Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Jak zmást konvoluční neuronové síte?

2022

Erik Daniel Murgaš

Vedoucí práce:
Mgr. Tomáš Mikula

Studijní obor: Aplikovaná informatika,
prezenční forma

Bibliografické údaje

Autor:	Erik Daniel Murgaš
Název práce:	Jak zmást konvoluční neuronové síte?
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2022
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Tomáš Mikula
Počet stran:	40
Přílohy:	1 DVD
Jazyk práce:	slovenský

Bibliograhic info

Author:	Erik Daniel Murgaš
Title:	How to confuse convolutional neural networks?
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2022
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Tomáš Mikula
Page count:	40
Supplements:	1 DVD
Thesis language:	Slovak

Anotácia

Transformácie, ktoré dokážu zmiasť neurónové siete, ale nie človeka, by mohli prispieť k skvalitneniu neurónových sietí. Odhalením transformácií, ktoré zmätú neurónové siete, ale nie človeka, môžeme tieto transformácie zaradiť medzi tréningové dáta, čím by sme natrénovali neurónovú sieť na rozpoznávanie okrem klasických obrázkov aj transformované obrázky. Je šanca, že by to mierne zvýšilo aj úspešnosť na obrázkoch bez transformácií. Opíšem základy umelej inteligencie a neurónových sietí a predstavím architektúry použitých modelov neurónových sietí. Pomocou dvoch experimentov budem skúmať vplyv transformácií na klasifikáciu obrazu u ľudí aj u neurónových sietí.

Synopsis

Transformations that can confuse neural networks, but not humans, could be used to improve neural networks. By finding transformations that confuse neural networks, but not humans, we can include these transformations as training data. This train neural networks for recognition of transformed images in addition to classic images. There is a chance that this would also slightly increase the success rate in the images without transformations. I will describe the basics of artificial intelligence and neural networks and present the architectures of the models of neural networks used. Using two experiments, I will investigate the impact of transformations on image classification in both humans and neural networks.

Kľúčové slová: Umelá Inteligencia; Konvolučné neurónové siete; Transformácie obrázkov;

Keywords: Artificial Intelligence; Convolutional neural networks; Image Transformations;

Ďakujem vedúcemu práce Mgr. Tomášovi Mikulovi, za jeho ochotu a vždy rýchlu reakciu, ďakujem PhDr. Danielovi Dostálovi PhD. za konzultáciu ohľadom dotazníku, ďakujem Jirkovi a Katke za trefné poznámky pri zostavovaní dotazníku, ďakujem rodine a priateľke Viktórii za plnú podporu počas celého štúdia.

Čestne vyhlasujem, že som celú prácu vrátane príloh vypracoval/a samostatne a za použitia iba zdrojov spomínaných v texte práce a uvedených v zozname literatúry.

dátum odovzdania práce podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam viet

Zoznam zdrojových kódov

1 Úvod

Modely neurónových sietí už od roku 2016 dosahujú lepšie výsledky v klasifikácií obrázkov ako ľudia. Podľa [1] je ľudská top-5 chyba okolo 5%, zatiaľ čo neurónové siete v roku 2016 dosahovali top-5 chybu na úrovni 3%. Čo sa ale stane, pokiaľ aplikujeme na obrázky určité transformácie? Do akej miery tieto transformácie ovplyvnia úspešnosť klasifikácie neurónovými sieťami ? A ako ovplyvnia úspešnosť klasifikácie u ľudí ? Niektoré transformácie dokážu úplne znemožniť neurónovým sieťam, ktoré nie sú špeciálne trénované, obrázok klasifikovať. Rovnaká transformácia ale u človeka vôbec nemusí ovpyvniť úspešnosť klasifikácie obrázkov. V prvom rade treba odhaliť, aké transformácie sú problémové pre neurónové siete, a zároveň neovplyvňujú schopnosť správne klasifikovať obrázky u ľudí. Transformácie, ktoré znemožnia správne klasifikovať obrázok neurónovým sieťam, ale nie ľuďom, sú potom nebezpečné z hľadiska nevhodného obsahu na internete. V dnešnej dobe existujú pokročilé technológie na filtrovanie nevhodného obsahu na sociálnych sieťach (Facebook, Instagram) alebo iných platformách určených na zábavu (YouTube, TikTok). Medzi tieto technológie patria aj neurónové siete, špeciálne trénované na klasifikáciu nevhodného obsahu. Aplikovaním transformácie na nevhodný obsah by sa zvýšila šanca, že sa nevhodný obsah na stránku dostane. Dostane sa teda k človeku, u ktorého schopnosť rozpoznať daný obsah nie je transformáciou tak moc ovplyvnená ako u neurónových sietí. Odhalenie takýchto transformácií by sa mohlo použiť na zvýšenie kvality rôznych modelov neurónových sietí, a teda aj filtrov nevhodného obsahu. Medzi tréningové dáta týchto neurónových sietí by boli zaradené aj transformované obrázky, čo by výslednej neurónovej sieti zvýšilo šance na správnu klasifikáciu aj takto transformovaných obrázkov.

Transformáciami na zlepšenie trénovania sa zaoberali napr. Wang s Perezom v [2] alebo Howard v [3]. V týchto prácach nešlo o využitie transformácií na zmätenie CNN. Snažili sa zväčšiť množinu tréningových dát tým, že okrem originálnych obrázkov zaradili aj nejaké jednoduché transformácie ako otáčanie, horizontálne či vertikálne prevrátenie alebo manipulovali s farbami. Tieto transformácie sú jednak výpočetne nenáročné, teda predlžujú čas tréningu neurónových sietí do únosnej miery, a zároveň zlepšujú výslednú úspešnosť na nových, predtým nevidených dátach. Vďaka tomu, sa tento spôsob tréningu neurónových sietí používa aj v praxi, kedy je bežné, že sa niektoré obrázky počas trénovania náhodne prevrátia, alebo sa manipuluje s farbami obrázkov [4].

2 Umelá inteligencia

Ľudia si často zamieňajú pojmy umelá inteligencia, strojové učenie, hlboké učenie a neurónové siete. Preto v nasledujúcej časti popíšem dané pojmy a ich vzťahy.

Umelá inteligencia Je veda o návrhu inteligentných strojov. Umelú inteligenciu delíme na dve hlavné kategórie, a to slabú a silnú. Slabá umelá inteligencia sa tiež nazýva plytká (Artificial Narrow Intelligence). Ide o umelú inteligenciu, ktorá je navrhnutá na jeden druh úlohy a zvláda len ten. Napríklad vyhrať šachovú partiu, klasifikovať obrázok, nájsť vzor. Silná umelá inteligencia sa potom delí na Artificial General Intelligence (AGI) a Artificial Super Intelligence (ASI). AGI je umelá inteligencia, ktorej inteligencia je rovná ľudskej. Teda je si vedomá svojho bytia, zvláda riešiť problémy, učiť sa a plánovať do budúcnosti. ASI je umelá inteligencia, ktorá by mala presahovať ľudské možnosti. Silná umelá inteligencia zatiaľ neexistuje v praxi.

Strojové učenie Je súčasťou umelej inteligencie. Skúma dáta a algoritmy snažiace sa imitovať proces učenia u človeka, čím postupne zlepšujú svoju presnosť. Prístupom k učeniu v strojovom učení sa venujem viac v kapitole 2.1

Hlboké učenie Pojem hlboké sa väčšinou spája s počtom vrstiev. Obvykle sa za hlboké neurónové siete považujú tie, ktoré majú 3 a viac vrstiev, okrem vstupnej a výstupnej. Je to časť strojového učenia. Využíva algoritmy, ktoré nepotrebujú tak veľký zásah ľudí do procesu učenia. V klasickom strojovom učení musia dáta analyzovať a pripravovať ľudia, často experti v danej oblasti. Je potrebné určiť, ktoré časti dát sú podstatné a do akej miery sú podstatné pre konkrétnu úlohu. V hlbokom učení táto analýza dát nie je potrebná, algoritmy hlbokého učenia si poradia aj s neštrukturovanými a neupravenými dátami. V týchto dátach potom vedia rozlýšiť podstatné časti a určiť ich dôležitosť pre danú úlohu.

Neurónové siete Konkrétne algoritmy hlbokého učenia. Je to matematický model založený na fungovaní reálnych neurónov v ľudskom mozgu. Neurónové siete popisujem v kapitole 2.2

2.1 Prístupy k učeniu

V strojovom učení poznáme tri prístupy k učeniu.

Učenie s učiteľom (Supervised Learning) Ide o proces, kedy máme štrukturované dáta a poznáme k nim očakávané výstupy. Daný algoritmus sa potom na základe týchto dát naučí vykonávať nejakú úlohu. Napríklad máme obrázky psov a mačiek. U každého obrázku je určené či je na ňom mačka alebo pes. Na základe týchto obrázkov sa algoritmus naučí rozpoznať či je na obrázku pes alebo mačka aj pre nové, predtým nevidené obrázky.

Učenie bez učiteľa (Unsupervised Learning) V tomto prípade sú tréningové dáta bez štruktúry a nepoznáme k nim očakávané výstupy. Algoritmus v týchto dátach sám nájde určité vzory, na základe ktorých potom vykoná nejakú úlohu. Dáta napríklad spája do skupín na základe nejakej vlastnosti v dátach. Niekedy sa používa aj kombinácia tohto prístupu s učením s učiteľom.

Učenie formou odmeny a trestu (Reinforcement Learning) Vstupné dáta sú neštrukturované a nepoznáme k nim očakávané výstupy. Tu algoritmus pozoruje svoje prostredie. Robí nejaké náhodné akcie a pozoruje ich výsledok. Pokiaľ boli správne, dostane odmenu a zapamätá si, že robil niečo správne. Tento proces pokračuje, až kým sa algoritmus nespráva podľa očakávaní.

Pri tréningu používame tri množiny dát. *Tréningové dáta* sú tie, ktoré sa používajú na trénovanie modelov. Na základe týchto dát si model prispôsobí všetky svoje parametre. *Validačné dáta* sa používajú ešte počas tréningu. Priebežne sa nimi hodnotí kvalita modelu. Aj napriek tomu, že model tieto dáta vidí už vo fáze tréningu, neučí sa na nich. *Testovacie dáta* sú poslednou skupinou. Tieto dáta sa používajú na konečné zhodnotenie modelu. Používajú sa až keď je tréning ukončený. Mali by pokrývať čo najviac prípadov, s ktorými sa bude model stretávať. Celkové dáta, ktoré máme k dispozícií na natrénovanie nejakého modelu, sa väčšinou rozdeľujú na 95% tréningové, 2,5% validačné, 2,5% testovacie [5].

