VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

URČENIE OBSADENOSTI PARKOVISKA Z OBRAZU

OCCUPANCY ESTIMATION OF A PARKING LOT FROM IMAGES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PAVOL DUBOVEC**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, PhD.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Dubovec Pavol**
Program: Informační technologie
Název: **Určení obsazenosti parkoviště z obrazu**
Occupancy Estimation of a Parking Lot from Images
Kategorie: Zpracování obrazu

Zadání:

1. Seznamte se s problematikou počítačového vidění v dopravě. Prostudujte a popište existující přístupy k počítání automobilů v obraze.
2. Vyhledejte a popište dostupné datové sady pro učení a vyhodnocování řešených algoritmů.
3. Pořizujte vlastní data z realistických scén pro vyhodnocování, případně učení algoritmů.
4. Experimentujte s vybranými algoritmy, zhodnoťte jejich praktickou použitelnost v různých scénářích.
5. Vyviňte vlastní řešení zadného problému. Zhodnoťte jeho použitelnost na vhodných datech.
6. Vytvořte demonstrátor svého řešení; diskutujte jeho použitelnost v praxi.
7. Iterativně vylepšujte své řešení směrem "k dokonalosti".
8. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- X. Jiang et al., "Density-Aware Multi-Task Learning for Crowd Counting," in IEEE Transactions on Multimedia, doi: 10.1109/TMM.2020.2980945.
- Dobeš et al., Density-Based Vehicle Counting with UnsupervisedScale Selection, DICTA 2020
- Bharath Ramsundar, Reza Bosagh Zadeh: TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning, O'Reily Media, 2018
- Gary Bradski, Adrian Kaehler: Learning OpenCV; Computer Vision with the OpenCV Library, O'Reilly Media, 2008
- Richard Szeliski: Computer Vision: Algorithms and Applications, Springer, 2011

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2, značné rozpracování bodů 3 až 5.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Herout Adam, prof. Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 30. října 2020

Abstrakt

Pri zisťovaní počtu vozidiel na obrázkoch parkovísk, ktoré nemajú vhodné parametre potrebné na spracovanie, môže byť problém spočítania vozidiel dosť komplexným. Cieľom tejto práce je vytvoriť aplikáciu, ktorá zistí počet vozidiel na zvolenej fotografii, bez ohľadu na to aký pohľad na parkovisko bol zvolený. Takéto zisťovanie bude prebiehať pomocou strojového učenia, na základe modelu, vytvoreného trénovaním na trénovacích dátach, ktoré pozostávajú z fotografii parkovísk z rôznych pohľadov a pozícií. Problém bol riešený nekonvenčným spôsobom, a to tak, že sa obrázky s parkoviskom rozdelia na niekoľko záujmových oblastí (zón) a z týchto oblastí sa vytvoria výrezy, pomocou vytvorenej aplikácie špecializovanej na túto úlohu. Následne prebehne anotácia obrázkov vytvorených týmto spôsobom, pomocou vytvorenej hodnotiacej aplikácie. Obrázky sa následne naformátujú na rovnakú veľkosť. Tieto pripravené výrezy sú následne predané API Keras, pomocou ktorého prebieha trénovanie modelu. Cieľom bolo vytvoriť model, ktorý by bol dostatočne univerzálny natoľko, aby vedel určiť počet vozidiel na fotografii v akomkoľvek prostredí (čas, počasie, poveternostné podmienky) a v čo najkratšom čase. V súčasnosti model dokáže predikovať správny počet vozidiel na výrezu na testovacích dátach s presnosťou 87 % a s pripustením chyby prvého rádu na 95 %. Táto práca sa cielene zameriava na riešenie tohto problému v reálnom čase. Jedná sa klasifikáciu do 7 tried (0-6 vozidiel). Toto riešenie by mohlo byť zaujímavé hlavne pre statické kamery na netypických miestach (napr. bočný pohľad), prípadne je pre ne dôležité snímanie určitých oblastí.

Abstract

When determining the number of vehicles in the pictures of carparks that do not have the appropriate parameters needed for processing, the problem of vehicle counting can be quite complex. The aim of this work is to create an application that detects the number of vehicles in the selected photo, regardless of selected carpark view. This detection will be performed using machine learning, based on a model, created by training on trained data, which consists of photographs of parking lots from different perspectives and positions. The problem was solved in an unconventional way, by splitting the pictures with the parking lot into several areas of interest (zones) and creating anotations from these areas, using the created application specialized for this task. The images are then formatted to the same size. These prepared cutouts are then loaded to the Keras API, which is used to train the model. The aim was to create a model that would be versatile enough to determine the number of vehicles in a photograph in any environment (time, weather, weather conditions) and in the shortest possible time. Currently, the model can predict the correct number of vehicles in the cutout on test data with an accuracy of 87 % and with a first order error of 95 %. This work focuses on solving this problem in real time. It is a classification into 7 classes (0-6 vehicles). This solution could be interesting especially for static cameras in atypical places (eg side view), or it is important for them to capture certain areas.

Kľúčové slová

Počítanie objektov, Počítanie vozidiel, Tvorba aplikácií, Deep Learning.

Keywords

Object counting, Vehicle counting, Application development, Deep Learning.

Citácia

DUBOVEC, Pavol. *Určenie obsadenosti parkoviska z obrazu.* Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing. Adam Herout, PhD.

Určenie obsadenosti parkoviska z obrazu

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána profesora Herouta. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Pavol Dubovec
11. mája 2021

Poďakovanie

Ďakujem vedúcemu bakalárskej práce prof. Ing. Adamovi Heroutovi Ph.D. za pomoc a rady pri písaní tejto práce.

Obsah

Kapitola 1

Úvod

Určovanie počtu vozidiel (prípadne iných objektov) z obrazu môže byť využité na mnohé úlohy, často použiteľné aj na riešenie praktických problémov. Problém počítania vozidiel je možno považovať za podproblém počítania objektov a ten za podproblém detekcie vozidiel. Práca sa zameriava najmä na problém s umiestnením kamier na parkoviskách. V súčasnosti sú mnohé parkoviská vybudované na strechách, či na miestach, kde nie je možné umiestniť kameru, tak že pohľad na jednotlivé vozidlá mali na fotografii rovnakú veľkosť a boli by zobrazené čelnou časťou vozidla, prípade z vtáčieho pohľadu. V prípade fotografií nespĺňajúcich tieto požiadavky, môže byť časovo náročné získať výsledok. Riešenie tejto práce sa preto zameriava na iný prístup k riešeniu tohto problému. Prístup fotografiu delí na určité zóny, v ktorých sa nachádzajú vozidlá. Týmto spôsobom je možné zjednodušiť tento komplexný problém, na problém klasifikácie do siedmych tried (0-6 vozidiel na obrázku), ktorý je rádovo jednoduchší na spracovanie a časovo zvládnuteľný v jednotkách sekúnd.

Riešením problému počtu vozidiel na obrázku je teda klasifikácia výrezov, vytvorených špecializovanou aplikáciou, získaných z obrázkov parkovísk pomocou modelu. Trénovanie modelu prebieha na základe ohodnotených obrázkov, pomocou vytvorenej hodnotiacej aplikácie. Keďže triedna distribúcia ohodnotených výrezov obrázkov nemá rovnomerné rozloženie, bolo nutné využiť class-balanced stratovú funkciu, ktorá tento problém rieši. Hodnotenie funkčnosti je založené na dvoch hlavných parametroch. Prvým z nich je presnosť riešenia, ktorú delíme na dve časti. Prvou je presnosť úplná, ktorá označuje, že počet vozidiel na hodnotenom výreze bol modelom určený správne (Pre obrázok s ohodnotením x, model určí hodnotu x). Druhou presnosťou je presnosť s povolením chyby prvého rádu. Jedná sa o presnosť, pri ktorej pripúšťame chybu modelu, ktorá však nie je väčšia ako jedna. Druhým parametrom je rýchlosť hodnotenia počtu vozidiel na obrázku, na základe modelu. Za uspokojivý výsledok rýchlosti hodnotenia je možné považovať určenie v stotinách sekúnd.

Kapitola 2

Neuronové siete pre spracovanie obrazu

Pre pochopenie princípu spracovania obrazu pomocou neurónových sietí je potrebné priblížiť si niekoľko základných pojmov, ktoré sú nevyhnutné pre pochopenie princípu ich činnosti. Tieto pojmy boli čerpané z kníh [16, 6, 14]. Nejedná sa o plnohodnotné definície týchto pojmov, iba o ich priblíženie, slúžiace pre pochopenie základného princípu tejto práce, bez znalostí problematiky neurónových sietí alebo počítačového videnia.

2.1 Základné teoretické pojmy

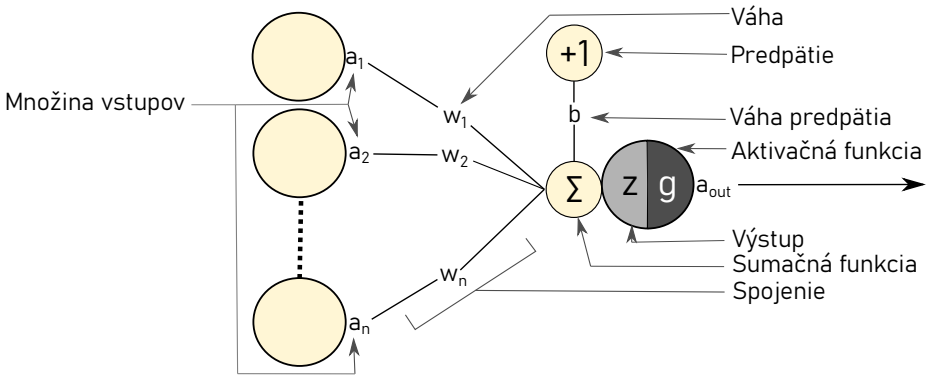
- **Umelé neurónové siete (ANNs)** – sú výpočtové systémy nepriamo inšpirované biologickými neurónovými sieťami, ktoré tvoria živočíšny mozog [9]. Využívajú zložité matematické modely pre spracovanie informácií. Základnými stavebnými jednotkami umelých neurónových sietí, sú umelé neuróny, viď obrázok 2.1. Signály prechádzajú z vstupnej na výstupnú vrstvu, po možnom viacnásobnom prechode vrstvami. Má schopnosť učiť sa a prispôsobovať sa vstupným dátam. Toto učenie prebieha na základe zmeny vnútorných parametrov váh (ω), na základe úspešnosti predikcie.

- **Umelý neurón** – matematická funkcia koncipovaná ako model biologických neurónov. Je základnou jednotkou umelej neurónovej siete. Výstup každého umelého neurónu sa počíta pomocou nejakej nelineárnej funkcie ako súčet jej vstupov. Je možné ich agregovať do vrstiev a priradiť im váhy (viď obrázok 2.1).

- **Spojenie (hrana)** – spája jeden neurón v jednej vrstve s druhým neurónom v inej vrstve alebo tej istej vrstve. Slúži k prenosu signálu (reálneho čísla) do ďalších neurónov. Má vždy priradenú hodnotu váhy. Cieľom trénovania je úprava hodnoty váhy tak, aby sa znížila hodnota straty (chyby).

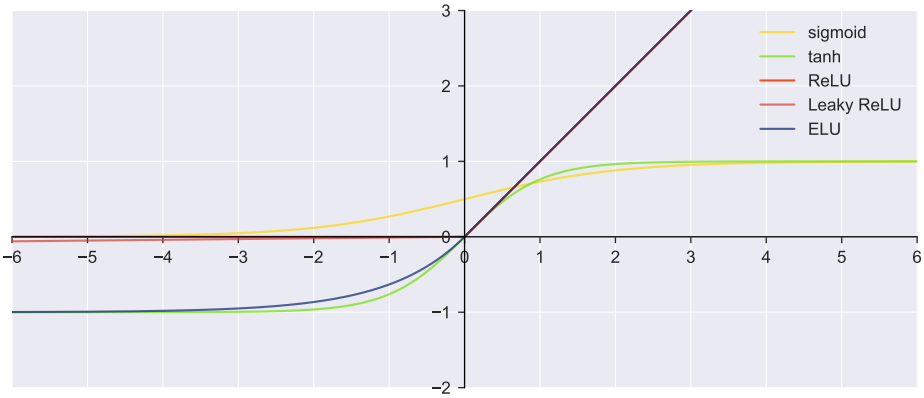
- **Predpätie (Offset)** – Je extra vstup do umelého neurónu a jeho hodnota je vždy 1. Má vlastnú hodnotu váhy spojenia. Zaisťuje, že aj ak všetky vstupy neurónu budú 0, dôjde k aktivácii neurónu.

- **Váha** – Predstavuje silu spojenia. Váha znižuje vplyv vstupnej hodnoty. Rozhoduje o tom, aký veľký vplyv bude mať vstup na výstup.



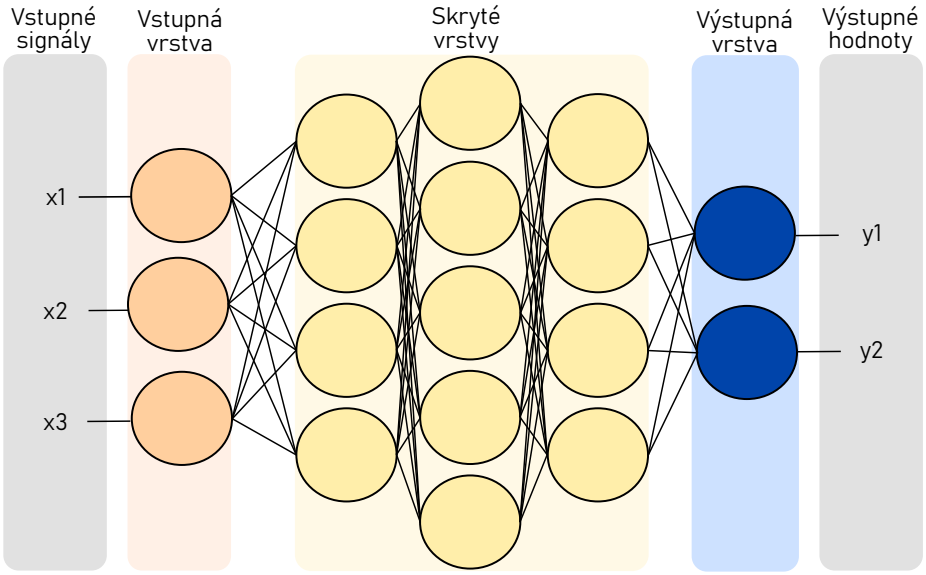
Obr. 2.1: Operácie v jednom neuróne.

- **Aktivačná funkcia** – Používa sa na modelovanie nelinearity do neurónovej siete. Umožňuje neurónu vyhodnocovať aj komplexnejšie problémy. Najčastejšie využívanými aktivačnými funkciami v konvolučných neurónových sieťach sú nasledujúce funkcie (viď obrázok 2.2):
  - Sigmoid – $f(x) = \frac{1}{(1+\exp -x)}$
  - Tanh – $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
  - ReLU(*Rectified linear unit*) – $f(x) = \begin{cases} 0, & \text{ak } x \leq 0 \\ x, & \text{ak } x > 0 \end{cases}$
  - Leaky ReLU – $f(x) = \begin{cases} 0.01x, & \text{ak } x \leq 0 \\ x, & \text{ak } x > 0 \end{cases}$
  - ELU – $f(x) = \begin{cases} \alpha(e^x - 1), & \text{ak } x \leq 0 \\ x, & \text{ak } x > 0 \end{cases}$



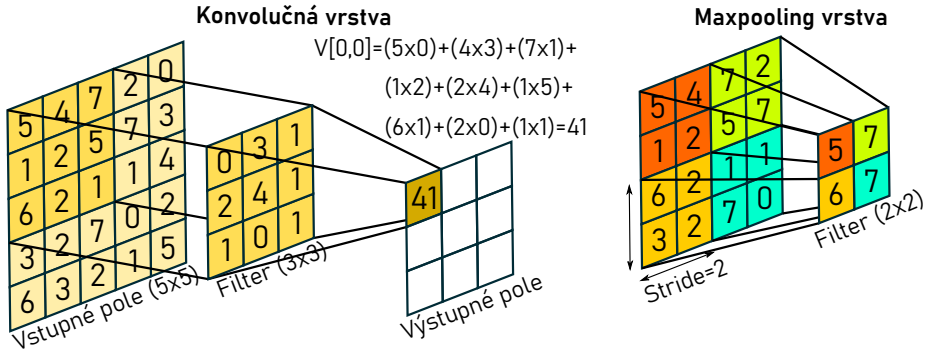
Obr. 2.2: Zobrazenie aktivačných funkcii vyššie na grafe. ReLU, Leaky ReLU a ELU sa po hodnote [0,0] zhodujú.

- **Vstupný tvar** – Je tvar vstupnej matice odovzdávanej do vstupnej vrstvy.



Obr. 2.3: Základné rozloženie umelej neurónovej siete

- **Vstupná vrstva** – Je prvou vrstvou neurónovej siete. Prijíma vstupné signály a prenáša ich do ďalšej vrstvy. Neaplikuje žiadne operácie na vstupné signály a nemá asociované žiadne váhy a predpätia. Niektoré vrstvy sú znázornené na obrázku 2.3.
- **Skrytá vrstva** – Obsahuje neuróny (uzly), ktoré na vstupné údaje aplikujú rôzne transformácie. Jedna skrytá vrstva je súbor neurónov naskladaných vertikálne.
- **Výstupná vrstva** – Je poslednou vrstvou v sieti. Prijíma vstup z poslednej skrytej vrstvy. Pomocou tejto vrstvy je možné získať požadovaný počet hodnôt v požadovanom rozsahu.
- **Úplne spojená (*dense*) vrstva** – Je vrstva vytvárajúca spojenie medzi každým vstupným prvkom a každým výstupným prvkom. Často sa vyskytuje v niekoľkých posledných vrstvách, ktoré kompilujú údaje extrahované predchádzajúcimi vrstvami a vytvárajú konečný výstup. Vrstva očakáva ako vstup vektor, takže je nutné vstupnú maticu vektorizovať.
- **Receptívne pole neurónu (veľkosť filtra)** – Priestorový rozsah prepojenia neurónu na malú oblasť v predošlej vrstve.
- **Konvolúcia** – je operácia medzi dvoma funkciami (f, g) vytvárajúca tretiu funkciu $(f * g)$, ktorá vyjadruje, ako tvar jednej z týchto funkcií ovplyvňuje druhú. Má definované konvolučné jadro (najčastejšie matica s veľkosťou $3x3$ alebo $5x5$ buniek), veľkosť kroku a padding. Operácia je prevádzaná tak, že sa konvolučné jadro postupne prikladá na vstupnú maticu a následne sa vynásobia na sebe ležiace prvky matíc a výsledok sa zapíše do matice výstupnej.
- **Konvolučná vrstva** – Vrstva vykonávajúca operáciu konvolúcie. Hlavnou úlohou konvolučnej vrstvy je detekovať miestne spojenia čŕt z predchádzajúcej vrstvy a mapovať ich vzhľad na mapu čŕt (so zachovaním informácii o ich polohe). Využívajú sa



Obr. 2.4: **Vľavo**: Vizualizácia príkladu kalkulácie v konvolučnej vrstve. **Vpravo**: Príklad výpočtu maxima z regiónu vrstvy maxpoolingu.

taktiež k zvýšeniu počtu kanálov výstupu na lepší prenos informácii. Vyžaduje 4 hyperparametre: počet filtrov, veľkosť receptívneho poľa neurónu, krok, rozsah dopĺňania núl (*padding*). Príklad kalkulácie v konvolučnej vrstve a vo vrstve maxpoolingu je znázornený na obrázku 2.4.

- **Maximálna združovacia (*maxpooling*) vrstva** – Jej úlohou je znižovať priestorový rozmer, tým aj počet parametrov ako prevenciu preučenia. Každá vrstva má definovanú veľkosť jadra, ďalej horizontálny a vertikálny krok, o ktoré sa jadro posúva voči vstupnej matici (najčastejšie veľkosť jadra $2x2$, krok 2). Funguje nezávisle na každom hĺbkovom reze vstupu a zmení jeho veľkosť pomocou operácie MAX.
- **Dopredná propagácia (*forward pass*)** – výpočet a ukladanie medzivýpočtov (vrátane výstupov) pre neurónovú sieť v poradí od vstupnej vrstvy k výstupnej vrstve. Rozmer výstupu je viazaný na funkciu zvolenej siete.
- **Spätná propagácia (*backpropagation*)** – metóda výpočtu gradientu parametrov neurónovej siete. Jedná sa o určovanie podielu jednotlivých váh neurónovej siete na výslednej chybe modelu. Metóda prechádza sieťou v spätnom poradí, od výstupu po vstupnú vrstvu a určuje pre každý parameter určuje parciálnu deriváciu chyby podľa parametru a túto deriváciu vyčísli. Algoritmus ukladá všetky prechodné premenné (parciálne derivácie) potrebné pri výpočte gradientu vzhľadom na určité parametre.
- **Miera učenia** – Určuje, ako rýchlo alebo ako pomaly sa budú aktualizovať hodnoty váhy. Miera učenia by mala byť dostatočne vysoká, aby sa konvergencia netrvala dlhý čas, a mala by byť dostatočne nízka, aby našla miestne minimá.
- **Správnosť** – Vyjadruje blízkosť vypočítanej hodnoty k známej hodnote.
- **Presnosť** – Vyjadruje vzájomnú blízkosť dvoch alebo viacerých meraní.
- **Senzitivita** – Meria podielu pozitív, ktoré sú správne identifikované.
- **Konvergencia** – je približovanie sa k určitej hodnote procesom iterácii.
- **Normalizácia** – je proces zmeny mierky jedného alebo viacerých atribútov v rozmedzí od 0 po 1. Využíva sa v prípade, keď distribúcia údajov nie je známa, prípadne ak táto distribúcia nie je gaussovská.

