VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Vojtěch Mikulec

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

VLIV POZADÍ A VELIKOSTI DATABÁZE NA TRÉNOVÁNÍ NEURONOVÝCH SÍTÍ PRO KLASIFIKACI OBRAZŮ

THE EFFECT OF THE BACKGROUND AND DATASET SIZE ON TRAINING OF NEURAL NETWORKS FOR IMAGE CLASSIFICATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Vojtěch Mikulec**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Martin Rajnoha**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Vojtěch Mikulec ***ID:*** 197774
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Vliv pozadí a velikosti databáze na trénování neuronových sítí pro klasifikaci obrazů

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s problematikou konvolučních neuronových sítí a jejich použití pro klasifikaci obrazu, dále s nástroji Keras a Tensorflow. Z dodané databáze obrázků odstraňte pozadí od klasifikovaných objektů a vytvořte tak kopii databáze. Natrénujte několik typových neuronových sítí na obou databázích a porovnejte jejich průběhy trénování a přesnost a otestuje přesnost sítí na reálných datech. Dále použijte váhy z modelů natrénovaných na databázi s odstraněným pozadím a dotrénujte nové modely za použití normální databáze a zhodnoťte přesnost. Následně zjistěte vliv velikosti datasetu na trénování neuronových sítí a roli pozadí při trénování.

DOPORUČENÁ LITERATURA:

[1] KRIZHEVSKY, Alex, SUTSKEVER, Ilya a HINTON, E. Geoffrey. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, s.1097–1105, 2012.

[2] CHOLLET, Francois. Keras: Deep Learning library for Theano and TensorFlow, 2015. Available: https://keras.io/

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Martin Rajnoha
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Tato bakalářská práce se zaobírá vlivem pozadí a velikosti databáze na trénování neuronových sítí pro klasifikaci obrazů. V práci jsou popsány techniky zpracování obrazů pomocí konvolučních neuronových sítí a vliv pozadí (šumu) a velikosti databáze na trénování. Práce navrhuje metody, se kterými lze dosáhnout rychlejšího a přesnějšího procesu trénování konvolučních neuronových sítí. Pro experimentování je vybrána binární klasifikace datové množiny označených tváří z různého prostředí, jejíž pozadí je pro každý experiment modifikováno nahrazením barvou nebo ořezáním. Velikost datové množiny je pro trénování konvolučních neuronových sítí klíčová, v této práci je experimentováno s velikostí trénovací množiny, což simuluje reálný problém s nedostatkem dat při trénování konvolučních neuronových sítí pro klasifikaci obrazů.

KLÍČOVÁ SLOVA

Data, klasifikace obrazu, konvoluční neuronová síť, LFW, pozadí, přenosové učení, šum.

ABSTRACT

This bachelor thesis deals with the impact of background and database size on training of neural networks for image classification. The work describes techniques of image processing using convolutional neural networks and the influence of background (noise) and database size on training. The work proposes methods which can be used to achieve faster and more accurate training process of convolutional neural networks. A binary classification of Labeled Faces in the Wild dataset is selected where the background is modified with color change or cropping for each experiment. The size of dataset is crucial for training convolutional neural networks, there are experiments with the size of training set in this work, which simulate a real problem with the lack of data when training convolutional neural networks for image classification.

KEYWORDS

Data, image classification, convolutional neural network, LFW, background, transfer learning, noise.

MIKULEC, Vojtěch. *Vliv pozadí a velikosti databáze na trénování neuronových sítí pro klasifikaci obrazů*. Brno, 2018, 61 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Martin Rajnoha

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Vliv pozadí a velikosti databáze na trénování neuronových sítí pro klasifikaci obrazů“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Martinovi Rajnohovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V dnešní době se stále častěji setkáváme se zpracováním obrazu souvisejícím s rychlým růstem technologického rozvoje. Díky rozvoji grafických procesorů je nyní možné provádět výpočty, které před několika lety nebyly zdaleka možné, což umožňuje vytvářet důmyslné aplikace pro zpracování obrazu za relativně krátkou dobu. Zpracování obrazu se díky tomu stává hodně využívaným odvětvím techniky, které se rychle rozvíjí s rostoucím počtem jeho aplikací. Jednou z těchto aplikací je klasifikace obrazu, která zpravidla využívá metod hlubokého učení, které ke správné funkci obvykle vyžadují obrovské množství dat [1]. Ve většině případů z praxe vzniká na fotografiích mnoho nechtěného šumu tvořící pozadí objektu, který je třeba klasifikovat. Tento šum může způsobovat degradaci datové množiny, čímž se může projevit neschopnost natrénovat klasifikační model.

Úkolem této práce je šum odstranit (zabarvením nebo ořezáním) tak, aby byla data použita pro trénování sítě zjednodušena. V každém provedeném testu jsou sledovány parametry jako je přesnost klasifikace neuronové sítě a doba jejího trénování v každém dosaženém postupovém bodu (epocha), které jsou ve výsledku porovnány, a ze kterých je vyvozen závěr.

V rámci praktické části jsou vytvořeny a natrénovány sítě, kde je následně porovnána přesnost na reálných datech při různých modifikacích datové množiny. Dále jsou vytvořeny modely s váhami natrénovanými na datové množině bez pozadí, které jsou následně přeneseny na originální datovou množinu. V tomto případě je práce zaměřena na průběh trénování.

Hlavním přínosem práce je otestování průběhu neuronových sítí při originálních vstupních datech a optimalizovaných datech od okolního šumu – tzn. při vytváření nových databází. Cílem této práce je zjistit a vyčíslit vliv pozadí a velikosti databáze obrazů v datové množině na trénování neuronových sítí, aby bylo umožněno snadnější, přesnější a optimalizovanější strojové určování obsahu obrazu.

Práce je strukturována následovně. První část pojednává o teorii neuronových sítí a konvolučních neuronových sítí pro klasifikaci obrazu. Druhá část se zabývá samotným experimentem a popisuje nástroje, kterých je využíváno, aby mohl proběhnout proces trénování a testování neuronových sítí. Poslední část práce tvoří výsledky testování a jejich zhodnocení. Nakonec jsou výsledky a poznatky získané v práci shrnuty v závěru.

1 Teoretický úvod

Klasifikace obrazu je jednou z často využívaných aplikací v oblasti zpracovávání obrazu. Pro její realizace se obvykle používají metody hlubokého učení, konkrétně použití konvolučních neuronových sítí, využívající principy strojového učení s učitelem [2]. Z tohoto důvodu je velmi důležité mít k dispozici data (obrazy) obsahující objekt, který je třeba klasifikovat. Při kolekci je téměř nemožné získat data obsahující pouze daný objekt, obrazy obvykle obsahují více objektů, proměnné pozadí nebo jiný šum, který tvoří zbytečnou zátěž při procesu trénování neuronové sítě. V této práci je za pozadí pokládáno vše kolem daného objektu. Tato práce se z části zabývá vlivem pozadí na rozpoznávání obrazu, který je ořezán tak, aby cokoliv kromě vybraného objektu bylo vymazáno nebo nahrazeno barvou pouze jednoho odstínu. K tomuto kroku bylo přistoupeno z důvodu poměrně velkého objemu dat v datových množinách, který je pro správné natrénování neuronových sítí nutností a právě odstraněné pozadí může pomoci neuronové síti k optimalizovanějšímu trénování.

1.1 Současná řešení

V práci [3] se autoři zabývají vlivem pozadí na přesnost modelu, kde je vyzkoušeno celkem 5 modifikací pozadí. Práce se zabývá zvyšováním a snižováním dat kolem určeného objektu pomocí maskování. Popisuje techniky, jakými je dosaženo odstranění pozadí od objektů, kde se testuje celkem 5 způsobů oddělení pozadí od popředí fotografie na datové množině o velikosti 500 obrazů. Nepopisuje ale neuronovou síť, která byla použita. Typy upravených fotografií od pozadí jsou rozděleny do 5 skupin:

Odstranění popředí, tj. odstranění klasifikovaného objektu od zbytku obrazu, dopočítání vzniklé mezery po objektu algoritmem *Patch-Match* [4] a natrénování neuronové sítě na pozadí s tím, že neuronová síť se naučí rozpoznávat, které pozadí patří ke kterému objektu z popředí, např. lodě jsou obvykle ve vodě, letadla jsou obvykle ve vzduchu atd. Je předpokládáno, že tato metoda nebude dosahovat velké přesnosti z důvodu vysoké variability prostředí a neuronová síť se naučí spíše hádat, než s větší jistotou určovat, který objekt patří do kterého prostředí. Výsledkem je přesnost 31,3 %.

Druhý způsob, který je otestován, se skládá z původní databáze s tím, že na obrazech byl ponechán objekt v popředí a pozadí bylo zamaskováno šedou barvou. Tento test je ovšem méně přesnější než první provedený pouze s pozadím. Dosahuje přesnosti 28,7 %.

Střední hodnota barev pozadí s ponechaným objektem je třetí testovaná metoda. Střední hodnotou pozadí se myslí upravení pozadí tak, aby se původní pozadí ob-

rázků spojilo do jedné průměrné barvy, například letadlo s pozadím oblohy, která je modrá a nachází se na ní bílé mraky. Barevnost oblohy je smíchána do jednoho kanálu světle modré barvy, který tvoří průměr barevných kanálů pozadí. Tato metoda dosahovala přesnosti 36,7 %.

Předposlední, čtvrtá testovaná metoda je vytvořena kombinací objektů z popředí a zároveň kombinací různých pozadí, ale ze stejné kategorie. To znamená, že pokud na obrázku je v původní datové množině v popředí pes a v pozadí šedá kamenná zem, v upravené verzi se nachází v popředí stejný pes, ale podlaha z jiného materiálu a mírně odlišného odstínu původní barvy. Při vybraných kombinacích navíc vzroste velikost databáze z 500 na 25000 vzorků a oproti trénování neuronových sítí na původních datech se zvyšuje přesnost o 15 %.

Poslední metoda transformace pozadí zahrnuje jen lehkou obměnu předchozí metody. Rozdílem je, že pozadí nepatří do stejné kategorie jako klasifikované objekty. Například letadlo s pozadím oblohy je odstraněno a místo něj je jako hlavní objekt pes z druhého obrázku. Počet kombinací tak vzroste na 250 000 obrázků a přesnost dosahuje 48,5 %, což je nepatrně lepší než trénování na neupravené datové množině, kde přesnost dosahuje 47,4 %.

1.2 Navyšování dat

Pokud je zapotřebí natrénovat síť, která má mít co nejpřesnější výsledky, je nutné mít k dispozici vhodná data. Ne vždy je to možné, a tak je v tomto případě nutné vymyslet způsob, kterým bude databáze rozšířena. Souhrn těchto metod se nazývá navyšování dat [5] a je hojně využíván při nedostatečné velikosti trénovací množiny. Každá provedená změna v databázi se dá považovat za rozšíření datové množiny, kdy je možné zhruba určit, jaká změna obrazu bude mít jaký následek pro výsledky trénování neuronové sítě. Pokud je vytvořena kopie obrazu od originálního a v této kopii je provedena pouze nepatrná změna, která je nabídnuta neuronové síti na zpracování, pravděpodobně nebude mít tato malá změna na trénování zásadní vliv a na validačních a testovacích datech se přesnost téměř nezmění. Rozšiřování dat může být provedeno mnoha způsoby, níže jsou uvedeny základní a nejpoužívanější z nich.

1. **Základní transformace**
   Je-li s kopiemi originálního obrazu provedena rozsáhlejší transformace, jako je například rotace, zrcadlení, zkosení, přiblížení nebo oddálení, bude mít neuronová síť k dispozici více odlišných vstupů, což bude mít vliv na výsledný natrénovaný model a výsledky na testovacích datech budou přesnější. Toto neplatí pouze o obrazové transformaci, ale také o obrazové výplni. Datovou množinu je možné rozšířit i přidáním kontrastu, mícháním barevných kanálů

nebo přidáním či ubráním ostrosti obrazu. Tyto metody jsou relativně jednoduché a při nastavování do vstupu neuronové sítě se ukázaly jako spolehlivé a jednoduše implementovatelné. Své popularity nabyly také díky tomu, že jsou dostupné pro většinu knihoven hlubokého učení [5].

2. **Generativní konfliktní sítě**

   V této metodě se používají dvě sítě, které mají za úkol spolu kolidovat [5]. První z těchto sítí se nazývá generativní, do které je vkládán obsah. Tato síť je zdrojová a na jejím výstupu je očekáván vygenerovaný realistický obraz zdroje. Tento obraz je pak využit ke vstupu do nové, navazující sítě nazvané diskriminativní a zde má za úkol zmást tuto síť, která je lépe trénovaná na rozpoznání reálných a falešných dat. Tato metoda může mít na svém výstupu příjemné výsledky, ale má pár problémů, které stojí za zmínku. Model generativní sítě může totiž generovat kvalitní obrazy, ale ty mohou mít naopak špatné maximální pravděpodobnosti. Modely, které disponují vysokou maximální pravděpodobností mohou mít zase špatné obrazy. Další z problémů zahrnují špatné korelace s realitou, tj. generování obrazů zvířat s nesprávným počtem končetin nebo generování obrazů objektů, které jsou příliš zarovnané k osám v trojrozměrném prostoru.

3. **Extrakce textur**

   Cílem této metody je z texturovaného obrazu odstranění textury tak, aby v obraze byl zachován pouze jeho obsah, tzn. aby byly stále viditelné okraje objektů a jejich velikost, kontury a stíny. Této metody se dá docílit například způsobem obrazového skládání, kdy se z malých kusů fotografií sestaví jeden úplně nový. Lze jí také docílit způsobem, který navrhl A. Hertzmann zvaným *„image analogies“* [6] (obrazové podobnosti). Tento způsob funguje na základě převzorkování, kdy je textura z jednoho obrazu převedena do druhého [5].

4. **Kombinace popředí a pozadí fotografie**

   V případě této metody je nutná extrakce všech objektů z popředí nebo samotná extrakce pozadí. Následným spojením popředí a pozadí, které „o sobě neví“ vzniká vysoký počet možných kombinací, čímž se zvyšuje účinnost metody. Bylo navíc otestováno, že tato metoda dosahuje přijatelných výsledků přesnosti [3].

5. **Další způsoby**

   Kromě výše popsaných metod se dá jakákoliv databáze rozšířit i jinými způsoby [5]. Jedním z nich je například metoda náhodného mazání, která je rychlá a poměrně snadná na implementaci do konvoluční neuronové sítě, přitom nám poskytuje relativně dobré výsledky. V této metodě se využívá přidání obdélníku šumu do původní fotografie, čímž se změní hodnoty vybraných pixelů v obdélníku a natrénovaný model neuronové sítě je poté masivnější a více

odolný proti šumu, který se v datové množině může objevit.

Navyšování dat zahrnuje v mnoha případech velmi užitečné metody pokud je k dispozici nedostačující databáze, ovšem nelze jich vždy využít. Typický příklad, kdy se nevyužívá navyšování dat je rozpoznávání tváří, kdy je nežádoucí mít v databázi různě zmutované tváře, které vznikly právě po aplikování některé z metod.

1.3 Neuronové sítě

S neuronovými sítěmi se v dnešní době setkáváme téměř všude. Každý z nás, kdo vlastní chytrý mobilní telefon si nosí neuronové sítě implementované v jeho výpočetních jednotkách v kapsách, aniž bychom si to uvědomovali. Pokaždé, když při otevřeném fotoaparátu chytrý telefon detekuje tvář nebo například extrahuje text vyfocený na fotografii do upravitelné podoby, s velkou pravděpodobností byl obraz klasifikován pomocí neuronových sítí [7]. Název „neuronové sítě“ je inspirován fungováním neuronových struktur v mozku živých organismů.

Část lidského mozku je stále neprobádána, a tak je možné se pouze s určitou přesností domnívat, jak funguje předávání informací mezi neurony. Každý neuron přijímá elektrické impulzy skrz krátké výběžky zvané dendrity, které jsou napojeny na další neurony v okolí, přičemž každý neuron je schopen přijmout do spojení 1000 – 5000 dalších neuronů v jeho blízkosti [8]. Po přijetí signálu pak tyto neurony posílají signál do svých vlastních dlouhých výběžků zvané axony, které dále tvoří rozsáhlou strukturu větví. Na konci každé z těchto větví se nachází synapse, což je označení pro spojení dvou neuronů pro výměnu vzruchů. Tato synapse předá vzruch dalšímu navazujícímu neuronu a celý proces vyústí v učení. Učení, které předchází pamatování si je pak opakování stimulace zakončení axonů, které se stávají více nebo méně citlivé na jednotlivé podněty [9].

Neuronové sítě v kontextu strojového učení fungují na podobném principu. Využít neuronové sítě můžeme prakticky kdykoliv, je-li třeba nejen zautomatizovat jakoukoliv činnost stroje, ale navíc stroji dát schopnost se učit. Jednotlivé neurony pro strojové učení disponují dvěma možnostmi operací [9]:

1. Učení – Neuron se trénuje na rozpoznání určitých příznaků. Společně s ostatními neurony vytváří model sítě, který se skládá z vah jednotlivých cest mezi neurony.
2. Trénování – Využití natrénovaných a „zapamatovaných“ vah k předání informace dále na výstup neuronu [9].

1.3.1 Proces strojového učení

V porovnání s neurony v živých organismech se strojové liší v jedné důležité schopnosti – „živé“ biologické struktury snadněji a kreativněji řeší komplexní problémy, naopak strojové struktury tyto schopnosti nemají [10]. Při trénování neuronových sítí rozlišujeme dva základní typy trénování:

1. Učení s učitelem *(supervised learning)*
2. Učení bez učitele *(unsupervised learning)*

1.3.2 Optimalizační algoritmy

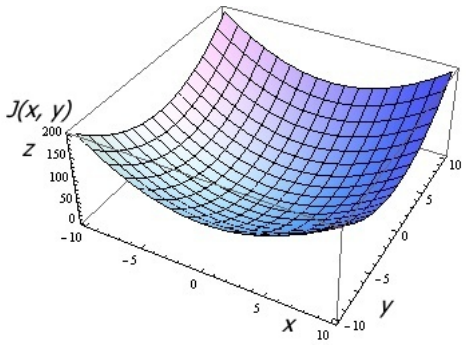
Po uvedení způsobů učení neuronových sítí jako celek je třeba zmínit způsob, jakým se učí a optimalizují samotné perceptrony (neurony). Při učení a optimalizaci se využívá iteračních metod pro optimalizaci a jejich obměn. Pojem gradient zastupuje zobecnění derivace o více proměnných – vektorovou funkci, jeho opakem je derivát, který zastupuje skalární veličinu. Vektory gradientu směřují podle nejvyšší frekvence nárůstu funkce. Dá se říci, že gradient znamená „směr růstu“. V kartézské soustavě souřadnic je gradient popsán vztahem

$$\nabla f = \frac{\partial f}{\partial x}\mathbf{i} + \frac{\partial f}{\partial y}\mathbf{j} + \frac{\partial f}{\partial z}\mathbf{k}, \tag{1.1}$$

kde $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ zastupují jednotkové vektory ve směru os x, y, z [12].

