ŽILINSKÁ UNIVERZITA V ŽILINE

AUTOREFERÁT
DIZERTAČNEJ PRÁCE

Žilina apríl 2021

Ing Peter Lukáč

Žilinská univerzita v Žiline
Fakulta riadenia a informatiky

Peter Lukáč, Ing.

Autoreferát dizertačnej práce

DOLOVANIE TRIED V ČIASTOČNE ANOTOVANÝCH OBRAZOVÝCH DÁTACH

na získanie akademického titulu **„philosophiae doctor“ (v skratke PhD.)**
v študijnom programe doktorandského štúdia
aplikovaná informatika*

v študijnom odbore:
informatika*

Žilina, apríl 2021

Dizertačná práca bola vypracovaná v dennej forme doktorandského štúdia doktorandského štúdia **na katedre informačných sietí, Fakulte riadenia a informatiky Žilinskej univerzity v Žiline**

Predkladateľ: **Ing. Peter Lukáč**
Katedra informačných sietí
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Školiteľ: **prof. Ing. Martin Klimo, PhD.**
Katedra informačných sietí
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Školiteľ špecialista: **Ing. Peter Tarábek, PhD.**
Katedra matematických metód a operačnej analýzy
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Oponent: **prof. Ing. Miloš Oravec, CSc.**
Fakulta elektrotechniky a informatiky,
Slovenská technická univerzita,
Ilkovičova 3, 812 19 Bratislava 1

Oponent: **doc. Ing. Roman Jarina, PhD.**
Fakulta elektrotechniky a informačných technológií,
Žilinská univerzita v Žiline,
Univerzitná 1, 01026 Žilina

Autoreferát bol rozoslaný dňa: .....

Obhajoba dizertačnej práce sa koná dňa 18.8.2021 o 14:00 h. pred komisiou pre obhajobu dizertačnej práce schválenou pracovnou skupinou odborovej komisie **informatika v študijnom programe aplikovaná informatika**, vymenovanou dekanom Fakulty riadenia a informatiky Žilinskej univerzity v Žiline dňa .....

prof. Ing. Karol Matiaško, PhD.*
predseda pracovnej skupiny odborovej komisie
v študijnom odbore **informatika**
v študijnom programe **aplikovaná informatika***

Fakulta riadenia a informatiky
Žilinská univerzita
Univerzitná 8215/1
010 26 Žilina

Obsah

Úvod

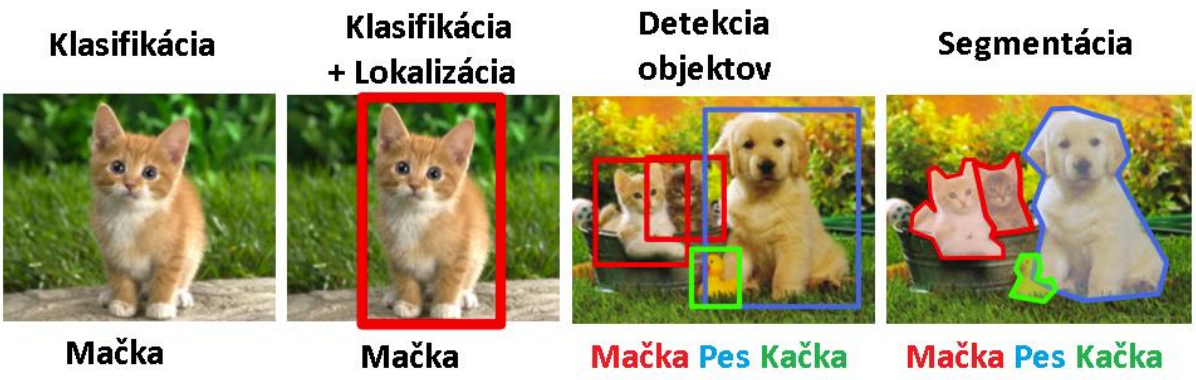
Jedna z dávnych túžob ľudí, ktorá siaha až do starovekého Grécka, je vytvoriť stroj, ktorý myslí. Inšpirovaný biológiou a podporovaný IT vznikol odbor umelá inteligencia (AI), ktorý sa snaží túto túžbu naplniť. Dnes je AI prosperujúca oblasť s mnohými praktickými aplikáciami a aktívnymi výskumnými témami. Vytvárame inteligentný softvér, ktorý má za úlohu automatizovať rutinné práce, pochopiť reč a obraz, robiť diagnostiku v medicíne a podporovať základný vedecký výskum.

V počiatkoch sa oblasť AI snažila riešiť problémy, ktoré sú z intelektuálneho hľadiska pre ľudí zložité, ale relatívne jednoduché pre počítače – problémy, ktoré môžu byť popísané matematickými pravidlami. Naozajstná výzva AI je riešiť úlohy, ktoré sú ľahké pre ľudí, ale je ťažké ich popísať formálne – problémy, ktoré vieme riešiť intuitívne, ako napr. rozpoznávanie hovorených slov, alebo tváre v obraze.

Táto práca sa zaoberá rozpoznávaním obrazu a problémami s nimi súvisiacimi. Hlavným problémom dneška pri práci s obrazom sú dáta. Ich získavanie je relatívne náročná práca, pretože si vyžaduje nielen jednoduché operácie (ako označovanie relevantných oblastí na obrázku a ich následné kategorizovanie), ale aj operácie náročné na sústredenie, dôsledkom čoho sa môžu vnášať chyby, ktoré nepriaznivo pôsobia na výsledné riešenie.

1. Úvod do problematiky

Z pohľadu tejto práce ktorá sa zaoberá rozpoznávaním obrazu môžeme použiť nasledovné kategorizovanie základných úloh s učiteľom v počítačovom videní pre statické obrázky:



Obrázok.1. **typy úloh v počítačovom videní**

Klasifikácia – Ako vstupné dáta existujú obrázky, ktoré majú určené, do akej kategórie (triedy) patria. Úloha spočíva v určení triedy hlavného objektu na vstupnom obrázku.

Klasifikácia + Lokalizácia – Rozšírenie úlohy klasifikácie o úlohu lokalizácie. Okrem samotného obrázku a triedy je potrebné určiť aj oblasť na obrázku, kde sa daný objekt nachádza. Pre úspešné riešenie stačí nájsť ľubovoľný výskyt hlavného objektu.

Detekcia objektov – Klasifikácia a lokalizácia rieši zjednodušenú úlohu - na obrázku sa hľadá len jeden hlavný objekt. Detekcia objektov je úloha, pri ktorej je potrebné označiť všetky objekty, ktoré sa majú rozpoznať a priradiť im príslušnú kategóriu.

Segmentácia – Je to v podstate detekcia objektov, len presnejšia. Namiesto označenia objektov pomocou BBOX-u je objekt označený presne podľa tvaru objektu formou pixelov.

Pre každú z týchto úloh je potrebné, aby boli vytvorené datasety, ktoré zahŕňajú také obrazové dáta, ktoré majú požadovanú kvalitu, čím sa myslí, že reprezentujú doménu úloh a majú čo najmenšiu chybovosť parametrov, dostatočnú diverzitu (nie je dobré, ak existujú dáta, ktoré

sú veľmi podobné. napr. ten istý objekt v rovnakom uhle) a početnosť (početnosť dát v kategóriách by mal byť rovnaký). Jeden z najväčších datasetov pre úlohy rozpoznávania obrazu je *ImageNet* [1]. Za jeho vznikom bola motivácia vytvoriť čo najväčší dataset 80 000 tried (pod pojmom trieda si môžeme predstaviť skupinu dát reprezentujúcu rovnakú množinu ako napr. zviera, auto,…) a na každú 500 až 1000 obrázkov. Predstavuje testovací dataset na ktorom sa môžu porovnávať vedecké riešenia. Súťaž *ILSVRC* (ImageNet Large Scale Visual Recognition Challenge), bola jednou v ktorej tímy z celého sveta súťažili v úlohách rozpoznávania obrazu na tomto datasete. Ako základ pre vznik datasetu si vybrali WorldNet [2] databázu slov čo je hierarchická štruktúra anglického jazyka, kde jednotlivé slová sú poprepájané vzťahom namiesto abecedného poradia. Napr. "mačka" je pod slovom "mačkovité šelmy" a tie sú pod "cicavce". Tento slovník obsahuje viac než 155 000 indexovaných slov. V roku 2009 keď vyšla prvá verzia ImageNet-u obsahovala 3.2 milióna anotovaných obrázkov rozdelených do 5 247 kategórii a zotriedených do 12 podstromov ako cicavce, nábytok,... (dnes je to 14 miliónov obrázkov v 22 000 kategóriách). Počiatok vzniku datasetu bol náročný, keďže najatí študenti, ktorý mali hľadať a pridávať obrázky nestačili na objem práce, ktorú bolo potrebné vykonať. Pri ich tempe by to trvalo 19 rokov než by sa dataset stal dostatočne veľkým. Použitie služby Amazonu *Mechanical Turk*, čo je systém kde veľké množstvo ľudí po celom svete sedí za počítačom a vykonáva jednoduché operácie za malú finančnú odmenu pár centov, pomohlo vytvoriť dataset v takej miere, aby ho bolo možné zverejniť. Pracovalo na ňom cez 50 000 ľudí prehľadávalo viac ako 160 miliónov obrázkových kandidátov, pričom pre zefektívnenie vyhľadávania bol vytvorený software na hľadanie nových obrázkov a tak isto aj na ich samotnú validáciu. Jednotlivé obrázky sa validovali 2-5 krát.

Zozbieranie takéhoto množstva dát potrebuje veľké úsilie ľudí, a nemalé technické a finančné prostriedky. Je preto dôležité sa venovať tejto problematike za účelom zjednodušenia a zlacnenia postupov pri získavaní dát. Preto si táto práca kladie za jeden z cieľov nájsť také metódy a postupy, ktoré by zefektívnili spôsob získavania a opravovania dát či už automatickým, alebo poloautomatickým spôsobom.

2. Cieľ práce

Pre riešenie veľkých úloh rozpoznávania obrazu je vytvorenie veľkého datasetu základnou nutnou podmienkou úspešného riešenia. Vytvorenie takéhoto datasetu predstavuje veľkú finančnú aj časovú záťaž. Buď je to z dôvodu náročného získavania dát z dôvodu absencie niektorých prípadov, alebo je potrebné zapojiť veľké množstvo ľudskej sily (dáta sú rozsiahle a je potrebná kontrola z dôvodu prevencie zavedenia chýb). Preto si táto práca dáva za svoj cieľ:

Navrhnúť postupy a metódy na zefektívnenie procesu získavania anotovaných dát pre potreby vývoja metód počítačového videnia založených na hlbokom strojovom učení.