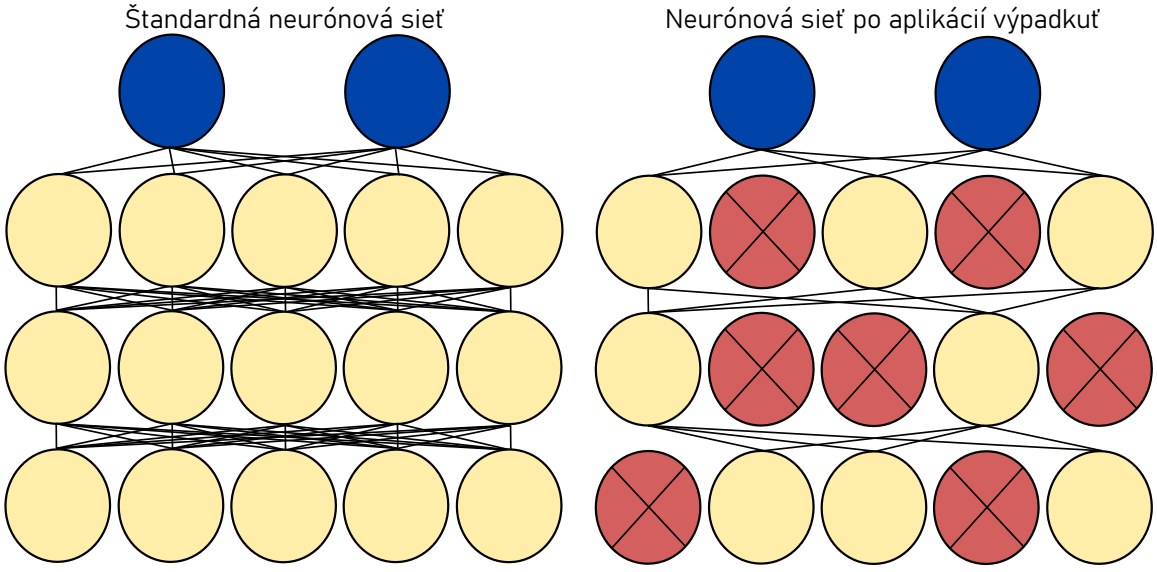
- **Stratová funkcia** – je mierou omylnosti modelu pokiaľ ide o jeho schopnosť odhadnúť vzťah medzi predikciou a referenčnou hodnotou (*ground truth*).
- **Stratová funkcia pre klasifikáciu** – Jedná sa o výpočtovo uskutočniteľné stratové funkcie predstavujúce cenu zaplatenú za nepresnosť predpovedí v klasifikačných problémoch (problémy s identifikáciou, do ktorej kategórie konkrétne pozorovanie patrí) [31]. Stratové funkcie určené ku klasifikácii môžeme rozdeliť do dvoch kategórií a to:
  - Pre klasifikáciu do dvoch tried:
    - Binary Cross-Entropy Loss,
    - Hinge Loss,
    - Squared Hinge Loss.
  - Pre klasifikáciu do viac ako dvoch tried:
    - Multiclass Cross-Entropy Loss,
    - Sparse Multiclass Cross-Entropy Loss,
    - Kullback Leibler Divergence Loss.
- **Trénovacie epochy** – Je číslo vyjadrujúce, koľkokrát je model vystavený množine trénovacích dát.
- **Metrika hodnotenia** – Parameter na základe ktorého budeme hodnotiť neurónovú sieť.
- **Veľkosť mini-balíka (*batch size*)** – Počet trénovacích príkladov v jednom prechode dopredného/spätného šírenia.
- **Optimalizátor modelu** – je vyhľadávacia technika používaná na aktualizáciu váh v modeli. V prostredí keras sú dostupné optimalizátory SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax, Nadam a Ftrl.

Algoritmus	Využiteľnosť	Výhody	Nevýhody
Optimalizácia hrubou silou	→ Ak je veľkosť databázy malá → V prípade, ak je presnosť dôležitejšia, než čas prevedenia	+ Najviac intuitívne riešenie + Ohodnotené sú všetky možnosti	– Čas na výpočet sa drasticky zvyšuje s počtom riešení
Optimalizácia pomocou gradientu	→ Ak je potrebné optimalizovať model rýchlo → Ak nie je možný lineárny výpočet parametrov (Je nutné vyhľadávať)	+ Jednoduché použitie + Stabilita + Výpočtovo efektívne	– Nepoužiteľné, ak existuje viacero lokálnych miním – Pokiaľ je miera učenia priveľká algoritmus môže preskočiť správne riešenie
Optimalizácia genetickým algoritmom	→ Je nevyhnutné vyhnúť sa uviaznutiu v lokálnych minimách	+ Môže nájsť riešenie vo veľmi krátkom čase + Vďaka náhodnosti ponúka veľké množstvo riešení	– Zložité vypracovanie novej heuristiky – Negarantuje optimálnosť riešenia

Tabuľka 2.1: Tabuľka znázorňujúca typy optimalizačných algoritmov a ich vlastnosti.

- **Klasifikácia** – Zaradenie do tried. Na základe úrovne cieľovej premennej a kombinácie vstupov vedúcemu ku konkrétnemu výsledku vedú, je vytvorený model vediaci klasifikovať nové dáta, ktoré mu budú poskytnuté.
- **Regresná analýza** – Je identifikácia vzťahu medzi závislou premennou a jednou alebo viacerými nezávislými premennými.
- **Preučenie (*overfitting*)** – Nadmerná optimalizácia modelu na základe trénovacích dát. Model sa naučí reprezentácie, ktoré sú špecifické len pre trénovací dataset a nezovšeobecňujú sa pre dáta, nepatriace do trénovacieho datasetu. Najdôležitejšími dôvodmi vzniku preučenia sú:
  - Model je príliš zložitý (rozhodovací strom, príliš hlboká sieť, príliš plytká sieť).
  - Veľký počet čŕt bez potrebného množstva trénovacích dát.

  Pre boj s pretrénovaním sa používajú nasledujúce techniky:
  - **Dropout (Riedenie)** – Je jednou z najefektívnejších a najbežnejšie používaných regularizačných techník pre neurónové siete. Výpadok aplikovaný na vrstvu spočíva v náhodnom vylúčení (nastavenie na nulu) niekoľkých jednotiek (skrytých alebo viditeľných) počas výpočtu. Čím vyššie je percento vylúčených jednotiek, tým vyššia je miera regularizácie a tiež čas potrebný pre trénovanie siete. Vizualizácia výpadku je znázornená na obrázku 2.5.
    * **Miera riedenia** – Je zlomok zahodených (vynulovaných) jednotiek. Obvykle sa nastavuje v rozmedzí od 0.2 do 0.5.



Obr. 2.5: Vizualizácia výpadku (*dropout*) na jednoduchej neurónovej sieti. Červené neuróny s krížikom znázorňujú práve vylúčené neuróny.

  - **Predčasné ukončenie** – Je zastavenie trénovania a uloženie parametrov v prípade, že monitorovaná metrika sa prestane zlepšovať. Keďže je možné, že nezlepšovanie metriky môže byť len dočasné, je možné učiť ďalšiu podmienku ukončenia. Príkladmi takýchto podmienok je trpezlivosť (v epochách), prípadne minimálnu zmenu kvality sledovanej metriky.

Obr. 2.6: Formy rozšírenia dát. **Zľava**: Originálny obrázok, geometrická augmentácia (otočenie), fotometrická augmentácia, MixUp.

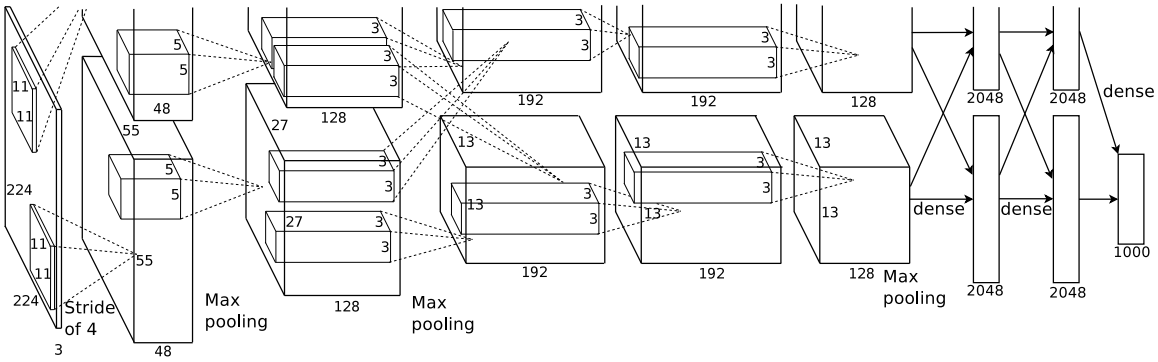
  - **Normalizácia dát** – Za predpokladu, že majú všetky prvky prispievať rovnakou váhou na výsledok predikcie, normalizácia zabezpečuje, že sú rozdiely kompenzované veľkosťou zodpovedajúcich parametrov siete, čo má za následok ovplyvnenie vývoja gradientu a teda aj optimalizáciu parametrov siete.
  - **Rozšírenie dát (data augmentation)** – sú techniky používané na zvýšenie množstva dát pridaním mierne upravených kópií (geometrické – priblíženie, otočenie, fotometrické – úprava jasu, odtieňa, kontrastu) už existujúcich údajov alebo novo vytvorených syntetických údajov z existujúcich údajov (viď obrázok 2.6).
    * **MixUp** – Je typ augmentácie, pri ktorej tvoríme nový obraz prostredníctvom váženej lineárnej interpolácie dvoch existujúcich obrazov.
    * **CutMix** – Výrezky sa vyrežú a prilepia medzi cvičné obrázky, pričom pravdivé štítky sa tiež upravia úmerne voči prilepeným výrezkom. Zvyšuje odolnosť modelu proti poškodeniu vstupu a jeho detekčným výkonom mimo distribúciu.

- **Tenzor** – je pole čísel usporiadané na pravidelnej mriežke s premenlivým počtom osí. Tenzor definujú tri kľúčové atribúty a to:
  - **Počet osí (stupeň)** – 0 ôs - skalár, 1 osa - vektor, 2 osy - matica, 3 osy - kváder atď.
  - **Tvar** – n-tica celých čísel, ktorá popisuje, koľko dimenzií má tenzor pozdĺž každej osi.
  - **Dátový typ** – typ dát obsiahnutých v tenzore.
- **Kód 1 z N (*one-hot encoding*)** – je taký spôsob kódovania, ktorý jednému stavu z N (napr. polohe), zodpovedá aktívny stav jednej stopy bitu. Je vysoko náročný na počet bitov.
- **Matica zámeny (matica chýb)** – Špecifické rozloženie tabuľky, ktoré umožňuje vizualizáciu výkonu algoritmu, zvyčajne sa jedná o učenie s učiteľom (*supervised learning*). Každý riadok matice predstavuje inštancie v predpovedanej triede, zatiaľ čo každý stĺpec predstavuje inštancie v skutočnej triede (alebo naopak) [29].
- **Riedka matica** – matica, ktorá obsahuje veľmi málo nenulových prvkov.

2.2 AlexNet

AlexNet je konvolučná neurónová sieť (CNN), ktorú navrhol Alex Krizhevsky v spolupráci s Iljou Sutskeverom a Geoffreym Hintonom, ktorý bol Krizhevského Ph.D. poradca. Jedná sa o sieť, ktorá mala veľký vplyv na rozvoj oblasti počítačového videnia. Sieť má veľmi podobnú architektúru ako CNN LeNet [23], avšak je hlbšia s väčším počtom filtrov na vrstvu a s naskladanými konvolučnými vrstvami (viď obrázok 2.7). Alexnet taktiež prináša niekoľko kľúčových inovácií architektúry CNN:

- Na doplnenie nelinearity sa využíva aktivačná funkcia ReLu namiesto TanH, čo malo za následok šesťnásobné zvýšenie rýchlosti pri rovnakej presnosti.
- AlexNet umožňuje tréning s dvoma grafickými procesormi, čo umožňuje umiestnenie polovice neurónov modelu na prvej GPU a druhej polovice na druhej GPU, čo umožňuje tréning väčších modelov a tiež skracuje čas potrebný na tréning.
- Prekrývajúce sa združovanie zmenšuje veľkosť siete. Znižuje chybovosť top-1 o $0,4\,\%$ a top-5 o $0,3\,\%$. Taktiež bolo zistené, že u modelov s prekrývajúcim sa združovaním je menej časté pretrénovanie [17].



Obr. 2.7: Architektúra CNN Alexnet.

Alexnet sa skladá z ôsmych vrstiev s váhami, z ktorých je 5 konvolučných (11×11, 5×5 alebo 3×3), priamo nasledované vrstvami max-poolingu a následne 3 vrstvy úplne spojených vrstiev. Využíva nesaturačnú aktivačnú funkciu ReLU, po každej konvolučnej a plne spojenej vrstve, ktorá dosiahla lepšie výsledky ako tanh a sigmoid funkcie [22]. Dropout sa aplikuje po prvej a po druhej plne prepojenej vrstve. Trénovanie siete Alexnet trvá 90 epoch a prebieha na dvoch kanáloch naraz. Využívajú sa pri tom dva grafické procesory Nvidia Geforce GTX 580.

Kedže má architektúra AlexNet celkovo 660-tisíc jednotiek, 61 miliónov parametrov a viac ako 600 miliónov spojení, je nutné riešiť problém s pretrénovaním. Na riešenie tohto problému využíva Alexnet dva prístupy:

- Dropout – Na riešenie problému pretrénovania využíva sieť dropout namiesto regularizácie. Miera dropout je pri architektúre Alexnet 0.5, čo zvyšuje čas potrebný pre konvergenciu modelu o 50%.
- Rozšírenie dát:

– Využitie konverzie a horizontálnych odrazov obrázkov (tento krok spôsobil zväčšenie trénovacej sady 2048 násobne).
– Využitie analýzy hlavných komponentov (PCA) na hodnoty pixelov RGB, čo znížilo chybovosť top-1 o viac ako 1%.

Na trénovanie konvolučnej neurónovej siete Alexnet bol použitý dataset ImageNet. Jedná sa o veľkú databázu fotografii, ktorá je určená pre výskum v oblasti rozpoznávania objektov z obrazu. Skladá sa z 14 197 122 obrázkov a 21 841 tried [21]. Obrázky boli získané z internetu a boli označené ľuďmi pomocou programu Amazon's Mechanical Turk. Keďže dataset ImageNet obsahuje obrázky rôznych rozlíšení, je nutné obrázky prevzorkovať na menšie rozlíšenie, v prípade súťaže ILSVRC to je rozlíšenie 256×256. V prípade, že ide o obdĺžnikový obrázok, zmenší sa jeho rozlíšenie a vyreže sa stredová plocha s veľkosťou 256×256.

V roku 2012 na súťaži ILSVRC-2012 dosiahla sieť ALexnet chybovosti top-5 chybovosti 15.3%, čo je o 11% menej ako druhá najlepšia sieť (model SIFT + FVs od Fei-Fei a kol.)

V súčasnosti je možné znížiť počet parametrov siete Alexnet pomocou skrátenia siete až deväťnásobne bez straty na presnosti [18].

Kapitola 3

Určovanie počtu objektov (vozidiel) z obrazu

3.1 Existujúce riešenia problému obsadenosti parkoviska

Problémom obsadenosti parkoviska sa zaoberá mnoho štúdií. Je takmer nemožné určiť, tak všeobecné riešenie, aby ho bolo možné použiť v akýchkoľvek podmienkach. Každé parkovisko má unikátne parametre, ktorými sa líši od iných parkovísk. Tak ako sa líšia parkoviská, líšia sa aj prístupy viažuce sa k danému typu parkoviska.

Vo väčšine prípadov sú informácie o obsadenosti parkoviska získané pre ich využitie v PGI (Parking guidance and information) systémoch. Jedná sa o systémy poskytujúce vodičom informácie o dostupnosti a polohe dostupných parkovacích miest. Existujúce PGI systémy rozdeľujeme do štyroch kategórií založených na metóde detekcie počtu vozidiel [20]:

1. **počítacie systémy**,
2. **drôtové senzorové systémy**,
3. **systémy založené na bezdrôtových magnetických snímačoch**,
4. **systémy založené na obraze alebo kamere**.

Záujmovou časťou tejto práce je spracovanie vizuálnych dát, je preto nutné zamerať pozornosť na túto oblasť. Systémy založené na obrazových senzoroch môžme rozdeliť podľa objektu záujmu na:

- **Car-driven algoritmy** – objektom záujmu týchto algoritmov je vozidlo. Problémom týchto algoritmov, je fakt, že vozidlá sa na snímkoch z fotoaparátu alebo z kamery nenachádzajú vždy v ideálnych podmienkach. Vozidlá sa môžu nachádzať v rôznych vzdialenostiach od kamery (totožné vozidlo má rôzny počet pixelov) a v rôznych uhloch, čo komplikuje ich detekciu.
- **Space-driven algoritmy** – objektom záujmu týchto algoritmov je parkovacie miesto. Často využívajú metódu odčítania pozadia[1] (background subtraction). Najväčším problémom týchto algoritmov, je ich náchylnosť na zmeny pozadia (počasie, svetlo), ktoré spôsobujú jej zvýšenú chybovosť. Najvhodnejším miestom využitia týchto algoritmov sú parkoviská umiestnené v interiéri.

[1] Metóda odčítania pozadia – je akákoľvek technika, umožňujúca extrahovanie popredia obrázka na ďalšie spracovanie (rozpoznávanie objektov atď.).

Typ senzoru	Výhody	Nevýhody
Počítacie systémy	+ jednoduchá inštalácia systému + relatívne nízka cena	– použitie v komplexných parkoviskách takmer nemožné – nedokážu poskytnúť bližšie informácie o parkovacom mieste
Drôtové senzorové systémy (ultrazvukové, laserové)	+ relatívne vysoká presnosť rozpoznania + nízke náklady na údržbu	– drahá vstupná investícia – riešenie je takmer nepoužiteľné v exteriéri
Systémy založené na bezdrôtových magnetických snímačoch	+ rýchle a relatívne presné určenie obsadenosti + dlhá životnosť	– drahá vstupná investícia
Systémy založené na obraze alebo na kamere	+ nákladovo efektívne + dlhá životnosť + vysoká presnosť	– nízka robustnosť

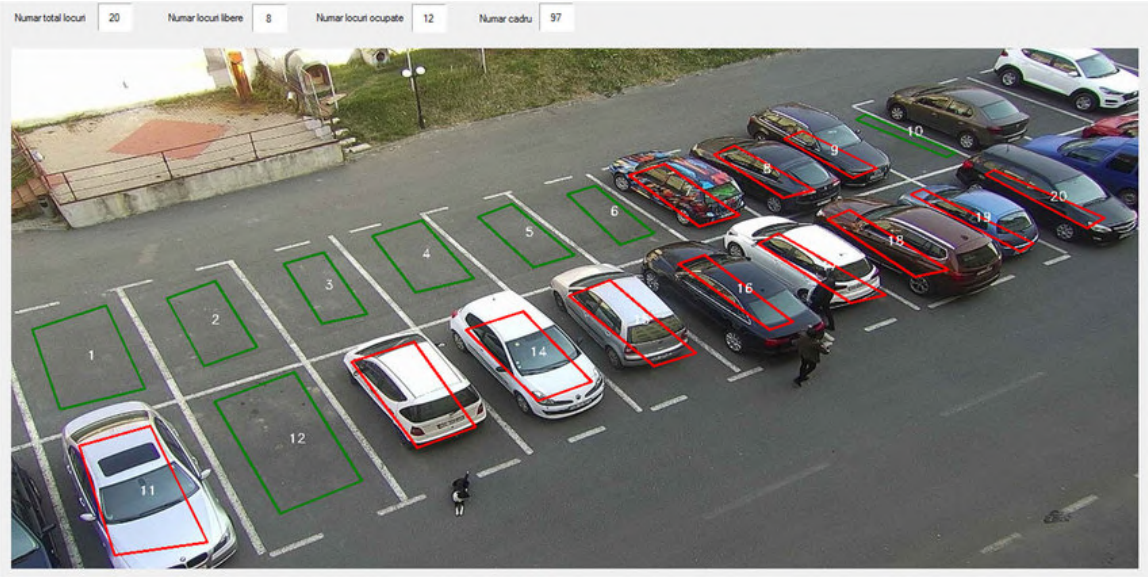
Tabuľka 3.1: Tabuľka znázorňujúca výhody a nevýhody jednotlivých systémov.

V súčasnosti sa však mnohokrát využíva kombinácia oboch týchto prístupov. Najčastejším prístupom hĺbkového učenia pre počítanie vozidiel je regresia jedným pohľadom (one-look), kde model priamo predpovedá počet vozidiel pre vstupný obraz. Vo výsledku sa môže jednať o nadhodnotenie vo forme celého čísla alebo presné reálne číslo. V sekcii 2.2 je priblížená architektúra, ktorá zmenila počítačové videnie – Alexnet. V ďalších sekciách 3.2 a 3.3 sú vysvetlené často využívané prístupy pri riešení obsadenosti parkoviska a v sekcii 3.4 sú ukázané špeciálne riešenia, konkrétne riešenie pomocou termokamery a riešenie pomocou satelitných snímok. Každá z týchto sekcií obsahuje klady a zápory týchto konkrétnych prístupov.