Níže je uveden popis optimalizace pomocí gradientního sestupování, konkrétně náhodného sestupování, které je využito pro neuronové sítě v experimentální části práce.

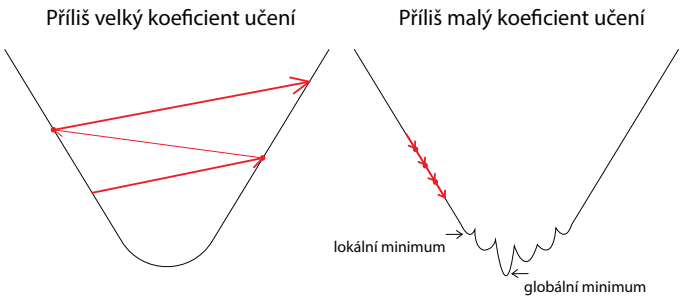
1. **Gradientní sestupování**
   Gradientní sestupování tvoří konvexní funkci, která minimalizuje dodanou funkci změnou jejích parametrů na požadovanou hodnotu. Vizualizace této funkce je na obrázku 1.1



Obr. 1.1: Vizualizace funkce postupného sestupování [11].

Tato metoda vybírá vzorky z datové množiny po skupinách. Je-li třeba natrénovat neuronovou síť s využitím metody gradientního sestupování a minimalizovat funkci $J(x, y)$ dodanou do sítě tak, aby dosáhla svého minima a tím dosáhnout optimalizace, je třeba začít experimentálně. Podle obrázku 1.1 nastavit parametry osy x a y náhodně – od tohoto parametru začíná metoda počítat postupně po krocích, kde je funkce nejstrmější až do bodu, kde je funkce na svém minimu.

2. **Stochastické gradientní sestupování (*SGD*)**
   Náhodné gradientní sestupování je iterativní metoda využívající stochastické (náhodné) aproximace gradientního sestupování. Oproti tomuto sestupování vybírá své vzorky z datové množiny náhodně, nikoliv po skupinách nebo v předem daném pořadí. V případě náhodného sestupu je z databáze vybrán pouze jeden vzorek pro vypočítání následujícího kroku s tím, že každý další vzorek je vybrán také náhodně. Každý krok je přitom definován hodnotou zvanou koeficient učení (*learning rate*), jež zastupuje rychlost změny právě optimalizovaného gradientu 1.2. U tohoto koeficientu je důležité vždy nastavit optimální hodnotu – pokud je příliš velká, bude pouze přeskakovat v mezích konvexní funkce tam a zpět. V případě nastavení příliš malé hodnoty mohou nastat dva případy – bude sice dosaženo výsledku, ale za nevýhodných časových podmínek nebo může trénování uvíznout v lokálním minimu funkce. V tomto případě je možné použít funkci zvanou `momentum` [13][14].



Obr. 1.2: Případy nevyhovujícího nastavení koeficientu učení.

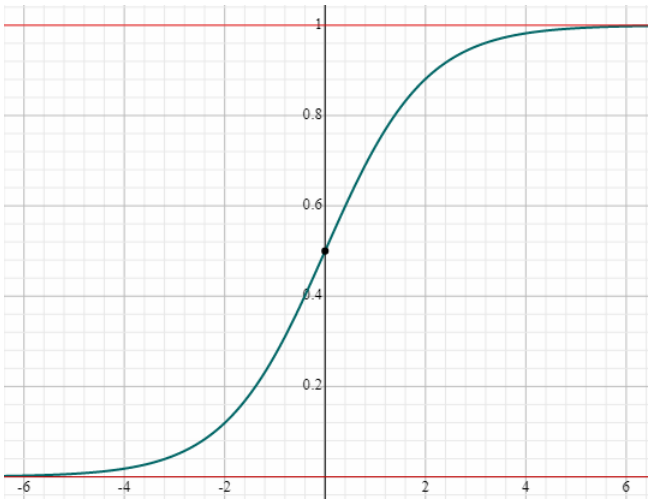
Při učení může ovšem dojít ke dvěma stavům, které znehodnocují výsledný model pro testování. Prvním z nich je přetrénování (*overfitting*) sítě – přetrénování nastává v okamžiku, kdy je model příliš komplexní a disponuje příliš mnoha parametry vzhledem k použité datové množině. Ve výsledku pak může dojít k dosahování vysoké přesnosti na trénovacích datech, ale při nasazení na testovací data, která síť nikdy neviděla vykazuje spíše opačné výsledky přesnosti [15]. Druhý problém vzniká při opačné situaci přetrénování, a tím je nedoučení (*underfitting*). Tento problém vzniká naopak při nedostatečné kom-

plexnosti modelu, kdy model není schopen zachytit souvislosti mezi příznaky v databázi a použít je na reálných datech. V mnoha případech je výsledný model problematický a neschopný klasifikovat data, na která byl natrénován [16].

1.3.3 Aktivační funkce

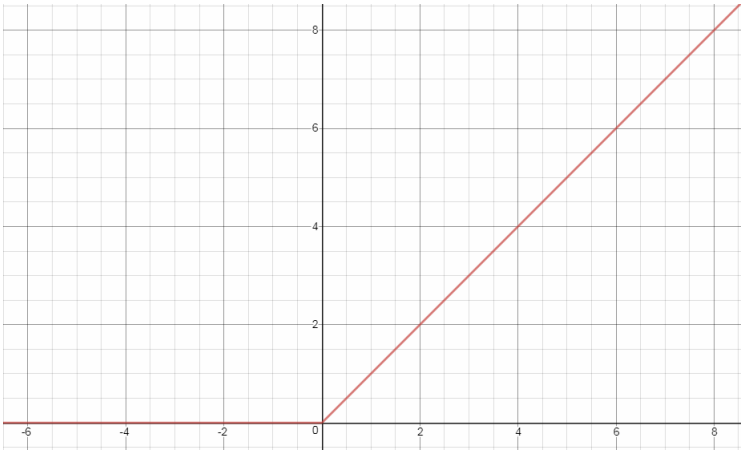
Pro správnou funkčnost by měla každá neuronová síť využívat aktivační funkce, které se řídí pravidly v angličtině označovanými jako *firing rules*. Úkolem této funkce je zmodulování výstupního signálu tak, aby mohl být zpracován v další vrstvě neuronové sítě, čímž se zajistí postup všech příznaků [9]. Při stavění sítě by vždy měla existovat snaha o konvergenci jejích výpočtů. V případě, že síť nezačne konvergovat, je třeba vrstvy upravit – většinou je testováno experimentálně podle datové množiny. Níže jsou uvedeny obecně nejpoužívanější funkce, se kterými bylo experimentováno v praktické části této práce.

1. **Sigmoid**
   Funkce Sigmoid je popsána následujícím vztahem
   $$\sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{1.2}$$
   a jejím úkolem je převést reálná čísla do rozmezí intervalu $\langle 0, 1 \rangle$. V minulosti byla často používaná z důvodu jednoduchého určení aktivační frekvence neuronu *(firing rate)* a krátké doby stavění modelů. Graf funkce je na obrázku 1.3. Její popularita při používání ve skrytých vrstvách ale klesla z důvodu přecházení funkce do saturace – funkce způsobuje ztrátu obsahu dat při procesu zpětné propagace. Při zderivování funkce sigmoidy dostaneme funkci
   $$\sigma(x) = \frac{\mathrm{e}^{-x}}{(1 + \mathrm{e}^{-x})^2}, \tag{1.3}$$
   ze které je možné si po dosazení určit, jakých maximálních hodnot bude dosahovat. Při dosazení 0 je výsledkem $0,25$, každá chyba ve vrstvách neuronové sítě tak bude $4\times$ více koncentrovaná. To znamená, že čím hlouběji bude funkce zasahovat do sítě, tím více dat bude ztraceno. Na druhou stranu, funkce Sigmoid je vhodná pro použití při binární klasifikaci v poslední výstupní vrstvě [17] [18].
2. **ReLU** *(Rectified Linear Unit)*
   Aktivační funkce ReLU je nejvíce používanou funkcí skrytých vrstev sítě a svým mechanismem je nejvíce podobná neuronům živých struktur. Pokud je vstup do funkce menší než 0, na výstupu je poskytnuta 0. Pokud je na vstupu číslo větší než 0, výstup je roven vstupu. Funkce se dá popsat vztahem
   $$f(x) = max(x, 0). \tag{1.4}$$



Obr. 1.3: Graf funkce sigmoid.

a její zobrazení je na obrázku grafu 1.4. Funkce ReLU je také oblíbená pro svou jednoduchost. Narozdíl od sigmoidy je výsledkem při derivaci kladného vstupu vždy 1, a tak při zpětné propagaci nevzniká zvyšování koncentrace chyb v síti. ReLU ale také není úplně bezproblémová funkce. Při příliš velkém gradientu, který projde funkcí ReLU se může stát, že ReLU jednotky se budou aktualizovat tak, že se neuron nikdy neaktivuje a procházející gradient bude ve zbytku sítě považován za nulový. Tento případ může nastat pokud je koeficient učení nastaven na příliš vysokou hodnotu [19].



Obr. 1.4: Graf funkce ReLU.

3. **Softmax**

   Funkce Softmax je odvozená od funkce Sigmoid. Jejich společný rys zahrnuje fakt, že obě funkce převádějí výstupy každé jednotky do rozmezí $\langle 0, 1 \rangle$. Softmax navíc provádí dělení každého výstupu tak, aby součet všech výstupů

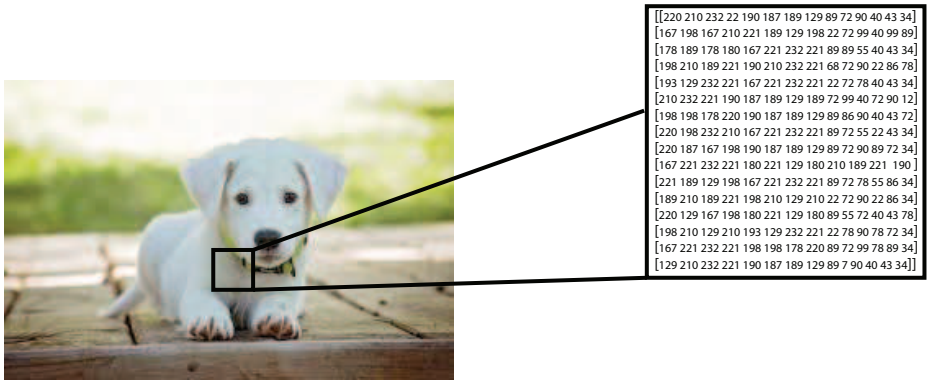
byl roven 1. Jejím výstupem je pak ekvivalent distribuční funkce. Matematicky je funkce softmax popsána vztahem

$$\sigma(z)_j = \frac{\mathrm{e}^{z_j}}{\sum_{k=1}^{K} \mathrm{e}^{z_k}}. \tag{1.5}$$

Funkce Softmax nachází hojné využití v rozpoznávání obrazu, jelikož může být použita pro velké množství tříd v datové množině. Díky jejímu charakteru se používá pouze u výstupních vrstev klasifikačních modelů [18].

1.4 Konvoluční neuronové sítě

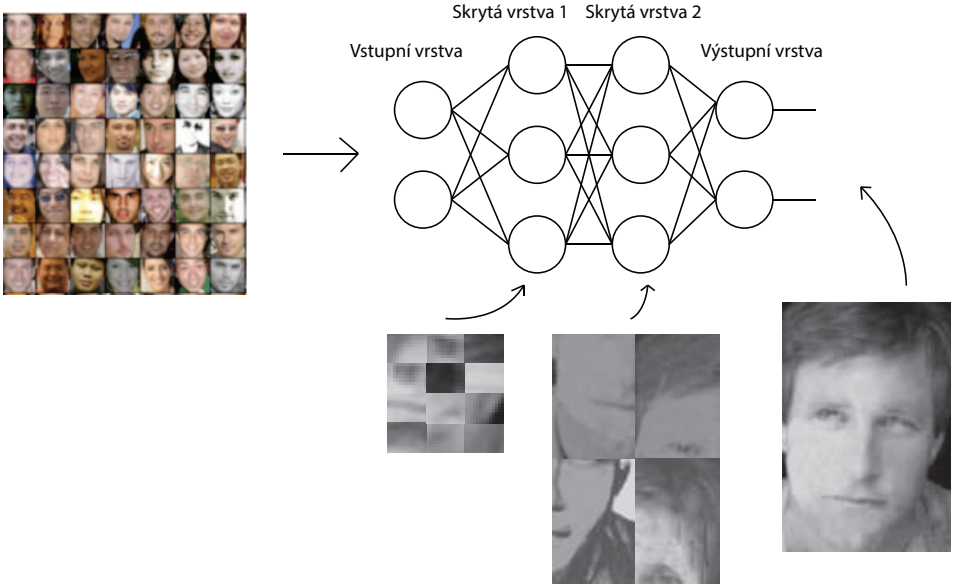
Klasifikace obrazu pomocí konvolučních neuronových sítí se stává čím dál běžněji využívanou metodou. Konvoluční neuronové sítě tvoří podmnožinu hlubokého učení, které jako celek využívá větší počet skrytých vrstev a je komplikovanější. Konvoluční postup klasifikace tvoří právě jednu z těchto mnoha vrstev hlubokého učení.
Při klasifikování obrazu je nutné poskytnout neuronové síti data (*samples*), kterým mají být na výstupu přiřazeny značky (*labels*). Obraz je pro počítač pochopitelný pouze jako matice vektorových hodnot (tenzor) s jedním rozměrem navíc, a tím je barva pixelů. Pokud je k dispozici obraz o velikosti 250×250 pixelů a obraz je navíc barevný, tzn. každý jeho pixel disponuje hodnotou barevného kanálu červené, zelené a modré barvy (RGB), tak pro počítač je tento obraz viditelný jako $250 \times 250 \times 3 = 187500$ hodnot. Každý pixel může nabývat kombinací hodnot od 0 do 255, kde 0 odpovídá černé a 255 bílé barvě. Právě těchto 187500 hodnot musí neuronová síť přečíst, vyhodnotit a obraz označit značkou [2]. Příklad takto rozloženého obrazu je na obrázku 1.5. Pro maximální účinnost by pak natrénovaný model měl být přesný tak, aby byl schopen klasifikovat i transformované obrazy a obrazy, které nikdy předtím neviděl, jak bylo vysvětleno v části navyšování dat 1.2 [20].



Obr. 1.5: Tenzor využitelný pro neuronovou síť [20].

1.4.1 Příznaková mapa

Na samotném vstupu do neuronové sítě se nachází obraz, který je třeba vyhodnotit vrstvami. Každá z nich extrahuje z obrazu příznaky tak, aby následující vrstva mohla s příznaky pracovat a povýšit je na další úroveň jak je znázorněno na obrázku 1.6. Vznikají tak příznakové mapy, jež jsou shromážděním výstupů neuronů po jeho aktivaci [21].



Obr. 1.6: Postup příznaků neuronovou sítí [33].

Mapa vzniká shromážděním těchto příznaků, které jsou v první vrstvě neuronové sítě analyzovány tzv. gabor filtry [22]. Gabor filtry jsou lineární filtry pro analýzu textur analyzující přítomnost obsahové frekvence a směru v obrazu založené na sinusových vlnových impulzech. V odvětví klasifikace obrazu a extrakci příznaků je možné je ve 2 rozměrném prostoru popsat následujícími vztahy

$$G_c[\mathrm{i},\mathrm{j}] = B\mathrm{e}^{-\frac{(\mathrm{i}^2+\mathrm{j}^2)}{2\sigma^2}}\cos(2\pi f(\mathrm{i}\cos\theta + \mathrm{j}\sin\theta)), \tag{1.6}$$

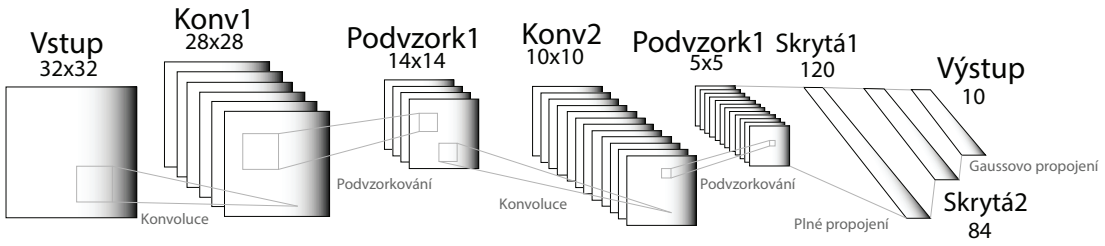
$$G_s[\mathrm{i},\mathrm{j}] = C\mathrm{e}^{-\frac{(\mathrm{i}^2+\mathrm{j}^2)}{2\sigma^2}}\cos(2\pi f(\mathrm{i}\cos\theta + \mathrm{j}\sin\theta)), \tag{1.7}$$

kde B a C jsou normalizující faktory, f zastupuje hledanou frekvenci v textuře, θ nachází texturu v určitém směru a σ umožňuje nastavení velikosti analyzované oblasti obrazu.

1.4.2 Funkce konvoluční neuronové sítě

S rozvojem strojového učení se rozvíjejí i klasifikační metody, využívající mnoho způsobů, jak obraz klasifikovat a po procesu jej zařadit do správné třídy s více

než uspokojivou přesností. Za zmínku stojí například metoda nejbližšího sousedního prvku [23] nebo rozhodovací stromy [24]. Tyto metody ale nevyužívají neuronových sítí a mohou sloužit pouze jako úvod k pochopení procesu klasifikace obrazu. Skládání neuronové sítě pro klasifikaci obrazu s vlastní datovou množinou je poměrně obtížný úkol, a proto je při stavění neuronové sítě mnohdy nutno postupovat spíše experimentálně s teoretickou přípravou. Konkrétně pro klasifikaci obrazu je možné využít předtrénované neuronové sítě nebo se jimi inspirovat při stavění vlastní.
Funkci sítí pro klasifikaci obrazu je možné si vysvětlit na příkladu již sestavené neuronové sítě LeNet-5 z roku 1998, jejíž struktura je znázorněna na obrázku 1.7 [25].



