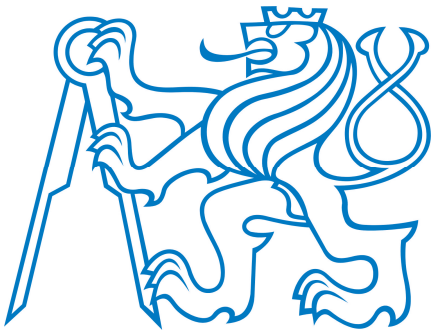
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta strojní – Ústav přístrojové a řídicí techniky



BAKALÁŘSKÁ PRÁCE

KLASIFIKACE RAKOVINY KŮŽE POMOCÍ ALGORITMŮ STROJOVÉHO UČENÍ

SKIN CANCER CLASSIFICATION USING MACHINE LEARNING ALGORITHMS

Martin Křeček

2021

Prohlašuji, že jsem tuto práci vypracoval samostatně s použitím literárních zdrojů a informací, které cituji a uvádím v seznamu použité literatury a zdrojů.

Datum: . . . .

Podpis

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Ing. Matoušovi Cejnekovi, Ph.D., za odborné vedení, cenné rady a čas, který mi věnoval po celou dobu psaní mé závěrečné práce.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Křeček** Jméno: **Martin** Osobní číslo: **482608**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Klasifikace rakoviny kůže pomocí algoritmů strojového učení

Název bakalářské práce anglicky:

Skin cancer classification using machine learning algorithms

Pokyny pro vypracování:

Zásady pro vypracování:
1. Nastudovat současný stav dané problematiky a popsat aktuální možnosti řešení.
2. Napsat teoretickou část k tématům týkající se této práce.
3. Vytvořit, natrénovat a otestovat klasifikačního model neuronové sítě, který bude použit v aplikaci.
4. Vytvořit webovou aplikaci s implementovaným modelem neuronové sítě. Aplikace bude schopna predikovat procentuální shodu se sedmi kategoriemi kožních lézí z vlastně pořízené fotografie.

Seznam doporučené literatury:

[1] MÜLLER Guido. Introduction to Machine Learning with Python: A Guide for Data Scientists 1st Edition. O'Reilly Media, 2016, ISBN 978-1449369415.
[2] BURKOV Andriy. The Hundred-Page Machine Learning Book. Andriy Burkov, 2019, ISBN 1999579518.
[3] GÉRON Aurélien. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 1st Edition. O'Reilly Media, 2017, ISBN 978-1491962299.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Matouš Cejnek, Ph.D., U12110.3

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **30.04.2021** Termín odevzdání bakalářské práce: **10.06.2021**

Platnost zadání bakalářské práce: _____

Ing. Matouš Cejnek, Ph.D.		prof. Ing. Michael Valášek, DrSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ _____

Datum převzetí zadání Podpis studenta

Anotace

Bakalářská práce popisuje problematiku rakoviny kůže v dnešní společnosti a snaží se o rozšíření snadné možnosti prevence do širší společnosti za využití klasifikace pomocí neuronových sítí. Práce popisuje aktuální informace o rakovině kůže a shrnuje dosavadní úspěchy v klasifikaci rakoviny kůže. Dále práce obsahuje základní informace o strojovém učení, tedy jeho historii, metody učení a druhy algoritmů s detailním zaměřením na neuronové sítě. Praktická část práce obsahuje vytvoření, natrénování a otestování prediktivního modelu neuronové sítě v jazyce Python. Pro vytvořený model je vytvořena webová aplikace v jazyce Python s implementovaným modelem neuronové sítě. Aplikace je schopna predikovat procentuální shodu se sedmi kategoriemi kožních lézí z vlastně pořízené fotografie.

Abstract

The bachelor's thesis describes the issue of skin cancer in today's society and seeks to extend the easy possibility of prevention to the wider society using the classification of neural networks. The work describes current information about skin cancer and summarizes the current successes in the classification of skin cancer. Furthermore, the work contains basic information about machine learning, ie its history, learning methods and types of algorithms with a detailed focus on neural networks. The practical part of the work contains the creation, training and testing of a predictive model of a neural network in Python. A web application in the Python language with an implemented neural network model is created for the created model. The application is able to predict the percentage agreement with seven categories of skin lesions from the photograph actually taken.

Obsah

1 Úvod 6
2 Strojové učení 9
2.1 Přehled 9
2.2 Historie 9
2.3 Vztahy s ostatními vědeckými obory 11
2.3.1 Umělá inteligence 11
2.3.2 Dolování dat 12
2.3.3 Statistika 12
2.4 Klíčové pojmy 12
2.4.1 Tréninková, validační a testovací data 12
2.4.2 Model a algoritmus 13
2.5 Metody strojového učení 14
2.5.1 Supervised Learning 14
Lineární regrese / Linear regression 14
Logistická regrese / Logistic regression 15
k-Nejbližších Sousedů / k-Nearest Neighbors (kNN) 16
Umělé Neuronové Sítě / Artificial Neural Networks (ANNs) 17
2.5.2 Unsupervised Learning 17
Shlukování / Clustering 17
Snížení Dimenzí / Dimensionality Reduction 18
2.5.3 Semi-supervised learning 18
2.5.4 Reinforcement learning 19
2.6 Umělé Neuronové Sítě / Artificial Neural Networks (ANNs) 19
2.6.1 Úvod 19
2.6.2 Perceptron 20
2.6.3 Hluboké versus Mělké učení 22
2.6.4 Přenosové funkce 25
2.6.5 Učení neuronové sítě / Účelová funkce a stochastický gradient 29
2.6.6 Konvoluční neuronové sítě 34
3 Praktická část 37
3.1 Data 38
3.2 Data preprocessing 39
3.3 Tvorba modelu 41
3.4 Výsledky 43
3.5 Tvorba aplikace 45
4 Závěr 48
Seznam použité literatury a zdrojů 51

Seznam použitého SW 57
Seznam příloh 58

1 Úvod

Za rok 2017 bylo v České republice evidováno 34 814 [1] případů rakoviny kůže, z toho 1 515 [1] se pacientům stalo smrtelných. Z celkového čísla 86 819 [2] pacientů diagnostikovanými se zhoubným nádorem a 27 320 [2] úmrtími se jedná o nejpočetnější diagnózu u nás. Počet případů rakoviny kůže v posledních letech strmě roste a i celosvětově se vyskytuje stále častěji. V přepočtu případů na 100 000 obyvatel se za rok 2008 jako národ nacházíme ve světovém žebříčku na poměrně vysokém 12. místě [1]. Mortalita se i přes zvyšující počet případu stabilizuje, ale díky pečlivé prevenci a včasnému odhalení zhoubného nádoru jsme schopni úmrtnost stále snižovat. Ve své práci se zabývám problematikou odhalovaní rakoviny kůže pomoci algoritmů strojového učení a neuronových sítí.

Cílem práce je vytvoření mobilní aplikace, která dokáže určit nejpravděpodobnější diagnózu kožní léze. Aplikace může sloužit lékařům či klinickým pracovníkům k rychlé klasifikaci pacientů s rizikovým nálezem, nebo jako prevence široké veřejnosti k včasné návštěvě zdravotnického zařízení. Základem aplikace je tzv. model. Model je soubor, naučený rozpoznat určité typy vzorů. V mém případe musí model rozpoznat jednotlivé typy kožních lézí. Pomocí algoritmů strojového učení a neuronových sítí lze model natrénovat na označených datech, klinických snímcích kožních lézí s označením o jaký typ diagnózy se jedná, a přichystat ho na nová data. K úspěšnému modelu je potřeba obrovské množství dat, proto se při vytváření modelů běžně využívá veřejně dostupných datových setů.

Ve své práci využiji volně přístupný datový set „The HAM10000“ zpřístupněný k události soutěže obrazové klasifikace, kterou pořádala mezinárodní organizace snímání kůže *ISIC, The Intrenationl Skin Imaging Collaboration*. Obsahuje přibližně 10 000 snímků kožních lézí a nabízí širokou škálu diagnóz. Datový set lze stáhnout z oficiálních stránek Harvardské nebo Cornellské Univerzity [3] a je dostupný i na jiných stránkách, například na stránce Kaggle.com [4]. Tato stránka slouží k publikování veřejně dostupných datových setů a kolaboraci jejich uživatelů. Kdokoliv má možnost stáhnout si libovolný dataset a použít ho ke své analýze, práci či projektu. Mnou zmíněný dataset je k dispozici již přes 2 roky a za tu dobu se našlo pár uživatelů, kteří se snažili o vytvoření funkčního modelu.

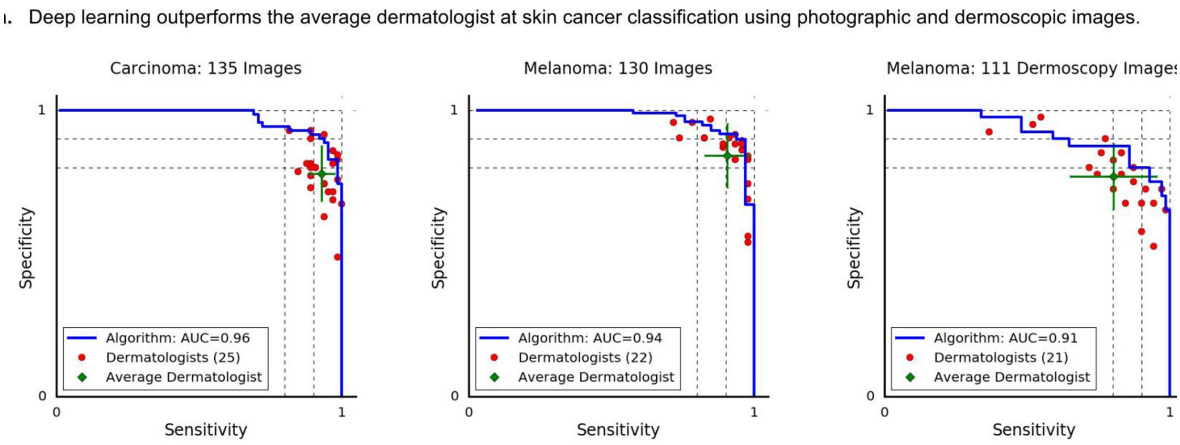
Mezi takové patří uživatel „Manu Siddhartha“ [5], který vytvořil funkční model s úspěšností kolem 77%. Uživatel si dataset nejprve náležitě připravil a správně setřídil, aby měly použité algoritmy strojového učení co největší účinek. Dále začal vytvářet model v jazyce Python, nejpoužívanějším programovacím jazyce v této oblasti, pomocí knihovny na vytváření neuronových sítí Keras, která pracuje na vrchu knihovny TensorFlow. Použil vlastní architekturu neuronové sítě, kterou vytvořil po dlouhém testování a rad z řad zkušenějších kolegů. Nyní mohl přejít k samotnému natrénování neuronové sítě a její vyhodnocení.

Datasetů s podobným problémem je dostupných několik. Za zmínku stojí například datový set "SIIM-ISIC Melanoma Classification" [7], zpřístupněný organizacemi SIIM (Society for Imaging Informatics in Medicine) a ISIC (International Skin Imaging Collaboration). Tyto organizace se zabývají vývojem a implementací informatiky v oblasti zpracování zdravotních snímků. Dataset obsahuje pouze snímky melanomů, nejagresivnějších kožních nádorů. Slouží k analýzám a studiím k včasnému odhalení tohoto typu nemoci, ale nenabízí potřebnou diverzitu. Datasety se snímky nádorů melanomu se řadí mezi nejpočetnější, ale většinou se liší pouze počtem či kvalitou snímků. Mezi další podobné se řadí dataset "Skin Cancer: Malignant vs. Benign" [8], obsahující snímky maligních (zhoubných) a benigních (nezhoubných) nádorů, a slouží k určení typu kožního nádoru. Tato sada je dostupná díky již zmíněné organizaci ISIC, ale též nenabízí přílišnou variabilitu. Zajímavým zdrojem může být archiv snímků organizace ISIC [9], který obsahuje cca 24 000 snímků a uživatel si může nastavovat jednotlivé parametry jako pohlaví a věk pacienta, velikost a typ léze či její tloušťku. Nevýhodou tohoto archivu je možnost zvolit a stáhnout pouze 300 snímků najednou, proto se jako lepší možnost nabízí použít vybrané datasety.

Klasifikací rakoviny kůže pomocí neuronových sítí se zabývají mimo jiné i některé odborné magazíny. Pozoruhodný článek *Dermatologist-level classification of skin cancer with deep neural networks* [10] publikovaný na začátku roku 2017 ve vědeckém týdeníku *nature* popisuje jednu z metod řešení této problematiky. Autoři této práce, profesoři a studenti doktorského studia ze Stanfordské Univerzity, použili k trénování neuronové sítě dataset tvořený 129 450 klinickými snímky kožních lézí včetně 3 374 dermoskopických snímků, skládající se celkem z 2 032 odlišných kožních chorob. Autoři článku použili pro vytvoření modelu jazyk Python a po zajištění snímků rozdělili dataset na tréninkový, validační a testovací set. Poté probíhala příprava dat, kdy je nutné odstranit nevhodné snímky. Mezi takové se řadí rozmazané či příliš malé fotografie. Autoři použili již vytvořenou architekturu konvoluční neuronové sítě od společnosti Google „Inception v3“ [11], trénovanou na více než milion snímcích. Nahradili poslední vrstvu sítě svým datasetem a trénováním pozměnili parametry sítě. K trénování, validaci i testování sítě byla použita softwarová knihovna TensorFlow. Výsledek demonstrovaný na třech vzorcích je vidět na Obr. 1. Vzorky tvoří 135 klinických snímků epidermálních chorob (65 maligních, 70 benigních), 130 klinických snímků melanocytů (33 maligních, 97 benigních) a 111 dermatoskopických snímků melanocytů (71 maligních, 40 benigních). Výkon sítě otestovali alespoň proti 21 certifikovaným dermatologům v otázce, zda je nález benigní nebo maligní. Úspěch neuronové sítě je v tomto případě měřen pomocí porovnávacích metrik, které jsou sensitivita a specificita:

$$sensitivita = \frac{true\ positive}{positive}$$

$$specificita = \frac{true\ negative}{negative}$$



Obr. 1: *Výsledek konvoluční neuronové sítě a dermatologů při klasifikaci kožních chorob [10].*

kde „true positive“ je množství správně určených maligních lézí, „positive“ je celkový počet maligních lézí, „true negative“ je množství správně určených benigních lézí a „negative“ je celkový počet benigních lézí. Každý červený bod na Obr. 1 reprezentuje hodnotu sensitivity a specificity jednotlivých dermatologů. Neuronová síť překonala dermatology, jejíchž hodnoty se nachází pod modrou křivkou, což je ve většině případů. Zelený bod signalizuje průměr všech dermatologů i se směrodatnou odchylkou. Plocha pod křivkou AUC (Area Under Curve) značí procentuální úspěšnost algoritmu, která je pro každý případ nad 91%. Článek dokazuje vysoký potenciál odhalování rakoviny kůže pomocí algoritmů strojového učení a neuronových sítí a možnost vyšší přesnosti při důkladnějším výzkumu.

2 Strojové učení

2.1 Přehled

Strojové učení, známé také pod svým anglickým názvem *Machine learning*, je podoblast umělé inteligence, která je jedna z mnoha větví počítačových věd. Zabývá se algoritmy, které se dokážou automaticky učit a zlepšovat na základě zkušeností. Principem strojového učení je vytvoření matematického modelu, který se na základě vzorku dat, známého jako *training data*, naučí tvořit předpovědi nebo rozhodnutí, aniž by k tomu byl přímo naprogramován [12][13][21]. Strojové učení se značně prolíná s oblastí statistiky, proto je také známé pod názvy „prediktivní analytika“ a „statistické učení“ [14]. Aplikace strojového učení se během posledních let stala všudypřítomnou součástí každodenního života. Od automatického doporučování filmů ke shlédnutí, přes přizpůsobené reklamy na základě zájmů uživatele, po rozpoznávání obličejů přátel na vašich fotkách nebo vyhledávání pomocí hlasu. Podobných aplikací algoritmů strojového učení je kolem nás nespočet a mnohokrát o nich nemáme nejmenší tušení [14].

Mimo komerční využití aplikací má strojové učení obrovský vliv na určité oblasti vědeckého výzkumu. Díky mnohem efektivnější práci s daty se algoritmy používají k pochopení vesmíru a hledání nových planet [15], objevování nových částic [16], analýze sekvencí DNA [17], nebo k odhalování a léčbě rakoviny [14][18]. Dále se učení hojně využívá v oblasti bezpečnosti. Představme si jedinou osobu, která sleduje několik bezpečnostních kamer. Těžká a nepříliš zábavná práce. Dnes mohou být kamerové systémy poháněny umělou inteligencí, která umožní rozpoznat zločin ještě před tím, než k němu dojde. Sledováním nestandardního chování osob může systém zhodnotit situaci a případně upozornit lidskou obsluhu [19][20]. Podobný princip odhalování nestandardního chování se využívá při detekci podvodů s kreditními kartami [21].

2.2 Historie

Pojem *Machine learning* poprvé použil americký průkopník v oblasti počítačových her a umělé inteligence Arthur L. Samuel v roce 1959 [22]. V článku „Some Studies in Machine Learning Using the Game of Checkers“, původně publikovaném v žurnálu IBM, popsal princip strojového učení následovně:

„Computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program.“ [23]

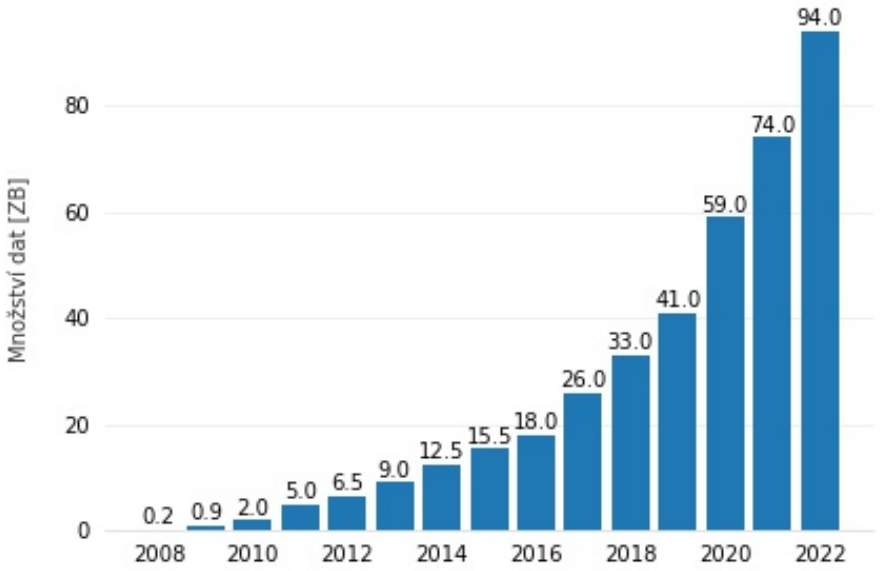
Volně přeloženo znamená, že počítač lze naprogramovat tak, aby se během hry Dáma učil a následně dokázal hrát lépe, než osoba, která program vytvořila. Arthurovi L. Samuelovi bývá většinou připisována jiná fráze:

„Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.“ [24]

Tedy, že strojové učení je studijní obor, který dává počítačům schopnost učit se bez nutnosti být tak explicitně naprogramován. Pravdou ovšem je, že tato definice nebyla v žádných z jeho publikací. Jedná se o parafrázi, která vznikla nejspíše v jiné publikaci později. Arthur L. Samuel ovšem svým článkem dokázal, že počítač opravdu lze naučit hrát lépe na základě dříve hraných her. Testy provedené na hře Dámy nebyly zdaleka stoprocentní, limitací byl ale výkon počítače. Pro pokus použil jeden z nejvýkonnějších počítačů své doby, IBM 704. Tento počítač nabízel výkon o velikosti 36 bitů a s potřebnou periferií zabral celou místnost [25]. Dnes jsou mobilní telefony, které téměř všichni nosíme neustále u sebe, řádově milionkrát výkonnější.

Ve stejný rok vytvořili Bernard Widrow a Marcian Hoff model „MADALINE“, který byl první neuronovou sítí aplikovanou na reálný problém. Model využíval adaptivní filtr, který eliminoval ozvěnu na telefonní lince a jeho princip se používá dodnes [26]. Během sedmdesátých a osmdesátých let po několika neúspěšných aplikací strojové učení nastala takzvaná *AI Winter* [27], neboli zima umělé inteligence. Mezi takové aplikace patří například nepovedený překladatel, který roku 1966 označil Poradní výbor pro automatické zpracování jazyků „ALPAC“ jako mnohem dražší, méně přesný a pomalejší, než překlad lidský [28]. Tento a další nepovedené aplikace ztlumili množství peněz vkládaných do výzkumu strojového učení a způsobili několikaletý propad zájmu o tuto disciplínu [29].

