VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2022 Bc. Patrik Židovský

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

APLIKOVÁNÍ TECHNIK HLUBOKÉHO UČENÍ NA DETEKCI ANOMÁLIÍ V POČÍTAČOVÝCH SÍTÍCH POMOCÍ GRAFICKÉ REPREZENTACE PROVOZU

APPLICATION OF DEEP LEARNING TECHNIQUES FOR ANOMALY DETECTION IN COMPUTER NETWORKS USING GRAPHICAL REPRESENTATION OF NETWORK TRAFFIC

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Patrik Židovský

AUTHOR

VEDOUCÍ PRÁCE — Ing. Yehor Safonov

SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. Patrik Židovský ***ID:*** 203721

Ročník: 2 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Aplikování technik hlubokého učení na detekci anomálií v počítačových sítích pomocí grafické reprezentace provozu

POKYNY PRO VYPRACOVÁNÍ:

Hlavním cílem diplomové práce je návrh a implementace modelů moderních konvolučních neuronových sítí pro detekci anomálií v síťovém provozu. V teoretické části práce nastudujte problematiku umělé inteligence se zaměřením na srovnání moderních konvolučních sítí a následně analyzujte možnosti vizualizace síťového provozu sloužícího k detekci kybernetických útoků. Z výsledků analýzy vyberte vhodné atributy síťového provozu k vizualizaci, proveďte jejich transformaci a napojení na dva modely konvolučních neuronových sítí. První model bude integrován na zařízení RaspberryPi a poslouží pro základní detekci anomálií. Pro urychlení výpočtů modelu použijte hardwarovou akceleraci v podobě Neural Compute Stick. Druhý model bude integrován na výpočetně výkonné zařízení (centrála) a poslouží pro precizní kategorizaci anomálií detekovaných zařízením RaspberryPi. V praktické části využijte veřejně dostupnou datovou sadu. Pro testování zvolte nejméně tři kybernetické útoky a otestujte úspěšnost vybraného modelu v experimentálním pracovišti. Ověřte výkonnostní limity naimplementovaného řešení a diskutujte možnosti optimalizace.

DOPORUČENÁ LITERATURA:

[1] PANG, GUANSONG, CHUNHUA SHEN, LONGBING CAO a ANTON VAN DEN HENGEL. Deep Learning for Anomaly Detection: A Review [online]. 2020 [cit. 2020-09-11]. Dostupné z: https://arxiv.org/abs/2007.02500

[2] CHOLLET, Francois. Deep learning with Python. Shelter Island, New York: Manning Publications Co., [2018]. ISBN 1617294438.

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 24.5.2022

Vedoucí práce: Ing. Yehor Safonov

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Táto diplomová práca sa zaoberá aplikáciou techník hlbokého učenia na detekciu anomálií v počítačových sieťach. Výberom vhodných vlastností komunikačnej siete bola vytvorená grafická reprezentácia sieťovej prevádzky za účelom trénovanie konvolučných neurónových sietí. Prvý natrénovaný model bol použitý v zariadení Raspberry Pi s hardvérovým akcelerátorom Neural Compute Stick. Druhý model bol umiestnený v centrále pre dodatočnú kontrolu výsledkov. Cieľom práce bolo navrhnúť a implementovať automatizovaný systém detekcie anomálií, ktorý bude otestovaný tromi zvolenými kybernetickými útokmi. Vyhodnotiť získané výsledky a navrhnúť možnosti optimalizácie.

KĽÚČOVÉ SLOVÁ

analýza, detekcia anomálií, kybernetické útoky, raspberry pi, sieťová komunikácie, umelá neurónová sieť, vizualizácia

ABSTRACT

This thesis deals with the application of deep learning techniques for anomaly detection in computer networks. By selecting appropriate features of the communication network, a graphical representation of the network traffic has been created in order to train convolutional neural networks. The first trained model was used in a Raspberry Pi device with a Neural Compute Stick hardware accelerator. The second model was placed in a central location for additional control of the results. The aim of this work was to design and implement an automated anomaly detection system to be tested by three selected cyber attacks. Evaluate the results obtained and propose optimization options.

KEYWORDS

analysis, anomaly detection, cyber attacks, raspberry pi, network communication, artificial neural network, visualization

ŽIDOVSKÝ, Patrik. *Aplikování technik hlubokého učení na detekci anomálií v počítačových sítích pomocí grafické reprezentace provozu*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačných technologií, Ústav telekomunikací, 2022, 83 s. Diplomová práca. Vedúci práce: Ing. Yehor Safonov

Vyhlásenie autora o pôvodnosti diela

Meno a priezvisko autora:	Bc. Patrik Židovský
VUT ID autora:	203721
Typ práce:	Diplomová práca
Akademický rok:	2021/22
Téma závěrečnej práce:	Aplikování technik hlubokého učení na detekci anomálií v počítačových sítích pomocí grafické reprezentace provozu

Vyhlasujem, že svoju záverečnú prácu som vypracoval samostatne pod vedením vedúcej/cého záverečnej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej záverečnej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto záverečnej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podpisuje iba v tlačenej verzii.

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce, pánovy Ing. Safonovi, za odborné vedenie, konzultácie, trpezlivost a konštruktívne návrhy k práci.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov

Úvod

Kažký deň sa objavujú nové zraniteľnosti zneužívané útokmi nultého dňa *(zero-day exploits)*, majúce devastujúce účinky pre sieťovú bezpečnosť. Za účelom zabránenia vzniku negatívnych javou v sieti boli vyvinuté nástroje ako firewall, systém pre odhalenie prienikov *(Intrusion Detection Systems)*, honeypot, atď. Najbežnejšie používané zariadenie na detekciu útokov a neautorizovaných prístupov je systém pre odhalenie prienikov(ďalej len ako IDS). IDS je efektívna technika na zvýšenie bezpečnosti v komunikačných sieťach ale aj bezpečnosť užívateľov a ich dát. Existujú dva typy IDS systémov. Prvý je založený na princípe signatúr známych útokov. Najväčším nedostatkom metódy je neprítomnosť prispôsobivosti pre rozličné scenáre a detekciu neznámych útokov. Výsledkom je veľmi veľa falošne negatívnych prípadov. Druhý typ je založený na detekovaní anomálií. Princíp fungovanie je založený na vytváraní profilov normálneho chovanie. Všetko správanie, čo sa od profilov líši je označené ako nenormálne a vyžaduje dodatočnú analýzu. Týmto spôsobom sú odhaľované nové typy útokov ale je zaznamenávaný veľký počet falošne pozitívnych prípadov. V posledných rokov začali vznikať IDS založené na detekcií anomálií pomocou strojového učenia.[1]

Táto práca sa bude zaoberať aplikovaním konvolučnej neurónovej siete na detekciu anomálií v počítačových sieťach pomocou grafickej reprezentácie prevádzky. Použitie sieťovej komunikácie na objavenie útokov je sľubné, pretože útočníci zahajujú útoky väčšinov cez internetovú sieť. Hlavným cieľom detekcie anomálií je dosiahnuť, čo najvyššej presnosti a čo najnižší falošne pozitívnych a negatívnych prípadov. Prvá časť tejto práce bude venovaná vysvetleniu základných techník a pojmov. V prvom rade budú spomenuté spôsoby zachytávania sieťovej komunikácie a nástroje, k tomuto účelu používané. Nasledované charakteristikou neurónových sietí od jej najmenšej časti nazývanej neurón, spoločne so všetkých operácií v ňom vykonávané. Spomenuté budú najpoužívanejšie typy arhitektúr a ich zloženie vrstiev z pohľadu vstupných dát, ktoré sú umelou neuonovou sieťou spracovávané. Každá neurónová sieť je hodnotená z pohľadu jej úspešnosti viacerými metrikami, ktoré majú jedinečnú výpovednú hodnotu. Preto budú popísané z oblasti účelu a spôsobu výpočtu. Ako posledné budú popísané tri generalizované hlboké neurónové siete. Tie obsahujú jedinečné zloženie vrstiev a prepojení medzi nimi, vytvárajúce unikátne štruktúry, ktorých výhody sú rozobrané z pohľadu jednotlivých neurónových sietí. Po teoretickej časti bude vytvorená rešerš štyroch moderných prístupov k riešeniu detekcie anomálií hlbokou neurónovou sieťou. Každá z nich zachytáva odlišnú oblasť, v ktorej môžu byť anomálie detekované. To bude zjavné z použitých dátových sád. Primárne sú rozoberané techniky transformácie dátovej sady na obrázky a dosiahnuté presnosti so zvolenými neurónovými sieťami.

Druhá časť bude venovaná praktickému riešeniu problematiky, ktorá bude rozdelená na tri skupiny. V prvej skupine sú popísané základné komponenty, potrebné na vykonávanie detekcie. Prvým z nich bude výber dátovej sady a definícia troch prístupov transformácie dát na obrázky. Každý z prístupov má odlišné zloženie vlastností sieťovej komunikácie. Následne použitím troch vopred vytvorených dátových sád, pozostávajúcich z obrázkov, budú natrénované generalizované hlboké neurónové siete popísané v teoretickej časti. Počas učenia budú ukládané hodnotiace metriky rozšírené o hodnoty získané po aplikácií testovacích dát na naučené modely. Z hodnôt budú vytvorené matice zámien a ostatné metiky popísané v teoretickej časti, podľa ktorých bude porovnaná úspešnosť jednotlivých modelov. Posledným komponentom, potrebným na prípravu je nástroj `CicFlowMeter`. Nástroj v štandardnom stave je nevyhovujúci, preto budú uskutočnené zmeny v jeho funkčnosti. Tie zabezpečia možnosť prepojenia nástroja s hlavným programom, nazvaným sonda.

V nasledujúcej druhej skupine bude popísaná celková programová implementácie na detekciu anomálií, zložená so sondy a centrály. Na začiatku bude načrtnuté možné umiestnenie sondy a centrály v komunikačnej sieti, pokračované ich detailným popisom. Ako prvá bude popisovaná sonda a jej harvérové a softvérové požiadavky. Po výpise všetkých náležitostí sondy, potrebných na jej fungovanie je vysvetlený proces vytvárania predikcií v dvoch módoch. Prvý bude používaný na zistenie kompletnej funkčnosti programu a druhý bude používaný na predikovanie dát z dostupnej komunikačnej siete. Ako posledná, so sondou súvisiaca časť, bude popísaná súborová štruktúra programu spoločne s nástrojmi, určenými na správu súborov. Ako druhá bude popísaná centrála a to v rovnakých oblastiach ako aj sonda.

V poslednej skupine bude testovaný vytvorený systém na detekciu anomálií. Na začiatku bude popísané virtualizované experimentálne pracovisko s pohľadu dostupných staníc, spoločne s rozdelením a nastavením komunikačnej siete. Nasledne budú zvolené a uskutočnené tri kybernetické útoky, pričom budú zachytávané sondov na spracovanie. Po vyhodnotení výsledkov testovania budú navrhnuté rozšírujúce návrhy na zlepšenie.

1 Popis problematiky

1.1 Detekcia anomálií

Detekcia anomálií je proces, pri ktorom sú identifikované neobvýklé vzory, zriedkavé udalosti, netypické správanie alebo odchylky z tzv. normálneho chovania sa počítačovej siete. Preto sieťovú komunikáciu vieme rozdeliť na bežnú a nezvyčajnú prevádzku siete. Pri bežnej komunikácii sú v sieti prenášané iba pakety, pre ktoré je sieť určená. Vyvolanie anomálii v komunikácii môže byť spôsobené legitýmnimi udalosťami spôsobené zariadeniami (aktualizácia systémov) alebo zvýšením počtu požiadaviek od používateľov, z dôvodu zvýšeného záujmu (akcie). Tieto typy anomálií sú považované za bežné ale môže nastať stav, kedy v komunikačnej sieti sa nachádzajú pakety určené na narušenie správneho fungovania celej siete alebo jednotlivých prvkov nachádzajúcich sa v sieti.[2]

Automatická detekcia takýchto odchyliek je žiadúca a poskytuje správcom sietí nástroj pre získavanie dodatočných informácií na diagnostiku siete a nájdenie zdroja pozorovaného chovania.[2]

Výzkum v oblasti detekcie anomálií sa zvyčajne riadi prístupom, rozdeleným do troch štádií. Prvé tri štádia definujú metódy detekcie, pričom posledné štádium je venované validácií prístupu. Prvý krok je zachytenie požadovanej sieťovej komunikácie, určenej na analýzu (zhromaždovanie dát). V druhom kroku vytvorené dáta podstupa analýzu, na základe ktorej sú vyextrahované najrelevantnejšie vlastnosti (analýza dát). Extrahované vlastnosti sú následne v treťom kroku rozdelené minimálne do dvoch skupín, a to na normálnu a neobvyklú komunikáciu (klasifikácia). Počet a druh skupín sa môže líšiť a zavysí od cieľa danej analýzy. Posledné štádium je určené na validáciu celého postupu analýzy, zistenia možných nedostatkov pri voľbe a uplatnení postupu (validácia). Cieľom je vytvoriť postup, dosahujúci najlešie výsledky detekcie nežiadúceho stavu pri využití zvolených vlastností.[2]

1.2 Zachytávanie sieťovej prevádzky

Zachytávaním sieťovej prevádzky je postup, pri ktorom sú zachýtávané IP pakety vo zvolenej dátovej sieti za účelom ich preskúmania a analýzy. Pre tento účel sú vytvorené softvérové nástroje, ktorých výstup je zvolená sieťová komunikácia, uložená zvyčajne vo formáte PCAP[1]. Takto vytvorené súbory sú výborným pomocníkom pre sieťových administrátorov na odhaľovanie alebo skúmanie bezpečnostných hrozieb retrospektívne alebo ešte počas ich priebehu.

[1]Viac informácií je možné nájst na: `https://www.reviversoft.com/file-extensions/pcap`.