2.2 Neurónové siete

Definícia 1 (Neurňová sieť). Neurónová sieť je matematická funkcia [5] v tvare

$$y = f_{NN}(x).$$

Definícia 2 (Viacvrstvová neurňová sieť). Pokiaľ hovoríme o viacvrstvových neurónových sieťach, ide o vnorené funkcie. Napríklad neurónová sieť s tromi vrstvami by vyzerala nasledovne:

$$y = f_{NN}(x) = f_3(f_2(f_1(x))),$$

kde f_1 a f_2 sú vektorové funkcie (ich výsledkom je vektor) v tvare

$$f_l(z) = g_l(W_l z + b_l),$$

kde l je index vrstvy, g_l je aktivačná funkcia, W_l(matica) a b_l(vektor) sú parametre, ktoré sú naučené pre každú vrstvu zvlášť počas tréningu. f_3 je potom funkcia podľa potreby, teda môže byť skalárna (jej výsledkom je skalár) alebo vektorová [5].

Príklad 3 (Priechod funkciou f_l). Uvažujme vektor z ako vstupný vektor do funkcie f_l. W_l je matica, ktorá má $size_l$ riadkov, teda toľko, koľko má

daná vrstva neurónov. Každý riadok matice W_l je vektor $w_{l,n}$ (n ako neurón), s rovnakou dimenziou ako vektor z. Pre každý neurón sa vypočíta vektor

$$a_{l,n} = w_{l,n}z + b_l.$$

Výsledkom bude vektor $[g_l(a_{l,1}), g_l(a_{l,2}), \ldots, g_l(a_{l,size_l})]$ [5].

Definícia 4 (Aktivačná funkcia). Je nejaká nelineárna funkcia g_l, kde l je index vrstvy. Je zvykom, že sa v celej architektúre používa rovnaká aktivačná funkcia, nie je to však pravidlo. Aktivačná funkcia musí byť diferencovateľná aspoň na väčšine definičného oboru (teda musí mať deriváciu na väčšine svôjho definičného oboru). Bežne používané sú **TanH** (hyperbolický tangens) a **ReLU** (recitified linear unit function).

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{1}$$

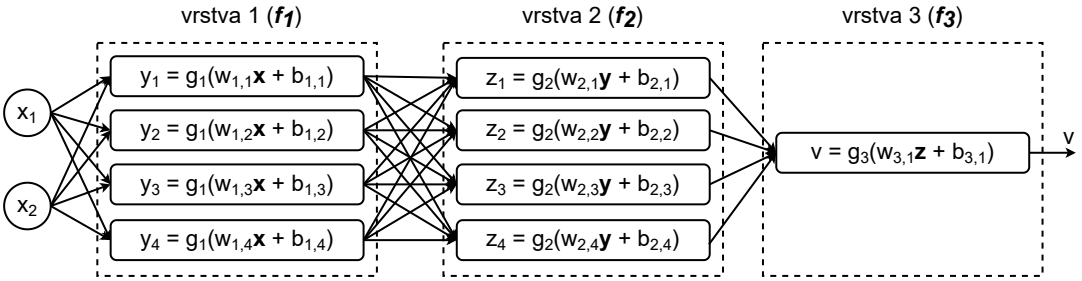
$$relu(z) = \begin{cases} 0, & \text{ak z} < 0 \\ z, & \text{inak} \end{cases} \tag{2}$$

V moderných návrhoch sa častejšie používa práve ReLU, pretože funkcia TanH má problém so saturáciou. Teda zmenšuje obor hodnôt na interval $[-1, 1]$. Používaním aktivačných funkcií, ktoré nesaturujú, je výpočetný proces o niečo rýchlejší. [4]

Definícia 5 (Dopredná neurónová sieť). Alebo Feed-forward neural network (FFNN) je špeciálny typ neurónovej siete, kedy spojenia medzi neurónmi nevytvoria cyklus. To znamená, že dáta idú len jedným smerom a každým neurónom prejdú len raz.

Neurónové siete sú reprezentované graficky pomocou takzvaných neurónov logicky usporiadaných do vrstiev. Neuróny sú reprezentované pomocou kruhu alebo obdĺžniku. V ňom je napísaná nejaká matematická operácia. Šípky prichádzajúce do neurónu reprezentujú vstup, naopak šípky vychádzajúce z neurónu reprezentujú výstup. Výstupom z neurónu je výsledok aplikácie matematickej operácie napísanej v neuróne na vstup neurónu. Neuróny v stĺpci reprezentujú vrstvu.

Ako príklad FFNN použijem architektúru nazývanú Multilayer perceptron (MLP) (obrázok 1). Uvažujme MLP s tromi vrstvami. Vstup bude dvojdimenzionálny vektor x, výstup bude číslo v. Každý neurón berie ako vstup výsledok všetkých neurónov z predchádzajúcej vrstvy. Z týchto čísel si neurón vytvorí vektor, a ten použije ako vstup pre svoju funkciu. Posledná vrstva má z pravidla jeden neurón. V modely na obrázku 1 berie každý neurón ako svoj vstup výstup každého neurónu z predchádzajúcej vrstvy. Takejto architektúre sa hovorí plne spojená (fully-connected alebo FC). V danej architektúre môžu byť plne spojené len niektoré vrstvy, iné môžu byť zasa spojené iným spôsobom.



Obr. 1: Multilayer perceptron so vstupným dvojdimenzionálnym vektorom x, dvomi vrstvami so štyrmi neurónmi a jednou výstupnou vrstvou s jedným neurónom. [5]

2.3 Konvolučné neurónové siete (CNN)

Konvolučná neurónová sieť alebo convolutional neural network (CNN), je špeciálny typ FFNN. V práci sa venujem modelom CNN, ktoré rozpoznávajú obraz, preto vysvetlím fungovanie CNN práve na tomto príklade.

Základnou myšlienkou je, že v obrázku, pixely, ktoré sú blízko pri sebe zvyčajne reprezentujú rovnakú informáciu, napríklad obloha, budova, človek, a podobne. Výnimkou sú práve hrany, časti obrázka, kedy prechádzame z jedného typu informácie na iný. Pokiaľ by sme dokázali naučiť neurónovú sieť rozpoznávať hrany a regióny s rovnakou informáciou, dokázala by určiť, čo sa na obrázku nachádza.

Vzhľadom na to, že väčšinou sa objekty na obrázku nachádzajú len na nejakej jeho časti, môžeme rozdeliť daný obrázok na menšie časti. To robíme pomocou techniky hýbajúceho sa okna. Môžeme teda natrénovať mnoho menších podmodelov, ktoré sú súčasťou jedného modelu, kedy každý podmodel bude rozpoznávať určitý typ informácie na obrázku. Jeden bude rozpoznávať oblohu, druhý budovy, tretí človeka a podobne.

Na rozpoznanie nejakého vzoru, sa musia podmodely naučiť parametre matice $\boldsymbol{F}$ (ako filter) veľkosti $p \times p$, kde p je veľkosť hýbajúceho sa okna. Predstavme si, že máme len čiernobiele obrázky, kde 0 reprezentuje biele pixely a 1 reprezentuje čierne. Uvažujme $p = 3$ a nasledujúcu časť obrázka:

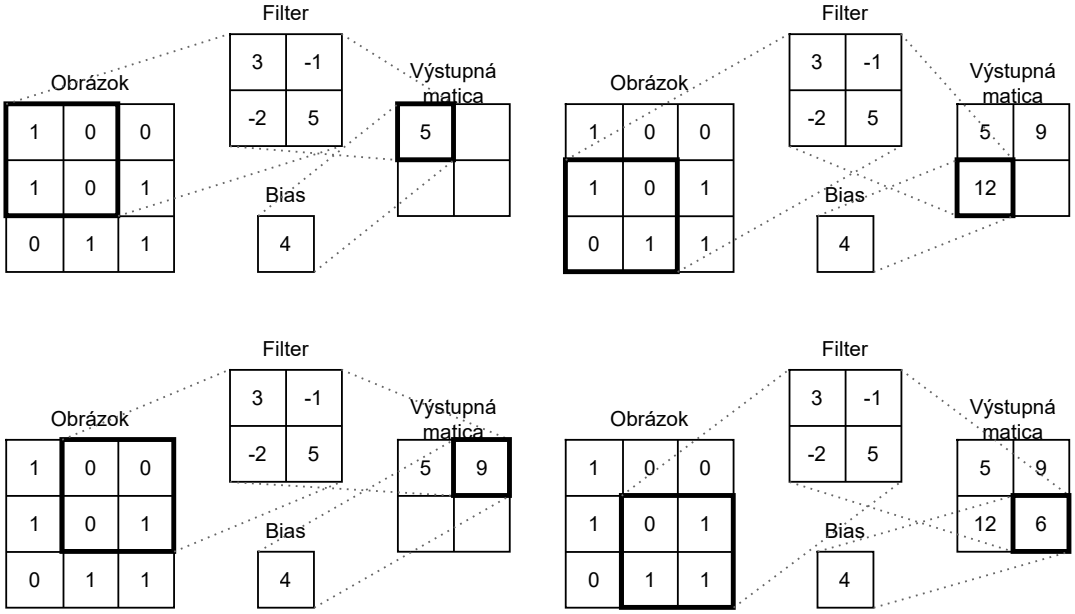
$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Ďalej uvažujme, že filter F má naučené parametre následovne

$$F = \begin{pmatrix} 4 & 2 & 0 \\ 0 & 4 & 1 \\ 0 & 1 & 7 \end{pmatrix}.$$

Spočítame konvolúciu týchto dvoch matíc, teda dostaneme

$$1 \cdot 4 + 0 \cdot 2 + 0 \cdot 0 + 0 \cdot 0 + 1 \cdot 4 + 0 \cdot 1 + 0 \cdot 0 + 0 \cdot 1 + 1 \cdot 7 = 15$$



Obr. 2: Príklad konvolúcie s filtrom veľkosti 2×2 a s krokom 2.

Čím väčšie číslo výjde ako výsledok, tým je zhoda väčšia. Predstavme si, že ďalšia časť obrázku je

$$P' = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

V tomto prípade by konvolúcia s filtrom F vyšla 5.

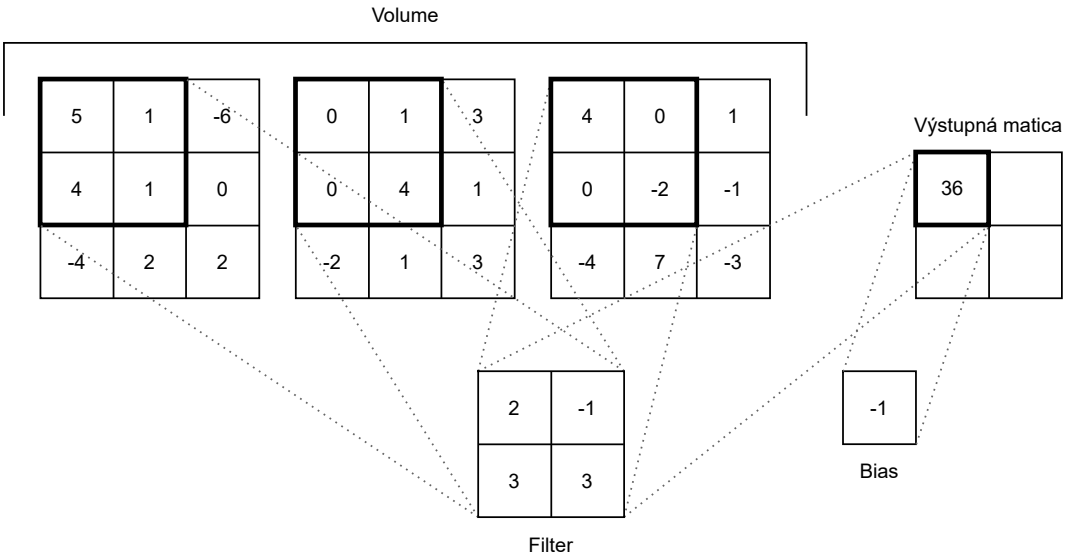
Ku každému filtru máme takisto číslo b (bias), ktoré je pripočítané ku výsledku konvolúcie ešte pred aplikovaním aktivačnej funkcie.

