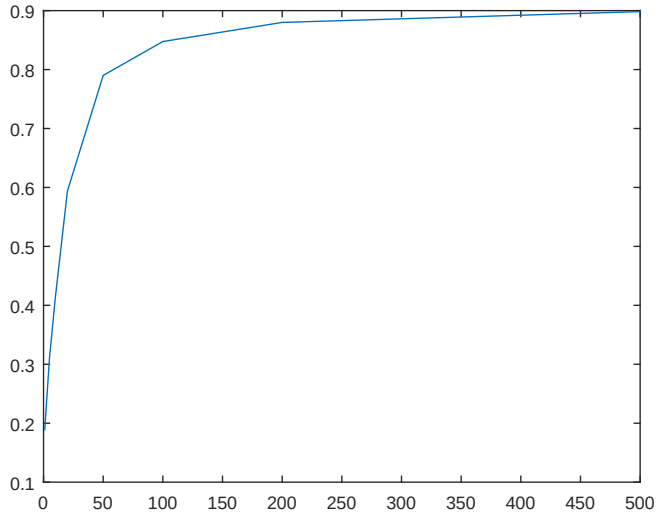
Najprv sme sa zamerali na zistenie minimálneho počtu neurónov v skrytej vrstve na dostatočnú klasifikáciu. Zlepšovanie úspešnosti klasifikácie je možné vidieť na Obr. 23.

Úspešnosť klasifikácie neustále stúpala zo zvyšujúcim sa množstvom neurónov v skrytej vrstve. Preto sme sa rozhodli ďalej používať 500 neurónov v skrytej vrstve.

Tabuľka 9: Porovnanie trénovania neurónovej siete s 1 skrytou vrstvou pre rôzne hodnoty parametrov.

m_{tren}=40 000, m_{test}=10 000, vzorka=50						
α	λ	N_2	$minJ_{tren}$	$minJ_{test}$	$maxUK_{tren}$	$maxUK_{test}$
0.005	0	500	0.5818(344)	1.0464(344)	0.9142(352)	0.8316(344)
0.005	0.5	500	0.5997(404)	1.0570(372)	0.9119 (404)	0.8295(387)
0.005	5	500	0.6071(430)	1.1612(369)	0.9105(430)	0.8153(386)

Zdroj: vlastné spracovanie.



Obr. 23: Grafické znázornenie úspešnosti klasifikácie vzhľadom na počet neurónov v skrytej vrstve. Zdroj: vlastné spracovanie.

Pomocou neurónovej siete s 1 skrytou vrstvou sa nám podarilo dosiahnuť úspešnosť klasifikácie skoro 83 %. To je signifikantný nárast v porovnaní s neurónovou sieťou bez skrytej vrstvy, kde bola úspešnosť klasifikácie približne 26%. Avšak pridaná regularizácia nespôsobila žiadne zlepšenie. Výsledky sa nachádzajú Tabuľke 9, pričom uvádzame minimálne hodnoty pre nákladové funkcie a maximálne hodnoty pre úspešnosť klasifikácie. Číslo v zátvorke značí, v ktorej iterácii sa nadobudla daná hodnota.

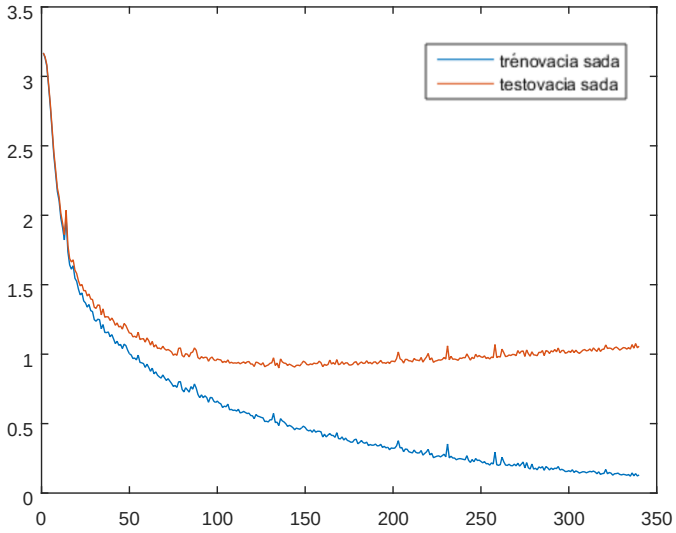
Následne sme sa rozhodli pridať ďalšiu skrytú vrstvu a pozorovať či to pomôže zlepšiť klasifikáciu. Výsledky uvádzame v Tabuľke 10.