Obr. 1.7: Struktura sítě LeNet-5 [25].

Tato síť byla vytvořena za účelem rozpoznávání čísel psaných rukopisem i přístrojově. LeNet-5 má 7 vrstev, každá obsahuje parametry k natrénování (váhy). Na vstupu se nachází obraz o velikosti 32×32 pixelů. První konvoluční vrstva obsahuje 6 příznakových map o velikosti 28×28 pixelů. Druhá vrstva je podvzorkovací – *pooling* vrstva s filtrem o velikosti 2×2 pixelů uzavřena funkcí sigmoid, která ve výsledku zredukuje rozměry obrazu na 14×14 pixelů. Na konci sítě se nachází další skryté a plně propojené výstupové vrstvy (*dense*), před kterými je další blok konvoluční a podvzorkovací vrstvy.
Konvoluční vrstvy v klasifikaci obrazu fungují na základě aplikací čísel konvolučních filtrů na obraz, kde je na každou oblast obrazu aplikován filtr pro vykonání matematických operací. Výsledkem je jedna hodnota do příznakové mapy. U konvolučních vrstev je hojně využívaná funkce ReLU pro zanesení nelinearit do výsledného modelu.
Podvzorkovací vrstvy podvzorkovávají data z konvolučních vrstev za účelem snížení procesního času a snížení náročnosti na paměť. Nejvíce je využívaná funkce `MaxPooling`, jejíž úkolem je extrakce malých částí příznakových map, ponechání maximální hodnoty a vyřazení všech ostatních hodnot.
Skryté vrstvy (*dense*) jsou plně propojené z důvodu klasifikace příznaků z předchozích konvolučních a podvzorkovacích vrstev. Každý neuron je propojen s každým neuronem z předchozí vrstvy [26].

1.4.3 Datové množiny

Pro trénování neuronových sítí je již vytvořeno mnoho databází, které mohou uživateli poskytnout dostatečnou oporu pro trénování vlastní neuronové sítě. Většina datových množin obsahuje jak označkovaná a do tříd přiřazená data, tak i originální a neupravená data pro větší uživatelskou kontrolu nad databází. Při výběru vhodné databáze je třeba brát ohled hlavně na účel, ke kterému má být použita. Tato podkapitola pojednává o obecně nejpoužívanějších databázích pro klasifikaci obrazu.

- **MNIST (Modified National Institute of Standards and Technology databáze)**
  Datová množina MNIST obsahuje obrazy číslic psaných rukopisem a je hojně využívána při trénování neuronových sítí pro klasifikaci obrazu. Jedná se o databázi vytvořenou promícháním již dostupné databáze NIST. Obsahuje 60 000 trénovacích a 10 000 testovacích vzorků, kde všechny vzorky mají jednotnou velikost 28×28 pixelů a jsou modifikované použitím techniky vyhlazování hran. Zatím nejnižší dosažená hodnota chybovosti (0,21 %) byla dosažena v roce 2013 v práci regularizace neuronové sítě pomocí odstranění vah [27].
- **ImageNet**
  Databáze ImageNet vychází ze struktury databáze WordNet, lexikální databáze pro anglický jazyk. Ve WordNetu se popisuje každý vzorek souslovím – tyto vzorky jsou pak nazývány „synsety“ a WordNet jich obsahuje více než 100 000. ImageNet pak poskytuje průměrně 100 obrazů k ilustrování synsetů, které jsou manuálně označené. Těchto obrazů využívají lidé z celého světa a trénují na nich své algoritmy s cílem označit další neviděné obrazy. Nejmenší chybovost byla dosáhnuta v roce 2014 – 6,7 % [28].
- **CIFAR-10 a CIFAR-100**
  Databáze CIFAR-10 obsahuje 60 000 barevných obrazů rozdělených do 10 tříd, v každé je 6 000 obrazů. Je rozdělen na 50 000 trénovacích a 10 000 testovacích vzorků. CIFAR-100 je pak kopií CIFARu-10 s tím rozdílem, že je rozdělen no 100 tříd, kde v každé je 600 obrazů. Trénovací množina se skládá z 500 vzorků a testovací ze 100 vzorků. Těchto 100 tříd je pak rozděleno do 20 „supertříd“, kde každý vzorek má svoje přesné označení třídy a své hrubé označení supertřídy [29].
- **Labeled Faces in the Wild**
  Pro zjišťování vlivu pozadí na trénování neuronových sítí byla zvolena volně dostupná datová množina Labeled Faces in the Wild. Tato databáze byla na-

shromážděna z různých webů, je určena pro klasifikaci tváří, obsahuje 13 578 obrazů rozdělených do 5 795 tříd (každá třída zastupuje jednoho člověka) a všechny obrazy jsou stejné velikosti 250 × 250 pixelů. Z těchto tříd obsahuje 1680 více než 2 fotky [30]. Dále jsou kromě originální verze dostupné další, které se od té originální liší zarovnáním obličejů při pohledu na celkový obraz [33]. Datová množina nabízí i doporučené rozdělení na trénovací a testovací množinu pro přesnější výsledky uživatelům, což ale v tomto případě nelze využít.

1.5 Přenosové učení

Přenosovým učením je nazýván proces, kdy je neuronová síť natrénována na jedné skupině dat a vzápětí použita k trénování a klasifikaci s jinou datovou množinou [34]. Například neuronová síť s parametry natrénovanými na rozpoznání aut může být použita na rozpoznávání nákladních automobilů. Poprvé byla tato metoda použita v roce 1993 v práci Loriena Pratta s názvem „Discriminability-based transfer between neural networks“ [35].

V této práci je využita technika přenosového učení, kdy jsou použity váhy z natrénovaných modelů na databázi s odstraněným pozadím a následně jsou na stejné neuronové síti dotrénovány nové modely za použití originální datové množiny. V tomto experimentu je přenosové učení aplikováno na dvě datové sady – originální a bez pozadí. Trénování je pozastaveno na určité přesnosti datové množiny bez pozadí, model a jeho váhy jsou uloženy a následně načteny s tím, že je zaměněna datová množina za originální. Jsou vyzkoušeny dvě modifikace této metody:

1. V případě, že přesnost při trénování na datové množině bez pozadí se již dále nezvyšuje, je sledováno, zda-li je přesnost na reálných datech nižší, stejná nebo vyšší. Důraz je kladen pouze na dosaženou validační přesnost.
2. Je určena hranice přesnosti, při které bude trénování ukončeno. V tomto případě je pozorována také přesnost, avšak s větším zaměřením na rychlost učení.

Pokud se stejná síť není schopná naučit na reálných datech, bude sledována přesnost. V případě, že se začne učit jak na datech s odstraněným pozadím, tak i na reálných datech, je sledována rychlost učení a počet epoch, kdy bude dosažena stejná přesnost, jako na reálných datech. Tento experiment by měl objasnit, zda je možné přenesením vah z databáze bez pozadí docílit rychlejšího procesu učení.

1.6 Aktivní učení

Aktivní učení referuje k různým způsobům, jak lépe interagovat mezi učícím se algoritmem a datovou množinou, na které je spuštěn. Tento algoritmus by měl být schopen rozpoznat a zaměřit se na části datové množiny, které zná nejméně. Aktivního učení je hlavně využíváno při nedostatku označených dat v datové množině, kdy u některých dat je velice časově náročné data označit. Příklady dat, se kterými je tento problém jsou například data pro rozpoznávání mluveného projevu, extrakci informací z různých dokumentů nebo pro klasifikaci a filtrování dat různého typu (např. rozdělování do tříd) [37].

Oblast aktivního učení není pro tuto práci středem zájmu, ale vzhledem k experimentování s velikostí databáze by mohl nastat případ, kterému se aktivní učení do podrobna věnuje. Při práci s omezenou datovou množinou může nastat případ, kdy konvoluční neuronová síť dosahuje horších výsledků s více trénovacími daty, než když jich má k dispozici méně. Tato situace nastává tehdy když testovací data právě nevyhovují modelu s určitými natrénovanými váhami. Samotné trénování ovlivňuje jak druh dat (v tomto případě fotografie) a rozdělení datové množiny, tak i právě pasování dat natrénovanému modelu. V praxi se může stát, že síť bude natrénovaná na 95 % a otestována na 5 % dat. Tato data zrovna budou pasovat na výsledný model a bude tak dosaženo vysoké přesnosti. V druhém případě se ale může stát, že síť bude natrénována na 90 % a otestována na 10 % dat, které zrovna nebudou pasovat natrénovanému modelu nebo naopak, i přes nižší poměr trénovacích dat dosáhne model vyšší přesnosti.

2 Experimentální část

Pro samotný experiment je třeba připravit kopii databáze tak, aby byl objekt v jeho vzorcích ořezán (odstraněno pozadí) od okolního šumu. Před tímto krokem je ale ještě třeba vybalancovat databázi tak, aby každá třída obsahovala stejný počet vzorků. Databáze LFW poskytuje obrazy s již vycentrovaným obličejem na středový pixel fotografie, čímž se práce zlehčuje. Následně jsou sledovány průběhy trénování a validace. Předpokládá se, že síť s odstraněným šumem okolo objektu bude naučena rychleji. Záleží ovšem na složení sítě a počtu trénovatelných parametrů.
Pro tento experiment je vybrána binární klasifikace obrazu. Při binární klasifikaci jsou porovnávány 2 nezávislé třídy a na jejím výstupu je vzorek přiřazen do jedné třídy. V tomto experimentu však není důležité přiřazení ani přesnost, ale průběh trénování. Pro trénování neuronových sítí a vyhodnocování na validační množině je nezbytně nutný výpočetní výkon, knihovny a nástroje, které jsou schopné tento úkol splnit.

2.1 Použití GPU

Při trénování neuronových sítí je důležitý výpočetní výkon. V počítačích existují dvě varianty, kde je možné provádět výpočty pro neuronové sítě – hlavní procesor (CPU) a grafická karta s vlastním procesorem (GPU). Za poslední desetiletí se grafická oblast techniky rozvíjela velmi rychle – díky akceleraci je možné vyobrazovat grafické objekty, které před pár lety nebylo možné zobrazit a to právě díky výpočtům grafického procesoru zpracovávající vektory a rychlému provádění maticových výpočtů. Vzhledem k tomu, že hlavní procesor je možné považovat za jakési „srdce“ počítače a funguje jako manažer výpočtů, GPU je možné si představit jako hlavní výpočetní jednotku v počítači. Procesor společně s pamětí RAM upřednostňuje rychlost vyřizování procesů i přesto, že je nezvládne všechny najednou, což je pro konvoluční neuronové sítě nevhodné. Oproti tomu grafický procesor zvládne počítat mnohonásobně více procesů najednou (v tomto případě například násobení matic), ale v porovnání s hlavním procesorem za delší dobu. Delší doba je v tomto případě vyřešena více-vláknovou architekturou grafického čipu. Poslední, ale hlavní výhodou využití GPU pro neuronové sítě je její architektura. Všechny výpočty v reálném čase jsou prováděny v registrech propojených s výpočetními jednotkami – u hlavního procesoru jsou to jádra, u grafické jednotky streamovací procesory, které jsou stěžejní pro výpočty neuronových sítí. Pro soubory registrů je ale omezena velikost paměti výrobní technologií, která se v dnešní době a blízké budoucnosti pohybuje okolo 7 nm.

V případě této práce byla použita grafická karta společnosti nVidia s typovým označením GeForce GTX 1050 upravená společností MSI, disponující grafickou pamětí 2 GB, paměťovým taktem 1531 GHz a 640 streamovacími procesory (CUDA) .

2.2 Keras

Konvoluční neuronové sítě v této práci jsou stavěné v Kerasu, což je vysokoúrovňové aplikační programovací rozhraní napsané v programovacím jazyce Python. Keras dále využívá knihoven pro hluboké učení jako jsou CNTK, Theano nebo Tensorflow. Byl vytvořen se záměrem uživatelské jednoduchosti a rychlého experimentování pro jeho uživatele.
Keras má vytvořenou a podrobně popsanou dokumentaci. Umožňuje jednoduché a rychlé vytváření prototypů díky modularitě a dostupným rozšířením. Podporuje konvoluční a rekurentní neuronové sítě a pro jejich výpočty podporuje spouštění jak na GPU, tak i na CPU [31].

2.3 TensorFlow

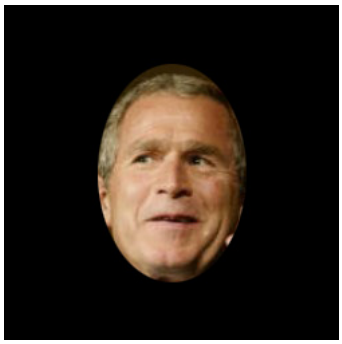
Keras pro svou správnou funkci využívá knihovnu TensorFlow. TensorFlow je otevřená knihovna pro vysoko-úrovňové výpočty strojového učení, která byla vydána společností Google v roce 2015, první stabilní verze pak v únoru roku 2017. Je používána mnoha společnostmi pro analýzu a zpracování dat (Google, Twitter, ebay, ...) a může běžet na více procesorech najednou. Její flexibilní architektura umožňuje provádět výpočty na zařízeních jako jsou servery, mobilní zařízení nebo síťové prvky [32].

2.4 Databáze pro experiment

Pro tento konkrétní úkol byla zvolena datová množina LFW a binární klasifikace. Je třeba v datové množině najít dvě osoby s největším počtem vzorků a natrénovat neuronovou síť na rozdělení obrazů do dvou tříd. V případě použité datové množiny se jedná o George W. Bushe s celkovým počtem 530 vzorků a Colina Powella s 236 obrazy. Dále byla vytvořena kopie této databáze s tím, že bylo odstraněno pozadí kolem obličeje ve vycentrované části (nahrazeno barvou) fotky a v neposlední řadě byly obě dvě třídy vybalancovány tak, aby každá měla stejný počet vzorků. U George W. Bushe muselo být odstraněno 294 snímků, aby datové množiny byly balancované. Balancování je důležité z důvodu rovnoměrného naučení neuronové sítě na každou třídu. V případě, že by databáze nebyly balancovány, síť by byla naučena na jednu

třídu více a měla by problémy s klasifikací druhé třídy. Zezačátku jsou k dispozici celkem dvě datové množiny – jedna originální s pozadím, které do neuronové sítě přináší zbytečný šum (je třeba klasifikovat osoby na základě tváře) a druhá – její kopie s pozadím nahrazeným černou barvou. V průběhu experimentu jsou dále přidávány modifikace barvy pozadí a ořezání klasifikovaného objektu. Příklad první modifikace pro experiment je na obr. 2.1.

(a) Originální fotka.



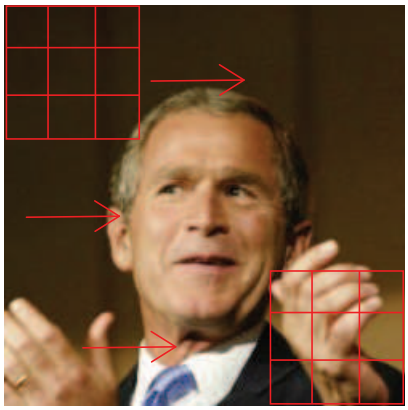
(b) Odstraněné pozadí.

Obr. 2.1: Příklady vzorků z obou datových množin [33].

Velikost datové množiny a přesnosti, které je třeba s ní dosáhnout vždy záleží na úkolu, který je třeba zrealizovat (tj. pro některé úkoly může být databáze menší, pro některé je zase třeba velké množství vzorků) [36]. Na obr. 2.2 je znázorněn přechod konvolučního jádra přes obrázek, kde je vidět, že síť se musí trénovat i na zašuměných částech obrazu (tzn. aplikace filtrů a dalších výpočtů, které v tomto případě zdržují proces). Oproti tomu síť na databázi se začerněným pozadím se učí pouze na obličeji, který je třeba klasifikovat a na jednolitém pozadí, mezi jehož pixely nachází stejné analogie. Pro ilustraci je uveden následující obrázek 2.2, který znázorňuje postup konvolučního jádra nad obrázkem, kde u obrazu s pozadím počítá se vším kromě klasifikovaného objektu, na rozdíl od začernění, kde klasifikuje pouze chtěný objekt a případné stejné analogie pozadí.

2.5 Popis natrénovaných neuronových sítí

Při trénování neuronových sítí se mnohdy využívá již předtrénovaných modelů, které ale v případě této práce nelze využít – je třeba postavit neuronovou síť od začátku. Předtrénované modely totiž obsahují váhy, které by ovlivňovaly výsledky experimentu na všech vyzkoušených databázích a nebylo by tak možné změřit vliv pozadí na trénování. Bylo natrénováno více neuronových sítí, které se od sebe lišily počtem



Obr. 2.2: Postup konvolučního jádra přes vzorek [33].

vrstev, jejich uspořádáním, počtem neuronů a s různými optimalizačními algoritmy a rychlostmi učení. Všechny průběhy ale měly společné dávkování vzorků (po jednom obrazu) a počet epoch (500) u prvního experimetu, u druhého byl počet epoch nastaven na 100. První konvergující neuronová síť byla 3.6a, následně byla přidána struktura 3.6b.