3.2 Riešenia problému obsadenosti parkoviska založené na klasifikácii

Klasifikácia objektov je v súčasnosti veľmi využívaným prístupom k riešeniu mnohých problémov od využitia v automobilovom priemysle až po využitie v lekárstve. Tento prístup je často využívaný aj pri riešení problému obsadenosti parkoviska. Objektom záujmu môže byť vozidlo, jedno parkovacie miesto, prípadne menšia skupina parkovacích miest. Existujú metódy založené na klasifikácii a metódy založené na hlbokom učení.

Klasifikačný prístup vyžaduje pred samotnou klasifikáciu previesť mapovanie parkoviska [33, 1]. Jedná sa o činnosť, pri ktorej algoritmu určíme polohu parkovacích miest alebo bloky obsahujúce parkovacie miesta pomocou súradníc, ktoré musí manuálne užívateľ zvoliť pomocou aplikácie, v prvej časti spracovania dát (počiatočná konfigurácia). Na to, aby bolo možné zistiť dostupný počet miest na parkovisku a ich lokalizáciu, je potrebné získať informácie týkajúce sa štruktúry monitorovaného parkovacieho priestoru. Často sa preto využíva grafické rozhranie so skúmanou plochou, do ktorej používateľ zvolí štyri rohy každého parkovacieho miesta, v prípade, že sa jedná o prístup skúmajúci každé parkovacie miesto individuálne. Útvarom tohto prístupu je štvoruholník, často obdĺžnikového, kosodĺžnikového alebo lichobežníkového tvaru. Alternatívnym implementačným prístupom je zadávanie len dvoch bodov. Užívateľ v tomto prípade zadáva len dva body uhlopriečky. Tento prístup je však možný len v prípade, ak je vstupným útvarom štvorec alebo obdĺžnik. Ak je skúmaná plocha väčšia, ako jedno parkovacie miesto, teda skúmame plochu



Obr. 3.1: Výsledná vzorka z referenčnej hodnoty 1, ktorá je súčasťou práce [33].

parkoviska pokrývajúcu viacero parkovacích miest, bude táto práca hovoriť o tzv. zónovom prístupe. Pri tomto prístupe prevažujú pre parkovacie miest zväčša pravidelné útvary, ako napr. obdĺžnik alebo štvorec (využitie dvojbodového prístupu). Po uložení týchto bodov sa vytvorí mapa parkovacej plochy, obsahujúca obrázok a označené plochy, očíslované zväčša automaticky alebo iným spôsobom. Toto nastavenie sa uloží pre ďalšie použitie.

Po tomto postupe sa prístup spracovania v rôznych prácach variuje. Práca [33] z takto vytvorenej mapy oblastí parkovacích miest, priradzuje stav každému parkovaciemu miestu na základe extrakcie niekoľkých rôznych obrazových čŕt z mapy poskytnutej v počiatočnej konfigurácii (viď obrázok 3.1). Následne prebehne predspracovanie obrázkov parkovacích miest pomocou odstránenia šumu a vyhladenia okrajov pomocou na to určených špeciálnych algoritmov. Toto výrazne zvýši efektívnosť a aj výsledky programu. Nasleduje adaptívne odstránenie pozadia pomocou algoritmu MOG2[2] a úprava perspektívy na základe štyroch bodov parkovacieho miesta. Väčšina metód, ktoré nie sú založené na hlbokom učení využíva (lineárny) klasifikátor SVM (*metóda podporných vektorov*). Jená sa o klasifikátor, ktorého cieľom je nájsť nadrovinu v N-dimenzionálnom priestore (N - počet znakov), ktorý zreteľne klasifikuje body dát.

Na základe takto upravených výsledných snímok parkovacích miest sa používajú rôzne metódy na rozlíšenie prázdnych a plných parkovacích miest. Typickými metódami sú:

- Histogram orientovaných gradientov (HOG) – je deskriptor čŕt používaný v počítačovom videní a spracovaní obrazu na účely detekcie objektov. Spočítava výskyty gradientnej orientácie v lokalizovaných častiach obrázka.

- Transformácia čŕt nemenných mierok (SIFT) – je algoritmus detekcie čŕt v počítačovom videní na detekciu a opis lokálnych čŕt na obrázkoch. Transformuje obrazové dáta na súradnice invariantné s mierkou vzhľadom na miestne prvky. Publikovaný v [25].

[2]MOG2 – je algoritmus počítačového videnia, rozlišujúci objekty popredia od objektov pozadia.

- Využitie prahov na rôznych farebných modeloch – neprekročenie/prekročenie určitého prahu určuje prázdnosť/plnosť parkovacieho miesta.

Silné stránky metód založených na klasifikácii pomocou gradientových metód:

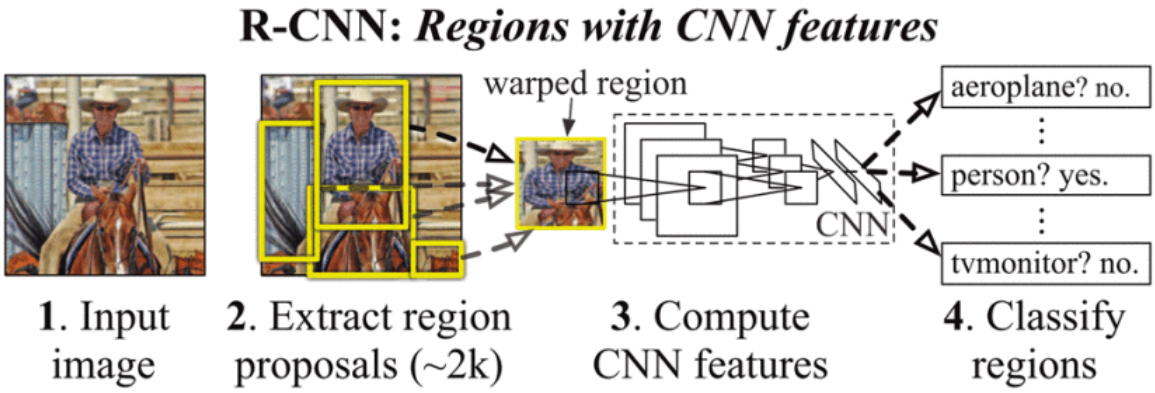
- Veľmi vysoká presnosť hodnotenia.
- V určitej miere umožňuje spracovanie netypických záberov.

Slabé stránky metód založených na klasifikácii pomocou gradientových metód:

- Nie vždy je možné spracovať všetky parkovacie miesta na obrázku (napr. neviditeľné celé miesto).
- Vyžaduje manuálne určenie súradníc parkovacích miest.

Ďalším spôsobom určovania obsadenosti vozidiel na parkovisku je riešenie hlbokým učením. Klasickými príkladmi pre riešenie detekcie objektov sú architektúry konvolučných neurónových sietí rodiny R-CNN[15] (R-CNN, Fast R-CNN, Faster R-CNN a Mask R-CNN).

Pri rodine neurónových sietí R-CNN sa optimalizuje vytváranie ohraničujúcich štvoruholníkov a obrázok sa neprechádza celý. Pre túto rodinu sietí je typický postup znázornený na obrázku 3.2.



Obr. 3.2: Prehľad detekčného systému [15]. 1. Vstupný obrázok, 2. Extrahovanie regionálnych výňatkov (cca. 2000) zdola nahor, 3. vypočítanie čŕt pomocou veľkej konvolučnej neurónovej siete. 4. Klasifikácia regionálnych výňatkov triedne viazanými lineárnymi SVMs.

Alternatívnymi metódami k architektúram z rodiny R-CNN, ktoré buď zvyšujú presnosť alebo znižujú čas výpočtu sú v súčasnosti architektúry YOLOv4, viditeľnej na obrázku 3.3, ďalej CenterNet2 a DetectoRS. Jednou z najpoužívanejších alternatívnych architektúr súčasnosti je architektúra YOLO, konkrétne v YOLOv4 alebo YOLOv5 (zatiaľ bez teoretického základu). Jedná sa o algoritmus založený na regresii – namiesto výberu zaujímavých častí obrázka určí triedy a ohraničujúce polia pre celý obrázok v jednom cykle algoritmu. Dva najznámejšie príklady zo skupiny regresných algoritmov sú algoritmy YOLO, spomínaný vyššie (*You Only Look Once*) a SSD (*Single Shot Multibox Detector*). Často sa používajú na detekciu objektov v reálnom čase, pretože vo všeobecnosti platí sú o niečo nepresnejšie (platí pre YOLOv4 len čiastočne), ale sú oveľa rýchlejšie. Majú všestranné využitie. Využívajú sa v detektoroch objektov vyhodnocujúcich v reálnom čase a existuje veľké množstvo štúdií, ktoré popisujú využitie tohto prístupu.

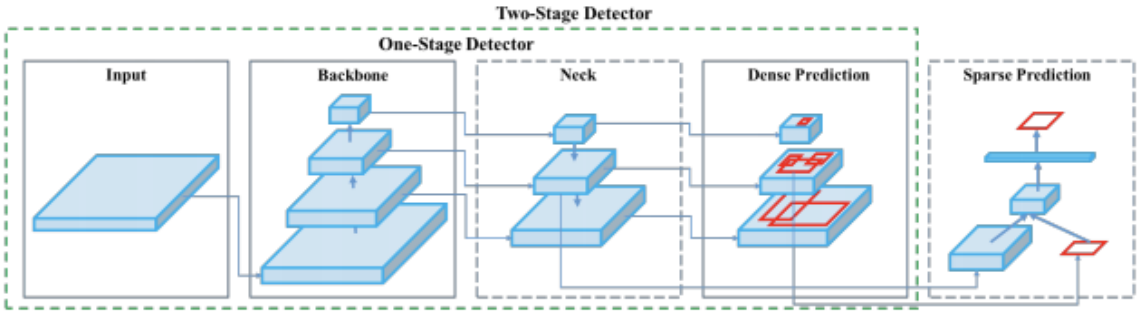
Príkladom je štúdia [8], ktorá využíva existujúce sledovacie kamery a zabudované zariadenia na výrobu efektívneho, vysokorýchlostného a ľahkého detekčného modelu. Pre vyhľadávanie voľných parkovacích miest je navrhovaná jednorázová detekcia objektov a pomocou spracovania obrazu sa identifikuje obsadenie parkovacej mriežky zo sekvenčného obrazu. V noci je systém pre detekciu odkázaný na pouličné osvetlenie. Toto riešenie je navrhované pre existujúce pouličné kamery, schopné využiť technológiu hlbokého učenia a sú použiteľné aj miestnymi počítačmi na výpočet výsledkov obsadenosti parkovacích miest pre celé strany ciest. Ohraničujúce štvoruholníky systému môžu nadobúdať jednú z tried: obsadené, voľné, oklúzia[3], tma (pri nedostatku svetla z pouličných lámp).

Silné stránky metód založených na klasifikácii pomocou hlbokého učenia:

- Vysoká presnosť a veľmi vysoká rýchlosť hodnotenia.
- Automatická detekcia objektu.
- YOLO: Spracováva dáta v reálnom čase.
- Faster/Mask R-CNN: Dobre detekuje aj malé objekty.
- Faster/Mask R-CNN: Skoro dosahujú reálny čas detekcie.

Slabé stránky metód založených na klasifikácii pomocou hlbokého učenia:

- YOLO: Slabá generalizácia v prípade, že objekty v obrázku vykazujú zriedkavé aspekty pomeru.
- Fast R-CNN, R-CNN: Dvojkroková architektúra zamedzuje v detekcii v reálnom čase.
- Slabšie zvládajú detekciu prekrývajúcich sa objektov (je nutné mať triedu oklúzie).



Obr. 3.3: Prehľad architektúry YOLOv4 z práce [8].

3.3 Riešenie problému založené na regresii

Iným spôsobom riešenia problému detekcie a tým aj obsadenosti parkoviska je riešenie pomocou regresnej analýzy. Riešenia založené na nejakej forme regresie sú často využívané v biomedicíne, detekcii a rekonštrukcii objektov.

Najčastejšie riešenia konvolučných neurónových sietí založené na regresii sú riešenia založené na tréningu premeny vstupného obrázku na mapu hustoty. V prístupoch založených

[3]Oklúzia – nastáva, ak objekt ktorý je sledovaný je skryty iným objektom.

na hustote sa najskôr predpovedá pravdepodobnostné rozdelenie hustoty na celom vstupnom obrázku a konečný počet počet objektov sa potom získa integráciou cez predpokladané rozdelenie, v prípade obrázka teda ide o jednoduchú sumu hodnôt na mape hustoty. Tento koncept bol prvýkrát predstavený v práci Lempitskyho a Zissermana [24].

Prvým krokom tohto riešenia je získanie trénovacích vzoriek tak, aby pre každý obrázok existovala mapa hustoty. Pre tento prístup existuje niekoľko rôznych druhov anotácie. K významným patria:

- Anotácia 2D bodom – Najjednoduchší typ anotácie. Okrem polohy neprenáša žiadne ďalšie informácie o anotovanom objekte. predstavuje matematicky deltafunkciu umiestnenú na hodnotenom vozidle.
- Anotácia čmáranicou – Je možné vykonávať dvoma kliknutiami a ťahom myšou (stále relatívne jednoduché), pri správnej anotácii dokáže prenášať ďaleko viac informácií ako anotácia bodkami (veľkosť objektu, natočenie objektu a i.).
- Anotácia kľúčovými bodmi – Viacero kľúčových bodov znázorňuje jeden objekt. Často využívaný spôsob anotácie v prípade osôb alebo tvárí ľudí. V prípade, že neskúmame špeciálne parametre vozidiel ako napríklad typ vozidla (SUV, sedan, atď.), je tento spôsob anotácie zbytočne zdĺhavý.
- Anotácia kvádrom – anotovaný objekt je obsiahnutý v kvádri. V prípade zisťovania obsadenosti parkoviska bez ďalších požiadaviek ide o zbytočne zdĺhavú anotáciu. Anotácia je vhodná pre výskum autonómnych vozidiel a i.

Deltafunkcie nad hodnotenými vozidlami sú následne konvolované gausovskými jadrami, ktoré spôsobia rovnomerné rozmazanie bodky po aj proti osiam x a y voči pôvodnej polohe. Funkcia hustoty pravdepodobnosti s rozmazanými bodkami sa však nezmení (nezmení sa teda ani celkový počet vozidiel po aplikácii rozmazania).

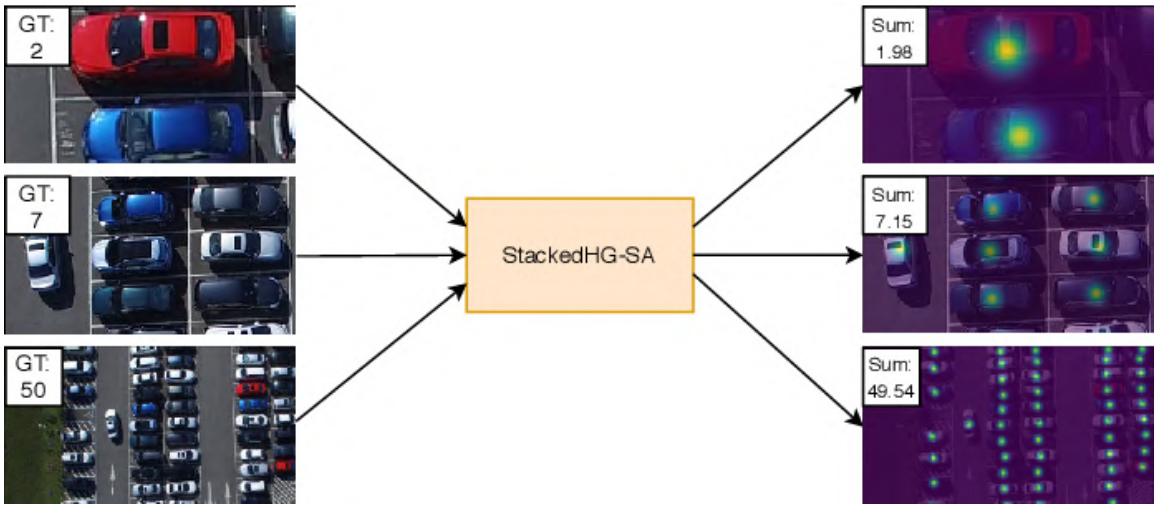
Táto metóda má však aj svoje problémy. Riešenie pomocou nelineárnej regresie bez ďalších obmedzení môže spôsobiť chybu generalizácie. Táto chyba nastáva na miestach, kde sú vozidlá v tieni alebo sú ťažko viditeľné a v tomto prípade sieť môže určiť aj objekty, ktoré nie sú vozidlá v pozadí ako vozidlá. Tento problém mnohé štúdie riešia prostredníctvom regulácie teplotnej mapy[4] (heatmap) [12], [2]. Štúdia [2] ukazuje, že regulácia medzi výstupom siete a mapou hustoty výrazne zlepšuje výsledky siete a rovnaký autori neskôr vylepšujú svoju sieť pridaním operácie globálneho združovania súčtov, ktorá umožňujúca spracovanie obrázkov s ľubovoľným rozlíšením. Autori štúdie [12] navrhujú úpravu Stacked-Hourglass network[5] modelu, tak aby bol model schopný automaticky vyhodnotiť kvalitu vygenerovaných máp hustoty a vyrovnať sa tak s výraznými zmenami veľkostí vstupných objektov. Vizualizáciu je možné vidieť na obrázku 3.4.

Silné stránky regresného prístupu

- Nie je nutná manuálna kalibrácia.
- Riešenie dokáže zvládnuť aj preplnené scény, kde sa objekty čiastočne prekrývajú.
- Dobre zvláda aj zmeny perspektívy a zmeny mierky.

[4] Teplotná mapa – je grafické zobrazenie dát, v ktorom je každá hodnota reprezentovaná farbou určitého spojitého farebného spektra.

[5] Stacked-Hourglass network – Je CNN založená na postupných krokoch združovania a vzorkovania, ktoré sa robia s cieľom dosiahnuť konečný súbor predpovedí.



Obr. 3.4: Model zohľadňujúci mierku, schopný automaticky hodnotiť kvalitu máp hustoty zo štúdie [12]. V ľavej časti obrázku sa nachádza obrázok s jeho hodnotou GT. V pravej časti obrázka sa nachádza vygenerovaná mapa hustoty s predikovanou hodnotou.

Slabé stránky regresného prístupu

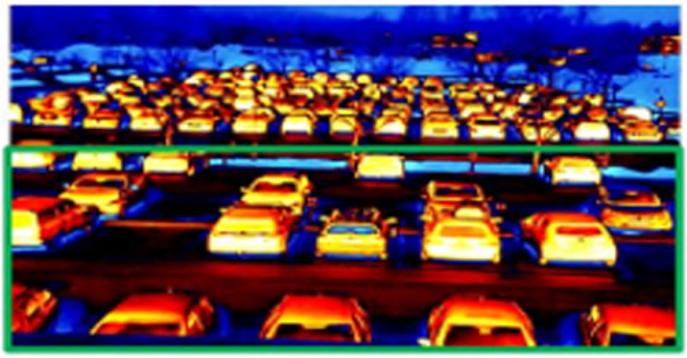
- Problém spracovania pri obrázkoch ktoré nemajú ideálne podmienky (výrazný bočný pohľad, veľmi výrazné prekryvy vozidiel)

Iným spôsobom riešenia regresným prístupom je stratégia rozdelenia vstupného obrázku do menších časti. Takto rozdelené menšie časti následne vstupujú do konvolučnej regresívnej neurónovej siete a následne dochádza k opätovnému spojeniu týchto menších častí do mapy predikcie. Pokiaľ sa tieto menšie časti obrázka prekrývajú je nutné vykonať kombináciu výsledkov týchto častí. Táto myšlienka prvýkrát objavila v práci [7], kde bol použitý regresný model s viacerými výstupmi na súčasnú regresiu regionálnych lokálnych počtov. Neskôr bol tento nápad prevedený do DL v práci [10].

3.4 Špeciálne riešenia problému obsadenosti parkoviska

3.4.1 Riešenie problému založené na termokamere

Ďalším možným prístupom riešenia tohto problému je riešenie prostredníctvom termokamery. Termokamery sa zvyčajne využívajú v údržbe, vo výrobe, v stavebníctve, v lekárstve či u bezpečnostných zložiek. Ich využitie na počítanie objektov nie je veľmi časté z dôvodu, že je možné detekovať len vozidlá v pohybe, či krátko po ňom. Využitie takéhoto zariadenia je preto vhodné najmä v nočných hodinách, prípadne v krajinách z kratším časom slnečného svitu. Použitie klasických vizuálnych snímacích zariadení je v takýchto podmienkach problematické. Väčšina štúdií využívajúcich tepelné kamery na rozpoznávanie alebo klasifikáciu objektov sú závislé na tepelnej stope vozidla. Tento prístup však nie je efektívny, kvôli závislosti na tepelnej stope, preto práca [28] navrhuje riešenie pri ktorom kombinuje využitie termokamery (viď obrázok 3.5) a smartparking systému. Smartparking systém by mal byť schopný neustále detekovať obsadenie parkovacieho miesta bez ohľadu na teplo emitované z vozidla. Jedná sa o riešenie v reálnom čase. Práca navrhuje vlastný algoritmus hlbokého učenia. Autori vykonávali pokusy s architektúrami Yolov2, Yolo-conv, GoogleNet,



Obr. 3.5: Pohľad na parkovaciu plochu termokamerou zo štúdie [28].