Návrat entusiasmu a optimismu okolo umělé inteligence se datuje na začátek devadesátých let. Zásadní zlom ve výzkumu umělé inteligence a strojového učení se považuje porážka ruského světového šachového mistra Garyho Kasparova v roce 1997. Gary Kaparov byl poražen počítačem *Deep Blue*, vytvořeným speciálně na hraní šachu společností IBM [29][30]. Se snižující se cenou počítačového vybavení a množství dat produkovaného společností se během 1. desetiletí 21. století investovalo do výzkumu a aplikací využívající algoritmy strojového učení velké množství prostředků. Raketový výstřel prožívá *Machine learning* v posledních několika letech, kdy na jeho algoritmy narazíme i několikrát denně. Za posledních deset let se množství veškerých dat pohybujících se mezi námi zvětšilo téměř 30 krát a během posledních 2 let bylo vyprodukováno více dat, než za celou historii lidstva na Obr. 2 [31].



Obr. 2: *Množství dat ve světě.*

Při pohledu na exponenciální nárůst množství dat proudících napříč světem lze očekávat rapidní vývoj umělé inteligence a strojového učení, za cílem co nejefektivněji zpracovat a rozumně použít co největší množství dostupných dat.

2.3 Vztahy s ostatními vědeckými obory

2.3.1 Umělá inteligence

Strojové učení se řadí mezi multidisciplinární oblasti vědy a je propojené s některými dalšími vědeckými disciplínami. Mezi hlavní patří vztah k umělé inteligenci, známé také pod anglickým názvem *Artificial Intelligence* a zkratkou „AI“. Umělá inteligence je obor informatiky zabývající se inteligencí strojů a jejich schopnostmi aplikovat kognitivní funkce, které se podobají chování lidské mysli, jako učení či řešení problémů [32]. Umělá inteligence se dělí na několik větví jako například *Natural language processing* neboli zpracování přirozeného jazyka, automatické plánování, pohyb a manipulace, sociální inteligence, obecná inteligence a nakonec i strojové učení. Větve se věnují jednotlivým přístupům a problematice umělé inteligence. Větev, zabývající se pohybem a manipulací, řeší možnosti využití umělé inteligence a jejich aplikací na pokročilé robotické ruce a průmyslové roboty. Na základě zkušeností se robot naučí, jak se pohybovat co nejefektivněji navzdory přítomnosti tření či prokluzu převodovky [32]. Strojové učení je elementární součástí umělé inteligence již od počátku oboru a zabývá se počítačovými algoritmy, které se na základě zkušeností dokážou zlepšovat [24]. Dlouhou historii se umělá inteligence a strojové učení považovala za jednotný obor. To se změnilo v devadesátých letech, kdy se strojové učení začalo oddělovat jako samostatný podobor. Hlavní odlišností byla změna cíle, který nyní nebyl dosažení umělé inteligence, ale schopnost vypořádat se s řešitelnými problémy

praktické povahy [36]. K dnešnímu dni se strojové učení stále připisuje jako podobor umělé inteligence, avšak objevují se již odlišné názory a někteří praktici dnes považují umělou inteligenci a strojové učení za rozdílné obory.

2.3.2 Dolování dat

Dolování dat, také známé pod svým anglickým názvem *Data mining*, je multidisciplinární obor počítačových věd a statistiky. Hlavním cílem této disciplíny je získávání informací inteligentními metodami z datových sad a přetvořit danou informaci do srozumitelné struktury pro další použití [37]. Častou chybou je domněnka, že obor se zabývá získáváním dat, jak napovídá název. Získáváním dat se ovšem tento obor vůbec nezabývá a správný název této disciplíny by možná měl být „Dolování informací z dat“ [38].

2.3.3 Statistika

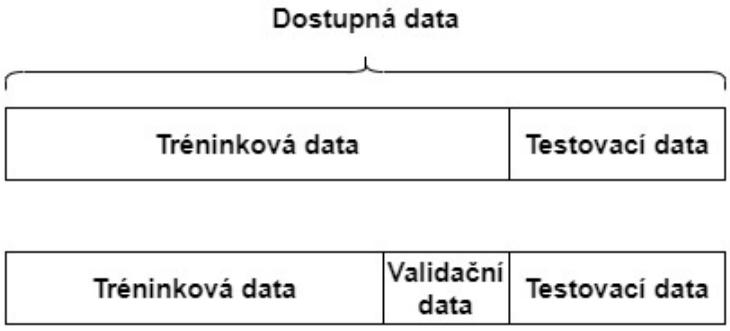
Statistika zkoumá a vyvíjí specifické metody pro určování hypotéz ve světle empirických vztahů [39]. Je založena na statistice matematické, která je větví aplikované matematiky. Historie statistiky sahá až do počátku období islámského zlatého věku v 8. století, kdy Al-Khalil napsal knihu „Knihu kryptografických myšlenek“, obsahující první použití permutací a kombinací za cílem popsat co nejvíce arabských slov se samohláskami i bez nich [40]. Z hlediska metod jsou statistika a strojové učení úzce související obory, liší se ve svém cíli. Zatímco hlavním úkolem statistiky je interpretovat a pochopit význam statistických hodnot ze vzorku dat, strojové učení hledá v tomto vzorku zobecnitelné prediktivní vzorce [41].

2.4 Klíčové pojmy

2.4.1 Tréninková, validační a testovací data

Základní „surovinou“ pro vytváření algoritmů strojového učení jsou data. Algoritmus naplníme daty a tím vytvoříme matematický model. Data, použitá k vytvoření finálního modelu, většinou pochází z několika datasetů, jinak řečeno kolekcí dat. Velmi běžné je použití tří datasetů, tedy tréninkového, validačního a testovacího, v odlišných fázích vývoje modelu [34]. Všechny datasety pocházejí ze stejného zdroje, jejich odlišným parametrem je pouze velikost. Neexistuje žádné přesné pravidlo na rozdělení dat, jsou ovšem známé obecně dodržované principy, které většina uživatelů dodržuje. Největší část by měla vždy tvořit tréninková data, většinou se udává rozmezí 70 až 80 %. Validační set je libovolný a není podmínkou k získání výsledného modelu, přinese ovšem hlubší pochopení modelu a nabízí jeho další vylepšení, proto se většinou používá. Jeho zastoupení se pohybuje v rozmezí 10 až 20 %. Testovací data zaberou zbylou část, tedy 10 až 30 % všech dat [42][43]. Grafické rozdělení dat lze pozorovat na Obr. 3. Prvotní vývoj modelu probíhá na tréninkových datech.

V této části vývoje je cílem upravit parametry modelu tak, aby byl co nejvhodnější pro tréninková data. Po úspěšném upravení modelu na tréninkových datech přichází na řadu validační dataset, který poskytuje objektivní zhodnocení upraveného modelu. S validačními daty model to této fáze nepracoval, tudíž může zhodnotit kvalitu modelu. Model si může na testovací data navyknout přespříliš a poté nedosahuje dobrých výsledků na nových datech, tomuto jevu se říká „přetrénování“, anglicky *overfitting*. Na základě výsledků se upraví parametry modelu, aby s novými daty pracoval lépe. Tyto dva kroky lze opakovat, dokud není funkce modelu optimální. Jako poslední se použijí testovací data, která nám poskytnou objektivní vyhodnocení finálního modelu[34].



Obr. 3: *Rozdělení dat na jednotlivé datasety.*

2.4.2 Model a algoritmus

Algoritmus a model jsou základní prvky každé aplikace strojového učení, avšak často bývají zaměňovány či dokonce brány jako jednotný prvek. Pravdou ovšem je, že model a algoritmus jsou rozdílné součásti.„Algoritmus“ je ve strojovém učení postup, který se spouští na datech za účelem vytvoření „modelu“ strojového učení [44]. V naší aplikaci bude algoritmus rozpoznávat vzory na jednotlivých snímcích kožních lézí a poté vytvoří a naučí model rozpoznávat vzory na nových snímcích, se kterými nikdy nepracoval. Existuje mnoho algoritmů strojového učení, například lineární a logistická regrese, rozhodovací stromy, algoritmus k-nejbližších sousedů a mnoho dalších [13][34]. Vybrané algoritmy budou popsány blíže v kapitole Metody strojového učení. Algoritmy mají řadu vlastností:

- Algoritmy strojového učení lze popsat pomocí matematiky a pseudokódu.
- Efektivitu algoritmů strojového učení lze analyzovat a popsat.
- Algoritmy strojového učení lze implementovat v kterémkoliv z řady moderních programovacích jazyků.

To znamená, že ve výzkumných pracích a učebnicích je možné vidět popis algoritmu strojového učení pomocí pseudokódu či lineární algebry. V pracích lze vidět porovnání výpočetní účinnosti konkrétního algoritmu s jiným konkrétním algoritmem [44]. Množství možných algoritmů je v zásadě neomezené a kdokoliv může vytvořit algoritmus nový a

začít ho používat na svých projektech. Pomoc při práci s algoritmy usnadňují knihovny programovacích jazyků, mezi nejrozšířenější patři knihovny *scikit-learn*, *TensorFlow* či *Keras* [14][45][46]. Model strojového učení je na druhou stranu výstupem algoritmu strojového učení spuštěného na datech [44]. Je to „věc“, která je uložena po spuštění algoritmu na tréninkových datech a definuje pravidla potřebná k vytvoření predikce. Nejlepším příkladem je přemýšlet o modelu strojového učení jako o „programu“, který se skládá jak z dat, tak z postupu pro použití dat k predikci [34][44].

2.5 Metody strojového učení

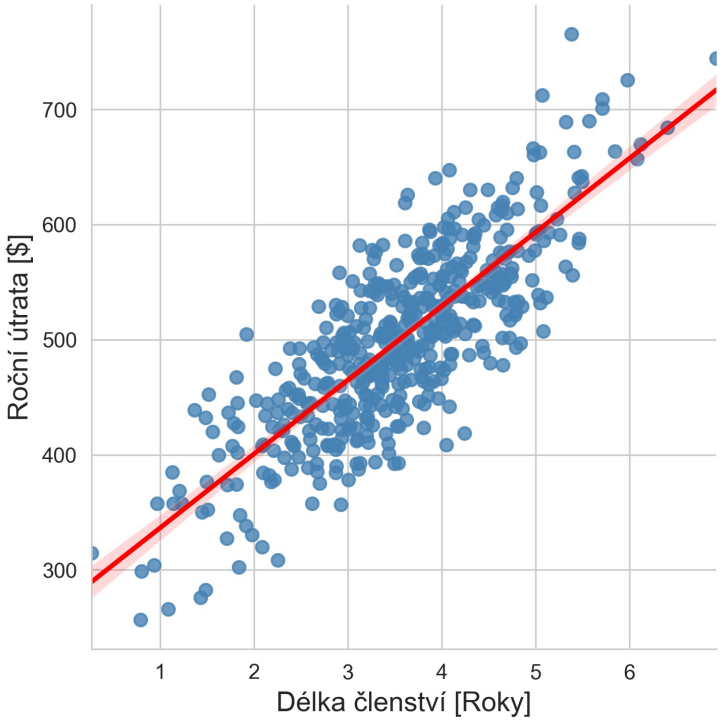
Existuje mnoho metod strojového učení, které jsou roztříděné kategorií na základě určitých kriterií. Mezi tato kritéria patří přítomnost či absence matematického modelu nebo rychlost učení. Primárně se ale systémy dělí dle množství dozoru, které je jim poskytnuto během procesu [13].

2.5.1 Supervised Learning

Supervised Learning neboli učení s učitelem je druh učení, kdy poskytneme algoritmu označený tréninkový dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, kde je každý element $\mathbf{x}_i$ napříč N takzvaný *feature vector*, neboli vektor prvků. To je vektor, ve kterém každá dimenze $j = 1, \ldots, D$ obsahuje hodnotu, která nějakým způsobem popisuje daný příklad. Tato hodnota se nazývá prvek a je označen jako $x^{(j)}$. Pokud každý příklad $\mathbf{x}$ v našem datasetu reprezentuje osobu, potom první prvek, $x^{(1)}$, by mohl obsahovat váhu v kg, druhý prvek, $x^{(2)}$, může obsahovat výšku v cm, $x^{(3)}$ může obsahovat pohlaví a tak podobně. Pro všechny příklady v datasetu bude prvek na pozici j ve vektoru prvků obsahovat stejný typ informace. Pokud by tedy prvek $x_i^{(2)}$ obsahoval váhu v kg v libovolném příkladu x_i poté $x_k^{(2)}$ bude také obsahovat váhu v kg v každém příkladu $\mathbf{x}_k, k = 1, \ldots, N$. y_i může být součástí konečné řady tříd $\{1, 2, \ldots, C\}$, nebo reálné číslo či komplexnější struktura, jako vektor, matice nebo graf [32][33]. Mezi typické úkoly učení s učitelem patří klasifikace a regrese. Jako klasifikaci si můžeme představit například filtr spamu v emailu. Poskytneme systému dostatečné množství dat v podobě emailů s označením, zda se jedná o spam či nikoli. Klasifikační algoritmus se naučí rozpoznat nevyžádanou zprávu a na základě zkušeností vytvoří model, který bude schopný tento princip aplikovat na nově příchozí emaily [14]. V takovém případě by y_i reprezentoval set s dvěma třídami $\{\,\mathrm{spam}, \mathrm{není\_spam}\,\}$ [33]. *Supervised Learning* má tedy za cíl vytvořit matematický model, který bude schopný vytvářet přesné předpovědi na datech, která nezná. Tento druh učení často vyžaduje vyšší náročnost na vytvoření tréninkového datasetu, ovšem poté zautomatizuje a zrychlí jinak náročné úkoly [14][33].

Lineární regrese / Linear regression je velmi snadný přistup učení s učitelem. Tento algoritmus se používá již řadu let a v porovnání s modernějšími statistickými přístupy

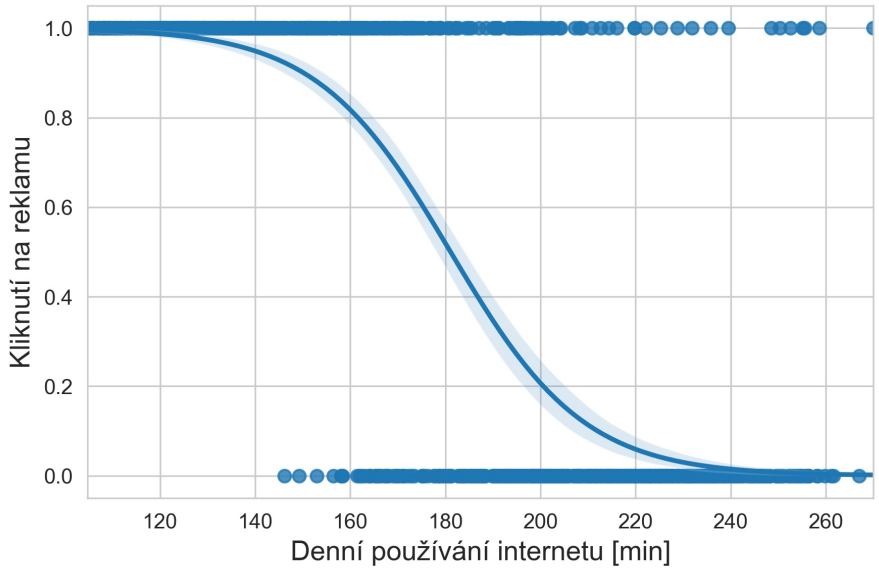
se může zdát zastaralý. Pravdou ovšem je, že se stále jedná o široce využívanou metodu statistického učení. Při použití algoritmu lineární regrese se snažíme vymodelovat závislost mezi proměnnými, často se využívá principu metody nejmenších čtverců [34]. Názorným příkladem je studie *A Multiple Linear Regression Approach For Estimating the Market Value of Football Players in Forward Position* [35], neboli predikce tržní hodnoty hráčů fotbalu pomocí několika lineárních regresí. Autoři naučili model předpovídat cenu hráče dle vložených fyzických vlastností a úspěchů za minulou sezónu. Jednoduchý příklad lze vidět na Obr. 4. Graf obsahuje data z obchodního domu, ve kterém mají zákazníci založená věrnostní členství. Výsledkem analýzy je korelace mezi roční útratou zákazníka a jeho délkou členství, z grafu lze vyčíst, že zákazník s déle trvajícím členstvím v obchodním domě také utratí ročně více peněz.



Obr. 4: *Lineární regrese*

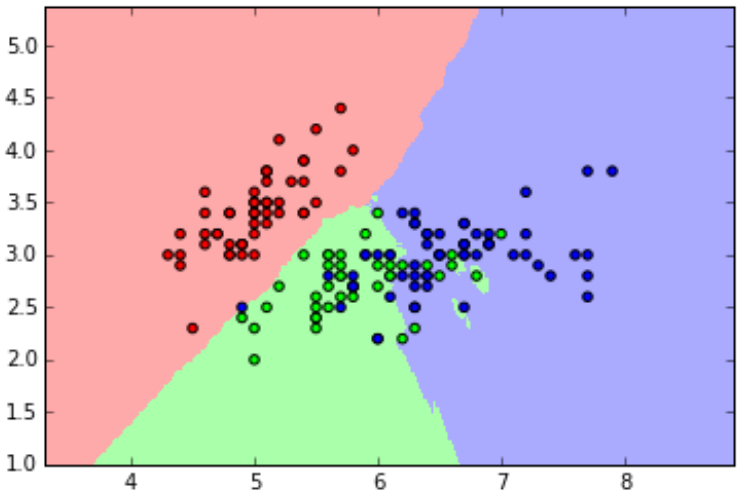
Logistická regrese / Logistic regression , anglicky *logistic regression*, je běžně používaný algoritmus pro určení pravděpodobnosti, zda příklad patří do určité třídy. Použití si můžeme představit na již zmíněném filtru spamu v emailu. Pokud odhadovaná pravděpodobnost je větší než 50 %, poté model predikuje, že email patří do spamu (patří do pozitivní třídy, označené „1“). V opačném případě predikuje, že se o spam nejedná (patří do negativní třídy, označené „2“) [13][34]. Důležité je si uvědomit, že ačkoliv logistická regrese obsahuje ve svém názvu slovo „regrese“, spadá tento algoritmus do klasifikační kategorie, jak lze poznat z popsaného příkladu. Logistická regrese se hojně využívá v oblasti neuronových sítí, kde je známá jako sigmoidální funkce. Na příkladu logistické regrese na

Obr. 5 je snaha lépe zacílit reklamu na návštěvníky webového serveru. Hlavními parametry byla zvolena závislost mezi dobou strávenou na internetu a pravděpodobností kliknutí na reklamu. Z grafu jde vyčíst, že větší pravděpodobnost kliknutí na reklamu je u zákazníku, kteří tráví na internetu méně času. Díky takové analýze lze reklamy lépe cílit na klíčové zákazníky.



Obr. 5: *Logistická regrese*

k-Nejbližších Sousedů / k-Nearest Neighbors (kNN) , označovaný zkratkou „kNN“, je anglický název pro algoritmus k-nejbližších sousedů. Velmi pravděpodobně se jedná o nejjednodušší algoritmus strojového učení. K vytvoření modelu stačí pouze nahrát tréninková data a algoritmus sám vyhledá k určené pozici nejbližší body - tedy své „nejbližší sousedy“ [14]. Algoritmus patří do skupiny klasifikačních algoritmů a řadí se mezi základní v této třídě. Díky svým vlastnostem by měl být jednou z prvních voleb pro klasifikační studii při slabé či žádné předchozí znalosti o rozložení zkoumaných dat [46]. Na Obr. 6 lze vidět finální výsledek algoritmu pro náhodná data se zvoleným koeficientem $K = 3$, pro rozdělení do 3 tříd.