2.5.1 Reproducibilita

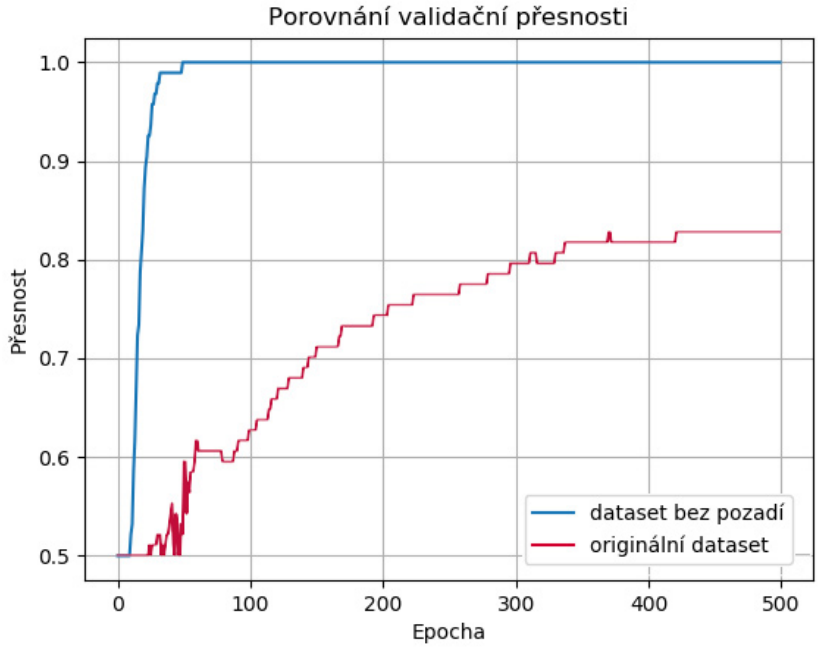
Pro zjišťování vlivu pozadí na trénování neuronových sítí je důležité, aby proces trénování a validace probíhal co nejméně náhodně. Při generování vzorků do sítě je možné použít více způsobů, zde je použit způsob dávkování vzorků pomocí generátoru implementovaného v Kerasu s názvem `ImageDataGenerator`. Je třeba zamezit náhodnému míchání pořadí vzorků jak v tomto generátoru, tak ve všech ostatních používaných knihovnách tak, aby bylo docíleno co nejmenších odchylek při opětovném spouštění procesu.
V tomto úkolu jsou použity knihovny `tensorflow`, `numpy` a `random`. U všech těchto knihoven je třeba nastavení náhodného koeficientu (`random_seed`), aby inicializace začínala vždy na stejných hodnotách, a aby průběh byl vždy v obdobném pořadí. Bohužel, při spouštění výpočtů na GPU s využitím knihovny `tensorflow` zatím není možné dosáhnout 100% nedeterministických výsledků z důvodů možné existence vlastních generátorů přímo v GPU a také více-vlákenného zpracovávání výpočtů. Při testování bylo zjištěno, že hodnoty se liší v desetinách procent. Takové hodnoty odchylky ale nejsou pro experiment ohrožující. V Kerasu je také možné více-vlákenné zpracovávání GPU vypnout a také tak bylo učiněno.

3 Výsledky

Tato kapitola se zabývá porovnáváním průběhů trénování a diskuzí dosažených přesností. Ve výsledcích je sledován parametr validační a testovací přesnosti, které nejvíce napoví o průběhu trénování neuronových sítí i přesto, že dosažení vysoké přesnosti není hlavním cílem této práce.

3.1 Modifikace pozadí

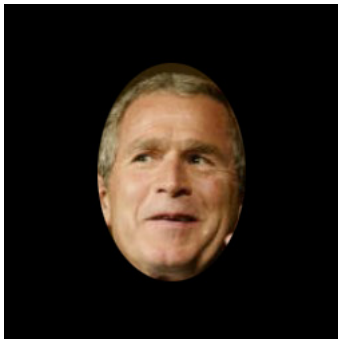
K dispozici jsou dvě přesně vybalancované datové množiny s tím rozdílem, že jedna z nich obsahuje originální fotografie a druhá obsahuje obrazy bez přebytečného šumu kolem klasifikovaného obličeje. Při experimentování bylo vyzkoušeno více struktur inspirovány architekturami již vytvořených sítí (VGG-16, AlexNet, ...). Nejméně časově náročné trénování a zároveň s nejlepšími výsledky validačních přesností měla síť 3.6a.
Je spuštěno 500 epoch, kde každá datová množina začíná konvergovat v jiné epoše. Z grafu 3.1 se potvrzuje teoretický předpoklad, že databáze bez pozadí má rychlejší průběh trénování a dřívější začátek konvergence.



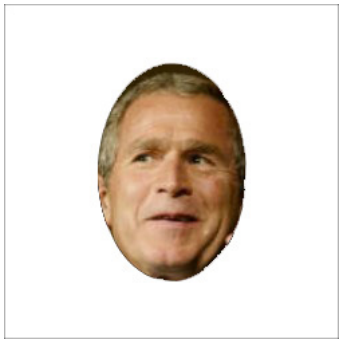
Obr. 3.1: Porovnání průběhů trénování na dvou datových množinách.

Databáze bez pozadí začala konvergovat v epoše č. 11, originální datová množina přešla z hádání na učení v epoše č. 27. Dále je z grafu 3.1 vidět o mnoho rychlejší

dosažení vysoké validační přesnosti, kde databáze bez pozadí dosahuje přesnosti 98 %, zatímco v tu samou epochu se originální datová množina po dlouhém kolísání vyšplhala na přesnost 53 %. Zatímco neuronová síť se vstupními daty bez pozadí vykazovala známky přetrénování, neuronová síť s původními daty se začala po malých krocích a bez většího pádu učit, kde v poslední epoše dosáhla validační přesnosti 83 %.
Pro další experimentování byly s datovými množinami provedeny následující akce. Byla vytvořena kopie databáze se začerněným pozadím. Toto černé pozadí bylo změněno na bílou barvu pro demonstraci vlivu hodnot okolních pixelů pozadí na trénování sítě. Všechny pixely, které měly hodnotu RGB kombinací 0 (černá) nyní mají hodnotu 255 (bílá) 3.2.



(a) Obraz s černým pozadím.



(b) Obraz s bílým pozadím.

Obr. 3.2: Příklady vzorků z modifikací databáze.

Další akce s databází má za úkol demonstrovat vliv poměru pozadí (šumu) na klasifikovaném objektu (signálu). Data byla ořezána (pozadí bylo reálně ořezáno, nikoliv pouze začerněno) tak, aby se zmenšil obsah šumu na minimum a zároveň aby bylo zachováno maximum signálu. Vzhledem k tomu, že data byla ořezána hranatě (klasifikovaný obličej nezakrýval celý obraz), byla provedena doplňková akce – začernění šumu v těchto rozích.

Vždy stejná neuronová síť má k dispozici všechny uvedené datové množiny, kde každý vzorek je klasifikován ve stejném pořadí. Všechny průběhy dohromady byly zaznamenány a zaneseny do grafu 3.4.

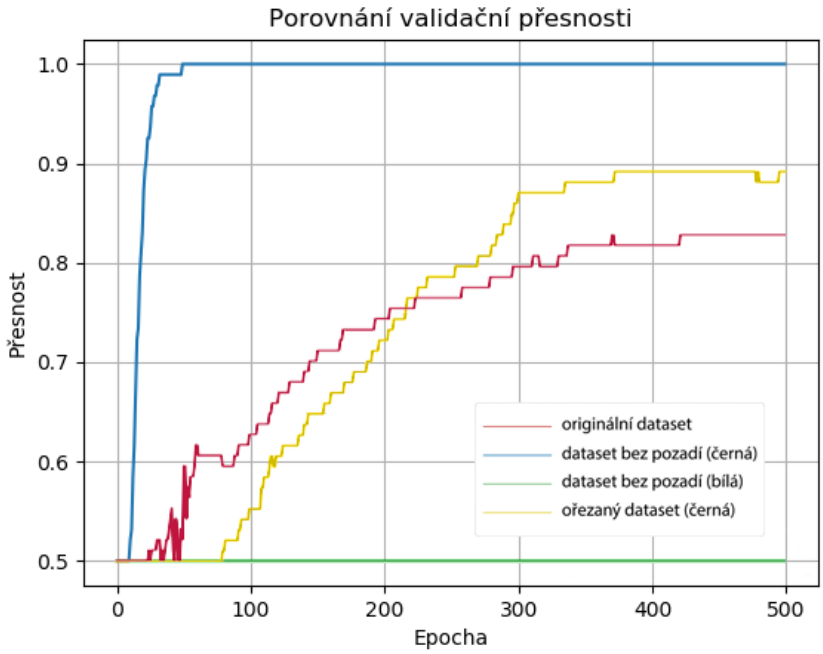
Při porovnání všech průběhů je vidět, že pozadí má vliv na průběh trénování i validační přesnost výsledků. Zatímco data s kompletně začerněným pozadím dosahovala nejlepších výsledků validační přesnosti za nejkratší čas, při nahrazení černého pozadí bílou barvou síť vůbec nezačala konvergovat. Další dva porovnatelné průběhy jsou trénování originální databáze bez jakýchkoli úprav a reálně ořezaných dat se začerněnými rohy obrazu jako je vidět na obrázku 3.3. Síť u těchto dat sice začala

(a) Originální fotka.

(b) Odstraněné pozadí.

Obr. 3.3: Příklady vzorků z druhé modifikace datových množin.

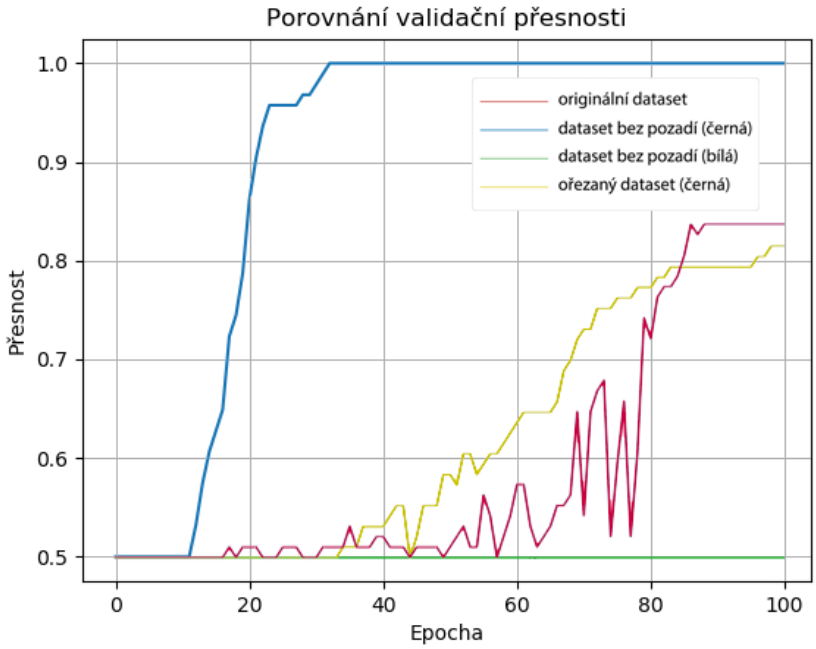
konvergovat později (v epoše č. 80) ale při konci trénování dosahovala vyšší validační přesnosti (89 %).



Obr. 3.4: Porovnání průběhů trénování na všech datových množinách při použití struktury 3.6a.

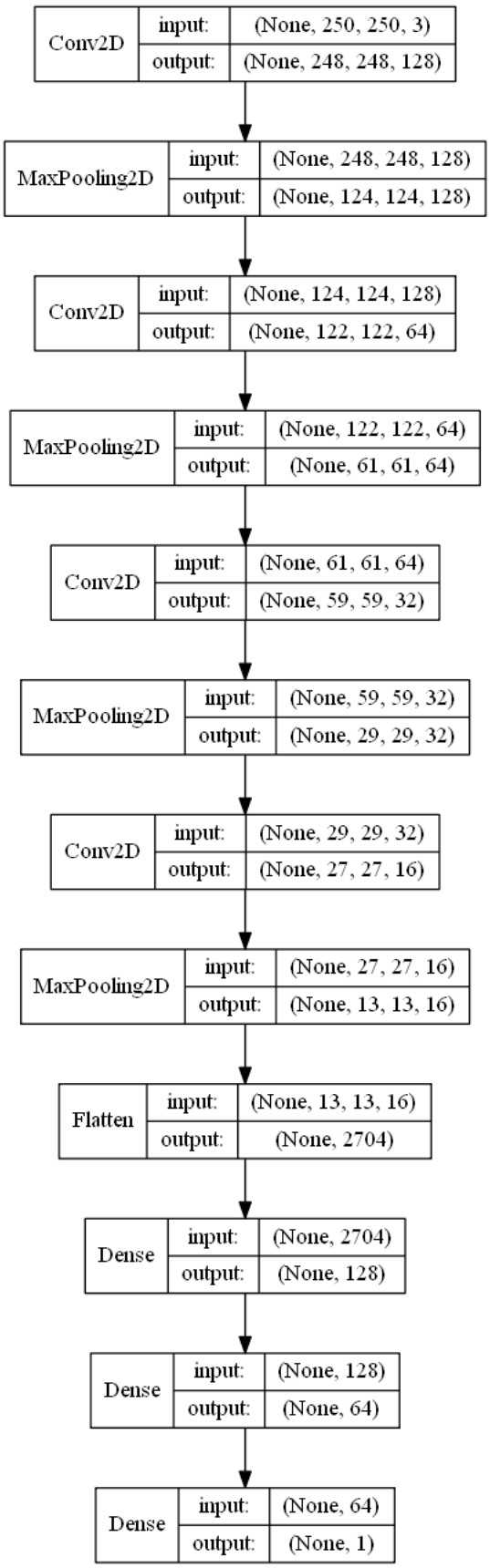
Jednou z dalších vyzkoušených struktur neuronových sítí je neuronová síť inspirována strukturou VGG-16, která se skládá z bloků konvolučních vrstev, kde každý blok je zakončený podvzorkovací vrstvou. Stejně jako u sítě LeNet-5 jsou poslední tři vrstvy plně propojené. U této struktury je spuštěno pouze 100 epoch z důvodu časové náročnosti výpočtů. Struktura sítě a její výsledky jsou jako v předešlém pří-

padě uvedeny na obrázku 3.6b a grafu 3.5.

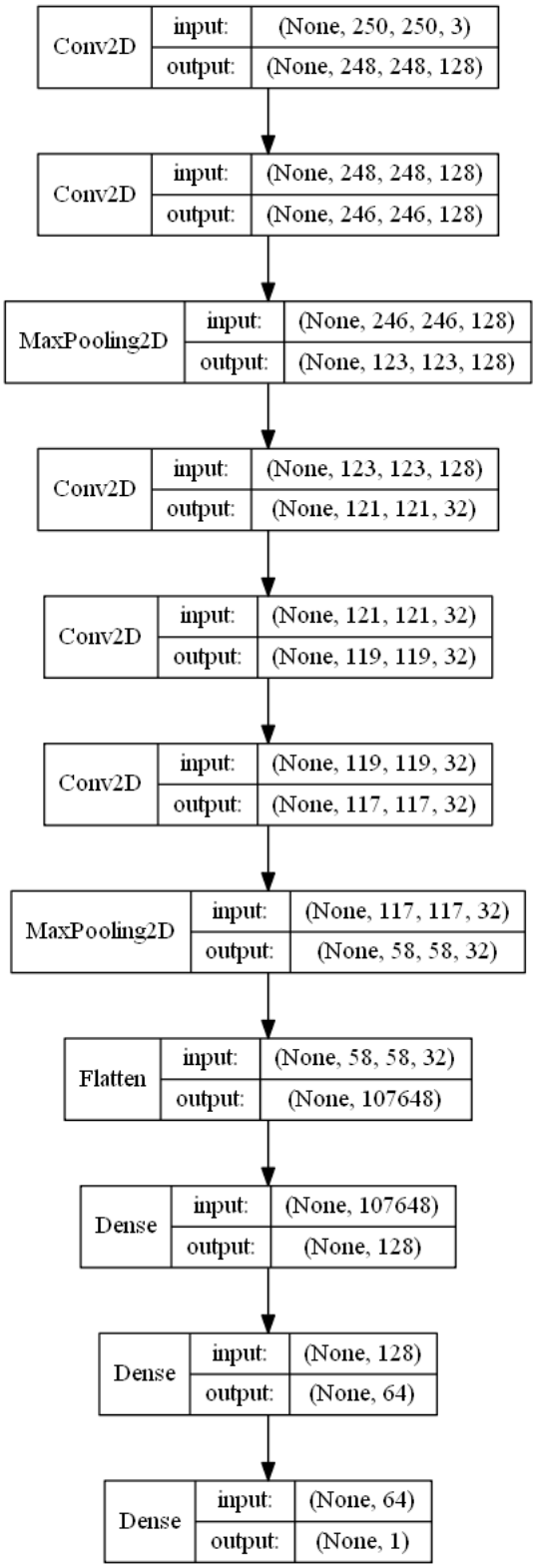


Obr. 3.5: Porovnání průběhů trénování na všech datových množinách při použití struktury 3.6b.

Z grafu 3.5 jsou patrné podobné výsledky jako u první struktury, což dokazuje, že tento trend se opakuje. Zatímco neuronová síť s kopií dat se začerněným pozadím byla naučena nejrychleji a nejpřesněji, kopie s bílým pozadím opět nezačala konvergovat. Originální data oproti ořezaným se začerněnými rohy, ale na rozdíl od prvních průběhů dosáhla v poslední epoše vyšší přesnosti, ale měla pomalejší a kolísavý proces validace, což je velmi podobné jako u prvního experimentu se strukturou sítě LeNet-5.



(a) Síť inspirována strukturou LeNet-5.



(b) Síť inspirována strukturou VGG-16.

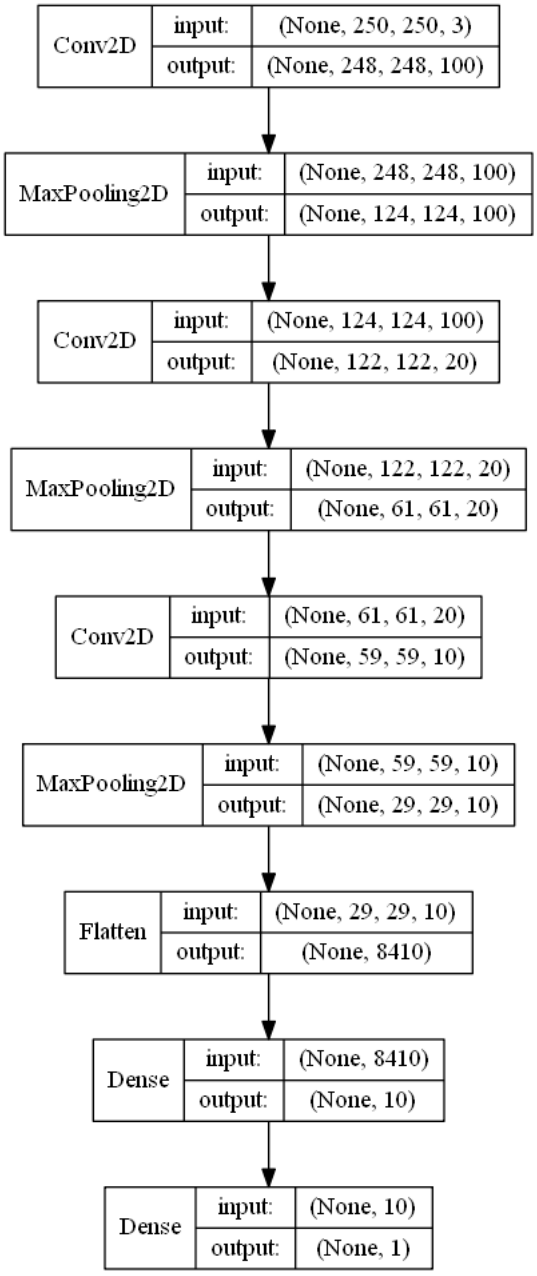
Obr. 3.6: Navržené struktury pro experiment s modifikacemi pozadí.