ResNet18 a ResNet50, z ktorých bol naúspešnejšia bola architektúra ResNet18.

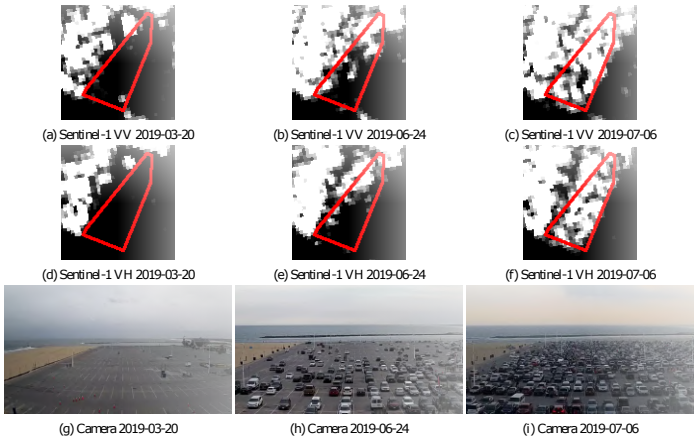
Silné stránky prístupu založeného na termokamere

- Riešenie vytvárané špeciálne pre fungovanie v nočných hodinách.
- Relatívne vysoká presnosť v špeciálnych podmienkach.

Slabé stránky regresného prístupu

- V bežných podmienkach je presnosť nízka.
- Využitie je veľmi limitované na špeciálne podmienky.

3.4.2 Riešenie problému založené na satelitných snímkach



Obr. 3.6: Výpisy obrazových sérií na overovacom mieste, od nízkej obsadenosti po takmer úplnú obsadenosť. Červený polygón znázorňuje masku parkoviska – miesto, v ktorom je vykonávaná regresia na základe pixelov. [13].

Ďalším možným prístupom je riešenie pomocou satelitných snímok SAR. Takémuto riešeniu sa venuje len niekoľko prác. Práca [13] navrhuje algoritmus berúci ako vstup sériu snímok Sentinel-1 spolu s maskou označujúcou, kde je umiestnené parkovisko, a pre každú snímku vracia pomer obsadenosti. Problém je riešený regresiou pixelov, ktorý je založený

na predpoklade lineárneho vzťahu medzi svetlosťou pixelu a medzi pomerom jeho obsadenosti. Autori navrhli prah hodnôt na základe monitorovania parkovísk, kde sú pravdivé pomery obsadenosti známe a následným minimalizovaním rozdielu medzi odhadovanými pomermi obsadenosti a pravdivými pomermi. Keďže zo satelitných snímok nie je možná jednoznačná anotácia, autori využívajú satelitné snímky z oblastí, ktoré sú nasnímané bezpečnostnými kamerami. Takto anotované snímky využívajú za pravdivé (GT) hodnoty. Príklad takejto anotácie na základe prirodzených kamier je znázornený na obrázku 3.6.

Silné stránky prístupu založeného na satelitných snímkach

- Relatívne nízky MAE[6] pri vhodne zvolenom prahu.
- Vytvorený dataset zo snímkami pre rozšírenie práce.
- Využiteľné na veľmi veľkých parkoviskách, ktoré nepokryjú bežné kamery.

Silné stránky prístupu založeného na satelitných snímkach

- Synchronizačná chyba sú približne 2 minúty (snímka zo satelitu voči snímke s GT hodnotou z kamery pri parkovisku).
- Využitie je limitované na satelitné snímky.
- Limitovaný počet dátových bodov.
- Výsledky práce boli validované len jednostranne (možná slabšia generalizácia).

3.5 Existujúce datasety pre riešenie obsadenosti parkoviska

Pre riešenie problému obsadenosti parkoviska je možné použiť niekoľko dostupných datasetov. Datasety môžme rozdeliť podľa senzoru, akou boli fotografie získané na:

- získané leteckým pohľadom,
- získané satelitom,
- získané kamerou umiestnenou na alebo pri parkovisku,
- získané dronom.

Datasety založené na fotografiách získaných leteckým pohľadom
Medzi najznámejšie datasety tejto skupiny patrí dataset **COVC** (*Cars Overhead With Context*), ktorý je podrobne popísaný prácou [26]. Každý pixel obrázka datasetu znázorňuje približne 15 cm na zemi. Fotografie boli získavané v nasledujúcich lokalitách: Toronto (CA), Selwyn (NZ), Postupim (GE), Vaihingen (GE) a Columbus (USA). Dataset obsahuje 32 716 originálnych anotovaných automobilov a 58 247 originálnych prázdnych miest. Anotačným štýlom tohto datasetu je centrálny bod vozidla. Skomprimovaná ukážka je znázornená na obrázku 3.7.

Datasety založené na fotografiách získaných satelitom
Do tejto kategórie datasetov patria najmä OIRDS (*Overhead Imagery Research Data Set*) z práce [32] a VEDAI(*Vehicle Detection in Aerial Imagery*) z práce [30] viď obrázok 3.8. Snímačom týchto datasetov je satelit.

[6] MAE – Mean Average Error

Obr. 3.7: Ukážka skomprimovanej kópie obrázku z datasetu COVC. Jená sa o obrázok z oblasti Toronto. Dataset je detailne popísaný v práci [26].

- **OIRDS** – Dataset vytvorený v roku 2009. Obsahuje približne 900 anotovaných obrázkov s približne 1800 identifikovanými cieľmi. Formátom anotácie je ohraničujúci štvoruholník. Aktuálna sada údajov OIRDS obsahuje iba anotácie vozidiel. Nezahŕňa ďalšie cieľové typy vozidiel. Viac o datasete v práci [32].
- **VEDAI** – Dateset vytvorený v roku 2014. Každý obrázok k dispozícii v niekoľkých farebných spekrtách a rozlíšeniach (512×512, 1024×1024). Obrázky obsahujú rôzne variácie orientácie, zmeny osvetlenia, prípadne tieňovania, zvláštnosti alebo oklúzie. Viac o datasete v práci [30].

Obr. 3.8: Ukážka fotografie z datasetu, **vľavo** – OIRDS, **vpravo** – VEDAI. Datasety sú detailne popísané v prácach [32, 30].

Datasety založené na fotografiách získaných kamerou umiestnenou na miestach na alebo pri parkovisku

Najviac zastúpená kategória datasetov, používaných pri riešení problému obsadenosti parkoviska. Senzorom týchto datasetov je kamera, prípade fotoaparát často umiestnený na vyvýšenom mieste. Medzi najznámejšie z nich patria:

- **CNRPark+EXT** – Dataset obsahujúci 4 323 obrázkov skladajúci sa z dvoch podsúčastí:
  - **CNRPark** – Snímacími zariadeniami tohto datasetu sú 2 kamery, snímajúce parkovisko počas dvoch dní za slnečného počasia v mesiaci júl. Dataset obsahuje 242 obrázkov, z ktorých bolo vytvorených 12 584 výrezov parkovacích miest.
  - **CNR-EXT** – Snímacími zariadeniami datasetu je 9 kamier s rôznymi perspektívami a uhlami pohľadu, snímajúcich parkovisko počas 23 dní v zimných mesiacoch. Snímky sú zachytávané v rôznych svetelných podmienkach a obsahujú vzory čiastočnej oklúzie, spôsobené prekážkami ako sú stromy, pouličné lampy či iné automobily, v prípade ktorých sa môže jednať o čiastočne alebo úplne zatienenie. Dataset obsahuje 4 081 obrázkov parkovísk, z ktorých je vytvorených 144 965 snímok parkovacích miest. Vizualizáciu obrázku z datasetu a niekoľko výrezov parkovacích miest je možné vidieť na obrázku 3.9.

Obr. 3.9: **Vľavo**: Ukážka fotografie z datasetu CNRPark+EXT. **Vpravo**: Ukážky príkladov výrezov. Viac informácií o datasete a o hlbokom učení nad ním vykonávaným je možné získať v prácach [5, 4].

- **PKLot** – Dataset obsahujúci 12417 obrázkov (viď príklad obrázkov 3.10) vozidiel z dvoch rôznych parkovísk, pričom na prvom parkovisku boli snímky zachytávané z dvoch rôznych uhlov. Snímacím zariadením datasetu je lacná full HD kamera Microsoft LifeCam HD-5000. Toto zariadenie je umiestnené na streche budovy, pre minimalizáciu oklúzie. Hlavným cieľom datasetu bolo získanie snímok v rôznom počasí tak, že sa každých 5 minút zaregistruje zmena prostredia. Výsledné farebné obrázky boli uložené JPEG s bezstratovou kompresiou[7] v rozlíšení 1280×720 pixelov. Viac o tomto datasete je možné zistiť v práci [3].

[7] bezstratová kompresia – je trieda algoritmov kompresie dát, ktorá umožňuje dokonalú rekonštrukciu pôvodných dát z komprimovaných dát.

Obr. 3.10: **Vľavo**: Ukážka fotografie z datasetu PKlot. **Vpravo**: Ukážky príkladov výrezov. Obrázky sú z datasetu podrobne popísaného prácou [3].

- **PUCPR+ (*Pontifical Catholic University of Parana+ Dataset*)** – Jedná sa o rozsiahly dataset zameraný na počítanie vozidiel v scénach rôznych parkovísk. Obsahuje informácie o 16 456 automobiloch. Kamera je umiestnená na desiatom poschodí univerzity PUCPR. Formát fotografii je jpg. Anotáciami sú textové súbory (*.txt), s automobilmi označenými po radoch, pričom meno sa zhoduje s názvom obrázku. Dataset je určený len pre výzkumné účely. Viac informácií o datasete v práci [19]. Príklady obrázkov z daného datasetu je možno vidieť na obrázku 3.11.

Obr. 3.11: Ukážka fotografie z datasetu PUCPR+ popísaného v práci [19].

Datasety založené na fotografiách získaných pomocou dronu
Najznámejším datesetom tohto druhu je dataset CARPK (*Car Parking Lot Dataset*). Obsahuje fotografie obsahujúce informácie o 89 777 vozidlách. Fotografie tohto datasetu sú formátu PNG, ktoré sú nasnímané pomocou dronu (*PHANTOM 3 PROFESSIONAL*). Ukážku týchto snímok je možné vidieť na obrázku 3.12. Obrázky pochádzajú z troch rôznych parkovísk. Spôsob anotácie je rovnaký ako pri datasete PUCPR+.

Obr. 3.12: Ukážka fotografie z datasetu CARPK popísaného v práci [19].

Kapitola 4

Návrh riešenia problému obsadenosti parkoviska zónovým prístupom

Návrh sa skladá zo 4 častí:

1. Aplikácia na vytváranie výrezov zón.
2. Aplikácia na anotáciu, normalizáciu a porovnávanie výsledkov.
3. Implementácia class-ballanced stratovej funkcie.
4. Návrh, tréning modelu a vyhodnotenie výsledkov.

4.1 Návrh aplikácie na vytváranie výrezov zón

Prvou úlohou tvorby tejto práce bola tvorba vhodného datasetu z obrázkov parkoviska. Keďže bol zvolený netradičný postup riešenia problému, ktorý zahŕňa deľbu obrázka na menšie časti, nebolo možné použiť žiaden voľne dostupný softvér, ktorý by túto činnosť vykonal dostatočne efektívne. Voľne dostupný softvér často umožňuje vytvorenie jedného výrezku a jeho následné uloženie, čo bolo v prípade tohto riešenia nevhodné.

Z tohto dôvodu bolo nutné v prvej časti tvorby vytvoriť aplikáciu, ktorá by dokázala obrázok vysokého rozlíšenia rozdeliť na niekoľko menších častí, ktoré v tejto práci nazývame zóny. Zóna je výrez z originálneho orezávaného obrázka tvaru štvorca ľubovoľnej zvolenej veľkosti, obsahujúci x počet vozidiel, kde $x \in \{0, 1, 2, 3, 4, 5, 6\}$. Navrhovaná aplikácia sa skladá z troch hlavných častí:

- Plocha vstupu s podkladom originálneho obrázku na voľbu súradníc výrezu.
- Plocha nastavení.
- Lišta s informáciami o zvolených bodoch a názve súboru.

Prvým krokom je zvolenie zložky s obrázkami, z ktorých je potrebné dataset vytvoriť, pomocou tlačidla „Vybrať zložku s obrázkami“. Obrázky by mali byť fotografované z rovnakého miesta a pod rovnakým uhlom pokiaľ chceme využiť funkcionalitu orezania všetkých fotografií. Po zvolení zložky sa načíta prvý obrázok a uloží sa ako podklad vstupnej plochy a je

Obr. 4.1: Ukážka aplikácie na vytváranie výrezov. Červené polopriehľadné štvorce zobrazujú zvolené zóny. Pre vytvorenie novej zóny je potrebné kliknúť dvakrát, ľavým (bod 1) a pravým (bod 2) tlačidlom.

umožnený vstup do plochy. Pokiaľ chce užívateľ vytvoriť výrez, zadá súradnice bodov želaného výrezu do na to určenej plochy pre vstup. Zadávanie prebieha kliknutím tlačidla myši. Súradnice bodu 1 užívateľ zvolí kliknutím ľavého tlačidla myši a súradnice bodu 2 kliknutím pravého tlačidla myši. Súradnice týchto bodov sa následne zobrazia na spodnej lište. V prípade, že je obrázok väčší ako dostupná veľkosť plochy preň určená, objavia sa slidebary, umožňujúce pohyb vo veľkom obrázku. Riešenie pomocou slidebarov bolo uprednostnené pred možnosťou zmeny veľkosti obrázka, z dôvodu možnej straty informácií (presnosti) v prípade príliš veľkých obrázkov. Toto je možné pozorovať na obrázku 4.1.

Po zvolení oboch bodov výrezu môže používateľ aplikácie vytvoriť zónu, kliknutím na tlačidlo „Pridaj štvorec“. Ak používateľ zvolí súradnice bodov, ktoré netvoria štvorec, sú dané súradnice normalizované tak, aby štvorec tvorili a to tak, že súradnice prvého bodu zostávajú v nezmenenom stave a súradnice druhého bodu sa dopočítajú ako $wh_n \doteq (w+h)/2$. Po dokončení tohto úkonu sa na ploche vstupu objaví polopriehľadný štvorec zóny červenej farby.

Užívateľ má možnosť interakcie s už vytvorenými zónami (štvorcami) nasledujúcim spôsobom:

- Dvojnásobným kliknutím ľavého tlačidla myši dôjde k označeniu štvorca v zozname štvorcov a sú sprístupnené jeho úpravy.
- Dvojnásobným kliknutím a potiahnutím štvorca je umožnený pohyb štvorca.
- Zväčšenie a zmenšenie označeného štvorca pomocou tlačidiel „+“ a „−“.
- Odstránenie označeného štvorca pomocou tlačidla „Odstrániť štvorec“.
- Kliknutím pravého tlačidla myši nad štvorcom sa uložia zmeny vykonané nad označeným štvorcom.

Aplikácia umožňuje používateľovi načítať rozloženie zón (štvorcov) z minulej relácie pomocou tlačidla „Načítaj z minulej relácie“. Tlačidlo otvorí dialógové okno, pre možnosť voľby súboru z minulej relácie. Tlačidlo „Výpis údajov do súborov“ vytvorí súbory znázorňujúce aktuálny stav na vstupnej ploche bez nutnosti spustenia orezávania alebo ukončenia aplikácie. Pomocou začiarkavacieho políčka „Zvoliť názov datasetu“, má používateľ možnosť súčasne upraviť názov všetkých výrezov a aj meno zložky do ktorej budú tieto výrezy uložené (predvolený názov je „result“). Používateľ má taktiež možnosť zvoliť prvé číslo číslovania datasetu. Táto funkcionalita je vhodná, pokiaľ nejaká zóna nebola zvolená správne (chybne zvolená poloha) a je nutné vytvoriť nové výrezy prislúchajúce k tejto zóne.

Poslednými časťami toho rozhrania sú prvky pre vytváranie výrezov. Po stlačení tlačidla „Orezať“, vytvorí aplikácia zložku s názvom datasetu a vytvorí výrezy ktoré si používateľ zvolil prostredníctvom zón. Aplikácia taktiež vypočíta spoločnú plochu zón, vzhľadom na každú zónu individuálne (z1/z2 môže byť iná ako z2/z1), ak sa dané zóny prekrývajú. V prípade, že je taktiež začiarknuté políčko „Orezať všetky fotografie“, vykoná sa daná operácia nad všetkými fotografiami v zvolenej zložke. Výrezy sú potom číslované v závislosti na zóne, nie v závislosti na obrázku (obrázky z rovnakej zóny sú umiestnené za sebou). Táto funkcionalita vo výraznej miere urýchľuje anotáciu. Následne je vytvorený súbor s rodičovskými súbormi a výrezmi, ktoré z daného súboru vznikli.

Výstupom aplikácie sú:

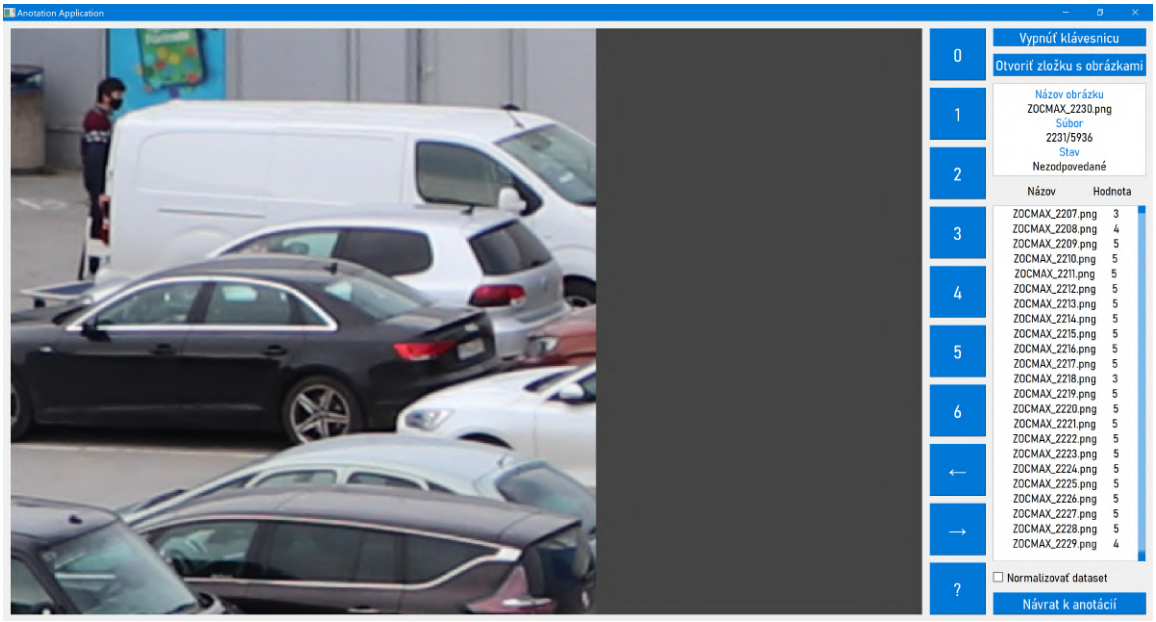
- Vytvorené výrezy zón s názvom zvoleného datasetu.
- Súbory s informáciami o zónach a výrezoch formátu JSON prislúchajúce k datasetu zvoleného názvu:
  - Súbor s koordinátmi zvolených zón pre možné opätovné načítanie.
  - Súbor s vypočítanou spoločnou plochou prekrývajúcich sa zón.
  - Súbor s názvami rodičovských súborov a‘výrezkov výtvorených z nich.

4.2 Návrh hodnotiacej aplikácie

Po prieskume možného hodnotiaceho softvéru, ktorý by bol vhodný pre túto úlohu, som sa rozhodol navrhnúť špecializovanú aplikáciu, ktorá by bola určená presne na tento účel. Existuje niekoľko vhodných riešení, ktoré bolo možné v tomto prípade použiť, príkladmi takýchto aplikácií sú napríklad aplikácie LabelImg, CVAT, labelme, Visual Object Tagging Tool a iné. Pri výbere aplikácie bolo nutné splniť nasledujúce základné podmienky:

1. Každé samostatné hodnotenie je výberom jednej zo siedmych tried.
2. Anotované plochy už nie je nutné vyberať, pretože sa už jedná o výrezy zón vytvorené aplikáciou vytvorenou na tento účel, ktorá je dôkladne popísaná v sekcii 4.1.
3. Je nutné mať možnosť upraviť hodnotenie predchádzajúceho snímku (Snímky z rovnakej zóny nasledujú sekvenčne za sebou.).
4. Aplikácia musí byť efektívna, aby nebolo nutné pri tejto činnosti tráviť dlhý čas.

Aplikácie, ktoré sú voľne dostupné, sú buď priveľmi zamerané na hodnotenie do slovných tried, čo výrazne spomaľovalo hodnotenie alebo nespĺňali jednu zo základných požiadaviek, vymenovaných v zozname vyššie. Čas potrebný na anotáciu takýmto spôsobom bol



Obr. 4.2: Ukážka hodnotiacej aplikácie v režime hodnotenia obrázkov. Aplikácia umožňuje úplne ovládanie myšou, prípadne prostredníctvom klávesnice.