Zachytávanie paketov sa radí do skupiny pasívnych techník, pri ktorých sú dáta iba zberané a nie odosielané. Preto je táto technika zložito vystopovateľná, ak je používaná nepovolenými osobami. Z toho dôvodu ju využívajú taktiež aj útočníci, ktorý týmto spôsobom sú schopný kradnúť heslá a iné citlivé údaje. [3]

1.2.1 Spôsoby zachytávania komunikácie

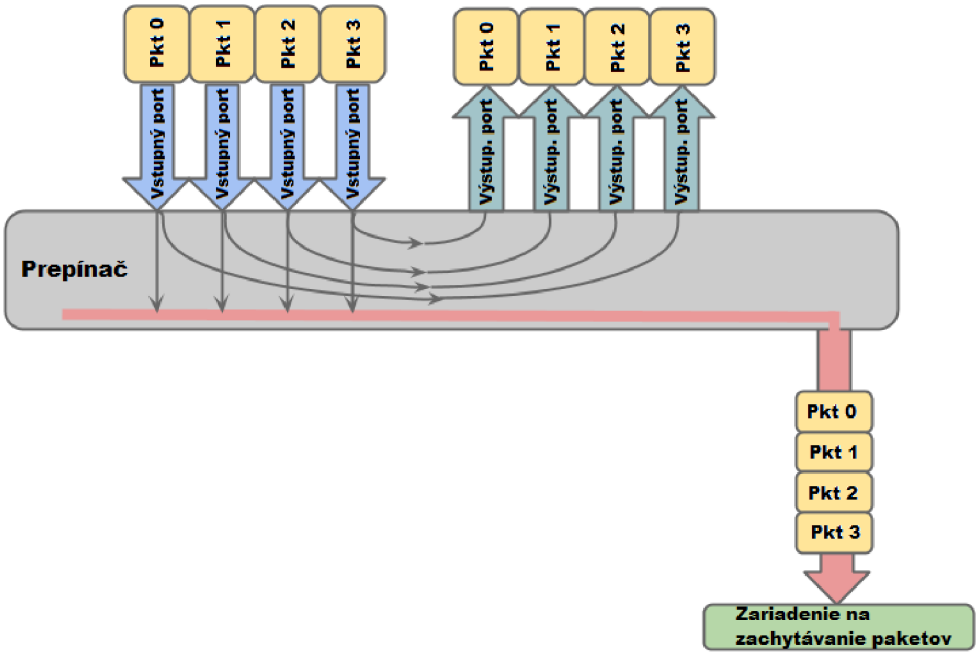
Každý spôsob zachytávania funguje na princípe vytvárania kópie vopred určených alebo všetkých paketov, prechádzajúcich v danom čase monitorovanov časťov sieťe. Preto je veľmi dôležité umiestnenie zariadenia a počet paketov prechádzakúcich cez zvolené miesto.

1. Najjednoduchším spôsobom zachytávanie je použitie súkromného počítača alebo notebooku. Na týchto zariadeniach sú sieťové rozhrania predvolene nastavené na monitorovací režim. Tento režim zabezpečuje zahadzovanie paketov, ktoré nie sú určené pre dané zariadenie. Pre prijímanie všetkých prichádzájúch paketov, je potrebné prepnúť dané rozhranie do promiskuitného režimu. Tento spôsob je obmedzený iba na komunikáciu, prechádzajúcu cez port prepínača, ku ktorému je zariadenie pripojené.[3]
2. Routre alebo prepínače majú funkcie nazývané zrkadlenie portov *(port mirroring)*, monitorovanie portov *(port monitoring)* alebo analýzer portov prepínača *(SPAN*[2] *– switch port analyzer)*, ktoré dovoľujú administrátorovi duplikovať sieťovú komunikáciu prechádzajúcu cez zariadenie a poslať ju na špecifikovaný port. Do portu je zvyčajne pripojené dedikované riešenie zachytávania paketov, ktoré je zobrazené na Obr. 1.1. [3]
3. Ak je zachytávanie vykonávané vo veľkých a rušných sieťach, najlepšou možnosťou je použitie prispôsobeného zariadenia, cez ktoré komunikácia prechádza ale zároveň sa zaznamenáva (anglicky označovaný ako *TAP*). Riešenie je, v porovnáni s ostatnými riešeniami cenovo najnáročnejšie, ale poskytuje potrebný výpočený výkon aby pripustnosť komunikačnej siete nebola degradovaná. Umiestnenie v sieti je zobrazené na Obr. 1.2.[3]

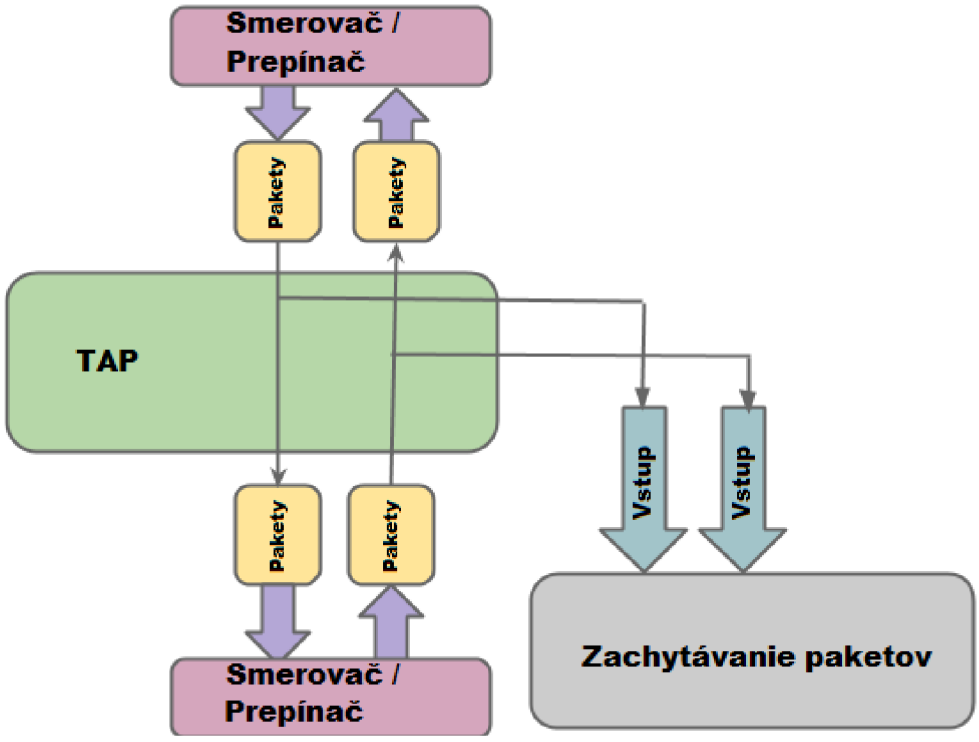
1.2.2 Nástroje pre analýzu prevádzky

Základnou súčasťou, každého nástoja sú knižnice umožnujúce prijímať sieťové rámce a poskytnúť ich používateľovi na ich zobrazenie, uloženie alebo úpravu. Medzi najznámejšie patria libpcap, winpcap a npcap. Knižnica libpcap vznikla ako

[2]Viac informácií je možné nájst na: `https://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/10570-41.html`.



Obr. 1.1: Zapojenie dedikovaného zariadenia na zachytávanie paketov do SPAN portu.



Obr. 1.2: Zapojenie dedikovaného zariadenia vo veľkých a rušných sieťach.

súčasť nástroja TCPdump, ktorý umožnuje prácu s paketmi na variáciách operačného systému Linux. Knižnice winpcap a npcap vznikli ako odpoveď na knižnicu libpcap a úmožnujú pracovať s paketmi na operačnom systéme Windows. [4, 5, 6]

Nástroje, používajúce spomínané knižnice sú:

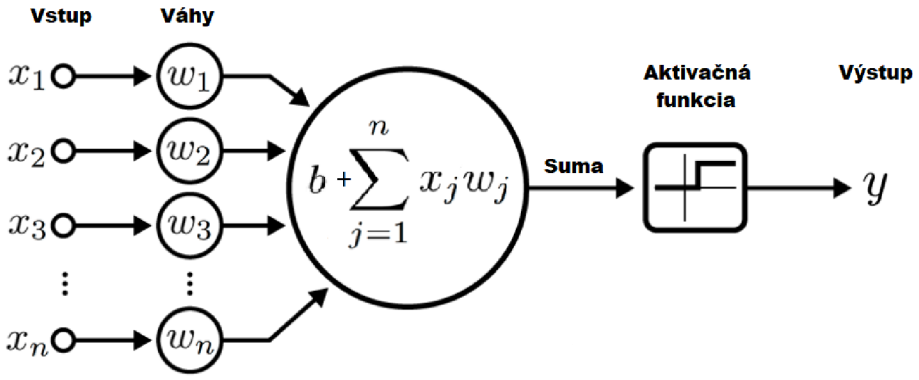
1. `Wireshark` je analyzátor sieťových protokolov alebo aplikácia na zachytávanie paketov zo sieťovej komunikácie. Ako jeden z najpoužívanejších nástrojov na analýzu paketov, umožnuje vytváranie filtrov dát na základe informácií nachádzajúcich sa vo vnútri paketov. Každý paket rozdelí na jednotlivé vrstvy TCP/IP modelu a odpovedajúce parametre pre každú vrstvu zvlášť. Ak je potrebné, `wireshark` dovoľuje graficky vizualizovať celkovú komunikáciu alebo jednotlivé sieťové toky.[7]
2. `TCPdump` je nástroj pracujúci cez príkazový riadok na operačnom systéme Linux. Rovnako ako `wireshark` umožňuje filrovať aktuálny alebo vopred zachytený sieťový prenos a výstup zobraziť alebo uložiť do súboru.[8]
3. `CICFlowMeter` je `open-source` analyzátor sieťového provozu. Dokáže vytvoriť viac ako 80 prevádzkových vlastností zvlášť pre jednotlivé relácie. Základné extrahované vlastnosti sú: identifikátor relácie, zdrojová IP adresa, cieľová IP adresa, zdrojový port, cieľový port a protokol. Medzi rozšírené vlastnosti patrí napr.: trvanie toku, počet paketov, veľkosť paketov, atď. Nástroj ponúka možnosť rozšírenia extrahovaných vlastností pridaním vlastných. Výstup je súbor vo formáte CSV, v ktorom sú jednotlivé vlastnosti označené a pripravené na analýzu.[9]

1.3 Umelé neuronové siete

Neurónové sieťe, taktiž známe ako umelé neurónové siete sú podmnožinou strojového učenia a základnou súčasťou algoritmou hlbokého učenia. Základnou inšpiráciou pre vznik neurónových sietí je ľudský mozog a napodobnenie komunikácie medzi neurónmi.[10]

Neurón

Neurón je základná stavebná jednotka neurónovej siete. Skladá sa zo vstupných spojení, jadra a výstupneho spojenia. Vstupné spojenia pozostávajú z výstupných hodnôt predchádzajúcej vrstvy a váh. Úlohou jadra je sčítať hodnoty na vstupe spolu s hodnotou bias. Výsledná hodnota je následne upravená na základe aktivačnej funkcie a predaná na výstup pre ďalšie spracovanie. Schéma neurónu je znázornený na Obr. 1.3.[10]



Obr. 1.3: Schéma neurónu.

Vstup

Pozostáva zo skupiny hodnôt (označovaných ako x_i) prichádzajúcich na vstup neurónu. Sú tvorené výstupom z neurónov predchádzajúcej vrstvy alebo hodnotami dát prichádzajúcich na vstup neurónovej siete.[13]

Váhy

Každé vstupné spojenie do neurónu má pridelenú váhovú hodnutu w_i. Hlavnou úlohou váh je pridelenie väčšej dôležitosti niektorým vstupom. Pridelenie väčšej váhy spojenie zabezpečí, že podiel vstupu na hodnote výstupu bude taktiež väčší. Pri prvotnej inicializácií sú váhy nastavené náhodne a počas procesu učenia sú postupne upravované tak, aby bolo dosiahnutého čo najlepšieho výsledku.[13]

Bias

Je konštantná hodnota b pripočítavaná k výstupu z jadra pred aplikovaním aktivačnej funkcie. Bias nie je povinná hodnota, ale pri jej použití je hodnota posunutá v kladnom alebo zápornom smere. Tým je zaručená zmena výstupnej hodnoty z aktivačnej funkcie a celková úspešnosť učenia.[13]

Aktivačné funkcie

Je využívaná ako rozhodovací základ pre výstup z neurónu. Na základe výberu aktivačnej funkcie, výstupné hranice hodnôt môžu mať lineárny alebo nelineárny charakter. Zabezpečujú ešte jednu podstatnú funkciu a to normalizáciu výstupu z dôvodu možného vzniku príliš veľkých hodnôt po prechode viacerými vrstvami.[11]

Najčastejšie používané aktivačné funkcie sú:

- Sigmoid – Mapuje vstup na hodnoty medzi $< 0\,,\, 1 >$.
- Tanh – Mapuje vstup na hodnoty medzi $< -1\,,\, 1 >$.
- ReLU – Kladné hodnoty zostávajú rovnaké a záporné sú namapované na hodnotu 0.[12]

1.3.1 Architektúra neurónových sietí

Pri vytváraní neurónovej siete sú používané vrstvy. Tie sú zložené z totožných neurónov a ich počet može byť odlišný od počtu neurónov v susedných vrstvách. Kažká z vrstiev može mať odlišný charakter ale existuje ich najzákladnejšie delenie a to na vstupnú, skrytú a výstupnú vrstvu.

Vstupná vrstva

Definuje formát vstupných dát a tým typ neurónovej siete. Najčastejšie sú používané obrázky, CSV súbory ale taktiež text ale aj upravenú zvukovú stopu. Podmienkou je zabezpečiť konštantnú veľkosť a nemenný formát vstupných dát. Na veľkosti vstupných dát závisí počet neurónov, z ktorých sa vrstva bude skladať.[11]

Skrytá vrstva

Označenie udáva to, že druh vrstiev, z ktorých je zložená nemá vopred definovaný charakter a môže ich byť zoskupených nedefinovaný počet. Zloženie vrstiev, ich veľkosť a použíté neuróny sa líšia od požadovaného účelu ale existuje fundamentálna postupnosť vrstiev, ktoré by mali následovať za sebou pre dosiahnutie požadovaného výsledku.[11]

Výstupná vrstva

Výstupom vrstvy sú predikované hodnoty po prechode vstupných dát modelom. Hodnoty sú ovplivňované použitím aktivačnej funkcie. Jej výber udáva formát výstupných hodnôt. Pre binárnu klasifikáciu je použitá funkcia Sigmoid alebo funkciu Softmax na viac-triednu klasifikáciu.[11]

1.3.2 Typy neurónových sietí

Neurónové siete sú delené podľa účelu, na ktorý sú určené. Tri najzákladnejšie a najviac sa vyskytujúce sú dopredné, konvolučné a regresívne neurónové siete.

