Technická univerzita v Košiciach

Fakulta elektrotechniky a informatiky

Automatická detekcia TLE na snímkach AMOS pomocou hlbokého učenia

Bakalárska práca

2022 **Petra Kamenská**

Technická univerzita v Košiciach

Fakulta elektrotechniky a informatiky

Automatická detekcia TLE na snímkach AMOS pomocou hlbokého učenia

Bakalárska práca

Študijný program:	Hospodárska informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra kybernetiky a umelej inteligencie (KKUI)
Školiteľ:	doc. Ing. Peter Butka, PhD.
Konzultant:	Ing. Viera Maslej Krešňáková, PhD.
	RNDr. Šimon Mackovjak, PhD.

Košice 2022 **Petra Kamenská**

Abstrakt v SJ

V súčasnosti nie je vytvorený jednotný spôsob automatického pozorovania TLE javov. K automatickému pozorovaniu je nutné vytvoriť aj automatické metódy na detekovanie javov a extrakciu informácií. Keďže tieto javy majú nepravidelnú a sporadickú štruktúru a vstupných dát z meracích prístrojov je veľa, je vhodné ich detekovať pomocou metód hlbokého učenia. Cieľom bakalárskej práce bude oboznámiť sa so základmi hlbokého učenia, konvolučných neurónových sietí, sieti YOLO a s ich využitím detekovať špecifické udalosti – TLE javy na obrázkoch z AMOS kamery.

Kľúčové slová

detekcia objektov, hlboké učenie, TLE javy, anotačný projekt, AMOS kamera, YOLO

Abstrakt v AJ

Nowadays, there is no uniform method for automatic observation of transient luminous events. For automatic observation, it is also necessary to create automatic methods for event detection and information extraction. Since these events have an irregular and radical structure and a lot of input data from measuring instruments, it is appropriate to use deep learning methods. The aim of the bachelor thesis is to get acquainted with the basics of deep learning, convolutional neural networks and the YOLO network and with their usage to detect specific events - TLE phenomena in images from AMOS camera.

Kľúčové slová v AJ

object detection, deep learning, transient luminous events, annotation project, AMOS camera, YOLO

65639

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Katedra kybernetiky a umelej inteligencie

ZADANIE BAKALÁRSKEJ PRÁCE

Študijný odbor: **Informatika**

Študijný program: **Hospodárska informatika**

Názov práce:

Automatická detekcia TLE na snímkach AMOS pomocou hlbokého učenia

Automatic detection of TLE in AMOS images using deep learning

Študent: **Petra Kamenská**

Školiteľ: **doc. Ing. Peter Butka, PhD.**

Školiace pracovisko: **Katedra kybernetiky a umelej inteligencie**

Konzultant práce: **Ing. Viera Maslej Krešňáková, PhD., RNDr. Šimon Mackovjak, PhD.**

Pracovisko konzultanta: **Ústav experimentálnej fyziky SAV**

Pokyny na vypracovanie bakalárskej práce:

1. Podať teoretický prehľad problematiky metód hlbokého učenia, s dôrazom na vhodné metódy pre automatickú detekciu TLE udalostí v obrazových dátach.
2. Analyzovať a predspracovať dostupnú dátovú množinu celoblohových snímok kamery AMOS.
3. Pripraviť anotačný projekt a získať anotácie pre detekciu udalostí TLE na snímkach.
4. Navrhnúť, realizovať a vyhodnotiť model hlbokého učenia pre detekciu TLE udalostí v danej množine dát.
5. Vypracovať dokumentáciu podľa pokynov katedry a vedúceho práce.

Jazyk, v ktorom sa práca vypracuje: slovenský

Termín pre odovzdanie práce: 27.05.2022

Dátum zadania bakalárskej práce: 29.10.2021

.....

prof. Ing. Liberios Vokorokos, PhD.

dekan fakulty

Čestné vyhlásenie

Vyhlasujem, že som bakalársku prácu vypracovala samostatne s použitím uvedenej odbornej literatúry.

Košice 27. 5. 2022 .....

Vlastnoručný podpis

Poďakovanie

Chcela by som sa srdečne poďakovať môjmu školiteľovi, doc. Ing. Petrovi Butkovi, PhD. za jeho cenné pripomienky a rady k mojej práci, takisto za zorganizovanie anotovacích sedení. Veľká vďaka patrí všetkým konzultantom práce. V prvom rade Ing. Viere Maslej Krešňákovej, PhD. za dlhé videohovory plné rád a pomoci ohľadom trénovania a testovania modelu, či písaní bakalárskej práce. Taktiež za jej ochotu a trpezlivosť. Ďalej RNDr. Šimonovi Mackovjakovi, PhD. za ochotu a rady v oblasti domény, a za prinášanie nových nápadov k riešeniu. Samuelovi Amrichovi ďakujem za prvotné uvedenie do tématiky domény a podrobné vysvetlenie s poskytnutím materiálov. Školiteľovi a všetkým konzultantom ďakujem za ich pomoc a obetovanie času pri anotovaní a pomáhaní študentom s nájdením správnych javov. V neposlednej rade ďakujem mojim spolužiakom na Hospodárskej informatike za spoluprácu pri anotovaní vstupného datasetu. Bez nich by nebolo možné uskutočniť túto prácu. Táto práca vznikla najmä vďaka Doc. RNDr. Jurajovi Tóthovi, PhD. ktorý poskytol všetky vstupné dáta, a ochotne a rýchlo reagoval na moje požiadavky. Veľké ďakujem patrí aj mojej rodine za ich morálnu podporu.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam symbolov a skratiek

AGO AGO Modra

AI Artificial Intelligence

AMOS All-Sky Meteor Orbit System

AP Average Precision

ARBO Arborétum Tesárske Mlyňany

CMOS Complementary Metal Oxide Semiconductor

CNN Convolutional Neural Network

CSP Cross Stage Partial Networks

DL Deep Learning

FN False Negative

FP False Positive

FPN Feature Pyramid Network

GB Gigabyte

IoU Intersection over Union

KNM Kysucké Nové Hvezdáreň Mesto

LED Light-emitting Diode

MaPMT Multi-anode Photomultipliers

MCP MicroCameras and Photometers

ML Machine Learning

MNIST Modified National Institute of Standards and Technology

mPA mean Average Precision

P Precision

R Recall

ReLU Recified Linear Unit

SBC Single Board Computer

SVHN Street View House Numbers

SVMN Slovenská videometeorická sieť

TARANIS Tool for the Analysis of Radiations from lightNings and Sprites

TGF Transforming Growth Factor

TLE Transient Luminous Events

TN True Negative

TP True Positive

YOLO You Only Look Once

Úvod

Témou tejto bakalárskej práce je detekcia objektov. Detekcia objektov je pomerne novou rozvíjajúcou sa metódou v oblasti počítačového videnia. My ako ľudia vieme pomerne bleskovo a presne identifikovať objekty v reálnom svete, na fotografii, či na videu. Síce technológie na rozpoznávanie objektov sa neustále zlepšujú, stále im chýba ľudská intuícia, vďaka ktorej človek dokáže lepšie definovať, čo vidí. Majú však obrovskú výhodu, a to, že dokážu spracovať obrovské množstvo dát za veľmi krátku dobu, čo je pre ľudí nemožné. So zberom veľkého množstva dát prichádza aj pojem *Big data*. Tie sú prítomné už v každej oblasti. Vesmírny výskum nie je výnimkou. Existuje mnoho systémov, ktoré sledujú oblohu za istým účelom a ukladajú stovky snímok za minútu. Takým systémom je aj AMOS, ktorý sníma celú oblohu predovšetkým kvôli meteorom. Na snímkach sa však môžu objaviť aj TLE javy, ktoré sa vyskytujú len počas búrok. TLE javy a ich vznik sú doposiaľ veľmi málo preskúmanou oblasťou. Našou úlohou bolo detekovať tieto TLE javy na snímkach z kamery AMOS. Keďže v súčasnej dobe metódy hlbokého učenia prevyšujú výsledky štandardných metód strojového učenia, rozhodli sme sa využiť špeciálnu neurónovú sieť s cieľom detekovať TLE javy. Tieto javy sú nepravidelné a sporadické štruktúry, hlboké učenie je perfektné metóda na vyriešenie tohto problému. Na trénovanie neurónovej siete potrebujeme označené dáta. Nakoľko doposiaľ neexistuje žiaden oficiálny dataset TLE javov, je potrebné manuálne anotovať tieto udalosti a pripraviť tak dataset na trénovanie neurnových sietí. Takáto spolupráca medzi vesmírnymi vedcami a informatikmi prináša veľa výhod. Uľahčuje prácu astronómom a hlavne šetrí množstvo času. Takýmto spojením je aj predložená bakalárska práca. Vznikla v spolupráci s RNDr. Šimonom Mackovjakom, PhD. zo Slovenskej akadémie vied, konkrétne z Oddelenia kozmickej fyziky a Doc. RNDr. Jurajom Tóthom, PhD., ktorý pôsobí na Fakulte Matematiky Fyziky a Informatiky na Univerzite Komenského v Bratislave, ktorého výskum sa zaoberá touto tématikou, a ktorému vďačíme za dáta z kamier AMOS.

V prvej kapitole budú čitateľovi predstavené základy detekcie objektov pomocou hlbokého učenia. Podrobne sú vysvetlené pojmy ako hlboké učenie, umelá inteligencia a strojové učenie. Kapitola obsahuje rozdelenie typov učení a typov spracovania obrazu. Zvyšok kapitoly sa venuje konkrétne detekcii objektov pomocou sieti YOLO.

Ďalšia kapitola obsahuje popis pojmu *crowdsourcing* spolu s vysvetlením platformy Zooniverze, ktorá je používaná na anotačné projekty.

Tretia kapitola sa venuje popisu domény. Konkrétne sa čitateľ môže dozvedieť základné teórie ohľadom TLE bleskov spolu s analýzou súčasného stavu detekcie týchto TLE javov. Sú tam popísané tri projekty, ktoré sa snažia o sledovanie a detekciu TLE.

Praktická časť je popísaná v kapitole číslo štyri. Ako bolo spomenuté vyššie prvým krokom pred samotnými experimentami bolo vytvorenie datasetu. Pre úspešnú klasifikáciu sme zvolili iteratívny postup, ktorý bude tiež popísaný v tejto kapitole. Záver práce obsahuje zhrnutie výsledkov a možnú budúcu prácu.

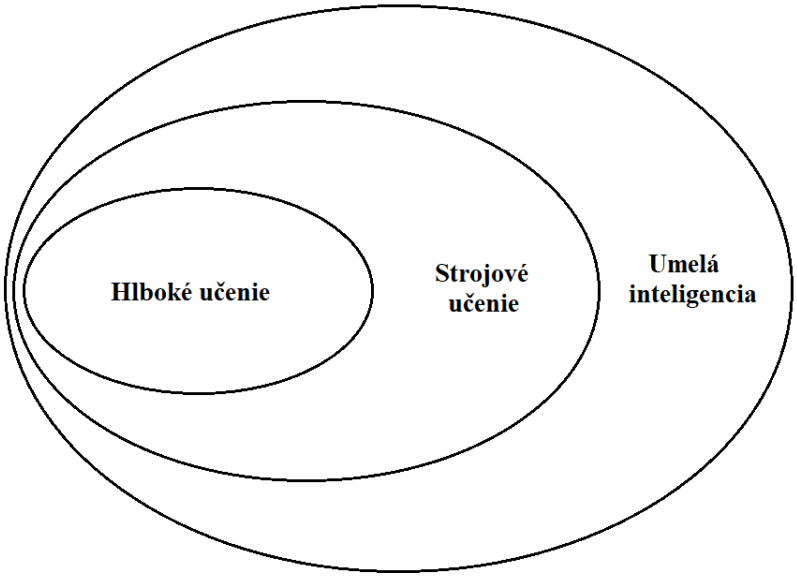
1 Úvod do detekcie objektov pomocou hlbokého učenia

1.1 Hlboké učenie

Už dávno v minulosti, ľudia snívali o vytvorení stroja, ktorý bude sám myslieť. V súčasnosti oblasť umelej inteligencie (AI) prudko napreduje. Počítače už dokážu nahradiť niektorú prácu ľudí, porozumieť reči, obrazom, pomáhať doktorom pri diagnostikovaní chorôb či pomáhať vo vedeckom výskume. Hlboké a strojové učenie sú v dnešnej dobe horúcou témou článkov a diskusií. Odborníci aj obyčajní ľudia rozoberajú, čo nám budúcnosť v tejto oblasti prinesie. Budú autá jazdiť samé? Stratia ľudia prácu a nahradia ich roboti? Preberú roboti nadvládu nad svetom? To sú pozitívne aj negatívne dôsledky, ktoré predpovedajú ľudia. Momentálne umelá inteligencia, dokáže riešiť relatívne priamočiare problémy s danou postupnosťou krokov. Jednoduché činnosti, ktoré ako ľudia vykonávame dennodenne, ako napríklad rozpoznávanie zvierat či priradenie významu k slovám, je pre umelú inteligenciu ťažším orieškom, kvôli zložitému vysvetleniu postupnosti krokov z ľudskej strany. Hlboké učenie sa nazýva hlboké z dôvodu, že modely sú zostavené z množstva vrstiev a jeho koncepty sú poukladané na seba. Prvým úspechom umelej inteligencie bol šachový systém Deep Blue od firmy IBM, ktorý v roku 1997 porazil v šachu vtedajšieho majstra sveta Garryho Kasparova Goodfellow et al. (2016). Šach je však jednou z tých činností, ktorá ma pevne ohraničené pravidlá. Preto sa to umelá inteligencia nemala problém naučiť, aj keď jej to trvalo dlhé roky kvôli kapacite výpočtovej sily v tých rokoch.

1.1.1 Umelá inteligencia, Strojové učenie, Hlboké učenie

V tejto kapitole sa budem venovať definovaniu pojmov umelá inteligencia, strojové a hlboké učenie. Ako spolu súvisia a aké sú rozdiely medzi nimi. Štruktúru týchto troch pojmov, môžete vidieť na Obrázku 1 – 1.



Obrázok 1 – 1 Porovnanie vzťahu medzi umelou inteligenciou, strojovým a hlbokým učením.

Umelá inteligencia (angl. *Artificial intelligence*) skratka AI, je inteligencia, ktorú prejavujú stroje. Zrodila sa v 50. rokoch 20. storočia, keď vedci v oblasti počítačov riešili otázky, či sa dá vytvoriť počítač, ktorý bude sám rozmýšľať. Umelá inteligencia je odvetvie vedy a technológia, ktorá vytvára inteligentné stroje a počítačové programy na vykonávanie rôznych úloh, ktoré vyžadujú ľudskú inteligenciu (Ertel, 2018). Snaží sa automatizovať intelektuálne úlohy, ktoré vykonávajú ľudia každý deň, ide o kognitívne funkcie ako učenie alebo riešenie problémov. Cieľom umelej inteligencie je vykonávať tieto činnosti lepšie a rýchlejšie ako človek. Za inteligentné je považované akékoľvek zariadenie, ktoré vníma prostredie a podniká kroky na maximalizáciu úspechu v nejakom cieli (Ongsulee, 2017). V AI sa používa veľa nástrojov vrátane rôznych verzií vyhľadávania, matematická optimalizácia, logika, metódy založené na pravdepodobnosti a ekonomika. V súčasnosti umelá inteligencia dokáže rozpoznávať ľudskú reč, súťažiť v strategických herných systémoch, pomáhať pri riadení áut či interpretovať komplikované údaje. AI sa rozmáha a pomáha v rôznych oblastiach vedy ako matematika, psychológia, lingvistika, filozofia, neuroveda, umelá psychológia a mnohé ďalšie.

Strojové učenie (angl. *Machine learning*) skratka ML, je to oblasť informatiky, ktorá podľa Ongsulee (2017) dáva počítačom schopnosť učiť sa bez toho, aby boli na to explicitne naprogramované. Strojové učenie vyplýva z otázky: mohol by počítač ísť nad rámec toho, čo my a naučiť sa samostatne vykonávať zadanú úlohu? Namiesto toho, aby programátori vytvárali pravidlá pre spracovanie údajov ručne, mohol by sa počítač automaticky naučiť tieto pravidlá prezeraním údajov? V klasickom programovaní, ľudia zadávajú pravidlá a údaje, ktoré budú spracované podľa týchto pravidiel a na výstupe sú odpovede. Pri strojovom učení, ľudia zadávajú údaje, ako aj odpovede očakávané k údajom a výstupom sú pravidlá. Tieto pravidlá sa potom môžu použiť na nové údaje, aby sa získali originálne odpovede. ML je založený skôr na trénovaní ako na programovaní. Tento prístup prináša omnoho lepší výkon a dokáže sa rýchlo prispôsobiť novým úlohám s minimálnou pomocou človeka. ML dostane mnohé príklady relevantné pre úlohu a v týchto príkladoch nachádza štatistickú štruktúru, ktorá nakoniec umožňuje systému prísť s pravidlami na automatizáciu úlohy. Najviac sa začal rozvíjať v 90. rokoch a stal sa najobľúbenejšou podčasťou umelej inteligencie. ML pracuje s veľkými a zložitými súbormi údajov ako napríklad súbor miliónov obrázkov, z ktorého každý obrázok pozostáva z tisícok pixelov (Chollet, 2021). V súčasnosti sa ML používa hlavne v sociálnych sieťach, webových vyhľadávačoch, eshopoch a je prítomný v mobilných zariadeniach či fotoaparátoch.

Hlboké učenie (angl. *Deep learning*) skratka DL, je známe aj pod pojmami *Deep structured learning*, *Hierarchical learning* alebo *Deep machine learning*. Hlboké učenie je špecifická podoblasť strojového učenia. Ide o štúdium neurónových sietí a strojového učenia, ktoré obsahuje viac ako jednu skrytú vrstvu ako je spomínané v článku od Ongsulee (2017). DL je založený na učení viacerých úrovní funkcií. Jeho výhodou je nahradenie ručne naprogramovaných funkcií efektívnymi algoritmami učenia. HĺBku modelu predstavuje počet vrstiev, z ktorých model pozostáva. Hlbokú sieť si môžeme predstaviť ako viacstupňovú informačno-destilačnú operáciu, kde informácie prechádzajú postupnými filtrami a vychádzajú čoraz čistejšie. Špecifikácia

toho, čo daná vrstva robí so svojimi vstupnými údajmi, je uložená v parametroch váh, ktoré sú v podstate zhlukom čísel. Jeho úlohou je nájsť množinu hodnôt pre váhy všetkých vrstiev v sieti tak, aby sieť správne mapovala vzorové vstupy na ich priradené ciele. Sieť môže však obsahovať milióny parametrov. Dnešné hlboké učenie často zahŕňa desiatky dokonca stovky po sebe idúcich vrstiev reprezentácií. Všetky sa učia automaticky z tréningových údajov. V porovnaní so strojovým učením je to omnoho viac (Chollet, 2021). Hlboké učenie prináša veľké pokroky pri riešení problémov, ktoré doteraz odolávali iným metódam AI. DL je perfektný pri objavovaní zložitých štruktúr vo viacrozmerných údajoch. Pri rýchlosti rozpoznania obrázkov či reči DL porazil štandardné techniky ML. Aj keď sa hlboké učenie rozmohlo len v posledných desiatich rokoch, stihlo už napredovať v oblastiach ako klasifikácia snímok, rozpoznávanie reči, prepis rukopisu, preklad, cielenie reklám na takmer ľudskej úrovni. DL sa bude aj v budúcnosti rozvíjať, keďže vyžaduje malý zásah od ľudí a vyvíjajú sa nové vzdelávacie algoritmy pre neurónové siete (LeCun et al., 2015).