V prvom rade je to potreba minimalizovať množstvo dát anotovaných človekom. Získavanie dát je drahá záležitosť a môže trvať mesiace úsilia vytvoriť požadovaný dataset s potrebnou kvalitou. Čím sú dáta horšej kvality (nepresnosť anotácii, alebo chybné zaradenie kategórii), tým sa trénované modely stávajú menej použiteľné. Je preto dôležité hľadať spôsoby ako zamedziť vnášaniu chýb do datasetov - či už anotovaním človekom prípadne automatizovanými postupmi anotovania, alebo hľadaním takých, ktoré prinášajú veľkú informačnú hodnotu. Tak isto čistenie a zlepšovanie kvality samotných dát sa môže robiť poloautomaticky kedy sieť sama dokáže identifikovať chybne anotované vzorky, ktoré sú v tréningových dátach ak je ich percento malé. Takýto postup sa môže iterovať čo by malo viesť k zlepšeniu kvality dát. V tejto práci za hlavnú výzvu sme si vytýčili inú organizáciu priestoru výstupných dát z neurónovej siete oproti klasickému unárnemu kódovaniu.

Predpokladáme, že by to mohlo viesť k lepšiemu modelovaniu neurčitosti klasifikácie a tým identifikovaniu problematických dát či už z pohľadu chýb v datasete, alebo z pohľadu dôležitosti pre anotovanie.

3. Metódy riešenia

Pri riešení klasifikačných úloh podporovanými algoritmami hlbokého strojového učenia s učiteľom sú kritickým faktorom pre dosiahnutie požadovaného výkonu algoritmu dobre anotované dáta. Anotovanie je typicky manuálny proces kde anotátor nasleduje požadované usmernenia akým spôsobom má proces vykonávať t.j. aké metadáta anotovaným dátam priradiť, či je to označenie nejakej oblasti (napr. orámovanie) alebo nastavenie rôznych príznakov. Výsledok takto spracovaných dát je vysoko závislí od zručnosti a skúseností anotátora čo môže mať dopad na konečný výsledok riešenia. Preto vo väčšine prípadov sa dáta validujú iteračnými postupmi tak, aby potvrdenie správnosti anotácie bolo viacnásobné a až potom sa výsledok prehlási za správny. Takéto získavanie dát je pritom neefektívne z pohľadu zdrojov a času. Existujú rôzne prístupy ako začleniť čiastočne anotované alebo neanotované dáta do procesu tréningu [3]. Úloha vytvoriť dataset pre úlohy strojového učenia nie je nová a preto v nasledujúcih kapitolách ukážem základné prístupy a problémy, pri ktorých riešitelia narazili.

1. Prístupy a metódy anotovania

ImageNet

Je to jeden z najväčších datasetov, ktorý vznikal niekoľko rokov a autori k nemu vydali niekoľko publikácií [1,7,8]. Pre vývoj algoritmov pracujúcich s obrazom je potrebné vytvoriť dostatočnú databázu anotovaných obrázkov, ktorá poskytne dáta pre trénovanie a testovanie. Jednou z takýchto iniciatív bolo vytvorenie databázy ImageNet. Tá si za cieľ dala vytvorenie databázy obrázkov, ktorá by indexovala všetky možné objekty na svete. Je založená na štruktúre WorldNet [2], ktorá vychádza z konceptu, že slovná fráza alebo viacero slov je popísaných synonymom, tzv. synset, a uložených do štruktúry pomocou ontológie. Týchto podstatných slov je v databáze okolo 80 000 a ImageNet si kládol za úlohu poskytnúť v priemere 500 až 1000 obrázkov na každé slovo, čo sú desiatky miliónov presne anotovaných obrázkov. Následne bola k tomuto datasetu vytvorená súťaž ILSVRC [7], ktorá dala možnosť vyvíjať a testovať algorimy na rozpoznávanie obrazu. Trvala 7 ročníkov od roku 2010 do 2017, kedy 29 z 38 súťažiacich tímov mali presnosť väčšiu ako 95% [5]. V 2018 bola pridaná úloha pre detekciu 3D objektov. Vytvorenie 3D model objektu je náročnejšia úloha ako anotovať 2D obrazové dáta. Tento typ úlohy mal pomôcť pri úlohách navigácie v robotike. Dnes už súťaž neprebieha z dôvodu že existuje konsenzus o tom že rozpoznávanie obrazu (klasifikácia a lokalizácia) je v princípe vyriešená téma a rôzne tímy sa presunuli na iné úlohy. Neskôr vznikli ďalšie datasety ako napr. COCO [6], v ktorom prebiehajú súťaže aj dnes. Tie su už zamerané na iné úlohy, ako je napríklad segmentácia obrazu. Keďže dataset sa mal používať na porovnávanie nových metód bolo na začiatku potrebné zadefinovať úlohy, na ktoré sa bude používať. Cieľom ILSVRC bolo riešenie nasledujúcich úloh:

Klasifikácia obrazu – Z pohľadu datasetu, sú dáta anotované podľa prítomnosti najvýraznejšej triedy patriacej do jednej z 1000 kategórií. Každý vzor obsahuje iba jednu pravdivú anotáciu. Pre každý vzor algoritmy produkujú zoznam najpravdepodobnejších 5 tried prítomných na obrázku. Metóda je úspešná ak sa pravdivá anotácia nachádza medzi

týmito piatimi predikciami (tzv. TOP 5 metrika) Pri tvorbe tried bola snaha pokryť čo najviac anglických slov určujúcich podstatné mená. Kategórie a obrázky boli zafixované pre poskytnutie štandardizovaného testu keďže sa dataset časom zväčšoval.

Lokalizácia jedného objektu – ide o rozšírenie úlohy klasifikácie. Okrem samotnej predikcie triedy objektu je cieľom aj orámovanie jednej inštancie daného objektu v obrázku. Dáta teda pochádzaju z úlohy *klasifikácia obrazu* a boli rozšírené o orámovania objektov. Algoritmy vytvárajú zoznam objektov v obraze pomocou orámovania, ktoré definuje pozíciu a veľkosť jednej inštancie objektu danej kategórii. Vyhodnotenie prebiehalo na základe správnej predikcie kategórie a zároveň sa porovnávalo aj prekrytie predikovaného orámovania voči anotácii.

Detekcia objektu – Dáta pre túto úlohu obsahujú nové fotky zozbierané z databázy Flickr použitím hľadaných výrazov pre scény. Táto úloha mala za cieľ otestovať algoritmy, ktoré budú vedieť rozpoznávať viaceré objekty v scéne. Obrázky sú anotované pomocou orámovania určujúce pozíciu a veľkosť každej inštancii objektu danej kategórie. Trénovacia množina je obohatená (a) z úlohy *lokalizácia jedného objektu*, ktorý obsahuje anotácie pre všetky inštancie iba jedného objektu danej kategórie a (b) negatívne obrázky, o ktorých vieme že neobsahujú žiadnu kategóriu v trénovacom datasete.
Takto stanovené úlohy pre vytvorenie ImageNet datasetu si vyžiadali vytvorenie nových prístupov k anotovaniu. Aby boli modely schopné vyriešiť úlohy, ktoré si autori stanovili bolo potrebné splniť nasledujúce vlasnotsti datasetu:

Hierarchickosť – štruktúra ImageNet vyhádza z hierarchie WorldNet, čo je vlastne hierarchycký strom kde sú uložené jeho listy podľa ontológie (napr. na vrchu podstromu sú cicavce, pod ním sú domáce zvieratá, pod ním je vetva mačka a tá sa následne delí na rôzne druhy). Takto zadefinovaná štruktúra umožňuje vyberať rôzne dáta tak, aby sa ich triedy neprekrývali v úlohách a dali sa vyberať podľa rôznej úrovne abstrakcie.

Presnosť - požadovaná presnosť dát bola na všetkých úrovniach datasetu. To je náročné docieliť hlavne na nízkych úrovniach. Rozlíšiť mačku a psa nie je problém (vyššia úroveň v hierarchii). Rozlíšiť Siamskú a Barmskú mačku (nižšia úroveň v hierarchii) je aj pre človeka, ktorý nie je odborníkom, problém.

Rôznorodosť – objekty v obrázkoch by mali mať rôzne pozície, uhly pohľadu, pózy, rôzne druhy pozadí a prekážok, ktoré zakrývajú objekt. Pre potreby ohondotenia rôznorodosti autori počítali priemer obrázkov každého synset a merali bezstratovú veľkosť JPEG súboru, ktorý odráža množstvo informácí v obrázku. Na základe toho vyberali potencionálne odlišných kandidátov.
Prie vytváraní datasetu autori postupovali v nasledujúcih krokoch:

Zbieranie obrazových kandidátov – prvým krok bolo zhromaždiť dostatok kandidátov pre každý synset. Priemerná presnosť hľadania obrázkov z internetu, v dobe keď dataset bol tvorený, bola 10% [4]. Keďže cieľ bol ponúknnuť 500-1000 obrázkov na synset bolo treba zozbierať obrovské množstvo kandidátov. Obrázky sa zbierali z internetu pomocou rôznych vyhľadávačov a pre každý sysnet sa hľadala sada synoným. Výsledky vyhľadávania boli v stovkách až tisícoch. Pre zvýšenie počtu nájdení sa používali názvy z rodičovského mena synset. Napríklad ak sa hľadal výraz "chrt" a vo WordNet databáze bol záznam "Štíhli pes šlachtený v Anglicku" tak do hľadaného výrazu sa pridal "pes chrt". Pre nájdenie viacerych a

rôznych kandidátov boli hľadané výrazy preložené do ďalších jazykov ako čínština, španielčina, holandština a taliančina.

Čistenie obrazových kandidátov – na to aby bolo možné získať vysoko presné dáta zozbierané v predchádzajúcom kroku je potrebné ich verifikovať človekom. Toto bolo zabezpečené pomocou online platfromy Amazon Mechanical Turk (AMT). Pomocou nej je možné zadať úlohu pre užívateľov, za ktorú dostanú zaplatenú odmenu. Pri každej úlohe anotovania boli ponúknuté sady obrázkov a definícia vzoru zo synset, vrátane odkazu na Wikipédiu. Potom užívatelia overovali či obrázok pochádza z daného sysnet. Tak tiež bolo užívateľom povedané aby vyberali obrázky, v ktorých je objekt čiastočne zakrytý, prípadne bol na scéne "neporiadok" aby sa zabezpečila rôzna diverzita. Aj keď sa užívatelia snažili robiť robotu najlepšie ako vedeli, bolo potrebné vytvoriť systém kontroly kvality, pretože je treba dbať na dva problémy. Prvým je chybovosť ľudí a v niektorých prípadoch aj nedodržiavanie inštrukcií. Druhým je, že užívatelia nie vždy navzájom súhlasia, hlavne ak sa jedná o sysnset, ktorý sa nachádza hlbšie v strome. Riešenie pre tieto problémy je mať viacero užívateľov pre rovnaký obraz. Obraz je považovaný za správne anotovaný ak dostane väčšinu hlasov. Zistilo sa že pre rôznu obtiažnosť anotácie je potrebný aj rôzny počet hlasov. Napríklad pri "Barmskej mačke" je potrebný väčší konzenzus ako pri obraze "mačka". Preto bol vytvorený jednoduchý algoritmus, ktorý dynamicky určuje počet súhlasných anotácií. Pre každý synset bola vybratá náhodne podmnožina obrázkov a potom najmenej 10 užívateľov malo anotovať tieto obrázky. Z tejto anotácie sa potom vypočíta miera spoľahlivosti podľa zhody užívateľov, ktorá pri dosiahnutí stanoveného prahu považuje anotáciu za spoľahlivú. Pre všetkých ostatných kandidátov daného sysnet, ktorý sa má anotovať, je tento prah použitý pri rozhodovaní o správnosti anotovania. Pri niektorých synset nebolo možné nájsť dostatočnú zhodu, čo ukazuje že nie všetky môžu byť znázornené obrazom. Tento jednoduchý prístup viedol k vysokej spoľahlivosti datasetu pre rôzne podstromy na úrovni 99,7%.