3.2 Testování modelu na reálných datech

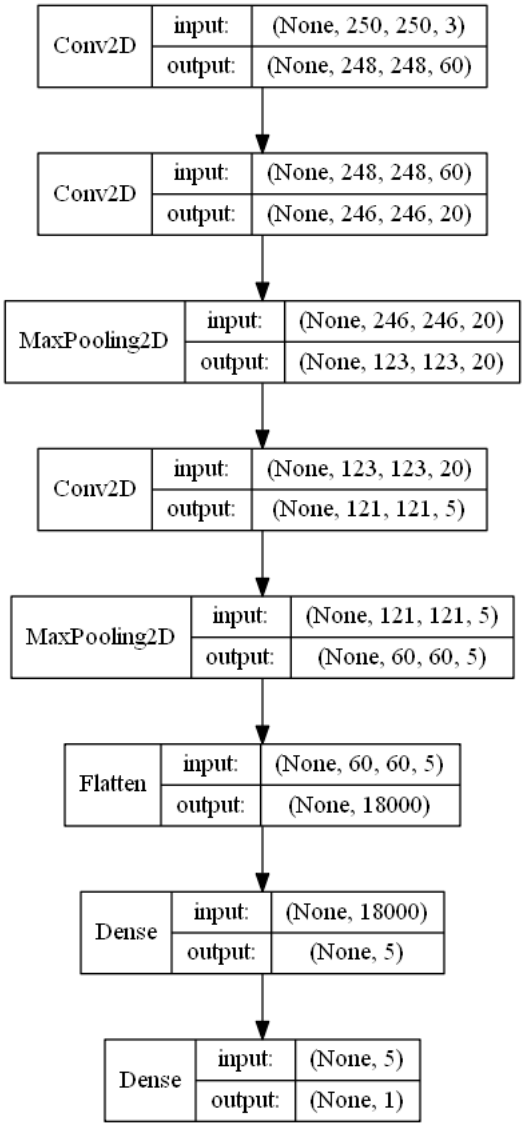
Při binární klasifikaci byly uloženy výsledné modely sítí inspirované strukturami obou sítí, aby mohly být následně otestovány na reálných datech. Pro generování testovacích dat byla použita funkce Kerasu `evaluate_generator`. Při použití původně navržených neuronových sítí bylo zjištěno přetrénování modelu, kdy s tak malou datovou množinou obsahují obě sítě příliš parametrů. Z tohoto důvodu bylo pro tento experiment navrženo několik dalších sítí, každá s jinými parametry (počet vrstev, počet perceptronů, velikost podvzorkovacího okénka, ...), kde nastává problém s odhadnutím komplexnosti sítě – pro tak malou datovou množinu bez pozadí je obtížné najít správnou konfiguraci sítě tak, aby začala konvergovat, ale zároveň nebyla přetrénovaná. Celkem bylo vyzkoušeno 5 různých neuronových sítí, kde datová množina byla rozdělena vždy stejně, a to v poměru 60:20:20 % (dvě třídy, v každé stejný počet snímků). Modely byly postupně ukládány a následně byl spuštěn test na reálných datech obou natrénovaných modelů (pro každou síť byl natrénován model na originální datové množině i na množině se začerněným pozadím okolo klasifikovaného objektu). Pro zjištění reálné přesnosti pak jsou oba modely spuštěny pouze na originálních testovacích datech. Tímto způsobem je možné zjistit, jak si oba modely vedou v konečné přesnosti a zda se model natrénovaný se začerněným pozadím je schopen naučit klasifikovat s přesností podobné originální datové množině. Níže jsou v tabulkách uvedeny struktury sítí s příslušným grafem průběhů jejich trénování (validační přesnosti) na obou datových množinách a dále je v grafu vyznačena křížkem konečná testovací přesnost dosažena na originálních datech.

U prvních dvou vyzkoušených sítí je z průběhu trénování na datové množině bez pozadí vidět, že obě sítě jsou s danými parametry datové množiny mírně přetrénované. U obou sítí 3.7 je patrné, že pro data bez pozadí by stačilo je zjednodušit, aby dosáhly vyšší testovací přesnosti. Se zanedbáním faktu přetrénování je z grafu 3.8 vidět, že testovací přesnost modelu bez šumu kolem klasifikovaného objektu je schopna soupeřit s přesností parametrů sítě s originálními daty a v nepřetrénovaných sítích by model s daty bez pozadí mohl překonat originální model.
Síť 3.7b vůbec nezačíná na originálních datech konvergovat, naopak pro data bez pozadí vykazuje známky přetrénování stejně jako v předchozím případě. Při tvorbě následujících neuronových sítí je proto více dbáno na vytvoření ideální struktury pro obě datové množiny, což by umožňovalo objektivnější porovnání průběhů a přesností na reálných datech.

Sítě 3.10a i 3.10b byly podstatně zjednodušeny tak, aby vyhovovaly podmínkám obou datových množin. Model se začerněným pozadím dosáhl přesnosti 65 %, model s originálními daty dosáhl při testování 52 % (tato přesnost není v grafu vyznačena).

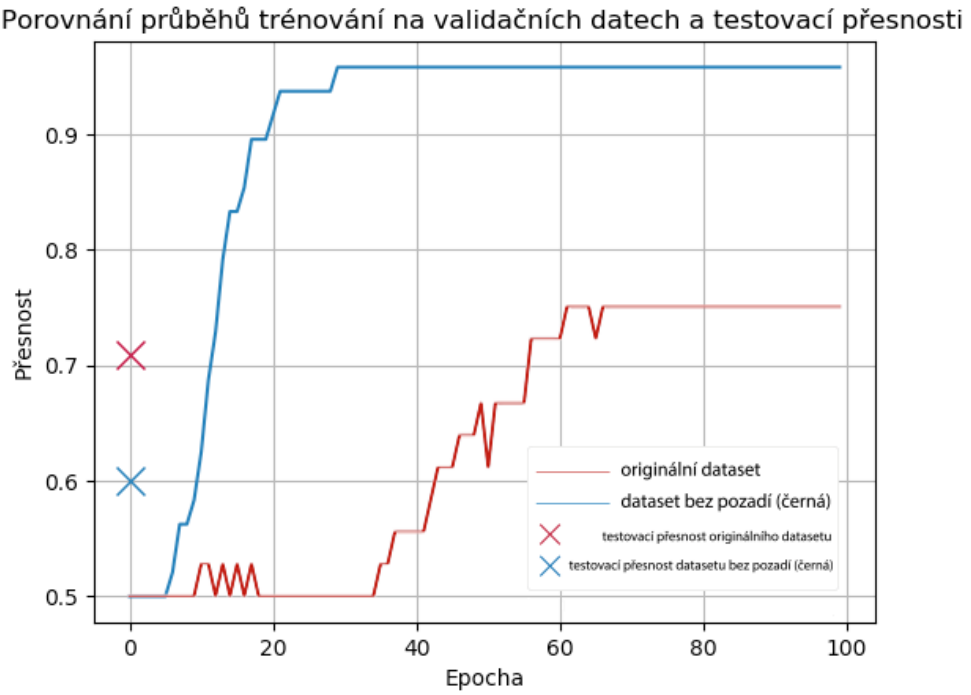


(a) Struktura první navržené sítě.

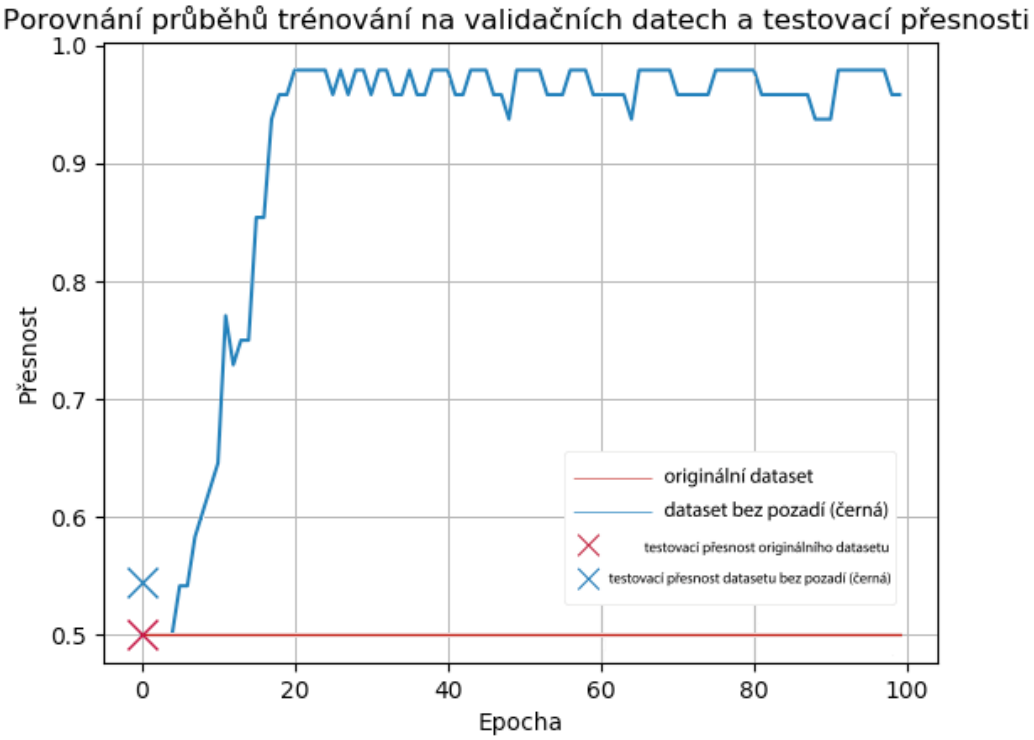


(b) Struktura druhé navržené sítě.

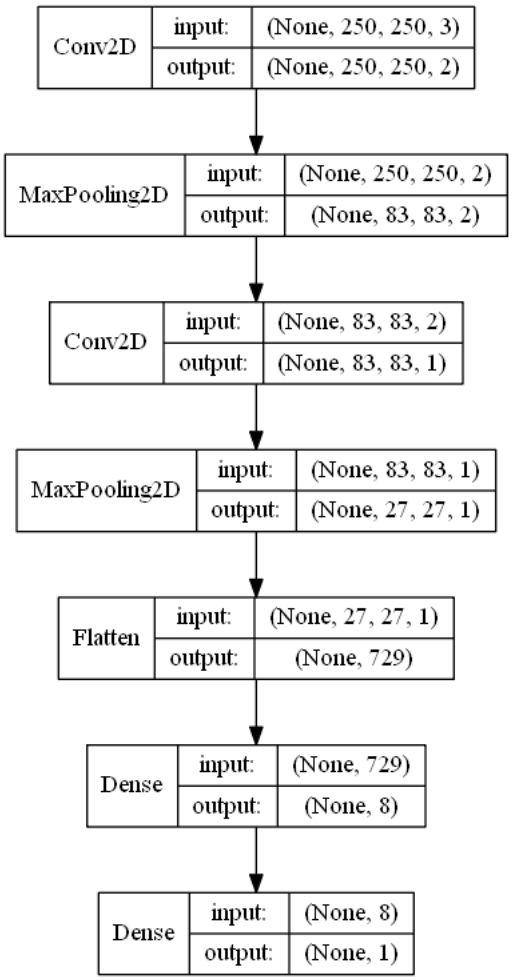
Obr. 3.7: Navržené struktury pro testování na reálných datech.



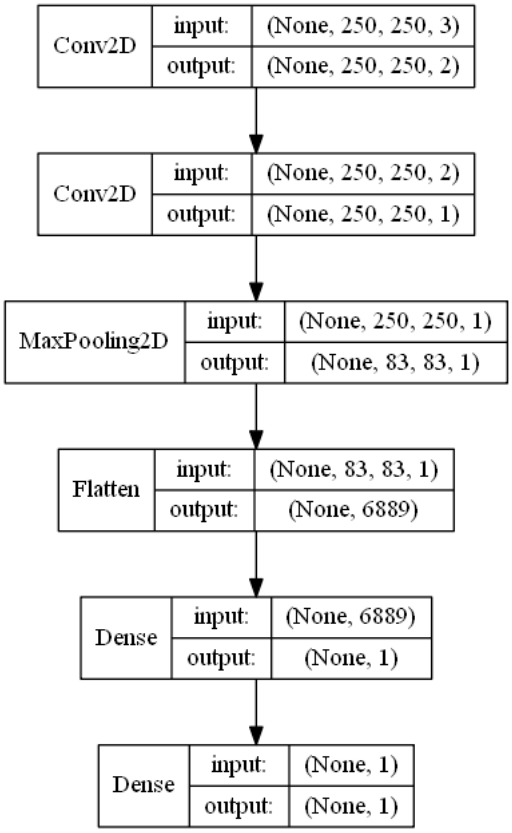
Obr. 3.8: Testovací přesnost a průběhy trénování sítě 3.7a.



Obr. 3.9: Testovací přesnost a průběhy trénování sítě 3.7b.



(a) Struktura třetí navržené sítě.

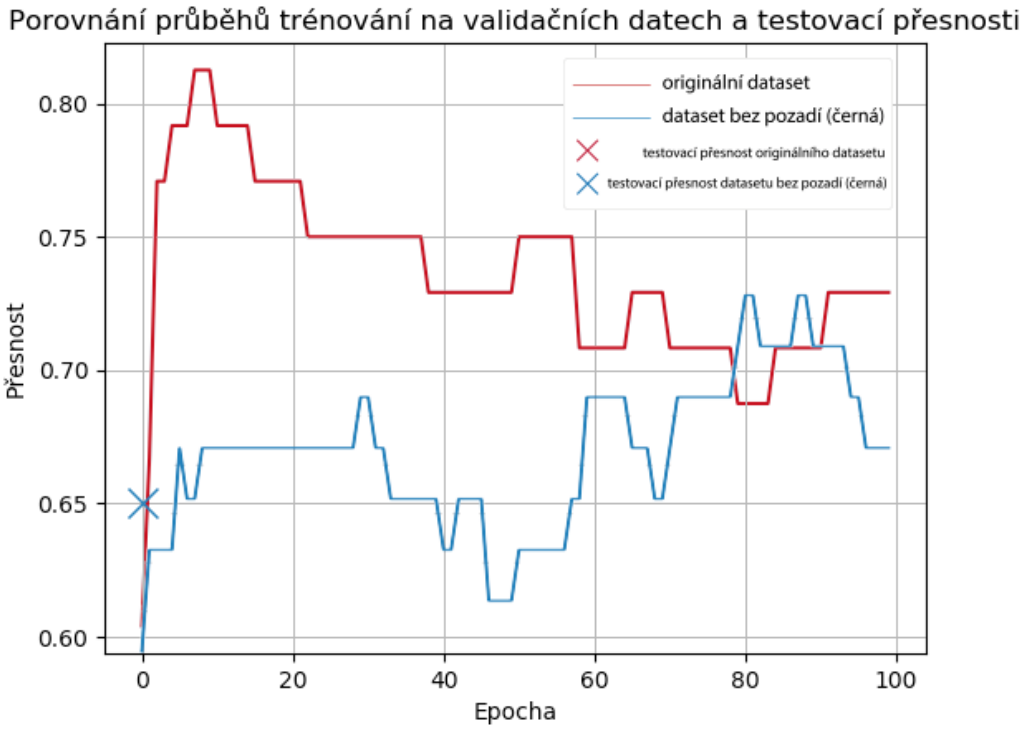


(b) Struktura čtvrté navržené sítě.

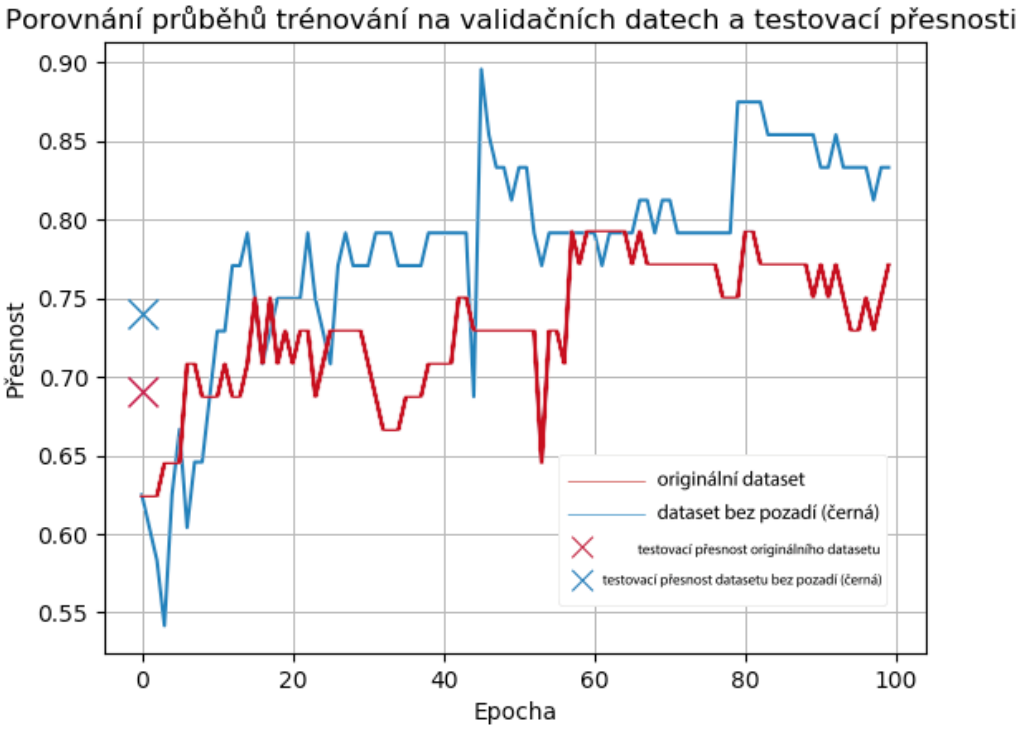
Obr. 3.10: Navržené struktury pro testování na reálných datech.

Tato síť naopak od předchozích případů vykazuje známky nedoučení na originální datové množině, kdy výsledný model není dostatečně komplexní, což se projevuje na testovací přesnosti. Na druhou stranu průběh 3.11 dokazuje, že pro data bez šumu je možné použít jednodušší síť. Výsledný model s méně parametry je pak na reálných datech schopen dosáhnout lepší přesnosti, než model s originálními daty.