Tabuľka 10: Porovnanie trénovania neurónovej siete s 2 skrytými vrstvami pre rôzne hodnoty parametrov

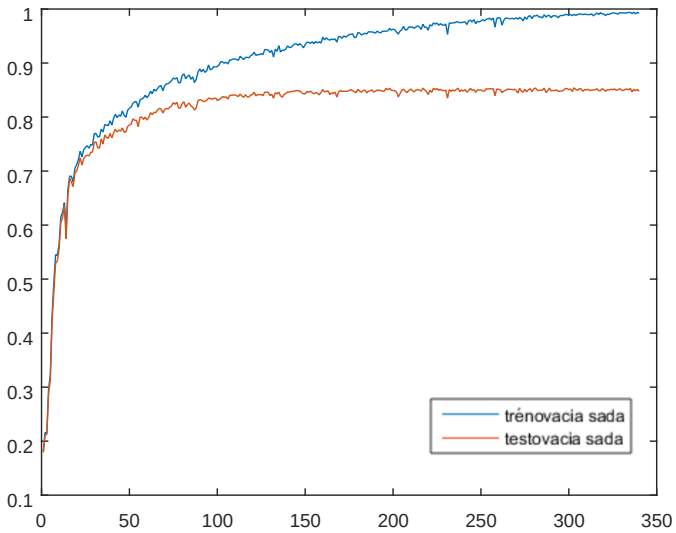
m_{tren}=40 000, m_{test}=10 000, vzorka=50						
α	λ	$N_2 \times N_3$	$minJ_{tren}$	$minJ_{test}$	maxÚK$_{tren}$	maxÚK$_{test}$
0.01	0	500x500	0.2187(217)	0.9150(135)	0.9803(217)	0.8522(217)
0.01	5	500x500	0.1216(335)	0.9005(135)	0.9941(337)	0.8542(303)

Zdroj: vlastné spracovanie.

Pridanie druhej skrytej vrstvy pomohlo zlepšiť klasifikáciu o približne 2,5%. Regularizácia avšak nemala žiaden vážny vplyv na zlepšenie. Na Obr. 24 a Obr. 25 je znázornený vývoj nákladovej funkcie a úspešnosť klasifikácie pre trénovacie a testovacie dáta pre sieť s najlepšími výsledkami.



Obr. 24: Grafické znázornenie nákladovej funkcie. Zdroj: vlastné spracovanie.



Obr. 25: Grafické znázornenie úspešnosti klasifikácie. Zdroj: vlastné spracovanie.

Vytvorením neurónovej siete s 2 skrytými vrstvami sa nám podarilo zvýšiť úspešnosť klasifikácie z 10% na 85%. Takže obyčajná neurónová sieť sa dá považovať za

vhodný algoritmus na tento typ klasifikácie.

Záver

Neurónové siete predstavujú veľmi výkonné algoritmy na riešenie niektorých druhov úloh. Medzi najznámejšiu oblasť, kde našli neurónové siete využitie patrí klasifikácia obrázkov. Na tento druh klasifikácie sme sa v tejto práci zamerali aj my.