Dopredné neurónové siete

Umelá, taktiež známa ako dopredná neurónová sieť je najpoužívanejšia a najzákladnejšia štruktúra usporiadania neurónov. Dopredná je nazývaná preto, že výstup z neurónov je použitý iba ako vstup do neurónu nasledujúcej vrstvy. Podľa počtu vstiev sa delia na jednovrstvé a viacvrstvé. Štruktúra jednovstvej siete je tvorená iba vstupnov a výstupnov vrstvou ale vstupná nie je započítavaná do celkového počtu, pretože v nej nie sú vykonávané žiadne matematické operácie. Vstupné dáta sú upravené odpovedajúcimi váhami a funkciou výstupných neurónov na valídny výstup zo siete. Pri pridaní minimálne jednej skrytej vrstvy sa model zmení na viacvrstvý. V tomto modeli nemusia byť neuróny medzi kažkou vstvou prepojené úlpne, t.j. kažký z kažkým, ale niektoré spoje môžu byť úmyselne vynechané, ak je to vyžadované. Pri vynechaní spojení sa model nazýva čiastočne prepojený *(partially connected)* a model bez vynechaných spojení je plne prepojený *(fully connected)*.[15]

Najväčšou výhodov doprednej neuróvovej siete je možnosť učenia pre rôzne nelineárne funkcie. To je dosiahnuté používaním aktivačných funkcií, ktoré dokážu vytvoriť komplexný vzťah medzi vstupom a výstupom.[14]

Konvolučné neurónové siete

Na rozdiel od doprednej neurónovej siete, ktorá sa považuje za univerzálnu, konvolučná sieť je určená na kvasifikáciu, spojenú s počítačovým videním. Poskytuje rozširovanelný prístup ku rozpoznávaniu objektov použitím princípov z násobenia matíc na identifikáciu vzorov v obrázkoch. Prístup je výpočetne veľmi náročný a vyžaduje grafické karty na trénovanie modelov.[16]

Použiteľné vrstvy sú rozšírené o tri základné druhy:

- **Konvolučná vrstva** *(convolutional layer)* – vykonáva výpočty spojené s hľadaním vzorov. Vyžadované sú vstupné dáta a filtre. Vstupné dáta sú zvyčajne v trojrozmernom tvare (výška, dĺžka a hĺbka) ako RGB obrázok a filtre, ktoré postupne prechádzajú dáta až kým nie je skontrolovaný celý obrázok. Filtre sú definované ich veľkosťou, veľkosťou posunu a typom výplne. Vybraná časť dát pomocou filtru je upravená váhovou maticou na jednomiestné číslo a pridané do výstupnej matice.[16]
- **Združovacia vrstva** *(pooling layer)* – uskutočňuje redukciu rozmerov. Funguje podobne ako konvolučná vrstva s tým rozdielom, že nepoužíva váhovú maticu ale filter ale aplikuje agregačnú funkciu na vybrané hodnoty. Najpoužívanejšie sú združovanie maximálnej hodnoty *(max pooling)* voliaci najväčšiu hodnotu a združovanie priemernej hodnoty *(average pooling)* počítajúci priemernú hodnotu z hodnôt.[16]

- **Plne prepojená vrstva** *(fully connected layer)* – vykonáva úlohu klasifikácie na základe vlastností extrahovaných z konvolučnej a združovacej vrstvy.[16]

Rekurentné neurónové siete

Dopredná neurónová sieť používa ako vstup od seba nezávislé dáta. Tento prístup nie je vhodný ak je nutné spracovávať dáta sequenčne, tak že dátový prvok je závislý na predchádzajúcom prvku. Preto sú vrstvy v sieti upravené tak, aby začlenili väzby medzi jednotlivými dátovými prvkami.[18]

Existujú tri varianty rekurentných neurónových sietí:

- BRNN – dvojsmerná rekurentná neurónová sieť *(Bidirectional Recurrent Neural Networks)* sa líši od jednosmernej tým, že na predikciu nepoužíva iba dáta použité pred prvkom ale aj dáta, ktoré len použité budú.[17]
- LSTM – dlhá krátkodobá pamäť *(Long Short-Term Memory)* implementuje do skrytých vrstiev tzv. bunky, ktoré obsahujú vstupnú bránu, výstupnú bránu a zabúdaciu bránu. Tieto brány kontolujú prúd informácií, potrebných na predikciu výstupu. Informácie ukladané do buniek sú vyberané z viacerých predchádzajúcich vstupov.[17]
- GRUs – Uzavreté rekurentné jednotky *(Gated Recurrent Units)* pristupuju k rovnakému problému ako LSTM ale namiesto stavu bunky používa skryté stavy a namiesto troch brán používa iba dve. Tie sú nazývané ako resetovacia a aktualizačná brána a kontrolujú koľko a ktoré informácie ponechať.[17]

1.3.3 Metriky hodnotenia presnosti modelu

Voľba modelu je závislá na hodnotiacich kritériách, podľa ktorých je posudzovaná jeho výkonnosť a spoľahlivosť. Veľkosť týchto hodnôt je požadované poznať pri procesu učenia a taktiež aj pri predikcií na reálnych dátach. Na výpočet stavu počas učenia je používaná strátová funkcia a celková správnosť. Pri hodnotení modelu na testovacích dátach je pridaná matica zámen, precíznosť, senzitivita a F1-miera.

Stratová funkcia

Stratová funkcia *(loss function)* počíta vzdialenosť medzi aktuálnym stavom modelu a stavom požadovaným. Najpoužívanejšou funciou pre výpočet je stredná kvadratická chyba *(MSE – mean square error)*, ktorej rovnica 1.1 je zobrazená nižšie.

Vypočítané hodnoty sú používané pri fáze učenia na zmenu váh pomocou gradientného zostupu *(gradient descent)*.[12]

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2 \tag{1.1}$$

Funkcia je veľmi senzitívna na odľahlé hodnoty a prideľuje im veľký význam. Preto ak je v dátach veľa šumu alebo náhodných nesúvysiacich hodnôt, výsledné MSE bude vysoké.[23]

Matica zámen

Matica zámen *(confusion matrix)* je sumarizácia predikovaných výsledkov na klasifikačný problem. Počty správnych a nesprávnych predikcií sú zoskupené aj s počtom hodnôt pre kažkú triedu. Pri binárnej klasifikácií sú hodnoty rozdelené do štyroch skupín, nazývaných skutočne pozitívny *(True Positive)*, skutočne negatívny *(True Negative)*, falošne pozitívny *(False Positive)* a falošne negatívny *(False Negative)*. Tento prístup umožňuje náhľad do chýb vytváraných modelom a určenie typu chyby.[24]

Celková správnosť

Celková správnosť *(accuracy)* je metrika, ktorou sa dá ihneď zhodnotiť, či model je trénovaný správne alebo aká bude jeho všeobecná výkonnosť. Na výpočet je použitý vzorec 1.2, udávajúci pomer medzi počtom správne predpovedaných a celkovým počtom vzorkov.

$$Celkovásprávnosť = \frac{správnepozitívny + správnenegatívny}{všetkyvzorky} \tag{1.2}$$

Nevýhodou metriky je možné skreslenie reálnej výkonnosti ak sú kalsifikované triedy nevyvážené. Tento problém riešia nasledujúce techniky.[19]

Precíznosť

Precíznosť *(precission)*, počitaný vzorcom 1.3, je používaný na vyjadrenie pomeru medzi správnymi pozitívnymi a celkovým počtom pozitívnych predpovedí.

$$Precission = \frac{správnepozitívny}{správnepozitívny + falošnepozitívny} \tag{1.3}$$

Tento spôsob je používaný ak cena výskytu falošne pozitívneho hodnotenia je príliš vysoká a môže vyvolať nežiadané udalosti.[19]

Senzitivita

Senzitivita *(recall)* udaná vzorcom 1.4 má podobné určenie ako metrika precíznosť ale pre falošne negatívny parameter. V niektorých prípadoch je nesprávna predikcia hodnoty ako negatívna nežiadúca. Preto výpočtom pomeru medzi správne pozitívnymi a súčtom správne pozitívnych a falošne negatívnych predpovedí je na danú chybu v predikcií poukázané. [19]

$$Recall = \frac{správnepozitívny}{správnepozitívny + falošnenegatívny} \tag{1.4}$$

F1-miera

F1 je celková metrika presnosti a výkonnosti modelu, kombinujúca dve predchádzajúce metriky. Kombináciou precíznosti a senzitivity, vyjadrenou vo vzorci 1.5, sú zohľadnené výskyty falošne pozitívnych a falošne negatívnych prípadov.

$$F1 = 2 \cdot \frac{precíznosť \cdot senzitivita}{precíznosť + senzitivita} \tag{1.5}$$

Ak sa hodnotá blíži k číslu 1, tak model identifikuje okolnosti správne a je nízka pravdepodobnosť výskytu nechcených poplachov.[19]

1.3.4 Generalizované hlboké neurónové siete

Pre prácu s neurónovými sieťami je nutné mať funkčný model. To môže byť docielené dvomi spôsobmi. Prvý je vlastnoručným poskladaním vrstiev v modeli a testovaním jeho úspešnosti. Alebo druhým spôsobom, pri ktorom je model vybraný z databázy známych a overených modelov. Dostupné sú aj dosiahnuté presnosti a definícia vstupnej a výstupnej vrstvy.

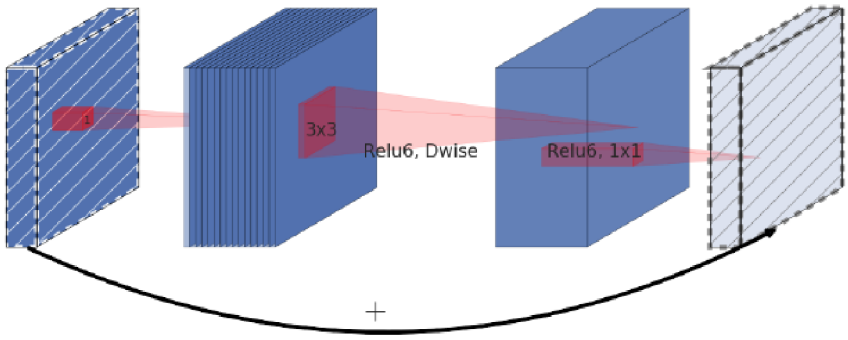
MobileNetV2

Neurónová sieť s názvom `MobileNetV2` vznikla za účelom sprístupnenia vysoko kvalitného počítačového videnie zariadeniam s obmedzenými výpočetnými prostriedkami. To je docielené zredukovaním počtu potrebných výpočetných operácií a zároveň znížením pamäťového zaťaženia pri zachovaní rovnakej výslednej presnosti.

Modul neurónovej siete, pomocou ktorej je dosiahnutý takýto výsledok sa nazýva `invertovaný zvyšok s lineárnym zúžením` *(The Inverted Residual With Linear Bottleneck*[3]*)*, ďalej iba ako `bottleneck`. Tento modul sa v neurónovej sieti vyskytuje sedemnásť krát a pozostáva z viacerých vrstiev, tvoriacich jeden celok.

[3]Viac informácií je možné nájst na: `https://towardsdatascience.com/mobilenetv2-inverted-residuals-and-linear-bottlenecks-8a4362f4ffd5`.

Vstupom do neurónovej siete je obrázok s troma kanálmi RGB. Ten je upravený pred vstupom do modulu `bottleneck` a to zmenšením prvých dvoch dimenzií na polovicu a rozšírením počtu kanálov na 32. To je docielené `hĺbkovo oddeliteľnov konvolúciou` *(Depthwise Separable Convolutions*[4]*)*, ktorá je taktiež súčasťou modulu `bottleneck`. Táto vrstva bola vytvorená práve pre model MobileNetV2 a jej výsledkom je drastická redukcia operácií, a to až o 8 až 9 násobok.



Obr. 1.4: Postupnosť vrstiev v module bottleneck.[20]

Tab. 1.1: Tabuľka MobileNetV2 vrstiev a ich parametrov

Vstup *(Input)*	Operátor *(Operator)*	Činiteľ zväčšenie *(Expansion factor)*	Kanály *(Channels)*	Opakovania *(Repetitions)*	Krok *(Stride)*
$224^2 \times 3$	Conv 2D	-	32	1	2
$112^2 \times 32$	Bottleneck	1	16	1	1
$112^2 \times 16$	Bottleneck	6	24	2	2
$56^2 \times 24$	Bottleneck	6	32	3	2
$28^2 \times 32$	Bottleneck	6	64	4	2
$14^2 \times 64$	Bottleneck	6	96	3	1
$14^2 \times 96$	Bottleneck	6	160	3	2
$7^2 \times 160$	Bottleneck	6	320	1	1
$7^2 \times 320$	Conv 2D	-	1280	1	1
$7^2 \times 1280$	Avgpool	-	-	1	-
$1 \times 1 \times 1280$	Conv 2D	-	k	1	-

[4]Viac informácií je možné nájst na: `https://towardsdatascience.com/understanding-depthwise-separable-convolutions-and-the-efficiency-of-mobilenets-6de3d6b62503`.

Modul `bottleneck` plní dve funkcie. Prvou funkciou je rozšírenie počtu kanálov podľa zvoleného činiteľa, aplikovať `hĺbkovú oddeliteľnú konvolúciu` a následne znížiť počet kánálov na požadovaný počet. Ako aktivačná funkcia je použitá `ReLU6`. Je to upravená aktivačná funkcia `ReLU` s pridanou hornou hranicou, ktorá je v tomto prípade hodnota 6. Takto vytvorené dáta sú pripravené na druhý krok. Počas neho je použítá metóda `invertovaných zvyškov`, ktorá spojí vstupné dáta s výstupnými. Proces je zobrazený na Obr. 1.4. Tento krok je nevyhnutný k vytváraniu modelov s vysokým počtom vrstiev.