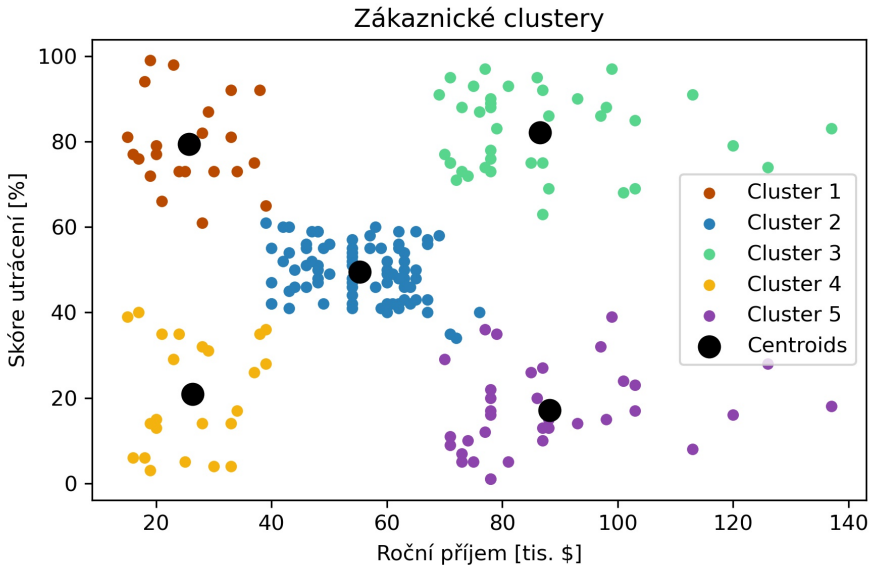
Obr. 6: *Algoritmus K-nejbližších sousedů*

Umělé Neuronové Sítě / Artificial Neural Networks (ANNs) *Artfificial Neural Networks*, je rodina algoritmů, známá pod českým názvem „Umělé neuronové sítě“, zkratkou „ANNs“, či zkráceně pouze jako „neuronové sítě“. Ve své práci používám algoritmy z této skupiny a jelikož problematika neuronových sítí je značně rozsáhlá, bude jim věnována kapitola 2.6.

2.5.2 Unsupervised Learning

Unsupervised Learning neboli učení bez učitele je druh učení, kdy poskytneme algoritmu neoznačená data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, kde x je opět *feature vector*. Cílem algoritmů je vytvořit matematický model, který použije *feature vector* a přetvoří ho na jiný vektor nebo hodnoty, a ta je následně použita k řešení problému [33]. Mezi hlavní druhy algoritmů sem patří *Clustering* neboli Shlukování a *Dimensionality Reduction*.

Shlukování / Clustering Shlukování tvoří početnou skupinu postupů pro hledání skupin nebo klastrů (z anglického slova *cluster*) v neoznačených datech. K tomu je potřeba vytvořit model, který dokáže najít mezi jednotlivými prvky podobnost. Představme si, že disponujeme obsáhlými daty o návštěvnících internetového zpravodajského serveru. Shlukovací algoritmus se pokusí najít podobné skupiny uživatelů bez lidské pomoci. Na Obr. 7 lze vidět rozdělení zákazníků do pěti klastrů, rozdělených dle ročního příjmu zákazníka a jeho mírou utrácení. Sledujeme, že zákazníci v klastru 1 jsou i přes nižší roční příjem věrnými zákazníky a obchodníkovi se vyplatí udržovat si s těmito zákazníky dobré vztahy. Úplným opakem jsou zákazníci z klastru 5, kteří i přes vysoké roční příjmy neutrácejí velké množství peněz. Těmito postupy lze lépe zacílit nabídky a reklamu každé skupině [13][34].



Obr. 7: *Algoritmus shlukování*

Snížení Dimenzí / Dimensionality Reduction Úkolem *Dimensionality Reduction* je zjednodušit data a neztratit příliš informací. Existuje několik metod, například sloučit několik prvků s určitým vztahem. Počet najetých kilometrů u automobilu je často v přímé úměře se stářím vozu. Algoritmus sloučí oba prvky do jednoho, který bude reprezentovat opotřebení vozidla. Provedeme-li tento postup u více prvků, dokážeme zredukovat množství dat a urychlíme si práci v dalších analýzách [13][14].

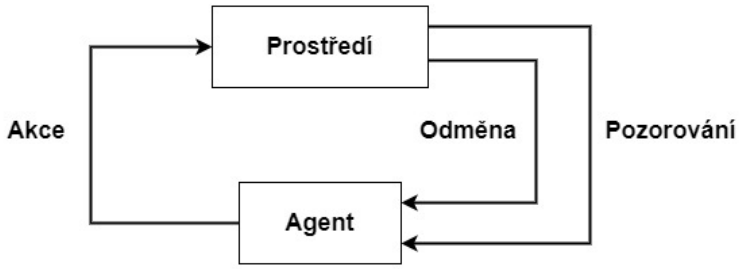
2.5.3 Semi-supervised learning

Semi-supervised learning neboli kombinace učení s učitelem a bez učitele spadá, jak již název napovídá, mezi metody učení s učitelem, *supervised learning*, a učení bez učitele, *unsupervised learning*. Vstupem pro tento druh učení jsou jak neoznačená, tak i označená data, avšak s většinovou přesilou těch neoznačených. Přidání neoznačených dat se může zdát jako kontraproduktivní, či vytváření větší nejasnosti do celého problému. Pravdou ovšem je, že s větším počtem neoznačených dat přinášíme více informací o daném problému, a větší vzorek dat vždy přinese reálnější a přesnější výsledek [13][14].

Ukázkový příklad metody *semi-supervised learning* jsou služby na archivaci a skladování fotografií, například aplikace „Fotky“ od společnosti *Google*. Jakmile nahrajete na tuto službu všechny své rodinné fotografie, automaticky rozezná, že stejná osoba A se nachází na fotografiích 1,6 a 14, zatímco osoba B se vyskytuje na snímcích 3,6 a 11. Toto je část metody učení bez učitele, přesněji se jedná o shlukování, *clustering*. Nyní stačí jediná označená fotografie a systém rozpozná danou osobu na všech ostatních snímcích [13].

2.5.4 Reinforcement learning

Reinforcment learning, v překladu zpětnovazebné učení, je způsob učení na základě zpětné vazby a dnes patří mezi nejpokročilejší kategorii. Na rozdíl od učení s/bez učitele, se učící systém, zvaný „agent“, souvisle vylepšuje za využití zpětných vazeb [12]. Systém pozoruje prostředí a samostatně provádí kroky, za které dostává odměnu (případně penalizaci ve formě negativní odměny). Pomocí formy odměn se systém naučí zvolit vždy tu nejlepší možnou strategii [13][47]. Ilustrovaný princip zpětnovazebného řízení lze pozorovat na Obr. 8. Časté využití zpětnovazebného učení sledujeme v oblasti robotiky. Jako názorný příklad si můžeme uvést autonomní robotické vysavače. Tyto přístroje v nabíjecí stanici zprovozníme a ony si samostatně zmapují plochu a na základě odměn zvolí nejlepší strategii, pro co nejefektivnější způsob úklidu [47]. Mezi další příklady se řadí využití v samořídících automobilech, především sledování jízdního pruhu či volba optimální rychlosti [60]. Další časté použití je v oblasti klasických her, jako například šachy [47], tak i ve hrách počítačových [49].

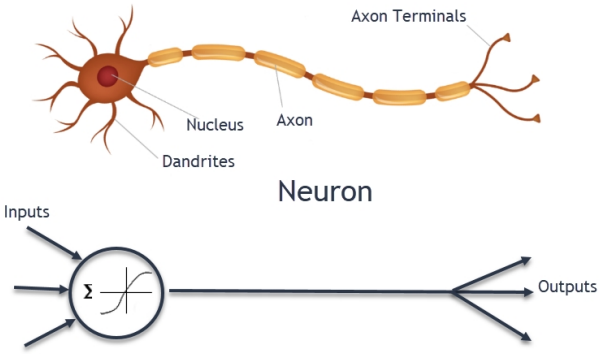


Obr. 8: *Zpětnovazebné učení*

2.6 Umělé Neuronové Sítě / Artificial Neural Networks (ANNs)

2.6.1 Úvod

Umělá neuronová síť, anglicky známá jako *Artifical neural network* a často označovaná pouze jako neuronová síť, je významná technika a hojně používaný výpočetní model strojového učení. Pojmenování „Umělá neuronová síť“ je inspirováno podobností algoritmu k lidskému mozku. Jelikož je mozek nejpropracovanější nástroj na zpracování informací který známe, je naší snahou napodobit jeho architekturu a implementovat ji do strojů. Mozek je tvořen především z nervových buněk, neboli neuronů. Neuron je specializovaná buňka schopná přijmout, vést, zpracovat a odpovědět na signál, kterou mimo jiné popsal roku 1835 český fyziolog Jan Evangelista Purkyně.

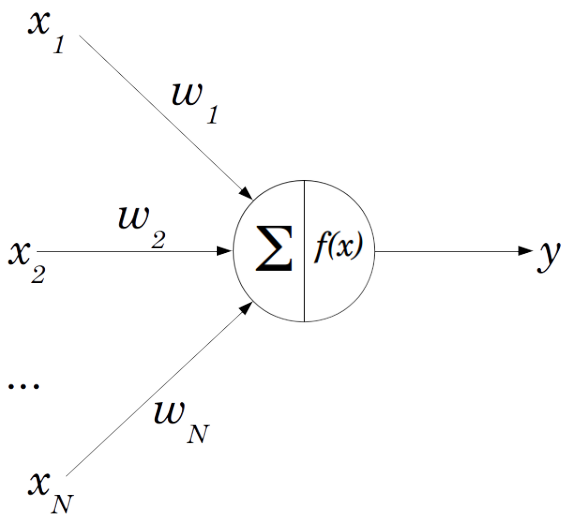


Obr. 9: *Schéma neuronu [13]*

Jednotlivé neurony jsou propojeny dendrity, kterými jsou přijímány vzruchy do neuronu. Neuron přijatý vzruch zpracuje a vede ho dále přes axon do dalšího neuronu. Dendritů muže mít neuron větší počet, avšak axon obsahuje pouze jeden [50]. Podobnou strukturu na bázi neuronů využívají i neuronové sítě. Avšak lidskému mozku se zdaleka nelze neuronovou sítí výkonově přiblížit. Počet neuronů v lidském mozku je přibližně 10^{11} a každý jeden neuron je propojen přibližně s 10^4 dalšími. Zajímavé porovnání vlastností lidského mozku a stroje je doba přenosu informace z neuronu na neuron. Zatímco lidský neuron zvládne tuto činnost nejrychleji za 10^{-3} sekundy, počítači stačí 10^{-10}. Interpretace, že umělá neuronová síť je pouze inspirací biologického procesu, je správná, neboť některé vlastnosti umělé neuronové sítě nejsou konzistentní s biologickými procesy. Zatímco v umělé síti lze mít výstup ve formě jedné hodnoty, biologický výstup neuronů je komplexní časová řada hodnot [51].

2.6.2 Perceptron

Nejjednodušším modelem neuronové sítě je takzvaný Perceptron vytvořený roku 1957 Frankem Rosenblattem [52]. Tento model neuronové sítě se skládá pouze z jednoho neuronu.



Obr. 10: *Perceptron model*

Perceptron slouží jako binární klasifikátor, který přijme vstupní hodnoty ve formě vektoru x = $(x_1; x_2; x_3; ....; x_n)$. Představme si použití perceptronu v bankovním prostředí, jehož úkolem bude určit, zda je bezpečné půjčit jednotlivým osobám peníze. Mezi vstupní hodnoty $x_1 - x_n$ zahrneme důležité informace osoby o schopnosti splácet půjčené peníze jako například věk, vzdělání, aktuální zaměstnání, celkový majetek či pravidelné příjmy. Neuronové sítě se dělí na vrstvy. Model perceptronu má pouze dvě vrstvy, vstupní vrstvu, která přijímá vstupní vektor hodnot a výstupní vrstvu. U dalších modelů neuronových sítí jsou přítomné mezivrstvy nazývané skryté vrstvy [33]. Jednotlivé vstupy jsou propojeny s neuronem takzvanými synapsemi, na které je každé vstupní hodnotě přiřazena numerická váha w_i, která určuje sílu a důležitost dané hodnoty. Poté, co jsou vstupním hodnotám w_i přiřazeny váhy w_i, vypočte se vážený součet:

$$\sum_i w_i x_i \tag{1}$$

Na ten se použije přenosová funkce f, která nám poskytne výstup:

$$vstup = \begin{cases} 0 \; if \; \sum_i w_i x_i \; \leq \; \text{práh } b \\ 1 \; if \; \sum_i w_i x_i \; > \; \text{práh } b \end{cases} \tag{2}$$

Práh symbolizuje reálnou hodnotu b, která určí, zda výstup perceptronu bude nula či jedna. Určením nižší hodnoty prahu bude model predikovat větší množství zákazníku jako schopné splácet svou půjčku. Naopak zvýšení prahu by připustilo půjčku menší skupince s lepší schopností splácet půjčku. Pro aktivaci neuronu, tedy odsouhlasení půjčky klientovi, je nutné mít na výstupu číslo 1. Existuje mnoho přenosových funkcí, ale pro model perceptronu

se nejčastěji používá skoková přenosová funkce. Samotný neuron není příliš efektivní nástroj, jelikož ho lze použít pouze na množiny lineárně separovatelné, avšak společně ve formě neuronových sítí dokáží přinést přesné odpovědi na komplexní problémy strojového vidění či umělé inteligence [51].

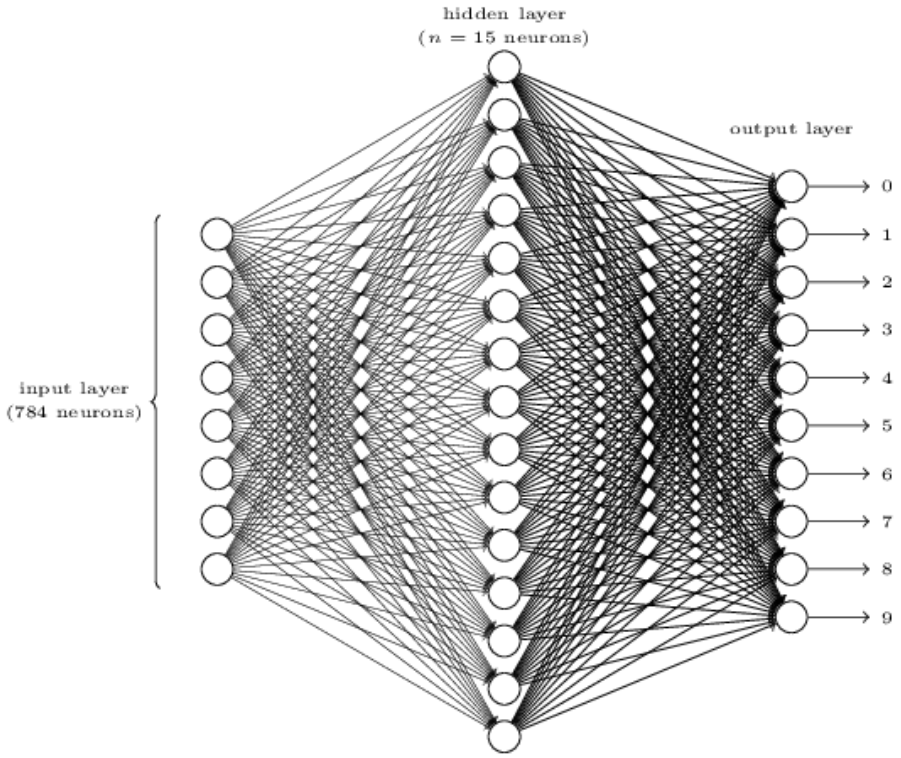
2.6.3 Hluboké versus Mělké učení

Již bylo popsáno, že neuronová síť se dělí na vrstvu vstupní, libovolný počet skrytých vrstev a výstupní vrstvu. Neexistuje přímá hranice počtu skrytých vrstev, která rozděluje hluboké a mělké učení. Všeobecný úzus říká, že pokud neuronová síť obsahuje alespoň dvě skryté vrstvy, jedná se o hluboké učení, z anglického *deep learning*. V případě jedné či žádné skryté vrstvy se jedná o mělké učení, z anglického *shallow learning* [33]. Jelikož se nejedná o psané pravidlo, někteří považují za hluboké učení použití například počtu až pěti či více skrytých vrstev. Z pojmenování *skryté vrstvy* se může zdát, že se jedná o něco záhadného, název ovšem poukazuje jen na to, že vrstva není ani vstupní ani výstupní [53]. Počet neuronů ve vstupní vrstvě je dán počtem vstupních proměnných. Budeme-li chtít predikovat cenu domu, mezi vstupní informace bude patřit rozloha domu, počet ložnic, koupelen, stáří domu či například lokalita. Počet těchto vstupních informací bude odpovídat počtu neuronů ve vstupní vrstvě neuronové sítě. Cílem bude odhadnou cenu domu, tedy jednu proměnnou. Výstupní vrstva bude složena pouze z jednoho neuronu [32].

Představme si, že jsme definovali neuronovou síť, kterou nyní použije na rozpoznávání ručně psaných číslic. Lidé podobný úkol zvládnou hravě, avšak pro počítač se jedná o složitý úkol. Úkolem je rozpoznat o jakou číslici se jedná.

Obr. 11: *Ručně psaná číslice 5 [54].*

Černobílý snímek ručně psané číslice na Obr. 11 nabízí rozlišení $28x28$ pixelů. Vstupní vrstva neuronové sítě bude tedy složená z $728 = 28\ x\ 28$ neuronů. Každý jeden vstupní neuron reprezentuje zbarvení daného pixelu. Jelikož se jedná o černobílý snímek, hodnota 0 reprezentuje bílou barvu, hodnota 1 reprezentuje černou barvu a hodnoty mezi reprezentují tmavnoucí odstíny šedi. K příkladu zvolíme neuronovou síť s jednou skrytou vrstvou, architekturu lze vidět na Obr. 12. Počet neuronů skryté vrstvy bude $n = 15$ [53].



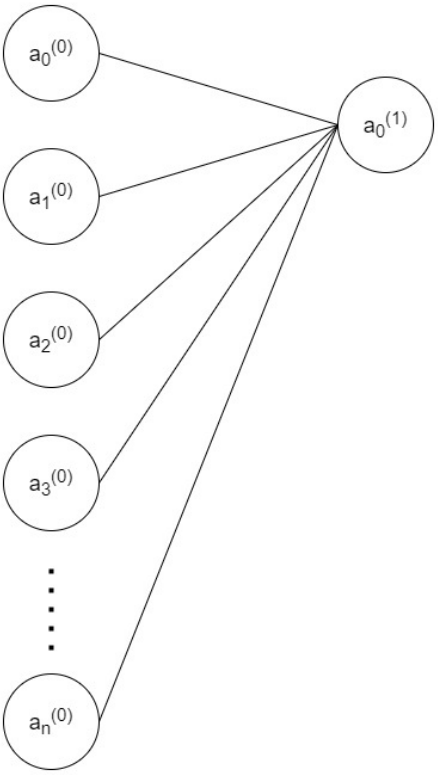
Obr. 12: *Architektura použité neuronové sítě [55].*

Určení počtu skrytých vrstev a počtu neuronů v jednotlivých vrstvách je zásadní otázka při tvorbě neuronových sítí, k jejíž odpovědi je zapotřebí velké množství testování či zkušeností. Po skryté vrstvě následuje výstupní vrstva čítající deset neuronů, každý pro jednu číslici $0-9$. Nelze jednoduše popsat jakým způsobem probíhá klasifikace číslic uvnitř neuronové sítě, skrytým vrstvám se totiž někdy také přezdívá „černá skříňka“ [33][53]. Představa je, že skrytá vrstva rozdělí snímek číslice na počet segmentů n. Pro lepší představu uvažujme počet skrytých neuronů $n = 4$, které rozdělí snímek na 4 segmenty jako na Obr. 13.

Obr. 13: *4 Segmenty číslice 5 [54].*

Tyto čtyři obrázky po složení představují číslovku pět. Zaměříme se na šestý výstupní neuron sítě, jehož úkolem je určit, zda se jedná o číslovku pět. Neuron přijme čtyři výstupy ze čtyř neuronů předchozí skryté vrstvy. Všechny čtyři segmenty se shodují s tvarem číslovky pět, tudíž všechny skryté neurony propustí signál dále do výstupního neuronu, který určí, že se jedná o číslovku pět. Stejnou ideologií se zaměříme na desátý výstupní neuron sítě, jehož

úkolem je určit, zda se jedná o číslovku devět. Nyní by možná jeden až dva skryté neurony určily, že se jedná o číslovku devět, ovšem zbylé segmenty snímku nezobrazují stejné tvary jako číslovka devět a nepošlou do výstupního neuronu signál. Výstupem neuronové sítě tedy bude, že ručně psaná číslice není číslovka devět [53]. Při zvolení většího počtu skrytých neuronů jako v naší ukázkové architektuře $n = 15$ docílíme rozdělení snímku na patnáct segmentů, díky kterým lze provést přesnější určení ručně psané číslice. Použití většího počtu skrytých vrstev umožňuje detailnější rozdělení snímku a zvyšuje úspěšnost predikce. Ledabylé přidávání skrytých vrstev a počet neuronů ovšem není správným směrem, neboť s větším počtem skrytých sítí a neuronů se zvyšuje komplexnost neuronové sítě a s ní i její nároky na výkon. Doporučuje se vždy využít nejmenší možný počet skrytých neuronů, který je dostačující na úspěšné provedení úkolu.