Každá vrstva CNN teda pozostáva z niekoľkých filtrov (každý s vlastnými parametrami a vlastným parametrom b) a každý filter prejde celý obrázok zľava doprava, zhora dole (obrázok 2).

Keďže pre každú vrstvu l môžeme mať až $size_l$ filtrov, výstup z danej vrstvy je kolekcia $size_l$ matíc. Takúto kolekciu nazývame *volume*. Veľkosť tejto kolekcie nazývame *hĺbka volume*. Každý filter vrstvy prechádza cez celý volume a výsledok konvolúcie je súčet výsledkov konvolúcií pre všetky matice vo volume (obrázok 3). Ako výsledok prvého proku konvolúcie teda dostaneme

$$5 \cdot 2 + 1 \cdot (-1) + 4 \cdot 3 + 1 \cdot 3) + (0 \cdot 2 + 1 \cdot (-1) + 0 \cdot 3 + 4 \cdot 3) + (4 \cdot 2 + \\ + 0 \cdot (-1) + 0 \cdot 3 + (-2) \cdot 3) + (-1) = 36$$

V praxi máme už vstupné dáta ako volume, pretože farebné obrázky sú reprezentované v troch kanáloch R, G a B.



Obr. 3: Príklad konvolúcie s filtrom veľkosti 2×2 aplikovanej na volume s hĺbkou 3.

Predtým, než budem pokračovať konkrétnymi architektúrami konvolučných neurónových sietí, musím zadefinovať pár dôležitých pojmov.

Stride Alebo krok, je veľkosť kroku hýbajúceho sa okna

Padding Pridanie rámiku núl okolo vstupnej matice. Používame v prípade, keď potrebujeme väčšiu výstupnú maticu, alebo nám nesedí stride. Špeciálne sa používa takzvaný *same padding* - aplikuje sa taký padding, aby rozmery výstupu ostali rovnaké ako rozmery vstupu.

Pooling Používame techniku hýbajúceho sa okna ako pre filter, ale namiesto filtra používame nejaký fixný operátor, napríklad funkciu MAX alebo AVERAGE. Pokiaľ aplikujeme pool na volume, tak sa na každú maticu aplikuje zvlášť, teda výsledkom nie je jedna matica ale volume s rovnakou hĺbkou. Zvyšuje presnosť modelu a rýchlosť tréningu, pretože zmenšuje počet parametrov.

Hyper-parameter Je parameter, ktorého hodnota je určená ešte pred začiatkom trénovania.

2.4 Klasifikácia obrazu

Konvolučné neurónové siete sa používajú okrem iného na klasifikovanie obrazu. Ide o priradenie kategórie obrázku. Kategória je určená na základe obsahu obrázku. Môžeme napríklad natrénovať neurónovú sieť aby rozpoznávala, aký šport

hrajú ľudia na obrázku. Povedzme, že náš model siete klasifikuje 10 športov. Pre svoj vstup (teda obrázok) vráti ako výstup ku každému športu číslo, s akou pravdepodobnosťou je daný šport kategória fotografie. To zariadi funkcia *softmax* (3).

Definícia 6 (Softmax). Je vektorová funkcia v tvare

$$\sigma(z) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \tag{3}$$

kde z je vstupný vektor, e je eulerovo číslo, z_i je prvok vstupného vektoru na pozícií i. Prvky vstupného vektoru z sú reálne čísla. Výsledkom funkcie je vektor, ktorého prvky sú čísla z intervalu $(0, 1)$.

Príklad 7 (Softmax). Ako vstupný vektor použijeme

$$z = [6, 0, -4],$$

ďalej vypočítame hodnoty $e^{z_1}, e^{z_2}, e^{z_3}$. Dostaneme

$$e^{z_1} = e^6 \doteq 403,4288;$$

$$e^{z_2} = e^0 = 1,0;$$

$$e^{z_3} = e^{-4} \doteq 0,0183.$$

Vypočítame súčet týchto hodnôt

$$\sum_{j=1}^{K} e^{z_j} = e^{z_1} + e^{z_2} + e^{z_3} \doteq 404,4471.$$

Výsledný vektor bude

$$\sigma(z) = [\frac{403,4288}{404,4471}, \frac{1,0}{404,4471}, \frac{0,0183}{404,4471}] = [0,99748, 0,00247, 0,00005].$$

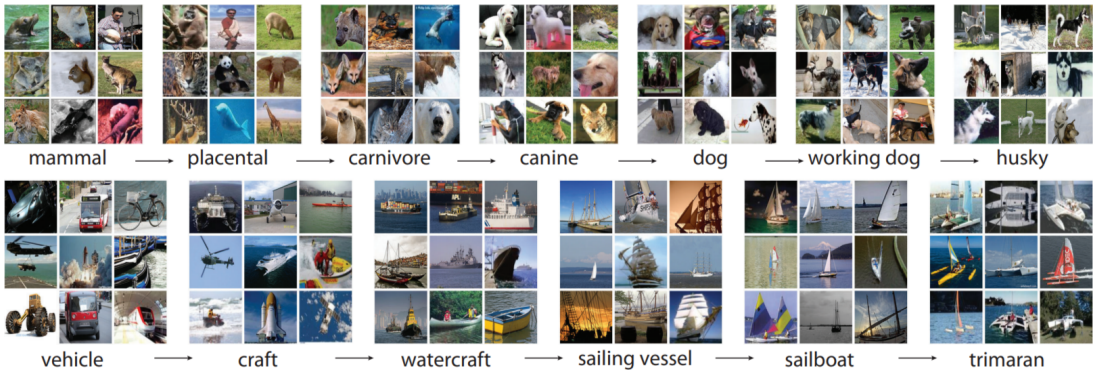
2.5 ImageNet a ILSVRC

Na natrénovanie kvalitného modelu potrebujete čo najviac čo najrôznejších dát. Model, ktorý natrénujeme na 100 obrázkoch, bude dosahovať omnoho horšie výsledky ako model, ktorý bol natrénovaný na 1000000 obrázkoch. So zámerom uľahčiť proces zbierania dát, na trénovanie neurónových sietí, na rôzne úlohy počítačového videnia, vznikla databáza *ImageNet*.

2.5.1 ImageNet

ImageNet je hierarchicky usporiadaná databáza približne 14 miliónov[1] klasifikovaných obrázkov.

[1]Podľa https://www.image-net.org/ v roku 2022



Obr. 4: Príklad vetiev stromovej štruktúry obrázkov v ImageNete. Horná vetva je vetva z podstromu cicavce. Spodná je z podstromu vozidlá. [6]

Obrázky sú hierarchicky usporiadané podľa databáze slov *WordNet*, ktorá je usporiadaná na základe vzťahov slov. WordNet spája slová do takzvaných *synsetov*. Ide o skupiny slov, ktoré sú synonymá. Tieto synsety sú potom usporiadané do stromovej štruktúry na základe logických vzťahou medzi nimi. Príklady vetiev tejto stromovej štruktúry môžete vidieť na obrázku 4.

ImageNet zbiera obrázky z internetu. Na zber obrázkov používa rôzne webové vyhľadávače. Tieto obrázky potom kontrolujú ľudia. Kontrolujú, či je obrázok v dostatočnej kvalite a či je vhodný pre danú kategóriu. Ako kategórie obrázkov sa používajú synsety. Aby bol tento proces spoľahlivejší, tak ľudia tieto úlohy robia cez službu Amazon Mechanical Turk. Ide o službu, kde zadávateľ môže zadať úlohu a používatelia, ktorí ju spracujú, dostanú za to zaplatené.

Klasifikované obrázky z ImageNetu sú voľne dostupné, pre nekomerčné použitie. Pokiaľ teda potrebujete natrénovať neurónovú sieť na rozpoznanie určitých kategórií, nemusíte si zostavovať dataset. Stačí použiť už pripravené obrázky z ImageNetu. To podstatne uľahčí celý proces trénovania.

2.5.2 ILSVRC

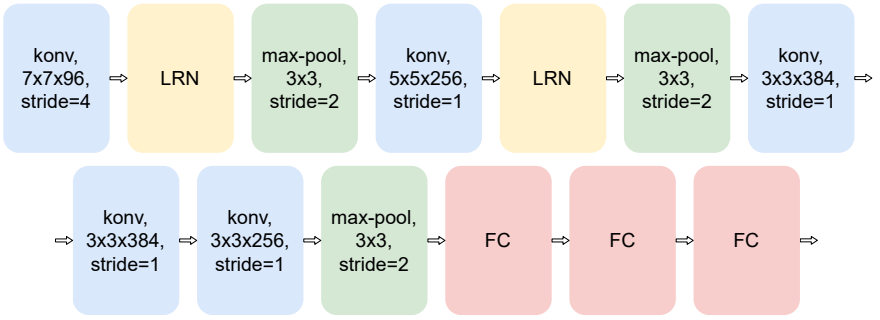
ILSVRC (ImageNet Large Scale Visual Recognition Challenge) je súťaž, ktorá sa konala medzi rokmi 2010-2017. Súťažilo sa v rôznych disciplínach od klasifikácie obrazu až po lokalizáciu objektov na videu. Pre moju prácu, je podstatná klasifikácia obrazu.

Pre súťaž bolo vybratých 1000 kategórií z databáze ImageNet. Boli vyberané tak, aby nebola nejaká kategória potomkom druhej v stromovej štruktúre ImageNetu. Napríklad, ak by sme vybrali kategóriu *dog* (z príkladu na obrázku 4), nemohli by sme vybrať kategóriu *husky*. Kategórie boli vyberané tak, aby čo najviac overili schopnosti súťažiacich algoritmov. Boli vyberané čo najkonkrétnejšie kategórie ako napríklad plemená psov.

Pre tieto kategórie bol vytvorený unikátny dataset obrázkov, ktoré nie sú súčasťou ImageNetu. Boli zozbierané rovnakým spôsobom ako v ImageNete.

Súťažiaci algoritmus v tejto úlohe dostane ako vstup testovaciu sadu obráz-

kov. Pre každý obrázok vráti zoznam kategórii, ktoré sú na obrázku. Každý obrázok má priradenú len jednu kategóriu aj v prípade, že je na jednom obrázku viac rôznych objektov. Pokiaľ je na obrázku človek sediaci za stolom, ako kategória je určený stôl. Algoritmus potom nemusí vedieť ktorú z kategóriu má určiť ako tú hlavnú. Z toho dôvodu algoritmus vráti až 5 kategórií pre každý obrázok. Ak sa medzi nimi nachádza kategória považovaná za tú správnu, odpoveď je braná ako správna. Tomuto prístupu k presnosti algoritmov sa hovorí *top-5 chyba*. Všetky informácie o súťaži a ostatných disciplínach sú v [1].