1.1.2 Typy učenia

Už v minulej podkapitole pri vysvetľovaní pojmov boli použité výrazy ako učenie bez učiteľa a s učiteľom. V tejto podkapitole budú bližšie priblížené.

Kontrolované učenie (angl. *Supervised learning*) inak nazývané aj učenie s učiteľom alebo učenie pod dohľadom. Tento druh strojového učenia je založený na trénovaní vzorky údajov zo vstupného datasetu s už priradenou správnou klasifikáciou, čiže vstupné dáta sú označené. Vstupné súbory údajov sú navrhnuté tak, aby natrénované algoritmy mohli s veľkou presnosťou predpovedať výsledky aj novým vstupným údajom. Keď je modelu priradený nový súbor príkladov, algoritmus učenia pod dohľadom analyzuje tréningové údaje a hľadá správny výsledok z označených údajov. Model učenia pod dohľadom využíva priamu spätnú väzbu, aby skontroloval, či predpovedá správny výstup alebo nie. Pomocou označených vstupov a výstupov môže model merať svoju presnosť a učiť sa v priebehu času, a taktiež zbierať údaje z predchádzajúcich skúseností. Pomocou predošlých skúseností vieme takisto opti-

malizovať kritériá výkonu. Na kontrolované učenie je potrebné veľké množstvo času (LeCun et al., 2015). Kontrolované učenie má dva typy algoritmov:

- **Klasifikácia** (angl. *Classification*) ide o metódu, kde výsledkom je priradenie vzorky do špecifických kategórií. Pojem klasifikácia môžeme vysvetliť ako zoskupenie výstupu v rámci triedy. Ak výsledkom klasifikácie sú presne dve triedy ide o binárnu klasifikáciu. Príkladom môže byť, či niečo je choroba alebo to nie je choroba. Ak ide o výber z viacerých tried nazývame ju klasifikácia viacerých tried. Výhodou je, že algoritmus klasifikácie vieme regulovať, aby sme predišli nadmernému preučeniu. Klasifikovanie veľkého množstva dát môže byť náročné (Donalek, 2011).

- **Regresia** (angl. *Regression*) táto metóda predpovedá jednu výstupnú hodnotu pomocou tréningových údajov. Algoritmus používa metódy na pochopenie vzťahu medzi závislými a nezávislými premennými. Regresné modely sú určené na predpovedanie číselných hodnôt. Príkladom môže byť predikovanie ceny nehnuteľnosti na základe lokality, veľkosti, stavu. Regresné algoritmy, ktoré poznáme sú napríklad lineárna regresia, logická regresia, polynomická regresia, či samotné neurónové siete.

Nekontrolované učenie (angl. *Unsupervised learning*) inak nazývané aj učenie bez učiteľa či bez dohľadu. Ide o učenie pomocou informácií, ktoré nie sú klasifikované ani označené. Úlohou algoritmu je zoskupiť nezoradené informácie podľa podobností, vzorov a rozdielov bez akéhokoľvek predchádzajúceho trénovania údajov. Stroj musí sám nájsť skrytú štruktúru v neoznačených údajoch. Algoritmus tak môže nájsť aj informácie a vzorce, ktoré doteraz neboli nájdené. Učenie bez dohľadu umožňuje vykonávať zložitejšie úlohy ako učenie s dohľadom. Výhodou je, že je dostupnejšie, keďže je jednoduchšie získať dataset neoznačených obrázkov ako dataset označených obrázkov, ktoré by museli byť označené ručne. Nevýhodou je, že poskytuje menej presné výsledky v porovnaní s kontrolovaným učením. Učenie bez dozoru sa viac približuje skutočnej umelej inteligencii, keďže sa učí podobne, ako keď sa

dieťa učí na základe svojich skúseností veciam každodenného sveta (Ayodele, 2010). Nekontrolované učenie je rozdelené do týchto algoritmov:

- **Zhlukovanie** (angl. *Clustering*) je to metóda hľadania štruktúry alebo vzoru v nekategorizovaných údajoch. Technika zoskupí neoznačené údaje na základe ich podobností a rozdielov. Môže ísť o veľkosť, farbu či tvar. Algoritmus spracuje vstupné údaje a nájde prirodzené klastre, ak existujú. Vieme určiť koľko klastrov – zhlukov by mal algoritmus identifikovať. Existuje niekoľko typov zhlukovacích algoritmov. Príkladom môže byť zoskupenie zákazníkov podľa ich nákupného správania.

- **Asociácia** (angl. *Association*) je to druh nekontrolovaného učenia, ktorý dokáže nájsť vzťah jednej položky z údajov k inej položke z údajov. Tieto vzťahy vieme ďalej použiť. Pravidlá asociácie umožňujú vytvoriť asociácie medzi dátovými objektmi vo veľkých databázach. Príkladom môže byť, že človek, ktorý si kúpil nový byt bude v blízkej dobe kupovať aj zariadenie do bytu.

Spojením oboch predošlých typov učenia je *Semi-supervised learning* v preklade učenie s polovičným dohľadom. Je to zlatý stred medzi kontrolovaným a nekontrolovaným učením. Používa sa tréningový súbor s označenými aj neoznačenými vstupmi. Je užitočný pri veľkom objeme dát. Funguje na princípe, že najprv sa model učí na označených dátach, ak je dosiahnutý dobrý výkon modelu, použije sa na zvyšných neoznačených dátach a označí ich s príslušnými pravdepodobnosťami. Nakoniec nasleduje trénovanie na celej skupine dát spolu.

1.1.3 Spracovanie obrázkov

Klasifikácia obrázkov (angl. *image classification*) je technika strojového učenia, ktorá má za úlohu určiť, aké objekty sú na obrázku alebo vo videu. Je označovaná aj ako rozpoznávanie obrázkov. Je to široká oblasť spracovania obrazu. Ako je spomínané v článku od Gavali and Banu (2019) ľudský mozog dokáže jednoducho

klasifikovať obrázky, ale pre počítač to nie je jednoduché, ak obrázok obsahuje ďalšie rušivé elementy. Ide o trénovanie modelu s cieľom určiť, či sú dané triedy prítomné. Obrázok analyzuje a zhodnotí záver, a zaradí, čo je na obrázku do triedy. Klasifikácia vie určiť len úroveň áno/nie pri rozhodovaní, či obrázok obsahuje alebo neobsahuje daný objekt. Ku klasifikácii prislúcha aj osobitná úloha lokalizácia, ktorá ma za úlohu určiť polohu klasifikovaného objektu na obrázku. Ak je na obrázku len jeden konkrétny objekt, ako napríklad pes ide o *single-label classification*, čiže klasifikáciu s jedným označením. Ak sa však na obrázku nachádza aj pes aj mačka ide už o *multi-label classification*, teda klasifikáciu s viacerými označeniami.

- **Klasifikácia s jedným štítkom** alebo anotáciou (angl. *single-label classification*) je jednou z najčastejších klasifikácií. Na obrázku je prítomný jeden objekt, preto výsledkom modelu je jedna hodnota alebo predikcia. Štandardné datasety, ktoré sú príkladom na túto klasifikáciu sú MNIST, SVHN a ImageNet.

- **Klasifikácia s viacerými štítkami** alebo anotáciami (angl. *multi-label classification*) je klasifikácia, ak obrázok obsahuje naraz viacero objektov. Je omnoho zložitejšia. Aj omnoho častejšia, keďže zvyčajne sa na obrázku nachádza viac ako jeden objekt. Pri tejto klasifikácii si model zvykne všímať aj vzťahy medzi objektami na obrázku. Napríklad, že oblak sa vyskytuje stále na oblohe (Wang et al., 2016). Zobrazenie tejto klasifikácie môžte vidieť na Obrázku 1 – 2 a).

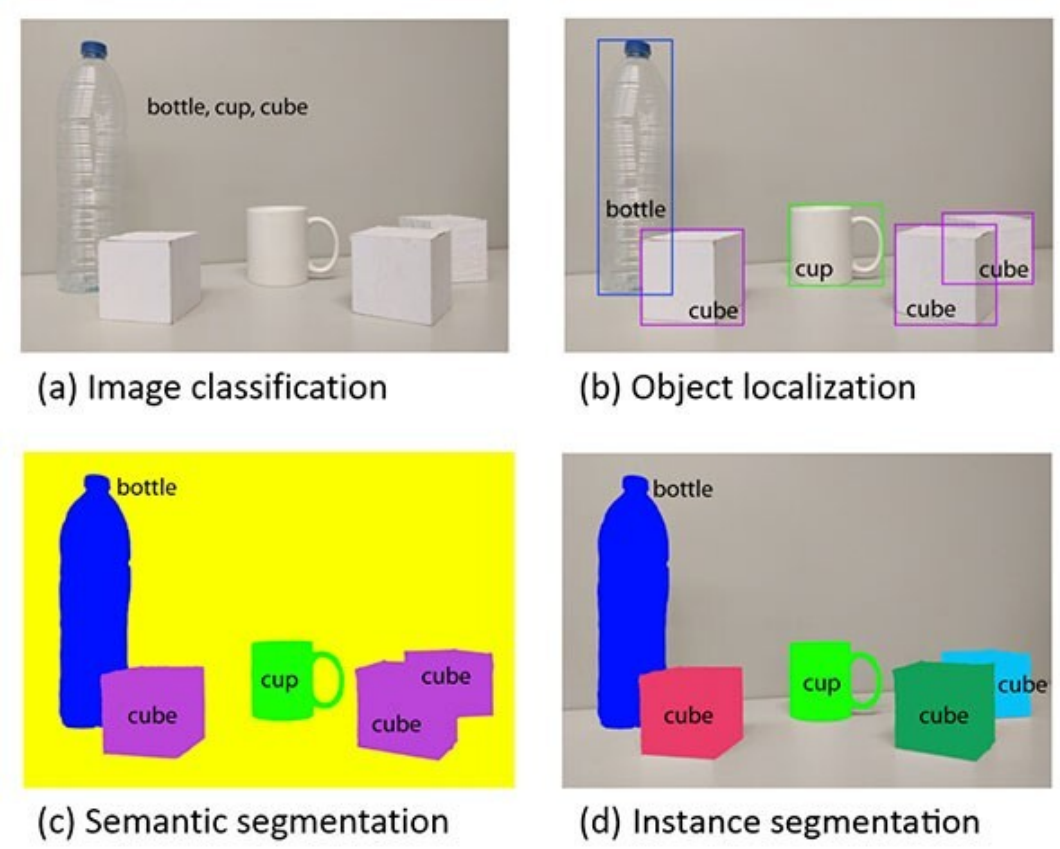
Detekcia objektov (angl. *object detection*) je technika, ktorá kombinuje klasifikáciu a lokalizáciu daného objektu. Určí, aké objekty sa nachádzajú na obrázku a určí ich polohu. Na určenie polohy používa takzvané *bounding boxes* spomenuté v článku od Zhao et al. (2019). Sú to rámčeky, ktoré označujú každý objekt na obrázku. Tento rámček je však obdĺžnikového alebo štvorcového tvaru, čiže nič nevypovedá o reálnom tvare klasifikovaného objektu. Existuje veľké množstvo využitia detekcie objektov v reálnom čase. Ide napríklad o detekciu vozidiel na cestách. Zobrazenie detekcie objektov môžte vidieť na Obrázku 1 – 2 b). Keďže v tejto práci sa

budeme venovať práve detekcii objektov viac o nej sa dozviete v ďalšej podkapitole.

Segmentácia obrázkov (angl. *image segmentation*) je technika podrobnejšia ako detekcia objektov. Ide o hlbšiu analýzu objektu na obrázku. Pri segmentácii sa obrázok rozdelí na rôzne časti nazývané taktiež aj segmenty, čo pomáha pri znižovaní zložitosti obrazu s cieľom zjednodušiť ďalšie spracovanie alebo analýzu obrazu. Niektoré segmenty neobsahujú dôležitú informáciu, iné zase áno. Obrázok sa skladá zo sady pixelov. Segmentácia zoskupuje pixely, ktoré majú podobné atribúty. Segmentácia obrázka vyznačí pixely, ktoré patria ku danému objektu. V jednoduchosti je to priraďovanie štítkov ku pixelom. Táto technika prináša oveľa väčšie pochopenie objektu na obrázku. V počítačovom videní väčšina modelov segmentácie obrazu pozostáva zo zoskupenia kóder-dekodér. Kóder zakóduje latentnú priestorovú reprezentáciu vstupu, ktorú dekodér dekóduje, aby vytvoril segmentové mapy alebo inými slovami mapy znázorňujúce umiestnenie každého objektu na obrázku. Príkladom, prečo je segmentácia dôležitá je napríklad detekovanie buniek rakoviny. Ak by sme použili len detekciu objektov, bunky budú síce nájdené ale nezískame žiadne informácie z ich tvaru. Práve tvar rakovinovej bunky je dôležitý pri určovaní závažnosti rakoviny. V súčastnosti je technika segmentácie používaná aj v systémoch riadenia dopravy, lokalizácii objektov na satelitných snímkach a pri autonómnych autách (Minaee et al., 2021). Aj segmentácia má viaceré typy:

- **Sémantická segmentácia** (angl. *Semantic segmentation*) klasifikuje každý z pixelov v obrázku do sémantických tried. Pixely patriace do určitej triedy sú jednoducho zaradené do tejto triedy bez toho, aby sa brali do úvahy ďalšie informácie alebo kontext. Príkladom je, že viacero ľudí na obrázku je označených ako dav dokopy. Podobný prípad je znázornený na Obrázku 1 – 2 c).
- **Segmentácia inštancií** (angl. *Instance segmentation*) je opakom prvého prípadu. Táto segmentácia klasifikuje pixely do kategórií na základe inštancií a nie tried. Vie oddeliť prekrývajúce sa alebo veľmi podobné oblasti objektov na základe ich hraníc. Príkladom je, že na obrázku s viacerými ľudmi označí kaž-

dého človeka samostatne a oddelí ho od okolitých objektov. Podobný prípad je znázornený na Obrázku 1 – 2 d).



Obrázok 1 – 2 Porovnanie klasifikácie obrázka – a), detekcie objektov – b) a dvoch rôznych druhov segmentácie, sémantickej segmentácie – c) a segmentácie inštancií – d) (Garcia-Garcia et al., 2017).

1.2 Detekcia objektov pomocou YOLO

„*You Only Look Once*“ (skrátene YOLO) je nový prístup k detekcii objektov v reálnom čase. Preklad tohto pojmu znamená, že na jeden pohľad je možné predpovedať, aké objekty sa nachádzajú na obrázku a kde sa nachádzajú (Du, 2018). Ľudia sa pozrú na obrázok a okamžite vedia, aké objekty sú na obrázku, kde sa nachádzajú a ako na seba vzájomne pôsobia. YOLO pracuje rovnako ako človek a

jeho oko. Je to algoritmus, ktorý funguje na princípe konvolučnej neurónovej siete. Spracovanie obrázkov s YOLO je jednoduché a priamočiare. Jedna konvolučná sieť súčasne predpovedá viacero ohraničujúcich políčok– anotácií a pravdepodobnostné triedy pre tieto polia. YOLO trénuje na celých obrázkoch a priamo optimalizuje výkon detekcie. Tento jednotný model má niekoľko výhod oproti tradičným metódam detekcie objektov (Redmon et al., 2016).

Konvolučná neurónová sieť (angl. *Convolutional neural network*, CNN) je dobre známa architektúra hlbokého učenia inšpirovaná prirodzeným vizuálnym vnímacím mechanizmom živých tvorov. V roku 1982 Kunihiko Fukushima inšpirovaný objavom, že bunky zrakovej kôry zvierat reagujú na detekciu svetla v receptívnych poliach navrhol „*Neocognitron*“, ktorý možno považovať za predchodcu CNN (Gu et al., 2018). CNN sú jednou z najpopulárnejších kategórií neurónových sietí, najmä pre veľkorozmerné dáta ako napríklad obrázky a videá. Fungujú spôsobom, ktorý je veľmi podobný štandardným neurónovým sieťam. CNN je užitočnou triedou modelov pre *supervised learning* – učenie s učiteľom aj pre *unsupervised learning* – učenie bez učiteľa (Khan et al., 2018). CNN sa skladá z viacerých vrstiev:

Vstupná vrstva , jej hlavnou úlohou je inicializovať vstupné obrazové dáta, aby boli všetky rozmery vstupných údajov vycentrované na nulu. Ďalej má za úlohu znormalizovať rozsah všetkých vstupných údajov na intervale od 0 po 1 (Du, 2018).

Konvolučná vrstva je najdôležitejšou súčasťou CNN, teda jadrom. Je reprezentovaná niekoľko za sebou idúcich konvolučných vrstiev s množstvom menších filtrov, ktoré sa postupne aplikujú na celú oblasť obrázka. Každý filter ignoruje iné vlastnosti a inú vlastnosť na vstupe detekuje. Koľko filtrov máme, toľko je aj výstupných príznakových máp (Khan et al., 2018).