Nepotrebujeme Orámovanie

Anotácia orámovania je únavná, časovo náročná a drahá práca. Napríklad anotovanie ILSVRC požadovala priemerne 42 sekúnd na vytvorenie anotácie pri použití AMT. Navyše bol potrebný špecializovaný SW, ktorý zefektívnil proces vytvárania orámovania. Za účelom zredukovania ceny orámovania sa vývoj sústredil na dve stratégie. Prvá, je učenie čiastočne využívajúce učiteľa kde napr. učenie rozpoznávania je iba na trénované na triedach, ktoré sú prítomné na obrázku bez bližšej špecifikácie. Lenže takýto model má iba polovičnú presnosť ako keby sme ho učili z orámovaných vzorov [9,10,11,12,15]. Druhá stratégia je aktívne učenie, kedy pre vytvorenie vzorov trénovania potrebujeme od človeka vytvoriť rám okolo objektu. Táto stratégia produkuje vysokú kvalitu detektorov, ale stále požaduje od ľudí kresliť rám okolo objektu a je tak limitovaná časom anotácie [13,14]. V tomto článku autori ukázali novú schému ako učiť detektor iba tak, že automaticky vytvára návrhy anotácii a ľudská interakcia je potrebná iba na overenie orámovania či je správne alebo nie. Odpoveď na túto otázku zaberie ďaleko menej času než nakreslenie orámovania. Schéma je založená na iteratívnej úprave objektov detekcie. V každej iterácii je použitý kontrolný signál od anotátora dvoma spôsobmi. Prvý spôsob spočíva v menení objektu iba vtedy keď je informácia od anotátora kladná. Tým pádom overený vzor už nemusí byť zaradený do ďalšej iterácie. Druhý spôsob je, že orámovanie, ktoré je označené za chybné, môže byť použíté na určenie priestoru kde sa objekt nenachádza a zredukovať tak priestor, ktorý sa má prehľadávať. Oba tieto spôsoby pomáhajú zvýšiť efektivitu hľadania objektov v ostávajúcich obrázkoch a minimalizujú tak ľudské úsilie, pretože eliminujú potrebu kresliť orámovanie.

Weakly-supervised lokalizácia objektu (WSOL) - Mnoho predchádzajúcich techník sa pokúšalo učiť detektory objektov s čiastočnou informáciou (weakly supervised). Napr. tréningové obrázky, ktoré obsahujú príklady istých tried objektov, ale nie ich lokáciu. Úloha je lokalizovať tieto objekty v tréningových obrázkoch a učiť detektor lokalizovať inštancie v nových obrázkoch.
Toto sa typicky deje iteratívnym striedaním medzi (A) znovu trénovaním objektového detektoru na dátach s aktuálnou selekciou pozitívnych inštancií a (B) znovu lokalizovaním rámov inštancií v pozitívnych obrázkoch použitím súčasného objektového detektora.

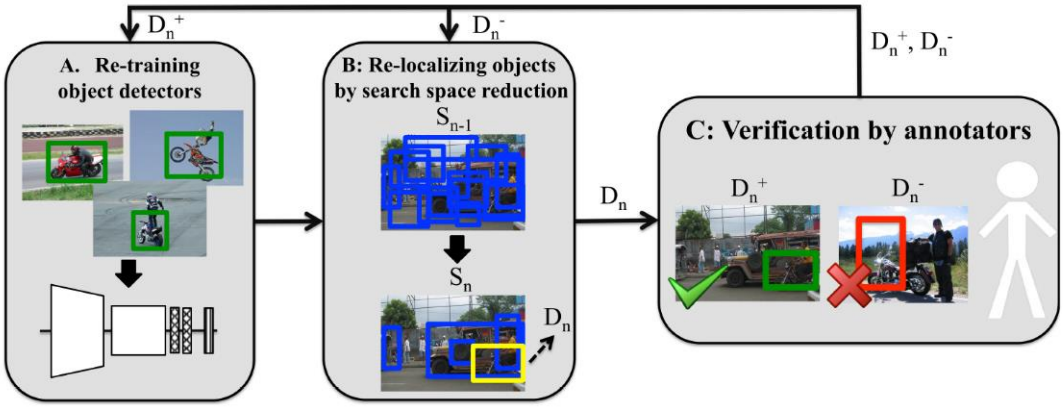
Človek v slučke (human in the loop) – Prístupy spolupráce ľudí a strojov boli úspešne použité v úlohách, ktoré sú aktuálne príliž náročné aby boli riešené len počítačovým videním, ako napr. jemnozrnné rozpoznávanie (fine grained) [16, 17], zhlukovanie s čiastočným dohľadom (semisupervised clustering) [18] a na atribútoch založená klasifikácia obrazu[16]. Tieto práce kombinujú výstupy predtrénovaných počítačových modelov na novom testovacom obrázku, s ľudským úsilím na to aby vyriešili úlohu. V doméne detekcie objekov, Russakovsky a kol. [19] navrhli takú schému, aby plne detekovala všetky objekty v obrázkoch komplexných scén. Avšak je dôležité poznamenať že ich detektory sú predtrénované na orámovaných objektoch z datasetu ILSVRC [7], nakoľko ich cieľom nie je vytvoriť tréningovú schému ktorá znižuje prácu na anotovaní.

Aktívne učenie - schémy aktívneho učenia iteratívne trénujú modely a v iteráciách požadujú ľudskú anotáciu pre sadu dát, ktorá je najviac zaujímavá pre učiaci sa algoritmus. Predchádzajúce práce o aktívnom učení boli zameraná na klasifikáciu obrazu [18,19], freeform region labelling [20,21].

Iné spôsoby redukcie anotačného úsilia – Niektorý autori sa pokúšali učiť detekciu objektu z videí, kde priestorová a časová súdržnosť snímok uľahčuje objektovú lokalizáciu [22]. Alternatívou je transferové učenie (transfer learning), kde učenie modelu pre novú triedu je podporované príbuznými triedami [23]

Metóda "nepotrebujeme orámovanie" – Na začiatku je daný dataset obrázkov anotovaný triedami na úrovni celého obrázka. Cieľom je získať objekty anotované orámovaním a natrénovať dobré detektory za minimalizovania ľudského anotačného úsilia. Autori navrhli schému kde anotátori potrebujú skontrolovať orámovanie ktoré je automaticky vytvorené učiacim sa detektorom. Ich schéma iteratívne strieda medzi (A) znovu natrénovaním objektových detektorov, (B) znovu lokalizovaním objektov v tréningových obrazoch a (C) žiadaním anotátorov o kontrolu (obrázok 2). Dôležité je že sa používa výsledok overenia aby pomohol znovu trénovať a znovu lokalizovať objekty.

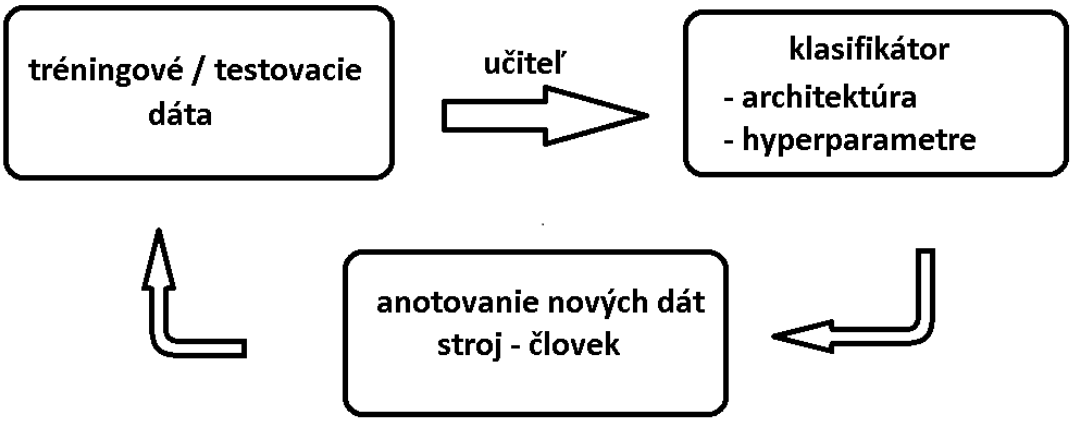
Nech I_n je sada obrázkov pre ktoré ešte nemáme pozitívne overné orámovanie na iterácií *n*. Nech S_n je zodpovedajúca sada možných objektových lokácií. . Na začiatku I_0 je kompletná tréningová množina a S_0 je množina navrhovaných objektov extrahovaných z obrázkov. Je to Na iterácii *n* máme sadu automaticky detekovaných orámovaní D_n ktoré sú dané anotátorom aby boli skontrolované. Detekcie ktoré sú posúdené ako správne D_n^+ sú použité pre znovunatrénovanie detektora (A) v ďalšej iterácii. Výsledok kontrolovania je tiež použitý na redukovanie priestoru S_{n+1} ktorý sa prehľadávania pre znovulokalizáciu (B).



Obrázok 2. Zdroj [7]. Metóda navrhovaná autormi iteruje medzi (A) znovu naučením detektorov, (B) znovu lokalizovaním objektov a (C) žiadaním anotáorov o kontrolu. Výsledný signál z (C) je použitý v oboch krokoch (A) aj (B)

Metóda dolovania tried v čiastočne anotovaných dátach

Pri riešení klasifikačných úloh okrem správnej zvolenej architektúry a algoritmu je veľmi dôležitou častou získanie potrebného množstva dát ale aj ich kvality (t.j mať správne nastavené atribúty podľa ktorých chceme klasifikovať.) Existujú situácie kedy zastúpenie niektorých typov dát je veľmi málo lebo sa reálne nevyskytujú a je potrebné úrehladať obrovské množstvo kandidátov, prípadne počiatočné dáta pri riešení úlohy nie sú dostatočné. Získanie nových dát môžeme realizovať buď anotovaním pomocou človeka alebo vytvoriť metódu ktorá pomáha človeku tieto dáta anotovať a to buď automatickým alebo poloautomatickým spôsobom. Pri poloautomatickom spôsobe predpokladáme že anotátor dostane vopred vyplnené atribúty dát neurónovou sieťou a ten výsledky potvrdí prípadne upraví. V taktomto prípadne je efektivita anotovania vyšia. Ostatné dáta ktoré nie sú problémové a výstup pokladáme za dostatočne dôveryhodný nie je potrebný zásah človeka a anotovanie sa stáva rýchlejšie. Vtedy môžeme hovoriť o automatickom anotovaní. Otázka je kedy si môže byť algoritmus natoľko istý aby nemusel požadovať potvrdenie od človeka? Existujú rôzne prístupy pomocou ktorých by sa dala takáto podmienka vyhodnotiť. My sme sa rozhodli použiť binárne kódovanie ktoré nám na výstupe vytvára nové rozdelenie priestoru a na základe tejto informácie sa pokúšame vyhodnotiť či danný výstup je dôveryhodný alebo nie. Pri počiatočnom anotovaní by mal mať algoritmus prísne kritéria aby nedochádzalo k zanášaniu chýb do tréningovej množiny. U takto nastavených kritériách je možné že dáta z ktorých sa bude dolovať budú akceptované len v malej miere a pre zníženie interakcie s človekom je lepšie pustiť algoritmus niekoľko krát, ako naraz nechať anotátora opravovať veľké množstvo dát. Takýto iteratívny prístup je preto základom celého dolovania dát. Ako vidno na obrázku 3. schéma je jednoduchá. V Prvom kroku vychádzame z počiatočných dát kedy natrénujeme CNN. Takto natrénovanú sieť potom môžeme použiť na neanotovaných alebo na čiastočne anotovaných dátach ktorým môže chýbať atribút, prípadne kedy chceme overiť správnosť dát z predchádzajúceho anotovania. Výstup potom podľa dôveryhodnosti ponúkneme anotátorovi alebo ho rovno zaradíme do trénovacej množiny.