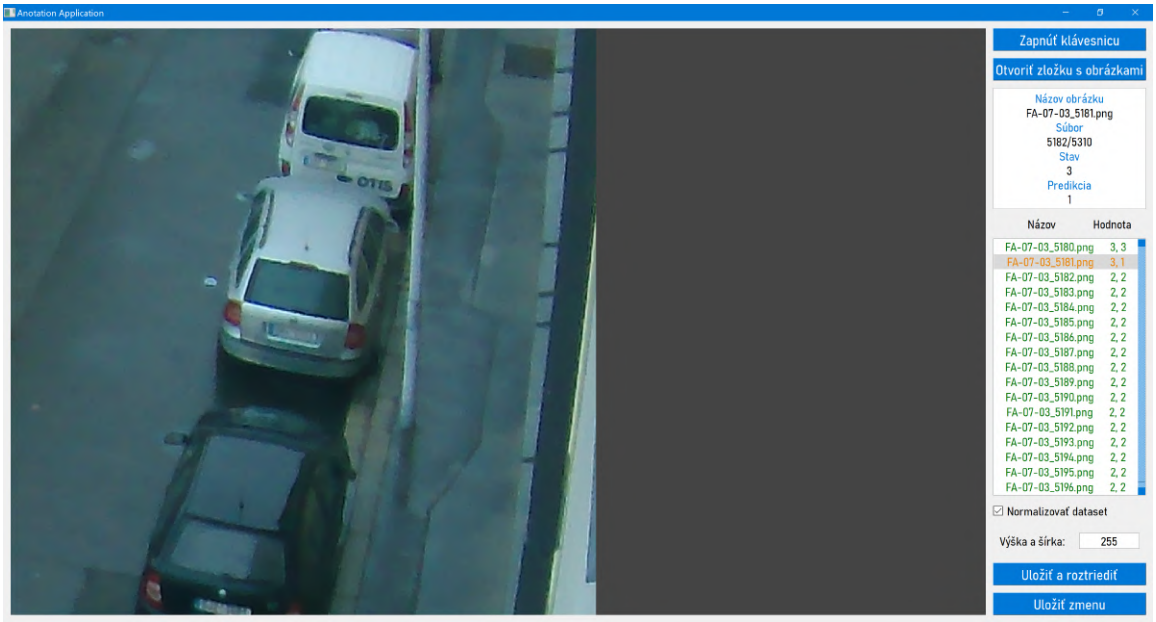
pri takomto jednoduchom číselnom hodnotení neprimerane veľký, vzhľadom na jednoduchosť tejto úlohy. Ďalšie z riešení neumožňovali návrat k už hodnotenému obrázku, čo výrazne predlžovalo úlohu v prípade, že užívateľ urobil chybu.

Aplikácia na hodnotenie výrezov sa skladá z nasledujúcich častí:

- Plocha zobrazujúca snímok, ktorý má užívateľ ohodnotiť.
- Plocha informácii (informácie o súčasnom snímku alebo o už ohodnotenom snímku a zoznam s už ohodnotenými snímkami).
- Voliteľná plocha zadávania pomocou myši.

Užívateľ si vyberie zložku datasetu s výrezmi, ktoré chce aplikáciou hodnotiť pomocou tlačidla „Otvoriť zložku s obrázkami“. Aplikácia môže následne prejsť do jedného z štyroch režimov: Režim hodnotenia, režim úprav, režim úprav s už ohodnoteným datasetom, režim úprav s už ohodnoteným datasetom a s výsledkami z modelu (viď režim hodnotenia obrázkov 4.2).

Do režimu hodnotenia vstupuje aplikácia v prípade, že sa v datasete ešte nachádzajú obrázky, ktoré neboli ohodnotené. Aplikácia najskôr skontroluje, či už existuje k vybranému datasetu výstupný súbor, čo znamená, že už nad týmto datasetom prebiehala anotácia. V prípade, že áno súbor sa načíta a pokračuje sa v anotácii od prvého nehodnoteného obrázku. Ak nie, vytvorí aplikácia nový výstupný súbor formátu JSON a následne sa načíta prvý obrázok zo zložky s datasetom. Hodnotenie prebieha, tak že používateľ zvolí hodnotu počtu vozidiel na obrazovke prostredníctvom klávesnice fyzickej, prípadne klávesnice virtuálnej dostupnej v aplikácií. Hodnotou môžu byť len čísla $\in \{1, 2, 3, 4, 5, 6\}$, prípadne špeciálny operátor (?), vyjadrujúci neurčiteľný počet vozidiel na obrázku, prípadne obrázok, ktorý nie je v súlade s ostatnými obrázkami z datasetu. Po hodnotení dôjde k automatickému posunu na ďalší obrázok datasetu. V ploche s informáciami sú zobrazené informácie



Obr. 4.3: Ukážka hodnotiacej aplikácie v režime porovnávania s výstupom modelu nad testovacími dátami. Čím červenšia je farba textu zoznamu s výrezmi, tým väčšej chyby sa model dopustil.

o obrázku ako je názov, poradie v rámci datasetu a stav, v prípade hodnotiaceho režimu je stav vždy „Nehodnotené“. Pod týmito informáciami sa nachádza zoznam s už ohodnotenými obrázkami a hodnotou, ktorú mu priradil užívateľ. Dvojitým kliknutím na názov už ohodnoteného obrázku sa užívateľ prepne do režimu úprav, prípadne vykonaním akcie posunu späť (tlačidlom „page down“ alebo šípkou späť na voliteľnej klávesnici).

V režime úprav môže používateľ upravovať hodnotenie obrázkov, ktoré už hodnotenie majú. Úprava sa prevádza tak, že užívateľ sa nastaví na súbor, ktorý chce upraviť a následne dvojklikom na obrázok potvrdí, že sa jedná o úpravu a stlačí tlačidlo jeho nového hodnotenia. V režime úprav sa snímok automaticky nepresúva na ďalší obrázok. Úprava sa neukladá okamžite, ale až po opustení režimu úprav pomocou tlačidla „Návrat k anotácii“ a tiež dôjde taktiež k úprave výstupného JSON súboru. V tomto režime môže užívateľ používať funkcie dopredu (→) a dozadu (←) pre pohyb v už ohodnotených obrázkoch. V ploche s informáciami je v časti stav aktuálne hodnotenie obrázku. Po zmene hodnotenia sa stav zmení a objaví sa hlásenie „Zmena vykonaná“ červenej farby.

Režim úprav s už ohodnoteným datasetom sa spúšťa v prípade, že dataset, ktorý užívateľ zvolil je už kompletne ohodnotený. V tomto móde už nie je možný návrat k anotácii, iba uloženie zmien v hodnotení do JSON súboru, kde sa zmena prejaví.

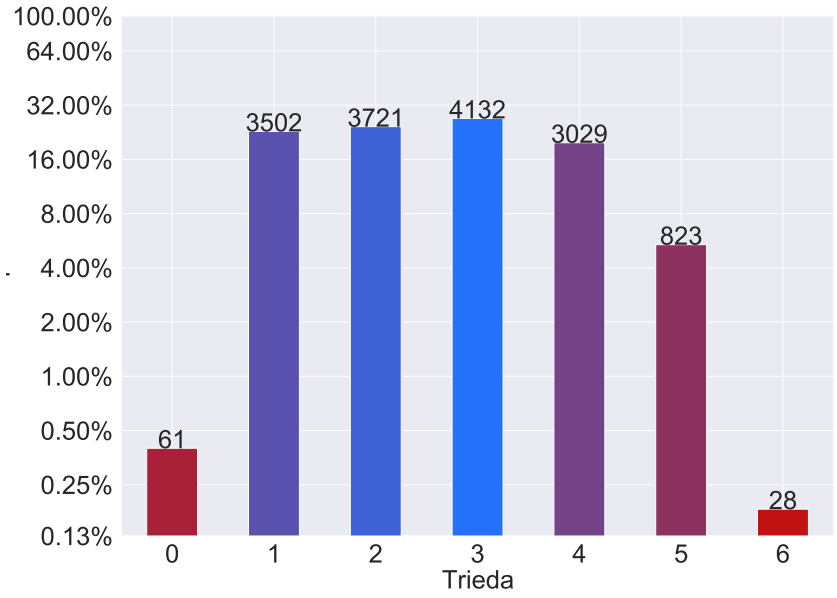
Režim úprav s už ohodnoteným datasetom a s výsledkami z modelu sa spúšťa, ak je dataset kompletne ohodnotený a taktiež obsahuje výstupný súbor z modelu formátu JSON, obsahujúci predikované hodnoty pre jednotlivé výrezy. V tomto režime sú v zozname hodnotených obrázkov v ploche s informáciami okrem hodnotení, ktoré boli zadané používateľom aj predikcie netrénovaného modelu nad rovnakými výrezmi. Text je farebne odlíšený podľa toho, ako veľmi sa reálna hodnota a predikcia líšia (viď obrázok 4.3).

Výstupom aplikácie je súbor formátu JSON, s názvom mena datasetu. Súbor je formátovaný tak, že obsahuje mená výrezov a hodnoty, ktoré im boli priradené hodnotením.

4.3 Návrh klasifikátora pomocou class-balanced stratovej funkcie

Riešenie opisované v tejto práci pracuje s parkoviskom ako so sériou zón. Tieto zóny obsahujú minimálne 0 a maximálne 6 vozidiel. Zóny sa môžu prekrývať, tento prekryv je neskôr zohľadnený pri samotnom výpočte vozidiel na obrázku. Vstupom do neurónovej siete je obrázok zóny normalizovaný na jednotnú veľkosť (predvolená veľkosť je 256×256). Dataset s takýmito obrázkami je vstupom do neurónovej siete. Ide o klasifikáciu výrezov zón do 7 tried podľa počtu vozidiel.

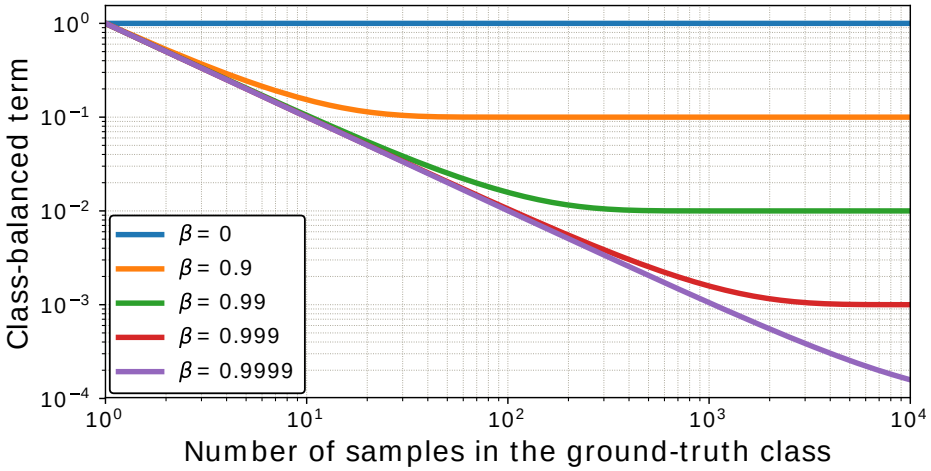
V súčasnom stave dokáže sieť určiť triedu pre 87 % vozidiel vo veľmi krátkom čase (v jednotkách sekúnd). Ak pripustíme chybu 1. rádu, tak ide o presnosť 95,1 % Takéto riešenie by bolo možné použiť na miestach, kde je využitie tradičných metód určovania počtu objektov na obrázku, neprodukovalo dostatočné výsledky, prípadne by ich použitie v danej situácii nebolo možné (bočný pohľad, vysoká miera prekrývajúcich sa vozidiel).



Obr. 4.4: Počet výrezov v jednotlivých triedach vytvoreného datasetu znázornený na logaritmickom meradle.

4.4 Stratová funkcia Class-Balanced Softmax Cross-Entropy

Na parkoviskách zachytených na využitých datasetoch nenastáva často situácia, kedy by v zóne nenachádzalo žiadne auto (zóna s kapacitou 6 je prázdna), prípadne, že by zóna bola plná (hraničná hodnotu veľkosti zóny). Výsledný počet výrezov s hodnotením 0 a 6 je teda v datasete zastúpený v disproporčne menšom počte ako v prípade ostatných hodnotení, znázornené na obrázku 4.4. V dôsledku tohto problému bola implementovaná triedne vyvážená funkcia straty [11] (class-balanced loss function) softmax cross-entropy.



Obr. 4.5: Vizualizácia vzorca $(1-\beta)/(1-\beta^{n_y})$, kde n_y je počet vzoriek v základnej pravdivej triede. Obe osi sú v logaritmickej mierke. Obrázok bol prevzatý z práce [11].

Funkcia Softmax je typom aktivačnej funkcie často využívanej v neurónových sieťach. Využíva sa pri počítaní pravdepodobností rozdelenia z vektoru reálnych čísel. Výstupom funkcie Softmax je reálne číslo v intervale (0, 1), pričom suma pravdepodobností je rovná 1. Funkcia softmax je definovaná vzťahom[27]

$$f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{4.1}$$

Funkcia Softmax sa používa vo viactriednych modeloch, ktoré vracajú pravdepodobnosti každej triedy s cieľovou triedou najvyššej pravdepodobnosti. Objavuje sa v takmer všetkých výstupných vrstvách architektúr strojového učenia. Softmax je často používaná na viactriedne úlohy klasifikácie.

Triedovo vyvážená verzia stratovej funkcie softmax cross-entropy je určená na riešenie problému trénovania z nevyvážených dát zavedením váhového faktoru, ktorý je nepriamo úmerný efektívnemu počtu vzoriek. Nech n_i je počet vzoriek pre triedu i. Je nutné vypočítať efektívny počet vzoriek pre triedu i, ktorý je vyjadrený rovnicou

$$E_{n_i} = \frac{1-\beta_i^{n_i}}{1-\beta_i}, \quad \text{kde} \tag{4.2}$$

$$\beta_i = \frac{N_i - 1}{N_i}. \tag{4.3}$$

Bez bližších špecifikácií dát je veľmi ťažké empiricky zvoliť množinu hyperparametrov N_i, pre všetky triedy. Preto v praxi predpokladáme, že N_i je viazané datasetom a množina $N_i = N$, $\beta_i = \beta = (N-1)/N$. Takže triedne vyrovnanú stratovú funkciu môžme zapísať ako

$$CB(p, y) = \frac{1-\beta}{1-\beta^{n_i}} \mathcal{L}(p, y), \tag{4.4}$$

kde n_y je počet prvkov v ground-truth triede y a p sú odhadované pravdepodobnosti tried modelu a $\mathcal{L}(p, y)$ je stratová funkcia modelu.

Predpokladajme predpovedané výstupy z modelu pre všetky triedy $z = [z_1, z_2, ..., z_C]^\top$, kde C je celkový počet tried. Funkcia softmax vyžaduje vzájomnú výlučnosť tried. Pravde-

podobnostnou funkciou vypočítanou cez všetky triedy vypočítame ako

$$p_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)}. \tag{4.5}$$

Predpokladajme vzorku s triednym štítkom y, tak je strata softmax cross-entropy (CE) pre túto vzorku zapísaná ako

$$CB_{softmax}(z, y) = -\log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right). \tag{4.6}$$

Takže pre triedne vyváženú softmax cross-entropy stratovú funkciu platí vzťah

$$CB_{softmax}(z, y) = -\frac{1-\beta}{1-\beta^{n_i}} \log\left(\frac{(\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right). \tag{4.7}$$

V ďalšom kroku sa výstup stratovej funkcie potom upraví. Úprava sa vykonáva z dôvodu, aby bolo možné urýchliť učenie siete. V tomto prípade ide o cieľ, kedy je potrebné, aby sa model nemýlil o niekoľko tried, teda chyba modelu nebola väčšia ako 1. Táto úprava je vykonaná pomocou multiplikátora, ktorým sa vynásobí strata vypočítaná pomocou triedne vyváženej softmax cross-entropy stratovej funkcie. Multiplikátor je vypočítaný ako

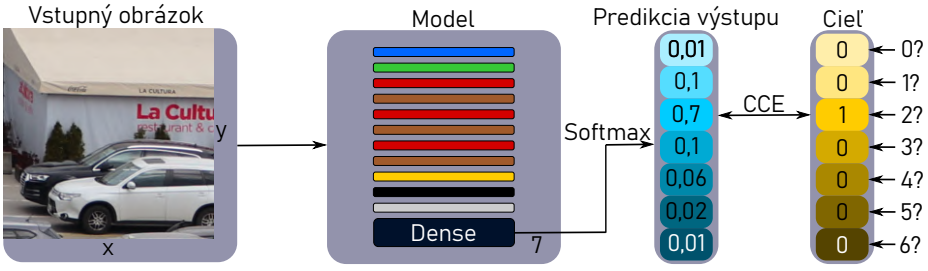
$$m = e^{(|p-t|)}, \tag{4.8}$$

kde m je multiplikátor stratovej funkcie, e je Eulerovo číslo, p je predpokladaná hodnota, t je pravdivá hodnota. Výsledná strata je vypočítaná ako

$$CB_{\mathrm{WS}}(z, y) = \left[-\frac{1-\beta}{1-\beta^{n_i}} \log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right)\right] \times m. \tag{4.9}$$

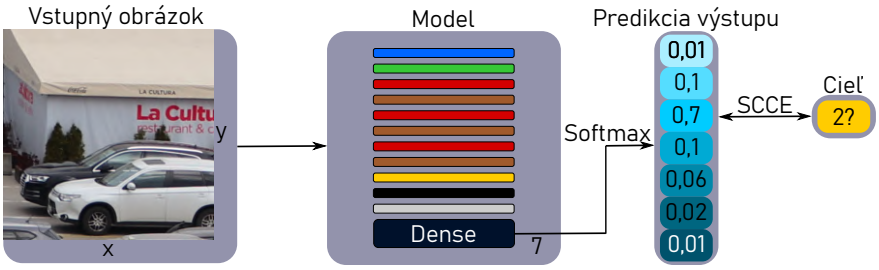
Výsledky pokusov s jednotlivými návrhmi je možné nájsť v sekcii 7.

4.5 Stratová funkcia (Sparse) Categorical Cross-entropy



Obr. 4.6: Znázornenie spôsobu vyčíslenia stratovej funkcie kategorickej krížovej entropie.

V prípade multitriednych klasifikačných problémoch sa veľmi často využíva stratová kategorická funkcia krížovej entropie (CCE). Stratová funkcia kategorickej krížovej entropie (CCE viď 4.6), kódovaná v kódovaní 1 z N (*one-hot encoding*) a stratová funkcia riedkej kategorickej krížovej entropie (SCCE viď 4.7), používajú rovnaký vzťah pre výpočet, teda



Obr. 4.7: Znázornenie spôsobu vyčíslenia stratovej funkcie riedkej kategorickej krížovej entropie.

ich výsledky majú rovnaký výstup a aj presnosť je rovnaká. Vzorec pre výpočet oboch týchto stratových funkcií je

$$CE = -\frac{1}{N}\sum_{\mathrm{i}=1}^{N}\sum_{\mathrm{c}=1}^{C}\mathbf{1}_{\mathrm{y_i}\in\mathrm{C_c}}\log p_{model}(y_\mathrm{i}\in C_\mathrm{c}), \tag{4.10}$$

kde i je pozorovanie, N je počet pozorovaní, c je trieda, ktorých počet je C, $\mathbf{1}_{y_\mathrm{i}\in C_\mathrm{c}}$ je indikátorová funkcia i-tého pozorovania patriaca do triedy c. p_{model} je pravdepodobnosť predpovedaná modelom pre i-té pozorovanie, ktoré patrí do triedy c. Ak existuje viac ako 2 triedy, je vytvorený vektor pravdepodobností C, z ktorých každá udáva pravdepodobnosť, že by vstup siete mal byť klasifikovaný ako patriaci do príslušnej kategórie. V prípade, že sa triedy vzájomne vylučujú (každá vzorka patrí presne do jednej triedy) je vhodné použiť stratovú funkciu SCCE. V prípade, že jedna vzorka môže mať viacero tried alebo štítky (*labels*) majú „mäkkú“ pravdepodobnosť (napr. $[0.1, 0.2, 0.6, 0.1]$) je nutné použiť stratovú funkciu CCE.

Celočíselný štítok	Štítok typu 1 ku N						
	0	1	2	3	4	5	6
2	0	0	1	0	0	0	0
5	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0
6	0	0	0	0	0	0	1
3	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0

Tabuľka 4.1: Spôsob prevodu celočíselných štítkov na štítky typu 1 z N (*one-hot*) zobrazený pomocou tabuľky.