Aktivačná vrstva inak nazývaná aj ako nelineárna aktivačná funkcia. Má za úlohu stanoviť hodnoty výstupov každého z neurónov prostredníctvom ich vnútorného potenciálu. Poznáme mnoho druhov aktivačných funkcií ako napríklad

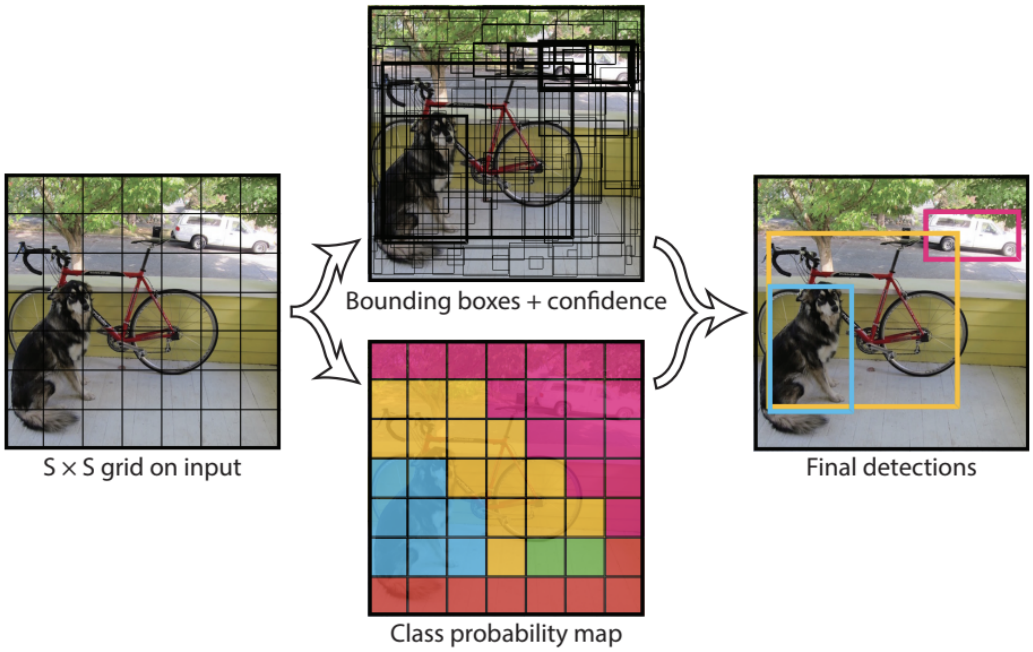
Sigmoidová funkcia, Tanh alebo ReLU. V poslednej dobe je najpopulárnejší Leaky ReLU, pretože konverguje rýchlejšie ako ostatné a je jednoduchší a efektívnejší (Du, 2018).

Vzorkovacia vrstva (angl. *Pooling layer*) sa používa na zmenšenie rozmeru máp prvkov z konvolučnej vrstvy. Typickými spôsobmi sú *max pooling* a *average pooling*. Pri *max pooling* vyberáme maximálnu hodnotu z danej oblasti vstupu. Pri *average pooling* inak aj priemerovanie hodnôt, počítame pre každú oblasť priemer prvkov (Nguyen et al., 2019).

Plne prepojená vrstva je často poslednou vrstvou na konci CNN. Zabezpečuje prenos výstupných dát, ako aj zjednodušenie a zrýchlenie výpočtu dát (Du, 2018).

YOLO umožňuje komplexný tréning a rýchlosť v reálnom čase pri zachovaní vysokej priemernej presnosti. Systém rozdelí vstupný obrázok na mriežku S × S. Každá z buniek mriežky má rozdielne detekčné úlohy. Keď stred objektu patrí do bunky mriežky, táto bunka mriežky je zodpovedná za detekciu tohto objektu. Každá bunka mriežky predpovedá ohraničujúce rámčeky a pravdepodobnosť pre tieto boxy. Metrika *confidence* odráža ako veľmi je model presvedčený, že box obsahuje predmet. Ak neexistuje v danej bunke žiadny objekt, pravdepodobnosť by malo byť nula. V opačnom prípade chceme, aby sa pravdepodobnosť správnosti rovnala hodnote medzi IoU predpokladaného rámčeka a realitou. Každý anotovaný rámček pozostáva z piatich údajov predpovedí: x, y, šírka, výška, a *confidence*. Súradnice (x, y) predstavujú stred rámčeka vzhľadom na hranice bunky mriežky. Šírka a výška sú predpovedané vzhľadom na celý obrázok. Posledným je už spomínaná pravdepodobnosť správnosti (Redmon et al., 2016). Tento postup je znázornený na Obrázku 1 – 3.

YOLO prešlo od svojho vzniku viacerými vylepšeniami. V mojej práci som použila aktuálnu verziu číslo 5 – YOLOv5, ktorá bude popísaná v nasledujúcom oddiele.



Obrázok 1 – 3 Postup ako pracuje detekčná sieť YOLO. Na prvej časti je ukázané rozdelenie obrázku do SxS mriežky. Druhý krok hore opisuje orámovanie objektov rámčekmi a zvolenie pravdepodobnosti. Druhý krok dole znázorňuje mapu s akou pravdepodobnosťou bunka patrí do danej triedy. Výsledkom je finálna detekcia (Redmon et al., 2016).

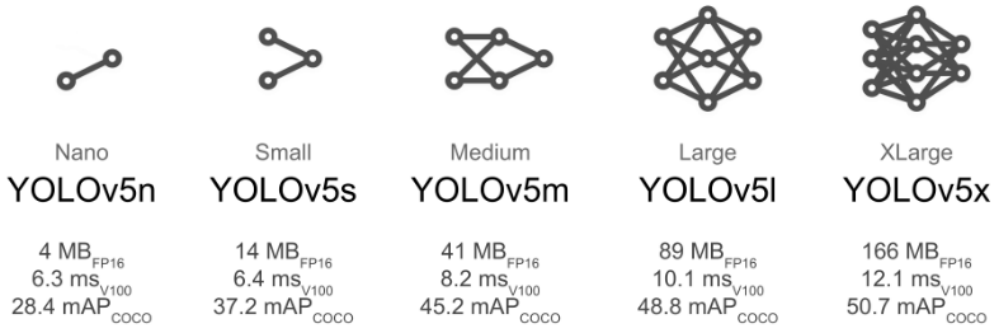
1.2.1 YOLOv5

YOLOv5 je piata generácia siete YOLO. Bola publikovaná na GitHub v máji roku 2020 autorom Glennom Jocherom Jocher (2020). Aj táto verzia funguje pomocou PyTorch5. YOLOv5 je stále v aktívnom vývoji, najnovšou verziou je verzia 5.0.

Presnosť detekcie tohto sieťového modelu je vysoká a taktiež aj rýchlosť odvodzovania, pričom najrýchlejšia rýchlosť detekcie je až 140 snímok za sekundu. Na druhej strane, veľkosť súboru váh modelu siete YOLOv5 je malá, je takmer o 90 percent menšia ako v YOLOv4, čo naznačuje, že model YOLOv5 je vhodný na nasadenie do vstavaných zariadení na implementáciu detekcie v reálnom čase. Preto sú výhody siete YOLOv5 jej vysoká presnosť detekcie, ľahké charakteristiky a zároveň vysoká rýchlosť detekcie. Nové funkcie a vylepšenia v YOLOv5 sú zamerané

hlavne na začlenenie najmodernejších sietí pre hlboké učenie, napríklad aktivačné funkcie a augmentácia dát (Yap et al., 2021). Jedným z najvýznamnejších aspektov rozšírenia dát je *mosaic loader*, v ktorom sa štyri obrázky menia a kombinujú sa do nového obrázku. To umožňuje detekciu objektov mimo ich normálneho kontextu a pri menších veľkostiach.

Architektúra YOLOv5 obsahuje päť predtrénovaných modelov, konkrétne pomenované ako YOLOv5n – Nano, YOLOv5s– Small, YOLOv5m – Medium, YOLOv5l– Large a YOLOv5x –XLarge. Tieto modely môžte vidieť na Obrázku 1 – 4.



Obrázok 1 – 4 Porovnanie piatich modelov architektúry YOLOv5.

Zdroj: `https://github.com/ultralytics/yolov5/wiki/Train-Custom-Data`

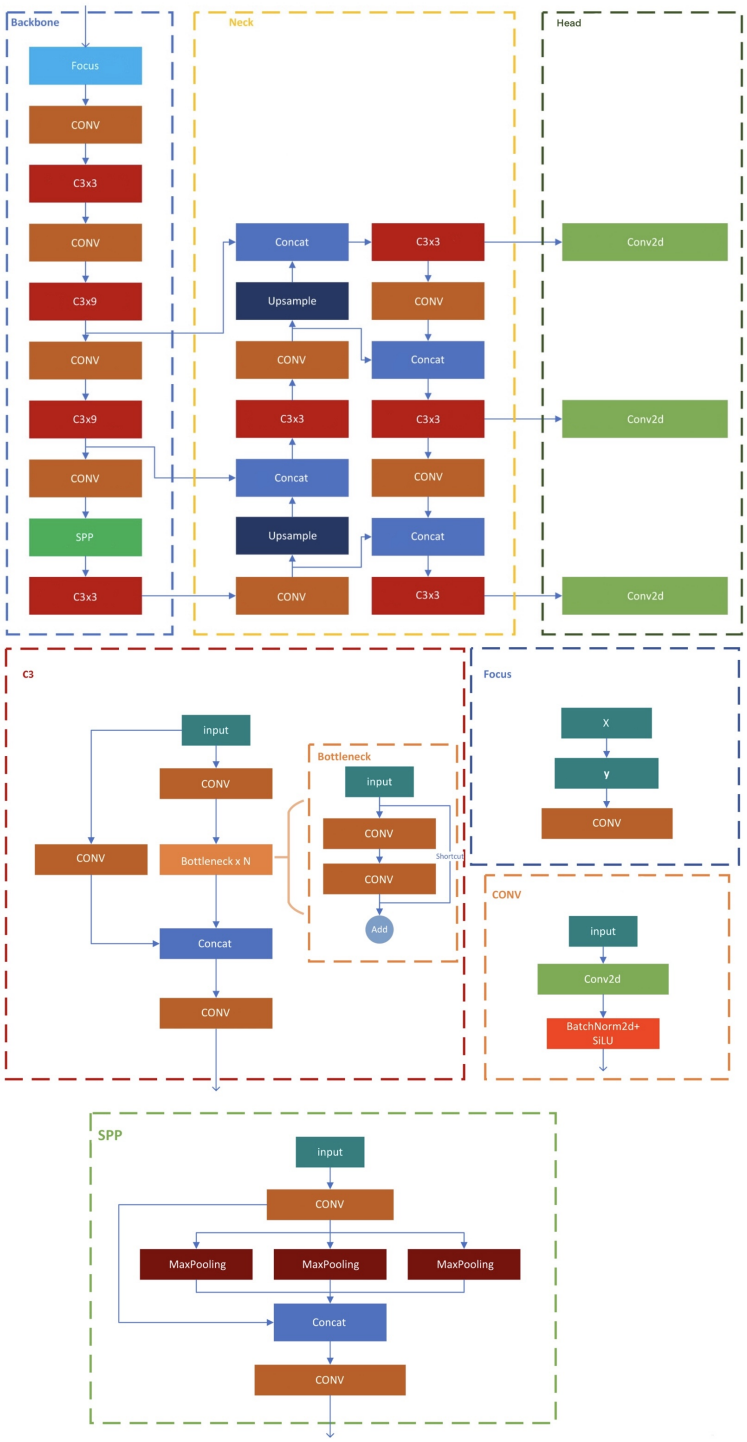
Štruktúra YOLOv5 pozostáva z troch hlavných častí. Tými sú *Backbone*, *Neck* a *Head*. Obrázok 1 – 5 zobrazuje celkovú architektúru siete YOLOv5.

Chrbtica (angl. *Backbone*) je to konvolučná neurónová sieť, ktorá agreguje rôzne jemnozrnné obrázky a vytvára obrazové prvky. Slúži najmä na extrakciu dôležitých funkcií z daného vstupného obrázku. V YOLOv5 sa CSP – *Cross Stage Partial Networks* Wang et al. (2016) používajú ako chrbtica na extrahovanie bohatých informačných funkcií zo vstupného obrazu.

Krk (angl. *Neck*) je séria vrstiev agregácie prvkov zmiešaných a kombinovaných obrazových prvkov, ktorá sa používa hlavne na generovanie FPN – *Feature Pyramid Network* Wang et al. (2016) (slov. siete pyramíd prvkov) a potom sa

výstupná mapa prvkov prenáša do siete detekcie (predikčná sieť). Pyramídy funkcií pomáhajú modelom dobre zovšeobecniť škálovanie objektov. Pomáha identifikovať ten istý objekt s rôznymi veľkosťami a mierkami.

Hlava (angl. *Head*) preberá prvky z krku a podniká kroky na predikciu rámčeka a tried. Hlava modelu sa používa hlavne na vykonanie konečnej časti detekcie. Aplikuje rámčeky na prvky a generuje výsledné výstupné vektory s pravdepodobnostnými triedami, skóre a hraničnými rámčekmi.



Obrázok 1 – 5 Architektúra YOLOv5. Sieť YOLO pozostáva z troch hlavných častí: chrbtica, krk a hlava (Maslej-Krešňáková et al., 2021).

2 Crowdsourcing a Zooniverse

2.1 Crowdsourcing

Táto kapitola sa bude venovať crowdsourcingu. Čo vlastne znamená pojem crowdsourcing? V preklade do slovenčiny znamená „*crowd*“ – skupina, dav či množstvo ľudí a „*source*“ je zdroj či pôvod. Spolu by sme tento termín mohli preložiť ako zdroj získavania informácií od skupiny ľudí.

V článku z magazínu Wired od autora Howe et al. (2006) je crowdsourcing definovaný ako obchodný model webového charakteru, ktorý využíva kreatívne riešenia skupiny jednotlivcov, ktorí sú dobrovoľne zapojení do činnosti. Zjednodušene povedané crowdsourcing predstavuje konanie firmy alebo inštitúcie, ktorá zadáva úlohu, ktorá bola v minulosti riešená zamestnancami alebo outsourcingom, širokej skupine dobrovoľníkov vo forme otvorenej výzvy pre každého. Vykonávaná môže byť formou partnerskej spolupráce medzi viacerými jednotlivcami alebo jednotlivo.

Ako je možné, že pomocou crowdsourcingu dokáže riešiť široká verejnosť aj úlohy, ktoré sú komplexné a zložité dokonca pre odborníkov? Ide o pojem *Crowd Wisdom* v preklade múdrosť skupiny. Úspech riešenia skupiny záleží od množstva ľudí v skupine. Často je skupina ľudí múdrejšia ako najmúdrejší člen v nej. Múdrosť skupiny nie je definovaná spriemerovaním riešení ale ich agregáciou. Internet a vymoženosti dnešnej doby nám pomáhajú so spájaním nezávislých myšlienok od množstva ľudí bez prílišnej komunikácie. Tento pojem je podrobnejšie rozobratý v článku (Brabham, 2008).

Podobným pojmom, ktorý podrobnejšie vysvetľuje princíp crowdsourcingu z článku Brabham (2008) je kolektívna inteligencia. Ide o formu univerzálne distribuovanej inteligencie, ktorá sa mení v čase, je neustále vylepšovaná a jej výsledkom je efektívna mobilizácia zručností. Každý vie niečo. Nikto nevie všetko. Preto je súbor poznatkov od všetkých lepší ako poznatky do jednotlivca.

Pri crowdsourcingu väčšinou dobrovoľníci za pomoc odmenu nedostávajú. Ich motiváciou je získanie nových poznatkov a zručností z rôznych oblastí a ich následné

využitie v živote či ako pomoc pri uchádzaní sa o zamestnanie. Ide o ľudí s vášňou o riešenie problémov a skúmanie rôznych problematík. Pre crowdsourcing je dôležité aby ľudia, ktorí sa ho zúčastňujú boli čo najrôznorodejší. Mali rôzne pohlavie, vek, národnosť, vzdelanie či náboženstvo. Každá osoba je unikátna, tým pádom je unikátna aj každá odpoveď. Táto podmienka zaručuje, že získame čo najrôznorodejšie riešenia problému, medzi ktorými sa bude nachádzať aj to najefektívnejšie.

Crowdsourcing je teda celkom novodobý koncept na riešenie problémov, uskutočňovaný cez web, pomocou ktorého sú spájané vedomosti a talent kolektívu a je znížený čas a financie potrebné na riešenie klasickým spôsobom.

2.2 Zooniverse

Zooniverse [1] je jedna z najväčších a najznámejších crowdsourcingových platform. Výskum na tejto platforme uskutočňujú dobrovoľníci z každého kútu sveta, ktorí takto pomáhajú riešiť problémy výskumníkom a vedcom. Tento web umožňuje výskum, ktorý by bol inak nemožný či náramne náročný. Výsledkom sú nové objavy či poznatky prospešné pre ďalšie skúmania či písanie publikácií a článkov.

Tak ako aj dobrovoľníkom aj výskumníkom sa môže na tejto platforme stať každý. Nie je potrebné žiadne špeciálne vzdelanie ani odborné znalosti. Stačí mať len úlohu, s ktorou potrebujete pomoc od dobrovoľníkov.

Na platforme sa nachádza veľký počet projektov. Participanti môžu pomáhať na projektoch z oblastí ako je vesmír, fyzika, história, klíma, príroda, medicína či jazyky. Projekty sú zväčša zložené z jednoduchých otázok, či označovaní javov na obrázkoch.

Platforma pomáha výskumníkom získavať informácie rýchlejšie, ktoré môžu ďalej analyzovať a robiť pokrok vo vede a výskume. Projekty sú založené na už vyššie spomenutej múdrosti davu, ktorá zaručuje spoľahlivé údaje. Keďže každý problém, vyrieši viac ľudí vieme určiť aj pravdepodobnosť chyby.

[1] `https://www.zooniverse.org/about`

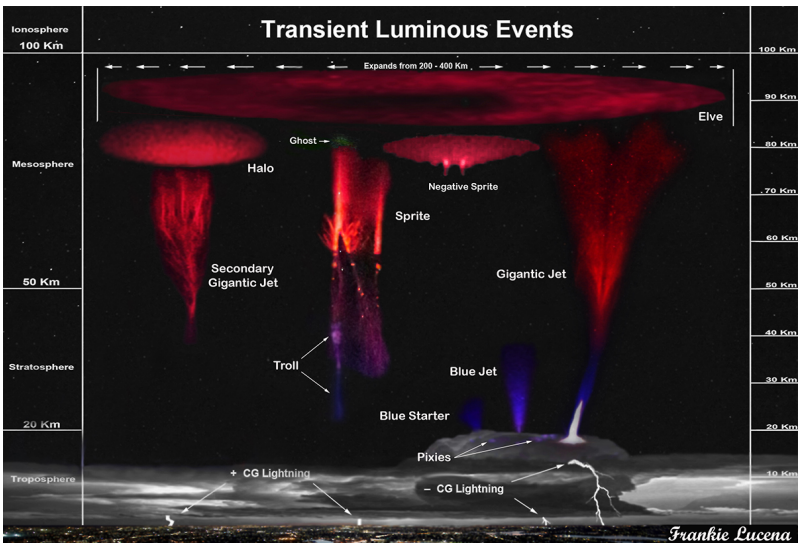
Zooniverse mení úsilie dobrovoľníkov na merateľné výsledky. Výsledky projektov prinášajú veľké množstvo publikovaných vedeckých prác a súborov analyzovaných dát. Nie raz sa podarilo účastníkom vytvoriť vedecky významné objavy. Ku skvelým výsledkom pomáha aj možnosť diskusie medzi účastníkmi a výskumníkmi na platforme. Konverzácia sa odohráva na diskusných fórach pri jednotlivých projektoch. Túto platformu sme sa rozhodli využiť aj pri získavaní anotácií v našom datasete.