Obrázok 3 Schéma iteračného postupu

Anotovanie dát na základe miery istoty predikcie

V predchádzajúcih kapitolách boli spomínané metódy, ktoré pomáhaju zredukovať cenu anotovania čo najviac tak, že anotátor robí čo najmenej zložitej práce (ako kreslenie orámovania objektu alebo výber triedy objektu z veľkého množstva možností) a tak zvýši svoju efektivitu. V našom prístupe sme sa zamerali na vytvorenie *anotovaného* datasetu špeciálne pre úlohy klasifikácie (na jednom celom obrázku je len jeden objekt). Testovali sme dve anotačné stratégie, ktoré sme nazvali *vysoká miera istoty anotácie* HCA (anglicky high confidence annotation) a *nízka miera istoty anotácie* LCA (anglicky low confidence annotation). V oboch stratégiách bol dataset vytvorený iteratívnym spôsobom tak, že sa postupne trénoval klasifikátor a následne sa použil jeho výstup na anotovanie nových ešte nevidených dát. Stratégia, ktorá vedie ku datasetu s vyššou klasifikačnou presnosťou pri danom množstve práce vynaloženej pri jeho tvorbe, je považovaná za lepšiu. Obe stratégie sa spoliehajú na manuálnu anotáciu, čo je aj kľúčová časť pri vytváraní spoľahlivého datasetu. Náš prístup k anotovaniu kandidátov je podobný ako u spomínaných metód (vybratie kandidátov a následné anotovanie) s tým, že využíva hodnotu predikcie, ktorá riadi výber anotovaných dát a zjednodušuje anotáciu navrhovaním správnej triedy. Zatiaľ čo HCA poskytuje presnejší výsledok triedy, LCA by mala identifikovať vzory s vysokou informačnou hodnotou pre dataset. Tieto stratégie je možné rozšíriť použitím ďalších prístupov, ktoré riešia problémy s nedostatkom údajov a pretrénovaním, ako sú napr umelé dáta, ktoré sme v experimentoch použili. Vyhodnocovanie bolo založené na troch kritériách: počet všetkých anotácií, počet komplexných anotácií, a počítačový čas ktorý bolo treba na použitie danej stratégie. Obe stratégie sú porovnávané so základnou metódou založenou na náhodnom anotovaní vzorov (vzor, ktorý sa pridá do trénovacej množiny je vybraný náhodne).

Metóda

Anotačný proces je iteratívny. V každej iterácii je klasifikátor naučený na aktuálnych dátach pustený na množinu neanotovaných kandidátov. Pre každého kandidáta je získaná miera istoty predikcie anotácie (maximálna hodnota zo softmax) ako aj trieda (argmax v softmax). Takto predpovedaná hodnota dôvery je použitá pre výber nových dát z anotácií. V HCA stratégii sú dáta vybrané začínajúc vzorom s najväčšou mierou istoty. Tieto môžeme považovať ako najjednoduhšie prípady pre klasifikátor. Preto navrhovaná trieda od klasifikátora je častejšie správna než pri LCA stratégii, kedy sú vyberané vzory s malou

mierou istoty. Nakoniec sú vybrané vzory manuálne anotované a použité na trénovanie nového klasifikátora. Tento proces sa iteratívne opakuje. Naše porovnávanie s náhodným anotovaním (RA) je vykonané tak, že sa náhodne vyberú dáta, na ktorých sa natrénuje klasifikátor, a výsledky sa porovnajú s navrhovanými stratégiami.
Za účelom vyhodnotenia sme definovali dva typy anotácií: *jednoduchú* a *komplexnú.* Keď sa klasifikátor použije na dáta, ktoré nevidel, jeho výstup predpovedá mieru istoty a triedu pre každý vzor. V priebehu anotácie je operátorovi predložený vzor a zároveň je mu tiež poskytnutý kandidát na triedu. Ak je kandidát na triedu správny, anotátorovi stačí odpovedať, že daný kandidát je správny. Toto nazývame *jednoduchá anotácia.* Vo veľa klasifikačných úloh potvrdenie že kandidát triedy je správny je výrazne jednoduchšia úloha ako vybrať správnu triedu. Ak je kandidát nesprávny, alebo ho nie je možné poskytnúť, operátor musí vybrať správnu triedu. Toto voláme *komplexná anotácia*, nakoľko ide o náročnejšiu úloha, hlavne v prípade úloh s veľkým počtom tried. Všetky anotácie v *náhodnej* stratégii (RA), validačné dáta použité pre zastavovanie tréningu a počiatočné tréningové dáta, z ktorých začínaju HCA a LCA stratégie, sú považované za *komplexné anotácie.* Počet všetkých anotácií je našim hlavným kritériom na meranie práce potrebnej na vytvorenie datasetu. Obsahuje obe, jednoduché aj komplexné, anotácie. Počet komplexných anotácií je dobrý indikátor pre úlohy, kde vybratie správnej triedy pre vzor je výrazne viac namáhavá práca ako skontrolovať, či kandidát na triedu je správny, alebo nie. Počítačový čas potrebný na anotovanie je pomocný indikátor na meranie ako je anotačná stratégia výpočtovo zložitá. . Je počítaný ako počet všetkých tréningových dát použítých vo všetkých epochách a iteráciách

2. Kódovanie výstupov neurónovej siete

Skôr než vysvetlíme použitie detekčných a opravných kódov pre rozhodovanie o anotácii dát, poukážeme na hlavné rozdiely medzi najčastejšie používaným spôsobom kódovania výstupov neurónovej siete, ktorým je one-hot kódovanie a všeobecným binárnym kódom na ktorom sú založené binárne detekčné a opravné kódy. Termín one-hot kód ponecháme v angličtine, pretože v slovenskej literatúre nemá výstižný preklad a aj slovenské učebnice ho takto používajú. Znamená to kód, ktorého kódové slová majú jeden bit jednotkový a ostatné bity nulové. To znamená, že na zakódovanie dát do n tried potrebujeme n bitov. Pretože Hammingova vzdialenosť medzi kódovými slovami one-hot kódu sú dva bity, kód nemá dobré opravné schopnosti. Avšak tým, že bity v kódovom slove sú viazané takýmto silným pravidlom (nazveme to holistickou vlastnosťou kódových slov), má veľmi dobré vlastnosti pre trénovanie neurónovej siete. Použitím metódy softmax na spracovanie výstupov neurónovej siete sa ľahko určí jednotkový bit, ktorý sa priradí výstupu po určitej transformácii s najvyššou hodnotou a ostatným bitom sa priradia nuly. Ak navyše transformované výstupy normujeme na tvar rozdelenia pravdepodobnosti, trénovanie konvolučných sietí spätným šírením gradientu sa ukázalo veľmi úspešným. Kódové slová one-hot kódu vytvárajú ortogonálnu bázu.
Môžeme však použiť aj nejednotkové ortogonálne bázy na tvorbu kódových slov. Takým príkladom je použitie Hadamardových kódových slov [24]. Hadamardova matica vytvára kódové slová pomocou rekurentného predpisu

$$\mathbf{H}_{i+1} = \begin{pmatrix} \mathbf{H}_i & \mathbf{H}_i \\ \mathbf{H}_i & -\mathbf{H}_i \end{pmatrix}, \mathbf{H}_0 = (1), \qquad (1)$$

kde na konci sú hodnoty “-1” nahradené hodnotami “0”. Okrem prvého riadku, v ktorom sú všetky prvky nulové, ostatné riadky obsahujú polovicu bitov jednotkových a polovicu bitov nulových. Užitočnou vlastnosťou Hadamardovej matice je aj ekvidištantné rozdelenie kódových slov v kódovom priestore s Hamingovou vzdialenosťou $2n - 1$ bitov medzi kódovými slovami, kde $2n$ je počet výstupov neurónovej siete. Poznamenajme, že počet tried

musí byť menší alebo rovný počtu výstupov siete. Ako upozorňuje [25], aj stĺpce (okrem prvého) Hamingovej matice majú rovnaký počet núl a jednotiek, nakoľko transpozíciou Hamingovej matice dostávame tú istú maticu. Túto vlastnosť sa snažili zachovať aj v prípade, že počet tried je menší ako počet riadkov v matici.

Všeobecné binárne kódovanie umožňuje vyššiu kapacitu kódu než one-hot alebo Hadamardove kódovanie. V uvedených dvoch kódoch sú kódové slová ortogonálne a teda vytvárajú bázu kódového priestoru, binárne kódovanie vytvára kódové slová aj lineárnou kombináciou bázických slov. Literatúra uvádza, že one-hot kódy sú menej odolné voči adverziálnym útokom [26], zatiaľ čo binárne kódy (napr. multi-way kódy) sú odolné viac [30]. Priradenie kódových slov jednotlivým kategóriám však nie je také priamočiare ako u one-hot kódu. Dobre preštudovaným prípadom je Error Correcting Output Code (ECOC) predstaveným v [27]. Rozpoznanie každého bitu výstupného kódu (ktorý je určený maticou kódu) je trénované zvlášť na binárnom rozdelení vstupných dát alebo príznakov. Cieľom je dosiahnuť nezávislosť bitov. ECOC kódové slová sú potom množinou nezávislých bitov, v ktorej každý bit indikuje prítomnosť odpovedajúceho príznaku v dátach. V tejto práci sa zameriavame na viactriedne vzory (multi-class), kde trieda nie je priradená jednotlivým bitom, ale kódovému slovu ako celku. Pripustenie závislosti medzi bitmi umožňuje využitie vzájomnej závislosti medzi bitmi podobne ako v softmaxe.