Obr. 14: *Propojení prvního neuronu ve skryté vrstvě s vrstvou vstupní.*

Hodnotu prvního neuronu lze matematicky zapsat ve skryté vrstvě rovnicí:

$$a_0^{(1)} = \sigma(w_{0,0} \cdot a_0^{(0)} + w_{0,1} \cdot a_1^{(0)} \ldots + w_{0,n} \cdot a_n^{(0)} + b_0) \tag{3}$$

, kde horní index (1) značí, že se neuron a nachází ve 2. vrstvě neuronové sítě a spodní index 0, označuje první neuron v dané vrstvě. Do sigmoidální přenosové funkce jsou načteny všechny neurony ze vstupní neurony ze vstupní vrstvy $a_0^{(0)} - a_n^{(0)}$, vynásobené příslušnou váhou $w_{0,0} - w_{0,n}$. Po sečtení vážených hodnot všech vstupních neuronů se k váženému

součtu přičte práh b. Maticovým zápisem lze vyjádřit přenosovou funkci aplikovanou na celou vrstvu následovně:

$$\sigma\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \dots & w_{0,n} \\ w_{1,0} & w_{1,1} & \dots & w_{1,n} \\ w_{2,0} & w_{2,1} & \dots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,0} & w_{m,1} & \dots & w_{m,n} \end{bmatrix} \cdot \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}\right) \tag{4}$$

, kde jsou váhy vstupních neuronů $w_{0,0} - w_{m,n}$, hodnoty vstupních neuronů $a_0^{(0)} - a_n^{(0)}$ a prahy $b_0 - b_n$. Zjednodušením zápisu lze vyjádřit přechod z jedné vrstvy neuronové sítě na druhou pomocí rovnice:

$$a^{(1)} = \sigma(W \cdot a^{(0)} + b) \tag{5}$$

Ve své podstatě je umělý neuron funkce, která přijme výstupy všech neuronů z předchozí vrstvy a vyšle dále číslo mezi $0 - 1$. Pokud tuto ideu rozšíříme, uvědomíme si, že celá umělá neuronová síť je jedna velká a komplikovaná funkce [53].

Vytvoření úspěšné architektury skrytých vrstev je víceméně umění a nelze představit jednotný postup. Výzkumníci neuronových sítí tedy vytvořili spoustu designů skrytých vrstev, určené pro specifické úkoly. Předem vytvořené uspořádání vrstev pomůže při hledání chování, které od neuronové sítě požadujeme [53]. Spojení neuronů, které vytvoří síť lze realizovat dvěma hlavními odlišnými způsoby. Do této chvíle jsme realizovali spojení mezi neurony pouze jedním směrem. Vstupní hodnoty projdou vstupním neuronem do dalších vrstev a tímto způsobem až do výstupní vrstvy. Informace postupují vždy směrem vpřed a nevytváří se žádné smyčky. Mluvíme o dopředné neuronové síti. Dopředná neuronová síť reprezentuje funkce svého aktuálního vstupu [32][33][53]. Rekurentní neuronové sítě naopak posílají svůj výstup zpět do vlastního vstupu sítě a tím vytvářejí dynamický systém. Slovo rekurentní znamená, že se určitý proces odehrává opakovaně po určitý čas. Představme si model, který chceme natrénovat na přeložení španělského slovíčka do angličtiny. Při přeložení slovíčka *banco*, jež znamená *pláž* nebo *lavička*, ovšem model neví v jakém kontextu se slovíčko používá. K vyřešení tohoto problému použijeme rekurentní neuronovou síť. Síť začne překládat větu slovo po slovu, úspěšně přeložená slova nahraje zpět do vstupu sítě a díky kontextu nahrané v paměti nebude mít problém s vybráním správného slova [57].

2.6.4 Přenosové funkce

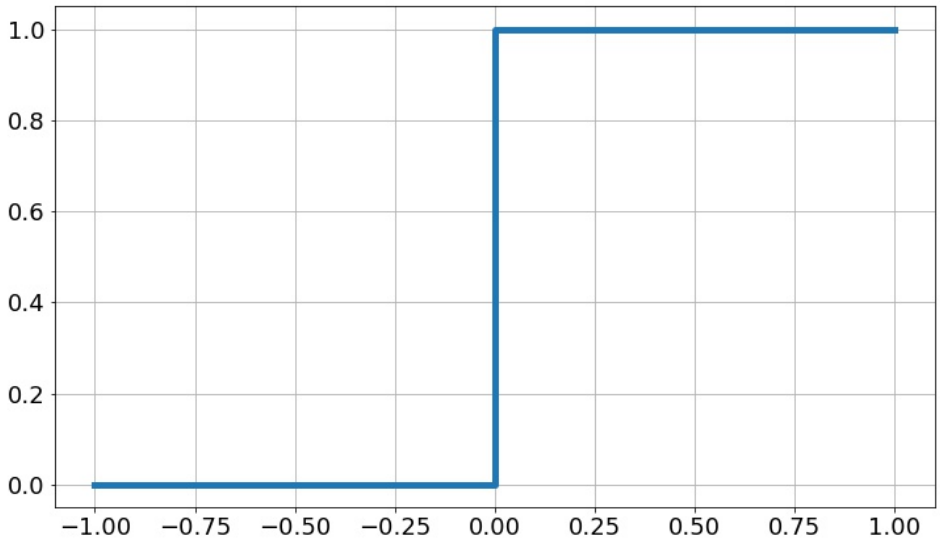
Přenosová, nebo také aktivační funkce, je matematická rovnice sloužící k určení výstupu neuronové sítě. Aktivační funkce je přidružena ke každému neuronu v síti, a rozhoduje,

zda má být výstup daného neuronu aktivován či nikoliv. Zda proběhne aktivace neuronu rozhodne pravidlo či práh. Aktivační funkce mapuje výsledné hodnoty mezi hodnoty 0 a 1, případně -1 a 1, záleží na volbě přenosové funkce. Existuje mnoho přenosových funkcí, představíme si několik nejznámějších funkcí.

Skoková přenosová funkce Skoková funkce patří mezi základní lineární přenosové funkce. Jedná se o jednoduchý binární klasifikátor. Výstup funkce závisí na velikosti vstupu. Pokud bude hodnota vstupu větší jak hodnota prahu $b = 0$, výstupem funkce bude 1. Naopak pokud bude vstup roven nebo menší než hodnota prahu, vrátí přenosová funkce 0.

$$vstup = \begin{cases} 0 & if\ y\ \leq\ 0 \\ 1 & if\ y\ >\ 0 \end{cases} \tag{6}$$

Nevýhodou přenosové funkce je její nelineárnost a možnost pouze dvou hodnot na výstupu. Hlavním cílem hlubokého učení je nastavit hodnoty vah a prahů takovým způsobem, aby umělá neuronová síť produkovala odhady blížící se co nejblíže reálné hodnotě. Použitím funkce, jejíž výstup je zvolen ze dvou možností, nám na snaze vytvořit přesné predikce příliš nepomůže. Později v procesu trénování neuronové sítě použijeme na správné nastavení vah a prahů u jednotlivých neuronů *gradient descent a backpropagation*, které vyžadují derivační aktivační funkci ke svému správnému použití. Skokovou funkci nelze derivovat a nastavení vah a prahů by tedy selhalo. Při trénování neuronových sítí se téměř vždy využívá jiných přenosových funkcí [13][58].

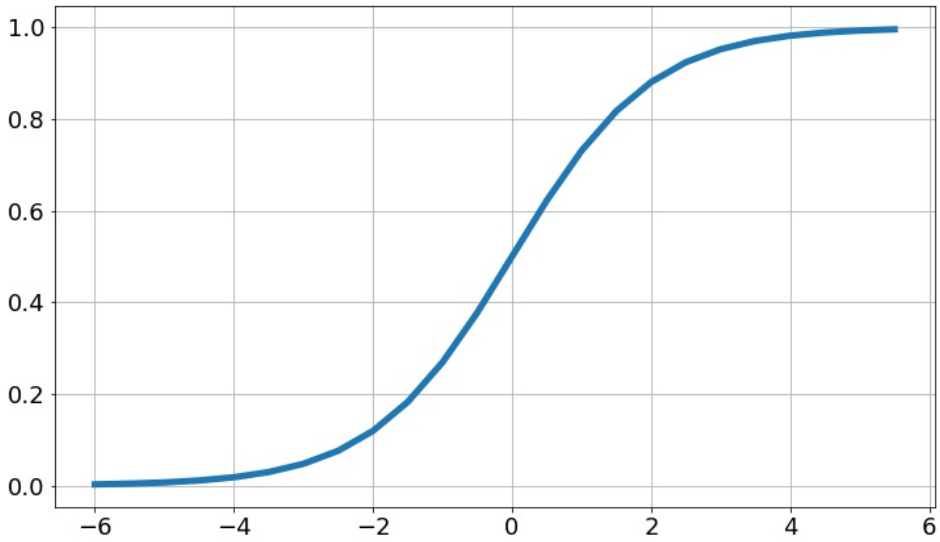


Obr. 15: *Skoková přenosová funkce*

Sigmoidální přenosová funkce Sigmoidnální přenosová funkce je jedna ze zástupců nelineárních funkcí. Nelineárnost přenosovým funkcím umožňuje provádět *backpropagation* a to díky možnosti derivace funkce. Dále umožňují právě nelineární přenosové funkce tvorbu oněch hlubokých neuronových sítí s množstvím skrytých vrstev, které jsou potřebné k natrénování komplexních datových setů s vysokou přesností [59]. Sigmoidální funkce je také známá pod názvem standardní logistická regrese.

$$\sigma(t) = \frac{1}{1 + \exp(-t)} \tag{7}$$

Sigmoidální přenosová funkce je definována vzorcem 7 a na obrázku 16. Funkce přijme jakoukoliv reálnou hodnotu a jejím výstupem je hodnota v mezích 0 a 1 [13]. Výhodou sigmoidální funkce je její tvar písmene S, který zajišťuje její hladký přechod a tím i prevenci před skoky v hodnotách výstupu. Všimněte si, že $\sigma < 0,5$ je pro $t < 0$ a $\sigma \geq 0,5$ je pro $t > 0$ [13][60]. Sigmoidální přenosová funkce je již lehce zastaralá a používá se méně než v minulosti, většinou bývá nahrazena funkcí ReLU [61].

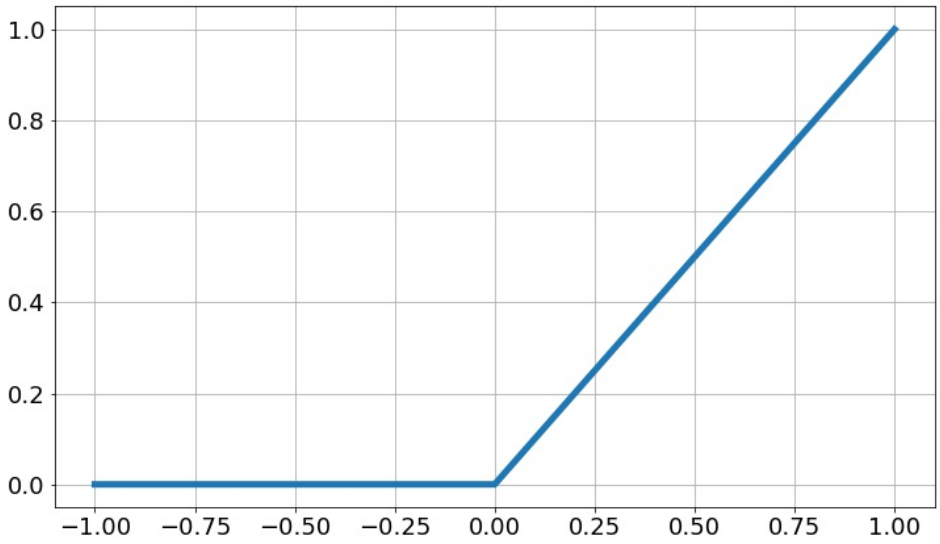


Obr. 16: *Sigmoidální přenosová funkce*

Přenosová funkce usměrněné lineární jednotky Přenosová funkce usměrněné lineární jednotky s anglickým názvem Rectified Linear Unit, ve zkratce ReLU, se řadí mezi hojně používané funkce v modelech hlubokého učení. Ve zkratce funkce vrátí 0, pokud do ní nahrajeme negativní vstup, a pokud funkce přijme pozitivní vstup, vrátí stejnou pozitivní hodnotu [33]. ReLU je zapsána ve tvaru:

$$vstup = \begin{cases} 0 & if\ y\ <\ 0 \\ y & \text{pro ostatní případy} \end{cases} \tag{8}$$

Pro správné pochopení funkcionality ReLU si představme její použití na jednom neuronu v neuronové síti. Neuron má pouze dva vstupy A a B s příslušnými váhami 2 a 3. Výstup neuronu v tomto případě bude $f(2A\ +\ 3B)$. Při použití funkce platí, že pokud je $2A\ +\ 3B$ je pozitivní, výstup neuronu je také $2A\ +\ 3B$. Pokud je $2A\ +\ 3B$ negativní, výstup neuronu se bude rovnat nule. Při dosazení za hodnoty $A\ =\ 1$ a $B\ =\ 1$ máme stále výstup $2A\ +\ 3B$ a při zvětšení A se souměrně zvětší i výstup. Na druhou stranu, pokud se bude $B\ =\ -100$ poté bude výstup roven 0 a ani při mírném zvětšení hodnoty A na se hodnota výstupu nezmění. Této funkce lze využít v oblasti zpracování obrazu. Součástí modelu natrénovaného pro rozpoznávání ryb v moři, bude neuron, jehož úkolem bude zachytit rybí oko. Tento neuron nebude aktivován, pokud by model zpracovával obrázky automobilů. Tato funkce má za následek řídkost neuronové sítě, kdy jsou výstupy u přibližně 50% skrytých neuronů rovny nule. Tímto způsobem se stane funkce více biologicky přijatelná a vede k matematickým výhodám [62].

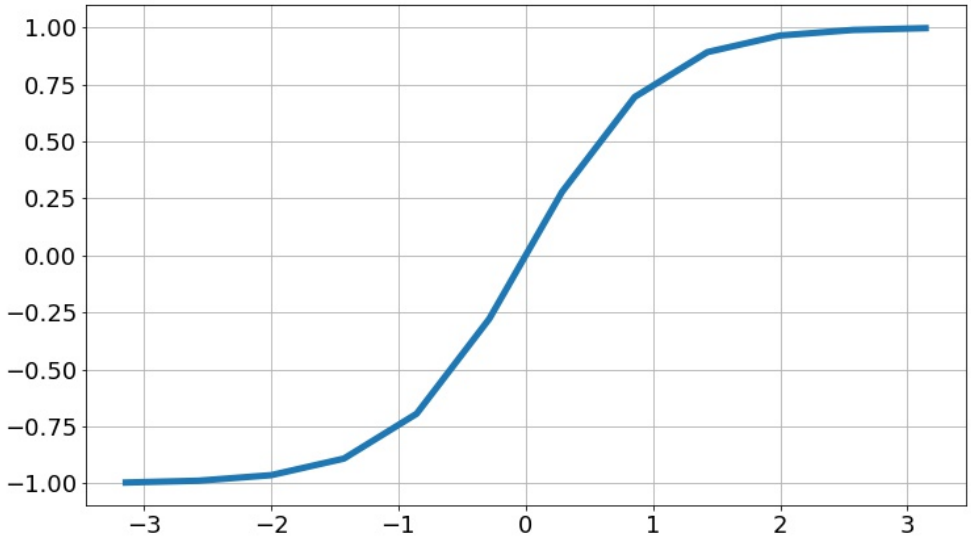


Obr. 17: *ReLU - Přenosová funkce usměrněné lineární jednotky*

Přenosová funkce hyperbolické tangenty Přenosová funkce hyperbolické tangenty, známá pod zkratkou TanH z anglického *Hyperbolic Tan gent*, se řadí mezi funkce sigmoidálního typu. Vyznačuje se tedy tvarem písmene „S“, ale výstupních hodnot může dosáhnout v rozmezí od -1 do 1. Tato vlastnost napomáhá normalizovat výstupy prvních neuronů kolem hodnoty 0 a vývoj sítě tím probíhá rychleji [13].

$$\tanh(z) = \frac{\sinh(z)}{\cosh(z)} = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{9}$$

Funkce je definována vzorcem 9, tedy jako poměr mezi hyperbolickým sinem a kosinem. Hlavní využití nabízí funkce hyperbolické tangenty při klasifikaci mezi dvěma třídami. Nabízí podobné výhody a nevýhody jako sigmoidální funkce.



Obr. 18: *TanH - Přenosová funkce hyperbolické tangenty*

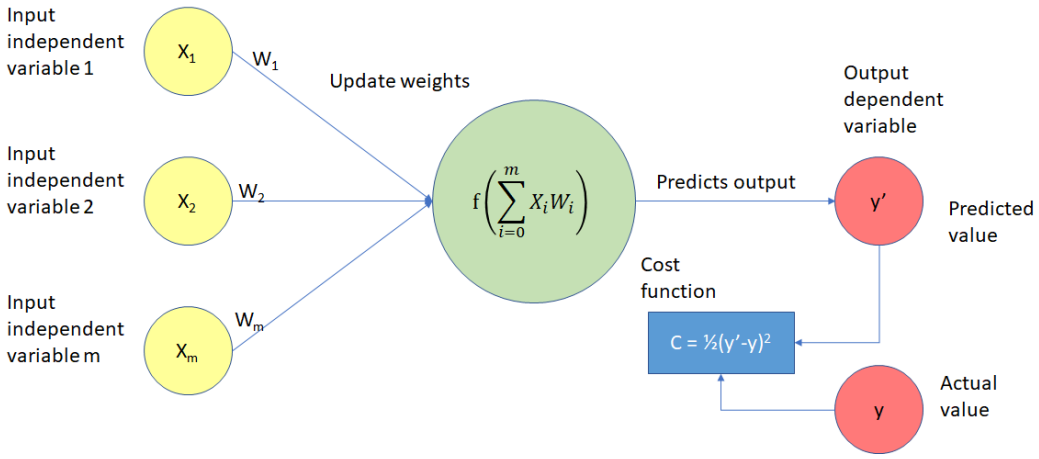
2.6.5 Učení neuronové sítě / Účelová funkce a stochastický gradient

V předchozích kapitolách jsme si již představili jak neuronové sítě pracují. Vstupní hodnota je vynásobena váhou dle své důležitosti, v neuronu je zpracována zvolenou aktivační funkcí a výsledek putuje do výstupní části neuronu. Hlavní podstatou hlubokého učení je ovšem schopnost učit se a neustále zlepšovat úspěšnost predikcí modelu. Toho docílíme upravováním hodnot vah a prahů pro jednotlivé vstupy. S tímto úkolem nám pomáhá účelová funkce C, anglicky *cost function*. Účelová funkce měří velikost chyby mezi predikovanou hodnotou $f_{w,b}(x)$ a reálnou hodnotou y [33].

$$C = \frac{1}{N} \sum_{i=1...N} (f_{w,b}(x_i) - y_i)^2 \tag{10}$$

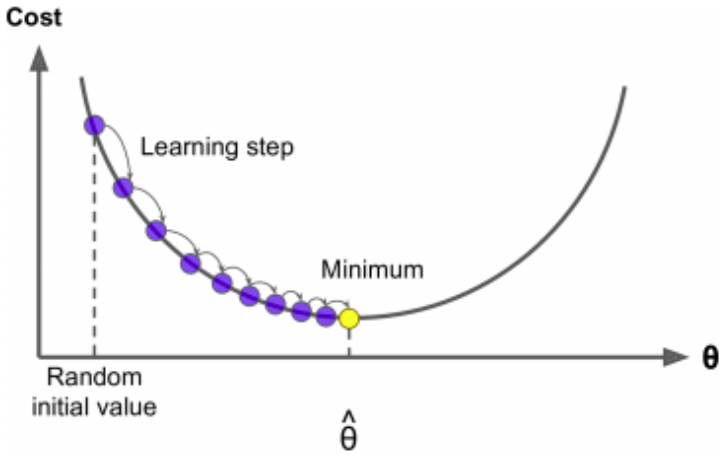
, kde N je počet příkladů. Predikované hodnoty jsou porovnány s reálnou hodnotou a je stanovena chyba pomocí ztrátové funkce $(f_{w,b}(x_i) - y_i)^2$, anglicky *loss function*. Ztrátová funkce vypočítá chybu pro jeden tréninkový příklad, účelová funkce je pouze průměr ztrátových funkcí pro celý dataset. Mezi nejčastěji používané ztrátové funkce patří

například již použitá střední kvadratická chyba, střední absolutní chyba či střední kvadratická logaritmická chyba. Výběr funkce závisí na použitém algoritmu, odlišnou funkci použijeme na regresní model a jinou na klasifikační model [63].