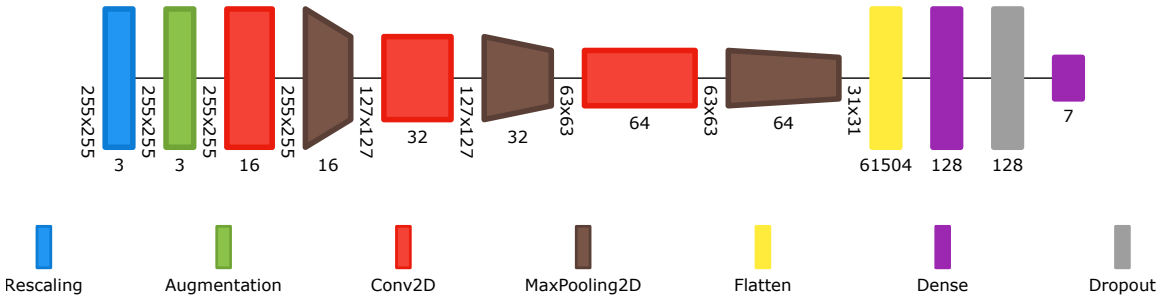
V prípade, že sa triedy vzájomne vylučujú nie je nutný vnútorný súčet triedy, pretože každá trieda v tomto prípade obsahuje len jeden nenulový element, ktorý nemá nenulovú hodnotu. Za predpokladu, že každá y_i patrí presne do jednej triedy $C_{\mathrm{y_i}}$, je možné napísať:

$$CE = -\frac{1}{N}\sum_{\mathrm{i}=1}^{N}\log p_{\mathrm{model}}(y_i\in C_{\mathrm{y_i}}). \tag{4.11}$$

V tomto prípade sa jedná o stratovú funkciu SCCE. Táto úprava umožňuje šetriť čas aj pamäť. Pri normálnom výpočte krížovej entropie sa na pravdepodobnosti y_{pred} použije veľa

operácií log, skalárneho súčinu či súm, ktorých výsledok budú nakoniec nuly. Jedná sa teda o zbytočné operácie. Namiesto toho, aby sme robili logaritmické operácie na všetkých y_{pred} a potom skalárný súčin s y_{true}, môžeme priamo zvoliť pravdepodobnosť z y_{pred} v indexe poskytnutom y_{true}. Následne môže prebehnúť logaritmovanie. Žiadne operácie skalárneho súčinu ani súm teda nie sú nutné. V prípade, že ide o klasifikáciu s veľkým počtom tried môže byť úspora dosť značná (logaritmus jedného čísla namiesto sumy cez C tried).

4.6 Návrh modelu klasifikátora



Obr. 4.8: Ukážka modelu.

Keďže sa jedná o klasifikáciu do malého počtu tried, bolo cieľom vyvinúť, čo najjednoduchší model, v čoho dôsledku by bolo možné trénovať sieť v čo najkratšom čase a s čo najväčšou presnosťou. Trénovanie modelu prebiehalo v knižniciach Tensorflow a Keras, ktorých funkcionalita je dôkladne popísaná v sekciách 6.3 a 6.4. Pre trénovanie je použitý dataset výrezkov zón, ktorý je tvorený z vlastných obrázkov parkovísk a taktiež obrázkov, ktoré mi boli poskytnuté vedúcim práce. Detailný popis datasetu je možné nájsť v sekcii 5.3. Pred trénovaním je dataset načítaný zo zložky funkciou z prostredia Keras `keras.preprocessing.image_dataset_from_directory()`. Dataset sa rozdelí v pomere $1:3$ na dve časti: trénovcí a validačný dataset.

Taktiež vzniká dataset testovacích fotografií, z ktorých chceme zistiť hodnotu obsadenosti parkoviska. Tieto dáta sú predané modelu, ktorý ich ohodnotí a následne nastáva reevaluácia pôvodnej fotografie, z ktorej boli dané výrezy vytvorené. Základná teória neurónových sietí je popísaná v sekcii 2.1.

Architektúra neurónovej siete sa skladá z troch konvolučných vrstiev s veľkosťou jadra 3×3 a s aktivačnou funkciou ReLU (viď obrázok 2.2). Každá konvolučná vrstva je nasledovaná maximálnou zdržovacou vrstvou (*maxpooling*) s veľkosťou združenia 3×3 a s veľkosťou pohybu okna združovania pri každom kroku 2×2. Po poslednom páre konvolučnej vrstvy a maxpooling vrstvy nasleduje splošťovacia (*flatten*) vrstva. Táto vrstva je nasledovaná plne prepojenou vrstvou so 128 jednotkami. Výpadok v sieti je nastavený na 0.5, čo vo výraznej miere zabraňuje pretrénovaniu siete. Poslednou vrstvou architektúry je plne prepojená vrstva so siedmimi jednotkami. Pred konvolučnými vrstvami sa nachádzajú vrstvy pre zmenu mierky (*rescaling*) na interval [0,1] a vrstva pre rozšírenie dát, zabraňujúca pretrénovaniu. Vizualizáciu tejto architektúry je možné vidieť na obrázku 4.8.

Kapitola 5

Dataset

Po dôkladom preskúmaní existujúcich datasetov bolo vhodné vytvoriť vlastný dataset fotografií, pre vytvorenie výrezov, ktoré by boli zamerané na zložitejšie prekryvy, či iné anomálie, ktoré by mohli dokázať možnú praktickosť vytvorenej metódy riešenia problému obsadenosti parkoviska. Dataset by mal byť univerzálny natoľko, aby bolo pre sieť jednoduché naučiť sa rozoznávať počet vozidiel v rôznych poveternostných aj svetelných podmienkach. V ideálnom prípade by mal obsahovať fotografie, ktoré nie sú viazané na určitý čas (teda obsahuje nočné snímky a západ, prípadne východ slnka), majú rôzne uhly snímania (pohyb kamery počas snímania), vozidlá sú v rôznych vzdialenostiach od kamery (vozidlá nie sú jednotnej veľkosti) a na viacerých miestach dochádza k okulzii.

5.1 Poskytnuté obrázky parkovísk

Vedúcim práce mi bol poskytnutý dataset fotografií parkovísk obsahujúci 4744 snímok, jednotného rozlíšenia 4608×3456 pixelov. Poskytnutý dataset spĺňa takmer všetky predpoklady pre ideálny dataset. Dataset obsahuje fotografie snímané celodenne, takže na snímkach sú zachytené aj špeciálne svetelné podmienky noci, ale aj západu a východu slnka. Kamera sa v datasete pohybuje len minimálne, ale predsa len k istému pohybu dochádza. Vozidlá na fotografiách sú v rôznych vzdialenostiach od kamery. Jediným nedostatkom je nižšia miera oklúzie (nižšia forma generalizácie prekryvov), tento problém však je možné riešiť správnym výberom zón. Príklad snímok z tohto datasetu je možné vidieť na obrázku 5.1.

Obr. 5.1: Ukážka obrázkov z poskytnutého datasetu. Obrázky sa líšia uhlom snímania kamery aj svetelnými podmienkami.

5.2 Vlastné obrázky parkovísk

Obrázky pre tvorbu vlastného datasetu boli snímané pomocou fotoaparátu (Canon EOS 77D) na niekoľkých pozíciach, ktoré by mohli mať ideálne parametre. Prvým takýmto pokusom výberu miesta snímania bolo parkovisko automobilky. Jedná sa o veľké parkovisko vozidiel určených na export. Tento nápad však nepriniesol požadované výsledky, keďže autá na fotografii aj s veľkou mierou zväčšenia boli primalé. Na obrázku 5.2, je možné vidieť niekoľko pokusných fotografií z tohto miesta.

Obr. 5.2: Príklady snímok z kopca Stránik pri Žiline, snímajúce parkovisko automobilky KIA. Snímky sa ukázali ako nevhodné z dôvodu nízkeho počtu pixelov zachytávajúcich jedno vozidlo.

Ďalším pokusom o vytvorenie datasetu bolo parkovisko obchodného centra MAX. Jená sa o strešné parkovisko, ktoré je možné snímať z‘neďalekého vyvýšeného miesta. Toto parkovisko má väčšinu vhodných parametrov – je možné meniť uhol pri fotografovaní, miesto na fotografovanie je umiestnené na vyvýšenom mieste, avšak nejedná sa o vtáči pohľad ani pohľad zvýšky, čiže obsahuje istú mieru oklúzie a autá sú prijateľnej veľkosti. Nedostatkom je nemožnosť získania fotografií vozidiel v noci, keďže sa vozidla na parkovisku v noci nenachádzajú. Príklady obrázkov z toho miesta je možné vidieť na obrázku 5.3.

Obr. 5.3: Snímky z fotografovania pri obchodnom centre MAX. Snímky je možné získať v rôznych uhloch.

5.3 Vytvorenie datasetu

Z fotografií spomínaných v sekciách 5.2, 5.1, bolo prostredníctvom implementovanej orezávacej aplikácie, podrobne popísanej v sekcii 4.1, vytvorených niekoľko datasetov. Anotácia týchto datasetov prebiehala pomocou aplikácie podrobne popísanej v sekcii 4.2. Obrázky z ďalej popísaných datasetov je možné vidieť na obrázku 5.4. Súčasťou každého vytvoreného datasetu sú

- kategorizované výrezy obrázkov v triednych zložkách do ktorých prislúchajú,
- súbor s anotáciami,
- súbor so súradnicami dvoch krajných bodov každej zóny, pre možné opätovné načítanie,
- súbor s rodičovskými súbormi, výrezov zón,
- súbor susedných zón každej zóny.

Medzi vytvorené datasety patria:

- Dataset FA – Dataset plne vytvorený z fotografií poskytnutých vedúcim práce. Obsahuje 9 podkategórií, rozdelených podľa dátumu nasnímania obrázku, z ktorého daný výrez vychádza. Obsahuje spolu 52336 výrezkov parkovísk, pričom sú všetky tieto výrezy majú prislúchajúce anotácie uložené prostredníctvom súboru JSON. Jedná sa o veľmi variabilný dataset, čo sa týka svetelných podmienok, čo je vhodné pre lepšiu generalizáciu pri učení neurónovej siete.
- Dataset MAX – Menší dataset vytvorený v prvotnom štádiu tvorby aplikácií.
- Dataset MAXCarpark – Dataset vytvorený z obrázkov, získaných pri ZOC MAX. Obsahuje spolu 16044 výrezov zón. Dataset je však obmedzený krátkym časovým obdobím snímania a tiež nižšou variabilitou svetelných podmienok.
- Dataset FA+MAXCarpark – Jedná sa o dataset vzniknutý kombináciu datasetov FA a MAX obsahujúci 68380 výrezov zón.

Obr. 5.4: Ukážky vytvorených výrezov zón z poskytnutých a získaných obrázkov.

Kapitola 6

Implementácia

K implementácii bolo použitých niekoľko známych aj menej známych nástrojov. V nasledujúcich sekciách sú zhrnuté najdôležitejšie z nich.

6.1 Jupiter Notebook

Jedná sa o open-source webovú aplikáciu umožňujúcu vytvárať a zdieľať dokumenty obsahujúce spustiteľný kód, rovnice, vizualizácie či popisný text. Využíva sa hlavne pre numerické simulácie, strojové učenie, vizualizáciu dát, čistenie a transformáciu dát a štatistické modelovanie. Tento softvér bol zvolený z dôvodu blokovej štruktúry a možnosti priamej interaktívnej vizualizácie výstupu. Trénovanie modelu, jeho tvorba a všetky vizualizácie sú vytvorené v tomto prostredí.

6.2 PyQT

Pre tvorbu aplikácii pre spracovanie fotografií (orezávanie a hodnotenie) bolo možné použiť niekoľko knižníc. Príkladmi knižníc spôsobilých takejto úlohy sú knižnice ako PyQT, Kivy, wxPython, TKinter. Z nasledujúceho výberu bola zvolená knižnica PyQT, z dôvodov popísaných nižšie. Nevýhody tejto knižnice neboli v tomto prípade zásadnou prekážkou.

Silné stránky knižnice PyQT

- Možnosť pridávať komplexné widgety.
- Multiplatformová knižnica.
- Rozsiahla dokumentácia.

Slabé stránky knižnice PyQT

- Licenčná limitácia v prípade komerčných aplikácií.
- Je nutná základná orientácia v jazyku C++.

PyQt je väzba Pythonu na Qt, čo je sada knižníc napísaných v jazyku C++ a vývojových nástrojov zahŕňajúce abstrakcie nezávislé na platforme pre GUI, prácu v sieti, prácu s vláknami, s regulárnymi výrazmi, databázy SQL, SVG, OpenGL, XML a mnoho ďalších funkcií.

Pre návrh GUI aplikácií spomínaných v sekciách 4.1 a 4.2 bola použitá knižnica PyQT5 vo funkcionálnej časti aplikácií a taktiež bol použitý vývojový nástroj balíka PyQT, Qt

Creator, konkrétne jeho časť **Qt designer**. Qt Designer je nástroj Qt na navrhovanie a vytváranie grafických užívateľských rozhraní (GUI) pomocou widgetov Qt. Využíva WYSIWYG[1] princíp. Nástroj tiež umožňuje pridanie signálov a slotov, čo umožňuje pridanie funkcie prvkom rozhrania.

Integrácia vytvorenej kostry aplikácie s prostredím python prebieha importom súboru typu ui vytvoreného nástrojom Qt Designer. Tento súbor bol importovaný do súboru s funkcionálnou časťou implementácia a pomocou funkcie `getchild('názov_widgetu')`, ktorej je predané meno widgetu odpovedajúceho želaného prvku z nástroja Qt Designer.

6.3 Tensorflow

Tensorflow je bezplatná, open-source multiplatformná softvérová knižnica pre strojové učenie vyvinutá spoločnosťou Google. Jadro tejto knižnice je vyvinuté v jazyku C++. Je všestranne využiteľná, ale zameriava sa primárne na predspracovanie údajov, vytváranie, tréning hlbokých neurónových sietí a ich následné hodnotenie. Knižnica obsahuje veľké množstvo nástrojov, knižníc a návodov. Umožňuje jednoduchú tvorbu aplikácií využívajúcich strojové učenie. Táto knižnica bola zvolená z dôvodu možnosti jednoduchej tvorby modelov, možnosti výberu z viacerých API (najmä Keras) a možnosti vizualizácie dát o trénovaní modelu vizualizačným nástrojom Tensorboard, ktorý je popísaný v sekcii 6.5.

6.4 Keras

Je softvérová open-source knižnica poskytujúca vysokoúrovňové rozhranie pre prácu s neurónovými sieťami. Umožňuje prácu s vstavanými API ako sú modely, vrstvy a spätné dopytovania (*callbacks*). Ďalej obsahuje moduly umožňujúce prácu s optimalizátormi, metrikami a stratovými funkciami. Knižnica tiež obsahuje nástroje pre predspracovanie dát. Jediným základom knižnice je po verzii 2.4 Tensorflow. Táto knižnica je v práci použitá, takmer na všetkých miestach, na ktorých sa pracuje s neurónovými sieťami. Miesta, kde nie je použitá sú charakteristické nutnosťou manuálneho prístupu k trénovaniu, či tvorba vlastnej loss funkcie.

6.5 Tensorboard

Je vizualizačný nástroj knižnice Tensorflow pre vytváranie meraní a vizualizácií potrebných počas pracovného toku strojového učenia. Umožňuje sledovanie metrík experimentu, ako sú strata, presnosť, vizualizácia grafu modelu, prípadne vizualizáciu histogramov. Nástroj je používaný pre sledovanie a ukladanie SVG súborov o presnosti, stratách a zmeny miery učenia.

6.6 Implementácia pomocných súčastí modelu

Pri rôznych fázach práce s modelom bolo nutné implementovať súčasti, ktoré sú pre jeho správnu funkciu nevyhnutné. Jedná sa primárne o také súčasti, ktoré nie sú obsiahnuté vo funkciách poskytnutých prostredím Keras, prípadne je nutné vykonať nejakú aktivitu

[1] WYSIWYG - je princíp verného prenosu informácie modelovanej na počítači do reality tak, že zodpovedá presne modelovanému obrazu, s čo najmenším, resp. nebadateľným skreslením.

pre to, aby bolo možné, želanú funkcionalitu dosiahnuť. Medzi takéto súčasti patrí implementácia spätného dopytovania (*callback*), úpravy stratových funkcií z implementačného hľadiska či vytvorenie vlastného trénovacieho cyklu.

Implementácia spätného dopytovania

Pretrénovanie siete je možné riešiť niekoľkými spôsobmi, pričom niektoré z nich sú popísané v sekcii 2.1. Jedným ešte nespomenutým spôsobom boja s pretrénovaním je skoré zastavenie tréningu, ktorý je možné implementovať pomocou spätného dopytovania (*callback*). Príklad spätného dopytovania, používaného pred implementáciou manuálneho trénovacieho cyklu je možno vidieť v nasledujúcom príkazovom bloku.

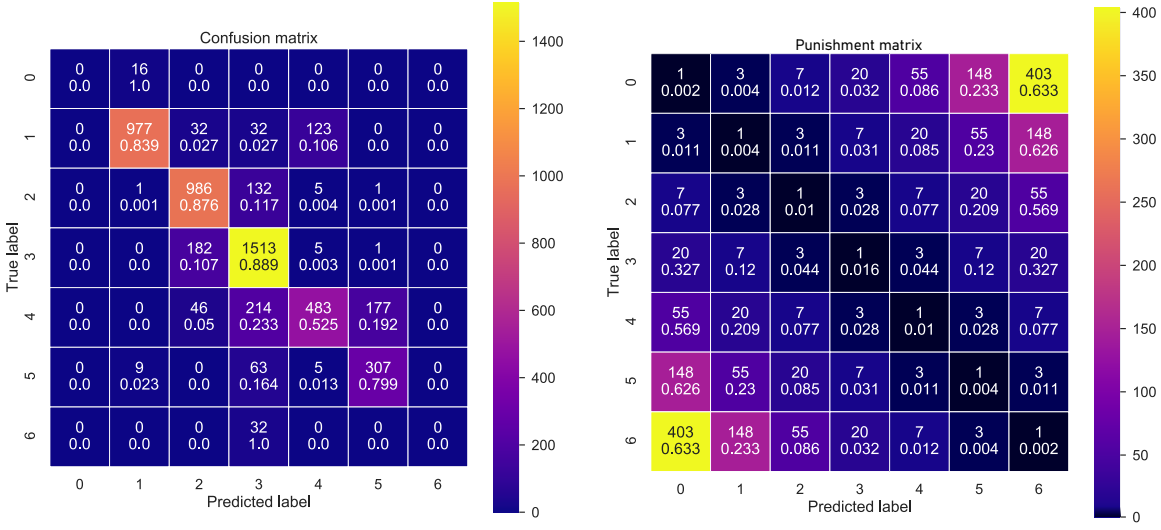
```
tf.keras.callbacks.EarlyStopping(
    monitor='val_loss', min_delta=0, patience=7, verbose=0,
    mode='max', baseline=None, restore_best_weights=True
)
```

Pre vizualizáciu pomocou prostredia tensorboard, je potrebné vytvoriť zapisovač súborov (*file writer*). Funkcia `tf.summary.create_file_writer` tento zapisovač vytvorí. Nasledujúca sekvencia príkazov zapíše skalárne hodnoty pre hodnoty trénovacej presnosti a straty, ktoré môžu byť neskôr analyzované v prostredí Tensorboard.

```
with train_summary_writer.as_default():
    tf.summary.scalar('epoch_loss',
                      float(tf.math.reduce_mean(loss_value)),
                      step=epoch)
    tf.summary.scalar('epoch_accuracy', float(train_acc), step=epoch)
```

Úprava stratových funkcií

V sekciách 4.4 a 4.5 je zhrnutý teoretický základ používaných stratových funkcií. Z implementačného hľadiska bolo potrebné upraviť stratovú funkciu CCE tak, aby bolo možné nastaviť vyššie váhy pre chyby, ktorých sa model dopúšťa a priradiť priamo úmerne vyšší multiplikátor tým misklasifikáciám, pri ktorých dochádza ku klasifikačným omylom o niekoľko tried. Pre tento účel bola implementovaná funkcia WCCE (*Weighted Categorical*



Obr. 6.1: **Vľavo**: Matica zámen po 5 epoche, z experimentu 6. **Vpravo**: Matica postihu.

Cross-Entropy). Bola vytvorená matica postihu viditeľná v pravej časti obrázku 6.1, pričom hodnota matice na mieste $M_P[y_{\text{pred}}, y_{\text{true}}]$ je definovaná vzťahom $e^{(|p-t|)}$. Potom váženú kategorickú krížová entropiu môžeme implementovať ako:

```
def weighted_categorical_crossentropy(weights):
    def internal_weighted_categorical_crossentropy(y_true, y_pred):
        index1 = K.argmax(y_pred, axis=1)
        index2 = K.argmax(y_true, axis=1)
        mul_mask = tf.gather_nd(weights, tf.stack((index1, index2), -1))
        return K.categorical_crossentropy(y_true, y_pred, from_logits=True)
        * mul_mask
    return internal_weighted_categorical_crossentropy
```

Vytvorenie vlastného trénovacieho cyklu

Pre zjednodušenie vizualizácie dát a možnosť evaluácie modelu po každej epoche bol vytvorený vlastný trénovací cyklus. Takéto manuálne nastavenie trénovania umožňuje vykresľovanie grafov a matic zámen, po skončení každej epochy po evaluácii modelu. Pre zachovanie grafovej exekúcie[2] je nutné využiť prostredie @tf.function, ktorým sa prepne predvolená exekúcia (*eager execution*) na režim exekúcie pomocou grafu. Funkcie v takto definovanom prostredí umožňujú optimalizáciu pomocou optimalizačného systému Grappler.

[2]Grafová exekúcia – znamená, že tenzorové výpočty sa vykonávajú ako graf TensorFlow

Kapitola 7

Experimenty

Princípom experimentov bolo zisťovanie ideálneho nastavenia hyperparametrov modelu, výber stratovej funkcie a optimalizátora s najlepšími výsledkami a úpravy augmentácie pre zvýšenie generalizovania modelu. Zmeny v parametroch v porovnaní s minulým experimentom sú znázornené v tabuľke parametrov tučným písmom.