Nastavenia modulu a počet opakovaní je odlišné. Líšia sa vstupnov a výstupnov veľkosťov, počtom kanálov, veľkosťov konvolučných krokov a činiteľom zväčšenia počtu kanálov. Nastavenie parametrov je zobrazené v Tab. 1.1. Parameter k udáva počet kanálov vo výstupnej vrstve neurónovej siete.[20]

MobileNetV3 Large

Ako nástupca neurónovej siete `MobileNetV2` vznikla sieť s názvom `MobileNetV3`. Jej hlavným prínosom je inovatívna architektúra vrstiev a využitie nástrojov ako hardvérovo orientované vyhľadávanie sieťovej architektúry *(Hardware-aware Network Architecture Search - NAS*[5]*)* a `NetAdapter`[6]. Oba vychádzajú s techniky posilneného učenia *(reinforcement learning)*, ktorého úlohov je nájdenie najoptimálnejšej konfigurácie neurónovej siete. Ako metrika v hodnotiacej funkcií je použitá presnosť modelu spoločne s latenciou pri prechode modelom. `MobileNetV3` používa ako základ vrstiev `invertovaný zvyšok s lineárnym zúžením`, do ktorej dola pridaná nové vrstvy so skupinovým názvom `stlačenie a excitácia` *(*Squeeze and Excitation[7]). Taktiež bola vytvorená nová aktivačná funkcia `h-swish`, ktorá je náhradov za funkciu `ReLU6`.

Aktivačná funkcia `swish`[8] bola vytvorená ako jednoduchá nelineárna náhrada za funkciu `ReLU`, ktorá výrazným spôsobom ovplivňuje presnosť neurónovej siete. Pri jej výpočte je využitá výpočetne náročná funkcia `sigmoid`, preto bola vytvorená obmena funkcie s názvom `h-swish`. Tá používa odlišnú funkciu, jednoduchšiu na výtočet, a to `ReLU6`. Vzťah pre výpočet uvádza časť 1.6.

$$h\text{-}swish = x \cdot \frac{ReLU6(x+3)}{6} \tag{1.6}$$

[5]Viac informácií je možné nájst na: `https://openaccess.thecvf.com/content_CVPR_2019/papers/Tan_MnasNet_Platform-Aware_Neural_Architecture_Search_for_Mobile_CVPR_2019_paper.pdf`.

[6]Viac informácií je možné nájst na:`https://openaccess.thecvf.com/content_ECCV_2018/papers/Tien-Ju_Yang_NetAdapt_Platform-Aware_Neural_ECCV_2018_paper.pdf`.

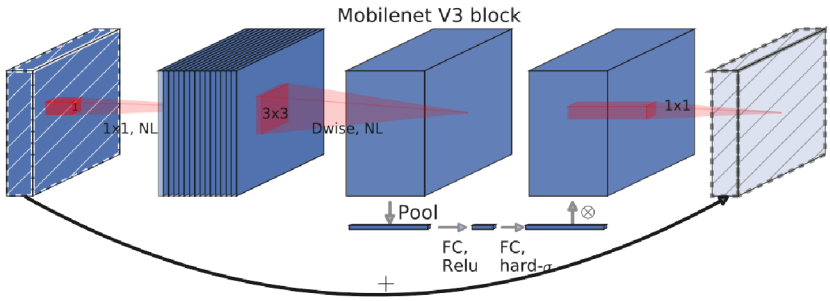
[7]Viac informácií je možné nájst na:`https://arxiv.org/pdf/1709.01507.pdf`.

[8]Viac informácií je možné nájst na:`https://arxiv.org/pdf/1702.03118.pdf`.

`Stlačenie a excitácia` je jednoduchý doplnok zobrazený na Obr. 1.5, ktorý je možné doplniť do konvolučnej neurónovej siete pre zvýšenie dosiahnuteľnej presnosti za cenu mierneho zvýšenia výpočetnej náročnosti. To je dosiahnuté adaptívným rekalibrovaním odozvy kanálov explicitným modelovaním vzájomných zavyslostí medzi kanálmi. Preto doležitejšie kanáli budú mať priradenú vyššiu váhu.

Pre dosiahnutie takéhoto stavu je zapotreby použiť 3 moduly:

- Stlačenie *(Squeeze Module)* - Tento modul je tvorený združovacou vrstvou, vytvárajúcou priemerné hodnoty. Tá príjíma ako vstup tenzor o veľkosti *(C, H, W)*, kde *C* vyjadruje počet kanálov, H výšku a W dĺžku obrázka. Po prechode obrázkov sa veľkosť zmení na *(C, 1, 1)*. To značí zredukovanie veľkosti každého kanálu na jednu zpriemerovanú hodnotu.
- Excitácia *(Excitation Module)* - Modul obsahuje 3 vrstvy. Prvá a posledná má veľkosť *C* a zastávajú úlohu vstupnej a výstupnej vrstvy. Stredná vrstva je plne prepojená s ostatnými a jej veľkosť je 16-násobne menšia ako *C*. Tým je zabezpečené zmenšenie a spätné zväčšenie počtu neurónov vo vrstvách pre vytvorenie vzájomných zavyslostí medzi kanálmi.
- Váhovanie *(Scale Module)* - Vstupmo sú hodnoty z predchádzajúceho modulu, na ktoré je použitá funkcia `sigmoid`. Takto upravené váhové hodnoty sú použité na úpravu kanálov. Kažký element jednotlivých kanálov je vynásobený hodnotou odpovedajúceho kanálu, čim sa docieli pripisovanie odpovedajúcej dôležitosti kanálom.



Obr. 1.5: Module bottleneck s pridanými časťami stlačenia a excitácie.[21]

Na rozdiel od MobileNetV2, ktorá má iba jednu variantu, MobileNetV3 sa ešte delí na malý *(*small) a veľký *(*large). Štruktúra vrstiev pre veľký variant je zobrazená v Tab. 1.2, v ktorej skratka SE značí použitie `stlačenia a excitácie`, AF značí použitú aktivačnú funkciu, RE označuje `ReLU` a HS označuje `h-swish`.[21]

Tab. 1.2: Tabuľka s MobileNetV3 large vrstvami a ich parametrami

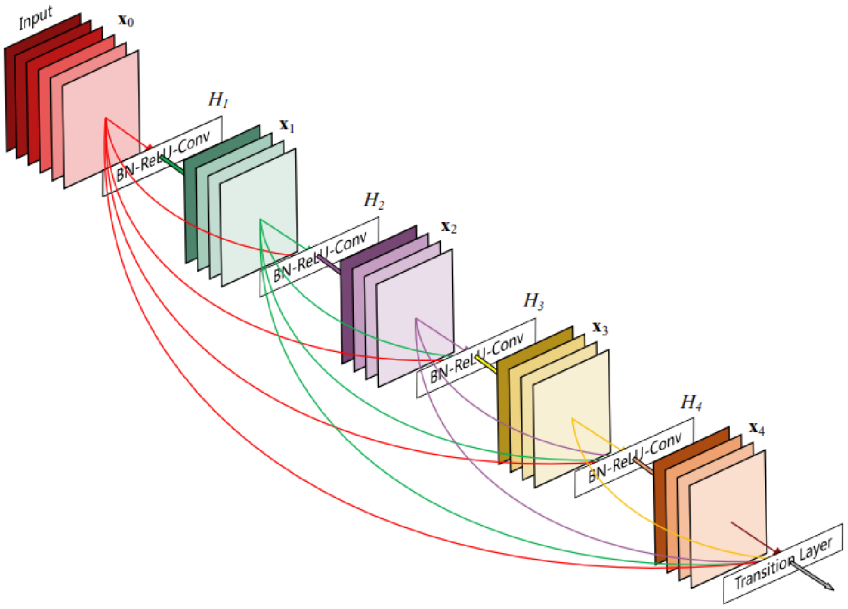
Vstup *(Input)*	Operátor *(Operator)*	Činiteľ zväčšenie *(Expansion factor)*	Kanály *(Channels)*	SE	AF	Krok *(Stride)*
$224^2 \times 3$	Conv 2D	-	16		HS	2
$112^2 \times 16$	Bottleneck	1	16	-	RE	1
$112^2 \times 16$	Bottleneck	4	24	-	RE	2
$56^2 \times 24$	Bottleneck	3	24	-	RE	1
$56^2 \times 24$	Bottleneck	3	40	+	RE	2
$28^2 \times 40$	Bottleneck	3	40	+	RE	1
$28^2 \times 40$	Bottleneck	3	40	+	RE	1
$28^2 \times 40$	Bottleneck	6	80	-	HS	2
$14^2 \times 80$	Bottleneck	2.5	80	-	HS	1
$14^2 \times 80$	Bottleneck	2.3	80	-	HS	1
$14^2 \times 80$	Bottleneck	2.3	80	-	HS	1
$14^2 \times 80$	Bottleneck	6	112	+	HS	1
$14^2 \times 112$	Bottleneck	6	112	+	HS	1
$14^2 \times 112$	Bottleneck	6	160	+	HS	2
$7^2 \times 160$	Bottleneck	6	160	+	HS	1
$7^2 \times 160$	Bottleneck	6	160	+	HS	1
$7^2 \times 160$	Conv 2D	-	960		HS	1
$7^2 \times 960$	Avgpool	-	-		-	1
$1 \times 1 \times 960$	Conv 2D	-	1280		HS	1
$1 \times 1 \times 1280$	Conv 2D	-	k		-	1

DenseNet

Moderné konvolučné neurónové siete sú stále väčšie a hlbšie. Tým sa ale informácie prechádzajúce modelom postupne strácajú, čo zapríčiňuje zníženie presnosti. Preto z tohoto dôvodu vznikla arhitektúra neurónových sietí s názvom `ResNet`[9]) *(Residula Network)*. Tá, ako prvá predstavila prístup, nazvaný skratkové spojenie identity *(identity shortcut connection)*, ktorý vytvorý sekundárne spojenie. To preskočí určitý počet vrstiev, na konci ktorých je sčítané s primárnym spojením. Opakovanie tohoto prístupu docieli lepšiu doprednú propagáciu informácií, zvýšenie presnosti a to pri nezmenenej výpočetnej náročnosti.

`DenseNet` alebo hustá konvolučná nerónová sieť *(Dense Convolution Network)*

[9]Viac informácií je možné nájst na: `https://arxiv.org/pdf/1512.03385.pdf`.



Obr. 1.6: Ukážka spojení medzi susednými hustými blokmi.[22]

rieši tento problém predstavením tzv. hustého bloku*(Dense Block)*, ktorej úlohov je prepojiť každú vrstvu s predchadzajúcimi vrstvami. Ukážka architektúry je zobrazená na Obr. 1.6 a znázorňuje spojenia medzi jednotlivými vrstvami. Na rozdiel od klasickej konvolučnej neurónovej siete, majúcej L spojení, hustá konvolučná sieť ich má $\frac{L \cdot (L+1)}{2}$.

Hustý blok je zložený zo šiestich vrstiev (normalizačná, ReLU, konvolučná, normalizačná, ReLU, konvolučná). Veľkosť výstupu z bloku je stále konštantný, označený ako k, na rozdiel od premenlivej veľkosti vstupných hodnôt. Po výstupe sú hodnoty spájané z hodnotami vstupu, po vzore architektúry modelu `ResNet`. Rozdiel je vo forme spájania, ktorá je zreťazenie hodnôt a nie sčítanie. Tým je zabezpečený rast veľkosti o k pri kažkom prechode blokom a zachovanie predchádzajúcich hodnôt, ktoré boli extrahované. Pre zlepšenie výkonu a zredukovanie počtu hodnôt, ktoré lineárne narastajú, bola vytvorená prechodová vrstva *(Transition Layer)*. Jej úlohov je zmenšiť počet hodnôt na polovicu a aplikovať priemerné združovanie, čiže average pooling. Prechodaná vrstva sa v modeli nachádza tri krát a husté bloky štyri krát. Presná postupnosť jednotlivých vrstiev je zobrazená v Tab 1.3.[22]

Tab. 1.3: Tabuľka s DenseNet vrstvami a ich parametrami

Vstup *(Input)*	Operátor *(Operator)*	Filtre (počet / veľkosť) *(Filter (number / size))*	Výstup *(Output)*
$224^2 \times 3$	Input		$224^2 \times 3$
$224^2 \times 3$	Conv 2D	96 / 7x7	$112^2 \times 96$
$112^2 \times 96$	Maxpool	96 / 2x2	$57^2 \times 96$
$57^2 \times 96$	Dense Block	6 / $\begin{pmatrix} 1\times1 & 4k \\ 3\times3 & k \end{pmatrix}$	$57^2 \times 384$
$57^2 \times 384$	Transition	1 / $\begin{pmatrix} 1\times1 & 192 \\ 2\times2 & 192 \end{pmatrix}$	$29^2 \times 192$
$29^2 \times 192$	Dense Block	12 / $\begin{pmatrix} 1\times1 & 4k \\ 3\times3 & k \end{pmatrix}$	$29^2 \times 768$
$29^2 \times 768$	Transition	1 / $\begin{pmatrix} 1\times1 & 384 \\ 2\times2 & 384 \end{pmatrix}$	$15^2 \times 384$
$15^2 \times 384$	Dense Block	36 / $\begin{pmatrix} 1\times1 & 4k \\ 3\times3 & k \end{pmatrix}$	$15^2 \times 2112$
$15^2 \times 2112$	Transition	1 / $\begin{pmatrix} 1\times1 & 1056 \\ 2\times2 & 1056 \end{pmatrix}$	$8^2 \times 1056$
$8^2 \times 1056$	Dense Block	24 / $\begin{pmatrix} 1\times1 & 4k \\ 3\times3 & k \end{pmatrix}$	$8^2 \times 2208$
$8^2 \times 2208$	Global Avgpool	2208 / 8x8	$1^2 \times 2208$
$1 \times 1 \times 2208$	2D FCL with Softmax		2

2 Rešerš vedeckých prístupov

V tejto kapitole budú spomenuté viaceré štúdie, zaoberájúce sa podobnou problematikou a jej softvérovou implementáciou. Každý z prístupov je zameraný na odlišný aspekt alebo typ internetovej siete. Hlavnými popisovanými parametrami budú využívané dátové sady, extrakcia vlastností z dát a ich reprezentácia v grafickej podobe. Na koniec bude spomenutý proces učenia neurónovej siete a dosiahnuté výsledky vo vybraných prostrediach.