3 Úvod do TLE (Transient Luminous Events)

3.1 TLE javy a systém AMOS

Transient luminous events (TLE) sú javy, ktoré boli objavené v Zemskej atmosfére len nedávno, v 90. rokoch 20. storočia. V článku od Pasko (2010) sú definované ako rozsiahle optické udalosti vyskytujúce sa v stratosférických, mezosférických a nižších ionosférických nadmorských výškach, ktoré priamo súvisia s elektrickou aktivitou v búrkach. Prvý objav tohto javu sa uskutočnil čisto náhodne počas testu televíznej kamery pri slabom osvetlení v roku 1989 (Franz et al., 1990). Odvtedy bolo zdokumentovaných viacero druhov TLE eventov ako napríklad *Elves*, *Sprites*, *Gigantic jets*, *Blue jets* a ďalšie. *Elves* či v preklade elfovia sú extrémne rýchlo sa pohybujúce svetelné prstence alebo blesky vyvolané zábleskmi expandujúce v ionosfére. *Sprites* v preklade škriatkovia sú koreňové komplexné vysoké štruktúry začínajúce v mezosférach a končiace v ionosférach a pohybujú sa rýchlo nadol rýchlosťou až 10 000 km/s. *Gigantic* a *Blue jets* – fontány sú dlhé úzke kužele bleskov so základňou v hornej časti búrkových oblakov smerujúce nahor ako fontána. Farba a intenzita ich svetelného vzhľadu závisí od konkrétnych interakcií nabitých častíc a elektrických polí v atmosfére (Amrich et al., 2021). Ako tieto druhy TLE udalosti vyzerajú môžete vidieť na Obrázku 3 – 1.

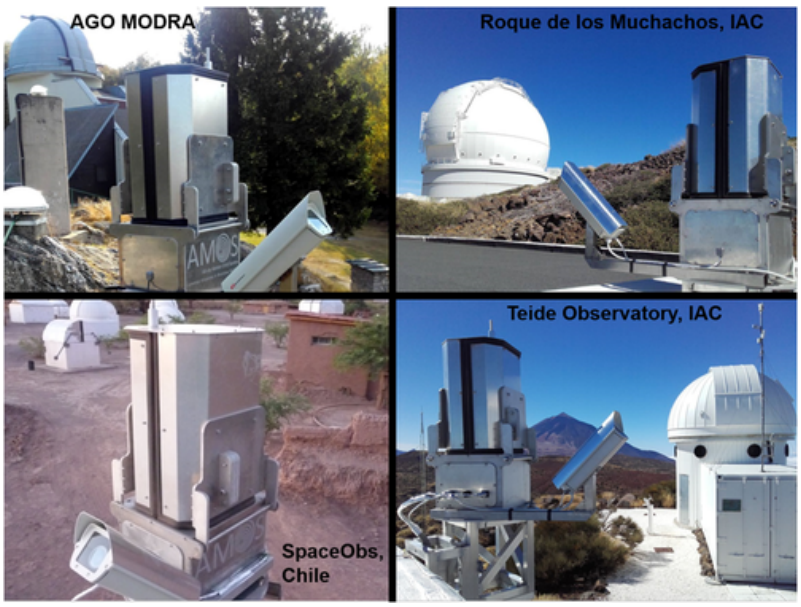
Jedny z veľmi účinných zariadení, ktoré pozorujú TLE javy na Slovensku sú kamery AMOS. AMOS je skratka pre *All-Sky Meteor Orbit System*. V preklade ide o celooblohový systém - kameru na detekciu svetelných javov na nočnej oblohe predovšetkým na snímanie meteorov. Tento systém sa skladá zo šošovky - rybie oko, zosilňovača obrazu a digitálnej kamery. Optický systém kamery je chránený vonkajším a vnútorným krytom a súpravou detektorov teploty, vlhkosti vzduchu. Tento systém je predovšetkým určený na pozorovanie meteorov, ale mohol by byť použitý aj na pozorovania z oblasti meteorológie, letectva či satelitov. Prevádzka kamier je automatizovaná a diaľkovo ovládaná. Snímková frekvencia videa je 20 za sekundu.



Obrázok 3 – 1 Druhy TLE javov.

Zdroj: `https://spaceweatherarchive.com/2020/03/07/get-ready-for-sprite-season/`

Prvý prototyp tohto zariadenia funguje na observatóriu AGO Modra od roku 2007, odvtedy fungujú ďalšie kamery na Slovensku aj zahraničí. V súčasnosti štyri AMOS kamery tvoria Slovenskú videometeorickú sieť (SVMN) na miestach AGO Modra (AGO), Arborétum Tesárske Mlyňany (ARBO), Kysucké Nové Hvezdáreň Mesto (KNM) a Važec. Priama vzdialenosť medzi jednotlivými stanicami je od 80 do 150 km a sú prevádzkované na diaľku. SVMN je riadená tímom astronómov z Fakulty Matematiky, Fyziky a Informatiky na Univerzite Komenského v Bratislave. Ako AMOS kamery vyzerajú u nás a v zahraničí môžete vidieť na Obrázku 3 – 2. Každá kamera AMOS zaznamená približne 10 000 – 20 000 meteorov ročne a približne 50 TLE javov ako škriatkovia či elfovia (Tóth et al., 2015). Prístup k dátam zo systému AMOS sme mali na základe spolupráce s Doc. RNDr. Jurajom Tóthom, PhD., ktorý je vedúcim programu AMOS.



Obrázok 3 – 2 AMOS kamery umiestnené na Slovensku v Modre, na Kanárskych ostrovoch a v Čile.

Zdroj: `https://fmph.uniba.sk/en/microsites/daa/division-of-astronomy-and-astrophysics/research/meteors/amos/`

3.2 Analýza súčasného stavu detekcie TLE javov

Táto kapitola analyzuje súčasný stav v danej problematike automatickej detekcie TLE javov.

Konzultant Samuel Amrich riešil v svojej práci Amrich et al. (2021) návrh a konštrukciu hardvéru a softvéru pre autonómne pozorovania TLE javov. Keďže TLE javy nemajú jednotný spôsob automatického pozorovania a následného spracovania, preto vytvoril projekt, ktorý sa zameriava na návrh a konštrukciu hardvéru na pozorovanie TLE eventov. Prístroj sa skladá z CMOS kamery so širokouhlou šošovkou, objektívu a SBC. Všetko sa nachádza vnútri vodotesného obalu, z dôvodu bezchybného fungovania vo vonkajšom prostredí aj pri nepriaznivom počasí. Ku hardvérovej časti vytvoril aj softvér na ovládanie hardvéru. Softvér sa sám rozhodne, kedy má snímať a kde je najväčšia šanca zachytiť TLE jav. Ďalej je tento softvér schopný spracovať a klasifikovať zachytené TLE javy pomocou neurónových sietí. Softvér

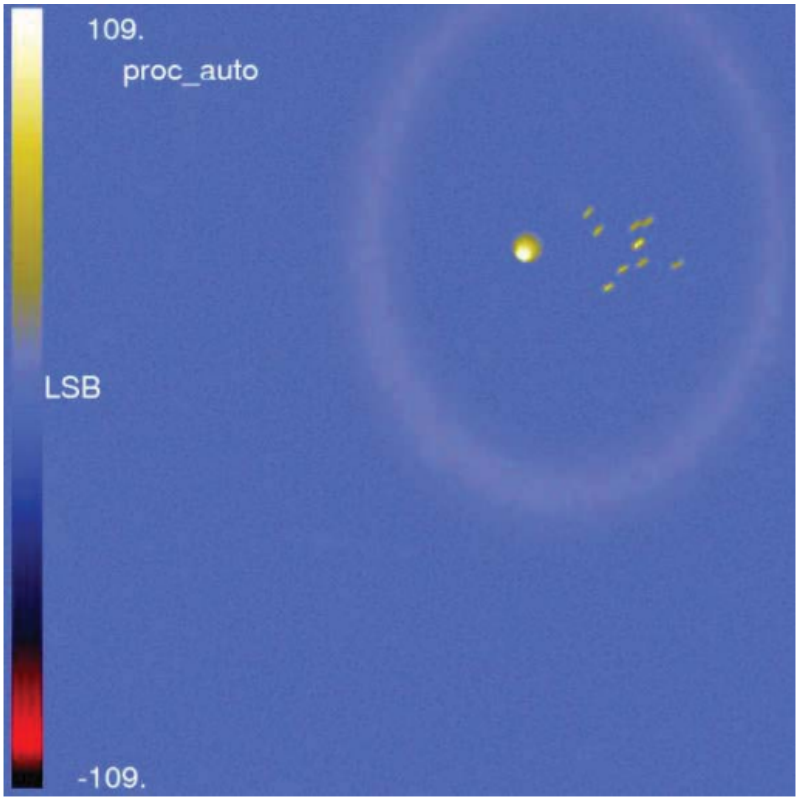
sa skladá z dvoch konvolučných neurónových sietí, ktoré chce v budúcnosti použiť na detekciu a klasifikáciu TLE javov. Finálny produkt je nainštalovaný na Astronomickom observatóriu na Kolonickom sedle na Slovensku. Tento projekt priniesol novú formu autonómneho pozorovania TLE bleskov. Prístroj je schopný urobiť dve snímky za sekundu. Za obdobie skoro jedného roku systém zaznamenal len dva TLE eventy, hlavne kvôli nedostatku búrok. Tento problém čiastočne vyriešil automatickým otáčaním prístroja. Na obrázku 3 – 3 je zobrazený jeden z dvoch nasnímaných TLE javov. Ide konkrétne o *Column sprite* čiže stĺp. Očakáva sa, že systém zachytí aj doposiaľ nedostatočne preskúmané druhy TLE. Plánom do budúcnosti je vytvoriť viac prístrojov a rozmiestniť ich na viaceré lokácie po celej Európe. Ďalej zlepšiť stabilitu, citlivosť a efektívnosť vytvoreného prístroja. Posledným krokom je zlepšenie dvoch neurónových sietí na klasifikáciu TLE.

Obrázok 3 – 3 TLE event, druh Column sprite zachytený prístrojom, ktorý vytvoril Samuel Amrich (Amrich et al., 2021).

Ďalšia práca, ktorá sa venovala pozorovaniu TLE eventov je práca od Kim et al. (2018) z Kórei. Práca má názov „Experimentálna validácia palubného systému na pozorovanie TLE pre VisionCube prostredníctvom prostredia Ground Simulation“. VisionCube CubeSat je prístroj vyvinutý špeciálne na zisťovanie výskytu TLE javov v hornej atmosfére a získavanie snímok javov z nízkej obežnej dráhy Zeme. Palubný systém sa skladá z multianódového fotonásobiča MaPMT a obrazového snímača CMOS. Ku prístroju je navrhnutý softvér, ktorý umožňuje detekciu TLE a zachytávanie obrázkov včas. Účelom tejto štúdie bolo preskúmať funkčnosť hardvéru a softvérového algoritmu, aby sa ukázalo, či palubný systém bude počas operácií vo vesmíre správne fungovať bez ľudského zásahu. Zariadenie je kompaktnej veľkosti. Na overenie funkčnosti sa uskutočnilo experimentálne overenie na pozemnom simulačnom zariadení. Pozemné simulačné zariadenie bolo navrhnuté tak, aby napodobňovalo TLE na zemi pomocou sady ultrafialových LED diód. Výsledky odhaľujú, že hardvér a softvérový algoritmus môžu efektívne detegovať TLE a získavať digitálne dáta, čo demonštruje schopnosť palubného pozorovacieho systému TLE na prevádzku na obežnej dráhe. Pozorovací systém TLE bude integrovaný do VisionCube a prejde kozmickými environmentálnymi testami. Jeho spustenie bolo naplánované na druhú polovicu roku 2018.

Publikácia od Blanc et al. (2017) o Taranis MCP sa zaoberá vytvorením nástroja na presné monitorovanie TLE vo vyšších vrstvách atmosféry. Mikrosatelit TARANIS je vyvinutý na štúdium impulzívnych presunov energie medzi zemskou atmosférou a vesmírnym prostredím, to sú napríklad TLE alebo TGF. Zhora pozoruje oblasti búrok na Zemi. Časť prístroja s názvom MCP má na starosti snímanie TLE. Jeho cieľom je identifikovať a presnejšie charakterizovať blesky a TLE v optických vlnových dĺžkach, určiť spektrálne vlastnosti ako jas, relatívnu polohu k ich materskému blesku a poskytnúť upozornenia pre všetky prístroje TARANIS na bežné TLE pozorovania vo vysokom rozlíšení. TLE je ťažké podrobne preskúmať zo Zeme kvôli elektromagnetickému spektru, preto chcú začať svoje prvé pozorovania z vesmíru, aby sa zlepšili ich vedomosti v tejto oblasti. Simuláciu identifikácie TLE prístrojom

TARANIS môžte vidieť na Obrázku 3 – 4. Štart misie, ktorý sa uskutočnil 17.11.2020 bol žiaľ neúspešný kvôli zlyhaniu nosnej rakety Malik (2020).



Obrázok 3 – 4 Simulácia identifikácie TLE prístrojom TARANIS Blanc et al. (2017).

4 Automatická detekcia TLE na snímkach AMOS pomocou hlbokého učenia

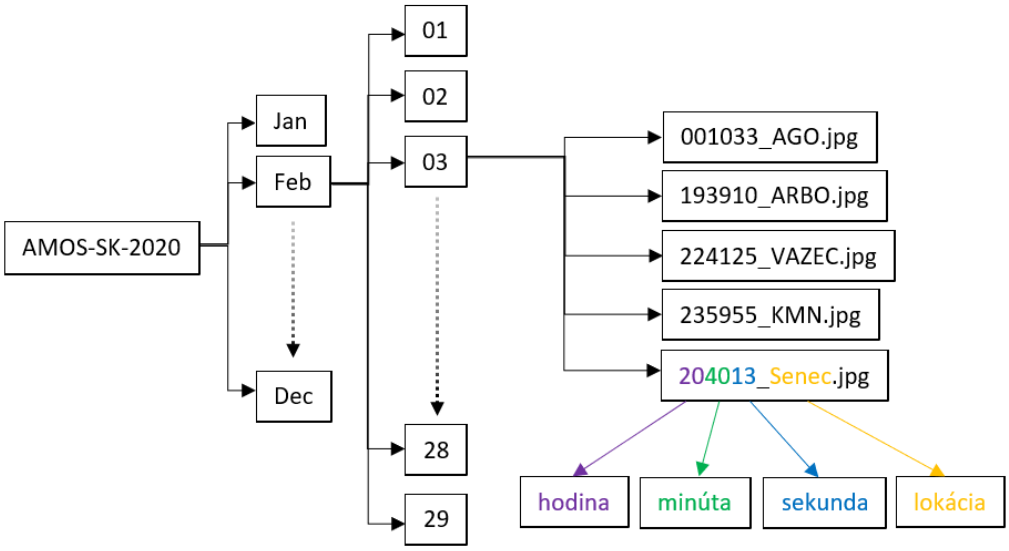
Cieľom tejto bakalárskej práce bolo vytvorenie anotovaného datasetu, trénovanie modelu založenom na hlbokom učení a následné vyhodnotenie jeho presnosti. Motiváciou je, že v oblasti vesmírneho výskumu sa vyskytuje nespočetné množstvo dát, na ktorých je potrebné detekovať špecifické udalosti. Takými sú aj nadoblačné blesky TLE, ktoré je možné pozorovať pomocou celooblohových kamier AMOS. Astronómovia majú k dispozícii za rok viac než 35 tisíc snímkov oblohy z kamery AMOS. Takýto počet dát prehľadať len s pomocou ľudskej sily je veľmi časovo náročné. Preto použijeme metódy hlbokého učenia, ktoré sú veľmi efektívne a rýchle pri detekcii nepravidelných a sporadických štruktúr. Plánovaným výstupom práce je neurónová sieť, ktorá bude sama schopná detekovať TLE javy na snímkach z kamier AMOS. Ďalším výstupom je anotovaný dataset doposiaľ nájdených TLE eventov na snímkach zo slovenských AMOS systémov. Ako posledným výstupom je tabuľka, v ktorej budú spracované základné údaje o nájdených javoch. Výstupnú tabuľku sme vytvorili spracovaním anotačného projektu a výstupov z neurónovej siete YOLOv5. Priebeh toho ako sme pri dosiahnutí týchto cieľov postupovali, je podrobne popísaný v nasledujúcich kapitolách.

4.1 Analýza dát

Dáta, ktoré boli základom na začatie bakalárskej práce nám dodal Doc. RNDr. Juraj Tóth, PhD., ktorý je vedúcim programu AMOS a pôsobí na Fakulte Matematiky Fyziky a Informatiky na Univerzite Komenského v Bratislave. Snímky pochádzajú zo systému AMOS.