Obr. 19: *Schéma použití Cost function [64].*

Ztrátová funkce řekne jak dobře si vytvořený model vede. Pokud jsou predikce modelu zcela chybné, výstupem ztrátové funkce bude vysoké číslo, model dostane vysokou penalizaci. Naopak pokud jsou predikce velmi dobré, výstupem funkce bude menší číslo, model dostane nízkou penalizaci. Dle hodnoty ztráty určíme, zda se model vylepšuje či nikoliv. Naší snahou je získání co nejmenšího výstupu ztrátové funkce.

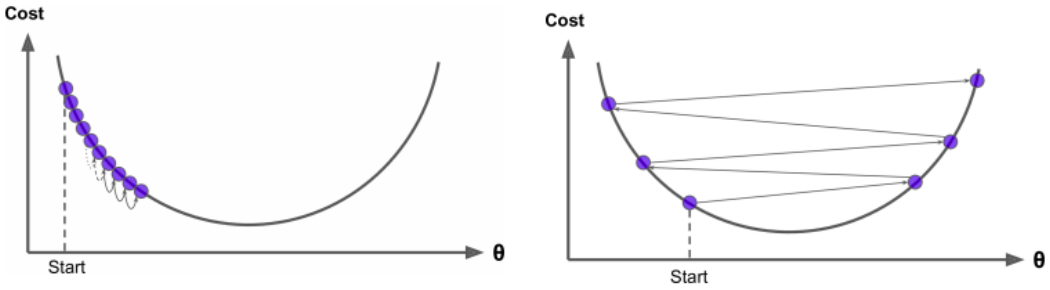


Obr. 20: *Klesání podle gradientu / Gradient descent [13]*

K tomu využijeme optimalizačním metody „Klesání podle gradientu“, která je známá spíše pod anglickým názvem *Gradient descent*. Jedná se o optimalizační algoritmus schopný najít optimální řešení pro široké množství problémů. Hlavním myšlenkou algoritmu je iterativně vyladit parametry w a b a minimalizovat účelovou funkci. Představme si, že jste ztracený v horách v husté mlze a jediné co vidíte je zem pod nohama. Dobrá strategie pro

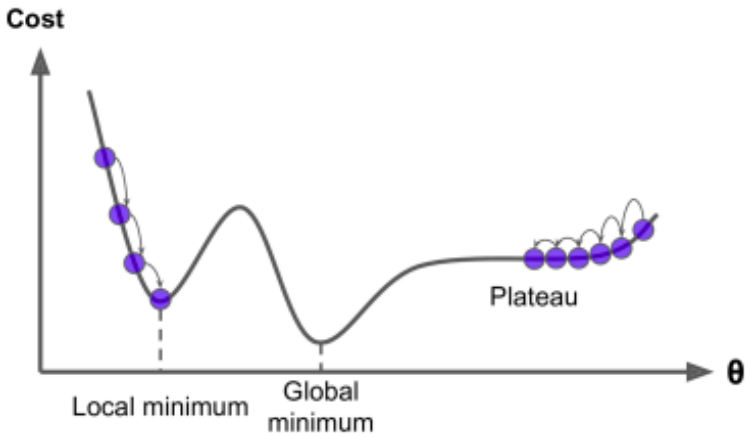
rychlý sestup je sejít dolů cestou největšího spádu a to přesně dělá algoritmus *Gradient descent* [65]. Grafickou ukázku použití klesání dle gradientu lze pozorovat na Obr. 20

Algoritmus si nejprve zvolí náhodně startovní bod, proloží ho tečnou a podle sklonu tečny zjistí, jakým směrem se má vydat. Naším cílem je najít minimum funkce, tudíž postupujeme směrem dolů. Dalším parametrem je velikost skoku, také míra učení, anglicky *Learning rate*, kterou je nutno správně zvolit. Pokud zvolíme délku kroku příliš malou, poté bude muset algoritmus provést příliš mnoho iterací a proces zabere mnoho času. Naopak pokud zvolíme délku kroku příliš velkou, můžeme přeskočit na druhou stranu funkce a případně se dostat do ještě vyšších hodnot, než původních.



Obr. 21: *Ukázka nesprávně volené délky kroku [13].*

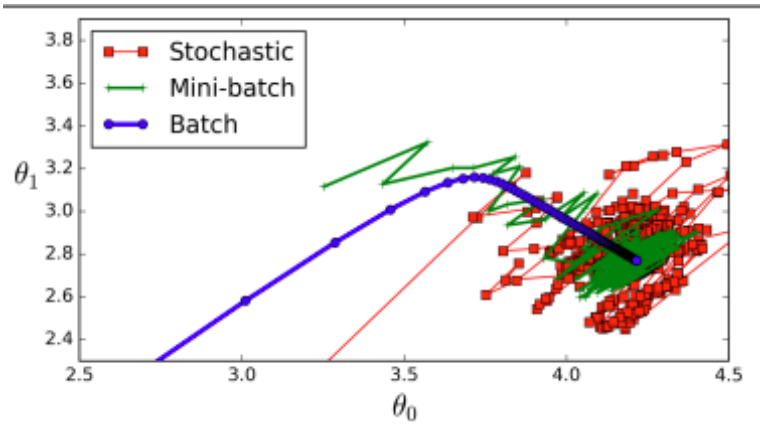
Bohužel všechny účelové funkce nemají podobný hezký tvar „misky“, ale většinou obsahují různé díry, plošiny a hřbety. Ná obrázku 22 lze takovou funkci vidět a můžeme si na ní vysvětlit dvě hlavní výzvy pro *Gradient descent*. Pokud náhodná inicializace začne algoritmus nalevo, bude konvergovat k lokálnímu minimu, které není tak dobré jako minimum globální. Pokud náhodná inicializace začne napravo, překonání plošiny zabere spoustu času a při předčasném ukončení nedosáhneme globálního minima nikdy [13].



Obr. 22: *Globální a lokální minimum [13].*

Metoda klesání podle gradientu se provádí více způsoby. Jedním ze způsobů je dávkový, *Batch gradient descent*, další je způsob stochastický, *Stochastic gradient descent*.

Hlavním problémem u dávkového gradientu je fakt, že se na vypočtení každého gradientu kroku použije celý tréninkový dataset, což má za následek vysokou časovou náročnost. Naopak stochastický gradient vybere pro každý krok náhodně jednu instanci a vypočte gradienty založené pouze na té jedné instanci. Velkou výhodou stochastického gradientu je rychlost výpočtu při použití velkých datových setů a vyšší možnost nalezení globálního minima nad lokálním minimem. Nevýhodou je ovšem povaha náhodného výběru a vysoká kmitavost, díky němuž nedokáže najít optimální hodnoty parametrů. Díky této vlastnosti je mnohem více používaný způsob dávkový, případně způsob mini-dávkový, *Mini-batch gradient descent*, který je kombinací dvou zmíněných způsobů [32]. Na obrázku 23 lze vidět vysokou přesnost dávkového gradientu a kolísavost gradientu stochastického.



Obr. 23: *Porovnání přesnosti dávkového / stochastického / mini-dávkového způsobu [13].*

Pro uplatnění metody „Klesání podle gradientu“ potřebujeme vypočítat gradient účelové funkce vzhledem ke každému parametru modelu. Jinými slovy potřebujeme zjistit jak moc se změní velikost účelové funkce, pokud lehce změníme parametry w a b. K tomuto úkolu použijeme parciální derivace účelové funkce.

$$C = \frac{1}{N} \sum_{i=1...N} (\text{Error})^2 \tag{11}$$

Pro zjednodušení použijeme proměnnou $Error$, která symbolizuje velikost chyby, penalizace. Dalším zjednodušením se zbavíme znaku sumy, který je důležitý zvláště při volbě způsobu stochastického versus dávkového. Nyní uvažujeme, že koukáme na každou chybu zvlášť. Použitím matematických pravidel získáme parciální derivace pro váhu w a práh b:

$$\frac{\partial C}{\partial w} = 2 \cdot \text{Error} \cdot \frac{\partial}{\partial w} \text{Error} \tag{12}$$

$$\frac{\partial C}{\partial b} \;=\; 2 \cdot \text{Error} \;\cdot\; \frac{\partial}{\partial b}\,\text{Error} \tag{13}$$

Nyní vypočteme gradient chyby v závislosti na oba parametry váhu w a práh b:

$$\frac{\partial}{\partial w}\,\text{Error} \;=\; \frac{\partial}{\partial w}\,(wx + b - y) \;\Rightarrow\; \frac{\partial}{\partial w}\,\text{Error} \;=\; x \tag{14}$$

$$\frac{\partial}{\partial b}\,\text{Error} \;=\; \frac{\partial}{\partial b}\,(wx + b - y) \;\Rightarrow\; \frac{\partial}{\partial b}\,\text{Error} \;=\; 1 \tag{15}$$

Derivací funkce $(wx + b - y)$ dle w dostaneme výsledek x a derivací funkce dle b dostaneme výsledek 1. Vypočtené gradienty vložíme zpět do účelové funkce a vynásobíme mírou učení (délkou kroku):

$$\frac{\partial C}{\partial w} \;=\; 2 \cdot \text{Error} \cdot x \cdot \text{Délka Kroku} \tag{16}$$

$$\frac{\partial C}{\partial b} \;=\; 2 \cdot \text{Error} \cdot \text{Délka Kroku} \tag{17}$$

Jelikož víme, že platí závislost $w^{(1)} \;=\; w^{(0)} - \Delta w$ pro váhu a závislost $b^{(1)} \;=\; b^{(0)} - \Delta b$ pro práh, kde w^0 je původní hodnota parametru, w^1 je nová, upravená hodnota parametru a Δw je vypočtený gradient chyby. Dostaneme dvě jednoduché rovnice, které jsou výsledkem metody *Gradient descent* [65]:

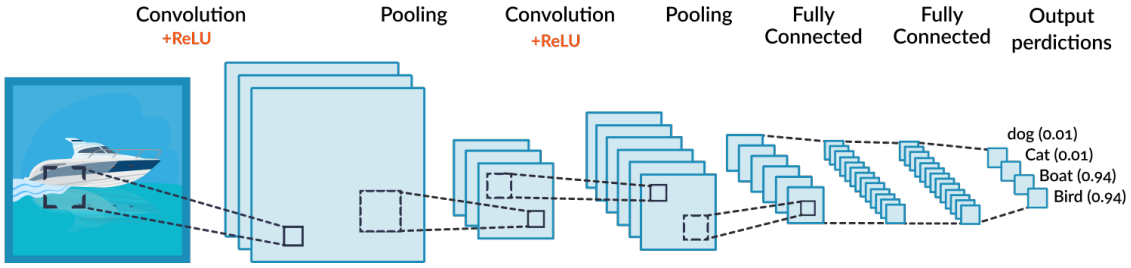
$$w^{(1)} \;=\; w^{(0)} - 2 \cdot \text{Error} \cdot x \cdot \text{Délka Kroku} \tag{18}$$

$$b^{(1)} \;=\; b^{(0)} - 2 \cdot \text{Error} \cdot \text{Délka Kroku} \tag{19}$$

Metodu „Klesání podle gradientu“ provádíme v epochách. Jedna epocha značí proces, při kterém se použije celý tréninkový set na nastavení parametrů, váhy w a prahu b [33]. Představme si datový set obsahující 200 vzorků, neboli řádků. Použijeme způsob mini-dávkového gradientu, zvolíme velikost jedné dávky rovno 20 vzorkům a počet epoch nastavíme na 500. To znamená, že celý dataset bude rozdělen na $200 \;:\; 20 \;=\; 10$ dávek, každá čítající 20 vzorků a parametry modelu budou upraveny po každé dávce. Aby všechny vzorky v datasetu prošli modelem a byla splněna podmínka jedné epochy, je potřeba upravit parametry celkem 10x. Při množství 500 epoch to znamená, že celkem bude použito 5000 dávek, každá o velikosti 20 vzorků.

2.6.6 Konvoluční neuronové sítě

Ačkoliv již v roce 1996 dokázal počítač *Depp Blue* od *IBM* porazit Garryho Kasparova, světového šampióna v šachu, donedávna nebyly počítače schopné provádět jednoduché vizuální úkoly, jako například rozeznat počet osob na fotografii či rozeznat psa od člověka. Člověk splní takový úkol hravě. Podívá se na fotografii, rozliší klíčové prvky a tvary, na základě kterých určí, zda se jedná o psa či člověka. Tuto vlastnost ale počítače nemají, a proto je nutné, převést obrázek do podoby, kterou snadno zpracuje. S tímto problémem si poté poradí konvoluční vrstvy v konvoluční neuronové síti, jednou z mnoha druhů umělých neuronových sítí. Konvoluční neuronové sítě, známé pod zkratkou CNN z anglického *Convolutional Neural Networks*, vznikly studiem mozkové zrakové kůry v osmdesátých letech minulého století. Důležitým milníkem se stala práce, kterou napsal francouzský počítačový vědec Yann LeCun [66]. Díky narůstajícímu výkonu strojů a množství obrazových dat v nedávné době, dokázaly konvoluční neuronové sítě dosáhnout nadlidských výkonů v oblasti strojového vidění, kde dokáží výrazně redukovat počet parametrů bez zásadní ztráty kvality modelu, a díky tomu se používají při zpracování obrazu a textu, kde svým výkonem hravě porazí standardní neuronové sítě [33]. Pohánějí dnes známé technologie jako samořídící auta, vyhledávání dle obrázku nebo aktivní překladače, které dokáží přeložit text v reálném čase pouhým namířením fotoaparátu na text [13]. Dále se využívají v agrokultuře, kde pomocí senzorů dokážou analyzovat snímky plodin a predikovat jejich zdraví, nebo ve zdravotnictví, kde pomáhají včasně diagnostikovat nemoci jako cukrovka, zápal plic nebo rakovinu, jenž je i úkolem této práce [67].



Obr. 24: *Architektura konvoluční neuronové sítě [67].*

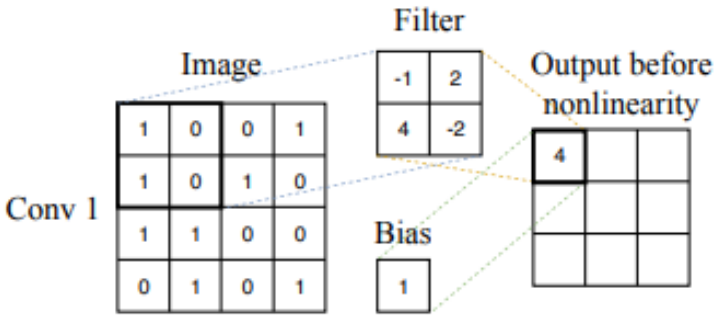
Konvoluční neuronová síť se liší od tradiční architektury vícevrstvého perceptronu svými operacemi s názvy „konvoluce“, anglicky *convolution* a „sdružování“, anglicky *pooling*, které dokáží zredukovat obrázek na podstatné prvky, a použít je na porozumění a klasifikací snímku. Architektura konvoluční neuronové sítě je vidět na Obr. 24. Mezi základní stavební prvky CNN patří konvoluční vrstva s aktivační funkcí, typicky funkcí ReLu. Dále obsahuje sdružovací *pooling* vrstvu, která má za úkol výrazně zredukovat velikost matice a následuje tradiční vícevrstvý perceptron, jehož jednodimenzionální výstup, který obdrží největší míru pravděpodobnosti, je výsledek klasifikace. V závislosti na architektuře sítě může model obsahovat libovolný počet konvolučních a sdružovacích vrstev.

Při použití obvyklé neuronové sítě na úkoly rozpoznání obrazu je kladen obrovský nárok na výkon, proto je lze použít pouze na malé snímky. Například obrázek o velikosti 100 x 100 má 10 000 pixelů. Při použití 1000 neuronů v první vrstvě to znamená celkem 10 000 000 spojení a to se jedná pouze o první vrstvu. CNN řeší tento problém částečně propojenými vrstvami pomocí filtrů [68].

Konvoluční vrstva Konvoluční vrstva je nejdůležitějším blokem konvoluční sítě. Neurony v první konvoluční vrstvě nejsou napojeny na každý jeden pixel vstupního snímku, jako u tradiční sítě, ale pouze na pixely v jejím receptivním poli. Síť nemusí analyzovat obrázek jako celek, ale stačí se zaměřit na důležité prvky ve zkoumaném okolí. Při pohledu na snímek ryby nám stačí se podívat na ploutve a dokážeme snadno poznat, že se jedná o rybu a nikoliv o psa. Počítač pracuje na podobném principu, hledá na snímcích podobné prvky a tvary. Proces v konvolučních vrstvách se nazývá konvoluce, kde je vstupní snímek označen písmenem X a filtr (kernel) písmenem f.

$$Z = X * f \tag{20}$$

Uvažujme obrázek o rozměrech 4x4 pixelů, ke kterému potřebujeme filtr s vhodnou velikostí, kterou zvolíme jako 2x2 pixelů. Oba vstupy vložíme do rovnice 20 a pomocí matematické operace dostaneme výstup na Obr. 25. Pro první konvoluční vrstvu získáme ve výstupu hodnotu 4 výpočtem $(1 * -1) + (0 * 2) + (1 * 4) + (0 * -2) + 1 = 4$, kde je *Bias* neboli práh = 1. Výstup konvoluční operace se nazývá aktivační mapa, nebo také mapa funkcí. Je možné použít větší krok, vzdálenost mezi dvěma receptivními poli [68][33].



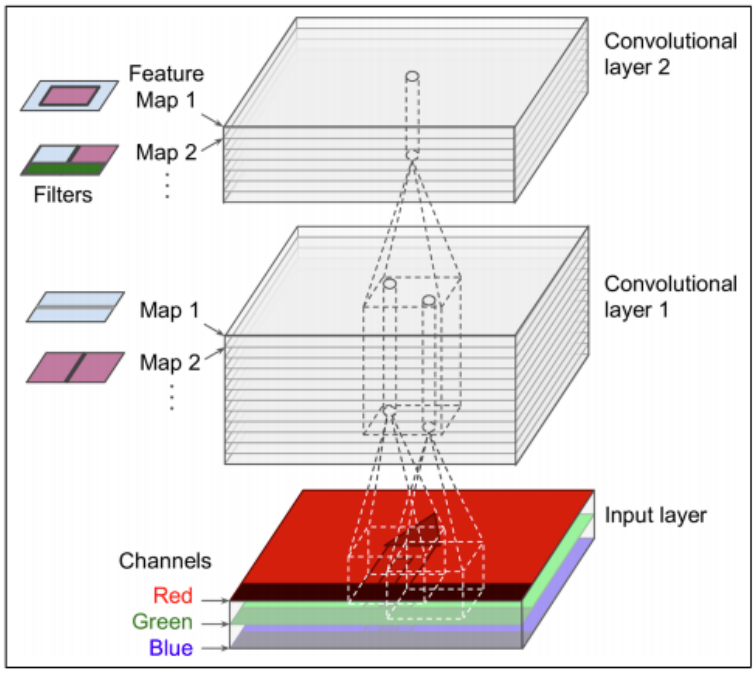
Obr. 25: *Ukázka operace konvoluční vrstvy [33].*

Filtry jsou reprezentace vah pro jednotlivé neurony, pomocí kterých se lze zaměřit ve snímcích na odlišné prvky. Na obr. 26 lze vlevo vidět vstupní fotografii ženy a vpravo výsledky konvoluční operace, aktivační mapy, získané pomocí 2 odlišných filtrů. Filtry zvýrazní do aktivační mapy místa na vstupním snímku, která jsou nejvíce podobná filtru. Konvoluční neuronová síť během tréninku nalezne nejvhodnější filtr pro daný úkol a učí se

je kombinovat do komplexnějších vzorů. Poté lze například vyvodit místa na snímku, kde jsou oba filtry aktivní [68].