Obr. 5: Architektúra AlexNet v grafickej podobe, konv je konvolučná vrstva, LRN je Local Response Normalization vrstva a FC sú fully-connected (plne spojené) vrstvy.

vrstva	typ vrstvy	veľkosť filtra	krok	počet neurónov
1	konv	11×11	4	96
2	LRN	-	-	-
3	max-pool	3×3	2	-
4	konv	5×5	1	256
5	LRN	-	-	-
6	max-pool	3×3	2	-
7	konv	3×3	1	384
8	konv	3×3	1	384
9	konv	3×3	1	256
10	max-pool	3×3	2	-
11	FC	-	-	4096
12	FC	-	-	4096
13	FC	-	-	1000

Tabuľka 1: Architektúra AlexNet, kde konv je konvolučná vrstva, LRN je Local Response Normalization a FC sú fully-connected (plne spojené) vrstvy.

3 Použité modely CNN

Použité modely som vyberal podľa súťaže ILSVRC . Sú to výhercovia alebo úspešné modely niektorých rokov. Vybrané modely boli výhercovia v rokoch 2012 (AlexNet), 2014 (GoogLeNet), 2015 (ResNet), druhý v poradí v roku 2014 (VGG) a druhý v poradí v roku 2016 (ResNeXt). Každý model použil nejaký nový koncept alebo vylepšil predošlý a vďaka tomu postupne dosahovali rok čo rok lepšie výsledky.

3.1 AlexNet

Predtým, než sa dostanem k samotnej architektúre, spomeniem dve dôležité metódy, ktoré vo svojom návrhu AlexNet používa. Ide o *Local Response Normalization* a *prekrývajúci sa pooling*.

Local response normalization Simuluje takzvanú *laterálnu inhibíciu*, ktorá funguje v reálnych neurónoch. Jej cieľom je potlačiť odozvu susedných neurónov na vnem a tým vytvoriť lepšiu odozvu aktuálneho neurónu na vnem. Funkcia používaná v CNN vyzerá následovne

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}, \tag{4}$$

kde $a^i_{x,y}$ je aktivita neurónu vypočítaná použitím filtra i na pozícií (x, y) a aplikovaním aktivačnej funkcie. Suma prechádza cez n susedných neurónov, spočíta ich aktivitu a tieto aktivity sčíta. N je celkový počet neurónov. k, n, α, β sú hyperparametre, použité hodnoty sú $k = 2$, $n = 5$, $\alpha = 10^{-4}$ a $\beta = 0,75$. $b^i_{x,y}$ je potom normalizovaná aktivita neurónu.

Aj napriek tomu, že ReLU (2) nemá problém so saturáciou, a teda nie je potrebná normalizácia, aj tak táto normalizácia pomáha pri generalizácií (schopnosti presnejšie určovať nové, predtým nevidené dáta) [4].

Prekrývajúci sa pooling P ovedzme, že určíme veľkosť okna ako $z \times z$. Tradične sa krok určí ako $s = z$. V prípade prekrývajúceho sa poolingu určíme krok $s < z$. V tomto konkrétnom prípade, je $z = 3$ a $s = 2$. Táto úprava mierne zlepšuje výsledky (zmenšuje top-5 chybu o 0,3%) [4].

AlexNet sa skladá z ôsmych vrstiev s parametrami, z nich je 5 konvolučných a 3 sú plne spojené. Po prvej a druhej konvolučnej vrstve nasledujú response-normalization vrstvy 4. Po oboch normalizačných a po piatej konvolučnej nasleduje prekrývajúca sa max-pooling vrsva. Architektúra je prehľadne zobrazená v tabuľke 1 a na obrázku 5. ReLU (2) je aplikované na výstup každej konvolučnej a plne spojenej vrstvy. Výstup poslednej plne spojenej vrstvy sa použije ako vstup do funkcie softmax (3). Tento výsledok reprezentuje pravdepodobnosť s akou priradzuje kategóriu danému obrázku [4].

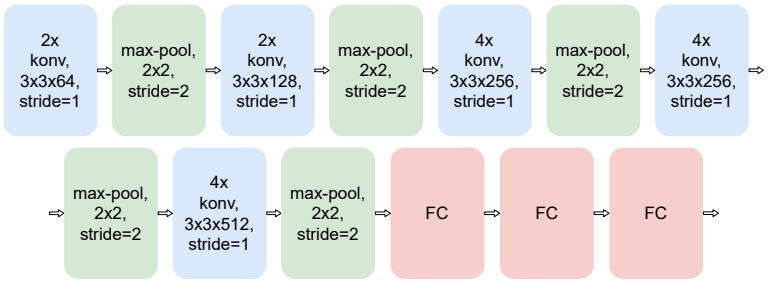
Vstupom je $227 \times 227 \times 3$ (trojkanálový, teda R, G, B) obrázok.

3.2 VGG

Pri VGG zvolili tvorcovia odlišný prístup ako ich predchodcovia, kedy namiesto jednej vrstvy s väčšími filtrami používajú viac vrstiev s menšími filtrami. Použitím dvoch vrstiev s filtrami veľkosti 3×3 nám dá podobné výsledky ako použitie jednej vrstvy s filtrami veľkosti 5×5. Pokiaľ ich použijeme tri, dávajú podobné výsledky ako 7×7. Toto prináša hneď dve výhody. Viackrát sa aplikuje ReLU (2) a zmenší sa počet parametrov.

Príklad 8 (Počet parametrov). Vrstva s filtrami veľkosti 7×7 potrebuje

$$7^2 n^2 = 49 n^2$$



Obr. 6: Architektúra VGG v grafickej podobe, konv je konvolučná vrstva, FC sú fully-connected (plne spojené) vrstvy. Pri konvolučných vrstvách je navyše uvedené, koľkokrát za sebou ide vsrstva s rovnakým zložením.

vrstva	**typ vrstvy**	**veľkosť filtra**	**krok**	**počet neurónov**
1	konv	3×3	1	64
2	konv	3×3	1	64
3	max-pool	2×2	2	-
4	konv	3×3	1	128
5	konv	3×3	1	128
6	max-pool	2×2	2	-
7	konv	3×3	1	256
8	konv	3×3	1	256
9	konv	3×3	1	256
10	konv	3×3	1	256
11	max-pool	2×2	2	-
12	konv	3×3	1	512
13	konv	3×3	1	512
14	konv	3×3	1	512
15	konv	3×3	1	512
16	max-pool	2×2	2	-
17	konv	3×3	1	512
18	konv	3×3	1	512
19	konv	3×3	1	512
20	konv	3×3	1	512
11	max-pool	2×2	2	-
22	FC	-	-	4096
23	FC	-	-	4096
24	FC	-	-	1000

Tabuľka 2: Architektúra VGG

parametrov, zatiaľ čo 3 vrstvy s filtrami veľkosti 3×3 potrebujú

$$3(3^2n^2) = 27n^2$$

parametrov, kde n je počet filtrov v danej vrstve. Dosahujú podobné výsledky, ale filtre veľkosti 3×3 potrebuje polovičný počet parametrov ako filter 7×7.

Samotná architektúra, teda pozostáva len z vrstiev s filtrami veľkosti 3×3, pre každú konvolučnú vrstvu je krok 1 a padding 1. Max-pool vrstvy sú všetky 2×2 s krokom 2. Na výstupy všetkých konvolučných vrstiev je použítá aktivačná funkcia ReLU (2). Architektúra je prehľadne zobrazená na obrázku 6 a v tabuľke 2.

3.3 GoogLeNet

Tvorcovia GoogLeNet zvolili vo svojej architekrúre prístup takzvaných *inception modulov*. Sú to opakujúce sa "stavebné bloky", zložené z viacerých vrstiev. Na svoj vstup použijú konvolúcie 1×1, 3×3, 5×5 a 3×3 max-pooling. Výsledné volume každej vrstvy potom spoja do jedného volume (obrázok 7). Používajú sa vylepšené moduly s redukciou, ktoré predtým, ako aplikujú 5×5 alebo 3×3 konvolúcie zmenšia hĺbku volume pomocou 1×1 konvolúcií a hĺbku výsledného volume z max-pooling vrstiev takisto zredukujú pomocou 1×1 konvolúcií (obrázok 7). To zmenšuje počet operácií násobnia.

Príklad 9 (Počet operácií násobenia). Počet operácií násobenia vypočítame ako

$$(veľkosť\ výstupu) \cdot (hĺbka\ výstupu) \cdot (veľkosť\ filtra) \cdot (hĺbka\ vstupu).$$

Pokiaľ by sme aplikovali konvolučnú vrstvu $5 \times 5 \times 32$ na vstup veľkosti $28 \times 28 \times 192$ musíme použiť

$$28 \cdot 28 \cdot 32 \cdot 5 \cdot 5 \cdot 192 \doteq 120mil$$

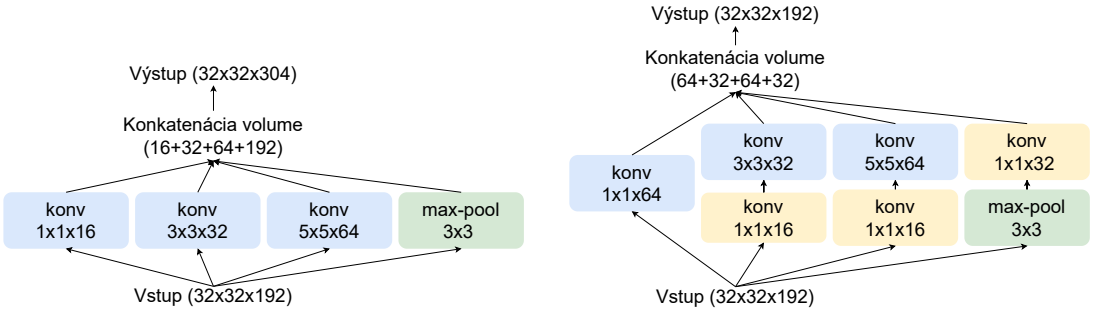
operácií. Pokiaľ predtým ale zmenšíme hĺbu aplikovaním 1×1 filtrov, radikálne sa zmenší počet vykonaných operácií násobenia. Povedzme, že použijeme $1 \times 1 \times 16$. Dostaneme

$$28 \cdot 28 \cdot 16 \cdot 1 \cdot 1 \cdot 192 + 28 \cdot 28 \cdot 32 \cdot 5 \cdot 5 \cdot 16 \doteq 12mil$$

operácií. Tým sme zmenšili počet operácií násobenia desaťnásobne.[2]

GoogLeNet sa skladá hlavne z inception modulov. Všetky použité inception moduly používajú redukciu hĺbky volume. Všetky vrstvy, ktoré sú súčasťou inception modulov používajú krok 1 a same padding. Po pool vrstvách, ktoré sú súčasťou inception modulov sa aplikuje 1×1 redukcia. Architektúra je popísaná v tabuľke 3.