Poskytnuté dáta boli v priečinku s názvom **AMOS-SK-2020**. Ako už z názvu vyplýva, dáta boli z obdobia roku 2020. Snímky boli rozdelené do ďalších priečinkov, podľa toho v ktorom dni a mesiaci boli vytvorené. Názov obrázku obsahoval čas vy-

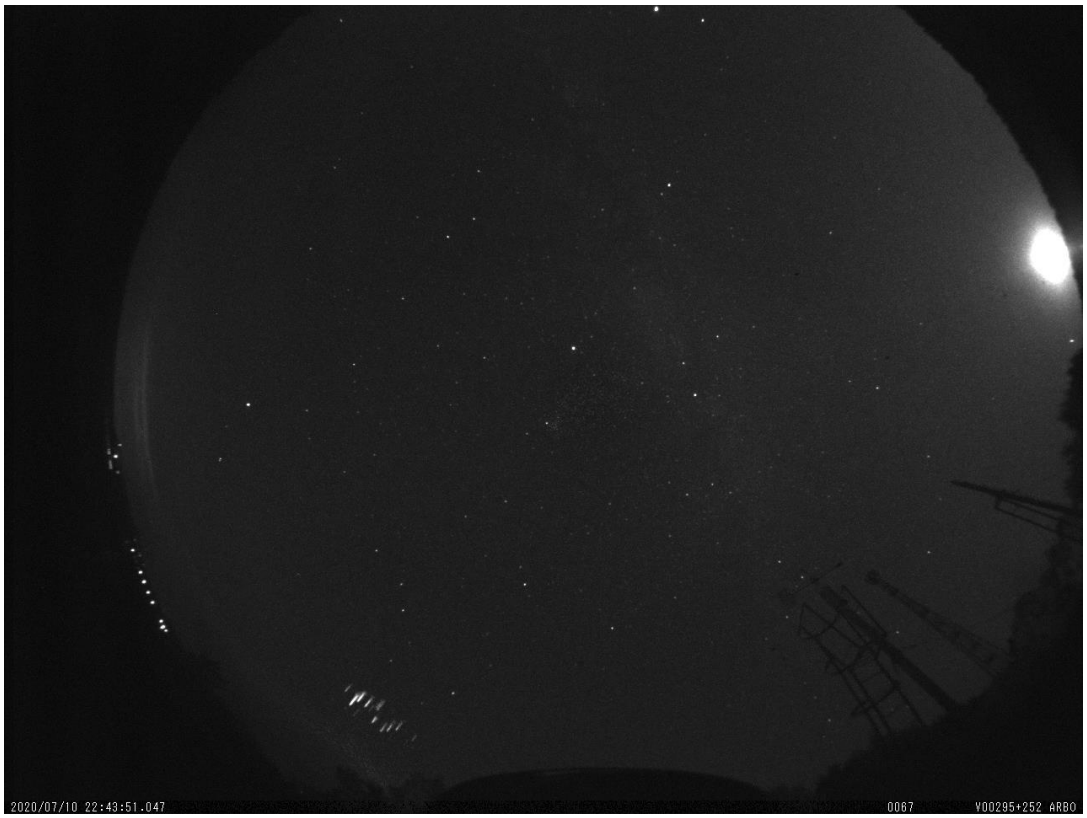
tvorenia snímky v tvare **HHMMSS_NNN.jpg**. NNN je lokácia AMOS kamery. Snímky boli z piatich lokácií na Slovensku. Konkrétne išlo o AGO, ARBO, KMN, VAZEC a Senec. Štruktúru priečinka s dátovým vstupom, ktorý bol vytvorený snímaním kamerou AMOS je možné vidieť na Obrázku 4 – 1.



Obrázok 4 – 1 Štruktúra priečinka, ktorý obsahoval vstupné dáta.

Priečinok obsahoval cez 35 tisíc dát. Dáta boli vo forme obrázku s rozmermi 1280 x 960 pixelov. Snímka zobrazuje celú oblohu v škále šedej. Snímka v podstate predstavuje priemet pozorovanej oblohy do roviny, čiže v strede snímky sa nachádza obloha v tvare kruhu, na okraji vidno časti zeme, ktoré kamera nasnímala. Blízko horizontu je možné vidieť napríklad vidieť stromy, osvetlenie, budovy či veže. Na oblohe sa nachádzajú hviezdy, meteory, oblaky, mesiac či iné javy. Medzi dátami sa nachádzali aj chybné snímky, kde kamera nenasnímala oblohu, lebo bola uzatvorená krytka. Na mnohých snímkach, hlavne tých viac presvetlených, môžeme vidieť hexagonálnu štruktúru kamery. Na každej snímke sa v dolnom ľavom rohu vyskytuje časový údaj o vytvorení snímky, ktorý obsahuje rok, mesiac, deň, hodinu, minútu, sekundu a milisekundu. V pravom dolnom rohu snímka obsahuje údaj o umiestnení kamery, teda lokáciu. Príklad snímky z kamery AMOS v lokalite ARBO je na

Obrázku 4 – 2.



Obrázok 4 – 2 Príklad snímky zo vstupných dát, na ktorej je možné vidieť TLE jav ale aj iné nami nepozorované objekty ako hviezdy, vežu, stromy, pouličné svetlá či mesiac.

Našou úlohou je na vstupných dátach nájsť TLE javy. Tieto javy sa nachádzajú výhradne v blízkosti horizontu, preto ich hľadáme vo vonkajšej časti kruhu. Pozorovať ich v strede celooblohovej snímky nie je možné.

4.2 Príprava dát

4.2.1 Sťahovanie dát a prvotné úpravy

Priečinok **AMOS-SK-2020** s dátami, ktorý sme obdržali, bol vo formáte zip a mal 4GB dát. Bolo ho potrebné stiahnuť do počítača na lokálne úložisko a ďalej extrahovať. Po tom čo bol priečinok prístupný, sme si ho prezreli a zistili, že obrázky sú vo viacerých priečinkoch. Bolo potrebné ich spojiť do jedného priečinka. Na to

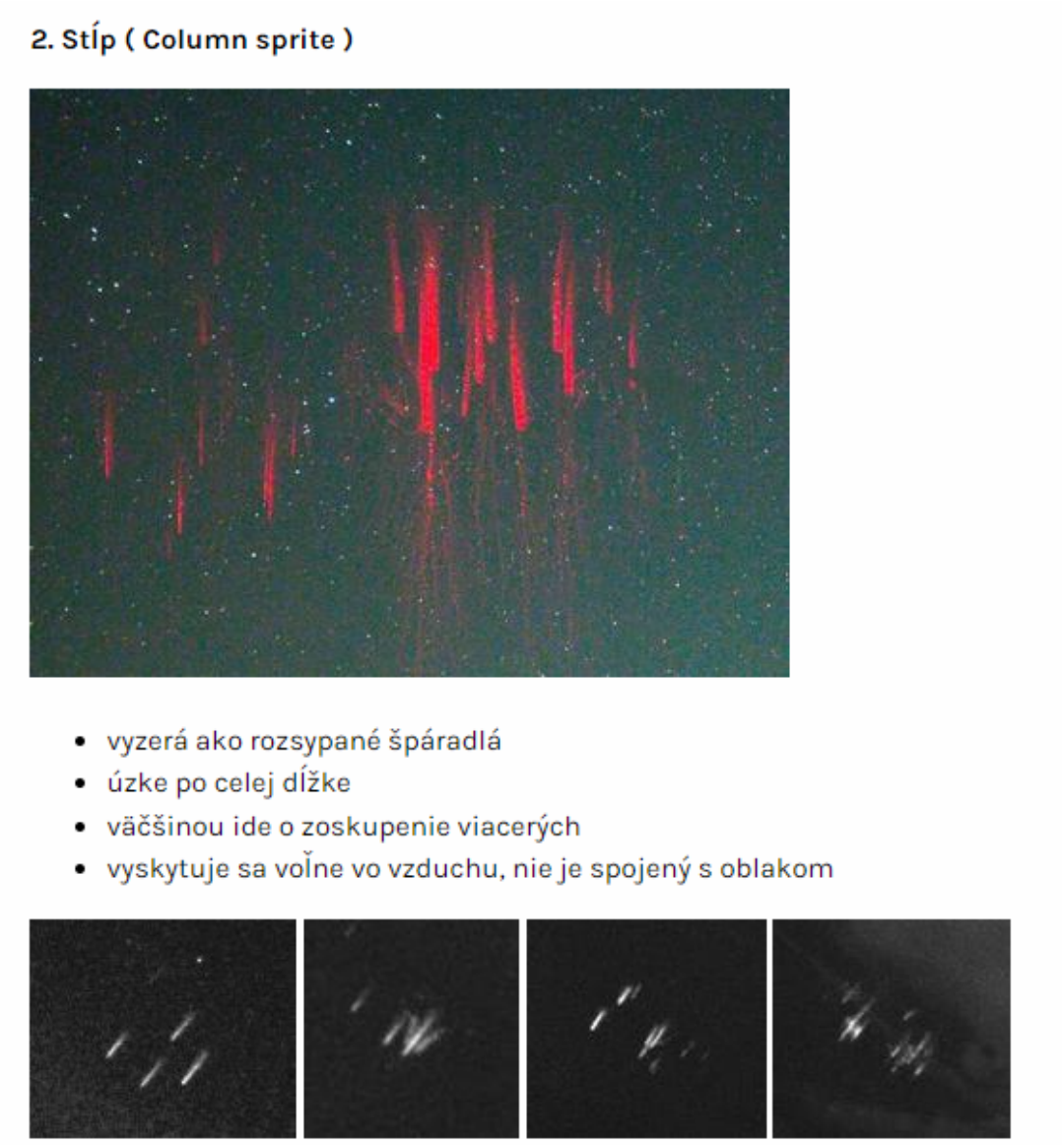
sme použili aplikáciu *Total Commander*, ktorá nám pomohla s touto úlohou. Žiadne ďalšie úpravy neboli potrebné aj keď dáta boli v surovom stave, rovno ako sa po pozorovaniach nahrali. Obrázky boli vo formáte jpg, tým pádom boli vhodné na následné použitie v anotačnom projekte.

4.2.2 Vytvorenie anotačného projektu

Prvým veľkým krokom práce bolo vytvorenie anotačného projektu. Anotačný projekt sme vyhodnotili ako najlepšie riešenie pre ručné prezretie tisícok obrázkov a označenie či anotovanie nájdených javov. Výsledkom projektu bude anotačný dataset už nájdených a rámčekom označených TLE eventov, ktoré použijeme ako vstup pri naučení neurónovej siete YOLOv5. S výberom vhodnej platformy sme to nemali obtiažne, konzultantka mi odporučila práve platformu Zooniverze, keďže sme mali niekoľko skúseností s využívaním tejto platformy v rámci iných výskumov a záverečných prác v našom centre. S touto platformou sme sa potrebovali najprv zoznámiť. Vyskúšali sme aj pár verejných anotačných projektov na rôzne témy, aby sme zistili ako to funguje. Po prvotnom zoznámení sme sa pustili do vytvorenia prvého anotačného projektu. Ten sme nazvali Detekcia TLE. Bolo potrebné vytvoriť úvodný štýl projektu. Ako obrázok, ktorý by ho mal prezentovať, sme zvolili fotku nádherného TLE eventu. Napísali som krátke úvodné predstavenie, o čo v tomto projekte pôjde, a čo je cieľom bakalárskej práce. V sekcii „*About*“ sme podrobne popísali, čo je cieľom anotačného projektu, kde majú dané javy anotátori hľadať a ako tieto javy môžu vyzerať. Tu sme pridali ku každému typu javu jeho slovný popis plus pár výstrižkov javov ako vyzerajú na snímkach z AMOS kamery. Anotátori mali za úlohu hľadať konkrétne tri najviac sa vyskytujúce javy, a to Mrkvu *(angl. Carrot sprite)*, Stĺp *(angl. Column sprite)* a Fontánu *(angl. Jet)*. Nahliadnutie tohto projektu je k dispozícii online.[2] Príklad opisu stĺpu môžete vidieť na Obrázku 4 – 3.

Ďalším krokom bolo nahratie všetkých dát do projektu. Potom sme vytvorili *workflow*, ide o postupnosť krokov, ktoré bude projekt mať. Tento projekt obsahoval

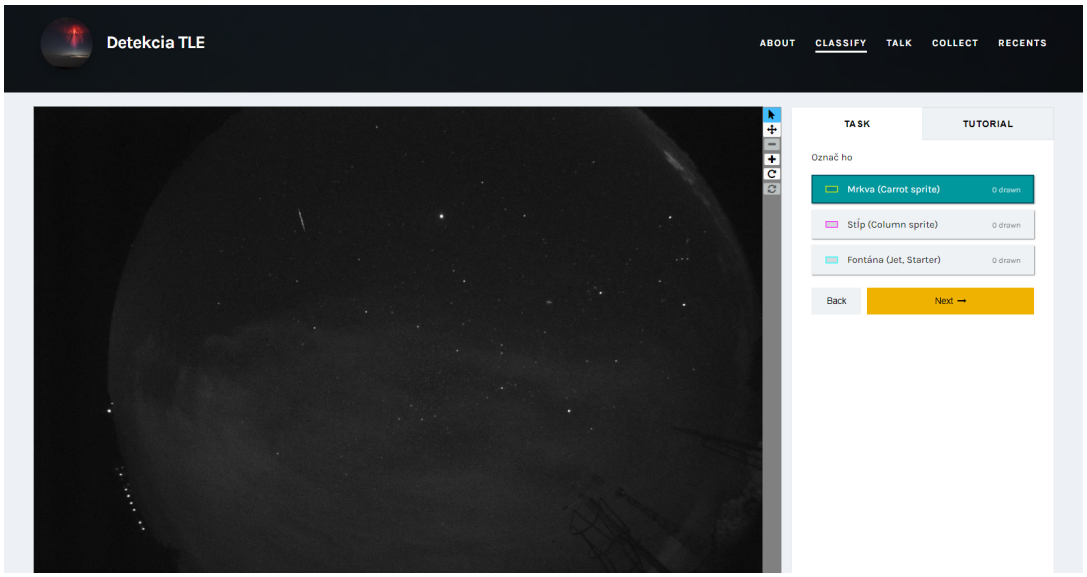
[2] `https://www.zooniverse.org/projects/petra-kamenska/detekcia-tle`



Obrázok 4 – 3 Výstrižok obrazovky z anotačného projektu na platforme Zooniverze zo sekcie About, kde je podrobne popísaný výzor stĺpu.

tri kroky. V prvom kroku mal anotojúci človek zistiť, či sa na obrázku nejaký event nachádza alebo nie. Krok obsahoval otázku „Vidíš na obrázku nejaký TLE?“ s odpoveďami „Áno“ alebo „Nie“. Ak si osoba nebola istá mala možnosť kliknúť na sekciu „*Need some help with this task?*“ v preklade potrebuješ pomoc s touto úlohou? Tá obsahovala označené príklady objektov, ktoré sa môžu na snímke vyskytovať ale nie sú TLE eventom, ako napríklad oblaky, svetlá či meteory a satelity. Ak

osoba zvolila možnosť nie, tak bolo jednoducho anotovanie danej snímky ukončené a projekt poskytol ďalšiu snímku. Ak však bolo zvolené áno, študent pokračoval na druhý krok, ktorým bolo výber druhu TLE a jeho následné označenie. Anotátor mal na výber už z troch, skôr spomínaných eventov a to Mrkvy, Stĺpu a Fontány. Klikol na typ a okolo blesku urobil rámček. Druhy TLE eventovom mali farebne rozlíšené rámčeky. Po skončení bolo potrebné kliknúť na tlačítko „*Next*“, teda ďalej. Následne bol zobrazený posledný krok, ktorý slúžil len na kontrolu, a to či anotátor naozaj označil všetky eventy, čo vidí. Pri tomto kroku bolo ešte možné sa vrátiť na krok dva. Po konfirmácii, že je naozaj všetko označené, projekt vygeneroval ďalší z obrázkov v databáze, ktorý ešte nebol nikým skontrolovaný. Na Obrázku 4 – 3 môžete vidieť náhľad projektu, konkrétne druhý krok.



Obrázok 4 – 4 Výstrižok obrazovky z anotačného projektu na platforme Zooniverze zo sekcie anotovanie, konkrétne zobrazujúci krok číslo 2, kde mali študenti za úlohu TLE jav označiť.

Po vytvorení postupnosti týchto krokov nám už len ostávalo vytvoriť *Tutorial* či postup pre anotujúcich. Tento tutoriál sa zobrazí hneď pri prvom otvorení projektu. Je veľmi dôležité, aby bol skonštruovaný jednoducho a pochopiteľne, a taktiež je dôležité, aby si ho anotátor, ktorý sa chystá anotovať, dôsledne naštudoval. Týmto predídeme možným chybným označeniam, ktoré by spôsobili ťažkosti pri učení ne-

urónovej siete. Postup pozostával z dvoch krokov. Ako prvým bola informácia o tom čo má anotátor robiť, a kde má jav hľadať. Bolo zdôraznené, aby si obrázok dôsledne prezrel, priblížil si ho a hlavne hľadal po okrajoch. Pre lepšie pochopenie čitateľa bol priložený aj nami upravený obrázok AMOS snímky so znázornením najčastejšej oblasti, kde sa javy nachádzajú. Tento krok môžete vidieť na Obrázku 4 – 5. V druhom kroku boli podrobne popísané druhy TLE bleskov spolu s ukážkami podobne ako v sekcii *About*. Tento postup bol kedykoľvek dostupný anotujúcej osobe, keď ho potrebovala.

Obrázok 4 – 5 Výstrižok obrazovky z anotačného projektu na platforme Zooniverze zo sekcie Tutorial, kde je vysvetlené, kde má anotátor TLE javy hľadať a čo má robiť.

4.2.3 Anotovanie

Anotačný projekt bol hotový a posledným krokom bolo ho zverejniť a začať anotovať. Anotovanie takéhoto veľkého množstva snímok jednou či pár osobami by bolo neskutočne časovo náročné, možno dokonca až nemožné. Preto nám s týmto pomohlo asi 80 mojich spolužiakov na odbore Hospodárska informatika z Technickej univerzity v Košiciach. Anotovanie bolo naplánované na štyri sedenia po hodine a pol. Na každom sedení sa zúčastnilo približne 20 študentov. Sedenie prebiehalo cez online hovor, na ktorom sme boli prítomní spolu s konzultantmi a školiteľom. Na začiatku bola študentom bakalárska práca predstavená a objasnené, čo budú robiť. Po úvode sa každý študent registroval, prihlásil a začal pracovať. Počas hovoru sme boli s konzultantmi k dispozícii na prípadné objasnenie situácie. Ak si študent nebol istý, zazdieľal nám obrazovku a my sme mu pomohli, či ide o TLE jav alebo nie.

Po ukončení všetkých štyroch sedení sme však zistili, že aj takéto množstvo ľudí nestihlo zanotovať všetky dáta. Preto bolo potrebné uskutočniť ešte jedno finálne sedenie, kde sa úloha dokončí. Na toto sedenie bol vytvorený úplne totožný projekt, ktorý obsahoval už len neoanotované dáta. Navyše boli ešte pridané dáta, ktorý nám poskytol konzultant Samuel Amrich, expert na TLE a študent Matematicko-fyzikálnej fakulty, Univerzity Karlovej v Praha, z ďalších rokov. Toto sedenie bolo úspešné a stihli sme prejsť všetky dáta z poskytnutého datasetu.