2.1 Aktuálny stav vedy a techniky

Detekcia anomálií v 5G sieťach pomocou strojového učenia

Autor práce popisuje spôsob implementácie SDS[1] do 5G sietí. Systém je navrhnutý spôsobom, ktorý umožňuje prístup k dvom sieťovým spojenia súčasne. Prvé spojenie je medzi chrbtovou sieťov a hlavnou sieťou a druhá je z prepojovacích liniek von z hlavnej siete (detekciu hrozieb bod-bod).

Používaná dátová sada, obsahujúca škodlivú a normálnu dátovú komunikáciu bola zvolená IDS-2018[2]. Zostavenie sady pozostáva z veľkého počtu útočiacich strojov, počítačov a serverov, na ktoré boli použité útoky ako DoS, brute force, útoky na webové stránky (OWASP top 10[3]) a ostatné a program `CICFlowMeter` na extrakciu vlastností sieťových spojení.

Vyberané vlastnosti boli rozdelene do dvoch skupín. Do prvej skupiny sú zaradené základné vlastnosti výtvárané `CICFlowMeterom`, je ich 6. Druhá skupina je zameraná na metriku IAT[4] a jej rôzne varianty, ktorých je 14. Vybrané vlastnosti sú uložené do CSV súborov a následne prevedené na obrázky o veľkosti 100x100x3. Obr 2.1 zobrazuje ukážku vytvorených obrázkov.

Vytvorená sada obrázkov bola použitá ako vstup pre neurónovú sieť `Google AutoML Vision`[5]. Trénovanie modelu trvajúce 3 hodiny dosiahlo priemernej presnosti 97,6 %. Matica zámen ukazuje bezchybnú identifikáciu normálnej prevádzky a škodlivú prevádzu identifikovanú ako normálnu s pravdepodobnosťov 3,6 %. [25]

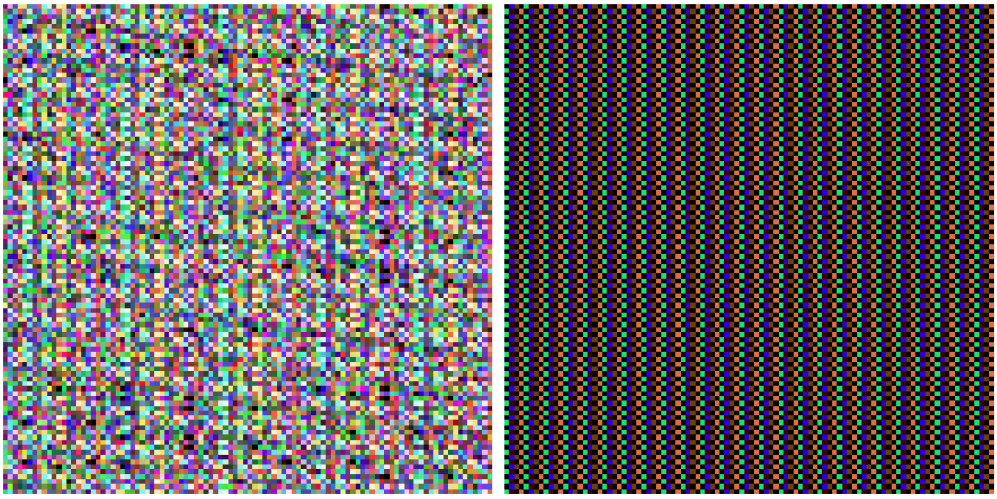
[1] Softvérovo definovaný systém.

[2] Dostupné na adrese: `https://www.unb.ca/cic/datasets/ids-2018.html`.

[3] Viac informácií je možné nájst na: `https://owasp.org/www-project-top-ten/`.

[4] Inter-Arrival Time.

[5] Viac informácií je možné nájst na: `https://cloud.google.com/vision/automl/docs`.



Obr. 2.1: Ukážka obrázkov sieťovéj prevádzky. Na pravo je normálna a ľavo je škodlivá komunikácia [25].

Využitie obrázkovej reprezentácie sieťovej prevádzky a umelého učenia na detekciu botnetov

Práca je zameraná na identifikáciu útokov vytváraných botnetom na rôzne typy sietí. Metodológia postupu je rozdelená do viacerých skupín:

- získavanie dát,
- spracovanie dát,
- detekcia botnetu.

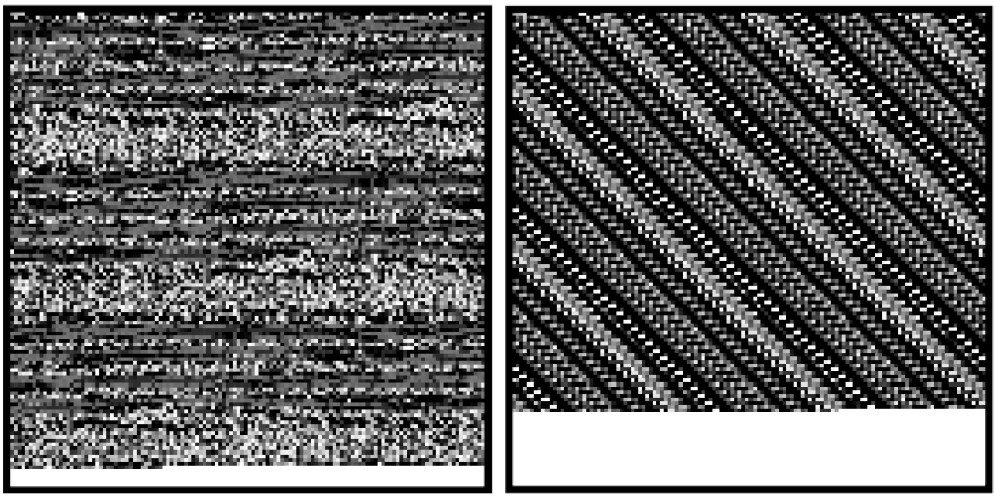
Dáta používané na trénovanie neurónovej siete boli v offline podobe a pochádzali z dátovej sady CTU-13[6]. Sada obsahuje normálnu sieťovú prevádzku a prevádzku spôsobenú botnetom. Dáta pre online (aktuálne dáta nachádzajúce sa v monitorovanej siete) analýzu boli zachytávané v UCF (University of Central Florida) kampuse.

Získané dáta boli normalizované a vyčistené od vzorkov majúcich negatívny dopad na proces učenia. Účinok bol dosiahnutý znáhodnením IP adries, adries portov v paketoch, odstránením paketov neobsahujúcich aplikačnú vrstvu a duplikátov (pakety zo zhodným obsahom) a doplnenie dát o hodnotu 0xFF (biela farba). Upravené dáta boli zakódované do binárneho ASCII formátu a prekonvertované na obrázok. Celková vytvorená sada obrázkov mala veľkosť 450 763 vzoriek. Ukažka je zobrazená na Obr. 2.2

Pre samotnú detekciu dotnetov bola požitá neurónová sieť `DenseNet`[7] s predt-

[6]Dostupné na adrese: `https://www.stratosphereips.org/datasets-ctu13`.

[7]Viac informácií je možné nájst na: `https://pytorch.org/hub/pytorch_vision_densenet/`.



Obr. 2.2: Ukážka vytvorených obrázkov. Na pravo je normálna a ľavo je škodlivá komunikácia [26].

rénovanými váhami. Pri dotrénovavaní modelu bolo experimentované z pôvodnými váhami. Trénovanie trvalo tri epochy a pozostávalo z piatich pokusov. Pri prvom bola použitá neurónová sieť bez predtrénovaných váh a dosiahna najnižšiu presnosť 33,4 %. Nasledujúce pokusy spočívali v uzamknutí váh na prvých 10, 75, 200 a 525 vrstvách. Najlepšiu presnosť 99.98 % dosiahol model s prvými desiatimi uzamknutými váhami.[26]

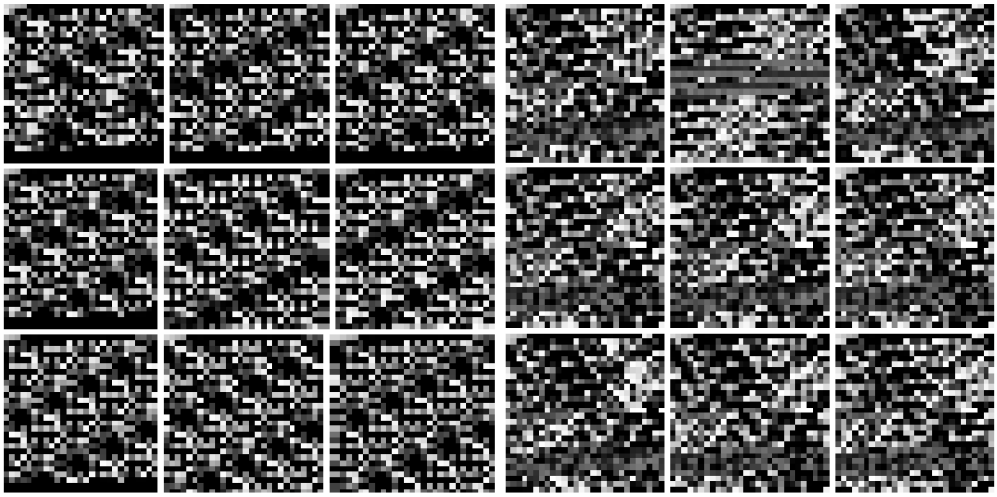
Klasifikácia malvérov v sieťovej prevádzke použitím konvolučnej neurónovej siete

Snaha autora práce bola vytvoriť model detekujúci anomálie, ktorý na vstupe prijíma normalizovanú surovú sieťovú komunikáciu transformovanú do podoby obrázka. Aby bola podmienka splnená, autor si vytvoril vlastnú dátovú sadu nazývanú USTC-TFC2016[8]. Pozostávajúcu z pcap súborov, ktoré sú rozdelená na normálnu prevádzku a prevádzku prenášajúcu malvéry. V normálnej prevázke sú zachytené sieťové komunikácie BitTorrent, Facetime, FTP, Gmail a MySQL. Na vytvorenie škodlivej komunikácie boli použité malvéri cridex, geodo, htbot, miuref, atď.

Prevádzka je rozdelená na prúdy paketov alebo na relácie. Oba pozostávajú z paketov majúcich rovnaký transportný protokol, cieľovú a zdrovovú IP adresu a porty. Rozdiel je, že prúd pozostáva z komunikácie iba jedným smerom na rozdiel od relácie, v ktorej sú obsiahnuté oba smery.

[8]Dostupné na adrese: `https://github.com/yungshenglu/USTC-TFC2016`.

Transformáciu dát na obrázky má na starosti nástroj USTC-TL2016[9]., vytvorený autorom. Pri vstupe sú IP adresy znáhodnené a odstránené sú prázdne[10] alebo duplicitné pakety. Výstupné obrazoky musia mať rovnakú veľkosť a to 784 bajtov. To je zabezpečené doplnením hodnoty 0x00 (biela farba) alebo odstrihnutím nadbytočných dát. Výsledný sivý obrázok je uložený do IDX formátu. Ukážka je zobrazená na Obr. 2.3.



Obr. 2.3: Ukážka výtvorenej sady obrázkov. Na pravo je normálna a ľavo je škodlivá komunikácia [27].

Testované boli štyri scenáre. Prúd paketov alebo relácie pozostávajúcich iba z aplikačnej vrstvy alebo z všetkých sieťových vrstiev. Kažky zo scenárov bol individuálne otestovaný v troch modeloch vytvorených autorom (binárna klasifikácia, 10 a 20 triedna klasifikácia). Pri binárnej klasifikácii dosiahol scenár pozostávajúci s prúdu paketov a všetkych sieťových vrstiev úspešnosť 100 %. Pre 10 a 20 triednu klasifikáciu bola úspešnosť v priemere 99 %.[27]

Schéma detekcie sieťových anomálií, založená na reprezenacií vlastností a dátovej argumentácií

Cieľom štúdie je znížiť miery falošne negatívnych predikcií, vyvolaných zriedkavo sa vyskytujúcimi anomaliámi. Za týmto účelom boli definované postupné procesy, ktoré sú: zachytenie paketov, extrakcia vlastností, spravocanie dát, trénovanie modelu a evaluácia modelu.

Použité dáta pochádzali z piatich dátových sád. Sada NSL-KDD[11] a UNSW-

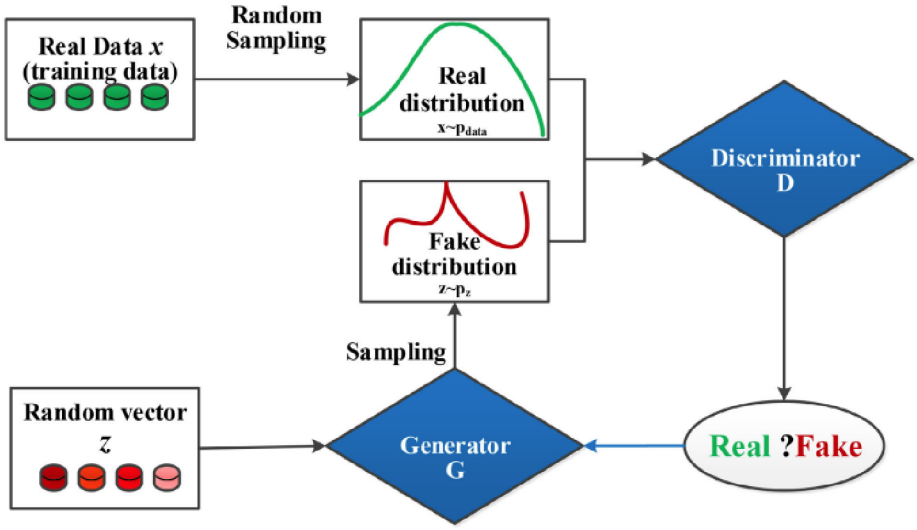
[9]Bližšie infornácie sú dostupne na: `https://github.com/yungshenglu/USTC-TK2016`

[10]Pakety neobsahujúce aplikačnú vrstvu.