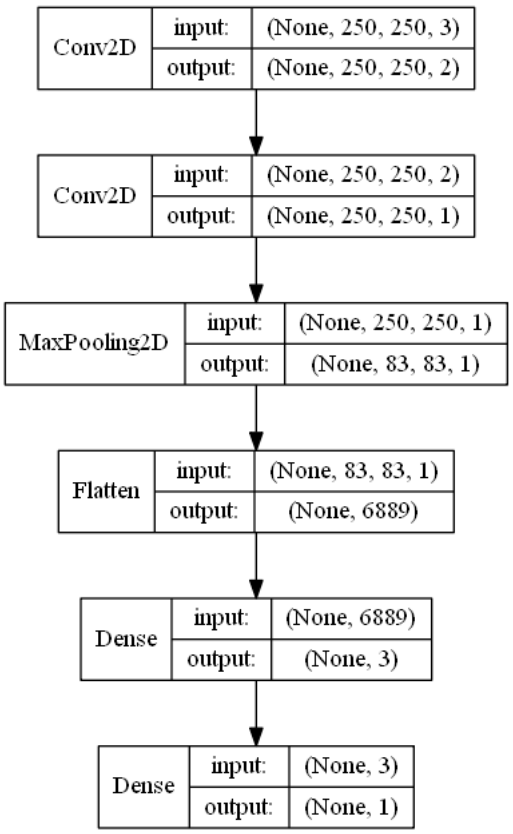
Při návrhu sítě 3.10b byl mírně navýšen počet trénovatelných parametrů. Z průběhů trénování na grafu 3.12 je vidět, že modely obou datových množin již nejsou výrazně přetrénované a jsou schopné dosáhnout vyšší testovací přesnosti. Model s originálními daty dosáhl menší výsledné přesnosti než model s daty bez pozadí, který měl sice horší průběh začátku trénování, ale během prvních 20 epoch dosahoval vyšší validační přesnosti. Z průběhu trénování 3.14 sítě 3.13 je vidět, že při razantnějším navýšení trénovatelných parametrů má model s originálními daty navrch v testovací přesnosti, kdy i přes téměř stejnou závěrečnou validační přesnost



Obr. 3.11: Testovací přesnost modelu bez pozadí 65 %, originální 52 %.



Obr. 3.12: Testovací přesnost modelu bez pozadí 74 %, originální 69 %.



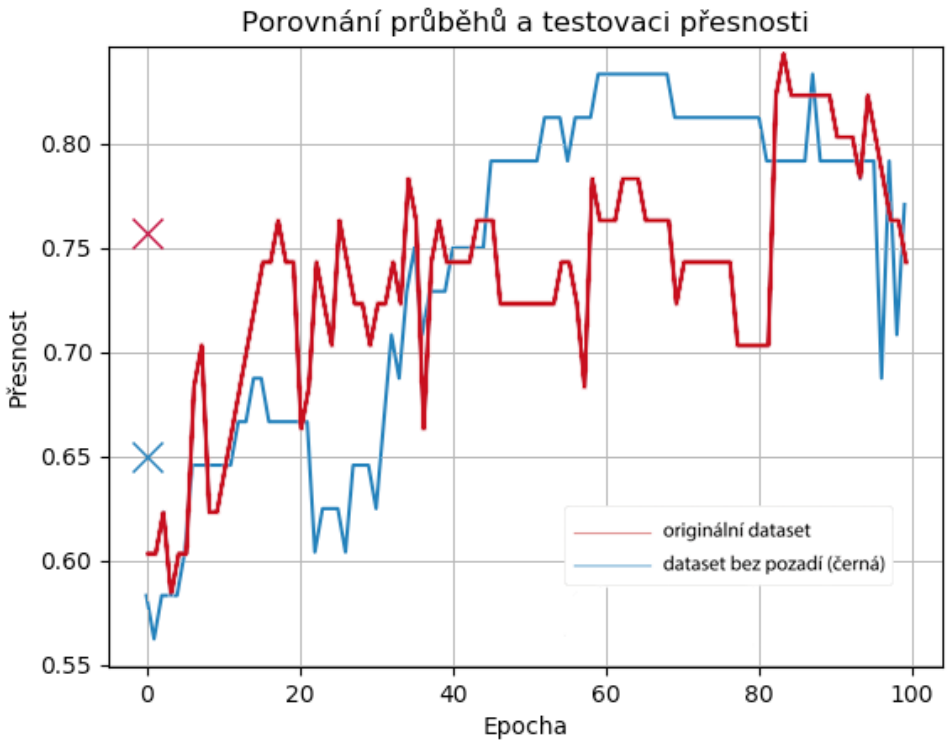
Obr. 3.13: Struktura páté navržené sítě pro testování na reálných datech.

dosahuje na testovacích datech o 11 % přesnějšího výsledku. Tento výsledek je logický. Vezmeme-li v úvahu větší šum na originální datové množině a vyšší počet parametrů navržené sítě, je natrénovaný model více robustnější a na testovací data, původem z originální datové množiny, je více připraven.

3.3 Přenosové učení

U tohoto experimentu jsou převážně využity stejně sestavené neuronové sítě jako u testování na reálných datech. Jak bylo vysvětleno v teoretické části 1.5, je proveden experiment pro dva případy.

Proces přenosového učení na obou databázích probíhá následovně: Do sítě jsou načteny vzorky s odstraněným pozadím, kdy probíhá trénování a následná validace. V prvním experimentu je při dosažení určité přesnosti proces zastaven pomocí funkce Kerasu (tzv. *callback*) `EarlyStopping` a model s váhami je průběžně ukládán pomocí funkce `ModelCheckpoint`. Tyto uložené váhy jsou následně znovu načteny, ale do sítě jsou nyní dodávána trénovací a validační data z originální datové množiny. Pro první ze dvou experimentů je z důvodu demonstrace opakování trendu vyzkou-

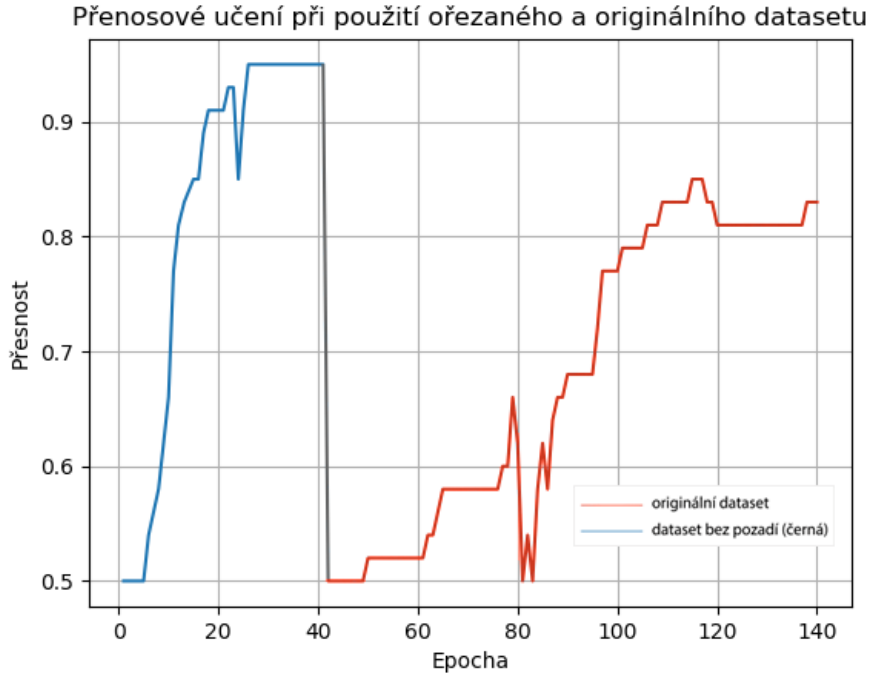


Obr. 3.14: Testovací přesnost modelu bez pozadí 65 %, originální 76 %.

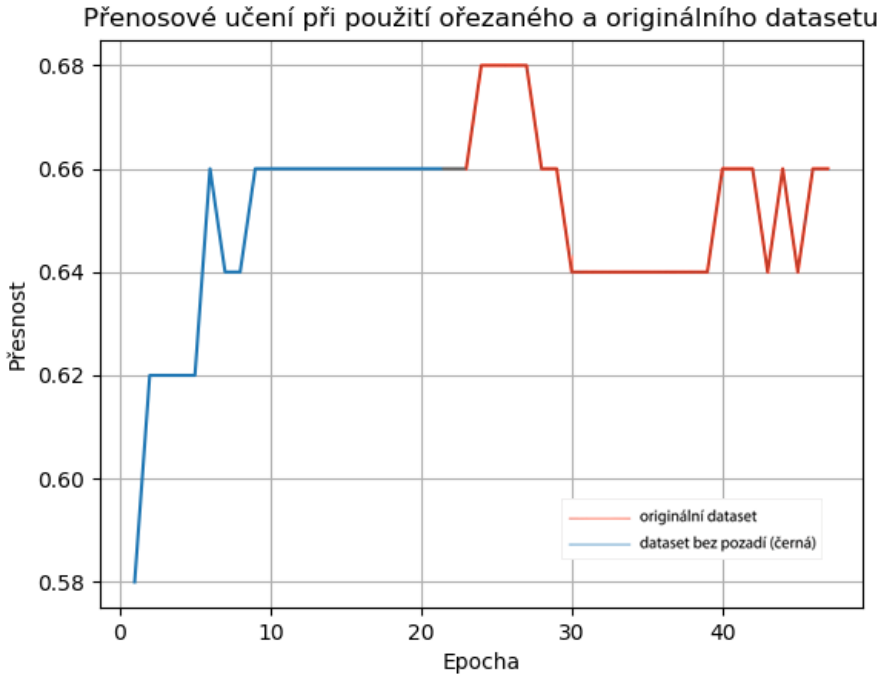
šeno více konvolučních sítí stejně jako v případě testování na reálných datech. Při porovnávání průběhů je sledována konečná přesnost, které dosáhne originální datová množina s přenesenými váhami.
V níže uvedených grafech jsou křivky zvýrazněny třemi barvami podle použitých datových množin. Modrá barva zastupuje datovou množinu bez pozadí, na které vždy začíná trénování. Černá barva v grafech znázorňuje přechod, kdy byly váhy přeneseny na originální (oranžovou) datovou množinu.

Stejně jako u testování na reálných datech je z průběhu 3.15 patrné přetrénování sítě na datech bez pozadí. Při přenesení tohoto modelu je z důvodu přetrénování zahájeno trénování na stejné validační přesnosti (50 %) od začátku. V tomto případě není možné objektivně zhodnotit dosaženou přesnost, jelikož v modelu datové množiny bez pozadí nejsou natrénované využitelné váhy pro trénování originální datové množiny (jsou přetrénované).

Navrhnutá neuronová síť 3.7b byla přeskočena, jelikož by s velkou pravděpodobností vykazovala podobné výsledky, jako síť 3.7a z důvodu mírné přetrénovanosti. Na grafu 3.16 je vyobrazen průběh až v pořadí třetí navrhnuté neuronové sítě 3.10a. Průběh učení této neuronové sítě byl na datové množině bez pozadí pozastaven už při 21. epoše. Váhy byly přeneseny a na začátku trénování se validační přesnost dále



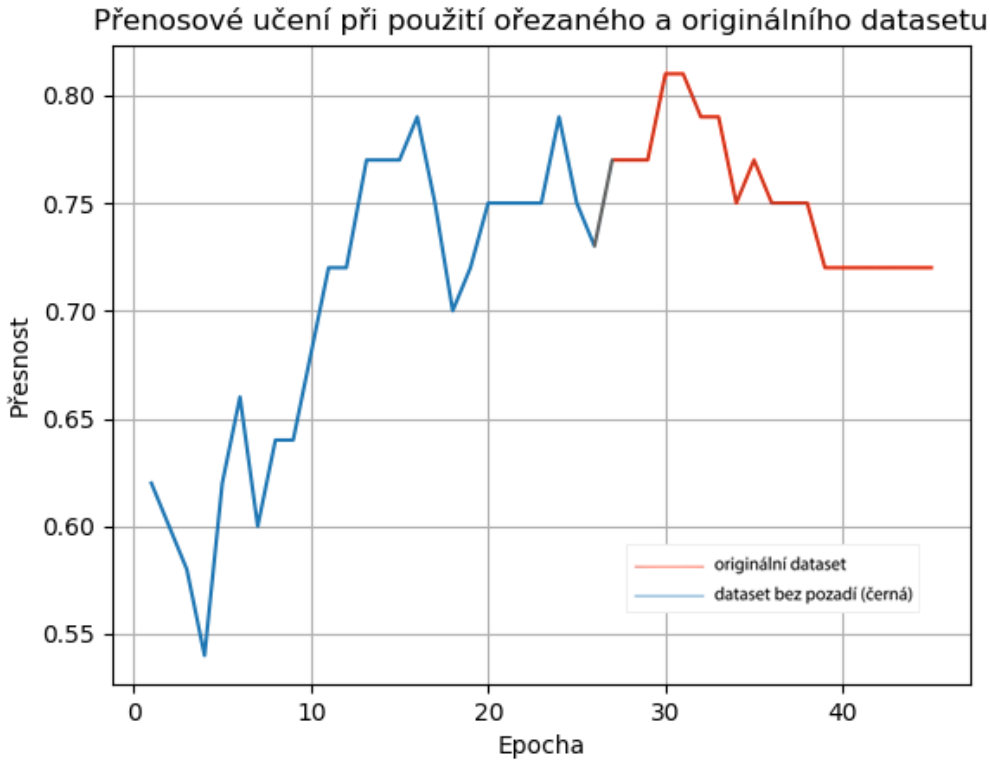
Obr. 3.15: Průběh validační přesnosti struktury sítě 3.7a.



Obr. 3.16: Průběh validační přesnosti struktury sítě 3.10a.

zvyšovala. Ve výsledku bylo ovšem dosaženo stejné přesnosti jako před přenesením vah.

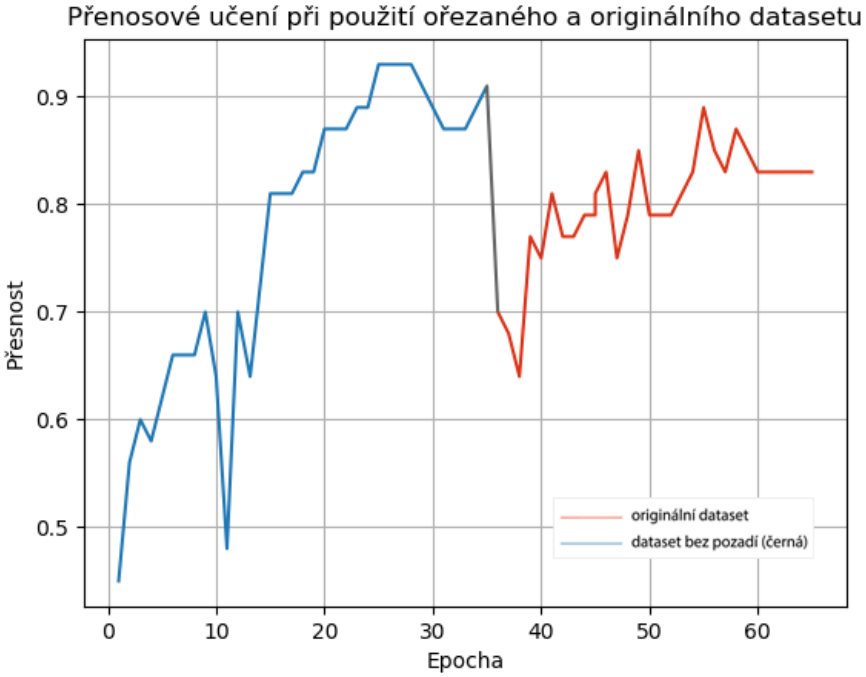
Podobný průběh trénování vykazuje i síť 3.10b na grafu 3.17, kde je po přenesení vah opět vidět nárůst, ale následně pokles, kdy se konečná hodnota ustálila na srovnatelné hodnotě validační přesnosti jako před přenesením vah.



Obr. 3.17: Průběh validační přesnosti struktury sítě 3.10b.

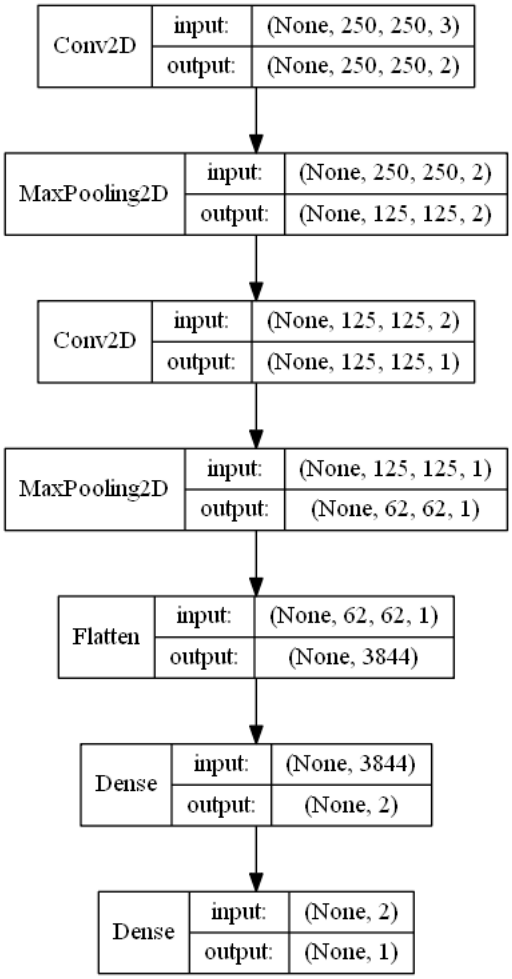
U struktury sítě 3.13 je průběh 3.18 opět podobný jako u předchozích, ale konečná přesnost je výrazněji menší než před přenesením vah (o 8 %). U této sítě byl navýšen počet parametrů od předchozích sítí, což se negativně projevilo na experiment s přenesením vah.

Poslední vyzkoušenou a nově přidanou strukturou je 3.20. U této sítě byl opět snížen počet trénovatelných parametrů tak, aby byl jejich počet vyšší než u ostatních nepřetrénovaných sítí, ale zároveň nižší než u struktury 3.13. Průběh 3.19 je posledním znázorněným a je zde vidět, že po přenesení vah zůstává hodnota validační přesnosti na zhruba stejné úrovni.
Tento experiment měl objasnit, zda-li se dá při zastavení trénování, kdy se přesnost dále nezvyšuje a při následném přenesení vah, dosáhnout vyšší, stejné nebo nižší přesnosti. Při natrénování modelu bez pozadí neobsahuje přebytečný šum. Při následném přenesení tohoto modelu na originální data trénování započne podle před-



Obr. 3.18: Průběh validační přesnosti struktury sítě 3.10a.