Hlavnou výhodou použitia binárnych kódov je vynikajúca úroveň poznania, ktorú dosiahla teória kódovania. Používame lineárne blokové kódy pre ich schopnosť oddelenia informácie o triedach od zabezpečovacej časti kódového slova. Pre porovnanie vlastností one-hot kódu a binárnym kódom sme zvoli desať bitové kódové slova. Akceptovanie prijatého kódového slova ako správneho je založené na výpočte syndrómu. Chybu detekujeme v prípade, že syndróm nie je nulový [29], alebo je nulový, ale kódové slovo nie je priradené žiadnej triede (napr. v našom prípade desiatich tried ide o šesť slov zo šestnástich). Ak toto pravidlo určí kódové slovo ako chybné, považujeme ho za nedôveryhodné a pri anotácii sa k nemu správame podľa stratégií popísaných v predchádzajúcich kapitolách. V tejto práci analyzujeme použitie (10, 4) blokových systematických cyklických kódov ako príkladu všeobecných binárnych kódov. Metodika hľadania suboptimálneho binárneho kódu je načrtnutá v neskoršej podkapitole, ale jej overenie sme nestihli a ponechávame ho na ďalší výskum. Tabuľky 1. predstavuje generujúce polynómy študovaných 10 bitových cyklických kódov. Všetky polynómy sú nutne ireducibilné. Prvých päť polynómov je aj primitívnych, čo sa v tomto prípade ukazuje ako nepodstatné.

CRC1: 100,0011	CRC2: 101,1011	CRC3: 110,0001	CRC4: 110,0111	CRC5: 111,0011
CRC6: 100,1001	CRC7: 101,0111	CRC8: 110,1101	CRC9: 111,0101	

Tabuľka.1. Binárny tvar generujúcich polynómov použitých cyklických kódov CRC.

Rozhodovacie metódy

Ako priradiť dáta odpovedajúcim triedam? Pokiaľ samotná neurónová sieť implementuje nelineárnu transformáciu $\mathcal{R}^m \rightarrow \mathcal{R}^n, m > n$, úlohou klasifikácie je priradiť vstupnej vzorke triedu z konečnej množiny, v našom prípade predstavenú kódovým slovom. Základnou otázkou je, ktoré ktoré slovo sa najviac podobá získanému výstupu neurónovej siete. Hoci kódové slovo je binárne, môžeme ho považovať za bod v Euklidovom vektorovom priestore. Potom môžeme nájsť k vektoru výstupu najbližší vektor spomedzi kódových slov, v závislosti na zvolenej metrike. Takýto spôsob rozhodovania založenom na vzdialenosti je postačujúci, pokiaľ rozhodnutie o triede je posledným krokom a nepotrebujeme ho pre ďalšie úkony. Na priradenie kódového slova výstupu sa môžeme pozerať ako na logický výrok a teda môžeme naň aplikovať pravidlá logiky a skúmať ďalšie vlastnosti výstupu neurónovej siete. Na

binarizované výstupy sa môžeme pozerať aj ako na kódové slová a použitím skalárov nad Galoisovým poľom vytvoriť konečno-rozmerný vektorový (kódový) priestor. Pomocou teórie kódovania potom môžeme študovať ďalšie vlastnosti výstupov. Avšak táto bohatá sada logických a kódovacích nástrojov je k dispozícii až po binarizácii výstupov neurónovej siete. Takýmto preklopením z Euklidovho vektorového priestoru do kódového však strácame informáciu o podobnosti výstupu a kódového slova. Aby sme ju zachovali a spojili výhody obidvoch priestorov, pozeráme sa na výsledok algebraickej operácie ako výrok, a určujeme jeho mieru pravdivosti použitím fuzzy logiky. Tento prístup nám napríklad umožní priradiť pravdivostnú hodnotu syndrómom a ukážeme, že najpravdivejšie kódové slovo má najpravdivejší syndróm. Ako bude uvedené neskôr, zvolili sme Zadehovu (štandardnú) fuzzy logiku a $t \in [0,1]$ je mierou pravdivosti.

Preto prvým krokom je normalizácia individuálnej hodnoty výstupu neurónovej siete. Nech odozvou neurónovej siete na daný vstup je $\boldsymbol{y} = (y_0, \dots, y_{n-1}) \in \mathcal{R}^n$, a $\tilde{\boldsymbol{y}} = (\tilde{y}_0, \dots, \tilde{y}_{n-1}) \in [0,1]^n$ je normalizovaný výstup, kde *n* je počet výstupov / bitov (*n*=10 v našom prípade). Používame tri druhy normalizácie pre *i = 0,…,n*−1:

$$\text{sigmoid } \tilde{y}_i = \sigma(y_i) = \frac{1}{1 + e^{-y_i}} \tag{2}$$

$$\text{lineárna } \tilde{y}_i = \frac{y_i - y_{min}}{y_{max} - y_{min}}, y_{min} = \min\{y_0, \dots, y_{n-1}\}, y_{max} = \max\{y_0, \dots, y_{n-1}\} \tag{3}$$

$$\text{vektorová } \tilde{\boldsymbol{y}} = \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|_2} \tag{4}$$

Po normalizácii priraďujeme vektor výstupu kódovému slovu. Najjednoduchším spôsobom, ktorý je rozšírením jednorozmerného prípadu, je rozhodnúť o každom bite nezávisle

$$c_i = \begin{cases} 0, \tilde{y}_i < 0.5 \\ 1, \tilde{y}_i \geq 0.5 \end{cases}, i = 0,1, \dots, n-1. \tag{5}$$

Túto metódu nazývame "prahová bitová metóda". Z nášho pohľadu budeme výstupnú hodnotu považovať za pravdivosť výroku, že daný výstup určuje bitovú zvolenú hodnotu. Podrobnejšie to vyjadruje vzťah (30).

Zložitejšie prístupy, ktoré nazveme holistickými, berú výstupný vektor $\tilde{\boldsymbol{y}}$ ako celok. Najpopulárnejší je softmax prístup založený na one-hot kódovaní, v ktorom celistvosť kódového slova je opretá o štruktúru v ktorej všetky bity sú rovné nule s výnimkou jedného, ktorý je jednotkový. Po exponenciálnej transformácii

$$\tilde{y}_i \leftarrow \frac{e^{\tilde{y}_i}}{\sum_{j=0}^{n-1} e^{\tilde{y}_j}}. \tag{6}$$

normujeme hodnoty na rozdelenie pravdepodobnosti. Najvyššej hodnote priradíme jednotkový bit, ostatným priradíme nulové bity. V [28] je predstavená metóda, ktorá kombinuje ECOC so softmaxom. Metóda nepoužíva detekčné kódovanie. Implementovali sme túto metódu s nahradením ECOCu cyklickým kódom, aby sme ohodnotili prínos použitia Zadehovej fuzzy logiky pri rozhodovaní.

Na vysvetlenie ako používame binárne kódové slová, začneme s dvojhodnotovou logikou. V tomto Booleovom prípade je výstup $\tilde{\boldsymbol{y}}$ mapovaný na kódové slovo $\tilde{\boldsymbol{c}} = (\tilde{c}_0, \dots, \tilde{c}_{n-1}) \in \{0,1\}^n$ a každá trieda *N* je priradená k reprezentujúcemu kódovému $\boldsymbol{c^N} = (c_0^N, \dots, c_{n-1}^N)$, $N \in \{0,1, \dots, n-1\}$. Vstupná vzorka bude klasifikovaná ako vzor triedy *N*, vtedy a len vtedy ak $\tilde{\boldsymbol{c}} = \boldsymbol{c^N}$, t.j. $\wedge_{i=0}^{n-1}(\tilde{c}_i = c_i^N)$. V Booleovej logike sa kódové slovo rovná kódovému slovu triedy *N* vtedy a len vtedy, ak pravdivostné hodnoty zhody všetkých bitov porovnávaných kódových slov sa rovnajú jednej

$$S^N = \wedge_{i=0}^{n-1}(\tilde{c}_i = c_i^N) \Rightarrow t(S^N) = \prod_{i=0}^{n-1} t(\tilde{c}_i = c_i^N) = 1, N \in \{0,1, \dots, 2^n - 1\} \tag{7}$$

kde $S^N = (\tilde{\boldsymbol{c}} = \boldsymbol{c^N}), \tilde{\boldsymbol{c}}, \boldsymbol{c^N} \in \{0,1\}^n, t(true) = 1, t(false) = 0.$

Aby sme zachovali tento holistický prístup k rozhodnutiu o celom kódovom slove, nahradili sme Booleovu logiku Zadehovou (štandardnou) fuzzy logikou a $t \in [0,1]$ je pravdivostná hodnota výroku. pravdivostná hodnota v Zadehovej fuzzy logike konjunkcie, disjunkcie a negácie výrokov A, B je

$$\begin{aligned} \mathrm{t}(\mathrm{A} \wedge \mathrm{B}) &= min\big(\mathrm{t}(\mathrm{A}), \mathrm{t}(\mathrm{B})\big), \\ \mathrm{t}(\mathrm{A} \vee \mathrm{B}) &= max\big(\mathrm{t}(\mathrm{A}), \mathrm{t}(\mathrm{B})\big), \\ \mathrm{t}(\overline{\mathrm{A}}) &= 1 - \mathrm{t}(\mathrm{A}). \end{aligned} \tag{8}$$

Hodnotu *i*-teho výstupu $\tilde{y}_i \in [0,1]$ budeme interpretovať ako pravdivostnú hodnotu. Aby sme určili správnosť bitu nezávisle na jeho hodnote, použijeme ako mieru zhody mieru pravdivosti [31] výroku $\tilde{y}_i = a, a \in \{0,1\}$. Aby sme sa vyhli nerozhodnuteľným výrokom (ak $\tilde{y}_i = 1/2$), pripočítame v tomto prípade k miere pravdivosti zanedbateľnú hodnotu $\varepsilon > 0$ a teda $t(\tilde{y}_i = a) \neq 1/2$

$$\mathrm{t}(\tilde{\mathrm{y}}_\mathbf{i} = \mathrm{a}) = \begin{cases} \tilde{\mathrm{y}}_\mathbf{i}, & \mathrm{a} = 1, \tilde{\mathrm{y}}_\mathbf{i} \in [0, ½) \cup (½, 1] \\ 1 - \tilde{\mathrm{y}}_\mathbf{i}, & \mathrm{a} = 0, \tilde{\mathrm{y}}_\mathbf{i} \in [0, ½) \cup (½, 1] \\ ½ + \varepsilon, & \mathrm{a} = 1, \tilde{\mathrm{y}}_\mathbf{i} = ½, 0 < \varepsilon \ll 1 \\ ½ - \varepsilon, & \mathrm{a} = 0, \tilde{\mathrm{y}}_\mathbf{i} = ½, 0 < \varepsilon \ll 1 \end{cases}. \tag{9}$$