Obr. 26: *Ukázka použití filtrů [69].*

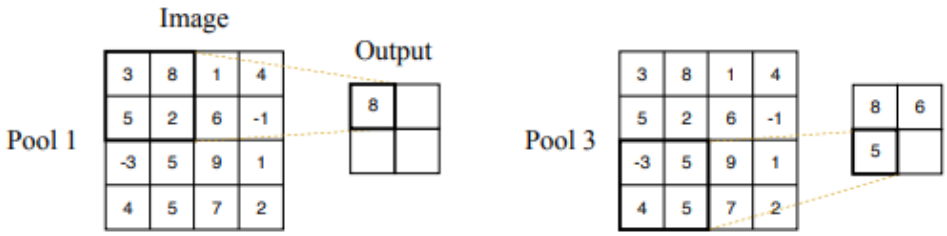
Konvoluční vrstva se skládá z několika aktivačních map a v rámci jedné aktivační mapy je použit stejný filtr na všechny neurony. Vrstva uplatňuje souběžně několik filtrů na vstupní snímky, díky čemuž dokáže zachytit podstatné prvky na každém místě zkoumaného obrázku. Důležité je upozornit, že tradiční barevný obrázek je složen ze tří mezivrstev, každá pro jeden barevný kanál. Barevné snímky jsou složeny z barevného spektra modré, zelené a červené barvy a každá z těchto barev potřebuje vlastní vstupní vrstvu. Snímky v odstínech šedi mají pouze jednu vrstvu, a intenzita každého pixelu je vyměřena hodnotou 0 až 256 (od bíle do černé). Některé snímky mohou mít dokonce více vrstev než 3, například některé satelitní snímky zachycují infračervené záření [13].



Obr. 27: *Detail konvolučních vrstev [13].*

Sdružovací vrstva Sdružovací *pooling* vrstva je často využívaná technika konvolučních sítí. Sdružování funguje na podobném principu jako konvoluce, ovšem místo

použití tréninkem vybraných filtrů se použijí fixní neměnné filtry, většinou filtry hledající maximum nebo průměr. Stejně jako o konvoluce lze zde nastavovat hyperparametry jako velikost filtru a délku kroku. Pro příklad zvolíme velikost filtru 2 x 2, délku kroku 2 a použijeme maximální filtr, nejběžněji používaný sdružovací filtr. Použití sdružování lze vidět na obr. 28 a nastavené hyperparametry nám sníží velikost snímku a tím množství parametrů o 75%. Snížením parametrů docílíme nižší paměťové a výkonové náročnosti[13][33].



Obr. 28: *Ukázka operace sdružování [33].*

Plně propojená vrstva Posledním blokem konvoluční sítě je plně propojená vrstva, tedy tradiční neuronová síť popsaná na začátku kapitoly 2.6. Vstupem plně propojené vrstvy musí být jednodimenzionální vektor. 2D výstup předchozí vrstvy je tedy nutné zploštit, anglicky se nazývá proces *flattening*. Vstupní data putují neuronovou sítí, která predikuje svůj výsledek. Výstupní neuron, který získá největší pravděpodobnost, je výsledek klasifikace [13][67].

Architektury CNN Po představení všech stavebních prvků konvoluční neuronové sítě je důležité správně poskládat jednotlivé komponenty do sebe. Typická konvoluční síť je tvořena několika konvolučními vrstvami s aktivační funkcí (tradičně funkcí ReLU), dále sdružovací vrstvou a poté znovu několika konvolučními vrstvami se sdružovací vrstvou v závěsu a tak dále. Na konci se vždy nachází tradiční neuronová síť s finální výstupní vrstvou. Množství a poloha jednotlivých vrstev hraje zásadní roli výkonu a úspěšnosti konvoluční sítě. Během let se vědci snaží vymyslet nejlepší architekturu pro vysokou úspěšnost predikce. Mezi nejznámější architektury patří *LeNet-5*, zřejmě nejznámější architektura popsaná otcem konvolučních sítí v roce 1998 Yannem LeCunem, *AlexNet*, vítěz několika soutěží díky své nízké chybovosti okolo 17%, či *ResNet*, která dosáhla chybovosti pod 3,6 %. Do své aplikace jsem se rozhodl použít architekturu *MobileNetV2*.

3 Praktická část

Cílem praktické části práce je vytvoření aplikace na klasifikaci kožních lézí. Vytvoření aplikace se skládá z několika dílčích kroků. Prvním krokem je získání potřebného množství

dat, která jsou nutná k natrénování neuronové sítě. Po získání příslušných dat se v několika krocích vytvoří model neuronové sítě. Mezi tyto kroky patří příprava dat, anglicky takzvaný *data preprocessing*, ve kterém se data očistí, upraví a nachystají na vstup do neuronové sítě. Na celém procesu tvorby modelu pracuji v jazyce *Python* s příslušnými knihovnami jako *NumPy*, *Pandas*, *Scikit-learn*, *TensorFlow* či *Keras*. Dalším krokem je vytvoření architektury neuronové sítě, tedy navrhnutí počtu vrstev, neuronů, přenosových funkcí a dalších parametrů. Často se ale využívají již vytvořené architektury neuronových sítí, které jsou navrženy na kategorický problém, například architektur neuronových sítí na zpracování fotografií a snímků je vytvořená celá řada. Jejich použitím se lze vyhnout dlouhotrvajícímu testování při výběrů správného počtu vrstev, neuronů a přenosových funkcí. Jejich použitím také dosáhneme většinou lepších výsledků, jelikož architektury jsou navrženy a otestovány velkým počtem osob. Po úspěšném vytvoření architektury neuronové sítě je čas na samotné trénování, při kterém jsou do architektury sítě vpouštěna upravená data. Model se snaží predikovat vkládaná data a postupně se učí ze svých chyb, úspěšnost trénování opět závisí na nastavení širokého počtu parametrů. Po úspěšném trénovaní proběhne konečná validace a zobrazení výkonu neuronové sítě. Pokud bude model dosahovat požadujících výsledků, uložíme jej do souboru, a ponecháme pro následné vložení do aplikace. Vytvoření aplikace lze provést různými metodami. Ve své práci jsem se přiklonil k vytvoření webové aplikace v jazyce *Python* s pomocí volně dostupné knihovny *Streamlit*.

3.1 Data

Základním kamenem každé aplikace využívající algoritmů strojového učení jsou data. V našem případě je nutné mít k dispozici velký vzorek fotografií kožních lézí. Tento požadavek splňuje dataset „The HAM10000“[3], neboli *Human Against Machine with 10000 training images*, v překladu „Lidé proti strojům s 10000 tréninkovými snímky“. Tento volně přístupný dataset snímků kožních lézí byl v roce 2018 autory zpřístupněn v rámci soutěže obrazové klasifikace, kterou pořádala organizace *ISIC*, *The International Skin Imaging Collaboration*. Soubor 10015 snímků kožních lézí byl tvořen déle než dvacet let na dvou odlišných místech, na oddělení dermatologie na lékařské univerzitě ve Vídni a v lékařské praxi doktora Cliffa Rosenahla v Queenslandu. Rozdílný původ snímků s sebou přináší vítanou diverzifikaci snímků. Lidé v Austrálii mají například více chronických onemocnění kůže díky horkému australskému počasí.

Snímky jsou seřazeny do sedmi kategorií kožních lézí. První kategorie je **Actinic Keratoses a Intraepithelial Carcinoma (akiec)** a jedná se o běžné neinvazivní varianty spinocelulárního karcinomu. Tyto novotvary lze běžně léčit lokálně bez chirurgického zákroku, je ovšem nutný včasný zásah odborníka. **Basal cell carcinoma (bcc)** je nejen nejrozšířenější formou rakoviny kůže, ale také nejběžnějším typem rakoviny vůbec. Díky pomalému růstu ji lze při včasném odhalení snadno odstranit, ovšem při zanedbání prevence

nastane destruktivní průběh nemoci. Další skupinou je **Benign keratosis (bkl)**. Jedná se benigní novotvar různých tvarů a různého zbarvení, obvykle hnědého, černého nebo světlého. Pohledem připomínají bradavice a většinou přibývají s věkem. Jejich odstranění je ve většině případech velmi snadné, ovšem není často ani nutné. **Dermatofibromy (df)** jsou neškodné benigní velmi časté kožní nálezy a jejich odstranění je možné v případě kosmetického problému. **Melanocytic nevi (nv)** jsou nejběžnější kategorií kožních lézí. Většinou se jedná o neškodné pihy, a jejich odstranění je opět nutné pouze, pokud představují kosmetický problém. **Melanoma (mel)** je velmi závažný a nejagresivnější typ kožního nádoru. Včasné odhalení je u tohoto nádoru kritické, a často rozhoduje o přežití pacienta. Poslední kategorie **Vascular skin leisons (vasc)** jsou ve většině případů mateřská znaménka. Jedná se tedy o benigní druh kožní léze a odstranění je opět možné z kosmetických důvodů [3].

3.2 Data preprocessing

Jak již bylo zmíněno výše, data je nutné před použitím očistit a upravit do přijatelného tvaru. Úpravy se liší podle použitého typu dat, fotografie upravíme jiným způsobem než číselná data či text. Pro potřeby počítače je hlavní převést data na formát, kterým stroje rozumí, tedy na jedničky a nuly. Po stáhnutí požadovaného datasetu je nutné nahrát všechna data a fotografie do programu. Dataset „The HAM10000“ obsahuje dvě hlavní části, samotné snímky a tabulku s informacemi o jednotlivých kožních lézí. S těmito informace jsme schopni přiřadit fotografii k diagnóze. Obecně patří mezi hlavní a základní úpravy dohledání a případné doplnění chybějících dat. Jelikož používáme data určená a připravená pro výzkum, tento krok nebude aplikován.

Na obr. 29 je vidět vzorek dat obsahující informace o kožní lézi a o pacientovi. Z příkladu lze pozorovat, že pro jedno ID kožní léze, **lesion_id** můžeme mít k dispozici více snímků, označených pomocí **image_id**. Takové duplicity následně odstraníme z testovací sady, kde by zkreslovaly výkon modelu. Pomocí určené diagnózy kožní léze **dx** jsme schopni natrénovat a následně validovat výkonnost vytvořeného modelu neuronové sítě. **dx_type** nám určuje, jakým způsobem byla určena výsledná diagnóza kožního výrůstku. Více jako polovina všech snímků v použitém datasetu je ověřena pomocí histopatologie. Jedná se o mikroskopické vyšetření vzorku tkáně, který byl získán při biopsii nebo v průběhu operace, jenž by měl přinést výsledek s naprostou přesností. Mezi další způsoby určení diagnózy na použitých snímcích patří lékařský konsensus, vyšetření vzorku na konfokálním mikroskopu nebo vyhodnocení delší dobu po léčbě. Informace jako věk, pohlaví a umístění léze, tedy sloupce **age**, **sex** a **localization**, je vhodné mít k dispozici pro analýzu, ovšem k natrénování modelu neuronové sítě nejsou důležité.

- **lesion_id**: ID kožní léze
- **image_id**: ID snímku kožní léze

– **dx**: zkratka prokázané diagnózy kožní léze

– **dx_type**: princip určení diagnózy kožní léze

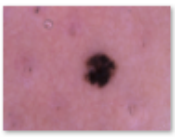
– **age**: věk pacienta

– **sex**: pohlaví pacienta

– **localization**: místo výskytu kožní léze

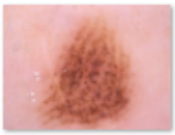
	lesion_id	image_id	dx	dx_type	age	sex	localization
0	HAM_0000118	ISIC_0027419	bkl	histo	80.0	male	scalp
1	HAM_0000118	ISIC_0025030	bkl	histo	80.0	male	scalp
2	HAM_0002730	ISIC_0026769	bkl	histo	80.0	male	scalp
3	HAM_0002730	ISIC_0025661	bkl	histo	80.0	male	scalp
4	HAM_0001466	ISIC_0031633	bkl	histo	75.0	male	ear
64	HAM_0001751	ISIC_0024698	nv	consensus	70.0	male	face
767	HAM_0004453	ISIC_0031825	bkl	confocal	65.0	female	face
768	HAM_0004453	ISIC_0030677	bkl	confocal	65.0	female	face
769	HAM_0005743	ISIC_0025038	bkl	confocal	60.0	female	face
770	HAM_0000999	ISIC_0029927	bkl	confocal	70.0	female	face
771	HAM_0000999	ISIC_0026567	bkl	confocal	70.0	female	face
832	HAM_0003030	ISIC_0028876	bkl	consensus	75.0	female	face
833	HAM_0003030	ISIC_0029527	bkl	consensus	75.0	female	face
834	HAM_0006064	ISIC_0032111	bkl	consensus	75.0	male	back
835	HAM_0006064	ISIC_0031132	bkl	consensus	75.0	male	back
1210	HAM_0000559	ISIC_0024693	nv	follow_up	45.0	female	upper extremity
2976	HAM_0004932	ISIC_0032212	nv	follow_up	45.0	female	foot
2977	HAM_0004516	ISIC_0025775	nv	follow_up	45.0	female	neck
2978	HAM_0006510	ISIC_0029828	nv	follow_up	55.0	female	lower extremity
2979	HAM_0007567	ISIC_0031588	nv	follow_up	45.0	female	lower extremity


ISIC_0029306.jpg


ISIC_0029313.jpg

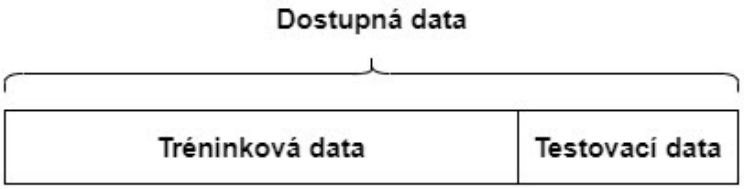

ISIC_0029320.jpg


ISIC_0029327.jpg


ISIC_0029334.jpg

Obr. 29: *Data o snímcích kožních lézí.*

V dalším kroku se musí data rozdělit na tréninkovou a testovací sadu. Knihovna *Scikit-learn* nabízí funkci *train_test_split*, pomocí které lze jednoduše rozdělit dostupná data na požadovaný tvar. Pro testovací vzorek, se kterými určíme končenou úspěšnost modelu, jsem zvolil 17% dat. Toto číslo jsem zvolil po několika zkušebních iteracích a rad od zkušenějších kolegů. Z testovacího vzorku je nutné odstranit již zmíněné duplicity a ponechat od každé kožní léze pouze jeden snímek. Případné duplicity by se nevhodně odrážely ve výsledné úspěšnosti modelu.



Obr. 30: *Ilustrace rozdělení dat na tréninkovou a testovací sadu.*

Po rozdělení máme k dispozici **9077** snímků v tréninkové sadě a **938** v sadě testovací. Testovací vzorek odložíme stranou a do výsledné vzorek nepoužijeme, dále budeme pracovat pouze s tréninkovou sadou. Po analýze tréninkové sady lze pozorovat nerovnoměrné rozdělení snímků pro jednotlivé kategorie.

nv	5954	5954
mel	1074	5920
bkl	1024	5920
bcc	484	5858
akiec	301	5217
vasc	131	5290
df	109	4410

Obr. 31: *Počet snímků v jednotlivých kategoriích tréninkových dat před a po augmentaci.*

Zatímco snímků z diagnózou *nv - Melanocytic nevi* je v tréninkové sadě 5954 a snímků s diagnostikovaným *df - dermatofibromem* máme pouze 109. Pro správné natrénování modelu je takto vysoká nerovnoměrnost velice nevhodná a je nutné data upravit. Vhodnou metodou pro tento problém je obrazová augmentace, technika používaná k umělému zvětšení velikosti tréninkového datasetu vytvořením modifikovaných verzí snímků. Zvětšení počtu snímků se provádí několika operacemi, které si hrají s natočením, jasem či přiblížením snímků. Obrazovou augmentací jsem docílil zvýšením počtu snímku, který lze pozorovat v pravém sloupci na obr. 31. Přípravou dat jsem vytvořil rovnoměrně rozložený tréninkový dataset, který dále použijeme k natrénování modelu neuronové sítě. Společně s ním byla vytvořena testovací sada, očištěná o duplicitní snímky, která bude použita ve finálním určení úspěšnosti modelu.

3.3 Tvorba modelu

S upraveným tréninkovým datasetem nastává práce na samotném modelu neuronové sítě. Prvním krokem je volba či vytvoření neuronové sítě. Ve své práci použiji architekturu konvoluční neuronové sítě *MobileNetV2*. Jedná se o architekturu neuronové sítě předtrénovavnou na databázi snímků *ImageNet*, obsahující přes 14 milionů snímků určených k výzkumu softwaru pro vizuální rozpoznávání objektů. Takto připravené sítě mají optimálně nastavené velikosti vah, prahů a dalších parametrů a jejich použití je v praxi velmi běžné.

Předtrénované architektury neuronových sítí se samozřejmě stále musí upravit a alespoň část vrstev natrénovat na cílových datech. Hlavní modifikací je úprava koncových vrstev neuronové sítě, které neodpovídají výstupním požadavkům. Jak již bylo řečeno v práci výše, počet výstupních neuronů musí odpovídat počtu výstupních tříd, tedy počtu diagnóz kožních lézí. V mém případě jsem odstranil 2 poslední vrstvy sítě a nahradil je nejprve *Dropout* vrstvou, která pomáhá proti přetrénování modelu, a výstupní *Dense* vrstvou obsahující 7 výstupních neuronů. Jako aktivační funkci pro výstupní vrstvu jsem zvolil funkci *Softmax*. Při predikci se dle $\%$ aktivace neuronů ve výsledné vrstvě určí predikce diagnózy kožní léze. Při 100% predikci snímku s diagnózou na melanom bude odpovídající výstupní neuron hlásit 100% aktivaci a ostatní neurony, značící ostatní diagnózy kožních lézí, nebudou vykazovat shodu žádnou.

Architektura *MobileNetV2* se skládá celkem ze 156 vrstev a na tréninkové sadě jsem natrénoval posledních 15 vrstev. K tomuto číslu jsem dospěl po diskuzi se zkušenými praktiky strojového učení a dlouhém testování. K sestavení neuronové sítě před samotným trénováním je potřeba ještě zvolit několik parametrů. Mezi další patří volba optimalizátoru a ztrátové funkce. Tyto dva parametry si lze představit jako nástroje používané ke snížení chyb a zvýšení přesnosti. Ztrátová funkce nám definuje chybu, kterou máme minimalizovat, a optimalizátor nám určí, jakým způsobem chybu snížit. Jako ztrátovou funkci jsem zvolil *categorical crossentropy*, neboli kategorickou křížovou entropii, funkce vytvořená přímo pro kategorickou klasifikaci a vhodná pro konvoluční neuronové sítě. Jako optimalizátor jsem zvolil Adam, široce využívaný algoritmus, který je rozšířením stochastického gradientu a v posledních letech se díky nadprůměrným výsledkům využívá v oblasti počítačového vidění velmi často.

Jednotlivým kategoriím lze nastavit před trénováním modelu váhu, díky které se sí více zaměřují na danou kategorii. Melanomu jsem nastavil váhu 3, díky které bude výsledný model citlivější na tuto diagnózu a zvýším tím šanci na odhalení. K samotnému trénování nám zbývá pouze určit počet epoch, velikost *batche*/dávky a počet kroků v jednotlivých epochách. Počet epoch jsem zvolil 30 a snímků v jedné dávce 10, opět po dlouhém testování a konzultacích s odborníky. Počet kroků v jedné epoše se odvíjí od celkového počtu vzorků a počtu vzorků v jedné dávce, vypočítá se dle rovnice:

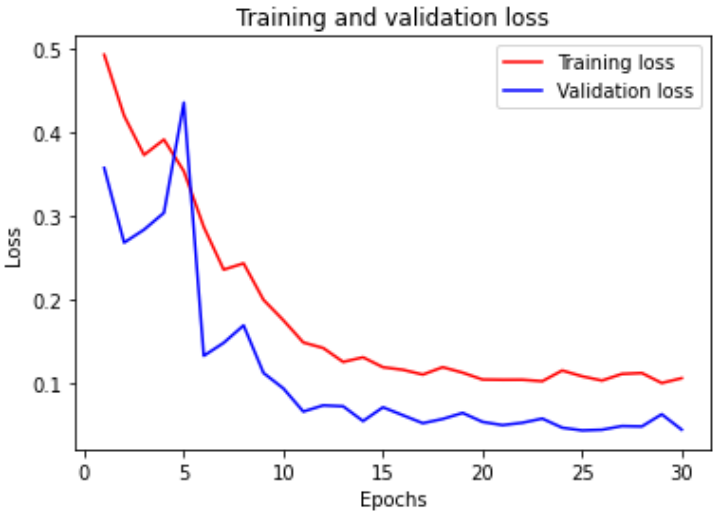
$$\text{Počet kroků} \;=\; \frac{\text{Celkový počet vzorků}}{\text{Počet vzorků v dávce/batchi}} \tag{21}$$

V mém případě dostanu celkem 908 kroků pro testovací sadu a 386 kroků pro validační sadu. Po nastavení parametrů se může spustit trénování neuronové sítě. Samotný trénink je nejnáročnější proces na výkon počítače a dle schopností zařízení a zvolených parametrů může trvat od několika desítek minut po několik hodin. V mém případě trval trénink neuronové sítě vždy kolem 120-ti minut.