[11]Dostupné na adrese: `https://www.unb.ca/cic/datasets/nsl.html`.

NB15[12] sú hlavnými použitými dátovými sadami. Sady JM1[13], PC5[14] a dataset kreditných kariet sú primárne použité na validáciu modelu pre aplikáciu v ostatných prostrediach. Pre extrakciu vlastností bol vytvorený postup kódovania do binárnej sústavy. Jednostlivé vlastnosti obsahujúce N prvkov sú zakódované do N-bitového čísla. Každé číslo obsahuje iba jednu jednotku a N-1 núl. Na vytvorené čísla sa aplikuje redukcia vlastností podľa významnosti v sade a dáta sa normalizujú. Takto zakódované a upravené vlastnosti sú spájané do 8-bitových čísel a prevedené do decimálnej sústavy. Pre vytvorenie obrázkov sú dáta rozdeľované každých 224 prvkov a spojovaných do matice o veľkosti 224x224. RGB hodnota je vytvorená kopírovaním čísla na mieste pixelu pre vznik šedého obrázka.

Vytvorená sada obsahovala nerovnomerné zastúpenie porovnávaných typov vzoriek. Pre odstránenie tohoto nedostatku bola použitá technika dátovej augumentácie. N_{max} je hodnota reprezentujúca počet vzoriek jedného typu v najväčšej skupine. Ostaným skupinám sú vzorky umelo pridávané, až kým ich počet nie je rovný hodnote N_{max}. Spôsob, akým proces vytvárania funguje je zobrazený na Obr. 2.4.



Obr. 2.4: Diagram popisujúci dátovú augumentáciu [28].

Zvolený model konvolučnej neurónovej siete bol `ResNet50`[15], ktorého trénovanie pozostávalo z pädesiatich epoch a 1 024 vzoriek v jednej dávke. Testovaním binárnej klasifikácie bola dosiahnutá presnosť 94,9 % a F1-miera 95,3 %. [28]

[12]Dostupné na adrese: `https://research.unsw.edu.au/projects/unsw-nb15-dataset`.

[13]Dostupné na adrese: `https://datahub.io/machine-learning/jm1`.

[14]Dostupné na adrese: `https://zenodo.org/record/268439`.

[15]Viac informácií je možné nájst na: `https://www.mathworks.com/help/deeplearning/ref/resnet50.html`.

2.2 Rekapitulácia prístupov

V tabuľke číslo 2.1 sa nachádzajú štyri prístupy detekcie anomálií pomocou hlbokej neurónovej siete. Tie sa líšia použitými dátovými sadami, modelmi neurónových sietí a dosiahnutou presnosťov. Predovšetkým, každý s prístupov použil unikátny spôsob transformácie sieťovej komunikácie na obrázky. Iba jeden z nich využil celý RGB rozsah pixelov, čim dosiehol vytváranie farebných obrázkov. V ostatných prípadoch mali hodnoty RGB rovnakú hodnotu, čim vytvorili obrázok v odtieni šedi. Avšak techniky transformácie do odtieňov šedi dosiahli najlepších konečných výsledkov. A to presnosť 99.98 % v práci zameranej na detekciu botnetov a prestiosť 99.00 % a 100,00 % v práci klasifikujúcej malvéry v sieťovej prevádzke.

Tab. 2.1: Porovnanie moderných metód detekcie anomálií hlbokov neurónovou sieťou

Rok	**Názov práce**	**Datová sada**	**Algoritmy**	**Presnosť**
2020	Detekcia anomálií v 5G sieťach pomocou strojového učenia [25]	IDS-2018	Google AutoML Vision	97,60 %
2018	Využitie obrázkovej reprezentácie sieťovej prevádzky a umelého učenia na detekciu botnetov [26]	CTU-13	DenseNet	99,98 %
2017	Klasifikácia malvérov v sieťovej prevádzke použitím konvolučnej neurónovej siete [27]	USTC-TFC2016	LeNet-5	99,00 % a 100,00 %
2020	Schéma detekcie sieťových anomálií, založená na reprezenacií vlastností a dátovej argumentácií [28]	NSL-KDD a UNSWNB15	ResNet50	94,90 %

3 Príprava komponent na detekciu anomálií

Táto kapitola je zameraná na popis prípravy stavebných prvkov, potrebných na detekciu anomálií. Ako prvá bude popísaná zvolená dátova sada a navrhnutý spôsob transformácie do grafickej reprezentácie. Následne, za použitia vopred vytvorených obrázkov budú natrénované vybrané neurónové siete a zobrazené ich dosiahnuté presnosti. A nakoniec bude popísaný nástroj na zachytávanie sieťovej komunikácie a jeho potrebné úpravy pre dosiahnutie požadovanej činnosti.

3.1 Transformácia dát

3.1.1 Dátová sada

Pri výbere dátovej sady boli definované dve podmienky, ktoré sú:

- Dátová sada musí odrážať sieťový prenos vyskytujúci sa v reálnom svete.
- Útoky, použité na jej tvorbu, musia mať rôznorodý charakter.

Tieto podmienky spĺňa dátová sada CSE-CIC-IDS2018[1], vytvorená za účelom detekcie anomálií. Sieťová prevádzka v tejto sade je rozdelená do dvoch profilov. Prvý profil (tvorcami nazvané ako B-profil) obsahuje normálnu sieťovú komunikáciu pozostávajúcu z protokolov ako: HTTP, HTTPS, SMTP, POP3, IMAP, SSH a FTP. Druhý profil (tvorcami nazvané ako M-profil) je tvorený útokmi ako: FTP a SSH bruteforce, DoS (Slowloris, Hulk, GoldenEye), DDoS + skenovanie portov, útoky na webové aplikácie a útoky za pomoci botnetu.

Celá dátová sada je veľkostne obsiahla, lebo obsahuje celú sieťovú komunikáciu, ktorá je uložená vo formáte PCAP. Vyextrahované systémové logy zo všetkých užívateľských staníc, na ktoré bolo útočené a v poslednej rade aj súbory vo formáte CSV. Tie obsahujú množstvo vlastností a vypočítaných hodnôt o každej relácií nástrojom `CICFlowMeter`. Bližšie informácie sú dostupné v kapitole 1.2.2.

3.1.2 Transformácia dát na obrázky

Na transformáciu obrázkov bol vytvorené nástroj v programovacom jazyku Python. Jeho funkcionalita spočíva v tom, že vstupné dáta vo formáte CSV upravý do podoby RGB obrázkov o konštantnej veľkosti. Preto postup je rozdelený na tri dielčie časti.

1. Úprava dat v formáte CSV.
2. Vytvorenie pixelov.

[1]Dostupne na adrese: `https://www.unb.ca/cic/datasets/ids-2018.html`.

3. Poskladanie obrázkov.

Úprava dat v formáte CSV

Pred tým ako je možné pracovať s vlastnosťami v súboroch, musia byť najskôr upravené do vyhovujúceho tvaru pre prípad výskytu neštandartnej hodnoty. Každý riadok obsahuje časové razítko, v ktorom je zaznamenaný čas generovania komunikácie, slúžiace na zoradenie dát podľa ich náväznej postupnosti. Daná časová postupnosť je v súboroch narušená a musí byť opravená zmenou formátu časového razítka a následne vzostupne zoradená (od staršej po mladšiu komunikáciu). Pri dodatočnej kontrole neboli zistené žiadné chyby, preto boli dáta posunuté do ďalšieho kroku.

Vytvorenie pixelov

Pixel je zložený z troch osembitových hodnôt označených ako RGB (červená, zelená a modrá farba). Úpravou viacerých vlastností z relácií, boli získané práve požadované tri bajty. Počas procesu výberu vlastností, vhodných na účely detekcie anomálií, bolo rozhodnuté pre tvorbu troch dátových sád z odlišným počtom obsihnutých vlastností. Tento postup bol vybraný za účelom porovnania množstva vybraných vlastností na úspešnosť trénovaného modelu.

Prvý prístup reprezentoval jeden riadok dát (jednu reláciu) ako dva pixely. Zvolené vlastnosti a počet bitov nimi zastúpené sú:

- `Flow duration` – 16b – časová dĺžka relácie v milisekundách.
- `Flags` – 8b – jednotlivé bity reprezentujú, či boli v prenose prítomné TCP príznaky. Poradie príznakov je nasledovné: `ECE, CWE, URG, ACK, PSH, RST, SYN, FIN`.
- `Port` – 16b – číslo cieľového portu.
- `Packet size average` – 8b – priemerná veľkosť paketu v relácií.

Pre **druhý prístup**, bola množina vlastností rozšírená z pôvodných štyroch na šesť a počet pixelov sa zvýšil o dodatočné dva. Pridané vlastnosti sú:

- `Flow IAT Mean` – 24b – stredná časová hodnota medzi dvoma paketmi v relácií
- `Flow Packet/s` – 24b – počet paketov v relácií za sekundu

V poslednom **treťom prístupe** bol počet vlastností rozšírený o dodatočnych päť a počet pixelov bol navýšený na osem. Pridané vlastnosti sú:

- `Tot Fwd Pkts` – 8b – celkový počet dopredných paketov v relácií

```
def binToDecSplit24b(value):
    binary = bin(value)
    bin1 = "0b"
    bin2 = "0b"
    bin3 = "0b"
    for i in range(32):
        if i <= 33-len(binary):
            if len(bin1) < 10:
                bin1 += "0"
            elif len(bin2) < 10:
                bin2 += "0"
            else:
                bin3 += "0"
        else:
            if len(bin1) < 10:
                bin1 += binary[len(binary)-32+i]
            elif len(bin2) < 10:
                bin2 += binary[len(binary)-32+i]
            else:
                bin3 += binary[len(binary)-32+i]
    return [int(bin1,2),int(bin2,2),int(bin3,2)]
```

Výpis 3.1.1: Funkcia rozdeľujúca 24 bitové čísla na tri 8 bitové čísla.

- `Fwd Header Len` – 16b – celkový počet bajtov, použítých v hlavičke doprednych paketov počas relácie
- `Fwd IAT Mean` – 24b – stredná časová hodnota medzi dvoma doprednými paketmi v relácií
- `Fwd IAT Std` – 24b – štandartná časová odchylka medzi dvoma doprednými paketmi v relácií
- `Bwd IAT Mean` – 24b – stredná časová hodnota medzi dvoma spiatočnými paketmi v relácií

V tabuľke číslo 3.1 sú pre prehľadnosť vyznačené definované prístupy a vlastnosti v nich obsiahnuté. Veľkosť niektorých vybraných vlastností prekračujú hodnotu 24 bitového čísla. Preto aby ich bolo možné binárne reprezentovať, museli byť upravené. Napríklad hodnota flow duration bola podelená bez zvyšku štyrmi, čím sme docielili zníženie hodnoty a reprezentáciu viacerých hodnôt jednou. Ak napriek redukcií stále presahovala stanovené maximum, bola nastavená na hodnotu 65 535. Podobným spôsobom boli upravené aj ostatné hodnoty a následne 16 a 24 bitové čisla sme

Tab. 3.1: Porovnanie prístupov a vlastností v nich obsiahnuté

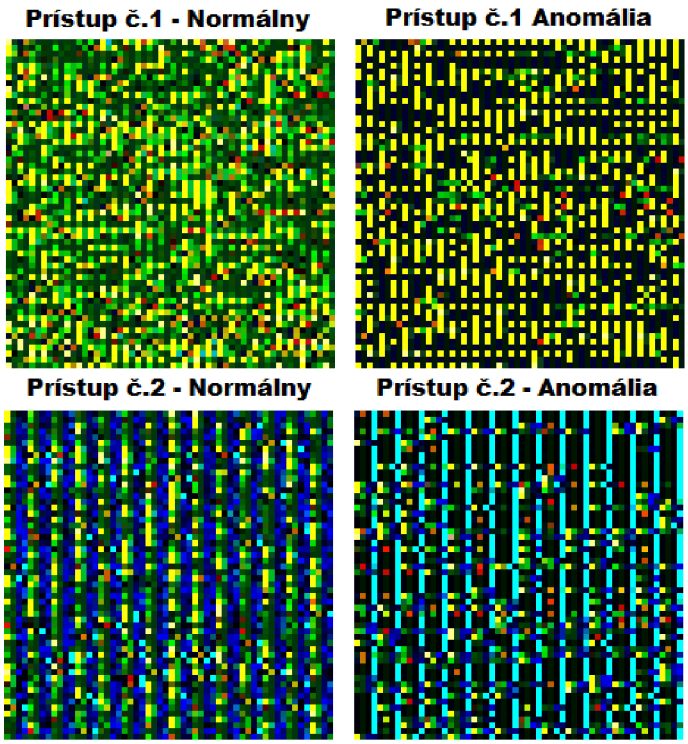
Vlastnosti	1. prístup	2. prístup	3. prístup
Flow duration	✓	✓	✓
Flags	✓	✓	✓
Port	✓	✓	✓
Packet size average	✓	✓	✓
Flow IAT Mean	✗	✓	✓
Flow Packet/s	✗	✓	✓
Tot Fwd Pkts	✗	✗	✓
Fwd Header Len	✗	✗	✓
Fwd IAT Mean	✗	✗	✓
Fwd IAT Std	✗	✗	✓
Bwd IAT Mean	✗	✗	✓

rozdelili na osembitové časti, reprezentujúce RGB. Výpis 3.1.1 je ukážka funkcie pre rozdelenie 24 bitového čisla.