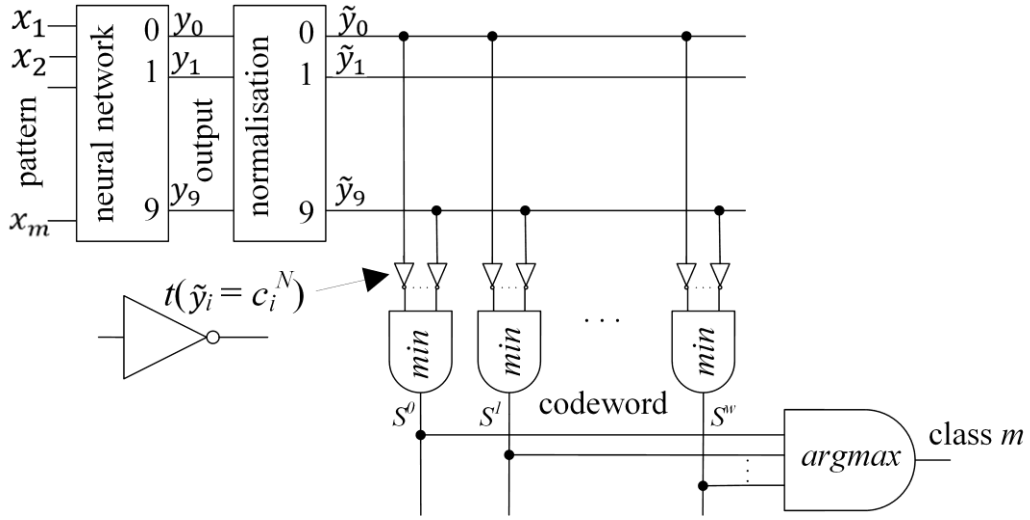
Pri implementácii nemusíme túto ochranu používať, pretože pravdepodobnosť nadobudnutia hodnoty $\tilde{y}_i = ½$ na spojitom jednotkovom intervale je nulová. Potom pri použití Zadehovej fuzzy logiky

$$t(S^N) = t(\tilde{\boldsymbol{c}} = \boldsymbol{c}^N) = \min_{i=0,\dots,n-1} t(\tilde{c}_i = c_i^N),\ N \in \{0,1,\dots,2^n - 1\}. \tag{10}$$

Notácia (31) je platná aj v prípade použitia Booleovej logiky, pretože

$$\min_{\mathrm{i}=0,\dots,\mathrm{n}-1} \mathrm{t}\big(\tilde{\mathrm{c}}_\mathrm{i} = \mathrm{c}_\mathrm{i}^\mathrm{N}\big) = \prod_{\mathrm{i}=0}^{\mathrm{n}-1} \mathrm{t}\big(\tilde{\mathrm{c}}_\mathrm{i} = \mathrm{c}_\mathrm{i}^\mathrm{N}\big), \text{ for } \mathrm{t}\big(\tilde{\mathrm{c}}_\mathrm{i} = \mathrm{c}_\mathrm{i}^\mathrm{N}\big) \in \{0,1\}. \tag{11}$$

To je aj dôvod, prečo na obrázku. 4 je použitá funkcia minima miesto násobenia aj v prípade Booleovej logiky.



Obr.ázok 4. Zdroj [33]. Navrhnutá klasifikačná schéma založená na Zadehovej fuzzy logike.

3. Experimenty využívajúce detekčné CRC kódy pre anotáciu

V naseledujúcom texte sú predstavené výsledky klasifikátorov na rôznych architektúrach pre vyhodnocovacie datasety ktoré sú v rámci aj mimo trénovaciu doménu. Znamená to že obrázky na ktorých sa model trénuje považujeme za dáta „*z rozdelenia*“ a obrázky ktoré pochádzajú a prezentujú inú doménu považujeme za dáta „*mimo rozdelenia*“. Pre základné vyhodnotenie sme používali meranie presnosťi (12) a spoľahlivosťi (13).

Vychádzali sme z dvoch základných stavov: „*akceptovaný*“ a „*odmietnutý*“ vzor. Keď sa vykoná rozhodnutie o neznámom vzore, najprv sa akceptuje alebo zamietne. Keď je vzor akceptovaný, určí sa trieda kam má patriť. Vzor je vždy pozitívny (true class label) alebo negatívny (false class label). Zamietnutý znamená že klasifikátor neurčí jeho triedu. Pre tieto

	Akceptovaný (A)		**Zamietnutý (R)**	**Σ**
	Pozitívny (P)	**Negatívny (N)**		
True (T)	TP	FN	FR	T
False (F)	FP	TN	TR	F
Σ	P	N	R	Q

Tabuľka 2 Konfúzna matica

dôvody máme 6 možných stavov zobrazených v tabuľke 2. Pre dáta mimo rozdelenia sú iba dva stavy: chybne pozitívny (false positive) a správne zamietnutý (true rejected). Pre tieto dáta používame meranie (16)

$$\text{Presnosť } p_{acc} = \frac{TP+TN}{Q} \tag{12}$$

$$\text{Spoľahlivosť } p_{conf} = \frac{TP+TN}{P+N} \tag{13}$$

$$\text{Dôvera } p_{conf} = \frac{TP}{P} \tag{14}$$

$$\text{Akceptácia } p_A = \frac{P+N}{Q} \tag{15}$$

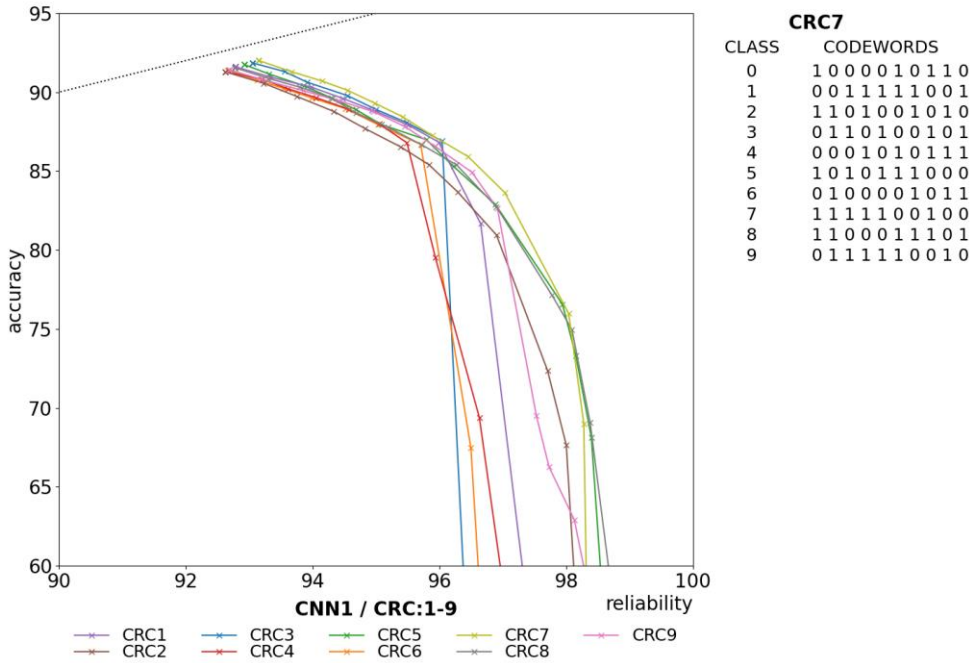
$$\text{Zamietnutie } p_R = \frac{R}{Q} \tag{16}$$

$$p_{acc} = p_{conf} p_A = p_{conf}(1 - p_R) \tag{17}$$

$$\frac{TP+TN}{Q} = \frac{TP+TN}{P+N} * \frac{P+N}{Q} \tag{18}$$

S ako prvými sme začali experimentovať s lineárnymi blokovými kódmi CRC. Výsledky pre CNN-4 architektúru s lineárnou normalizáciou výstupov a Zadehovou rozhodovacou funkciou sú na obrázku 5. Presnosť je počítaná podľa (12). Pre meranie spoľahlivosti (13) sme použili detekčné schopnosti binárnych kódov tak, že sme použili celý kódový priestor využívajúc 1024 kódových slov namiesto 10 tried. Ak výstupný kód nie je zhodný s desiatimi kódovými slovami priradené triedam, vzor je zamietnutý ako nedôveryhodný. Jeden bod na obrázku 5 znázorňuje *presnosť - spoľahlivosť* pre jednu prahovú hodnotu aplikovanú v celom rozhodovacom procese. Rôzne body môžu byť dosiahnuté s rôznymi prahovými hodnotami. Ak žiadny vzor nie je zamietnutý, potom presnosť je rovná spoľahlivosti. Tomu korešponduje

bodkovaná čierna čiara v ľavo hore. CRC kódy môžu detekčnou schopnosťou zamietnuť nedôveryhodný vzor a preto môžu mať spoľahlivosť vyššiu aj pri nulovej prahovej hodnote. Výsledky spoľahlivosti klasifikátora môžu byť zlepšené nastavením vyššieho prahu, výmenou za nižšiu presnosť. Experimenty sme pustili na každom CRC kóde a vytvorili graf presnosti a spoľahlivosti. Je možné vidieť, že každý kód dáva inú presnosť a to ponúka ďalšie možnosti ako hľadať najvhodnejší kód. Čo nebolo cieľom hlbšieho študovania v tejto práci. Pre naše účeli sme vybrali kód CRC7 ktorý mal najlepšiu presnosť.



CRC7

CLASS	CODEWORDS
0	1 0 0 0 0 1 0 1 1 0
1	0 0 1 1 1 1 1 0 0 1
2	1 1 0 1 0 0 1 0 1 0
3	0 1 1 0 1 0 0 1 0 1
4	0 0 0 1 0 1 0 1 1 1
5	1 0 1 0 1 1 1 0 0 0
6	0 1 0 0 0 0 1 0 1 1
7	1 1 1 1 1 0 0 1 0 0
8	1 1 0 0 0 1 1 1 0 1
9	0 1 1 1 1 1 0 0 1 0

Obrázok 5 Charakteristika presnosti a spoľahlivosti pre rôzne CRC (v ľavo). CRC7 kódové slovo (v pravo)

V ďalšej sekcií popisujem náš hlavný experiment. Vyhodnotili sme rôzne architektúry, výstupné kódovanie a rozhodovanie klasifikátora na základe presnosti a spoľahlivosti. Všetky experimenty sme robili na CIFAR-10 datasete. Všeobecne v metodológii kllasifikačnej úlohy sa priraďuje každému výstupu neurónovej sieti jedna trieda počas trénovania a testovania. Pri použití detekčných kódov môžeme rozšíriť rozhodovanie na celý kódový priestor. Použili sme 10 bitový výstupný kód pre všetky prípady zaznamenané v tabuľke 3 okrem Hadamard kódovania, ktorý požaduje minimálne 15 bitov na pokrytie 10 tried a zaistenie tak vlastností opísaných v [32]. To znamená že celý kódový priestor je 1024 možností pre CRC7 a 10 pre one-hot kódovanie. Presnosť (12) sa používa na meranie výkonu vtedy keď všetky kódové slová (výstupné kódy patriace k triedam) patria triedam. Pri použití celého kódovacieho priestoru, môžeme dostať výstupné kódy, ktoré nepatria žiadnej triede. V tomto prípade je použitá detekcia chyby a vzor je odmietnutý ako nedôveryhhodný. Pre meranie tejto charkteristiky sme zaviedli pojem FCA (full code accuracy), ktorý sa používame keď chceme hovoriť o presnosti na plnom kódovom priestore.