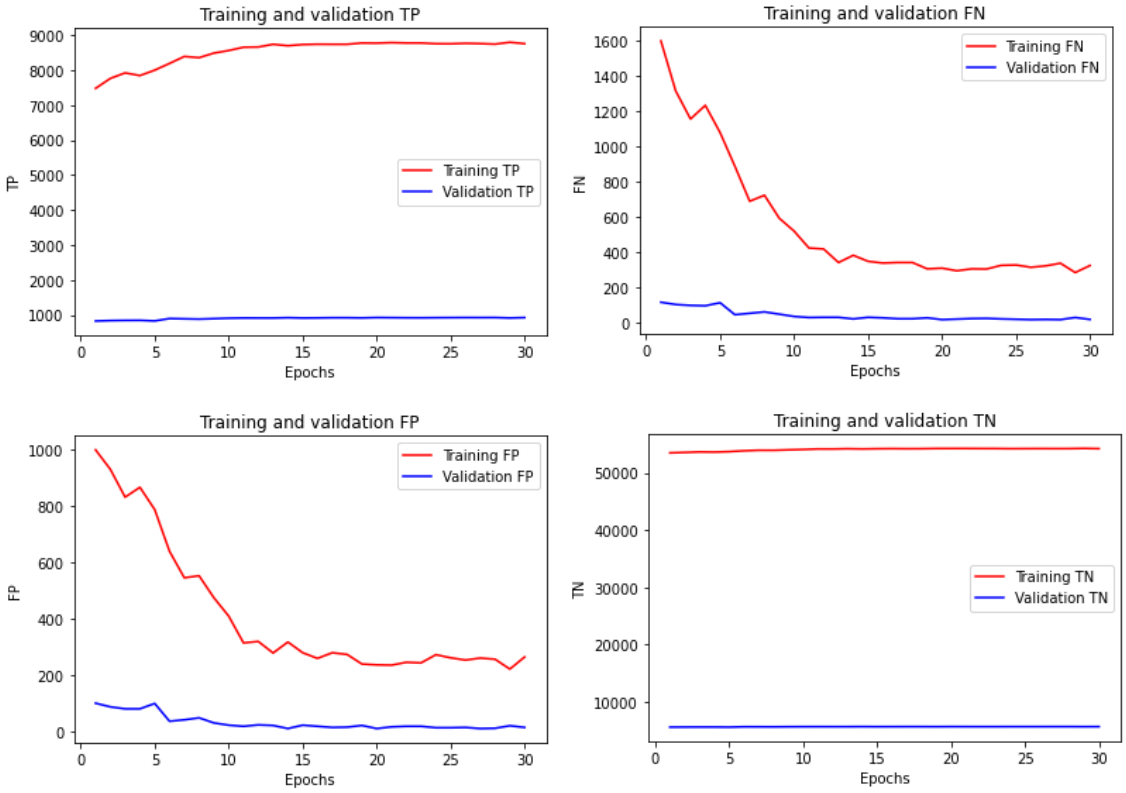
3.4 Výsledky

Po úspěšném natrénování modelu neuronové sítě přichází na řadu evaluace pomocí testového datasetu. Do právě vytvořeného modelu nahrajeme testovací snímky a necháme změřit důležité metriky jako přesnost, sensitivita či specificita. Na obr. 32, 33 a 34 lze pozorovat průběh trénování neuronové sítě a stále se zlepšující přesnost, snižující ztrátu a lepší poměr parametrů TP, TN, FP a FN. Tato zlepšení jsou odrazem úprav parametrů na konci každé epochy. Výsledná přesnost modelu neuronové sítě na testovacích datech je $88,7\%$.

Obr. 32: *Vývoj přesnosti během tréninku neuronové sítě.*



Obr. 33: *Vývoj ztráty během tréninku neuronové sítě.*



Obr. 34: *Vývoj parametrů TP, TN, FP a FN během tréninku neuronové sítě.*

Primárními stavebními kameny metrik, které použijeme k evaluaci klasifikačního modelu neuronové sítě patří čtyři parametry:

- **True Positive TP**: počet správně predikovaných snímků za pozitivní třídu
- **False Positive FP**: počet správně predikovaných snímků za negativní třídu
- **True Negative TN**: tedy počet nesprávně predikovaných snímků za pozitivní třídu
- **False Negative FN**: počet nesprávně predikovaných snímků za negativní třídu

Hodnoty jednotlivých parametrů jsou shrnuty v následující tabulce:

TP	FP	TN	FN
822	108	5520	116

TP je tedy případ, kdy model správně určí, zda se jedná o danou diagnózu, a naopak FP je případ, kdy model nesprávně určí že se jedná o danou diagnózu. Analogicky TN vyjadřuje případy správného odmítnutí dané diagnózy a FN nesprávně přijme jinou diagnózu. Pomocí těchto parametrů lze vypočítat několik metrik, kterými se hodnotí úspěšnost modelu. Mezi takové metriky patří *Precision*, *Recall / Sensitivity*, *F Score* a *Specificity*:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{22}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP + FN}} \tag{23}$$

$$\text{F1 Score} = \frac{2 * \text{Precision * Recall}}{\text{Precision + Recall}} \tag{24}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN + FP}} \tag{25}$$

Mezi další důležité parametry měřící úspěšnost modelu patří metrika s názvem AUC, anglická zkratka pro *Area Under Curve*, neboli plocha pod křivkou. Tato metrika měří plochu pod křivkou ROC *Receiver Operating Characteristic curve*. Křivka ROC zobrazuje výkonnost klasifikačního modelu a zobrazuje dva parametry, takzvaný *True Positive Rate (TPR)* tedy již známý *Recall* a *False Positive Rate (FPR)*:

$$\text{TPR} = \frac{\text{TP}}{\text{TP + FN}} \tag{26}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP + TN}} \tag{27}$$

Ideální výsledek by znamenal, že všechny snímky odhadl model bezchybně. Mnou vytvořený model dosáhl hodnotu AUC = 0.9753. Veškeré výsledky, jenž obsahují hodnoty v intervalu od nejhorší 0 po nejlepší 1, jsou vyneseny do tabulky níže:

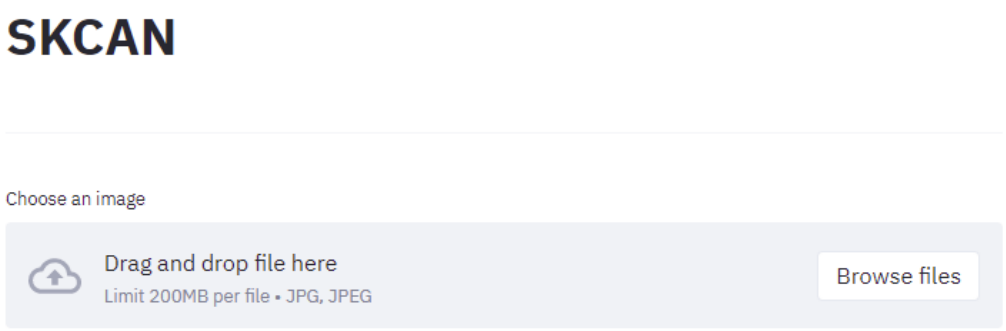
Accuracy	Precision	Recall	F1 Score	Specificity	AUC
0.8871	0.8839	0.8763	0.8801	0.9794	0.9753

Výsledný model konvoluční neuronové sítě si uložíme do souboru s příponou *.h5*, která označuje hierarchický datový formát verze 5, používaný pro uchování a organizaci velkých struktur a množství dat. Tento soubor bude nahrán do webové aplikace a predikovat jednotlivé diagnózy kožních lézí

3.5 Tvorba aplikace

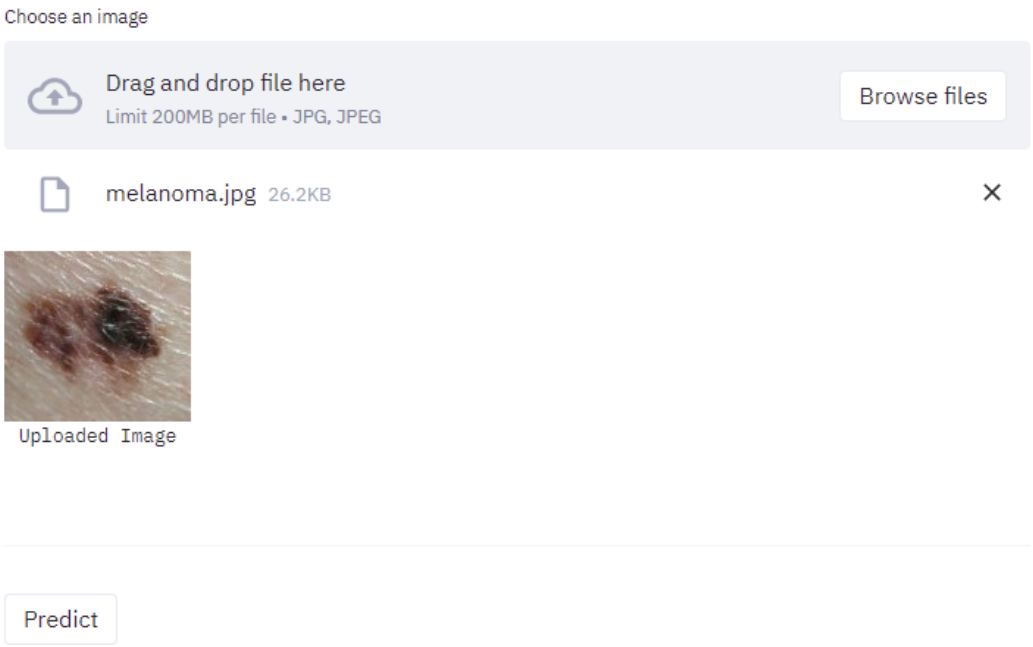
Po úspěšném vytvoření modelu konvoluční neuronové sítě je nutné vytvořit webovou aplikaci pro preventivní diagnózy kožních onemocnění. Při zpracování jsem se snažil vytvořit jednoduchou, výkonově nenáročnou a přístupnou aplikace, kterou může použít co nejširší spektrum lidí. Vzhledem k těmto kriteriím jsem zvolil webovou aplikaci, tedy aplikaci běžící na webovém serveru. Tato volba zajistí dostupnost a nenáročnost programu, a k jejímu použití stačí pouze zadat do webového prohlížeče URL adresu uvedenou v

online repozitáři na URL `https://github.com/XtraHamster/SKCAN`, na které je aplikace nahrána. K naprogramování programu jsem stejně jako k tvorbě modelu neuronové sítě použil programovací jazyk Python. K vytvoření webové aplikace jsem použil poměrně nový a zajímavý *open-source framework Streamlit*, s jehož pomocí lze snadno vytvořit přehledné a vizuálně zajímavé webové aplikace strojového učení. Volbou tohoto *frameworku* jsem se snažil odlišit od většiny webových aplikací vytvořených v Pythonu, které většinou využívají velmi známé nástroje jako *Flask* nebo *Django*. Dále nabízí *framework* silnou, stále se rozrůstající základnu uživatelů a nadšenců, kteří se proaktivně podílí na zlepšování stávajících a přidávání nových funkcí. V procesu sestavování aplikace jsem narazil na několik problémů, ovšem s pomocí velmi vlídné komunity jsem všechny nedokonalosti hravě odstranil.



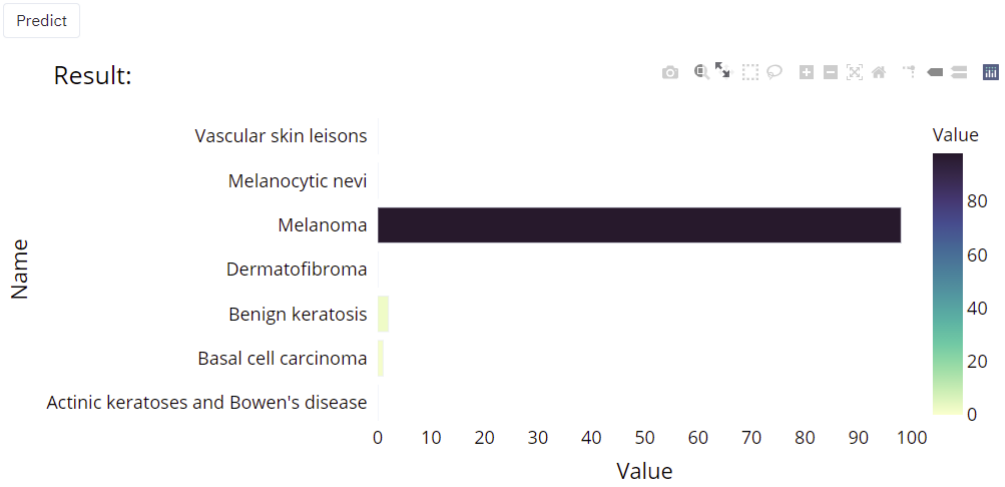
Obr. 35: *Úvodní stránka aplikace.*

Samotná aplikace se skládá z několika částí. Po načtení stránky pomocí dané URL se zobrazí uživateli pohled, který lze pozorovat na obr. 35. Pro aplikaci jsem vytvořil název „SKCAN“ a jedná se o zkratku delšího názvu aplikace *Skin Cancer Scan*, v češtině „Sken Rakoviny Kůže“. Pod názvem „SKCAN“ vybízí aplikace uživatele k nahrání souboru pořízené fotografie kožní léze. Fotografie kožní léze by měla vyplňovat většinu plochy a měla by být umístěna uprostřed fotografie. Dále musí být použit snímek s dostatečnou ostrostí, ve vhodné kvalitě a splňovat kompresní standard formátu „.jpg“ nebo „.jpeg“. Nahrání souboru iniciuje uživatel buď přetáhnutím snímku do šedého pole, nebo přímým nahráním pomocí tlačítka *Browse files*.



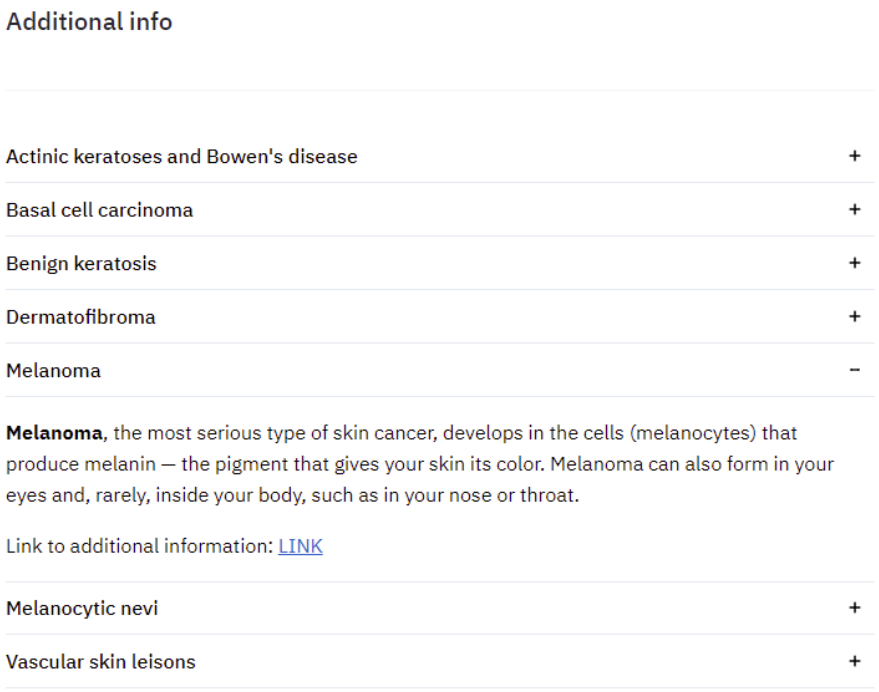
Obr. 36: *Nahrání snímku kožní léze do aplikace.*

Po nahrání snímku kožní léze se zobrazí náhled použité fotografie na kterém lze provést kontrolu. Pro finální diagnózu stačí nyní kliknout na tlačítko „Predict“.



Obr. 37: *Výsledná procentuální diagnóza snímku.*

Během okamžiku predikuje model neuronové sítě procentuální pravděpodobnosti diagnózy kožní léze. Výsledný graf zobrazuje na ose x shodu v % v rozmezí $< 0; 100 >$ a na ose y kategoricky sedm definovaných diagnóz. Pro přehlednost je shoda doplněna značením ve formě barevného gradientu.



Obr. 38: *Doplňující informace k jednotlivým diagnózám.*

Po predikci lze ve spodní části aplikace v sekci *Additional info* dohledat doplňující informace k jednotlivým diagnózám s odkazem na odbornou dermatologickou stránku obsahující podrobný popis, příčiny, následky a případně léčbu daného kožního útvaru a v případě podezření na nebezpečný melanom, či jinou vážnější nemoc se doporučuje návštěva odborníka.

4 Závěr

Ve své práci jsem shrnul aktuální situaci okolo problematiky rakoviny kůže v České republice a ve světě. Dále jsem popsal aktuální snahy s aplikacemi určenými pro danou problematiku a poskytl jsem teoretický základ všech použitých technologií, tedy strojového učení a umělých neuronových sítí. Cílem práce bylo vytvoření aplikace, která dokáže určit nejpravděpodobnější diagnózu kožní léze, a poskytne snadnou, rychlou a dostupnou prevenci proti nebezpečným kožním nemocem. Pro vytvořenou aplikaci bylo nutné vytvořit prediktivní model neuronové sítě, který provádí samotnou klasifikaci. Takto nastavený cíl práce byl úspěšně splněn, vytvořil jsem model neuronové sítě s přesností $88,7\%$, který jsem následně vložil do vytvořené webové aplikace. Webová aplikace je nasazena na webovém serveru a je dostupná kdykoliv z kteréhokoliv osobního počítače či chytrého mobilního telefonu. Pro zobrazení programu stačí uživateli vložit do webového prohlížeče URL adresu `skcan.herokuapp.com`.

Při vývoji jsem narazil na dvě hlavní limitace, které nedovolí posunout přesnost modelu neuronové sítě na vyšší stupně procenta. První limitací je velikost použitého datasetu snímků kožních lézi. Tento problém je ovšem alfou a omegou všech aplikací neuronových sítí, jelikož úspěšnost modelu vždy stojí na kvalitě a množství poskytnutých dat k natrénování modelu. Se zlepšujícími se možnostmi, dostupností kvalitních fotoaparátů a ostatní podpůrné techniky lze ovšem očekávat stálé zlepšování přesností obdobných aplikací. Při pohledu na možnosti a velikosti dostupných datasetů deset let zpět v čase lze pozorovat exponenciální pokrok a neočekává se jeho zpomalení. Tato limitace je tedy spíše technického směru a očekává se kontinuální zlepšování v blízké i daleké budoucnosti. Druhou limitací, na kterou jsem při vývoji modelu neuronové sítě narazil, je výkon zařízení, na kterém jsem prováděl trénování neuronové sítě. Trénování neuronové sítě je výkonnostně náročný proces, zvláště při použití dat ve formě fotografií, a stává se určitou limitací při testování. Velké vědecké týmy používají pro podobné testování řádově výkonnější zařízení a čas pro natrénování neuronové sítě se mnohonásobně zkrátí. Tato limitace nebyla zásadní, ovšem při možnosti použití podobně výkonného zařízení by se dalo provést více testů a možná nalézt i vhodnější nastavení parametrů neuronové sítě.