Experimenty majú parametre, ktoré sa počas tréningu nemenili a to z dôvodu návrhu architektúry, prípadne charakterom datasetu. Medzi takéto nemenné parametre paria:

- **Veľkosť obrázka** – 255×255 px
- **Spôsob načítania datasetu** – zo zložky
- **Pomer trénovacich dát ku validačným** – $3 : 1$
- **Počet klasifikačných tried** – 7

Výsledky experimentov boli vykresľované niekoľkými spôsobmi:

- **Maticou zámen** – Vyjadruje, či a o akú hodnotu sa líši reálna hodnota výrezu od hodnoty určenej modelom. Taktiež znázorňuje podielové vyjadrenie určenia do konkrétnej triedy z pravdivých hodnôt z celkového počtu hodnôt danej triedy pravdivých hodnôt. Taktiež farebne znázorňuje počet hodnôt v danej pozícii určenej súradnicami x a y je z (počet prvkov, ktoré model určil ako x, a mali byť určené ako y je z). Túto hodnotu je možné zistiť z legendy umiestnenej v pravej časti grafu. Je v trénovacom cykle vykresľovaná po každej epoche.
- **Graf bodov** – Graficky pomocou bodov znázorňuje vzťah, reálnej hodnoty celkového počtu vozidiel na parkovisku a predpokladanej hodnoty určenej pomocou modelu. Bod na grafe je určený pomocou dvoch parametrov:
  - **Polohou** – Je definovaná súradnicami x a y, kde x je reálny počet vozidiel na parkovisku, y je predpokladaný počet vozidiel na parkovisku určený modelom.
  - **Veľkosťou** – Vyjadruje počet hodnôt výsledkov, prislúchajúcich k danej polohe.
- **Graf presnosti** – Znázorňuje trénovaciu a validačnú presnosť počas jednotlivých epoch. Graf je vytvorený pomocou vizualizačného balíka tensorboard.
- **Graf hodnoty straty** – Znázorňuje trénovaciu a validačnú hodnotu straty počas jednotlivých epoch. Graf je vytvorený pomocou vizualizačného balíka tensorboard.
- **Graf presnosti a chýb počas validácie** – Znázorňuje presnosť modelu na testovacom datasete. Presnosť v tomto prípade môže byť:

- **Presnosť úplná** – Presnosť po epoche na testovacích dátach.
- **Presnosť s prípustnou chybou 1. rádu** – Presnosť pri ktorej, sa omyl o 1 neuznáva ako chyba (napr. 2 namiesto 3).

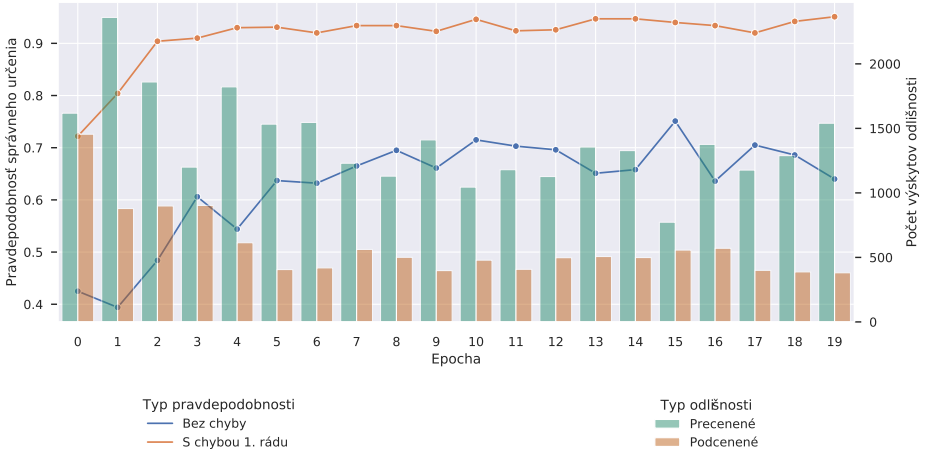
- **Graf miery rýchlosti učenia** – Graf je vytvorený pomocou vizualizačného balíka knižnice tensorflow – tensorboard.

7.1 Experimenty

7.1.1 Experiment 1

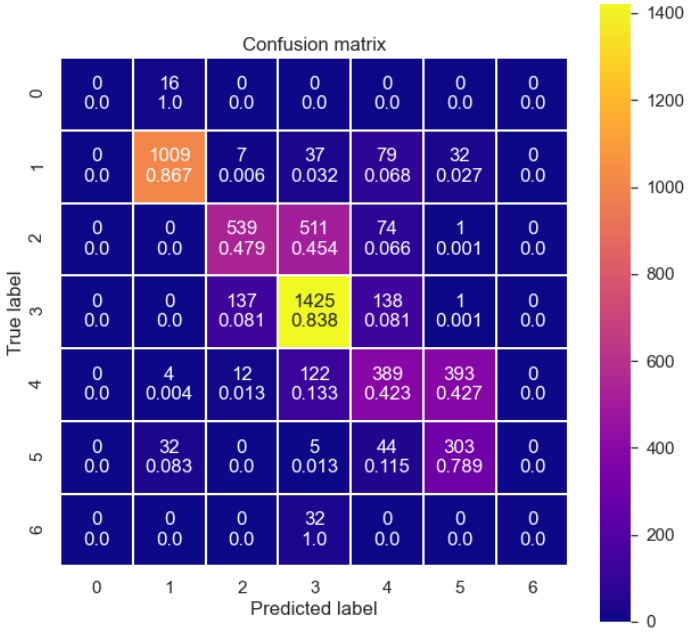
Parametre experimentu 1	
Stratová funkcia	Class-Balanced Softmax Cross-Entropy
Aktivačná funkcia	ReLU
Veľkosť mini-balíka	1
Optimalizátor	Adam
Miera učenia	0.001
Upadanie miery učenia	Žiadne
Počet epoch	19
Augmentácia dát	Náhodné horizontálne preklopenie, Náhodné otočenie (0.2)

Cieľ experimentu – Pokus zameraný na testovanie funkcionality triedne vyváženej funkcie softmax cross-entropy a jej vplyv na správne určenie triedy výrezu.



Obr. 7.1: Súhrnný výsledok experimentu 1 pomocou triedne vyváženej stratovej funkcie softmax cross-entropy na 20 epochách.

Výsledok experimentu – Z pokusu je viditeľné, že po poslednej epoche bolo 64.04% výrezkov bolo určených správne a 94.97% výrezkov bolo určených s chybou prvého rádu. Spolu bolo 1329 snímkov precenených a 388 snímkov podcenených. Je viditeľné, že stratová



Obr. 7.2: Matica zámen experimentu 1. Multiplikátor triedne vyváženej funkcie straty spôsobuje väčší pohyb nesprávne určených prvkov ku uhlopriečke a samotná funkcia straty pomáha k správnemu určeniu u málopočetne zastúpených tried.

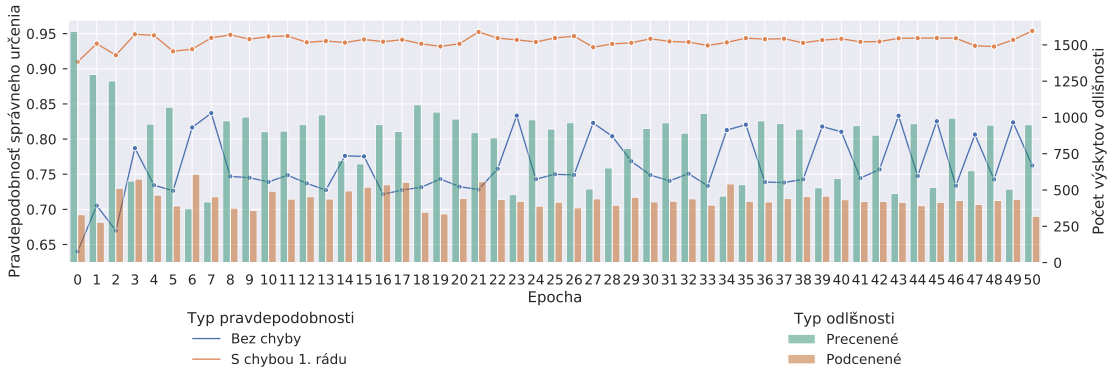
funkcia má kladný vplyv na výsledok pokusu, hlavne v prípade správneho určenia s chybou prvého rádu. V prípade správnej presnosti určenia však nejde o najlepší výsledok.

7.1.2 Experiment 2

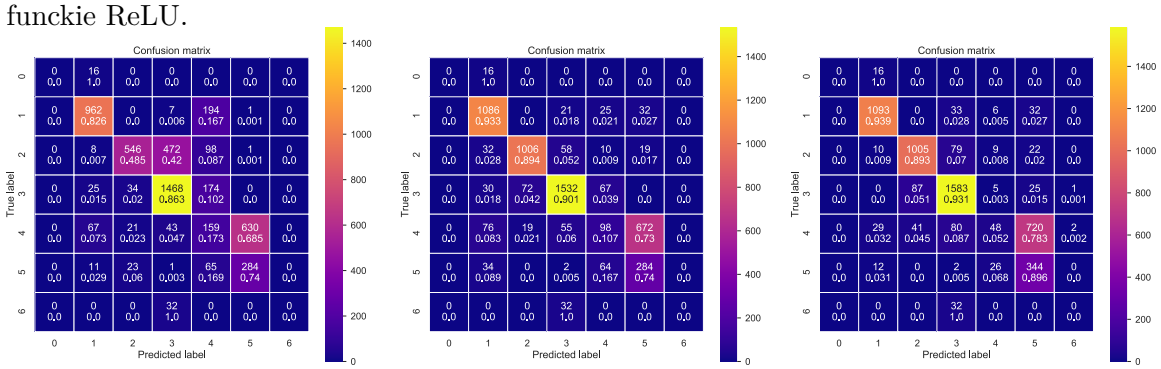
Parametre experimentu 2	
Stratová funkcia	**Sparse Categorical Cross-entropy**
Aktivačná funkcia	ReLU(Rectified Linear Unit)
Veľkosť mini-balíka	**32**
Optimalizátor	Adam
Miera učenia	0.001
Upadanie miery učenia	**Exponenciálne** (dr=0.96, ds=5000)
Počet epoch	**51**
Augmentácia dát	Náhodné horizontálne preklopenie, Náhodné otočenie (0.2)

Cieľ experimentu – Pokus, pri ktorom bolo skúmané, či je možné aj bez triedne vyváženej stratovej funkcie dosiahnuť prijateľnú presnosť určenia (95%) s chybou prvého rádu. Bola zvolená stratová funkcia SCC-E.
Výsledok experimentu – Presnosť správneho určenia je po 51 epoche pre úplnú presnosť 0.762 a pre presnosť s pripustením chyby 1. rádu 0.954. Bolo podcenených 247 výrezov a

precenených 950 výrezov. Rýchlosť evaluácie jedného výrezu je v priemere 0.05s. Nastavenie úpadku miery učenia spôsobuje spomalenie učenia každých 5000 trénovacích krokov. Funkcia ReLU výrazne vylepšila rýchlosť trénovania.



Obr. 7.3: Súhrnný výsledok experimentu 2 pomocou stratovej funkcie SCC-E a aktivačnej funckie ReLU.

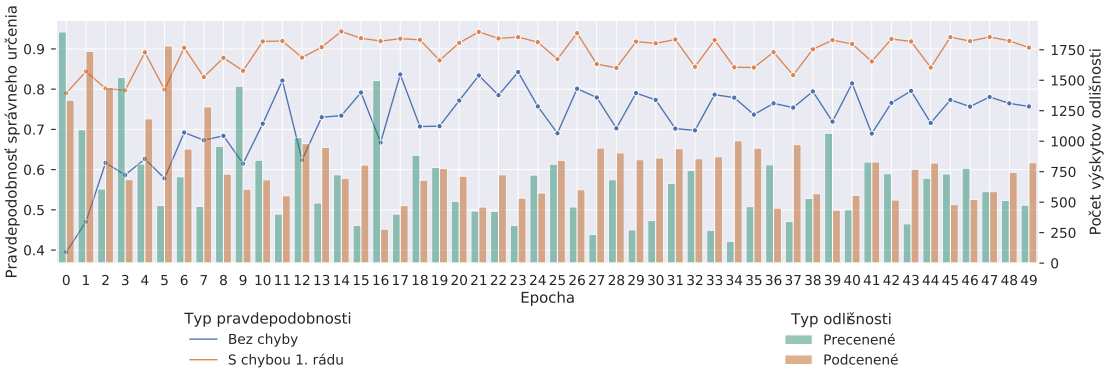


Obr. 7.4: Matica zámen experimentu 2. Pri zmene funkcie straty sa zmenilo rozloženie chýb v matici zámen. Dochádza k veľkej chybe v prípade výrezov s hodnotou 4. Pri ostatných klasifikáciách sú hodnoty prijateľné.

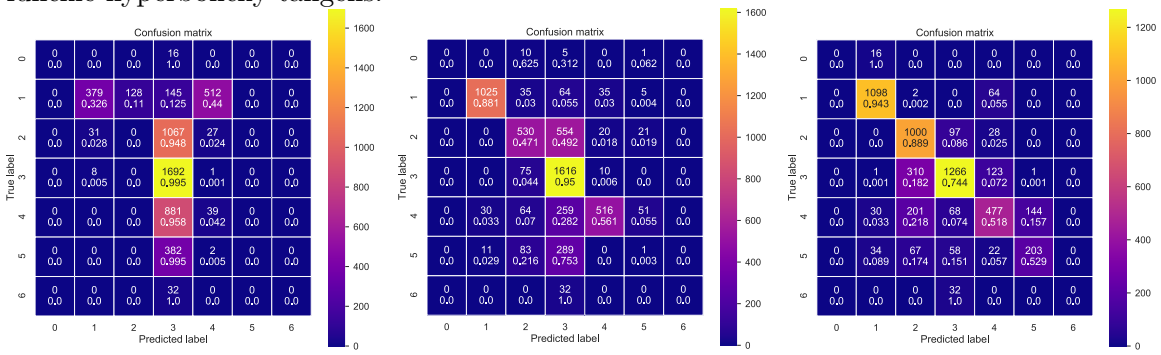
7.1.3 Experiment 3

Cieľ experimentu – Cieľom pokusu bolo overenie faktu, ktorý tvrdí, že funkcia ReLU je pre konvolučné neurónové siete najvhodnejšou aktivačnou funkciou. Stratovou funkciou experimentu je SCC-E. Overenie prebiehalo tak, že sa aktivačná funkciu ReLU nahradí inou aktivačnou funkciou, príčom sa ostatné parametre zachovajú v pôvodnom stave a následne sa výsledky týchto výstupov porovnajú. Pre tento experiment bola zvolená aktivačná funkcia hyperbolický tangens (tanh).
Výsledok pokusu – Presnosť správneho určenia je po 50 epoche pre úplnú presnosť 0.757 a pre presnosť s pripustením chyby 1. rádu 0.903, čo je výrezne nižšie ako u presnosti s aktivačnou funkciou ReLU. Bolo podcenených 475 výrezov a precenených 823 výrezov. Rýchlosť evaluácie jedného výrezu je v priemere 0.057 s. Nastavenie úpadku miery učenia spôsobuje spomalenie učenia každých 5000 trénovacích krokov (*decay steps*[ds]) so základom 0.96 (*decay rate*[dr]).

Parametre experimentu 3	
Stratová funkcia	Sparse Categorical Cross-entropy
Aktivačná funkcia	**hyperbolický tangens (tanh)**
Veľkosť mini-balíka	32
Optimalizátor	Adam
Miera učenia	0.001
Upadanie miery učenia	Exponenciálne (dr=0.96, ds=5000)
Počet epoch	**50**
Augmentácia dát	Náhodné horizontálne preklopenie, náhodné otočenie (0.2)



Obr. 7.5: Súhrnný výsledok experimentu 3 pomocou stratovej funkcie SCC-E a aktivačnej funckie hyperbolický tangens.



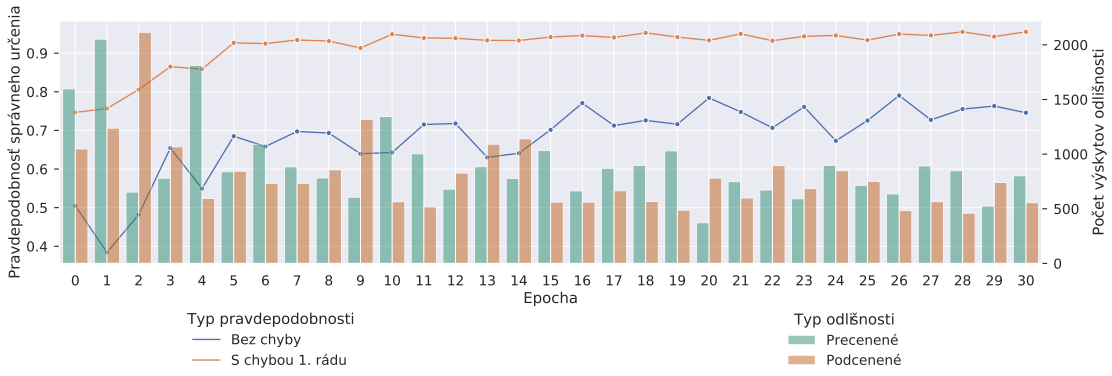
Obr. 7.6: Matica zámen v epochách $0, 25, 50$ počas experimentu 3. Po zmene aktivačnej funkcie dochádza k vyššiemu počtu chýb, pri klasifikácii hodnôt 2 a 3, ale dochádza k nižšej chybovosti pri klasifikácii hodnoty 4.

7.1.4 Experiment 4

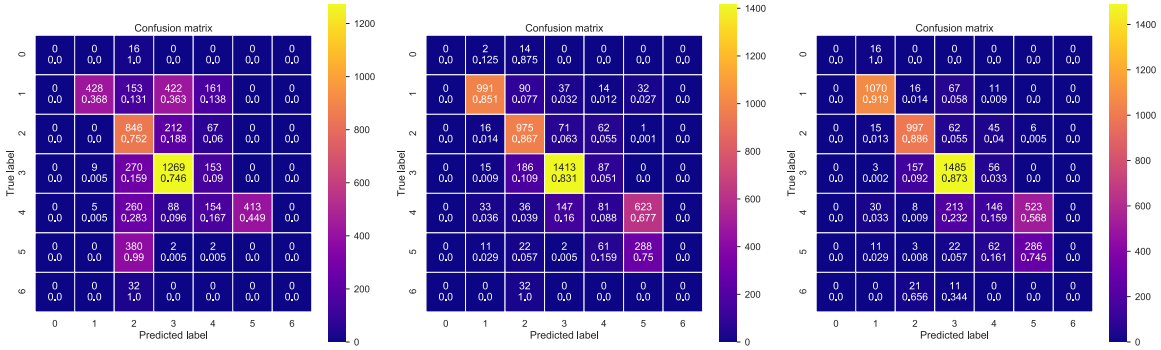
Cieľ experimentu – Cieľom experimentu je zistenie vplyvu výrazného zvýšenia miery augmentácie na výsledky tréningu.

Parametre experimentu 4	
Stratová funkcia	Sparse Categorical Cross-entropy
Aktivačná funkcia	ReLU(Rectified Linear Unit)
Veľkosť mini-balíka	32
Optimalizátor	Adam
Miera učenia	0.001
Upadanie miery učenia	Exponenciálne (dr=0.96, ds=5000)
Počet epoch	**31**
Augmentácia dát	Náhodné horizontálne preklopenie, **náhodná transformácia (0.2)**, náhodné otočenie (**0.4**), **náhodná zmena kontrastu (0.2)**.

Výsledky experimentu – Presnosť správneho určenia je po 50 epoche pre úplnú presnosť 0.746 a pre presnosť s pripustením chyby 1. rádu 0.955. Bolo podcenených 556 výrezov a precenených 802 výrezov. Rýchlosť evaluácie jedného výrezu je v priemere 0.049 s. Trénovacia presnosť bola po poslednej epoche 0.869 a stále dochádzalo k jej nárastu, čo naznačuje, že trénovanie modelu bolo ukončené predčasne. Riešením tohto problému je zvýšenie počtu epoch siete.



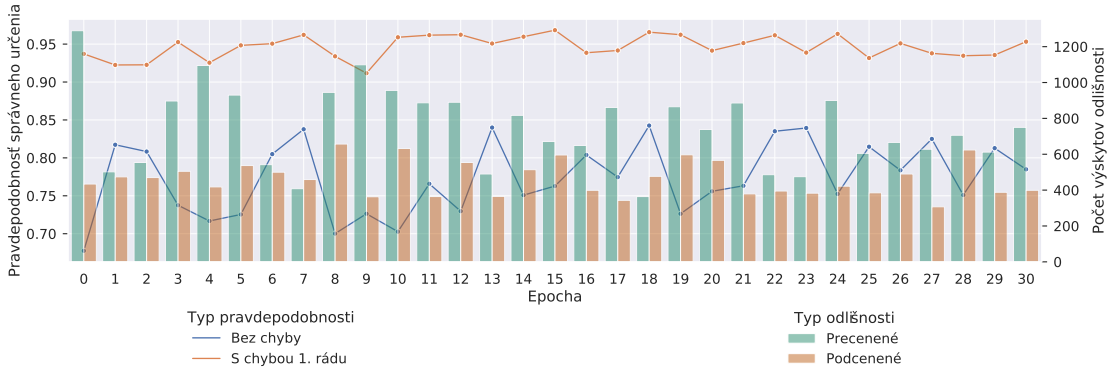
Obr. 7.7: Súhrnný výsledok experimentu 4. Experiment pridáva vyššiu mieru augmentácie pre zvýšenie generalizovania siete.