Hoci ako primárny cieľ je určiť vlastnosti binárneho kódovania, metóda rozhodovania značne ovplivňuje správnosť klasifikácie a schopnosť použiť plný kódový priestor. V tabuľke 3 porovnávame presnosť podľa bitového rozhodnutia (5) a holistického rozhodnutia slova reprezentovaného euklidovskou vzdialenosťou aplikovanej na Hadamard kóde, softmax použitý na on-hot kóde ako aj navrhnuté Zadeh rozhodnutie (aplikované na one-hot a CRC kóde). Najskôr sme testovali všetky prístupy s CNN-4 architektúrou. Pre každú kombináciu sme robili 10 opakovaní tréningu a použili výsledok s najlepšou presnosťou. Potom sme vybrali 4 najlepšie prístupy podľa presnosti a testovali ich na CNN-5 a ResNet20v2

architektúre. Vykonali sme 5 opakovaní tréningu pre každú architektúru a použili najlepší výsledok podľa presnosti.
Náš hlavný záujem je určiť možnosť použitia binárneho kódovania (repreznetovaného CRC7) s chybovou detekčnou schopnosťou. Najskôr sme testovali prístup prahovej bitovej hodnote (5) nakoľko to je priamy prístup rozhodovaniu. Pre tento prístup bola ako účelová funkcia použitá MSE a bitový prah počas testovania. Ďalej sme testovali Zadeh rozhodovanie na CRC7 a one-hot kódovanií. Použitie Zadeh rozhodovania s one-hot kódovaním je ako použitie binárneho kódovania, kde kódové slová majú one-hot vlasnosti. Pre porovnanie používame one-hot kódovanie so softmax normalizáciou ako základnú metódu. Inšpirovaný [28] sme tiež testovali kódovanie CRC7 so softmax. To umožňuje kombinovať výhody binárnych kódov a softmax rozhodovania ale pri neschopnosti detekcie chýb. Napokon sme otestovali prístup [32] kde euklidovské rozhodnutie je použité na Hadamard kódovaní. Výsledky naznačujú prevahu rozhodovania Zadeh, softmax, Euklid nad rozhodnutiami na individuálnych bitoch. Môžme vidieť že CRC7 so Zadehovým rozhodnutím dosahuje podobnú presnosť ako one-hot kódovanie so softmax. V one-hot kódovaní so softmax, kódová štruktúra a rozhodovacie pravidlo sa navzájom podporujú. Vieme že len jeden bit je rovný "1" a výstupné hodnoty sú normované ako pravdepodobnosť.
Rozhodnutie a trénovanie založené na euklidovskej vzdialenosti zlepšuje v priemere výstupové hodnoty a taktiež vidíme priemerné výsledky na Hadamardových kódových slovách, napriek použitiu kódových slov dlhých 15 bitov. Podľa Zadehovej fuzzy logiky rozhodnutie má tendendiu znižovať maximálnu výstupnú chybu danú triednemu kódovému slovu počas trénovania. Počas tohto testu je výstup priradený kódovému slovu z minimom maximálnych bitových chýb. Jendoducho povedané zatial čo one-hot, softmax ťahá bitovú hodnotu k "1" navrhované Zadeh rozhodnutie ťahá najhoršiu chybu na "0". Tabuľka 3 ukazuje na oboch stratégiách podobné hodnoty a môžeme usúdiť, že to čo dáva softmax one-hot enkódovaniu, dáva Zadehova fuzzy logika binárnemu kódovaniu.

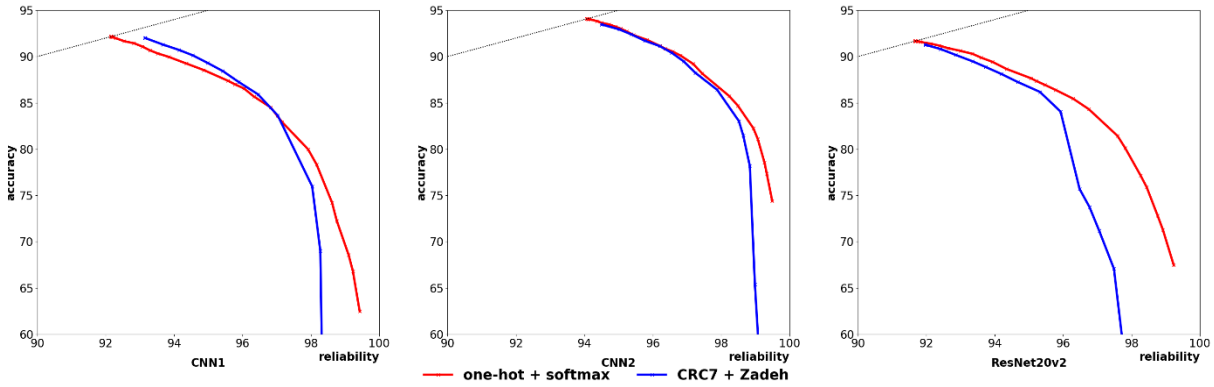
Architektúra	Výstupný Kód	Rozhodnutie	Presnosť	FCA	Nedetekova-teľná Chyba	Spoľahli-vosť
CNN1	CRC7	Zadeh	92.44	92.01	6.77	93.15
CNN1	one-hot	Zadeh	92.01	91.56	7.29	92.63
CNN1	one-hot	softmax	92.15	92.15	7.85	92.15
CNN1	CRC7	softmax	91.85	91.85	8.15	91.85
CNN1	CRC7	bit threshold	89.5609	89.5609	6.87	92.894
CNN1	Hadamard	Euclid	89.92	88.6	8.2	91.53
CNN2	CRC7	Zadeh	93.97	93.47	5.44	94.5
CNN2	one-hot	Zadeh	93.4	92.66	5.76	94.14
CNN2	one-hot	softmax	94.07	94.07	5.93	94.07
CNN2	CRC7	softmax	93.88	93.88	6.12	93.88
ResNet20v2	CRC7	Zadeh	91.56	91.27	7.97	91.97
ResNet20v2	one-hot	Zadeh	84.16	83.28	14.5	85.17
ResNet20v2	one-hot	softmax	91.67	91.67	8.33	91.67
ResNet20v2	CRC7	softmax	91.34	91.34	8.66	91.34

Tabuľka 3 Výsledky rôzneho kódovania a rozhodovacích stratégii

V Prípade Zadeh rozhodnutia môžeme ponúknuť viac kódových slov. To ale znižuje FCA presnosť. Rozdiel medzi presnosťou na 10 bitoch naznačuje, koľko vzoriek bolo priradených ku kódovým slovám mimo tried. Ak sú použité iba kódové slová tried, neexistuje rozdiel

medzi presnosťou a FCA. To je prípad one hot kódovania so softmax kde môže byť použité len kódové slovo triedy. Teda ak klasifikácia ponúkne jedno z kódových slov, musí byť akceptované. Nesprávne klasifikované kódové slovo spôsobuje nedetekovateľnú chybu. Tabuľka 3 naznačuje niektoré benefity chybnej detekcie. Vylúčenie nedôveryhodných vzoriek zvyšuje klasifikačnú spoľahlivosť.

Dobre známy spôsob ako zvýšiť spoľahlivosť je zaviesť úroveň rozhodovania. Všetky roshodovacie pravidlá sú založené na hodnotení kvantitatívnej hodnoty premenej. Obvyklý spôsob na zlepšenie spoľahlivosti je rozdeliť interval premennej prahom Δ , odmietnuť všetky vzorky pod prahom a obísť rozhodovaciu procedúru. Toto pravidlo môže byť tiež aplikované na softmax koncept v ktorom premenná je v rozmedzí $[1n\diagup,1]$ kde n je počet tried. Teda prahové nastavenie pod $1/n$ nemá žiadny význam a nedáva žiadne zlepšenie. V softmax rozhodnutí je vzorka akceptovaná pre klasifikáciu ak hodnota výstupu je vyšia ako prah. Môžeme aplikovať tento istý prístup v Zadehovom rozhodnutí a rozhodnúť ak $max\{S0,\ldots,Sw\}\geq\Delta$. Theorém 1 hovorí že prah $\Delta<0.5$ neprináša žiadne zlepšenie v plnom kódovom rozhodovaní. Na obrázku 6 ukazujeme že graf presnosť-spoľahlivosť s použitím rozličných prahových hodnôt na navrhnuté CRC7 zo Zadehovým rozhodnutím a základným one-hot softmax kódovaním.



Obrázok 6 Charakteristika presnosti a spoľahlivosti podľa one-hot + softmax a CRC7 + Zadeh rozhodovaní pri použití prahovej hodnoty

Obrázok 6 zobrazuje hodnoty z tabuľky 3 ktoré sú na ľavej strane, kde nie je aplikovaný žiadny threshold. Ako threshold narastá body sa posúvajú na pravo. Vlasnosť detekcie chybových kódov posúva nulové hodnoty prahu mimo líniu $y=x$. Môžeme vidieť že obe metódy poskytujú podobné výsledky.

Detekcia chyby mimo distribúcie.

Detekcia chýb je prístup ktorým sa určuje či kódové slovo sa zhoduje s vlasnosťami ktoré definujú konkrétny podpriestor kódového priestoru [29]. Ako trénujeme sieť na trnsformovanie tréningových vzorov do kódových slov detekujúcich chybu predpokladáme že umiestnime tieto vzory do podpriestoru udržujúceho schopnosť chybovej detekcie. Majme hypotézu že ak vzorka je kvalitatívne odlišná od tréningových vzorov, výstupové kódové slovo bude mať iné vlastnosti, napr. bude ležať mimo kódového podriestoru generovaného tréningovou sadou. Hoci hlavný cieľ je ukázať že binárne kódovanie môže byť použité na natrénovanie dosiahnutia vysokej presnosti, detekcia mimo distribúcie "*outlier"* je postranný efekt.

V našom prípade tréningová sada pre "*v rozdelení*" bol CIFAR-10 dataset. Ako "*mimo rozdelenia*" dáta sme si vybrali datasety s rôznymi zložitosťami od najjednoduchších (MNIST, Fashion MNIST), cez podobné (CIFAR-100) až po najzložitejšie (biely šum). Prirodzená (a veľmi nežiadúca) vlasnosť rozpoznávacich systémov ktorá zobrazuje všetky vstupné vzory do jednej z výstupových tried je ich bezbrannosť voči "*outlier"*.

U nás je to prípad softmax rozhodovania. Na druhej strane kódy ktoré detekujú chybu majú implicitnú schopnosť filtrovať kódové slová mimo podpriestoru kódových slov tried. Tento prístup vytvára odmietnuté kódové slová a zlepšuje spoľahlivosť. Tabuľka 4 ukazuje presnosť zachytávať "*outlier"* z rozličných mimodistribučných datasetov. Môžme vidieť že "*outlier"* sú prirodzene detekované sieťami trénovanými priamo z chybovej detekcie, dokonca bez aplikovania prahu.