[2]Príklad z: https://towardsdatascience.com/deep-learning-understand-the-inception-module-56146866e652



(a) Inception modul bez redukcie. (b) Inception modul s redukciou.

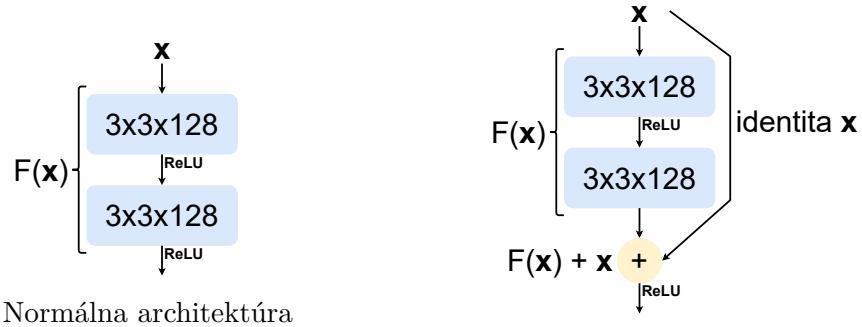
Obr. 7: Ukážka inception modulu bez redukcie aj s redukciou [8].

typ	**veľkosť filtra**	**krok**	**hĺbka**	$\mathbf{1 \times 1}$	$\mathbf{3 \times 3}$ **reduce**	$\mathbf{3 \times 3}$	$\mathbf{5 \times 5}$ **reduce**	$\mathbf{5 \times 5}$	**pool**
konv	7×7	2	1	-	-	-	-	-	-
max-pool	3×3	2	0	-	-	-	-	-	-
konv	3×3	1	2	-	64	192	-	-	-
max-pool	3×3	2	0	-	-	-	-	-	-
inception	-	-	2	64	96	128	16	32	32
inception	-	-	2	128	128	192	32	96	64
max-pool	3×3	2	0	-	-	-	-	-	-
inception	-	-	2	192	96	208	16	48	64
inception	-	-	2	160	112	224	24	64	64
inception	-	-	2	128	128	256	24	64	64
inception	-	-	2	112	144	288	32	64	64
inception	-	-	2	256	160	320	32	128	128
max-pool	3×3	2	0	-	-	-	-	-	-
inception	-	-	2	256	160	320	32	128	128
inception	-	-	2	384	192	384	48	128	128
avg-pool	7×7	1	0	-	-	-	-	-	-
FC	-	-	-	-	-	-	-	-	-
FC	-	-	-	-	-	-	-	-	-
FC	-	-	-	-	-	-	-	-	-

Tabuľka 3: Architekrúra GoogLeNet. Stĺpce s **reduce** značia počet neurónov s filtrom veľkosti 1×1, ktoré redukujú hĺbky volume.

typ vrstvy	veľkosť filtra	krok	počet neurónov	residua1
konv	7×7	2	64	-
max-pool	3×3	2	-	-
residual	-	-	-	$\begin{bmatrix} 1 \times 1 \times 64 \\ 3 \times 3 \times 64 \\ 1 \times 1 \times 256 \end{bmatrix} \times 3$
residual	-	-	-	$\begin{bmatrix} 1 \times 1 \times 128 \\ 3 \times 3 \times 128 \\ 1 \times 1 \times 512 \end{bmatrix} \times 8$
residual	-	-	-	$\begin{bmatrix} 1 \times 1 \times 256 \\ 3 \times 3 \times 256 \\ 1 \times 1 \times 1024 \end{bmatrix} \times 36$
residual	-	-	-	$\begin{bmatrix} 1 \times 1 \times 512 \\ 3 \times 3 \times 512 \\ 1 \times 1 \times 2048 \end{bmatrix} \times 3$
avg-pool	-	-	-	-
FC	-	-	-	-
softmax	-	-	-	-

Tabuľka 4: Architektúra ResNet



(a) Normálna architektúra

(b) Residual block

Obr. 8: Residual Block

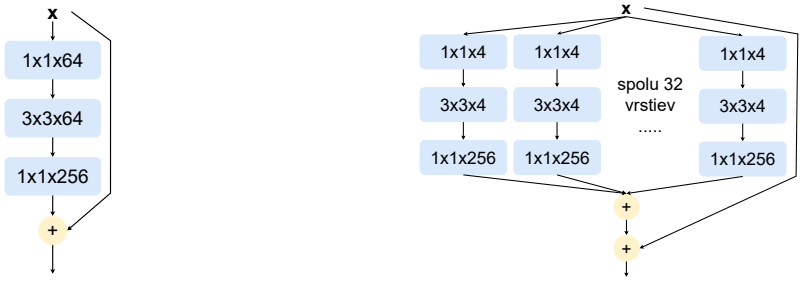
3.4 ResNet

Dalo by sa očakávať, že čím viac má konvolučná neurónová sieť vrstiev, tým lepšie výsledky bude dosahovať. V praxi sa ale ukázalo, že pokiaľ pridáme vrstiev príliš veľa (viac ako 50), tak daný model dosahuje horšie výsledky. Tento problém sa podarilo vyriešiť pomocou takzvaných *residual blokov*. V ňom sa používajú skratkové spojenia, teda vrstva nemusí byť spojená priamo z následujúcou vrstvou. Tieto skratky sa v residual blokoch používajú na prenos identity, ktorá je potom pripočítaná k výstupu preskočených vrstiev (obrázok 8b). Tento

prístup umožňuje natrénovať modely so 100 a vaic vrstvami tak, aby dosahovali dobré výsledky.

3.5 ResNeXt

Architektúra ResNeXt spája dve dôležité metódy pri návrhu modelu konvolučnej neurónovej siete. Okrem používania residual blokov, ako ResNet 3.4, rastie do hĺbky podobným spôsobom ako inception moduly v GoogLeNet 3.3. Konvolučné vrstvy v residual blokoch rozdelí na viac vrstiev s menším počtom filtrov. Výsledné volume vrtiev sčíta. K tomuto výsledku potom ešte pripočíta identitu vstupu ako v residual blokoch (obrázok 9b). Spojenie týchto dvoch metód prispelo k zjednodušeniu modelu, ktorý dosahuje lepšie výsledky ako jeho predchodcovia. Napríklad 101-vrstvový ResNeXt dosahuje o niečo lepšie výsledky ako 200-vrstvový ResNet. Pritom je menej zložitý.



(a) Residual block

(b) Resnext block

Obr. 9: Porovnanie residual blocku a bloku architektúry ResNeXt.

kategória	včela	brokolica	sviečka	gitara	kôň	reťaz	lienka	lama	páv	zajac	okuliare	tiger	zebra
počet obrázkov	1	2	3	2	1	2	3	3	3	1	2	1	2

Tabuľka 5: Kategórie obrázkov datasetu a počty fotografií v každej kategórií.

včela gitara lienka lama páv

Obr. 10: Príklad originálnych obrázkov z datasetu bez aplikovaných transformácií.

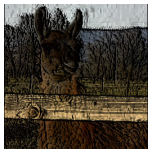
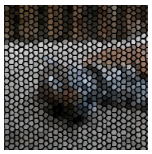
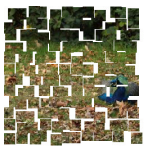
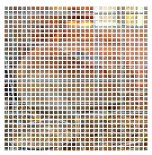
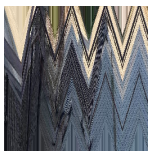
4 Experimenty

Na zistenie, ako vplývajú transformácie na správnosť klasifikácie obrázkov u ľudí a ako u konvolučných neurónových sietí, som spravil dva experimenty. V jednom som zisťoval vplyv transformácií na modely konvolučných neurónových sietí natrénovaných na ImageNete. V druhom, som prostredníctvom dotazníkov overoval vplyv transformácií u ľudí.

4.1 Dataset

Pre obidva experimenty som použil rovnakú sadu obrázkov. Kategórie obrázkov som vyberal z 1000 kategórií súťaže ILSVRC. Predtrénované modely, ktoré sú dostupné v knižnici PyTorch, sú trénované práve na datasete zo súťaže ILSVRC. Tento dataset však obsahuje veľmi konkrétne kategórie, väčšinou zo zvieracej ríše. Nemohol som teda použiť napríklad kategóriu *pes*, pretože miesto nej sú súčasťou datasetu konkrétne plemená. Kategórie som teda vyberal čo najvšeobecnejšie, aby pre rozpoznanie originálneho obrázka nebola potrebná žiadna odborná znalosť. Pokiaľ by som používal obrázky z internetu, nemal by som záruku, že nie sú súčasťou trénovacích dát. ImageNet je zostavený práve z obrázkov z internetu za použitia rôznych webových vyhľadávačov. Výber kategórií som prispôsobil aj mojim možnostiam vyfotiť danú kategóriu. Vybral som 13 kategórií. Celkový počet obrázkov je 26, pričom ku každej kategórií som odfotil 1-3 obrázky. Kategórie sú *včela, brokolica, sviečka, gitara, kôň, reťaz, lienka, lama, páv, zajac, okuliare, tiger, zebra*. Prehľad kategórií a počtov obrázkov k danej kategórií je v tabuľke 5. Obrázky sú zámerne také, aby niektoré objekty boli buď viac priblížené alebo menšie a nie v strede obrázka (obrázok 10).

Pri výbere transformácií som sa zameriaval hlavne na tie, ktoré nejakým spôsobom upravujú hrany objektov. Teda ich prerušujú alebo menia ich tvar. Používam aj transformácie, ktoré menia farby objektov, tie ale menej. Už od architektúry AlexNet je bežné, že súčasťou algoritmu, je manipulácia s tréningo-

transformácia	popis transformácie	príklad
contrast	Kontrast nastavený na nízke hodnoty. Tým pádom nie sú prechody medzi farbami tak výrazné.	
emboss	Vytvára pečiatku. Svetlejšie miesta ”zdvihne” a tmavšie ”prehĺbi”.	
engrave	Obrátok sa skladá len z čiernych čiar na bielom pozadí. Vyzerá ako staré ilustrácie v knihách.	
hurl	Pridanie farebného šumu.	
lensblur	Rozostrenie, simuluje slabo vidiaceho človeka.	
mosaic	Vytvorenie mozaiky z obrázka.	
oilify	Simulovanie olejomaľby.	
papertile	Rozdelenie na obdĺžniky rôznej veľkosti a náhodne posunuté nejakým smerom.	
pixelize	Rozpixelovanie obrázka a zmenšenie pixelov. Simuluje preloženie bielou mriežkou.	
ripple	Rozvlnenie obrazu.	