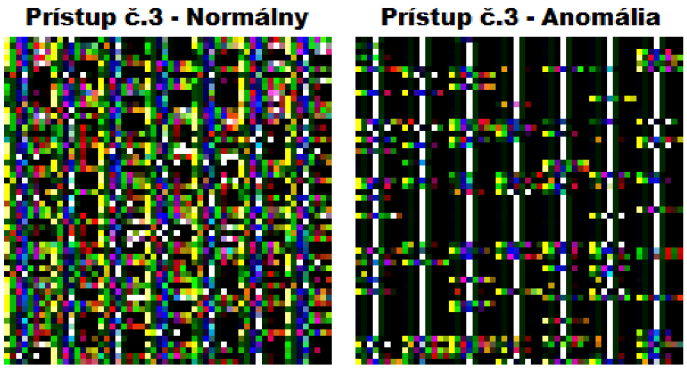
Kompozícia obrázkov

Na zostrojenie obrázka je potrebné zoskupiť iný počet relácií v závyslosti na zvolenom prístupe. V tomto prípade bol tovrený obrázok o veľkosti 56x56 pixelov. Pri tejto veľkosti jeden obrázok reprezentuje 1 568 relácií s použitím prvého prístupu, 784 relácií s použitím druhého prístupu a 392 relácií potrebuje posledný prístup. Na Obr. 3.1 sú zobrazené prvé dva prístupy a na Obr. 3.2 je zobrazený posledný, tretí prístup.

Pixely boli ukladané do jednorozmerného poľa. V prípade, že sa využili všetky dáta z CSV súborov a počet pixelov nemal dostatočnú veľkosť, boli použité čierne pixely ako výplň. Druhou možnosťou pri nedostatku dát pre obrázok bolo jeho zahodenie. Tá možnosť použitá nebola z dôvodu straty dát. Naplnené pole bolo upravené na dvojrozmerné a uložené ako obrázky v png formáte. Každému obrázku bolo pridelené identifikačné číslo, pod ktorým bol uložený. Vytvorené boli aj jeho označenia, na základe obsiahnutých dát. Pri binárnom klasifikácií bola značka 0 použitá pre normálnu prevádzku a 1 ak obsahoval dve a viac škodlivých relácií. Taktiež bolo vytvorené označenie pre viactriednu klasifikáciu. To tak, že hodnotám väčším alebo rovným ako 1 bol namapovaný jeden z možných útokov. Výsledné označenia boli pridané do CSV súborov, v ktorom sú asociované s identifikátorom obrázka ako *(cesta k obrázku,označenie)*.



Obr. 3.1: Ukážka vytvorených obrázkov prvým a druhým prístupom. Zobrazená anomália je FTP bruteforce.



Obr. 3.2: Ukážka vytvorených obrázkov tretím prístupom. Zobrazená anomália je FTP bruteforce.

Vytvorené boli tri dátové sady z odlišným počtom obrázkov, z dôvodu rozličného počtu potrebných pixelov na reprezentáciu jednej relácie. Najmenšia sada o veľkosti 20 709 prvkov bola vytvorená s prvým transformačným prístupom. Nasleduje druhý prístum s veľkosťov sady 41 391 prvkov. A posledný prístup má najväčší počet prvkov, a to 109 295.

3.2 Proces učenia neurónovej siete

Po vytvorení troch dátových sád začal postup postupného trénovania neurónových sietí. Na to bol použitý framework s názvom `PyTorch`[2]. Tento nástroj je vytvorený v programovacom jazyku python a umožnuje vytváranie ale aj použitie overených, generalizovaných modelov, ktoré je možné upravovať podľa potreby. Zo sady modelov boli vybrané tri:

1. `MobileNetV2`[3],
2. `MobileNetV3 Large`[4],
3. `DenseNet 161`[5].

Každý z modelov bol trénovaný na rozličnej dátovej sade alebo skupinke dátových sád. Presnejšie na trénovanie modelu `MobileNetV2` boli použité dátové sady s dvomi a štyrmi pixelmi na jednu reláciu. Model `MobileNetV3 Large` mal ako vstupnú dátovú sadu so štyrmi a ôsmimi pixelmi na jednu reláciu. Ako posledný učený model bol `DenseNet 161`, na ktorý bola použítá iba dátová sada s ôsmimi pixelmi na jednu reláciu.

Pred tým ako bolo spustené učenie modelov, musel byť definovaný prístup k používanej dátovej sade a nastavené parametre ovplivňujúce proces učenia. Na to aby `Pytorch` mal prístup k datám, musela byť vytvorená trieda, ktorá na základe indexu načíta požadavaný obrázok. Jej vstupnými parametrami sú: cesta k CSV súboru s označeniami obrázkov, cesta k koreňovej zložke obrázkov a dodatočná `PyTorch` trieda na normalizáciu obrázku. Z výpisu kódu 3.2.1 je vidieť to, že po prijatí indexu je obrázok načítaný, normalizovaný a vrátený ak s jeho značkov.

Následne nastavované parametre sú rozdelené do dvoch skupín. V prvej sú tie, ktoré sú zhodné pre všetky zvolené modeli a v druhej skupine sa nachádzajú parametre líšiace sa od použítého modelu. Medzi nemeniace sa pararametre patrí pomer rozdelenia dát na trenovanie a validačné, strátová funkcia a optimalizačný algoritmus. Rozdelenie dátovej sady bolo urobené náhodným výber v pomere 80 % trénovacích a 20 % validačných dát. Náhodným výberom je zabezpečený výskyt všetkých útokov, ktoré sa v dátovej sade vyskytovali. Strátová funkcia používaná pri binárnej ale aj viactriednej klasifikácií je krížová entropia *(Cross Entropy Loss*[6]*)*. A ako po-

[2]Viac informácií je možné nájst na: `https://pytorch.org/`.

[3]Viac informácií je možné nájsť v kapitole 1.3.4 alebo `https://pytorch.org/hub/pytorch_vision_mobilenet_v2/`.

[4]Viac informácií je možné nájsť v kapitole 1.3.4 alebo `https://pytorch.org/blog/torchvision-mobilenet-v3-implementation/`.

[5]Viac informácií je možné nájsť v kapitole 1.6 alebo `https://pytorch.org/hub/pytorch_vision_densenet/`.

[6]Viac informácií je možné nájst na: `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html`.

```python
class AnomalyDataset(Dataset):
    def __init__(self,csvFile,rootDir,transform=None):
        self.annotations = pd.read_csv(csvFile)
        self.rootDir = rootDir
        self.transform = transform

    def __len__(self):
        return len(self.annotations) #Vráti veľkosť dátovej sady

    def __getitem__(self, index):
        imgPath = os.path.join(self.rootDir, \
            self.annotations.iloc[index,0])
        image = io.imread(imgPath)
        yLabel = torch.tensor(int(self.annotations.iloc[index,1]))
        if self.transform:
            image = self.transform(Image.fromarray(image))
        return (image,yLabel)
```

Výpis 3.2.1: Definícia triedy na načítanie obrázka z dátovej sady.

sledný volený parameter je optimalizačný algoritmus. Zvolený bol najpoužívanejší algoritmu *odhadu adaptívneho momentu*, nazývaný aj ako `ADAM`*(ADAptive Moment estimation*[7]*)*. Tento algoritmus je používaný na výpočet zmeny váhových hodnôt medzi neurónmi počas procesu učenia. Proces je taktiež označovaný ako spätná propagácia (*back propagation*[8]).

Do skupiny s líšiacimi sa parametrami patrí veľkosť dávky *(Batch Size)*, rýchlosť učenia *(learning rate)*, počet epoch a dátová sada. Výčet všetkých pridelených hodnôt aj s odpovedajúcimi modelmi sú zobrazené v Tab. 3.2. Pre model `DenseNet` bola rýchlosť učenia postupne menená až kým nebola dosiahnutá požadovaná hodnota. Veľkosť zmeny mala lineárny charakter a prebiahala po dobu 20 epoch.

Po nastavení všetkých potrebných parametrov bolo spustené samotné učenie modelov. Rozložené bolo medzi tri rozličné stanice s rozličnými grafickými kartami. `MobileNetV2` bolo možné trénovať na grafickej karte, ktorá mala veľkosť pamäte 8GB, preto na to bola využítá nvidia RTX 3070. Nasledujúci model `MobileNetV3` mal výššie požiadavky na veľkosť grafickej pamäte. Tento problém bol vyriešený použítím služby Google Colab, ktorá poskytuje grafickú kartu s 15 GB pamäťov na maximálnu dobu používania 12 hodín. Na učenie modelu `DenseNet` služba Google

[7]Viac informácií je možné nájst na: `https://arxiv.org/pdf/1412.6980.pdf`.

[8]Viac informácií je možné nájst na: `https://www.sciencedirect.com/science/article/pii/095418109400011S`.

Tab. 3.2: Tabuľka s hodnotami parametrov pre modely neurónových sietí

Model *(Model)*	Dátová sada *(Dataset)*	Veľkosť dávky *(Batch size)*	Rýchlosť učenia *(Learning rate)*	Počet epoch *(Epochs)*
`MobileNetV2`	1. prístup	100	$5 \cdot 10^{-3}$	100
`MobileNetV2`	2. prístup	100	$5 \cdot 10^{-3}$	100
`MobileNetV3`	2. prístup	100	$5 \cdot 10^{-3}$	50
`MobileNetV3`	3. prístup	100	$5 \cdot 10^{-3}$	50
`DenseNet`	3. prístup	16	$2 \cdot 10^{-2}$ - $5 \cdot 10^{-3}$	50

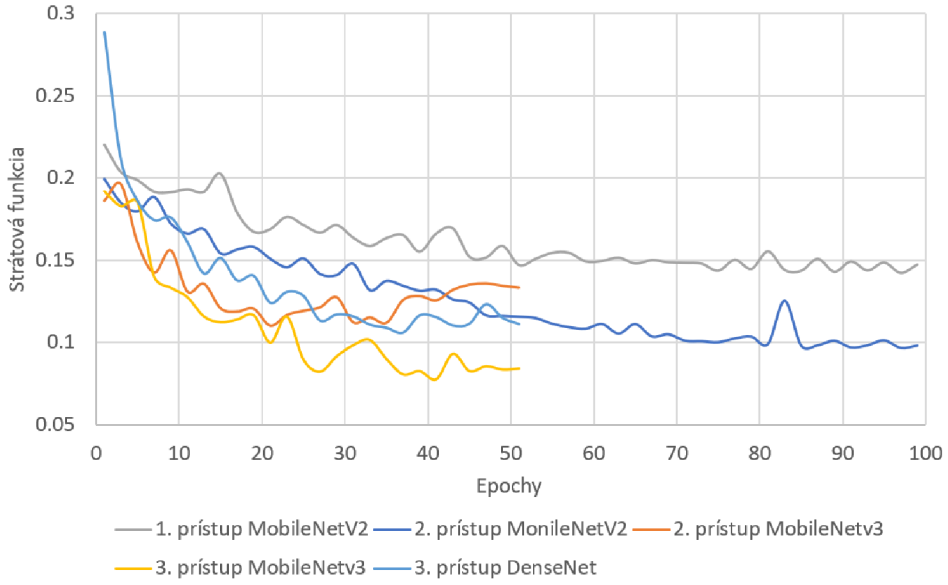
Colab vyhovujúca nebola, pretože vyžadovaný čas na učenie prevyšoval maximálny časový limit používania. Preto bol sprístupnený, s pomocou vedúceho práce, server s grafickou kartou nvidia GTX 1080. Jej veľkosť grafickej pamäte je 12 GB a preto bola nastavená veľkosť dávky na hodnotu 16 a nie na 100, ako pri ostatných modeloch.

Dosiahnuté výsledky z trénovania modelov

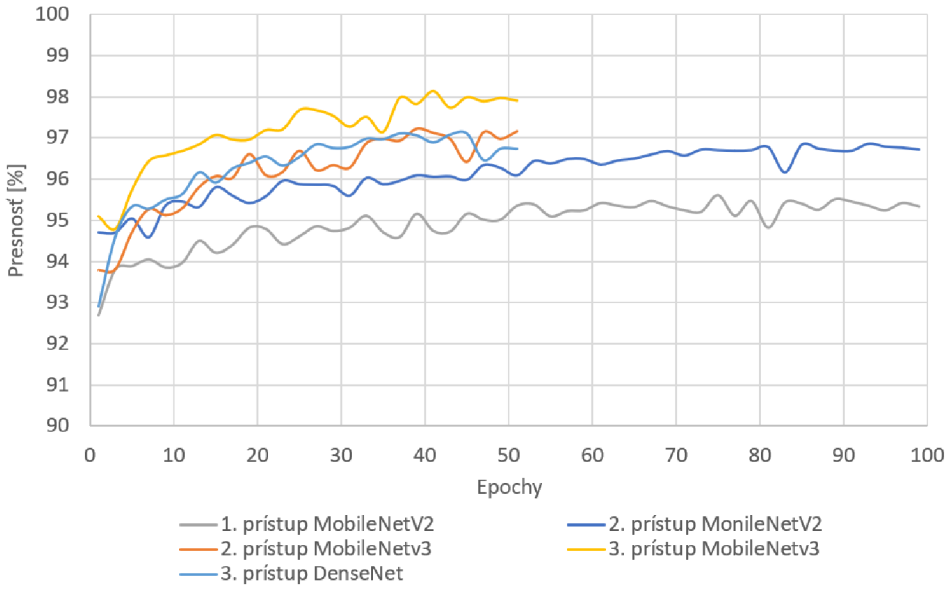
Bolo natrénovaných päť modelov, na troch typoch dátovej sady. Na obrázku číslo 3.3 sú zobrazené hodnoty strátovej funkcie na validačných dátach. Z grafu je zrejmé, že najlepších výsledkov bolo dosiahnuté s modelom `MobileNetV3` s dátovou sadou vytvorenou tretím prístupom a model `MobileNetV2` s dátovou sadou vytvorenou druhým prístupom. Hodnoty dosahované počat celého priebehu tréningu majú klesavý charakter, čo značí pozitívny vývoj vykonávaných predikcií. Pretože daný trend pokračoval až kým tréning neskončil, boli dosiahnuté najnižšie hodnoty stratovej funkcie. Na druhu stranu, najhoršie výsledky boli dosiahnuté s modelom `MobileNetV2` s dátovou sadou vytvorenou prvým prístupom. Trend krivky má klesavý charakter ale aj tak má najväčšiu konečnú hodnotu. Z grafu je možné vidieť značný stúpavý charakter hodnôt od druhej polovice trénovacieho času pre model `MobileNetV3`. Táto nepriaznivá zmena bola spôsobená pretrénovaním modelu.