Architektúra	Výstupný kód	Rozhodnutie	MNIST	Fashion MNIST	CIFAR-100	Šum
CNN1	CRC7	Zadeh	10.29	11.59	11.16	20.6
CNN1	one-hot	softmax	0	0	0	0
CNN2	CRC7	Zadeh	12.37	15.55	11.56	46.05
CNN2	one-hot	softmax	0	0	0	0
ResNet20v2	CRC7	Zadeh	10.77	10.49	5.09	11.61
ResNet20v2	one-hot	softmax	0	0	0	0

Table 4 Zamietnutie na datasete mimo distribúciu bez prahovej hodnoty

Pre vysoké FCA úrovne v rámci distribúcie, presnosť v obch metódach je veľmi podobný. Pre nižsie presnosti v rámci distribúcie, one-hot kódovanie so softmaxom poskytuje lepší výsledok mimo distribúcie. CRC7 so Zadeh rozhodnutím má lepšie detekčné vlastnosti na datasete bieleho šumu hlavne pre ResNet20v2 architektúru. Naproti tomu one-ho kódovanie so softmaxom ukazuje lepší výkon v zvyšku Resnet20v2 experimentoch.

4. Záver

Metódy strojového učenia v posledných rokoch ukázali že sa dokážu vyrovnať s takými úlohami, ktoré sú pre človeka intuitívne a jednoduché (napr. rozpoznávanie tváre) ale sú veľmi ťažko interpretovateľné v algoritmickom jazyku. Rozpoznávanie obrazu je jednou z domén, kde strojové učenie, konkrétne hlboké neurónové siete, dosahuje dobré výsledky, ktoré vedia prekonať už aj človeka. V tejto práci som sa zaoberal rozpoznávaním obrazu na úrovni tvorby trénovacieho dátového súboru tak, aby úsilie človeka bolo pri jeho vytváraní čo najmenšie. Cieľom bolo navrhnúť postupy a metódy na zefektívnenie procesu získavania anotovaných dát pre potreby vývoja metód počítačového videnia založených na hlbokom strojovom učení. Niektoré prístupy sa snažili zredukovať množstvo ľudskej práce na minimum a zjednodušiť samotnú úlohu. Stále však prístup učenia s učiteľom ostáva pri klasifikácii rovnaký a to, že na výstupe sa objaví trieda objektu ktorú určí model. Anotátor potom musí potvrdiť alebo vyvrátiť tento výstup. Neurónová sieť nemá priamu podporu ako povedať že si nie je istá výsledkom, alebo že nevie. Neznámu triedu môžeme určiť len na základe miery ohodnotenia výsledku. Existujú prístupy, v ktorých môžeme vytvoriť novú triedu pre objekty neznámej neznámej triedy, ale na tento prístup potrebujeme dáta ktoré chceme považovať za neznáme, čo zvyšuje nároky na vytváranie dátového súboru. Bolo by však dobré, keby model dokázal priamo určil výstup neurónovej siete o ktorom vieme povedať že vstup patrí do iného rozdelenia resp. triedy. Takouto úvahou sme boli vedení aj v tejto práci. Snažili sme sa pre výstup neurónovej siete neučiť jeden bod v priestore (ako je to pri one-hot kódovaní), ale naučiť model aby na výstupe dával rozdelenie do viacerých bodov ktoré môžeme potom ďalej vyhodnocovať. Za základ sme si zobrali binárne kódovanie pomocou ktorého kódové slová určujú výstupné body priradené jednotlivým triedam. Tento prístup nám ponúka použiť teóriu kódovania na vyhodnocovanie výstupov. Naše zámer bol naučiť neurónovú sieť viacero binárnych kódových slov ktoré reprezentujú vstupy danej

triedy, namiesto toho aby sme učili len jeden výstup ako je to pri one-hot kóde. Takto naučená sieť nám dáva možnosť, že pri neznámom alebo ťažko rozponateľnom vzore dostaneme na výstupe modelu iné kódové slovo ako je jedno hlavné priradené triede, alebo dokonca slovo ktoré sme nepriradili žiadnej triede. V tejto práci sme sa snažili naučiť CNN túto vlastnosť, a potom ju využiť pri anotovaní novej dátovej sady, alebo pri kontrole už existujúcej.
Taktiež sme sa zamerali na samotnú anotáciu a snažili sme sa priniesť pohľad na to, ako je časovo aj výpočtovo náročné robiť anotáciu ktorá je len ponúkaná na kontrolu, alebo ktorú treba namáhavým spôsobom vykonať. Tieto príspevky priamo neponúkaju riešenie nášho cieľa ale ukazujú ako sa dajú niektoré procesy zefektívniť a potvrdzujú že anotácia človekom je drahá úloha. Za hlavný príspevok tejto práce považujeme učenie neurónovej siete na klasifikáciu obrazov pomocou binárnych kódov. Bola to náročná úloha, pretože one-hot kódovanie so softmax rozhodovacou funkciou je dlhoročný spôsob používania hlbokých neurónových sietí na klasifikačné úlohy, s ktorým sú bohaté skúsenosti pri optimalizácii sieťových architektúr aj hyperparametrov, čo umožňuje dobré výsledky ktoré sme ťažko dosahovali.
Na one-hot softmax sa môžeme pozerať ako na špecíalny prípad binárneho kódu ktorý je učený len jedným bitom a jeden výstup je do neho pri učení tlačený. Preto náš ďalší postup bol v hľadaní spôsobu ako trénovať binárny kód pomocou jedného bitu. To nám ponúkla Zadehova fuzzy logika ktorá nám pomáha tlačiť najväčšiu chybu kódového slova smerom k nule. Tento prístup sa osvedčil a v niektorých prípadoch ukázal aj mierne zlepšenie. Jeho hlavným prínosom teda je, že umožnil natrénovať CNN podľa rozdelenia binárneho kódu rovnako dobre softmax u one-hot kódovania. Hoci výsledky dokumentujú len použitie CRC kódov, metodická časť obsahuje aj návrh suboptimálneho opravného kódu ktorý minimalizuje pravdepodobnosť chyby nesprávnej opravy.Naše navrhované riešenie nie je úplne hotové pre priamu implementáciu anotácie, ale ponúka príspevok ako sa inak pozerať na dáta počas anotovania nového datasetu alebo hľadania anomálií v ňom. Ukázali sme, že metóda ktorá dokáže odhaľovať dáta dôležité pre anotáciu má potenciál znížiť množstvo práce, prípadne iniciovať hľadanie nových dát ktoré sa javia z pohľadu binárneho kódu ako odlišné.

5. Literatúra

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database. *IEEE Computer Vision and Pattern Recognition (CVPR), 2009.*

[2] C. Fellbaum.WordNet: An Electronic Lexical Database. BradfordBooks, 1998.

[3] Xiao, Tianjun & Xu, Yichong & Yang, Kuiyuan & Zhang, Jiaxing & Peng, Yuxin & Zhang, Zheng. (2015). *The application of two-level attention models in deep convolutional neural network for fine-grained image classification*. 842-850. 10.1109/CVPR.2015.7298685.

[4] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. PAMI, 30(11):1958–1970, November 2008.

[5] Gershgorn, Dave (10 September 2017). Gershgorn, Dave (10 September 2017). *"The Quartz guide to artificial intelligence: What is it, why is it important, and should we be afraid?"*. Quartz. Retrieved 3 February 2018. Quartz. Retrieved 3 February 2018

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays,Pietro Perona, Deva Ramanan, Piotr Doll´ar, and C LawrenceZitnick. *Microsoft coco: Common objects in context.* InEuropean conference on computer vision, pages 740–755.Springer, 2014

[7] D. P. Papadopoulos, J. R. R. Uijlings, F. Keller and V. Ferrari, *"We Don't Need No Bounding-Boxes: Training Object Class Detectors Using Only Human Verification,"* 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 854-863, doi: 10.1109/CVPR.2016.99.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015.

[9] J. Deng, O. Russakovsky, J. Krause, M. Bernstein, A. Berg, L. Fei-Fei. Scalable multi-label annotation. *ACM conference on human factors in computing (CHI)*, 2014.

[10] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with posterior regularization. In BMVC, 2014.

[11] R. Cinbis, J. Verbeek, and C. Schmid. *Weakly supervised object localization with multi-fold multiple instance learning*. arXiv:1503.00949, 2015.

[12] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In ECCV, 2010.

[13] S. Vijayanarasimhan and K. Grauman. *Large-scale live active learning: Training object detectors with crawled data and crowds*. IJCV, 108(1-2):97–114, 2014.

[14] A. Yao, J. Gall, C. Leistner, and L. Van Gool. *Interactive object detection*. In CVPR, 2012.

[15] C. Wang, W. Ren, J. Zhang, K. Huang, and S. Maybank. *Large-scale weakly supervised object localization via latent category learning.* IEEE Transactions on Image Processing, 24(4):1371–1385, 2015.

[16] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. *Visual recognition with humans in the loop.* In ECCV, 2010.

[17] J. Deng, J. Krause, and L. Fei-Fei. *Fine-grained crowdsourcing for fine-grained recognition*. In CVPR, 2013.

[18] S. Lad and D. Parikh. *Interactively guiding semi-supervised clustering via attribute-based explanations.* In ECCV, 2014.

[19] O. Russakovsky, L.-J. Li, and L. Fei-Fei. *Best of both worlds: human-machine collaboration for object annotation.* In CVPR, 2015.

[20] A. J. Joshi, F. Porikli, and N. Papanikolopoulos. *Multi-classactive learning for image classification*. In CVPR, 2009.

[21] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. *Active learning with gaussian processes for object categorization*. In ICCV, 2007.

[22] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In CVPR, 2010.

[23] S. Vijayanarasimhan and K. Grauman. Multi-level active prediction of useful image annotations for recognition. In NIPS, 2008.

[24] C. Leistner, M. Godec, S. Schulter, A. Saffari, and H. Bischof. Improving classifiers with weakly-related videos. In CVPR, 2011.

[25] M. Guillaumin and V. Ferrari. Large-scale knowledge transfer for object localization in imagenet. In CVPR, 2012.

[26] A. Grigoryan and M. Grigoryan, "Hadamard Transform," in Brief Notes in Advanced DSP, 2009.

[27] S. Yang, P. Luo, C. C. Loy, K. W. Shum, and X. Tang, "Deep representation learning with target coding," in Proceedings of the National Conference on Artificial Intelligence, 2015.

[28] N. Akhtar and A. Mian, "Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey," IEEE Access, vol. 6. Institute of Electrical and Electronics Engineers Inc., pp. 14410–14430, 16-Feb-2018

[29] T. G. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Output Codes," J. Artif. Intell. Res., 1995.

[30] P. Rodríguez, M. A. Bautista, J. Gonzàlez, and S. Escalera, "Beyond one-hot encoding: Lower dimensional target embedding," Image Vis. Comput., 2018.

[31] R. E. Blahut, Algebraic Codes for Data Transmission. 2003.

[32] D. Kim, S. A. Bargal, J. Zhang, and S. Sclaroff, "Multi-way Encoding for Robustness," in Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020, 2020, pp. 1352–1360.

[33] S. Lian, Principles of imprecise-information processing: A new theoretical and technological system. 2016.

[34] S. Yang, P. Luo, C. C. Loy, K. W. Shum, and X. Tang, "Deep representation learning with target coding," in Proceedings of the National Conference on Artificial Intelligence, 2015.

[35] M. Klimo, P. Lukáč, P. Tarábek, Deep neural networks classification via binary error-detecting output codes In: Applied Sciences. - ISSN 2076-3417 . Roč. 11, č. 8 (2021), https://www.mdpi.com/2076-3417/11/8/3563