Při tvorbě modelu a aplikace jsem se snažil odlišit od již vytvořených prací podobného směru. Například při tvorbě modelu neuronové sítě jsem použil metodu obrazové augmentace, techniky používané k umělému zvětšení velikosti tréninkového datasetu vytvořením modifikovaných verzí snímků. Díky tomuto kroku jsem rozšířil kategorie snímků kožních lézi a připravil velikostně rovnoměrné sady pro každou ze sedmi kožních diagnóz. Tento krok zlepšil výslednou přesnost neuronové sítě o několik jednotek procent. Jako hlavní osobní přínos vnímám použití nového a nepříliš známého *frameworku* jazyka *Python* na tvorbu webových aplikací *Streamlit*. Na rozdíl od zaběhnutých a nejpoužívanějších *frameworků* jako je *Django* a *Flask*, jsem přijal výzvu vytvořit aplikaci v nepříliš probádaném prostředí. Tento nástroj mě náramně překvapil a procesně i vizuálně pracuje na vysoké úrovni.

Tvorba modelu neuronové sítě a následně webové aplikace mi přinesla mnoho nových teoretických informací, ale hlavně nenahraditelných praktických zkušeností. K pochopení problematiky rakoviny kůže jsem zkoumal množství lékařských dokumentů a prací, abych lépe porozuměl aktuálnímu stavu a následně možnostem léčby daných nemocí. Pro samotný vývoj aplikace a modelu jsem nastudoval materiály ohledně teorie i praktických použití strojového učení a neuronových sítí. Při vývoji modelu neuronové sítě a webové aplikace jsem narazil na spoustu překážek, které jsem musel jednu po jedné odstranit. Tento proces nalezení chyb a následné hledání příčin a řešení pro jejich odstranění vnímám jako největší obohacení, jelikož tento postup věrohodně napodobuje reálné situace z běžného pracovního života.

Jelikož se jedná o hojně zastoupené odvětví v dnešním světe, lze v příštích letech

očekávat nepřetržitý a důsledný výzkum v oblasti neuronových sítí. Stejným způsobem budou v budoucnu k dispozici stále výkonnější a lepší osobní počítače, na kterých bude stále jednodušší a rychlejší provádět testování a vývoj modelů neuronových sítí. Pokud k těmto predikcím připočteme zlepšující se kvalitu a velikost dostupných datových setů, můžeme v krátké budoucnosti očekávat zlepšení přesnosti modelů, stále častější využití obdobných aplikací v běžné praxi a v daleké budoucnosti částečnou náhradu za odborníky v daném směru.

Seznam použité literatury a zdrojů

[1] DUŠEK Vladislav, MUŽÍK Jan, KUBÁSEK Miroslav, KOPTÍKOVÁ Jana, ŽALOUDÍK Jan, VYZULA Rostislav. *Epidemiologie zhoubných nádorů v České republice [online]*. Masarykova univerzita, 2005, [cit. 2020-11-22]. Dostupný z WWW: http://www.svod.cz. Verze 7.0 [2007], ISSN 1802 – 8861.

[2] Výbor České onkologické společnosti ČLS JEP. *Česká republika a rakovina v číslech [online]*. Linkos - česká onkologická společnost, 2018, [cit. 2020-11-22]. Dostupný z WWW: https://www.linkos.cz/narodni-onkologicky-program/co-musite-vedet/ceska-republika-a-rakovina-v-cislech/. ISSN 2570 – 8791.

[3] TSCHANDL Philipp. *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions [online]*. Harvard Dataverse, 2018, [cit. 2020-11-22]. Dostupný z WWW: https://doi.org/10.7910/DVN/DBW86T.

[4] MADER K Scott. *Skin Cancer MNIST: HAM10000 In: kaggle.com [online]*. 2018, [cit. 2020-11-23]. Dostupný z WWW: https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000.

[5] MARSH. *Step wise Approach : CNN Model Web App In: kaggle.com [online]*. 2018, [cit. 2020-11-23]. Dostupný z WWW: https://www.kaggle.com/sid321axn/step-wise-approach-cnn-model-77-0344-accuracy.

[6] ZHU Menglong. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications [online]*. Cornell University, 2017, [cit. 2020-11-24]. Dostupný z WWW: https://arxiv.org/abs/1704.04861.

[7] SIIM ISIC. *SIIM-ISIC Melanoma Classification In: kaggle.com [online]*. 2020, [cit. 2020-11-24]. Dostupný z WWW: https://www.kaggle.com/c/siim-isic-melanoma-classification/overview/timeline.

[8] FANCONI Claudio. *Skin Cancer: Malignant vs. Benign In: kaggle.com [online]*. 2019, [cit. 2020-11-24]. Dostupný z WWW: https://www.kaggle.com/fanconic/skin-cancer-malignant-vs-benign.

[9] CUMMINGS Angel. *ISIC Archive [online]*. ISIC, 2019, [cit. 2020-11-24]. Dostupný z WWW: https://www.isic-archive.com/#!/topWithHeader/tightContentTop/about/isicArchive.

[10] ESTEVA, A., KUPREL, B., NOVOA, R. et al. *Dermatologist-level classification of skin cancer with deep neural networks [online]*. Nature 542, 115–118, 2017 [cit. 2020-11-24]. Dostupný z WWW: https://doi.org/10.1038/nature21056.

[11] . GOOGLE Developers. *Advanced Guide to Inception v3 on Cloud TPU [online]*. Google, 2020 [cit. 2020-11-24]. Dostupný z WWW: https://cloud.google.com/tpu/

docs/inception-v3-advanced.

[12] THEOBALD Oliver. *Machine Learning For Absolute Beginners: Second Edition*. Scatterplot Press, 2018, ISBN 978-1549617218.

[13] GÉRON Aurélien. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 1st Edition*. O'Reilly Media, 2017, ISBN 978-1491962299.

[14] MÜLLER Guido. *Introduction to Machine Learning with Python: A Guide for Data Scientists 1st Edition*. O'Reilly Media, 2016, ISBN 978-1449369415.

[15] KREMER J., STENSBO-SMIDT K., GIESKE F., PEDERSEN K. S. and IGEL C. *Big Universe, Big Data: Machine Learning and Image Analysis for Astronomy [online]*. IEEE Intelligent Systems, vol. 32, no. , pp. 16-22, Mar.-Apr. 2017 [cit. 2020-11-24]. Dostupný z WWW: https://ieeexplore.ieee.org/document/7887648.

[16] BALDI P., SANDOWSKI P. & WHITESON D. *Searching for exotic particles in high-energy physics with deep learning In: Nature.com [online]*. Nature.com, 2014 [cit. 2020-11-24]. Dostupný z WWW: https://www.nature.com/articles/ncomms5308/.

[17] LIBBRECHT M., NOBLE W. *Machine learning applications in genetics and genomics [online]*. Nat Rev Genet 16, 321–332, 2015 [cit. 2020-11-24]. Dostupný z WWW: https://doi.org/10.1038/nrg3920.

[18] KOUROU Konstantina, EXARCHOS P. Themis, EXARCHOS P. KONSTANTINOS, et. al. *Machine learning applications in cancer prognosis and prediction [online]*. Computational and Structural Biotechnology Journal, Volume 13, 2015, Pages 8-17 [cit. 2020-11-24]. Dostupný z WWW: https://doi.org/10.1016/j.csbj.2014.11.005 ISSN 2001-0370.

[19] YANG Y., LIN L. *Automatic Pedestrians Segmentation Based on Machine Learning in Surveillance Video*. 2019 IEEE International Conference on Computational Electromagnetics (ICCEM), Shanghai, China, 2019, pp. 1-3 [cit. 2020-11-24]. DOI: 10.1109/COMPEM.2019.8779084.

[20] DAFFODIL Software. *9 Applications of Machine Learning from Day-to-Day Life [online]*. Medium, 2017 [cit. 2020-11-24]. Dostupný z WWW: https://medium.com/app-affairs/9-applications-of-machine-learning-from-day-to-day-life-112a47a429d0.

[21] AWOYEMI J. O., ADETUNMBI A. O., OLUWADARE S. A. *Automatic Pedestrians Segmentation Based on Machine Learning in Surveillance Video*. 2019 IEEE International Conference on Computational Electromagnetics (ICCEM), Shanghai, China, 2019, pp. 1-3 [cit. 2020-11-24]. DOI: 10.1109/COMPEM.2019.8779084.

[22] MCCARTHY J., FEIGENBAUM E. A. *In Memoriam: Arthur Samuel: Pioneer*

in Machine Learning. AI Magazine [online]. AI Magazine, 11(3), 10 ,1990 [cit. 2020-11-30]. DOI: https://doi.org/10.1609/aimag.v11i3.840

[23] SAMUEL Arthur L. *Some Studies in Machine Learning Using the Game of Checkers [online]*. IBM Journal, 1959 [cit. 2020-11-30]. Dostupný z WWW: http://www2.stat.duke.edu/~sayan/R_stuff/Datamatters.key/Data/samuel_1959_B-95.pdf

[24] KOZA J. R., BENNETT F. H., ANDRE D., KEANE M. A. *Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. In: Gero J.S., Sudweeks F. (eds) Artificial Intelligence in Design '96*. Springer, Dordrecht, 1996 [cit. 2020-11-30]. DOI: https://doi.org/10.1007/978-94-009-0279-4_9

[25] IBM. *704 electronic data-processing machine: manual of opearation [online]*. IBM, 1954 [cit. 2020-12-02]. Dostupný z WWW: http://www.bitsavers.org/pdf/ibm/704/24-6661-2_704_Manual_1955.pdf

[26] WIDROW B. *An Adaptive "ADALINE"Neuron Using Chemical "Memistors"[online]*. Stanford University, 1955 [cit. 2020-12-02]. Dostupný z WWW: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html

[27] UMBRELLO Steven. *AI Winter [online]*. Encyclopedia of Artificial Intelligence: The Past, Present, and Future of AI. Santa Barbara, USA [cit. 2020-12-02]. Dostupný z WWW: https://philpapers.org/rec/UMBAW

[28] HUTCHINS John. *The history of machine translation in a nutshell [online]*. 2005 [cit. 2020-12-02]. Dostupný z WWW: http://www.hutchinsweb.me.uk/Nutshell-2005.pdf

[29] SMITH Chris, MCGUIRE Brian, et al. *The History of Artificial Intelligence [online]*. University of Washington, 2006 [cit. 2020-12-02]. Dostupný z WWW: https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf

[30] BUCHANAN Bruce, G. *A (Very) Brief History of Artificial Intelligence [online]*. AI Magazine, 26(4), 53, 2005 [cit. 2020-12-06]. DOI: https://doi.org/10.1609/aimag.v26i4.1848

[31] HOIST Arnie. *Amount of information globally 2010-2024 In: Statista.com [online]*. 2020 [cit. 2020-12-02]. Dostupný z WWW: https://www.statista.com/statistics/871513/worldwide-data-created/

[32] RUSSEL Stuart, NORVIG Peter. *Artificial Intelligence: A Modern Approach: Fourth edition*. Pearson, 2020, ISBN 9780134671864.

[33] BURKOV Andriy. *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019, ISBN 1999579518.

[34] GARETH James. *Introduction to Statistical Learning*. Springer, 2013, ISBN 9781461471370.

[35] BIRINCI Hasan, KANALMAZ Sevde Ilgaz et al. *A Multiple Linear Regression Approach For Estimating the Market Value of Football Players in Forward Position [online]*. Abdullah Gül University, 2018 [cit. 2020-12-10]. Dostupný z WWW: https://arxiv.org/ftp/arxiv/papers/1807/1807.01104.pdf

[36] LANGLEY P. *The changing science of machine learning [online]*. Mach Learn 82, 275–279, 2011 [cit. 2020-12-10]. DOI: https://doi.org/10.1007/s10994-011-5242-y

[37] CHAKRABARTI Soumen, ESTER Martin, FAYYAD Usama et al. *Data Mining Curriculum: A Proposal (Version 1.0) [online]*. SIGKDD, 2006 [cit. 2020-12-14]. Dostupný z WWW: https://www.kdd.org/exploration_files/CURMay06.pdf

[38] KAUFMANN Morgan. *Data Mining: Concepts and Techniques: 3rd Edition*. Elsevier Science, 2011, ISBN 978-9380931913

[39] ROMEIJN, Jan-Willem. *Philosophy of Statistics [online]*. Metaphysics Research Lab, Stanford University, 2017 [cit. 2020-12-15]. Dostupný z WWW: https://plato.stanford.edu/entries/statistics/

[40] BROEMELING Lyle D. *An Account of Early Statistical Inference in Arab Cryptology [online]*. The American Statistician, 65:4, 255-257, 2012 [cit. 2020-12-15]. DOI: https://doi.org/10.1198/tas.2011.10191

[41] BZDOK Danilo, ALTMAN Naomi, KRZYWINSKI Martin. *Statistics versus machine learning In: nature.com [online]*. Nat Methods 15, 233–234, 2018 [cit. 2020-12-15]. DOI: https://doi.org/10.1038/nmeth.4642

[42] BROWNLEE Jason. *Train-Test Split for Evaluating Machine Learning Algorithms In: machinelearningmastery.com [online]*. Machine Learning Mastery, 2020 [cit. 2020-12-17]. Dostupný z WWW: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

[43] OLUWAROTIMI Samuel. *Re: Is there an ideal ratio between a training set and validation set? Which trade-off would you suggest? [online]*. Research Gate, 2016 [cit. 2020-12-17]. Dostupný z WWW: https://www.researchgate.net/post/Is-there-an-ideal-ratio-between-a-training-set-and-validation-set-Which-trade-off-would-you-suggest/56ec20d6cbd5c22c0101ece6/citation/download

[44] BROWNLEE Jason. *Difference Between Algorithm and Model in Machine Learning In: machinelearningmastery.com [online]*. Machine Learning Mastery, 2020 [cit. 2020-12-18]. Dostupný z WWW: https://machinelearningmastery.com/difference-between-algorithm-and-model-in-machine-learning/

[45] ALBON Chris. *Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning*. O'Reilly Media, 2018, ISBN 978-1491989388

[46] PETERSON Leif E. *K-nearest neighbor In: scholaropedia.org [online]*. Scholaropedia, 2009 [cit. 2020-12-20]. Dostupný z WWW: http://scholarpedia.org/article/K-nearest_neighbor

[47] SUTTON Richard S. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning) (Adaptive Computation and Machine Learning series) second edition*. The MIT Press, 1998, ISBN: 978-0262193986

[48] SHALEV-SCHWARTZ Shai, SHAMMAH Shaked, SHASHUA Amnon. *Safe, Multi-Agent, Reinforcement Learning for Autonomous Driving [online]*. Cornell University, 2016 [cit. 2020-12-21]. Dostupný z WWW: https://arxiv.org/pdf/1610.03295.pdf

[49] VINYALS O., BABUSCHKIN I., CZARNECKI W. M. et al. *Grandmaster level in StarCraft II using multi-agent reinforcement learning In: Nature.com [online]*. Nature 575, 350–354, 2019 [cit. 2020-12-21]. DOI: https://doi.org/10.1038/s41586-019-1724-z

[50] ČIHÁK Radomír. *Anatomie 3*. Grada Publishing, 2016, ISBN: 978-80-247-5636-3

[51] MITCHELL TOM M. *Machine Learning 1st Edition*. McGraw-Hill Education, 1997, ISBN: 978-0070428072

[52] ROSENBLATT Frank. *The Perceptron: A Probabilistic Model for Information storage and organization in the brain [online]*. Cornell University ,1958 [cit. 2020-12-29]. Dostupný z WWW: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf

[53] GOODFELLOW Ian, BENGIO Yoshua, COURVILLE Aaron. *Deep Learning [online]*. MIT Press, 2016 [cit. 2020-12-29]. Dostupný z WWW: http://www.deeplearningbook.org

[54] BROWNLEE Jason. *How to Develop a CNN for MNIST Handwritten Digit Classification [foto] In: machinelearningmastery.com [online]*. Machine Learning Mastery, 2020 [cit. 2021-04-05]. Dostupný z WWW: https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-/digit-classification/

[55] NIELSEN Michael. *Using neural nets to recognize handwritten digits [foto] In: neuralnetworksanddeeplearning.com [online]*. Nerual Networks and Deep Learning, 2019 [cit. 2021-04-05]. Dostupný z WWW: http://neuralnetworksanddeeplearning.com/chap1.html

[56] GAD Ahmad. *Beginners Ask "How Many Hidden Layers/Neurons to Use in Artificial Neural Networks?" In: towardsdatascience.com [online]*.

2019 [cit. 2020-12-30]. Dostupný z WWW: https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks\-51466afa0d3e

[57] BM Nechu. *What is a Recurrent Neural Network (RNN)? In: medium.com [online]*. 2020 [cit. 2020-12-30]. Dostupný z WWW: https://medium.com/swlh/introduction-to-recurrent-neural-networks-rnn-c2374305a630

[58] BOHR (https://stats.stackexchange.com/users/188522/bohr). *Why is step function not used in activation functions in machine learning? In: stackexchange.com [online]*. 2017-12-14 [cit. 2020-12-30]. Dostupný z WWW: https://stats.stackexchange.com/questions/271701/why-is-step-function-not-used-in-activation-functions-in-machine-learning

[59] KRUGER Moshe. *7 Types of Neural Network Activation Functions: How to Choose? In: missinglink.ai [online]*. 2019 [cit. 2020-12-30]. Dostupný z WWW: https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/

[60] SHALEV-SCHWARTZ Shai,. *Understanding Machine Learning (From Theory to Algorithms) 1st Edition*. Cambridge University Press, 2014, ISBN: 978-1107057135

[61] WOOD Thomas. *What is the Sigmoid Function? In: deepai.org [online]*. 2018 [cit. 2021-01-03]. Dostupný z WWW: https://deepai.org/machine-learning-glossary-and-terms/sigmoid-functiony

[62] GLOROT Xavier, BORDES Antoine, BENGIO Yoshua. *Deep Sparse Rectifier Neural Networks [online]*. DIRO, Universit´e de Montr´eal, 2018 [cit. 2021-01-03]. Dostupný z WWW: http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf

[63] MULLA Zeeshan Mohammed. *Cost, Activation, Loss Function|| Neural Network|| Deep Learning. What are these? In: medium.com [online]*. 2020 [cit. 2021-01-04]. Dostupný z WWW: https://deepai.org/machine-learning-glossary-and-terms/sigmoid-functiony

[64] MULLA Zeeshan Mohammed. *Cost function [foto], In: medium.com [online]*. 2020 [cit. 2021-04-05]. Dostupný z WWW: https://medium.com/@zeeshanmulla/cost-activation-loss-function-neural-network-deep-learning-what-are-these-91167825/a4de

[65] PANDEY Parul. *Understanding the Mathematics behind Gradient Descent. In: towardsdatascience.com [online]*. 2019 [cit. 2021-01-04]. Dostupný z WWW: https://towardsdatascience.com/understanding-the-mathematics-behind-gradient-descent-dde5dc9be06e

[66] LECUN Yann, BOTTOU Léon, BENGIO Yoshua, HAFFNER Patrick. *Gradient-Based*

Learning Applied to Document Recognition [online]. Proceedings of the IEEE, 1998 [cit. 2021-01-05]. Dostupný z WWW: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

[67] KRUGER Moshe. *Convolutional Neural Network Tutorial: From Basic to Advanced In: missinglink.ai [online]*. 2019 [cit. 2021-01-05]. Dostupný z WWW: https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-tutorial-basic-advanced/

[68] WU Jianxin. *Introduction to Convolutional Neural Networks [online]*. Nanjing University, China, 2017 [cit. 2021-01-08]. Dostupný z WWW: https://cs.nju.edu.cn/wujx/paper/CNN.pdf

[69] HOOKER Dwight. *Lena Forsén [foto]*. USA, 1973 [cit. 2021-04-03]. Dostupný z WWW: https://en.wikipedia.org/wiki/Lenna#/media/File:Lenna_(test_image).png

[70] PANDEY Pranjal. *Data Preprocessing : Concepts. In: towardsdatascience.com [online]*. 2019 [cit. 2021-04-08]. Dostupný z WWW: https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825

Seznam použitého SW

- Texmaker
- MiKTeX (LaTeX)
- Python
- Python knihovny - NumPy, pandas, Matplotlib, Keras, TensorFlow, scikit-learn, shutil, os, Pillow, time, Streamlit
- Draw.io
- Github
- Heroku

Seznam příloh

Příloha 1: model.py (skript modelu neuronové sítě)

Příloha 2: app.py (skript webové aplikace)

Příloha 3: classify.py (skript klasifikátoru webové aplikace)

Všechny přílohy jsou k dispozici v online repozitáři na URL `https://github.com/XtraHamster/SKCAN`