4.2.4 Spracovanie výstupu z anotačného projektu

Na platforme sme vyžiadali exporty z oboch anotačných projektov, ktoré boli zaslané na email. Nahrali sme ich do datalabu a začali spracovávanie. Súbory sa volali `detekcia-tle-classifications-3.csv` a `detekcia-tle-2-classifications.csv`. Údaje v týchto súboroch boli zapísané vo formáte JSON. Našou úlohou bolo teda tieto informácie zo súborov extrahovať a spracovať. Spracovali som skript, ktorý mal za úlohu spracovať tabuľku, ktorá bude obsahovať len obrázky, na ktorých bol jav zaznačený. Tabuľka obsahovala stĺpce, či je tam event, názov obrázka, id, súradnice

rámčeku: x, y, šírka a výška a typ TLE. Na Obrázku 4 – 14 môžte vidieť náhľad tejto tabuľky.

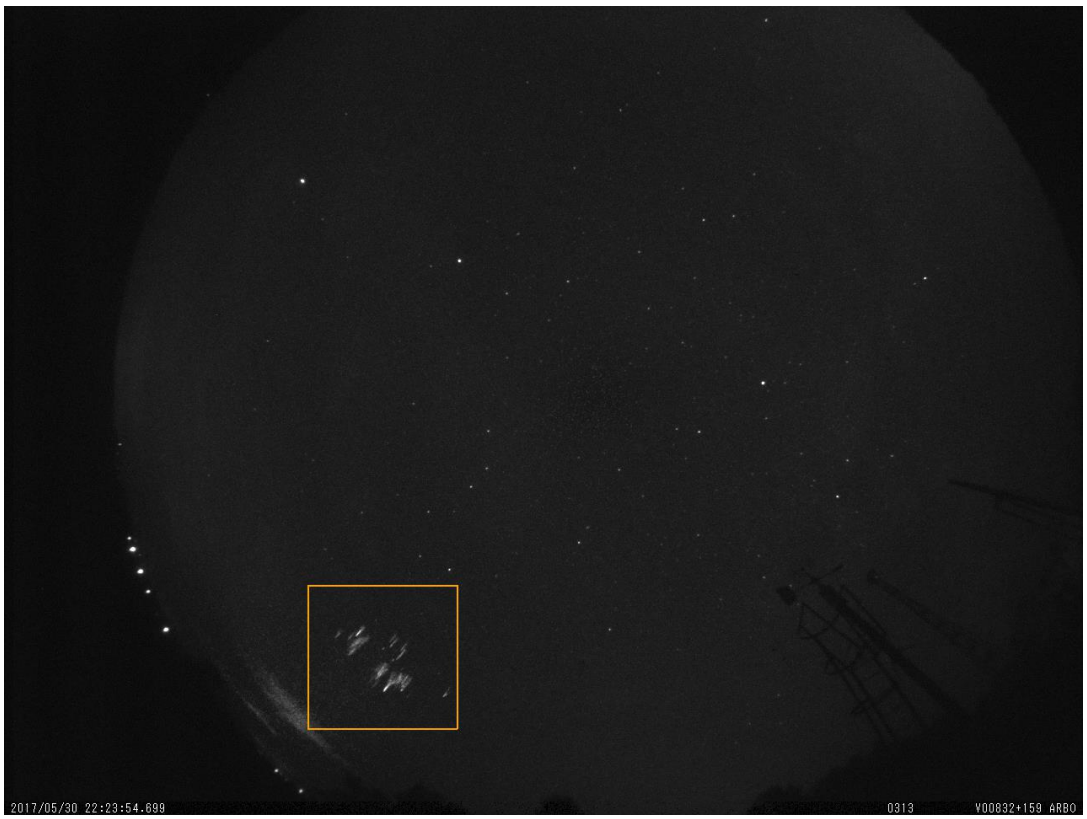
	is_tle	event	subject_id	x	y	width	height	type
1	Áno	530_222354_ARBO_P.jpg	70026921	360.1607971191406	687.64208984375	177.27883911132812	169.702880859375	Mrkva (Carrot sprite)
2	Áno	s205550_ARBO.jpg	70027115	499.9825439453125	0.21508385241031647	133.079833984375	37.21723960340023	Stĺp (Column sprite)
3	Áno	s232322_ARBO.jpg	70027166	241.5023956298828	208.9098663330078	92.40086364746094	109.20106506347656	Mrkva (Carrot sprite)
4	Áno	s231955_ARBO.jpg	70027165	231.01905822753906	128.31336975097656	152.71302795410156	279.0170135498047	Stĺp (Column sprite)
5	Áno	003159_AGO.jpg	70028961	992.318115234375	254.92942810058594	45.0386962890625	30.249893188476562	Stĺp (Column sprite)
6	Áno	530_205827_ARBO_P.jpg	70026904	402.4268493652344	799.02197265625	113.81173706054688	93.19677734375	Stĺp (Column sprite)
7	Áno	170331_AGO-Spec.jpg	70035616	1364.658203125	705.4586181640625	54.074462890625	73.99664306640625	Fontána (Jet, Starter)
8	Áno	s212829_ARBO.jpg	70027145	505.7397155761719	9.855901718139648	107.00692749023438	77.7208194732666	Stĺp (Column sprite)
9	Áno	s222928_ARBO.jpg	70027148	191.4687957763672	351.10357666015625	53.073028564453125	78.18795776367188	Stĺp (Column sprite)
10	Áno	s013140_ARBO.jpg	70027093	283.8411560058594	121.36839294433594	33.791656494140625	30.412490844726562	Mrkva (Carrot sprite)
11	Áno	s013140_ARBO.jpg	70027093	315.38006591796875	94.33507537841797	23.654144287109375	24.780548095703125	Mrkva (Carrot sprite)
12	Áno	3_213350_ARBO_P_C.jpg	70027071	56.1033821105957	6.553805351257324	27.12337875366211	41.48281192779541	Mrkva (Carrot sprite)
13	Áno	s235513_ARBO.jpg	70027173	1105.64111328125	489.1504211425781	107.5794677734375	106.06423950195312	Mrkva (Carrot sprite)
14	Áno	0_210417_ARBO_P_A.jpg	70026990	87.36114501953125	123.0000228881836	16	23.11107635498047	Stĺp (Column sprite)
15	Áno	222045_ARBO.jpg	70058908	200.48838806152344	529.12109375	34.918060302734375	45.0555419921875	Mrkva (Carrot sprite)

Obrázok 4 – 6 Náhľad výstupnej tabuľky z exportu z anotačného projektu, ktorá obsahuje údaje o nájdených TLE javoch.

Z tejto tabuľky som zistili, že nájdených je 290 eventov. Tieto bolo najprv potrebné skontrolovať doménovým expertom RNDr. Šimonom Mackovjakom, PhD. Pre túto kontrolu bolo nevyhnutné najprv spätne vykresliť tieto javy na obrázky. Napísali sme program, ktorý na dané obrázky pomocou súradníc vykreslil farebne rozlíšené rámčeky dávnejšie vykreslené študentami. Príklad môžete vidieť na Obrázku 4 – 7. Dataset vykreslených obrázkov bol poskytnutý doménovému expertovi, ktorý ich prezrel a vytriedil. Predtým ešte boli z tohto datasetu vytriedené výseky obrázkov, ktoré sa tam nachádzali ale neboli by vhodné pre ďalší postup. Po skontrolovaní sme obdržali dataset správnych obrázkov s počtom 142. Na týchto obrázkoch bolo nájdených 171 eventov. Pre lepší prehľad úspešnosť študentov v nájdení TLE javov bola 59 percent.

4.2.5 Výpočet stredu a normalizácia súradnic

Aby sme mohli naučiť neurónovú sieť YOLOv5 na rozpoznávanie TLE javov potrebovali sme ešte textové súbory s údajmi o umiestnení javu. Pomocou podobného programu ako na vytvorenie výstupnej tabuľky sme vytvorili pre každý obrázok, kde bol nájdený jav aj textový súbor. Tento súbor mal rovnaký názov ako obrázok ku



Obrázok 4 – 7 Spätne vykreslená anotácia z výstupnej tabuľky pre potreby skontrolovania správnosti datasetu. Oranžový rámček znamená, že ide o mrkvu.

ktorému patril s koncovkou .txt. Vnútri sa nachádzala hlavička s názvami: *label*, x, y, *width*, *height* a jeden riadok alebo viac so súradnicami a typom podľa toho, či sa na obrázku nachádzal jeden alebo viac javov. Príklad textového súboru môžete vidieť na Obrázku 4 – 8. Pre vysvetlenie prvý údaj udáva typ TLE blesku. Ak je to „0“ ide o **Mrkvu** ak „1“ ide o **Stĺp** „2“ ide o **Fontánu**. Ďalšie dva údaje predstavujú súradnicu ľavého horného rohu anotácie, prvá je pre os X a druhá pre os Y. Štvrtý údaj je šírka a posledný výška obdĺžnika anotácie. Medzi poslednými prípravami

```
label x y width height
0 1003.8726196289062 697.9915771484375 45.33966064453125 41.39715576171875
```

Obrázok 4 – 8 Náhľad textového súboru pred normalizáciou.

na naučenie neurónovej siete bola normalizácia súradníc javov. Normalizáciu bolo

nutné uskutočniť z dôvodu, že dané súradnice by platili len pre pôvodný rozmer obrázka. Ak by sme však obrázok zväčšili či zmenšili, súradnice by nesedeli. Vytvorili sme teda skript, ktorý prepočítal súradnice na interval od nula po jeden. Po lepšom preskúmaní metodiky YOLOv5 sme ešte zistili, že nie je nutné aby textové súbory obsahovali hlavičku s názvami stĺpcov, takže sme ju vymazali. Taktiež dôležitým zistením bolo, že YOLOv5 potrebuje miesto súradníc ľavého horného rohu súradnice stredu obdĺžnika. Tieto sme prepočítali pomocou daných vzorcov.

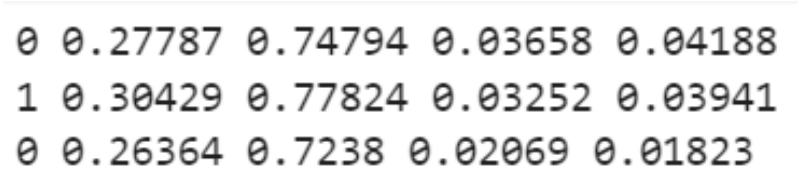
$$norm_x = \frac{x + \frac{width}{2}}{picture_{width}} \tag{4.1}$$

$$norm_y = \frac{y + \frac{height}{2}}{picture_{height}} \tag{4.2}$$

Vzorec (4.1) vypočíta normalizovanú hodnotu osi x stredu anotovaného obdĺžnika. V čitateli vzorca prepočítavame hodnotu z ľavého horného rohu na stred pripočítaním polovice šírky obdĺžnika. Túto hodnotu vydelíme celkovou šírkou obrázka *Picture width*, čo bolo u nás 1280 pixelov. Vzorec (4.2) funguje na rovnakom princípe len vypočíta normalizovanú hodnotu osi y stredu anotovaného obdĺžnika. *Picture height* bola v tomto prípade výška obrázka a to 960 pixelov. Hodnoty súradníc sme pevne zaokrúhlili na päť desatinných miest pre lepší prehľad. Hodnoty sú oddelené medzerou a každý nový riadok predstavuje novú anotáciu. Druhá hodnota predstavuje znormalizovanú x-ovú súradnicu stredu, a tretia znormalizovanú y-ovú súradnicu stredu anotovaného rámčeka. Zvyšné dve hodnoty predstavujú šírku a výšku anotovaného rámčeka. Takto vznikli nové prepísané textové súbory bez hlavičky už s normalizovanými hodnotami od nula po jeden, ktoré môžete vidieť na Obrázku 4 – 9.

4.2.6 Rozdelenie dát

Výsledný dataset 142 pozitívnych obrázkov, bol skoro pripravený na použitie. Už sme ho len rozdelili do troch množín: trénovacej, validačnej a testovacej. Pomer sme zvolili 8:1:1, keďže je dôležité, aby trénovacia množina na ktorej sa model hlbokého



Obrázok 4 – 9 Náhľad textového súboru po normalizácii.

učenia učí, obsahovala čo najviac dát. Táto množina obsahovala 114 obrázkov. Obe, validačná aj testovacia množina, obsahovali po 14 kusov obrázkov. Validačná množina slúži ako kontrola pred ukončením učenia neurónovej siete. K testovacej nemá model prístup počas učenia. Na tejto množine testujeme už naučený model. Všetky tri množiny sú od seba nezávislé, teda každá obsahuje jedinečné obrázky. Po prvotných skúšaniach modelu sme však zistili, že testovacia množina by mala obsahovať aj negatívne snímky, teda snímky bez výskutu TLE eventu. Preto sme do testovacej množiny vložili 50 negatívnych snímok, pre porovnanie aké falošne pozitívne javy model označuje.

4.3 Proces modelovania

4.3.1 Prvotné trénovanie

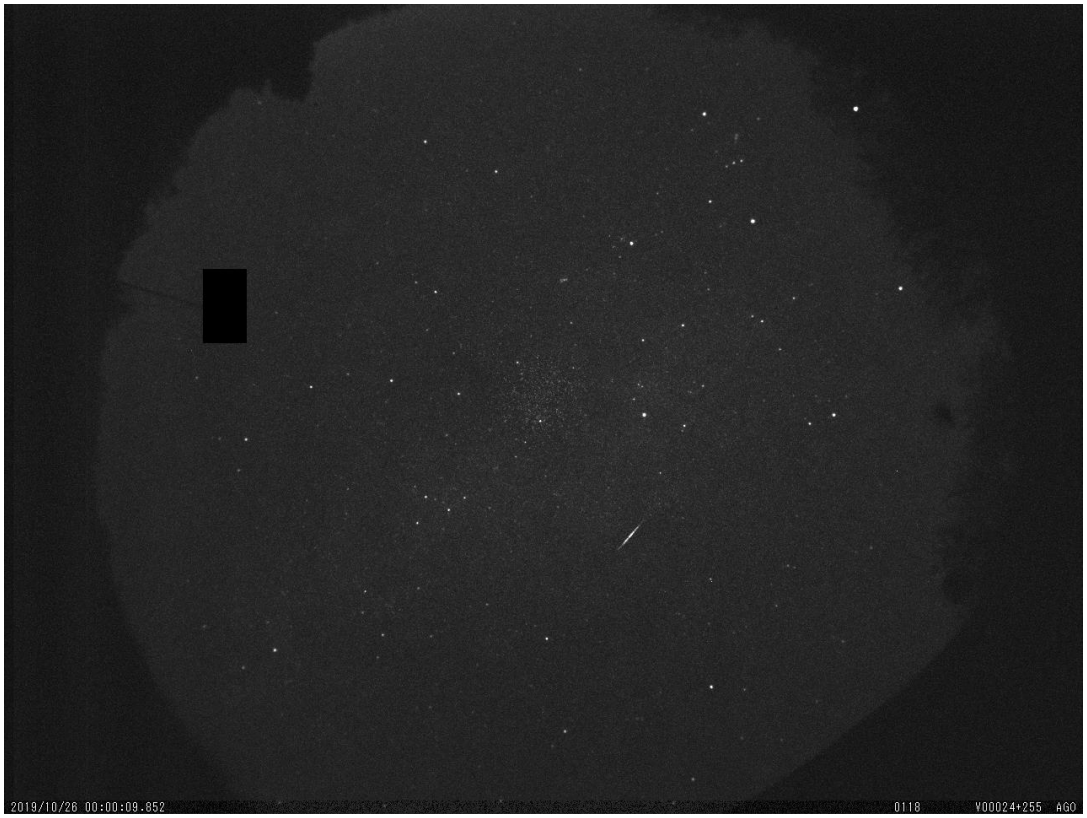
Po ukončení prípravy datasetu sme prešli na fázu trénovania. Trieda mrkva obsahovala 72 javov, trieda stĺp 84 javov a fontána len 15 javov v trénovacej množine. Po prvotných pokusoch s trénovaním s takým malým množstvom vstupných dát sme zistili, že najlepšie bude všetky tri druhy TLE javov spojiť to jedného. Minimálny počet dát pre trénovanie v jednej triede, by mal byť okolo 1000 obrázkov. Po spojení troch tried je trieda zložená zo 171 dát. Ideálne by bolo ak by trénovacia množina obsahovala medzi 10 až 100 tisíc dát. Množstvo dát je potrebné pre strojové učenie, aby získalo veľa informácií a jeho výsledky boli presné. Takže sme upravili textové súbory, aby typ TLE obsahoval jedine číslo nula, ktoré reprezentuje TLE event. V budúcej práci po detekovaní eventov ako takých rozšírime tento dataset a budeme

sa pokúšať vytvoriť model, ktorý dokáže rozlíšiť o aký druh TLE javu ide. Prvé trénovanie však nedosahovalo dostatočnú úspešnosť aj napriek spojeniu druhov TLE javov. Model hľadal veľa falošne pozitívnych javov a tie, ktoré mal nájsť, nenašiel. Jediným východiskom bolo získanie ďalších dát. A preto sme poprosili Doc. RNDr. Juraja Tótha, PhD. o zaslanie ďalších množín z pozorovaní systémom AMOS. Postupom času zaslal dáta z rokov 2016 až 2021. Každý z rokov obsahoval do 50 tisíc snímok. Pri každom roku sme postupovali rovnako. Ako prvé sme si museli daný zazipovaný priečinok stiahnuť a extrahovať. Pretože všetky priečinky mali rovnaké členenie ako pôvodný priečinok AMOS-SK-2020, museli sme všetky dáta spojiť do jedného priečinku. Na to som použili vytvorený kód. Tým pádom sme mali pripravenú novú testovaciu množinu. Na nej sme otestovali stále ten najlepší model, ktorý bol k dispozícii. Kvôli veľkému množstvu dát tento proces trval aj niekoľko hodín. Výsledkom bol priečinok s anotáciami od modelu. Tých bolo však aj 10 tisíc. Po ručnom prezretí väčšiny obrázkov sme prišli na zistenie, že model označuje falošne pozitívne javy ako napríklad nejaké svetlo či časť veže. Pre počítačové videnie to bolo samozrejmosťou, keďže tieto objekty sa veľmi podobali na TLE javy. Ľudskému oku však neujde, že to nie je správne. Bolo potrebné použiť nejakú metódu postprocessingu alebo preprocessingu, ktorá tieto chyby odstráni.