Tabuľka 6: Transformácie použité v experimente.

vými dátami. Rôznym spôsobom sa transformujú aj farby, preto sú tieto siete už natrénované tak, aby si do určitej miery poradili so zmenenými farbami. Transformácie som aplikoval pomocou softvéru GIMP. Použil som plugin BIMP (Batch Image Manipulation Plugin), ktorý aplikuje určitú transformáciu na celú zložku obrázkov. Výber transformácií som prispôsobil ponuke tohoto pluginu. Transformácie budem nazývať práve menami transformácií v tomto plugine. Vybrané transformácie sú *contrast, emboss, engrave, hurl, lansblur, mosaic, oilify, papertile, pixelize, ripple*. Transformácie sú popísané v tabuľke 6.

4.2 Experiment s CNN

V tomto experimente som skúmal vplyv transformácií na úspešnosť vybraných modelov konvolučných neurónových sietí. Na prácu s modelmi som použil knižnicu jazyka Python PyTorch.

4.2.1 PyTorch

Súčasťou tejto knižnice je modul torchvision.models, ktorý obsahuje niektoré modely konvolučných neurónových sietí. Poskytuje možnosť stiahnúť už natrénované modely na datasete používanom v ILSVRC. Stiahnutie modelov je zobrazené v ukážke 1.

```
1 import torchvision.models as models
2
3 alexnet = models.alexnet(pretrained=True)
4 vgg = models.vgg19_bn(pretrained=True)
5 googlenet = models.googlenet(pretrained=True)
6 resnet = models.resnet152(pretrained=True)
7 resnext = models.resnext101_32x8d(pretrained=True)
```

Zdrojový kód 1: Stiahnutie natrénovaných modelov.

Knižnica obsahuje aj modul transforms, ktorý poskytuje funkcie na prevod obrázka do vhodnej podoby na vyhodnotenie. Táto podoba je daná v dokumentácií. Obrázok musí byť prevedený na tensory a následne normalizovaný (zdrojový kód 2). Parametre normalizácie sú takisto určené v dokumentácií.

```
from torchvision import transforms

transform = transforms.Compose([
   transforms.CenterCrop(227), # Vyreže oblast' 227x227 px od stredu
   transforms.ToTensor(), # Prevedie obrázok na dátový typ Tensor
   transforms.Normalize( # Normalizuje obrázok
      mean=[0.485, 0.456, 0.406], # Hodnota z~dokumentácie
      std=[0.229, 0.224, 0.225])]) # Hodnota z~dokumentácie
```

Zdrojový kód 2: Úprava obrázkov do vhodnej podoby na vyhodnotenie.

Po vyhodnotení sa na výstup ešte použije funkcia softmax (3). Príklad vyhodnotenia môžete videiť v zdrojovom kóde 3.

```
# Úprava do vhodnej podoby
img_t = transform(photos[img])
batch_t = torch.unsqueeze(img_t, 0)
# Vyhodnotenie
model.eval()
out = model(batch_t)
# Aplikovaný softmax na výstup
percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
```

Zdrojový kód 3: Vyhodnotenie obrázka modelom.

4.2.2 Modely

Architektúry všetkých použitých modelov som popísal v kapitole 3. Tieto modely boli každým rokom vylepšením predošlých modelov. Od roku 2012, kedy sa na klasifikáciu obrazu začali používať konvolučné neurónové siete, nastal viditeľný pokrok v tejto úlohe. AlexNet bola v roku 2012 prelomová architektúra v rozpoznávaní obrazu. V ILSVRC vyhrala prvé miesto s top-5 chybou 15,3% [4]. Druhá najlepšia architektúra mal v tom roku top-5 chybu 26,2% [4]. VGG v roku 2014 obsadilo druhú priečku s top-5 chybou 7,3%. Nestačilo to na prvé miesto, ktoré získala architektúra GoogLeNet s top-5 chybou 6,7% [8]. ResNet bola najúspešnejšia architektúra súťaže ILSVRC v roku 2015. Dosiahla top-5 chybu iba 3,57% [9]. Posledná použitá architektúra ResNeXt stačila v roku 2016 na druhé miesto s top-5 chybou 3,03% [10]. V ďalších častiach skúmam, ako úspešné budú tieto architektúry na transformovaných dátach.

4.2.3 Klasifikácia

Ku klasifikácií obrazu som pristupoval tak ako v súťaži ILSVRC, teda top-5 chybou. Nechal som teda obrázok vyhodnotiť konvolučnou neurónovou sieťou.

Z výsledných kategótií, zoradených zostupne podľa pravdepodobnosti, som porovnal prvých 5 s očakávaným výstupom. Pokiaľ sa medzi nimi nachádzal správny výstup, označil som daný pokus ako úspech. Ak ani jeden z prvých piatich kategórií na výstupe neodpovedal očakávanému výstupu, pokus bol označený ako neúspešný. Pre každú transformáciu, a aj originálne obrázky bez aplikovanej transformácie, som vygeneroval csv súbor. V ňom je pre každú dvojicu model-obrázok hodnota 1, ak model správne klasifikoval daný obrázok s aplikovanou transformáciou. V opačnom prípade je na tejto pozícií 0.

4.2.4 Výsledky

Ako prvé som nechal CNN vyhodnotiť originálne obrázky. Už tu sa stalo, že niektoré obrázky robili problém všetkým architektúram CNN a na základe toho som musel viackrát dataset upravovať.

Medzi finálnymi dátami som ale nechal aj obrázky, u ktorých sa stalo, že ich niektoré architekrúry neklasifikovali správne. Je to z toho dôvodu, že vybrať obrázok, ktorý by rospoznali všetky architektúry nie je jednoduché, hlavne keď je medzi architektúrami taký veľký rozdiel v top-5 chybe. Najproblematickejšou architektúrou bola AlexNet, ktorá si neporadila až so 6 obrázkami, architektúra ResNet neklasifikovala správne 3 obrázky a všetky ostatné architektúry mali problém s dvomi obrázkami. Pri vyhodnocovaní transformovaných obrázkov sa ale ukázalo, že niektoré obrázky boli opakovane klasifikované správne aj napriek tomu, že ich originál nebol klasifikovaný srávne. Toto sa najviackrát podarilo architektúram AlexNet a ResNet, konkrétne päťkrát. Až na jednu výnimku sa tento úkaz vyskytol pri kategórií reťaz. Teda to môže byť spôsobené práve tým, že daná transformácia mohla buď reťaz na obrázku mierne zvýrazniť, alebo vytvoriť inú na inom mieste (napríklad transformácia mosaic vytvorila vzor, ktorý mohol pripomínať reťaz). Spomínaná výnimka bola pri kategórií lienka(transformácia mosaic).

Pri vyhodnocovaní úspešnosti architektúr na transformovaných obrázkoch som teda u každej architektúry nebral do úvahy tie, ktoré neboli klasifikované správne už ako originál, a to aj ak boli po aplikovaní transformácie klasifikované správne.

Z výsledkov v tabuľke 7 vidieť transformácie, ktoré mali na úspešnosť architektúr dopad. Napríklad pri transformáciach engrave, mosaic, pixelize mali CNN úspešnosť iba okolo 3%. Ukázali sa aj rozdiely medzi architektúrami. Podľa očakávaní, najhoršie výsledky dosiahla architektúra Alexnet, najlepšie výsledky mala architektúra ResNeXt. Transformácie emboss a oilify neboli v zmätení CNN také úspešné, pri obidvoch mali architektúry okolo 50% úspešnosť.

Výsledky konvolučných neurónových sietí dopadli podľa očakávaní. Architektúra AlexNet si počínala najhoršie, ResNeXt zasa naopak najlepšie. Všetky transformácie ovplyvňujú konvolučné neurónové siete do takej miery, že im takmer úplne znemožňujú správnu klasifikáciu. Čím viac porušujeme hrany (engrave, ripple, mosaic, pixelize), tým menej je model úspešný. Pokiaľ hrany nezdeformujeme príliš (oilify) alebo prechody zvýrazníme (emboss), úspešnosť je stále

	AlexNet	VGG	GoogLeNet	ResNet	ResNeXt	Celkovo
contrast	0,00 %	16,67 %	25,00 %	17,39 %	20,83 %	16,52 %
emboss	30,00 %	45,83 %	58,33 %	47,83 %	66,67 %	50,43 %
engrave	0,00 %	8,33 %	8,33 %	0,00 %	0,00 %	3,48 %
hurl	0,00 %	4,17 %	8,33 %	8,70 %	8,33 %	6,09 %
lensblur	10,00 %	8,33 %	12,50 %	17,39 %	37,50 %	17,39 %
mosaic	5,00 %	8,33 %	0,00 %	0,00 %	4,17 %	3,48 %
oilify	45,00 %	37,50 %	45,83 %	60,87 %	58,33 %	49,57 %
papertile	0,00 %	16,67 %	16,67 %	13,04 %	33,33 %	16,52 %
pixelize	5,00 %	0,00 %	0,00 %	0,00 %	8,33 %	2,61 %
ripple	10,00 %	8,33 %	0,00 %	4,35 %	8,33 %	6,09 %

Tabuľka 7: Tabuľka úspešností architektúr pri klasifikácií transformovaných obrázkov.

ovplyvnená, ale nie tak negatívne.

4.3 Experiment s ľuďmi

Na zistenie, ktoré transformácie sú naozaj nebezpečné tým, že oklamú neurónové siete, ale ľuďom predajú cielenú informáciu, je potrebné zistiť aký vplyv majú transformácie na ľudí.

4.3.1 Dotazník

Schopnosť rozpoznať obrázky s transformáciami u ľudí som overoval dotazníkmi. Dotazníkov som spravil celkovo 6 verzií aby som využil čo najviac obrázkov. Každá transformácia má v dotazníkoch dva obrázky tak, aby sa v žiadnom dotazníku neobjavila tá istá kombinácia obrázok-transformácia viackrát. Dotazník som zostavoval tak, že som si nechal náhodne vybrať 20 obrázkov zo všetkých 26. Z týchto 20 vybraných, som náhodne vybral 6, ktoré som nahradil za tie, ktoré sa nedostali do prvých 20. Tým som využil všetkých 26 obrázkov. Vznikli 2 kolekcie obrázkov. Aplikovaním transformácií som vytvoril 6 verzií dotazníkov, z každej kolekcie 3. Na určenie, ktorá dvojica obrázok-transformácia má byť použitá v danej verzií dotazníku som napísal program v Pythone. Z celkového počtu možných dvojíc obrázok-transformácia 260 (26 orázkov $\times$ 10 transformácií) bolo použitých 120.

Do dotazníku som zaradil aj otázky ohľadom stavu zraku respondentov. Pýtal som sa na zrakové vady, a či dotazník vypĺňajú s okuliarmi, šošovkami alebo bez nápravy zraku. Pokiaľ by bol respondent napríklad ďalekozraký (teda nevidí dobre na krátke vzdialenosti) a vypĺňal by dotazník bez okuliarov, jeho schopnosť rozpoznať obrázky je do značnej miery ovplyvnená už týmto. Preto jeho odpovede nemusia byť úplne smerodajné. To isté platí u ľudí s poruchou farbocitu (napríklad Daltonizmus).

Všetky obrázky v dotazníku boli rozmerov 256×256 pixelov. Ľudia dostali inštrukcie, aby napísali všetky objekty, ktoré na obrázku vidia. Pri vyhodnocovaní som postupoval rovnako ako pri CNN, teda pomocou top-5 chyby. Pokiaľ sa v prvých 5 odpovediach nachádzala správna odpoveď, celá odpoveď je braná ako správna.