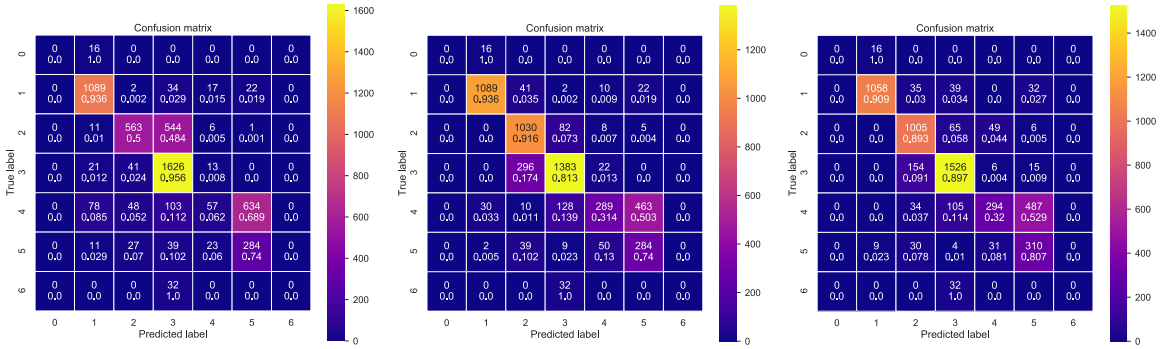
Obr. 7.8: Matica zámen experimentu 4 v epochách 0, 15, 30. Zvýšenie augmentácie spôsobuje spomalenie rýchlosti učenia siete, ale aj stabilizáciu výsledkov presnosti nad testovacími dátami.

7.1.5 Experiment 5

Parametre experimentu 5	
Stratová funkcia	Sparse Categorical Cross-entropy
Aktivačná funkcia	ReLU(Rectified Linear Unit)
Veľkosť mini-balíka	32
Optimalizátor	**RMSprop**
Miera učenia	0.001
Upadanie miery učenia	Exponenciálne (dr=0.96, ds=5000)
Počet epoch	31
Augmentácia dát	Náhodné horizontálne preklopenie, náhodné otočenie (**0.2**)



Obr. 7.9: Súhrnný výsledok experimentu 5. Zmena optimalizátora nemala zásadný vplyv na zmenu presnosti určovania výrezov testovacích dát.



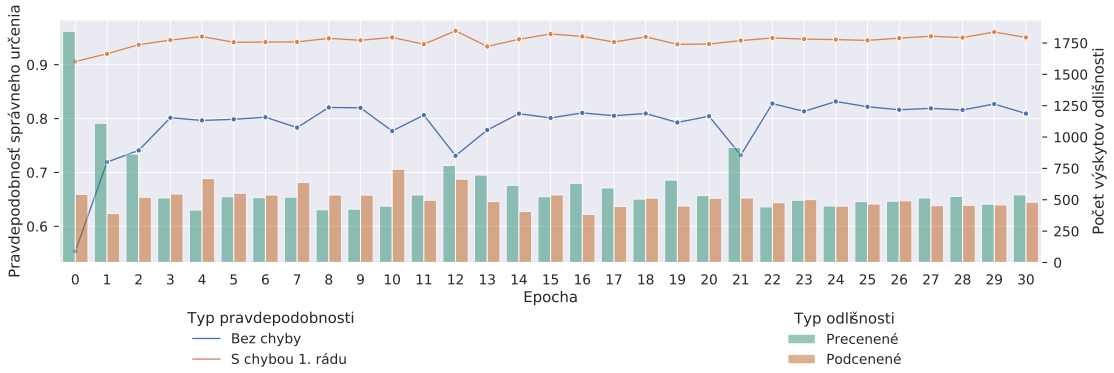
Obr. 7.10: Matica zámen experimentu 5 v epochách $0, 15, 30$. Zmena optimalizátora nemala zásadný vplyv na výsledky siete.

Cieľ experimentu – Cieľom experimentu je zistenie vplyvu zmeny optimalizátora Adam na optimalizátor RMSprop na výsledky tréningu.
Výsledky experimentu – Experiment ukázal, že zmena optimalizátora nemala zásadný vplyv na výslednú presnosť na miestach zhodujúcich sa epoch.

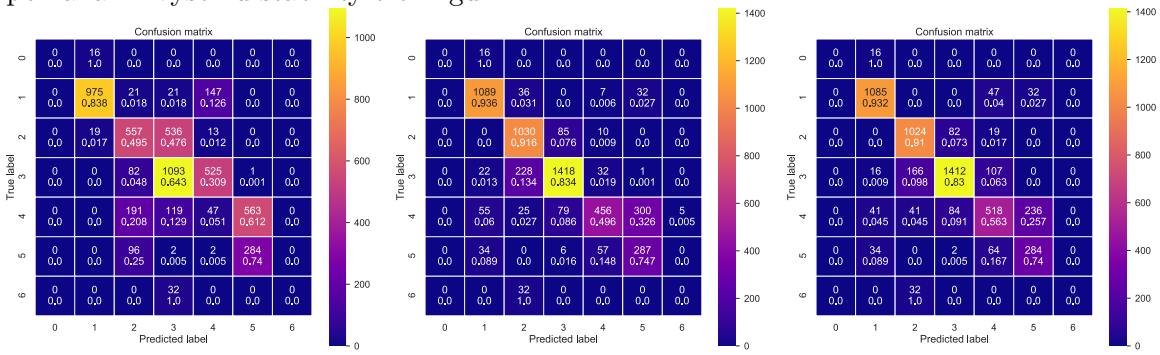
7.1.6 Experiment 6

Parametre experimentu 6	
Stratová funkcia	**Weightend Categorical Cross-entropy**
Aktivačná funkcia	ReLU(Rectified Linear Unit)
Veľkosť mini-balíka	32
Optimalizátor	**Adam**
Miera učenia	0.001
Upadanie miery učenia	Exponenciálne (dr=0.96, ds=5000)
Počet epoch	31
Augmentácia dát	Náhodné horizontálne preklopenie, náhodné otočenie (**0.2**).

Cieľ experimentu – Cieľom experimentu je zistenie vplyvu zmeny funkcie straty na váženú riedka kategorickú krížovú entropiu (*Weighted Categorical Cross-entropy*), ako sa dá porovnať s jej neváženým náprotivkom.
Výsledky experimentu – Presnosť správneho určenia je po 31 epoche pre úplnú presnosť 0.809 a pre presnosť s pripustením chyby 1. rádu 0.955. Bolo podcenených 480 výrezov a precenených 539 výrezov. Rýchlosť evaluácie jedného výrezu je v priemere 0.049 s. Trénovacia presnosť bola po poslednej epoche 0.937. Taktiež došlo k výraznému zlepšeniu v riešení problému s mylnou klasifikáciou do triedy 4.



Obr. 7.11: Súhrnný výsledok experimentu 6. Zmena stratovej funkcie spôsobila zvýšenie presnosti správneho určenia sieťou bez chyby, ale aj s chybou prvého rádu. Váha taktiež pomáha k zvýšeniu stability tréningu.

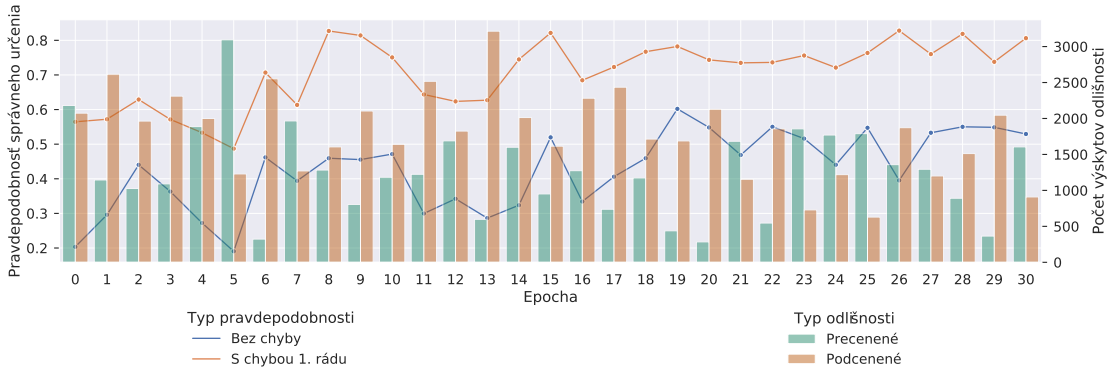


Obr. 7.12: Matica zámen experimentu 5 v epochách $0, 15, 30$. Zmena stratovej funkcie zlepšila výsledky siete a čiastočne vyriešila problém so zlým určovaním do triedy 4. Stále existujú výrezy ktoré sú misklasifikované o niekoľko tried, ale ich počet je výrazne nižší ako vo funkcii bez váh.

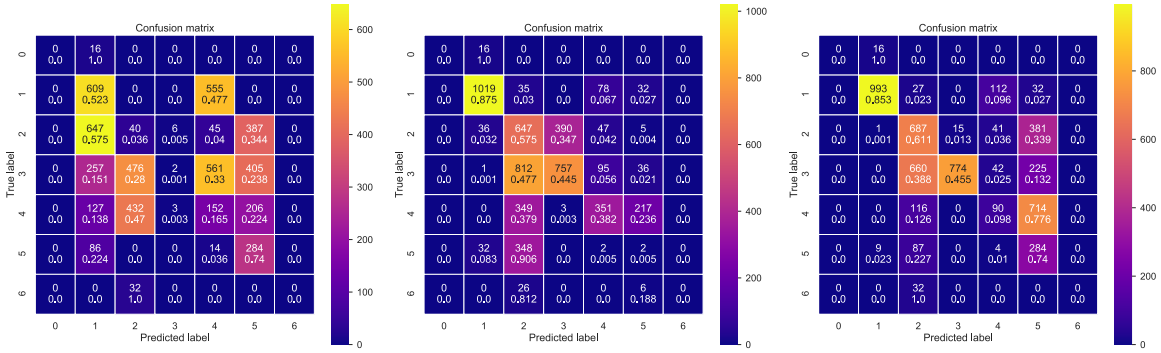
7.1.7 Experiment 7

Cieľ experimentu – Cieľ experimentu bolo zistiť, či zvýšenie augmentácie zlepší výsledky určovania testovacích výrezkov dát so stratovou funkciou WCC-E. Je nutné zistiť polaritu vplyvu, akú výrazná augmentácia prináša.
Výsledky experimentu – Z výsledkov je možné pozorovať, že zvýšená miera augmentácie, má negatívny vplyv na trénovaciu (0.695 proti 0.937), validačnú (0.776 proti 0.966) aj testovaciu (0.530 proti 0.809) presnosť modelu.

Parametre experimentu 7	
Stratová funkcia	Weightend Categorical Cross-entropy
Aktivačná funkcia	ReLU(Rectified Linear Unit)
Veľkosť mini-balíka	32
Optimalizátor	Adam
Miera učenia	0.001
Upadanie miery učenia	Exponenciálne (dr=0.96, ds=5000)
Počet epoch	31
Augmentácia dát	Náhodné horizontálne preklopenie, **náhodná transformácia (0.2)**, náhodné otočenie (**0.4**), **náhodná zmena kontrastu (0.2)**.



Obr. 7.13: Súhrnný výsledok experimentu 6. Evaluácia po 31 epoche má slabé výsledky. Riešením je zvýšenie počtu epoch, prípade zníženie miery augmentácie.



Obr. 7.14: Matica zámen experimentu 7 v epochách $0, 15, 30$. Výrazné zvýšenie augmentácie má negatívny vplyv na trénovanie siete so stratovou funkciou WCC-E.

Kapitola 8

Záver

Práca spracovávajúca iný pohľad na dobre známy problém obsadenosti parkoviska. Zvolený prístup rozdeľuje parkovisko na sériu zón tvaru štvorca, pričom zóna môže obsahovať od 0 po 6 vozidiel, následne sa určujú hodnoty týchto zón pomocou natrénovaného klasifikátoru, rozlišujúceho 7 tried.

Hlavné úspechy tohto prístupu sú relatívne vysoká rýchlosť estimácie, ktorá je v priemerne 0.049 s na snímok, jednoduchosť modelu, či nutnosť len jedného prístupu od používateľa pri spracovaní zón. Práca taktiež rieši problém nevyváženého datasetu vznikajúceho pri určení konečného intervalu, v ktorom sa môže počet vozidiel v zóne nachádzať, pomocou triedne vyváženej stratovej funkcie softmax cross-entropy. Taktiež bola implementovaná váhová verzia stratovej funkcie kategorickej krížovej entropie ako altrenatíva ku triedne vyváženej stratovej funkcii, ktorá sa tiež vyznačuje veľmi dobrými výsledkami.

Boli tiež implementované dve GUI aplikácie, pre vytváranie výrezkov a hodnotenie. Aplikácia pre vytváranie výrezov zón uľahčuje používateľovi vytváranie zón pomocou dvoch kliknutí (ľavého a pravého) do plochy s obrázkom. Aplikácia umožňuje úpravy ako posun zóny, zväčšenie, zmenšenie, označenie a následne vytvorenie výrezov pre jednu fotografiu, či pre celý dataset obsahujúci fotografie snímané pod rovnakým uhlom. Hodnotenie zón je možné následne previesť pomocou vytvoreného rýchleho anotačného nástroja umožňujúceho úpravu a prehliadanie už ohodnotených snímkov. Aplikácia taktiež obsahuje modul pre prehliadanie výstupu z modelu pre porovnanie pravdivých a predpokladaných hodnotení pre zóny.

Pri pokračovaní tejto práce by bolo vhodné, pracovať s automatizáciou činnosti, ktoré v súčasnosti musí zadávať používateľ manuálne – pozície zón. Túto činnosť by bolo možné zautomatizovať, buď regresívnym riešením, zaisťujúcim polohu parkovacích miest na základe parkovacích čiar, prípadne na základe vozidiel. Následne by sa zhluky parkovacích miest ohraničili zónovým štvorcom. V prípade parkovísk snímaných pomocou dronu by bolo jednoduché riešenie, pri ktorom by sa parkovisko prekrylo mriežkou s jednotnou veľkou štvorcovej plochy tejto mriežky, tak aby sa v každom štvorci tejto mriežky nachádzalo 0 až 6 vozidiel, keďže v tomto prípade nie je nutné riešiť problém s uhlom v ktorom kamera sníma parkovisko. Tieto štvorce by pokračovali do neurónovej siete, ktorá by určovala či sa jedná o plochu parkoviska alebo nie a následne do vytvoreného klasifikátora.

Práca bola prezentovaná na študentskej konferencii Excel@FIT.

Literatúra

[1] Acharya, D.; Yan, W.; Khoshelham, K.: Real-time image-based parking occupancy detection using deep learning. In *Research@ Locate*, 04 2018, s. 33–40.

[2] Aich, S.; Stavness, I.: Improving Object Counting with Heatmap Regulation. *CoRR*, ročník abs/1803.05494, 2018, 1803.05494.
URL http://arxiv.org/abs/1803.05494

[3] Almeida, P.; Soares de Oliveira, L.; Jr, A.; aj.: PKLot - A Robust Dataset for Parking Lot Classification. *Expert Systems with Applications*, ročník 42, 02 2015, doi:10.1016/j.eswa.2015.02.009.

[4] Amato, G.; Carrara, F.; Falchi, F.; aj.: Car parking occupancy detection using smart camera networks and deep learning. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*, IEEE, 2016, s. 1212–1217.

[5] Amato, G.; Carrara, F.; Falchi, F.; aj.: Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, ročník 72, 2017: s. 327–334.

[6] Burkov, A.: *The Hundred-Page Machine Learning Book*. Quebec City, Canada: Andriy Burkov (January 13, 2019), 2019.

[7] Chen, K.; Loy, C. C.; Gong, S.; aj.: Feature Mining for Localised Crowd Counting. In *Proceedings of the British Machine Vision Conference*, BMVA Press, 2012, ISBN 1-901725-46-4, s. 21.1–21.11, doi:http://dx.doi.org/10.5244/C.26.21.

[8] Chen, L.-C.; Sheu, R.-K.; Peng, W.-Y.; aj.: Video-Based Parking Occupancy Detection for Smart Control System. *Applied Sciences*, ročník 10, č. 3, 2020, ISSN 2076-3417, doi:10.3390/app10031079.
URL https://www.mdpi.com/2076-3417/10/3/1079

[9] Chen, Y.-Y.; Lin, Y.-H.; Kung, C.-C.; aj.: Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes. *Sensors*, ročník 19, 05 2019: str. 2047, doi:10.3390/s19092047.

[10] Cohen, J. P.; Lo, H. Z.; Bengio, Y.: Count-ception: Counting by Fully Convolutional Redundant Counting. *CoRR*, ročník abs/1703.08710, 2017, 1703.08710.
URL http://arxiv.org/abs/1703.08710

[11] Cui, Y.; Jia, M.; Lin, T.-Y.; aj.: Class-Balanced Loss Based on Effective Number of Samples. 2019, 1901.05555.

[12] Dobeš, P.; Špaňhel, J.; Bartl, V.; aj.: Density-Based Vehicle Counting with Unsupervised Scale Selection. In *2020 Digital Image Computing: Techniques and Applications (DICTA)*, 2020, s. 1–8, doi:10.1109/DICTA51227.2020.9363401.

[13] Drouyer, S.; de Franchis, C.: PARKING OCCUPANCY ESTIMATION ON SENTINEL-1 IMAGES. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, ročník V-2-2020, 08 2020: s. 821–828, doi:10.5194/isprs-annals-V-2-2020-821-2020.

[14] François, C.: *Deep learning with Python*. Manning Publications Co., 2018.

[15] Girshick, R.; Donahue, J.; Darrell, T.; aj.: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, s. 580–587, doi:10.1109/CVPR.2014.81.

[16] Goodfellow, I.; Bengio, Y.; Courville, A.: *Deep learning*. The MIT Press, 2017.

[17] Graham, B.: Fractional Max-Pooling. *CoRR*, ročník abs/1412.6071, 2014, 1412.6071.
URL http://arxiv.org/abs/1412.6071

[18] Han, S.; Pool, J.; Tran, J.; aj.: Learning both Weights and Connections for Efficient Neural Network. In *Advances in Neural Information Processing Systems*, ročník 28, editace C. Cortes; N. Lawrence; D. Lee; M. Sugiyama; R. Garnett, Curran Associates, Inc., 2015.
URL https://proceedings.neurips.cc/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf

[19] Hsieh, M.-R.; Lin, Y.-L.; Hsu, W. H.: Drone-based Object Counting by Spatially Regularized Regional Proposal Network. 2017, 1707.05972.

[20] Ichihashi, H.; Notsu, A.; Honda, K.; aj.: Vacant Parking Space Detector for Outdoor Parking Lot by Using Surveillance Camera and FCM Classifier. In *2009 IEEE International Conference on Fuzzy Systems*, FUZZ-IEEE'09, IEEE Press, 2009, ISBN 9781424435968, str. 127–134.

[21] ImageNet. http://www.image-net.org/about, accessed: 12-04-2021.

[22] Krizhevsky, A.; Sutskever, I.; Hinton, G.: ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems*, ročník 25, 01 2012, doi:10.1145/3065386.

[23] Lecun, Y.; Bottou, L.; Bengio, Y.; aj.: Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, ročník 86, č. 11, 1998: s. 2278–2324, doi:10.1109/5.726791.

[24] Lempitsky, V.; Zisserman, A.: Learning to count objects in images. 01 2010, s. 1324–1332.

[25] Lowe, D. G.: Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, ročník 60, č. 2, Listopad 2004: s. 91–110, ISSN 0920-5691, doi:10.1023/B:VISI.0000029664.99615.94.
URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[26] Mundhenk, T. N.; Konjevod, G.; Sakla, W. A.; aj.: A Large Contextual Dataset for Classification, Detection and Counting of Cars with Deep Learning. *CoRR*, ročník abs/1609.04453, 2016, 1609.04453.
URL http://arxiv.org/abs/1609.04453

[27] Nwankpa, C.; Ijomah, W.; Gachagan, A.; aj.: Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv preprint arXiv:1811.03378*, 12 2018.

[28] Paidi, V.; Fleyeh, H.; Nyberg, R.: Deep learning-based vehicle occupancy detection in an open parking lot using thermal camera. *IET Intelligent Transport Systems*, ročník 14, 09 2020: s. 1295–1302, doi:10.1049/iet-its.2019.0468.

[29] Powers, D. M.: Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[30] Razakarivony, S.; Jurie, F.: Vehicle Detection in Aerial Imagery : A small target detection benchmark. *Journal of Visual Communication and Image Representation*, ročník 34, 03 2015, doi:10.1016/j.jvcir.2015.11.002.

[31] Rosasco, L.; Vito, E. D.; Caponnetto, A.; aj.: Are loss functions all the same? *Neural computation*, ročník 16, č. 5, 2004: s. 1063–1076.

[32] Tanner, F.; Colder, B.; Pullen, C.; aj.: Overhead imagery research data set — an annotated data library tools to aid in the development of computer vision algorithms. In *2009 IEEE Applied Imagery Pattern Recognition Workshop (AIPR 2009)*, 2009, s. 1–8, doi:10.1109/AIPR.2009.5466304.

[33] Tătulea; Călin; Brad; aj.: An Image Feature-Based Method for Parking Lot Occupancy. *Future Internet*, ročník 11, 08 2019: str. 169, doi:10.3390/fi11080169.