4.3.2 Odstránenie falošne pozitívnych javov

Zvolili sme *preprocessing*, takže úpravu dát pred použitím. Zostavili sme skript, ktorý na všetkých dátach testovacej množiny, vyčierni oblasti, ktoré model falošne označuje za pozitívne. Našli sme si najčastejšie oblasti, ktoré sa na obrázku vyskytujú stále pre každú oblasť. Z toho čo model označil, sme vytiahli súradnice. Tieto súradnice boli súradnicami stredu. My sme potrebovali súradnice ľavého horného rohu pre rozmery obrázka 1280 x 960 pixelov. Na tento výpočet sme použili presne opačné vzorce ako pri normalizácii. Po zistení všetkých súradníc sme ich doplnili do kódu, ktorý mal za úlohu nakresliť čierny obdĺžnik na danú polohu. Ukážku môže vidieť na Obrázku 4 – 10. Tento kód sme použili na všetky dáta od roku 2016, ktoré

sme potrebovali prejsť.



Obrázok 4 – 10 Náhľad snímky po zbavení sa falošne pozitívnych javov vyčiernením statických objektov, ktoré sa vyskytujú na každej snímke z danej lokácie.

4.3.3 Iteratívny prístup zlepšovania modelu

Po opätovnom spustený toho doposiaľ najlepšieho natrenovaného modelu na upravenej testovacej množine preprocessingom sme dostali omnoho lepší výsledok. Pri každom roku sa číslo nájdených anotácií pohybovalo okolo 2 až 3 tisíc. Tieto výsledky sme museli prejsť ručne a nájsť len pozitívne javy. Keďže výstup z YOLOv5 testovania obsahuje priečinok so všetkými obrázkami, aj tými na ktorých sa nič nenašlo, museli byť vytriedené. Bol použitý skript, ktorý zozbieral názvy len anotovaných obrázkov a zvyšné vymazal. Bolo začaté ručné triedenie. Pri každom datasete bolo nájdených približne 50 pozitívnych javov. Tieto sme vytriedili a vložili ich k

pôvodnej trénovacej množine. Takto sa počet v trénovacej množine zvyšoval, a tým pádom sa aj zlepšoval výstup z modelu. Po otestovaní všetkých datasetov, ktoré boli poskytnuté sa počet v trénovacej množine zdvojnásobil. Najlepší model, ktorý bol dostupný, bol použitý na kontrolu. Ešte raz sme otestovali všetky datasety, ktoré už neobsahovali pozitívne snímky, ktoré našiel minulý model. Zase boli pretriedené a ku každému roku pribudlo ešte pár nájdených javov. Toto je dôkazom, že sa model zlepšoval. Vo výsledku sme mali 357 obrázkov, na ktorých sa našlo 408 pozitívnych javov. Čo je o 2,5 krát väčší počet ako bol pôvodný dataset.

4.3.4 Finálna konfigurácia modelu

S trénovaním sme pokračovali, až kým sme nenašli najlepšiu kombináciu všetkých parametrov. Toto je príkaz, ktorý bol použitý na trénovanie:

```
!python yolov5/train.py --img 640 --rect --batch 60 --epochs 300
--data ./yolo/tle8.yaml --cfg ./yolov5/models/yolov5s.yaml
--save-period 10 --workers 0
```

Teraz vám budú bližšie predstavené dané časti kódu:

- `!python`

  Časť kódu volajúca Python súbor.

- `yolov5/train.py`

  Časť kódu, ktorá odkazuje na cestu ku skriptu na trénovanie. Tento súbor je uložený v priečinku `yolov5`.

- `--img 640`

  Je to parameter, ktorý reprezentuje veľkosť obrázka v pixeloch. Presnejšie ide o šírku. Pôvodná veľkosť obrázka bola 1280 pixelov. Rozhodli sme sa ale tré-

novať s polovičnou veľkosťou, kvôli nedostatočnému výkonu počítača. [3] Takto vstupné obrázky budú veľkosti 640 x 480 pixelov.

- `--rect`

  Tento parameter, má za úlohu zabezpečiť, aby YOLOv5 nezmenil trénovaciu množinu obrázkov na štvorce s rovnakou šírkou aj výškou. Ak by bol tento parameter vynechaný, vstupy by boli zdeformované s veľkosťou 1280 x 1280 pixelov. Použitím teda dosiahneme aby boli obrázky v tvare obdĺžnika.

- `--batch 60`

  Parameter *batch size* sme zvolili 60. Ide o najväčšie číslo, ktoré nám dovolil výkon počítača. Čím je číslo väčšie, tým sú aj výsledné štatistiky presnejšie. Ide o počet trénovacích vstupov, ktoré sú použité v jednej iterácii naraz.

- `--epochs 300`

  Ide o počet epoch. Tento parameter určí, koľkokrát prejde každý z trénovacích vstupov cez neurónovú sieť. My som zvolili číslo 300. Keďže sa trénovanie aj tak zastavilo po 245. epoche kvôli tomu, že trénovanie neprejavovalo znaky zlepšenia, nebolo potrebné počet epoch zväčšiť. Zmenšiť počet epoch by nebolo efektívne, lebo by som mohli prísť o najlepšie výsledky trénovania.

- `--data ./yolo/tle8.yaml`

  Tento parameter predstavuje kompletnú cestu ku súboru `tle8.yaml`. V tomto súbore sú uložené cesty na všetky tri množiny: trénovaciu, validačnú aj testovaciu. Navyše je tam uložená informácia o triedach. Keďže my som všetky druhy spojili, nachádzala sa tam len jedna trieda, ktorú reprezentovalo číslo nula s názvom *event*. Súbor sa volá `tle8` z dôvodu, že ide o ôsmu generá-

[3] Na trénovanie bola použitá grafická karta Quadro RTX 4000. Jej technické špecifikácie sú dostupné na `https://www.nvidia.com/en-us/design-visualization/previous-quadro-desktop-gpus/`.

ciu tohto súboru, keďže sme postupne pridávali nájdené pozitívne snímky do učenia.

- `--cfg ./yolov5/models/yolov5s.yaml`

  Je to parameter s cestou k modelu, ktorý bol použitý. My sme použili model YOLOv5s. Ide o veľkosť S a druhý najmenší model, ktorý YOLOv5 ponúka. Dôvodom použitia práve tohto modelu je výkon počítača. Mali som možnosť použiť aj M model, vtedy však muselo byť *batch size* veľmi nízke, čo neprinášalo dobré výsledky.

- `--save-period 10`

  Tento parameter je nepovinný a ide o periódu ukladania výsledkov. Použili sme ho z dôvodu, nájdenia najlepších výsledkov. Bez použitia tohto parametra YOLOv5 uloží len najlepšie a posledné výsledky. Tieto najlepšie neboli častokrát najlepšími kvôli nízkemu počtu *batch size* parametra, preto sme zvolili radšej ručné prezretie výsledkov učenia. Takto sme mali uloženú každú desiatu epochu od začiatku až pokiaľ sa trénovanie skončilo.

- `--workers 0`

  Parameter označuje počet procesov, ktoré sa paralelne generujú v dávkach. Nastavené na 0, keďže len v tomto prípade išla nastaviť veľká hodnota batch.

4.4 Detekcia

Po natrénovaní finálneho modelu, bolo potrebné ho otestovať. Detekciu bola spustená týmto kódom:

```
!python yolov5/detect.py --weights
./yolov5/runs/train/exp103/weights/epoch190.pt
--source ./yolo/test3 --iou 0.3 --conf 0.45 --img 640 --save-txt
```

Predtým než sme našli túto najlepšiu kombináciu, odskúšali sme každú desiatu epochu na rovnakej testovanej množine. Do excelovského súboru sme si zapisovali štatistiku výsledkov pre následné vyhodnotenie. Ten môžete vidieť na Obrázku 4 – 11. Na obrázku je žltým vyznačený najlepší výsledok. Takúto tabuľku sme si robili pre všetky natrénované modely, ktoré sme počas procesu mali. Na týchto výsledok je možné vidieť progres zlepšovania sa modelu, pridávaním ďalších dát do trénovacej množiny. Testovacia množina, na ktorej sme testovali, obsahovala 14 pozitívnych

!python yolov5/train.py --img 640 --rect --batch 60 --epochs 300 --data ./yolo/tle8.yaml --cfg ./yolov5/models/yolov5s.yaml --save-period 10 --workers 0								
skoncilo pri 245	exp103	epoch	iou	conf	vsetko	dobre	zle	
	exp419	best	0.3	0.45	76	14	62	
	exp420	last	0.3	0.45	55	14	41	
	exp421	epoch240	0.3	0.45	46	14	32	
	exp422	epoch230	0.3	0.45	55	14	41	
	exp423	epoch220	0.3	0.45	75	14	61	
	exp424	epoch210	0.3	0.45	71	14	57	
	exp425	epoch200	0.3	0.45	45	14	31	
	exp426	epoch190	0.3	0.45	25	14	11	
	exp427	epoch180	0.3	0.45	82	14	68	
	exp428	epoch170	0.3	0.45	50	14	36	
	exp429	epoch160	0.3	0.45	39	12	27	
	exp430	epoch150	0.3	0.45	108	13	95	
	exp431	epoch140	0.3	0.45	85	12	73	
	exp432	epoch130	0.3	0.45	41	14	27	
	exp433	epoch120	0.3	0.45	24	13	11	
	exp434	epoch110	0.3	0.45	33	14	19	
	exp435	epoch100	0.3	0.45	23	14	9	
	exp436	epoch90	0.3	0.45	66	14	52	
	exp437	epoch80	0.3	0.45	24	13	11	
	exp438	epoch70	0.3	0.45	13	13	0	
	exp439	epoch60	0.3	0.45	53	14	39	

Obrázok 4 – 11 Náhľad súboru v exceli, kam som si značila štatistiky z rôznych epoch.

obrázkov a 500 negatívnych obrázkov. Táto množina bola treťou v poradí preto je nazvaná `test3`. Tento príkaz pracuje so súborom `detect.py`. Ďalšie parametre, ktoré boli nastavené sú:

- `--img 640`

  Veľkosť obrázka ako aj pri trénovaní je nastavená na 640 pixelov. Teda polovičná veľkosť pôvodného obrázka.

- `--save-txt`

  Táto časť má za úlohu, uložiť textové súbory anotovaných obrázkov. Ak by v príkaze chýbala, jediné čo by obsahoval výstupný priečinok sú obrázky s anotáciami bez súradníc.

- `--iou 0.3`

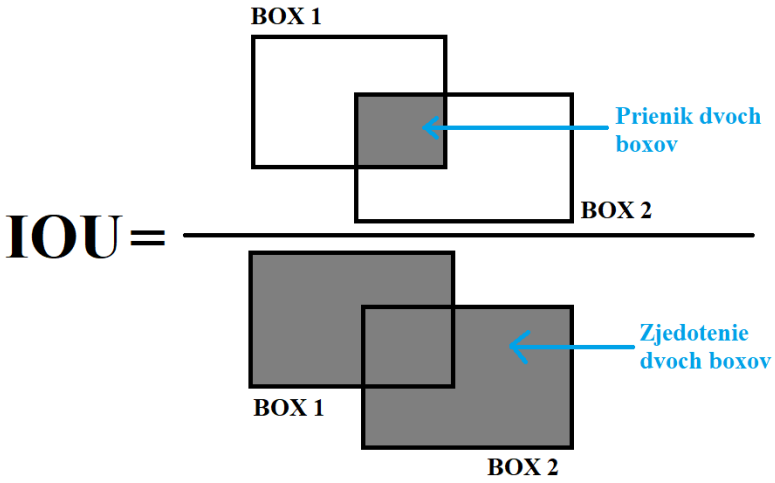
  Je to skratka pre pojem *Intersection over Union*. Ide o metriku, ktorá opisuje rozsah prekrytia dvoch polí. Čím väčšie je prekrytie, tým väčší je parameter IoU. Je najobľúbenejšia hodnotiaca metrika používaná pri detekcii objektov, ako je písané v článku od (Rezatofighi et al., 2019). Tento parameter je možné vypočítať na základe vzťahu:

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{prienik}{zjednotenie} \tag{4.3}$$

- `--conf 0.45`

  Confidence alebo *Confidence threshold* je pravdepodobnosť s akou patrí daná anotácia do danej triedy. Všetky anotácie, ktoré majú pravdepodobnosť väčšiu ako zadaná budú vykreslené, menšie nie.

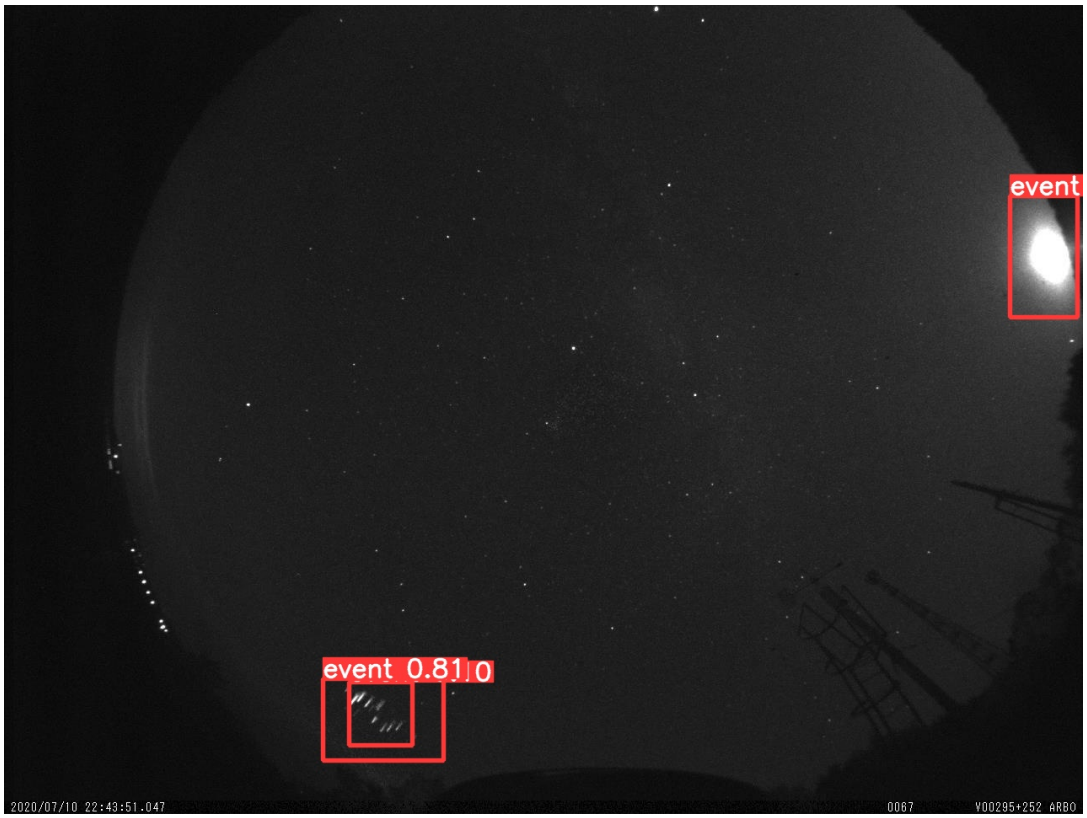
Graficky znázornený Vzorec 4.3 môžte vidieť na Obrázku 4 – 12. Po viacerých kom-



Obrázok 4 – 12 Graficky znázornený Vzorec 4.3 výpočtu metriky IoU.

bináciách parametrov IoU a *confidence* sme prišli na najlepšiu kombináciu a to iou = 0.3 a conf = 0.45. Začali sme s predvolenými hodnotami parametrov a to iou = 0.25 a conf = 0.45. Ak sme volili vyššie číslo pri parametri IoU, na jednom obrázku bolo viac označených anotácií okolo jedného TLE javu. Príklad vysokého IoU

a jeho následkov môžete vidieť na Obrázku 4 – 13. Ak sme hodnotu IoU zvolili veľmi nízku, model našiel veľa negatívnych javov. Pravdepodobnosť sme vybrali na hranici 45 percent. Táto hodnota je pre náš prípad optimálna z dôvodu, že nájde väčšinu skutočne pozitívnych javov a nie veľké falošne pozitívnych. Aj keď pozitívne javy v testovacej množine majú vysoké pravdepodobnosti okolo 70-80 percent nezvyšovali sme ju. Stále je lepším prípadom, ak model nájde falošne pozitívny jav, ktorý sa pri ručnej kontrole odfiltruje ako by model nezachytil jav skutočne pozitívny. Ak by sme pravdepodobnosť nastavili na nižšiu hodnotu, len by pribúdali falošne pozitívne javy.



Obrázok 4 – 13 Označené anotácie pri veľkom IoU.

4.4.1 Výstupná tabuľka

Nakoniec sme vytvorili tabuľku, obsahujúcu všetky pozitívne obrázky nájdené počas tejto práce. Spojili sme všetky roky a dokopy máme 357 obrázkov, na ktorých je 408 javov. Vytvorili sme skript, ktorý vytvorí tabuľku. Tá pozostáva z údajov ako *filename*, *x*, *y*, *width*, *height* . Čiže názov súboru a súradnice stredu so šírkou a dĺžkou anotácie. Časť tabuľky môžete vidieť na Obrázku 4 – 14.

	filename	x	y	width	height
1	20140804_225613_ARBO	0.19463	0.22703	0.03873	0.04435
2	20140804_234916_ARBO	0.8996	0.52025	0.08702	0.1627
3	20140804_235131_ARBO	0.90294	0.49236	0.05525	0.08428
4	20140804_235513_ARBO	0.90581	0.56477	0.08405	0.11048
5	20140805_000233_ARBO	0.91317	0.50452	0.06109	0.07443
6	20140805_001019_ARBO	0.89752	0.55136	0.09352	0.21623
7	20140805_001512_ARBO	0.89764	0.49953	0.04832	0.10452
8	20140805_013140_ARBO	0.23495	0.14227	0.0264	0.03168
9	20140805_013140_ARBO	0.25563	0.11117	0.01848	0.02581
10	20150717_223106_ARBO (1)	0.198	0.31353	0.03545	0.07947
11	20150717_223106_ARBO (2)	0.1966	0.31598	0.03278	0.03669
12	20150717_224037_ARBO	0.18378	0.35117	0.04558	0.04931
13	20150717_225317_ARBO	0.18232	0.37977	0.02846	0.03202
14	20150717_225317_ARBO	0.19566	0.327	0.04803	0.05455
15	20150717_230031_ARBO	0.22697	0.25745	0.03115	0.02714
16	20150717_230031_ARBO	0.21417	0.28953	0.02468	0.04195
17	20150717_230031_ARBO	0.20769	0.33354	0.0438	0.03619

Obrázok 4 – 14 Výstupná tabuľka, so všetkými nájdenými TLE eventami, počas mojej práce.