Pri nárvhu formy dotazníku som uvažoval o dvoch spôsoboch, ako dať respondentom na výber z kategórií. Pokiaľ by mali na výber, čo i len z časti všetkých 1000 kategórií, napríklad by som vybral 200 kategórií, prezerať tieto kategórie by bolo zdĺhavé a náročné. Mohlo by to buď respondentov demotivovať vo vypĺňaní a tým by som prišiel o dáta, alebo by zoznam aj tak ignorovali a písali podľa seba. Pokiaľ by ich zasa bolo veľmi málo, napríklad 50, respondenti by nemuseli obrázok rozpoznať ale práve vďaka tomuto zoznamu by si vybrali nejakú kategóriu. Ďalší spôsob, ako dať respondentom na výber z kategórií, je pomocou našepkávača. Tu by ale bolo riziko, že respondenti takisto nerozpoznajú obrázok, ale skúšali by rôzne možnosti čo im našepkávač poradí, a správne by klasifikovali aj obrázky, ktoré nerospoznali. Preto respondenti nemali na výber zo žiadnych kategórií.

Tým, že respondenti nemali jasne určené kategórie, z ktorých by vyberali a odpovede za nich nedopĺňal našepkávač, znemožnilo to automatizované vyhodnotenie odpovedí. Pre každú kategóriu existuje mnoho synonym a je možnosť, že by som na nejaké zabudol pri písaní algoritmu, ktorý by automaticky vyhodnotil dotazník. Ďalším problémom sú preklepy a gramatické chyby. Respondenti vypĺňali dotazník pomocou služby Google Forms, teda na počítači. Medzi odpoveďami sa objavili napríklad *brkolica* alebo *okliare*. Z týchto dôvodov som odpovede z dotazníkov vyhodnocoval ja.

4.3.2 Respondenti

Respondentov, ktorý vyplnili dotazník bolo 92. Každý dotazník vyplnilo 13-17 respondentov. Väčšinou sa jednalo o študentov rôznych ročníkov Katedry informatiky Prírodovedeckej fakulty Univerzity Palackého a študentov medicíny Lekárskej fakulty Univerzity Palackého. Väčšina respondentov netrpela žiadnymi poruchami zraku. Druhou najpočetnejšou skupinou boli krátkozrakí respondenti. Pri nich nerobilo problém ak vypĺňali dotazník aj bez okuliarov alebo šošoviek. Niektorí respondenti trpeli aj astigmatizmom. Všetci ďalekozrakí respondenti vypĺňali dotazník s okuliarmi. U dvoch respondentov je riziko, že nesprávna klasifikácia nebola spôsobená transformáciou, ale zrakovými vadami. Jeden respondent trpel daltonizmom a druhý astigmatizmom, ale vypĺňal dotazník bez okuliarov či šošoviek. Počínali si ale dobre, preto som tieto odpovede rátal s rovnakou váhou ako ostatné. Jednu odpoveď som vylúčil, keďže respondent uviedol, že netrpí zrakovou poruchou, ale vyplnil odpoveď len k jednému obrázku. Ostatné odpovede nechal úplne prázdne, preto túto odpoveď považujem za nesprávne vyplnenú.

4.3.3 Klasifikácia

U ľudí som neoveroval, či spoznajú originálne obrázky bez aplikovaných transformácií. Po konzultácií s PhDr. Danielom Dostálom Ph.D. sme dospeli k záveru, že dané kategórie sú dostatočne všeobecné a obrázky sú vhodné na to, aby ich ľudia nemali problém rozpoznať.

Každý respondent vypĺňal vždy len jeden dotazník. Pokiaľ by vypĺňal viac verzií, mohlo by sa stať, že obrázok s danou transformáciou by normálne nespoznal, ale v predošlom dotazníku mal rovnaký obrázok aplikovanú transformáciu, ktorá nemí taký vplyv na rozpoznanie. Tým, že by si tento obrázok zapamätal, by neboli výsledky presné.

4.3.4 Výsledky

Výsledky u ľudí som vyhodnocoval ručne. Všetky výsledky mali rovnakú váhu. Uznával som aj odpovede, ktoré neboli úplne v súlade s inštrukciami. Podľa inštrukcií mali respondenti zadávať odpovede v tvare *včela, kvet, tráva*. Niektoré odpovede ale boli pomenovanie obrázka spôsobom, čo sa na obrázku deje. Napríklad *včela opeluje kvet*. Odpovede beriem ako správne, pretože pokiaľ respondent pomenoval takto konkrétne obrázok, je jasné, že rospoznal objekty na obrázku.

Určíme stupnicu, koľko percent respondentov muselo byť na danej transformácii úspešných, aby sme mohli povedať či ovplyvňuje klasifikáciu alebo nie.

95% - 100% ... neovplyvňuje klasifikáciu

75% - 94% ... mierne ovplyvňuje klasifikáciu

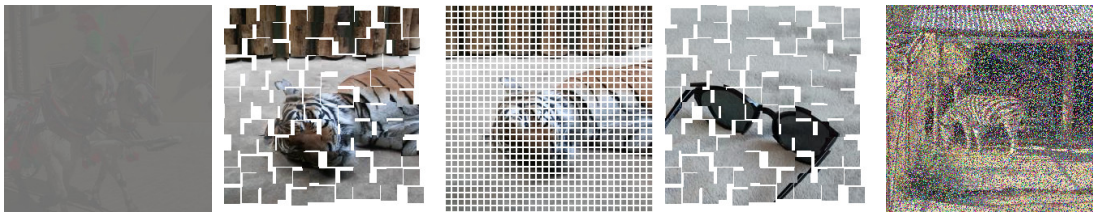
50% - 74% ... ovplyvňuje klasifikáciu

20% - 49% ... veľmi ovplyvňuje klasifikáciu

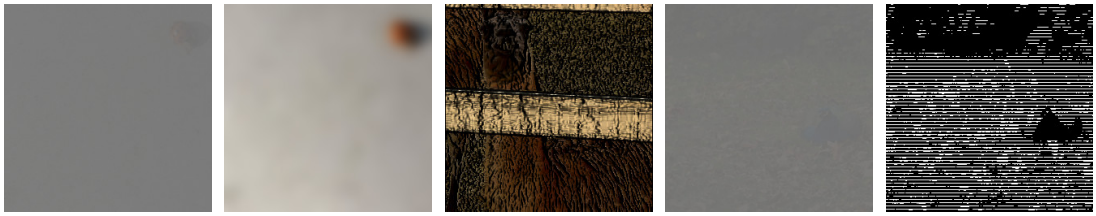
0% - 19% ... znemožňuje klasifikáciu

Každá transformácia aspoň mierne ovplyvňuje klasifikáciu. Najmenej ovplyvňuje klasifikáciu transformácia papertile. Pri nej respondenti dosiahli úspešnosť až 84%. Do tejto kategórie ďalej spadajú aj transformácie emboss a oilify. Ostatné transformácie buď ovplyvňujú alebo veľmi ovplyvňujú klasifikáciu. Len transformácia ripple znemožňuje klasifikáciu obrázkov aj u respondentov. Kompletné výsledky môžete vidieť v tabuľke 8.

Hlavným problémom pre respondentov bol príliš malý objekt, ktorý sa po transformácií ešte viac stratil, alebo príliš priblížený objekt. Naopak, pokiaľ bol objekt primeranej veľkosti a bol správne umiestnený na obrázku (väčšinou v strede), respondenti s tým nemali problém. Dokonca sa častokrát stalo, že obrázok klasifikovali správne všetci respondenti. U viacerých sa stávalo, že nesprávne klasifikoval len jeden či dvaja respondenti. Nižšia úspešnosť je teda u ľudských respondentov spôsobená prevažne obrázkami, ktoré mali objekt príliš malý alebo nevhodne umiestnený. V takýchto prípadoch klasifikovali obrázok správne iba dvaja až traja respondenti.



(a) Príklady obrázkov, kde všetci respondenti odpovedali správne a všetky neurónové siete nesprávne.



(b) Príklady obrázkov, kde väčšina respondentov odpovedalo nesprávne.

Obr. 11: Príklady úspešných a neúspešných obrázkov u ľudí.

transformácia	**úspešnosť respondentov**
contrast	68,13 %
emboss	78,02 %
engrave	48,35 %
hurl	60,44 %
lensblur	45,05 %
mosaic	30,77 %
oilify	75,27 %
papertile	84,07 %
pixelize	69,78 %
ripple	8,24 %

Tabuľka 8: Tabuľka úspešností respondentov pri klasifikácií transformovaných obrázkov.

transformácia	úspešnosť respondentov	úspešnosť cnn	rozdiel
contrast	68,13 %	16,52 %	51,61 %
emboss	78,02 %	50,43 %	27,59 %
engrave	48,35 %	3,48 %	44,87 %
hurl	60,44 %	6,09 %	**54,35** %
lensblur	45,05 %	17,39 %	27,66 %
mosaic	30,77 %	3,48 %	27,29 %
oilify	75,27 %	49,57 %	25,7 %
papertile	84,07 %	16,52 %	**67,55** %
pixelize	69,78 %	2,61 %	**67,17** %
ripple	8,24 %	6,09 %	2,15 %

Tabuľka 9: Porovnanie respondentov a konvolučných neurónových sietí.

5 Zhodnotenie

Poznáme výsledky neurónových sietí a poznáme výsledky ľudí. V tejto kapitole budeme skúmať porovnanie týchto výsledkov, ako aj konkrétne použitie výsledkov.

5.1 Porovnanie

Hlavným rozdielom medzi ľuďmi a neurónovými sieťami v tomto experimente je, že pri ľuďoch do veľkej miery záleží na tom, aký je originálny obrázok. Pokiaľ je veľký akurát, teda nie je príliš malý alebo príliš zblízka, je väčšia šanca, že človek obrázok rozpozná (obrázok 11a). U neurónových sietí ale nič také neplatí, ako náhle sa nám podarí dostatočne rozbiť hrany (transformácie pixelize, papertile, engrave), tak úplne znemožníme neurónovým sieťam obrázky klasifikovať správne.

Dôležité sú tie transformácie, u ktorých sa ukázal veľký rozdiel medzi ľuďmi a neurónovými sieťami. Nebudú nebezpečné tie, ktoré buď príliš neovplyvňujú ani ľudí ani neurónové siete, alebo tie, ktoré znemožňujú správnu klasifikáciu obom. Nebezpečné sú práve tie, ktoré dokázali zmiasť neurónové siete a nie ľudí.

Najväčšie rozdiely môžeme vidieť u transformácií papertile, pixelize a hurl (tabuľka 9). V týchto prípadoch si neurónové siete počínali oproti ľuďom veľmi zle. Ak by sme použili obrázky, ktorých objekt je správnej veľkosti, tento rozdiel by sa u každej transformácií ešte zväčšil.