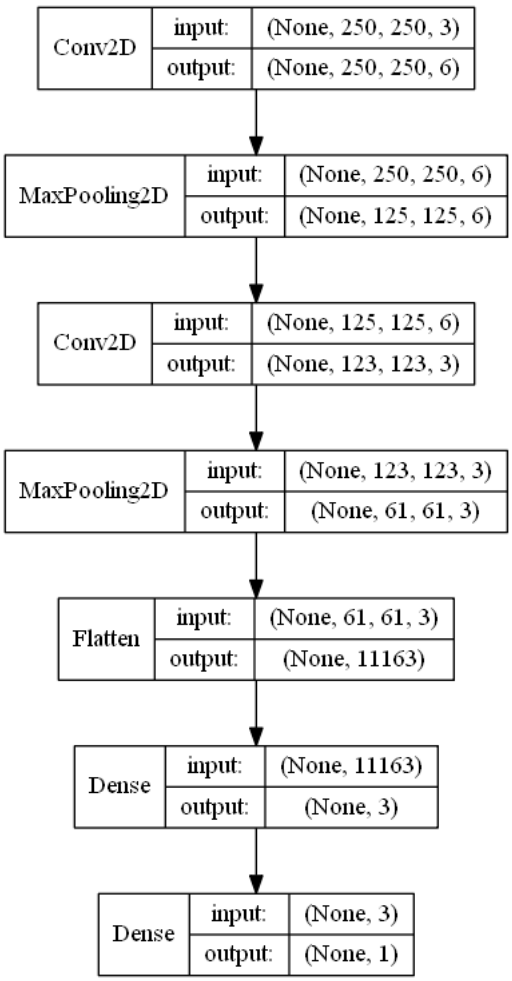
Obr. 3.19: Průběh validační přesnosti struktury sítě 3.20.



Obr. 3.20: Struktura páté navržené sítě pro testování na reálných datech.

pokladu (v nepřetrénovaném případě začíná trénování na vyšší přesnosti), ale při použití na trénování originálních dat už ve vyzkoušených případech není schopen dosáhnout vyšší přesnosti, než které bylo dosaženo před přenesením vah.

Při druhém experimentu přenosového učení je u každého průběhu trénování určena pevná hranice přesnosti a počet epoch. Je vynechána mírně přetrénovaná síť 3.7a a je nahrazena nově navrženou sítí 3.21. Průběh experimentu je následující: Trénování konvoluční neuronové sítě je spuštěno na originální datové množině, kde za daný počet epoch je dosaženo určité přesnosti. V tomto případě je počet epoch nastaven na 50. Paralelně s tímto průběhem trénování je spuštěn na stejné síti proces trénování také na datech bez pozadí, kde je určena přesnost, na které se trénování pozastaví (nižší než přesnost na originální datové množině). Tyto váhy (datové množiny bez pozadí) jsou následně přeneseny na stejnou neuronovou síť, ale s originální datovou množinou, kde je sledováno, jaký počet epoch je nutný k dosažení stejné



Obr. 3.21: Struktura navržené sítě pro druhý experiment.

přesnosti jako na trénování 50 epoch originální datové množiny. Z tohoto experimentu je možné určit, zda-li se tímto způsobem přenesení vah dá urychlit proces trénování při použití dat bez pozadí. Trénování není zastaveno, ale pokračuje dále na stejný počet epoch jako originální datová množina (50). Tentokrát je zase možné určit, zda-li přenos vah ne jen zrychlí průběh učení při omezené datové množině o šum, ale také jestli může být tímto způsobem dosaženo vyšší přesnosti. Experiment je prováděn na stejném rozdělení datové množiny jako v předešlých případech.

Výsledky tohoto experimentu jsou uvedeny v tabulce 3.1. V prvním případě bylo trénování originální datové množiny pozastaveno na přesnosti 66 %, které bylo dosaženo na 50. epoše (v tabulce označeno jako N u originální datové množiny). Zároveň bylo trénování spuštěno na datové množině bez pozadí, kde už na 4. epoše bylo dosaženo přesnosti 83 %. Tento model byl uložen a následně přenesen na originální datovou množinu, kde již po 11. epoše (v tabulce označeno jako N u datové množiny

Tab. 3.1: Výsledky druhého experimentu s přenosovým učením

Síť	Datová množina	Stop (epocha)	Přesnost (%)	N	Přesnost (%)	Zrychlení (%)	Přesnost: zlepšení/ zhoršení (%)
3.21	Bez pozadí	4	83	11	66	70	19
	Originální	–	–	50	66	–	–
3.10a	Bez pozadí	6	66	2	73	84	7
	Originální	–	–	50	66	–	–
3.10b	Bez pozadí	10	68	2	73	79	6
	Originální	–	–	50	73	–	–
3.13	Bez pozadí	11	64	14	81	50	2
	Originální	–	–	50	79	–	–
3.20	Bez pozadí	11	64	8	66	62	0
	Originální	–	–	50	66	–	–

bez pozadí) bylo dosaženo stejné přesnosti, jako dosáhla originální datová množina po 50. epochách. Z těchto výsledků je možné určit zrychlení učení (porovnání počtu epoch, kdy bylo dosaženo stejné přesnosti). Pro neuronovou síť 3.21 byl proces trénování zrychlen o 70 %. Tento proces nebyl pozastaven a byl nechán doběhnut do 50. epochy, stejně jako u originální datové množiny. Z tohoto bylo možné zjistit zlepšení nebo zhoršení přesnosti, kde u první vyzkoušené struktury bylo dosaženo lepší přesnosti o 19 %. Se stejným postupem jsou uvedeny další výsledky v tabulce pro všechny vyzkoušené neuronové sítě.

U všech struktur došlo ke znatelnému zrychlení procesu trénování přenesením vah z datové množiny bez pozadí na originální datovou množinu. V některých případech byla konečná hodnota validační přesnosti vyšší v porovnání s původním průběhem na originální datové množině.

3.4 Velikost databáze

Velikost a kvalita datové množiny jsou pro trénování konvolučních neuronových sítí klíčové vlastnosti. V další části této práce je experimentováno s velikostí datové množiny a jejím vlivu na výslednou přesnost modelu. Je sledována dosažená testovací přesnost při binární klasifikaci 4 datových množin (třídy) po dvou dvojicích (A a B, C a D). Všechny třídy jsou dále rozděleny na 3 části – trénovací, validační a testovací množinu, kde prvky validační a testovací množiny jsou fixní. Datová množina ob-

Tab. 3.2: Testovací přesnosti při daném počtu vzorků trénovací množiny sítě 3.7a.

Třídy	**AB**
N	Přesnost (%)
140	66
120	63
100	70
80	63
60	56
50	53
40	66
30	53

Třídy	**CD**
N	Přesnost (%)
140	76
120	73
100	73
80	73
60	70
50	63
40	70
30	63

(a) Třídy A a B.

(b) Třídy C a D.

Obr. 3.22: Závislosti přesnosti na počtu trénovacích vzorků.

sahuje vždy celkem 200 vzorků. Trénovací množina obsahuje vždy stejně proměnný počet prvků (30 – 140) s krokem 10 od 30 do 60 a následně s krokem 20 od 60 do 140. Tento počet je v grafech označen jako N. Všechny třídy jsou rozděleny ve stejném poměru s tím, že poměr trénovací množiny ku validační a testovací s postupujícím pokusem klesá. Pro demonstraci vlivu nedostatku trénovacích dat na přesnost nejsou udržovány poměry rozdělení do tří množin. Jako v předchozích případech je použita datová množina LFW, ze které jsou vybrány 4 třídy s nejvyšším počtem vzorků pro experiment. Do tabulek jsou uvedeny počty prvků trénovacích množin sestupně a dosažená testovací přesnost. Z těchto výsledků jsou následně vyneseny závislosti.

Předpokladem pro výsledky tohoto experimentu je, že se snižujícím se počtem trénovacích dat by se měla snižovat i schopnost modelu správně klasifikovat na testovacích datech. I přes kolísavý průběh se tento předpoklad u prvních dvou vyzkou-

Tab. 3.3: Testovací přesnosti při daném počtu vzorků trénovací množiny sítě 3.24.

Třídy	**AB**
N	Přesnost (%)
140	76
120	73
100	73
80	66
60	66
50	53
40	50
30	53

Třídy	**CD**
N	Přesnost (%)
140	66
120	66
100	63
80	50
60	73
50	73
40	70
30	70

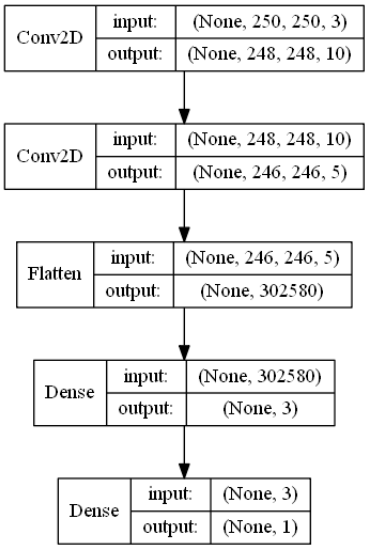
(a) Třídy A a B.

(b) Třídy C a D.

Obr. 3.23: Závislosti přesnosti na počtu trénovacích vzorků.

šených neuronových sítí potvrdil. V případě výsledkové tabulky 3.3 pro síť 3.24, konkrétně třídy C a D je ale vidět, že průběh se tomuto trendu vymyká. Na tomto průběhu je nejlépe vidět jev, kterým se zabývá oblast studia konvolučních neuronových sítí zvaná aktivní učení. Je zde vidět, že například při 60 trénovacích vzorcích dosahuje síť vyšší testovací přesnosti než při 140 a naopak u 80 vzorků je přesnost 50 % navzdory přesnosti 75 % u 60 vzorků. Tento jev může být právě způsoben správným výběrem reprezentativních vzorků v datové množině, na kterých je síť lépe naučena, než na větším množství dat, které mohou do sítě přinést šum. Nicméně u dalších vynesených testovacích přesností je vidět, že přesnost se s klesajícím počtem vzorků zmenšuje.

Síť 3.10a nebyla schopná dosáhnout lepších výsledků testovací přesnosti než 53 % u obou binárních klasifikací, mnohdy se přesnost pohybovala i pod hranicí 50 %.



Obr. 3.24: Struktura navržené sítě pro experiment s velikostí databáze.

Pro tuto síť byl nedostatek trénovatelných parametrů a nebyla tak schopná správně klasifikovat. Zde se taktéž projevuje jev, kdy testovací data zrovna nepasují natrénovanému modelu. Poslední vyzkoušenou strukturou pro tento experiment je síť 3.10b, jejíž výsledky jsou uvedeny v tabulce 3.5. U binární klasifikace tříd A a B se opět projevil jev, kdy data při 50 a 100 vzorcích nepasovala natrénovanému modelu. U tříd C a D je ale opět vidět, že s klesajícím počtem trénovacích vzorků klesá výsledná přesnost modelu.

Tab. 3.4: Testovací přesnosti při daném počtu vzorků trénovací množiny sítě 3.10a.

Třídy	**AB**
N	Přesnost (%)
140	53
120	46
100	43
80	50
60	46
50	50
40	53
30	53

Třídy	**CD**
N	Přesnost (%)
140	50
120	43
100	53
80	50
60	46
50	46
40	50
30	46

Při sestavování neuronových sítí byly v rámci ukázky vlivu pozadí na trénování vyzkoušeny i struktury s inspirací k jejich složení u globálně známých modelů konvolučních neuronových sítí jako je LeNet-5, VGG-16 nebo AlexNet. Při množství vzorků v použité datové množině a počtu nastavených parametrů sítí vykazovaly

(a) Třídy A a B.

(b) Třídy C a D.

Obr. 3.25: Závislosti přesnosti na počtu trénovacích vzorků.

sítě na datové množině bez pozadí vysokou přetrénovanost, kdy nebyly schopny vyšší testovací přesnosti než 50 %. Tyto výsledky jsou vhodné pro demonstraci nedostatečné velikosti databáze na trénování konvolučních neuronových sítí s vysokým počtem trénovatelných parametrů. Dvě navržené struktury jsou uvedeny na obrázcích 3.27a a 3.27b s příslušným průběhem trénování a vyznačením testovací přesnosti pro obě datové množiny.

Tab. 3.5: Testovací přesnosti při daném počtu vzorků trénovací množiny sítě 3.10a.

Třídy	**AB**
N	Přesnost (%)
140	66
120	63
100	53
80	60
60	70
50	56
40	60
30	63

Třídy	**CD**
N	Přesnost (%)
140	63
120	66
100	56
80	53
60	53
50	50
40	46
30	43

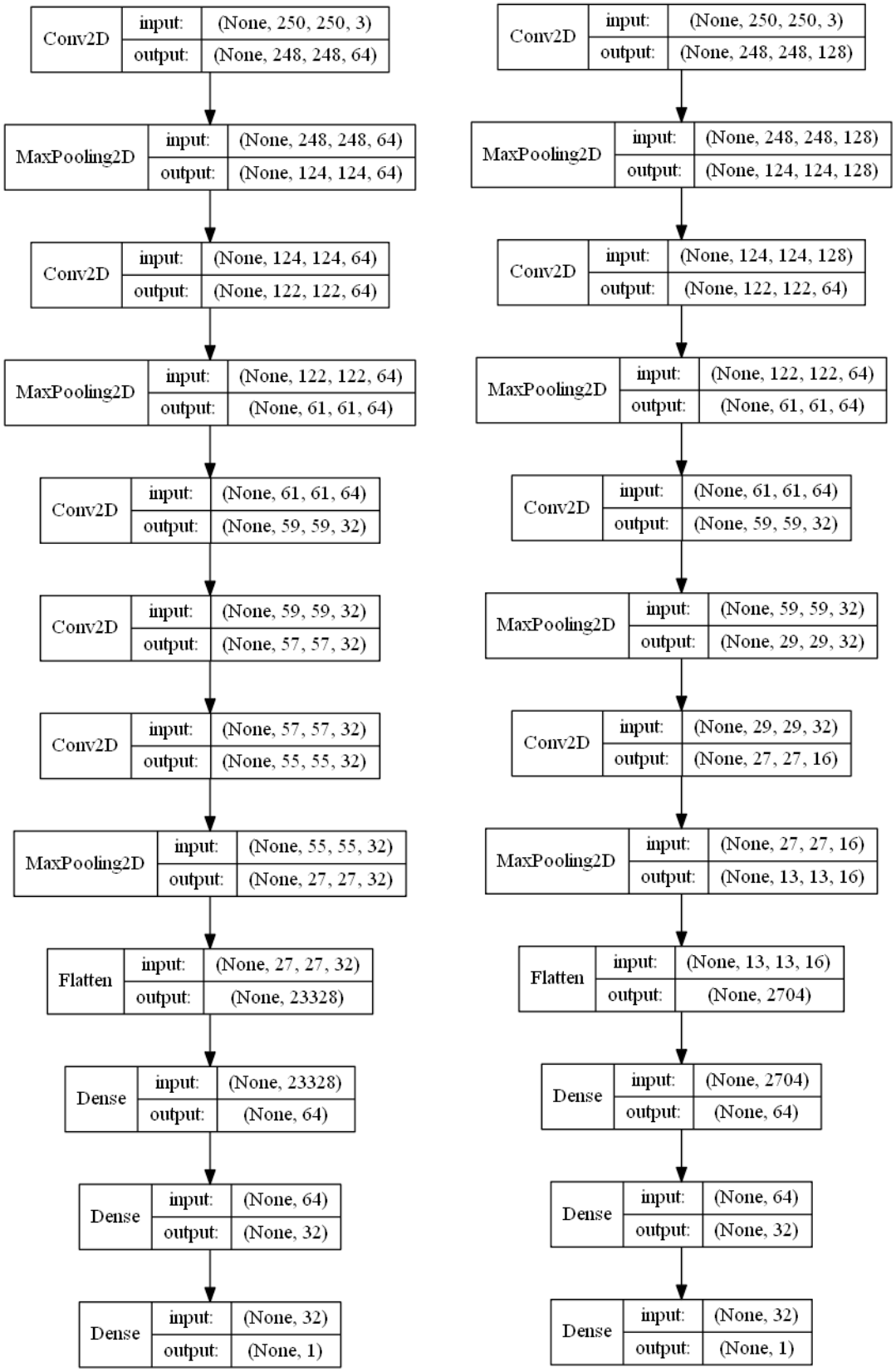
(a) Třídy A a B.

(b) Třídy C a D.

Obr. 3.26: Závislosti přesnosti na počtu trénovacích vzorků.

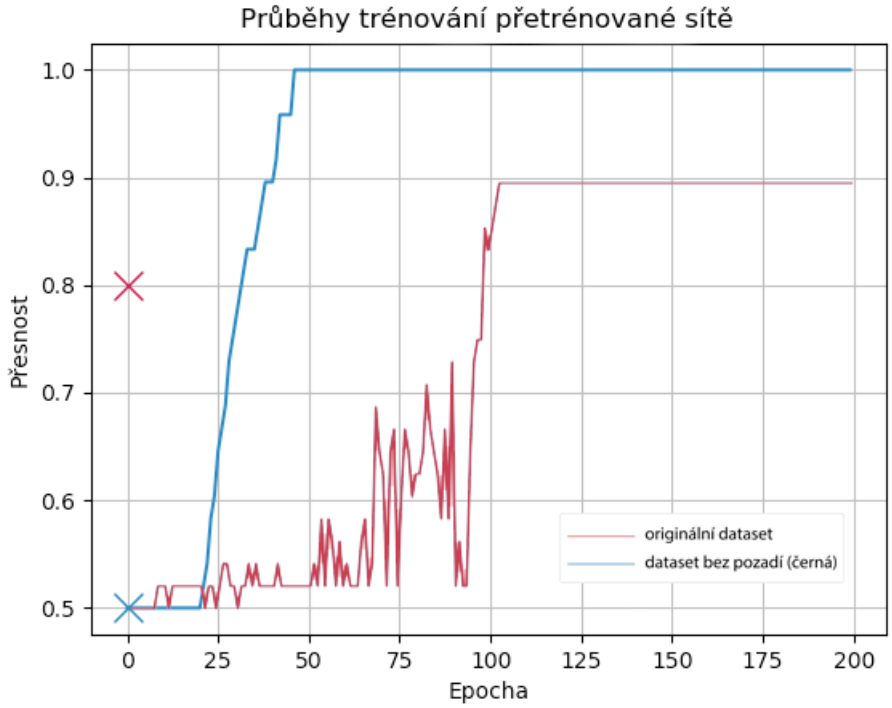
U sítě 3.27b je na grafu 3.28 vidět zřetelná přetrénovanost modelu datové množiny bez pozadí, kde během prvních 50 epoch bylo dosaženo validační přesnosti 100 %, ale na testovacích datech model nebyl schopný vykázat přijatelný výsledek. Originální datová množina si vedla poměrně dobře, kdy při testování dosáhla 80% přesnosti. Ten samý trend se opakoval i u druhé vyzkoušené struktury 3.27b v grafu 3.29, kde bylo pouze dosaženo nižší přesnosti u originální datové množiny z důvodu snížení počtu trénovatelných parametrů. Pro obě sítě a pro datovou množinu bez pozadí je příliš trénovatelných parametrů vzhledem k velikosti dostupné datové množiny.