4.5 Vyhodnotenie výsledkov

Po ukončení experimentu nám ostal len krok vyhodnotenia. Niektoré grafy a metriky, vypočíta sám YOLOv5 na validačnej množine. Po natrénovaní modelu sme dostali tabuľkový výstup, ktorý môžete vidieť na Obrázku 4 – 15. Vysvetlenie da-

Class	Images	Labels	P	R	mAP@.5	mAP@
all	14	19	0.667	0.526	0.474	0.243

Obrázok 4 – 15 Tabuľka s metrikami, ktorý vyhodí YOLOv5 po trénovaní.

ných stĺpcov:

- *Class*, keďže sme nakoniec trénovali len s jednou triedou - event tabuľka, má len jeden riadok. Prvý stĺpec označuje presne túto skutočnosť.

- *Images* hovorí o tom, koľko dát je vo validačnej množine. V našom prípade je to 14 obrázkov, na ktorých je tak isto 14 javov, čo je desať percent z pôvodnej trénovacej množiny.

- *Labels* je stĺpec, ktorý označuje, koľko anotácií urobil model na validačnej množine. V tomto prípade ich bolo 19.

- *P* je skratka pre metriku zvaná *precision* v preklade presnosť. Ide o schopnosť modelu identifikovať iba relevantné objekty. Je to percento správnych pozitívnych predpovedí modelu Padilla et al. (2020). Inak by sme túto metriku mohli nazvať aj precíznosť. Vyjadruje pomer medzi skutočne nájdenými pozitívnymi javmi a všetkými nájdenými pozitívnymi javmi teda súčtom skutočne pozitívnych a falošne pozitívnych javov. Pre výpočet presnosti sa používa Vzorec 4.4. Presnosť nášho modelu je 66,7 percenta.

$$Presnosť = \frac{TP}{TP + FP} = \frac{skutočne\ pozitívne}{všetky\ nájdené\ predikcie} \tag{4.4}$$

- *R* je skratka pre metriku zvaná *recall* v preklade návratnosť. Je to schopnosť modelu nájsť všetky relevantné prípady, všetky pôvodne označené anotácie. Je to percento správne predikovaných pozitívnych prípadov medzi všetkými označenými anotáciami na vstupe. Matematický Vzorec 4.5 vyjadruje pomer medzi skutočne nájdenými pozitívnymi javmi a súčtom skutočne pozitívnych a falošne negatívnych javov. Návratnosť nášho modelu je 52,6 percenta.

$$Návratnosť = \frac{TP}{TP + FN} = \frac{skutočne\ pozitívne}{všetky\ anotácie} \tag{4.5}$$

- *mAP* je skratka pre metriku zvaná *mean Average Precision*. Ide o strednú priemernú presnosť. Počíta sa z hodnôt návratnosti od nula po jeden. Všeobecnou definíciou priemernej presnosti (AP) je nájdenie oblasti pod vyššie

uvedenou krivkou presnosti. AP sa vypočíta ako vážený priemer presností pri každom *thresholde*. mAP je priemer AP. mAP je kompromis medzi presnosťou a návratnosťou, a maximalizuje účinok oboch metrík. YOLO počíta mAP v desiatich intervaloch pre IoU medzi 0,5 a 0,95. Desať mAP hodnôt sa spriemeruje pre mAP@0,5:0,95 a prvý interval sa použije pre mAP@0,5.

Výsledky nie sú síce dokonalé, ale na počet obrázkov v trénovacej množine sú dostačujúce. Výstup z trénovania v YOLOv5 sa skladá aj z kontingenčnej tabuľky *(angl. confusion matrix). Event* je jediná trieda, ktorú model predikoval, keďže sme všetky typy TLE javov spojili do jedného. Ľavá strana tabuľky predstavuje skutočné hodnoty získané z anotácií a horná strana predikované hodnoty modelom. Tabuľka obsahuje dalšie hodnoty a to *Background False Negative* čiže falošne negatívne hodnoty, ide o množinu javov, ktoré model nenašiel, ale v anotáciách boli označené a *Background False Positive* v preklade falošne pozitívne hodnoty, teda tie ktoré model našiel, no v anotáciách neboli označené.

Finálny najlepší model otestovaný na testovacej množine pozostávajúcej z 514 obrázkov označil 25 anotácií. Z tohto počtu model našiel 14 skutočne pozitívnych javov z celkového počtu 14 pozitívnych v datasete. Čiže našiel všetky – 100 percent. Zvyšných 11 javov bolo falošne pozitívnych. Vytvorili sme kontingenčnú tabuľku 4 – 1 pre množinu, na ktorej som testovali.

Tabuľka 4 – 1 Nami vytvorená kontigenčná tabuľka pre výsledky z testovacej množiny

		predikované hodnoty	
		Event	**background FP**
skutočné hodnoty	**Event**	14	11
	background FN	0	

Hodnoty, ktorá obsahuje kontingenčná tabuľka a použili sme ich aj vo Vzorcoch 4.4 a 4.5 sú:

- **TP** je skratka pre *True Positive*. Ide o skutočne pozitívne prípady. U nás ide

o prípady, kedy model predikoval, že označená časť obrázka je TLE jav a v skutočnosti to TLE jav bol. Takýchto prípadov bolo 14.

- **TN** je skratka pre *True Negative*. Ide o skutočne negatívne prípady. Ide o prípady, kedy model neoznačil, že sa na obrázku nachádza TLE jav a naozaj sa tam nenachádzal. Takéto prípady sme nemali, keďže sme označovali len jednu triedu.

- **FP** je skratka pre *False Positive*. Ide o falošne pozitívne prípady. U nás ide o prípady, kedy model označil na obrázku anotáciu, že ide o TLE jav ale v skutočnosti tam jav nebol. Takýchto prípadov sa vyskytlo 11 z 514 obrázkov.

- **FN** je skratka pre *False Negative*. Ide o falošne negatívne prípady. Ide o prípad, kedy model nenašiel nejaký TLE jav, ktorý tam v skutočne je. U nás sa táto možnosť nevyskytla, čiže hodnota je 0 z 514.

Z výsledku sme vypočítali metriky, ktoré môžete vidieť v Tabuľke 4 – 2.

Tabuľka 4 – 2 Vyhodnotenie metrík, finálneho modelu na testovacej množine, pri hodnotách iou=0.3 a conf=0.45

Obrázky	**Javy**	**Presnosť**	**Návratnosť**	**F1 skóre**
514	25	0.56	1	0.718

Metriky Presnosť a Návratnosť vám už boli predstavené. Ostáva len metrika F1 skóre. Cieľom F1 skóre je spojiť metriky presnosť a návratnosť do jednej metriky. Skóre F1 je definované ako harmonický priemer presnosti a návratnosti. Matematický vzorec pre výpočet F1 skóre môžete vidieť na Vzorci 4.6.

$$F1\ skóre = 2 * \frac{Presnosť * Návratnosť}{Presnosť + Návratnosť} \tag{4.6}$$

Keďže skóre F1 je priemerom *Precision* a *Recall*, znamená to, že skóre F1 má rovnakú váhu ako *Precision* a *Recall*:

- **Vysoké F1 skóre** ak hodnoty presnosti aj návratnosti sú vysoké.

- **Stredné F1 skóre** ak jedna z hodnôt je vysoká a druhá nízka.
- **Nízke F1 skóre** ak hodnoty presnosti aj návratnosti sú nízke.

5 Záver

Problém, ktorý bol na začiatku zadaný Doc. RNDr. Jurajom Tóthom, PhD. sa podarilo úspešne vyriešiť. Je zostavený program na automatickú detekcie TLE eventov. Taktiež je skompletizovaný doposiaľ jediný dataset s obrázkami, ktorý obsahuje TLE eventy.

Počas práce na tejto úlohe sme prekonali veľa prekážok. Prvou najväčšou bolo zistenie, aký malý počet pozitívnych dát sme získali z prvotného datasetu z roku 2020. Obrázkov bolo len 142, čo nepostačovalo na dokonalé naučenie neurónovej siete. Ak bol dataset rozdelený ešte aj to tried podľa druhov TLE javov, teda troch tried, efektivita učenia sa rapídne zmenšila.

Zvolili sme teda spojenie všetkých troch tried do jednej, čo zlepšilo výsledky, ale takto sme prišli o hodnotný údaj – druh TLE blesku. Tento aspekt by sa dal v budúcnosti vylepšiť napríklad vytvorením umelých TLE javov, ktoré by boli vystrihnuté z pôvodných snímok a prilepené na iné snímky poprípade následnou klasifikáciou už detekovaných eventov pomocou klasifikačnej neurónovej siete. Po opakovanom získavaní a pridávaní dát z ďalších rokov sa počet dát v databáze zväčšil, no stále aj to je malý počet pre neurónovú sieť. Tá však aj napriek tomu dokázala nájsť TLE eventy s vysokou presnosťou, aj keď stále detekuje množstvo falošne pozitívnych eventov. Avšak prejsť ročne 2 až 3 tisíc obrázkov (čo je mesačne v priemere 250 obrázkov) je omnoho priateľnejšie riešenie ako ručne prehľadávať viac ako 50 tisíc obrázkov ročne (čo je mesačne v priemere 4200).

To, že finálny model nachádzal veľa falošne pozitívnych javov, je nevýhodou nášho riešenia. Veľa z nich bolo po okraji snímky teda na povrchu Zeme. Išlo väčšinou o statické objekty, ktoré sa pre počítačové videnie podobali na nami hľadané TLE javy. Tieto objekty, ktoré sa nachádzali na každej snímke z danej lokality sme odstránili v predspracovaní dát. Vyčiernili sme dané oblasti, a tak sme zlepšili efektivitu hľadania javov modelom. Z pôvodných 10 tisíc nájdených anotácií modelom sa počet zmenšil na 2 až 3 tisíc.

Zvyšné objekty, ktoré sa nepodarilo odfiltrovať sú dynamické objekty ako napríklad meteory či lietadlá a satelity. Vyskytujú sa stále na inom mieste a majú podobný tvar stĺpu, keďže sú tvaru čiarky. Tieto objekty vieme odfiltrovať jedine ručne. Ak za rok astronómovia nazbierajú 50 tisíc dát, z toho model označí 3 tisíc dát v prepočte je to 8 snímok na deň. Astronóm, ktorý už za pár sekúnd zistí, či na obrázku je alebo nie je TLE event, prejde 8 obrázkov do 30 sekúnd. Bez tohto riešenia by astronóm musel prechádzať za deň 137 obrázkov, čo by mu trvalo približne 12 minút. Naše riešenie prináša zefektívnenie a zrýchlenie práce astronómov.

V detekčnom modeli je zaručené, že model nájde väčšinu TLE javov. Je ľahšie odfiltrovať falošne pozitívne javy, ako by sme mali stratiť javy, ktoré model neoznačil. Nájdené javy majú celkom vysoké hodnoty pravdepodobnosti. V testovacej množine sa hodnoty pravdepodobnosti pohybovali okolo 70 až 80 percent, čo označuje vysokú úspešnosť modelu. Model našiel v testovacej množine všetky TLE javy, čo taktiež svedčí o vynikajúcom fungovaní modelu.

Ďalším možným vylepšením existujúceho modelu by bolo pokračovanie v iteratívnom prehľadávaní ďalších rokov a pridávanie eventov do trénovania. Dáta sa dajú získať otestovaním datasetov z rokov, s ktorými sme v našej práci nepracovali. Keďže prvá AMOS kamera bola nainštalovaná v roku 2007, je ďalších osem datasetov s približným počtom 50 tisíc obrázkov na rok, čo je dokopy 400 tisíc surových dát. Z tých by sa dalo získať minimálne ďalších 400 TLE javov, ktoré by bolo vhodné pridať do trénovania a zlepšiť výsledky modelu. Takisto každým nasledujúcim budúcim rokom, by bolo potrebné dáta pridávať, a tým zabezpečiť neustále sa zlepšovanie modelu.

Druhou možnosť je pridať do datasetu dáta zo zahraničných AMOS kamier. Keďže predspracovanie dát je špecificky robené na každú lokáciu kamery, bolo by potrebné pridať vyčiernovanie aj pre tieto kamery. Týmto spôsobom by bolo možné získať veľké množstvo dát.

Tretou možnosťou ako získať viac dát, je už vyššie spomínanie umelé vytvorenie. Dali by sa vystrihnúť TLE eventy z pôvodných obrázkov a prilepiť ich na iné snímky.

Porozmiestňovať ich okolo, kde by sa reálne mohli nachádzať, a takto zväčšiť veľkosť trénovacej množiny. Otázkou je, či by tieto dáta boli užitočné. Možno by mali práve opačný účinok na efektivitu modelu kvôli tomu, že by boli neprirodzené a model by hľadal viac chýb.

Výsledok našej práce poputuje späť ku astronómovi doc. RNDr. Jurajovi Tóthovi, PhD., ktorému uľahčí každodennú prácu a ušetrí čas. Skvalitnenie modelu a pridanie ďalších dát do trénovacej množiny by bolo vhodné na ďalšiu bakalársku či diplomovú prácu. Pri väčšom počte dát by bolo možné spätne dáta rozdeliť na druhy TLE eventov a zabezpečiť aby model vedel určiť aj triedu TLE.

Práca na tejto problematike autorke, priniesla veľa nových poznatkov. Od nových vedomostí vo vesmírnej problematike až po prácu s neurónovými sieťami a spracovaním dát. Získala skúsenosti s prácou s neurónovou sieťou YOLOv5. Prvýkrát mala možnosť pracovať na anotačnom projekte. Pri anotovaní dát s spolužiakmi si zlepšila prezentačné a vysvetľovacie vlastnosti. Spoznanie odborníkov v oblasti astronómie bolo pre ňu tiež hodnotnou skúsenosťou. Vzhľadom na širokú oblasť problematiky a časovú náročnosť tohto projektu vie získané vedomosti využiť ďalej v odbornej praxi.

Literatúra

Amrich, S., Mackovjak, Š., Strhárskỳ, I., Baláž, J. and Hančikovskỳ, M. (2021). Design and construction of hardware and software for autonomous observations of transient luminous events, *Journal of Instrumentation* **16**(12): T12016.

Ayodele, T. O. (2010). Types of machine learning algorithms, *New advances in machine learning* **3**: 19–48.

Blanc, E., Farges, T., Jehl, A., Binet, R., Hébert, P., Le Mer-Dachard, F., Ravel, K. and Sato, M. (2017). Taranis mcp: a joint instrument for accurate monitoring of transient luminous event in the upper atmosphere, *International Conference on Space Optics—ICSO 2012*, Vol. 10564, International Society for Optics and Photonics, p. 1056404.

Brabham, D. C. (2008). Crowdsourcing as a model for problem solving: An introduction and cases, *Convergence* **14**(1): 75–90.

Chollet, F. (2021). *Deep learning with Python*, Simon and Schuster.

Donalek, C. (2011). Supervised and unsupervised learning, *Astronomy Colloquia. USA*, Vol. 27.

Du, J. (2018). Understanding of object detection based on cnn family and yolo, *Journal of Physics: Conference Series*, Vol. 1004, IOP Publishing, p. 012029.

Ertel, W. (2018). *Introduction to artificial intelligence*, Springer.

Franz, R., Nemzek, R. and Winckler, J. (1990). Television image of a large upward electrical discharge above a thunderstorm system, *Science* **249**(4964): 48–51.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V. and Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation.

Gavali, P. and Banu, J. S. (2019). Deep convolutional neural network for image classification on cuda platform, *Deep learning and parallel computing environment for bioengineering systems*, Elsevier, pp. 99–122.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*, MIT Press. `http://www.deeplearningbook.org`.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J. et al. (2018). Recent advances in convolutional neural networks, *Pattern Recognition* **77**: 354–377.

Howe, J. et al. (2006). The rise of crowdsourcing, *Wired magazine* **14**(6): 1–4.

Jocher, G. (2020). Yolov5 documentation.
URL: *https://docs.ultralytics.com/*

Khan, S., Rahmani, H., Shah, S. A. A. and Bennamoun, M. (2018). A guide to convolutional neural networks for computer vision, *Synthesis Lectures on Computer Vision* **8**(1): 1–207.

Kim, S., Nam, T. and Jung, D. (2018). Experimental validation of an onboard transient luminous events observation system for visioncube via ground simulation environment, *Aerospace* **5**(4): 100.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning, *nature* **521**(7553): 436–444.

Malik, T. (2020). European vega rocket suffers major launch failure, satellites for spain and france lost.
URL: *https://www.space.com/vega-rocket-launch-anomaly-november-2020*

Maslej-Krešňáková, V., Kundrát, A., Mackovjak, Š., Butka, P., Jaščur, S., Kolmašová, I. and Santolík, O. (2021). Automatic detection of atmospherics and tweek

atmospherics in radio spectrograms based on a deep learning approach, *Earth and Space Science* **8**(11): e2021EA002007.

Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N. and Terzopoulos, D. (2021). Image segmentation using deep learning: A survey, *IEEE transactions on pattern analysis and machine intelligence* .

Nguyen, D. T., Nguyen, T. N., Kim, H. and Lee, H.-J. (2019). A high-throughput and power-efficient fpga implementation of yolo cnn for object detection, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **27**(8): 1861–1873.

Ongsulee, P. (2017). Artificial intelligence, machine learning and deep learning, *2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE)*, IEEE, pp. 1–6.

Padilla, R., Netto, S. L. and Da Silva, E. A. (2020). A survey on performance metrics for object-detection algorithms, *2020 international conference on systems, signals and image processing (IWSSIP)*, IEEE, pp. 237–242.

Pasko, V. P. (2010). Recent advances in theory of transient luminous events, *Journal of Geophysical Research: Space Physics* **115**(A6).

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016). You only look once: Unified, real-time object detection, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I. and Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 658–666.

Tóth, J., Kornoš, L., Zigo, P., Gajdoš, Š., Kalmančok, D., Világi, J., Šimon, J., Vereš, P., Šilha, J., Buček, M. et al. (2015). All-sky meteor orbit system amos

and preliminary analysis of three unusual meteor showers, *Planetary and space science* **118**: 102–106.

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C. and Xu, W. (2016). Cnn-rnn: A unified framework for multi-label image classification, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2285–2294.

Yap, M. H., Hachiuma, R., Alavi, A., Brüngel, R., Cassidy, B., Goyal, M., Zhu, H., Rückert, J., Olshansky, M., Huang, X. et al. (2021). Deep learning in diabetic foot ulcers detection: a comprehensive evaluation, *Computers in Biology and Medicine* **135**: 104596.

Zhao, Z.-Q., Zheng, P., Xu, S.-t. and Wu, X. (2019). Object detection with deep learning: A review, *IEEE transactions on neural networks and learning systems* **30**(11): 3212–3232.

Zoznam príloh

Príloha A CD médium – záverečná práca v elektronickej podobe, príručky v elektronickej podobe a zdrojový kód

Príloha B Používateľská príručka

Príloha C Systémová príručka