5.2 Filtrovanie nevhodného obsahu

Nástupom sociálnych sietí ako Facebook alebo Instagram či iných platforiem určených na zábavu, kde môže obsah pridávať ktokoľvek a zadarmo, ako napríklad YouTube alebo TikTok, začala byť otázka kontroly pridávaného obsahu dôležitejšia, než kedykoľvek predtým. Existujú pokročilé metódy ako nevhodný obsah

filtrovať (napríklad sa používajú niektoré modely CNN), no napriek tomu sú stále hlavným filtrom takzvaní moderátori obsahu. Ich úlohou je prezerať či už obsah, ktorý sa už na platformu dostal a bol nahlásený uživateľom ako nevhodný alebo obsah, pri ktorom nie je jednoznačne určené či je nevhodný alebo nie. Potom sú to práve oni, ktorí rozhodujú o tom či je daný obsah naozaj nevhodný. To je ale veľký problém, pretože títo ľudia sú dennodenne vystavovaný rôznemu nevhodnému obsahu od násilia na zvieratách alebo ľuďoch až po samovraždy. Títo moderátori preto trpia rôznymi psychickými problémami od depresií až po posttraumatickú stresovú poruchu. Objavili sa už aj prípady, ako napríklad na konci roku 2021[3] alebo skôr v roku 2018[4], kedy moderátori podali obžalobu na spoločnosť, pre ktorú obsah moderovali. Vďaka takýmto prípadom sa dostáva viac do povedomia téma psychického zdravia takýchto zamestnancov.

Preto by sa okrem investícií do lepších programov na starostlivosť o psychické zdravie mal klásť dôraz aj na vylepšenie automatizovaných filtrov nevhodného obsahu. Práve obohatenie o rozpoznávanie aj transformovaných obrázkov by mohol byť ďalší krok za zlepšením týchto filtrov.

5.3 Budúci vývoj

Niektoré transformácie, ktoré sa ukázali ako problémové pre neurónové siete, ľuďom nerobili až taký problém (pixelize, hurl, papertile). Mohli by sa zaradiť medzi tréningové dáta pri trénovaní bežných modelov. Tu je ale otázka či prinesú tak podstatné zlepšenie, aby sa oplatilo používať ich vždy pri procese trénovania. Tieto transformácie už nie sú tak výpočetne nenáročné ako v [2] alebo v [**relate2**]. Preto treba zvážiť či výsledné zlepšenie kvality modelu bude dostačujúce, aby sa tieto transformácie oplatilo aplikovať. Pokiaľ by sme ale chceli natrénovať model, ktorý by bol chopný rozpoznávať dané transformácie ako dodatok k bežným dátam, je na mieste takto transformované obrázky zaradiť do tréningu aj napriek výpočetnej zložitosti.

[3] https://edition.cnn.com/2021/12/29/tech/tiktok-lawsuit-content-moderator/index.html

[4] https://www.theverge.com/2020/5/12/21255870/facebook-content-moderator-settlement-scola-ptsd-mental-health

Záver

Aj napriek tomu, že neurónové siete dosahujú skvelé výsledky v počítačovom videní, stojí za to stále hľadať nové slabé miesta. Odhalenie týchto slabých miest je prvý krok na ich odstránenie. Jednoduché transformácie, ako horizontálne alebo vertikálne prevrátenie, rotácia či manipulácia s farbami, už sú bežnou súčasťou procesu trénovania neurónových sietí. Výpočetná sila dnešnej doby, nám ale dovoľuje v tréningovom procese zaradiť aj zložitejšie transformácie, ako papertile alebo pixelize, čím by sme mohli zlepšiť kvalitu neurónových sietí.

Conclusions

Although neural networks achieve great results in computer vision, it is still worth looking for new weaknesses. Exposing these weaknesses is the first step to eliminating them. Simple transformations, such as horizontal or vertical flipping, rotation or color manipulation, are already a common part of training process of neural networks. However, the computational power of today allows us to include more complex transformations, such as papertile or pixelize in the training process, which could improve the quality of neural networks.

A Ukážka dotazníku

Dotazník pozostával z dvoch otázok ohľadom zraku a 20 obrázkov. Prvá otázka bola na konkrétne poruchy zraku, teda krátkozrakosť, ďaľekozrakosť, daltonizmus, astigmatizmus, žiadne, alebo iné (možnosť dopísať). Druhá otázka bola, či respondent vypĺňa dotazník s okuliarmi, šošovkami alebo bez.

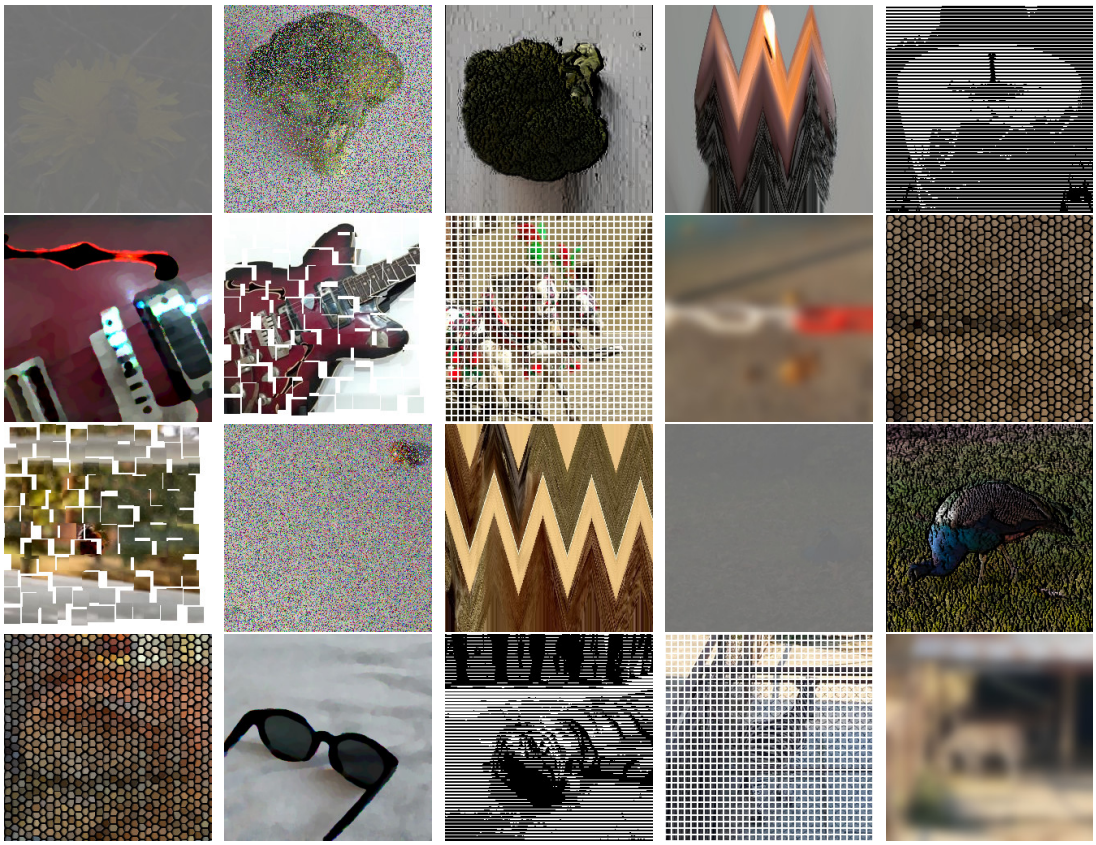
Druhá časť sa skladala z 20 obrázkov. Inštrukcie k obrázkovej časti vyzerali nasledovne:

Pomenujte objekty na fotografii. V prípade vymenovania viacerých objektov oddeľte odpovede čiarkou a medzerou. Vzorová odpoveď: "kôň, koč, strom".

Do úvahy bude braných len prvých 5 uvedených objektov. Teda z odpovede "zubná pasta, zubná kefka, zrkadlo, umývadlo, sprcha, záves"by "záves"nebol braný do úvahy.

Pokiaľ neviete nájsť na obrázku žiadny objekt, nechajte pole s odpoveďou prázdne.

Príklad obrázkov dotazníka môžete vidieť na obrázku 12. Všetky dotazníky nájdete na priloženom DVD vo formátte .pdf.



Obr. 12: Príklad obrázkov dotazníku.

B Obsah priloženého DVD

dataset/
: Obsahuje všetky obrázky v originálnej podobe aj s aplikovanými transformáciami.

dotazníky/
: Obsahuje všetky dotazníky vo formáte .pdf ako aj priečinky s fotkami použitými na dotazníky.

vysledky_cnn/
: Obsahuje výsledky pre každú transformáciu vo formáte .csv.

vysledky_ludia/
: Obsahuje nespracované odpovede na každú verziu dotazníku vo formáte .csv.

src/
: Obsahuje všetky zdrojové kódy.

used_filtes.txt
: Popisuje použité transformácie aj s použitými hodnotami pluginu BIMP v programe GIMP.

vysledky.xlsx
: Excel tabuľka spracovaných a upravených výdledkov.

Literatúra

[1] RUSSAKOVSKY, Olga; DENG, Jia; SU, Hao a kol. ImageNet Large Scale Visual Recognition Challenge. 2014. Available also from WWW: ⟨arXiv: 1409.0575 (cs.CV)⟩.

[2] PEREZ, Luis; WANG, Jason. The Effectiveness of Data Augmentation in Image Classification using Deep Learning. 2017. Available also from WWW: ⟨arXiv: 1712.04621 (cs.CV)⟩.

[3] HOWARD, Andrew G. Some Improvements on Deep Convolutional Neural Network Based Image Classification. 2013. Available also from WWW: ⟨arXiv: 1312.5402 (cs.CV)⟩.

[4] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet classification with deep convolutional neural networks. 2012. Available also from WWW: ⟨http://dx.doi.org/10.1145/3065386⟩. ISSN 0001-0782.

[5] BURKOV, Andriy. 2019. 1,5,6. The Hundred-Page Machine Learning Book. Available also from WWW: ⟨https://www.ebook.de/de/product/42052142/andriy_burkov_the_hundred_page_machine_learning_book.html⟩. ISBN 199957950X.

[6] DENG, Jia; DONG, Wei; SOCHER, Richard a kol. ImageNet: A large-scale hierarchical image database. In. *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009. Available also from WWW: ⟨http://dx.doi.org/10.1109/cvpr.2009.5206848⟩.

[7] SIMONYAN, Karen; ZISSERMAN, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014. Available also from WWW: ⟨arXiv: 1409.1556 (cs.CV)⟩.

[8] SZEGEDY, Christian; LIU, Wei; JIA, Yangqing a kol. Going Deeper with Convolutions. 2014. Available also from WWW: ⟨arXiv: 1409.4842 (cs.CV)⟩.

[9] HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. 2015. Available also from WWW: ⟨arXiv: 1512.03385 (cs.CV)⟩.

[10] XIE, Saining; GIRSHICK, Ross; DOLLÁR, Piotr; TU, Zhuowen; HE, Kaiming. Aggregated Residual Transformations for Deep Neural Networks. 2016. Available also from WWW: ⟨arXiv: 1611.05431 (cs.CV)⟩.