Předešlé experimenty byly provedeny na velmi omezené datové množině o nízkém počtu vzorků. Z tohoto důvodu je proveden experiment s větší datovou množinou s názvem Psi versus kočky, která obsahuje v trénovací množině 25 000 různě vel-

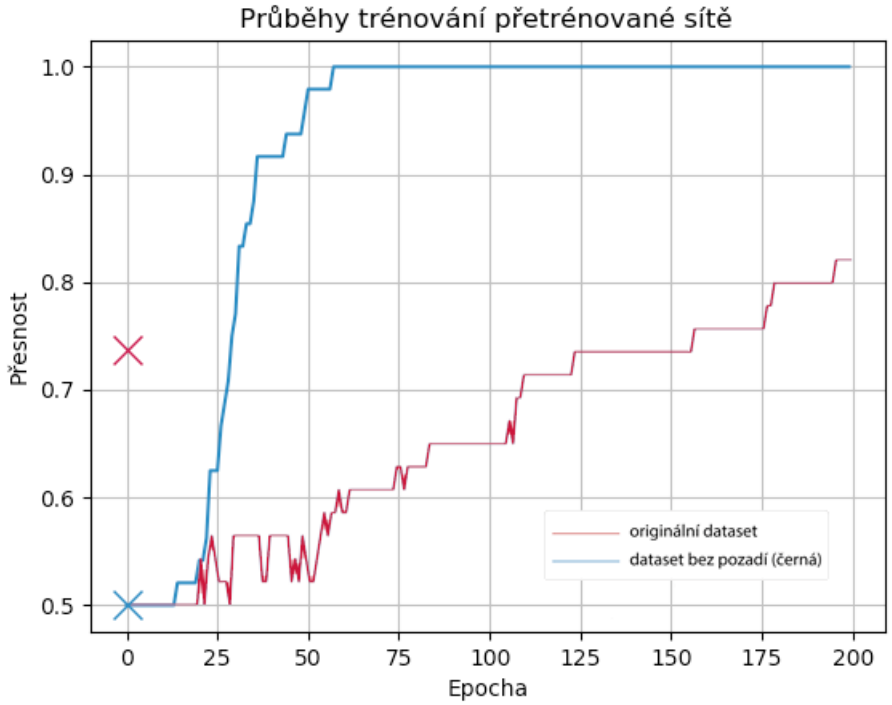


(a) První navržená síť. (b) Druhá navržená síť.

Obr. 3.27: Navržené přetrénované struktury pro experiment.

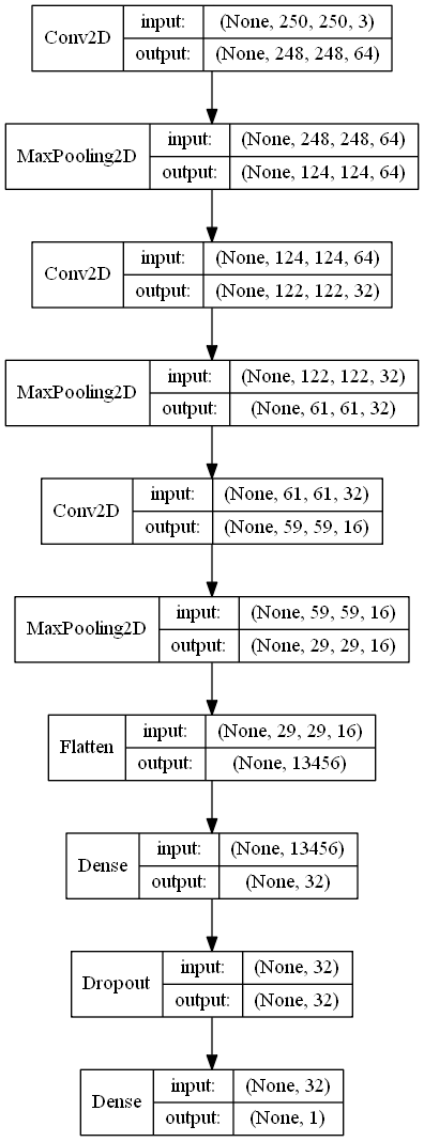


Obr. 3.28: Průběh trénování a dosažená testovací přesnost struktury sítě 3.27a.



Obr. 3.29: Průběh trénování a dosažená testovací přesnost struktury sítě 3.27b.

kých obrazů rozdělených do dvou tříd. K volnému stažení a použití je k dispozici i testovací množina. V následujícím experimentu, který probíhá stejně jako pokusy se snižováním počtu vzorků trénovací množiny, je navržena nová konvoluční neuronová síť, jejíž struktura je na obrázku 3.30. V počátku pokusu jsou množiny navzájem rozděleny v poměru 70:15:15 %, postupně jsou odebírány vzorky z trénovací množiny.

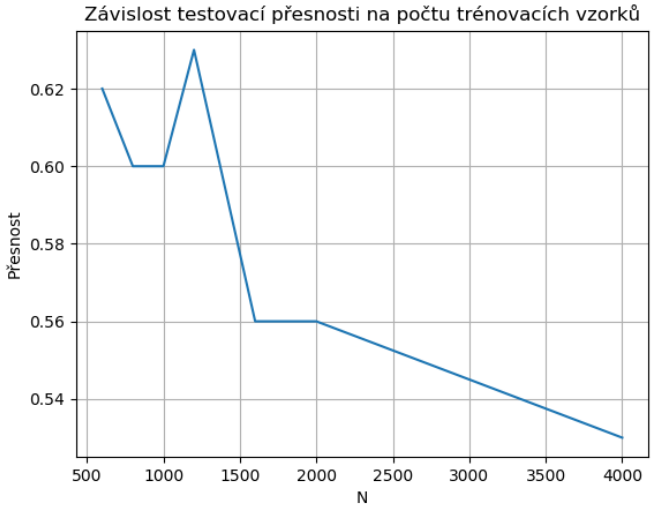


Obr. 3.30: Struktura navržené sítě pro experiment s velikostí databáze.

Výsledky testovacích přesností při určitém počtu vzorků v databázi jsou uvedeny v tabulce 3.6 a grafu 3.31. Z výsledků je patrný stejný trend jako u předešlého pokusu. Je zde vidět snižování přesnosti modelu s větším počtem trénovacích vzorků, což odporuje teorii, která vyplývá z funkce konvoluční neuronové sítě. Je očekáváno, že při zvýšení objemu trénovací množiny dojde k přidání informací, což by ve vý-

Tab. 3.6: Testovací přesnosti při daném počtu vzorků trénovací množiny sítě 3.30.

Databáze	**Psi vs. kočky**
N	Přesnost (%)
4000	53
2000	56
1600	56
1200	63
1000	60
800	60
600	62



Obr. 3.31: Graf hodnot z tabulky 3.6.

sledku mělo znamenat vyšší testovací přesnosti. Průběh tohoto experimentu ovšem dokazuje, že ne vždy toto pravidlo platí a při tvorbě či výběru datové množiny je třeba dbát na reprezentativnost datové množiny vůči klasifikovaným datům.

4 Závěr

Tato práce se zabývá vlivem pozadí (šumu) a velikosti databáze na trénování neuronových sítí pro klasifikaci obrazu. V teoretické části byla zpracována současná řešení, byl rozebrán princip neuronových sítí pro klasifikaci obrazu a byly uvedeny globálně používané datové množiny s výběrem jedné pro experiment. Na závěr teoretické části byly zmíněny typy učení.

V experimentální části byly porovnány dva průběhy – originální datová množina a datová množina se začerněným pozadím, kde byly prováděny pokusy s parametry a různými strukturami sítí. Při inspiraci již existujících sítí dosahovala poměrně dobrých výsledků (z časového hlediska a validační přesnosti) struktura inspirována sítí LeNet-5, dále byla vyzkoušena struktura inspirována sítí VGG-16. Výsledkem tohoto experimentu je, že pozadí klasifikovaného objektu má vliv na trénování konvolučních neuronových sítí pro klasifikaci obrazu v případě nízkého poměru signálu a šumu. Tento vliv tedy závisí jak na poměru šumu a signálu fotografie, tak i na barvě šumu.

Při trénování byly ukládány modely a postupně byla testována přesnost na reálných datech. Při tomto pokusu se ukázalo, že model s daty bez pozadí je schopen klasifikovat originální data s poměrně dobrou přesností, mnohdy i vyšší než model s originálními daty, kdy ale záleží na počtu trénovatelných parametrů každé sítě.

Hlavním přínosem práce je experiment s využitím přenosového učení za účelem zrychlení procesu trénování, případně i zvýšení testovací přesnosti při přenesení vah z datové množiny s odstraněným pozadím na originální data. Cílem bylo zjistit, zdali při přenesení vah dat bez pozadí na originální data je možné zrychlení procesu trénování při dané epoše k zastavení trénování. Ve všech případech bylo zjištěno výrazné zrychlení trénování, v některých případech i zvýšení konečné validační přesnosti.

Poslední částí bylo experimentování s vlivem velikosti datové množiny na schopnost modelu klasifikovat reálná data. Bylo zjištěno, že při snižování počtu trénovacích dat se ve většině případů snižuje testovací přesnost. V některých případech došlo ke zvýšení přesnosti, což může být způsobeno vhodnými reprezentativními vzorky dat, které lépe natrénují model (tomuto se ale podrobně věnuje oblast aktivního učení).

Literatura

[1] *Data augmentation for improving deep learning in image classification problem* [online]. A. Mikolajczyk, M. Grochowski, [cit. 3. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8388338>

[2] *Simple convolutional neural network on image classification* [online]. T. Guo, J. Dong, H. Li, Y. Gao, [cit. 3. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8078730>

[3] *Context Augmentation for Convolutional Neural Networks* Purdue University: [online]. A. Dundar, I. Garcia-Dorado, Purdue: 2017, [cit. 27. 10. 2018]. Dostupné z URL: <https://arxiv.org/pdf/1712.01653.pdf>.

[4] *PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing* [online]. C. Barnes, E. Shechtman, A. FinkelStein, D. B. Goldman, [cit. 27. 10. 2018]. Dostupné z URL: <https://gfx.cs.princeton.edu/pubs/Barnes_2009_PAR/patchmatch.pdf>.

[5] *Data augmentation for improving deep learning in image classification problem* [online]. A. Mikolajzcyk, M. Grochowski, [cit. 27. 10. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/document/8388338>.

[6] *Image Analogies* [online]. A. Hertzmann, CH. E. Jacobs, N. Oliver, B. Curless, D. H. Salesin [cit. 27. 10. 2018]. Dostupné z URL: <https://mrl.nyu.edu/publications/image-analogies/analogies-72dpi.pdf>.

[7] *AI Benchmark: Running Deep Neural Networks on Android Smartphones*, [online].[cit. 11. 11. 2018]. Dostupné z URL: <https://arxiv.org/pdf/1810.01109.pdf>

[8] PSTRUŽINA, Karel. *Etudy o mozku a myšlení*, Praha: Vysoká škola ekonomická, 1994. ISBN 80-7079-280-9.

[9] *Neural networks* [online]. Ch. Stergiou, D. Siganos, [cit. 3. 11. 2018]. Dostupné z URL: <https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html>.

[10] *Supervising an Unsupervised Neural Network*, [online].[cit. 4. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/abstract/document/5176011>

[11] *Gradient2.png*, [online]. [cit. 10. 11. 2018]. Dostupné z URL: <https://commons.wikimedia.org/wiki/File:Gradient2.png>

[12] *The Gradient and Directional Derivative*, [online]. [cit. 10. 11. 2018]. Dostupné z URL: <https://math.oregonstate.edu/home/programs/undergrad/CalculusQuestStudyGuides/vcalc/grad/grad.html>

[13] *Closing the Generalization Gap of Adaptive Gradient Methods in Training Deep Neural Networks*, [online]. [cit. 10. 11. 2018]. Dostupné z URL: <https://arxiv.org/pdf/1806.06763.pdf>

[14] *Optimizers*, [online]. [cit. 10. 11. 2018]. Dostupné z URL: <https://keras.io/optimizers/>

[15] *Overfitting problem and the over-training in the era of data*, [online]. [cit. 29. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8260032>

[16] BURNHAM, K. P.; ANDERSON, D. R. (2002), *Model Selection and Multimodel Inference* (2nd ed.), Springer-Verlag. [cit. 29. 11. 2018].

[17] *Why Tanh: Choosing a Sigmoidal Function* [online]. B. L. Kalman, S. C. Kwasny, [cit. 3. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=227257>.

[18] *Activation Functions: Comparison of Trends in Practice and Research for Deep Learning* [online]. Ch. E. Nwankpa, W. Iljomah, A. Gachagan, S. Marshall, [cit. 3. 11. 2018]. Dostupné z URL: <https://arxiv.org/pdf/1811.03378.pdf>.

[19] *Deep Learning using Rectified Linear Units (ReLU)*, [online]. [cit. 20. 2. 2019]. Dostupné z URL: <https://arxiv.org/pdf/1803.08375.pdf>

[20] *CS231n Convolutional Neural Networks for Visual Recognition*, Stanford University. [online]. [cit. 3. 11. 2018]. Dostupné z URL: <http://cs231n.github.io/classification/>

[21] *Augmented Convolutional Feature Maps for Robust CNN-based Camera Model Identification*, [online]. [cit. 17. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8297053>

[22] PETKOV, N.; WIELING, M. B., *Gabor filter for image processing and computer vision*, University of Groningen, [online]. [cit. 17. 11. 2018]. Dostupné z URL: <http://matlabserver.cs.rug.nl/edgedetectionweb/web/edgedetection_params.html>

[23] *Classification with Learning k-Nearest Neighbors*, [online]. J. Laaksonen, E. Oja, [cit. 17. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=549118>

[24] *Overview of Use of Decision Tree algorithms in Machine Learning*, [online]. A. Navada, A. N. Ansari, S. Patil, B. A. Sonkamble, [cit. 17. 11. 2018]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5991826>

[25] *Gradient-Based Learning Applied to Document Recognition*, [online]. Y. LeCun, L. Bottou, Y. Bengoi, P. Haffner, [cit. 17. 11. 2018]. Dostupné z URL: <http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf>

[26] *Build a Convolutional Neural Network using Estimators*, [online]. [cit. 11. 11. 2018]. Dostupné z URL: <https://www.tensorflow.org/tutorials/estimators/cnn>

[27] *MNIST database*, [online]. [cit. 23. 11. 2018]. Dostupné z URL: <http://yann.lecun.com/exdb/mnist/>

[28] *ImageNet*, [online]. [cit. 23. 11. 2018]. Dostupné z URL: <http://image-net.org/about-overview>

[29] *The CIFAR-10 dataset*, [online]. [cit. 23. 11. 2018]. Dostupné z URL: <https://www.cs.toronto.edu/~kriz/cifar.html>

[30] *Labeled Faces in the Wild (README)*, [online]. [cit. 23. 11. 2018]. Dostupné z URL: <http://vis-www.cs.umass.edu/lfw/README.txt>

[31] *Keras: The Python Deep Learning library*, [online]. [cit. 24. 11. 2018]. Dostupné z URL: <https://keras.io/>

[32] *An open source machine learning framework for everyone*, [online].[cit. 24. 11. 2018]. Dostupné z URL: <https://www.tensorflow.org/>

[33] *Labeled Faces in the Wild*, [online]. [cit. 25. 11. 2018]. Dostupné z URL: <http://vis-www.cs.umass.edu/lfw/>

[34] *Transfer Learning*, [online]. L. Torrey, J. Shavlik, University of Wisconsin [cit. 13. 3. 2019]. Dostupné z URL: <ftp://ftp.cs.wisc.edu/machine-learning/shavlik-group/torrey.handbook09.pdf>

[35] *Discriminability-based transfer between neural networks*, [online]. L. Y. Pratt [cit. 13. 3. 2019]. Dostupné z URL: <http://papers.nips.cc/paper/641-discriminability-based-transfer-between-neural-networks.pdf>

[36] *How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?*, [online]. J. Cho, K. Lee, E. Shin, G. Choy, S. Do, [cit. 24. 11. 2018]. Dostupné z URL: <https://arxiv.org/abs/1511.06348>

[37] *Active learning literature survey*, [online]. [cit. 3. 4. 2019]. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.167.4245>

Seznam symbolů, veličin a zkratek

tzn.	to znamená
tj.	to je
SGD	náhodné gradiální klesání – Stochastic Gradient Descent
ReLU	usměrněná lineární jednotka – Rectified Linear Unit
RGB	červená, zelená, modrá – Red, Green, Blue
Dataset	datová množina – Dataset
MNIST	modifikovaná databáze národního institutu pro standardy a technologie – Modified National Institute of Standards and Technology database
NIST	databáze národního institutu pro standardy a technologie – National Institute of Standards and Technology database
CIFAR	kanadský institut pro pokročilý výzkum – Canadian Institute for Advanced Research
CPU	centrální procesorová jednotka – Central Processing Unit
GPU	grafická procesorová jednotka – Graphic Processing Unit
RAM	paměť s náhodným přístupem – Random Access Memory
CUDA	jednotná architektura pro výpočetní zařízení – Compute Unified Device Architecture
CNTK	sada poznávacích nástrojů společnosti Microsoft – The Microsoft Cognitive Toolkit
LFW	databáze označených tváří z různého prostředí – Labeled Faces in the Wild
vs.	versus
∇	diferenciální operátor vektorové analýzy
∂	parciální derivace
$\sigma(x)$	funkce sigmoid
$\sigma(z)_j$	funkce softmax
G_c[i, j]	2-D gabor filtr
G_s[i, j]	2-D gabor filtr
N	počet prvků trénovací množiny
nm	nanometr