V prvej kapitole sme vysvetlili čo predstavuje umelý neurón a aký má súvis s biologickým neurónom. Najpodstatnejšou častou umelého neurónu je aktivačná funkcia, ktorej sme sa následne podrobnejšie venovali. Spomenuli sme najznámejšie aktivačné funkcie, medzi ktoré patria binárna funkcia, sigmoid funkcia, hyperbolický tangens, ReLu funkcia a ako medzi sebou navzájom súvisia. Záver kapitoly sme venovali architektúram a jednotlivým druhom neurónových sietí.

Druhú kapitolu sme rozdelili na 2 časti. Prvá časť bola teoretická, v ktorej sme definovali logistickú regresiu a jej súvis s jednoduchou neurónovou sieťou. Následne sme sa zaoberali hľadaním vhodnej konvexnej nákladovej funkcie, ktorá nám umožnila zlepšovať klasifikáciu dát pomocou neurónovej siete. To, že naša nákladová funkcia je konvexná, sme v kapitole 2 podrobne dokázali. Vďaka konvexnosti nákladovej funkcie sme pri optimalizácii váh a biasu mohli využiť gradientnú metódu a následne aj jej stochastickú alternatívu. Vylepšovanie prebiehalo pomocou zmien váh a biasu takým spôsobom, aby nákladová funkcia klesala. Druhá časť kapitoly bola praktická. Využili sme nadobudnuté informácie a zostrojili jednoduchú neurónovú sieť bez skrytých vrstiev. Na tejto neurónovej sieti sme následne nechali natrénovať 2 sady dát. Prvá sada pozostávala z obrázkov psov a mačiek, takže sa jednalo o binárnu klasifikáciu. Úspešnosť klasifikácie sa nám podarilo zvýšiť z 50 % na približne 60%. Experimentovali sme s rôznymi nastaveniami parametra α ako aj s veľkosťou trénovacej sady, no zlepšenie klasifikácie nenastalo. Avšak úspešnosť klasifikácie 60% vzhľadom na primitívnosť neurónovej siete je podľa nás pozitívny výsledok. Druhá sada dát predstavovala snímky čísiel. Tam sa nám podarilo zvýšiť úspešnosť klasifikácie z 10% na približne 26%.

Tretia kapitola sa sústredila na zložitejšie neurónové siete, ktoré obsahujú skryté vrstvy. Na začiatku poslednej kapitoly sme sa venovali teórii spätnej propagácie. Ďalej sme sa v tretej kapitole zaoberali spôsobom, ako zlepšiť fungovanie neurónovej siete. Medzi základnými vylepšeniami neurónovej siete sme spomenuli napríklad vhodnú

úvodnú inicializáciu váh a biasov, ktorá slúžila k zrýchleniu trénovania neurónovej siete. Aby sme zabránili pretrénovaniu neurónovej siete, predstavili sme 2 známe techniky. Prvou bolo takzvané predčasné zastavenie trénovania, ak sa úspešnosť klasifikácie testovacej sady dát prestala zlepšovať. Druhou možnosťou boli $L1$ a $L2$ regularizácie, ktoré upravili nákladovú funkciu tak, aby penalizovala veľké hodnoty váh. Záver kapitoly sa venuje aplikácii neurónových sietí so skrytými vrstvami na vyššie spomenuté sady dát. Experimentovali sme s rôznymi nastaveniami siete a podarilo sa nám dosiahnuť nasledujúce výsledky.

Pri prvej sade dát, teda klasifikácii obrázkov psov a mačiek, sa nám podarilo zvýšiť úspešnosť klasifikácie na 69 % pri sieti s 2 skrytými vrstvami. Avšak ďalším pridávaním vrstiev a neurónov sa nám nepodarilo dosiahnuť lepšiu klasifikáciu. Vzhľadom na dosiahnuté výsledky môžeme predpokladať, že takýto druh neurónovej siete nie je dostačujúci pre klasifikáciu takejto komplexnej sady dát.

Pri druhej sade malo pridanie skrytých vrstiev lepšie výsledky. Po rôznych experimentoch a nastaveniach sa nám podarilo zvýšiť úspešnosť klasifikácie číslic až na 85 %. Tu môžeme prehlásiť, že v prípade tejto sady dát bola neurónová sieť pomerne vhodným algoritmom na správnu klasifikáciu čísiel.

Na záver by sme skonštatovali, že sa nám podarilo splniť všetky vopred stanovené ciele.

Zoznam použitej literatúry

[1] BOYD, S., VANDENBERGHE, L.: *Convex Optimization*, Cambridge University Press, New York, 2004

[2] GOODFELLOW, I., BENGIO, Y., COURVILLE, A.: *Deep Learning*, MIT Press, Cambridge, 2016, dostupné na internete (02.05.2019): http://www.deeplearningbook.org

[3] GORDON, G., TIBSHIRANI, R.: *Optimization*, elektronické študijné materiály, School of Computer Science, Carnegie Mellon University, 2012, dostupné na internete (02.05.2019): https://www.cs.cmu.edu/∼ggordon/10725-F12/scribes/10725_Lecture5.pdf

[4] HORN, R., A., JOHNSON, CH., R.: *Matrix Analysis*, Cambridge University Press, Cambrigde, 1985

[5] JANOŠKOVÁ, M.: *Siete typu NARX pre predikciu dát*, bakalárska práca, STU FEI, Bratislava, 2011,

[6] KAGGLE: *Dogs vs. Cats data*, dostupné na internete (13.01.2019): www.kaggle.com/c/dogs-vs-cats/data

[7] KRIESEL, D.: *A Brief Introduction to Neural Networks*, 2007, dostupné na internete (13.01.2018): www.dkriesel.com/_media/science/neuronalenetze-en-zeta2-1col-dkrieselcom.pdf

[8] NIELSEN, M. A.: *Neural Networks and Deep Learning*, Determination Press, Cambridge, 2015

[9] POHL, P.: *Rozpoznávanie textu pomocou neurónovej siete*, diplomová práca, Žilinská univerzita v Žiline, Žilina, 2006

[10] RUMELHART, D. E., HINTON, G. E.: *Learning representations by back-propagating errors*, Nature vol. 323, 1986, dostupné na internete (02.05.2019): http://www.cs.toronto.edu/∼hinton/absps/naturebp.pdf

[11] SHAI, S., S., SHAI, B. D.: *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, Cambridge, 2014, dostupné na internete (02.05.2019): http://www.cs.huji.ac.il/∼shais/UnderstandingMachineLearning/copy.html

[12] SRIVASTAVA, N. a spol.: *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research 15 (2014) 1929-1958, Toronto, 2014, dostupné na internete (12.05.2019): http://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_ content=buffer79b43&utm_ medium=social&utm_source=twitter.com&utm_ ampaign=buffer

[13] TRNOVSKÁ, M.: *Motivácia k metódam voľnej optimalizácie*, elektronické študijné materiály, FMFI UK, Bratislava, 2018, dostupné na internete (13.01.2018): http://pc2.iam.fmph.uniba.sk/institute/trnovska/MVOprednaskaAppl.pdf

[14] TRNOVSKÁ, M.: *Logistická regresia*, zbierka úloh, FMFI UK, Bratislava, 2018, dostupné na internete (13.01.2018): http://pc2.iam.fmph.uniba.sk/institute/trnovska/logregresia.pdf

[15] TRNOVSKÁ, M., HAMALA, M.: *Nelineárne programovanie*, EPOS, Bratislava, 2013

[16] TRNOVSKÁ, M.: *Metódy voľnej optimalizácie*, elektronické študijné materiály, FMFI UK, Bratislava, 2018

[17] WANG, T. a spol.: *Reading Digits in Natural Images with Unsupervised Feature Learning*, NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011, dostupné na internete (12.05.2019): http://ufldl.stanford.edu/housenumbers/