Na obrázku číslo 3.4 je zobrazený vývoj dosahovanej presnosti jednotlivých modelov na validačných dátach počas učenia. Je možné z neho odvodiť rovnáký záver ako s grafu predchadzajúceho. Najlepšie výsledky boli dosihnuté modelom `MobileNetV3` s dátovou sadou vytvorenou tretím prístupom a najhoršie výsledky s `MobileNetV2` s dátovou sadou vytvorenou prvým prístupom.

Nasledovala aplikácia testovacích dát na naučené modely. Ako prvé boli testované oba modely `MobileNetv2` a z výstupných predikcií boli vytvorené matice zámien, zobrazená na obrázku číslo 3.5. Matica zámien zobrazená na ľavej strane patrí modelu, učenom na dátovej sade vytvorenej prvým prístupom a na pravej strane



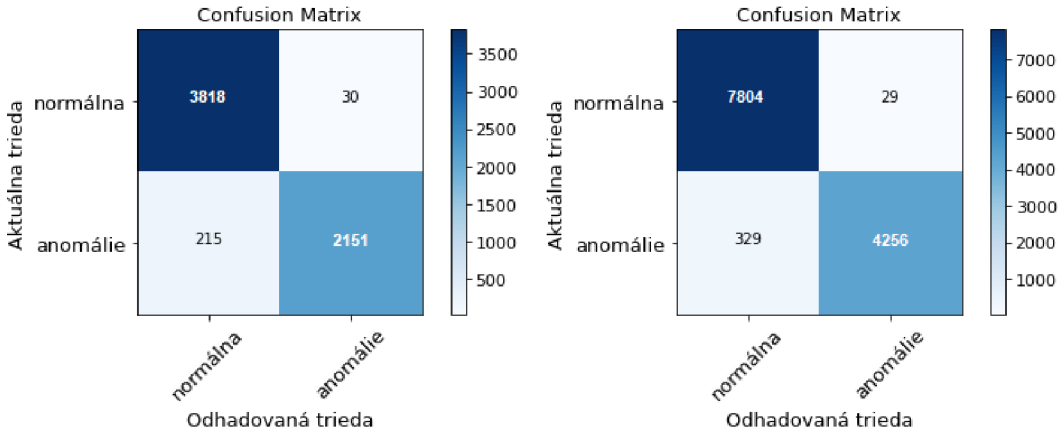
Obr. 3.3: Graf zobrazujúci hodnoty pre stratovú funkciu na validačných dátach počas učenia.



Obr. 3.4: Graf zobrazujúci dosahovanú presnosť v percentách na validačných dátach počas učenia.

dátovou sadou vytvorenou druhým prístupm. Rozdiel v celkovom počte použitých testovacích dát, uvedených v maticiach zámien je zapríčinené rozdielnym počtom prvkov v dátových sadách. Z prezentovaných údajov je taktiež zrejmé, že je zvýšený

výskyt falošne pozitívnych prípadov. Čo značí identifikáciu anomálií ako normálnu sieťovú komunikáciu. Tento výsledok je zrejmý taktiež z vypočítaných hodnôt presnosti v tabuľke číslo 3.3, ktoré sú v porovnaní s presnosťov modelov znížené. Týmto výsledkom je možné usúdiť, že vytvorené modeli budú zachytávať anomálie s nižšou pravdepodovnosťov. Na druhú stranu, počet prípadov identifikácie normálneho prenosu ako anomálie je nízky, čo je zrejmé aj s vypočítanej senzitivity. Tá je v porovnaní s presnosťov modelov vyššia, čo značí nízku pravdepodobnosť výskytu falošných hlásení anomálnych stavov.

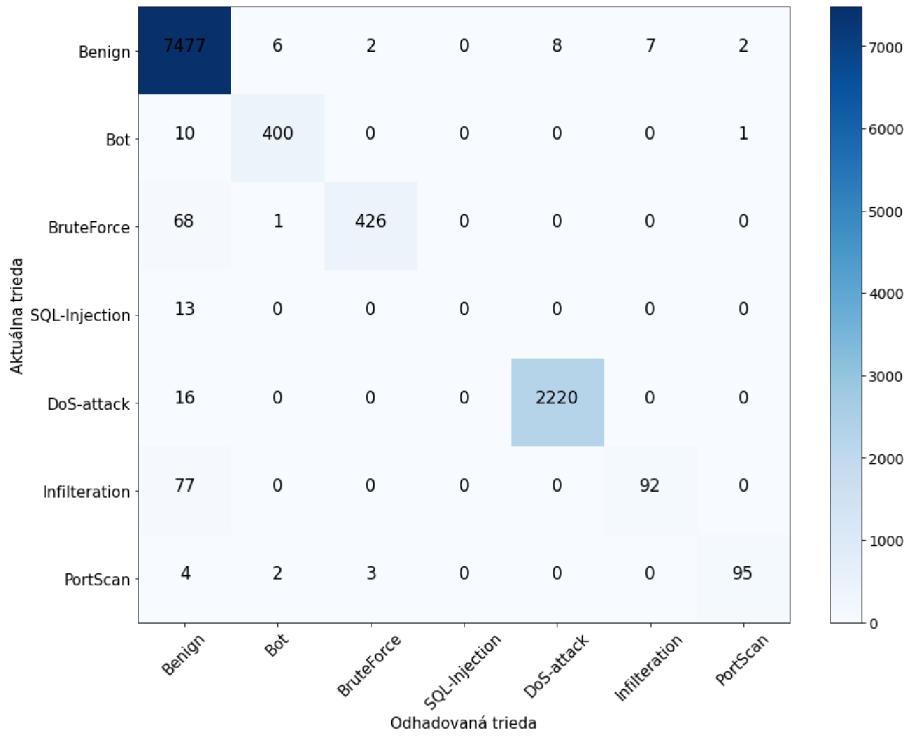


Obr. 3.5: Matice zámien pre oba naučené modely `MobileNetv2`.

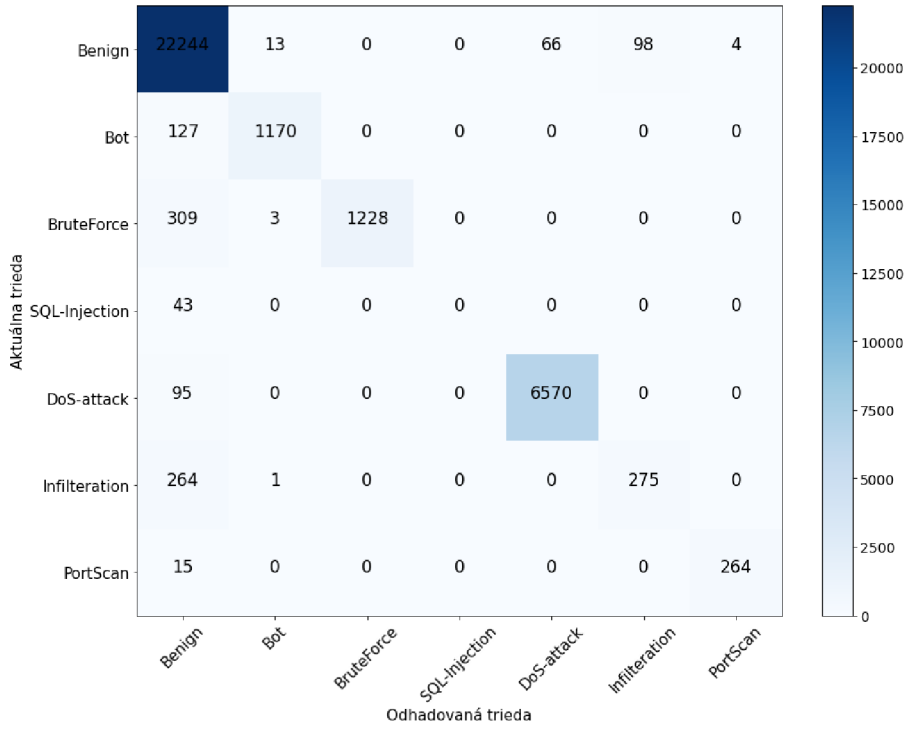
Rovnako boli aplikované testovacie dáta na modeli `MobileNetv3` a `DenseNet` a z ich výstupných hodnôt boli vytvorené matice zámien s viacerými kategóriami, zobrazené na obrázku číslo 3.6 a na obrázku číslo 3.7. Rozdielný celkový počet testovacích dát v maticiach zámien je zapríčinený odlišným pomerom rozdelenia dátovej sady na trénovaciu, validačnú a testovaciu. Model `DenseNet` bol trénovaný z nižším počtom dát pre jeho vysoké požiadavky na trénovancí hardvér. Preto bolo použitých viac dát na jeho testovanie. Z výsledkov modelu `MobileNetv3`, učený dátovou sadou vytvorenou druhým prístupov, nebola vytvorená matica zámien lebo jeho učenie malo za úlohu porovnať dva prístupy tvorby dátovej sady na jednom modely. Preto boli iba vypočítané jeho hodnotiace metriky, zobrazené v tabuľke číslo 3.3. Z dát, zobrazených v maticiach zámien je možné vyvodiť rovnaký výsledok ako pri modeloch `MobileNetv2`. Počet anomálií, identifikovaných ako normálny prenos je vyšší a identifikácia normálneho prenosu ako anomálie je nízky. Tento trend je viditeľný aj v tabuľke číslo 3.3. Podľa hodnotiacej metriky F1-miera, dosiahol najlepšieho výsledku 96.8 % model `MobileNetv3` s dátovou sadou vytvorenou tretím prístupom. Rovnako to platí aj pre ostatné hodnotiace metriky.

Tab. 3.3: Tabuľka metrík pre všetky naučené modely

Model	**Presnosť** [%] *(Accuracy)*	**Precíznosť** [%] *(Precision)*	**Senzitivita** [%] *(Recall)*	**F1-miera** [%] *(F1 score)*
`MobileNetv2 1. prístup`	95.3	90.9	98.6	94.6
`MobileNetv2 2. prístup`	96.7	92.8	99.3	95.9
`MobileNetv3 2. prístup`	96.5	92.5	99.0	95.6
`MobileNetv3 3. prístup`	97.9	94.5	99.2	96.8
`DenseNet 3. prístup`	97.1	88.4	97.3	92.6



Obr. 3.6: Matica zámien naučeného modelu `MobileNetv3` 3.prístupom.



Obr. 3.7: Matica zámien naučeného modelu `DenseNet` 3.prístupom.

3.3 Úprava nástroja `CicFlowMeter`

V kapitole 1.2.2 bol krátko spomenutý nástroj `CicFlowMeter`. Ten bol vybraný ako nástroj, ktorý bude zachytávať sieťovú komunikáciu a extrahovať z nej potrebné vlastnosti pre analýzu, pretože má otvorený kód a je vytvorený v programovacom jazyku python. Práve kôli týmto predispozíciám bolo možné nástroj jednoducho upravovať a tak docieľiť požadované správanie.

Zloženie nástroja `CicFlowMeter`

`CicFlowMeter` je rozdelený do viacerých častí, ktoré majú svoju konkrétnu úlohu. Jednotlivé časti sú:

- **setup.py** – jeho úlohov je zkontrolovať verziu pythonu a dostupnosť požadovaných python knižníc *numpy*, *scipy* a *scapy*. Ak je všetko v poriadku, inštalačný proces je inicializovaný. Po dokončení je nástoj plne funkčný a na jeho spustenie je nutné mať oprávnenie používať príkaz *sudo* lebo sú vyžadované oprávnenia root používateľa. Ak je uskutočnená zmena v neakom súbore, ktorý patrí nástroju, tak pre jej aktiváciu je nutné spustiť inštalačný proces znova.
- **sniffer.py** – je koreňový súbor, spušťaný ako prvý. Kontroluje parametre,

definované pri štarte, či sú v správnom tvare. Ak sú v poriadku, spustí podľa nich zachytávač paketov.

- **flow_session.py** – v súbore sú všetky zachytené pakety rozdeľované do individuálnych relácií. Ak je relácia uznačená za ukončenú, je inicializovaný proces výpočtu vlastností relácie. Tie sú následne vložené do CSV súboru.
- **flow.py** – v súbore sú definované všetky vlastnosti, ktoré je možné z relácie získať. Práve v tomto súbore je možné odoberať jednotlivé nepotrebné vlastnosti alebo, ak je to potrebné, definovať vlastné pravidlá pre výpočet.
- **utils.py** – obsahuje pomocné funkcie, potrebné pre správny chod nástroja.
- **features** – je zložka, v ktorej sú obsiahnuté definície tried na výpočet vlastností ako počet IP flagov, počet paketov, veľkosť paketov, časové charakteristiky paketov, určenie smeru prenosu paketu a časové charakteristiky medzi dvomi paketmi.

Vytvorené zmeny v súboroch

Aby nástroj bolo možné používať na účeli zachytávania sieťovéj prevádzky a následnu analízu vyextrahovaný vlastností, boli uskutočnené dve zmeny v kóde nástroja. V prvom zásahu bol zmenený počet vlastností, ktoré sú z relácií extrahované. Úprava bola uskutočnená v súbore *flow.py*, v ktorom sa nachádza slovníková štruktúra s názvom *data*. Tá je tvorená prvkami, ktoré sú definovné ako *kľúč: hodnota*. Kĺúč v tomto prípade udáva názov pozorovaných vlastností, ku ktorému je pripojená odpovedajúca hodnota. V počiatočnom stave sú prvkami odsiahnuté všetky vlastnosti. Ak budú definované nové metody, musia byť počet prvkov rozšírený. V opačnom prípade znižovania počtu, prvky musia byť odstránené alebo zakomentované.

Zo štrukturi boli zakomentované všetky prvky okrem 22 používaných, ktorých názvy sú: Src ip, Dst ip, Src Port, Dst Port, timestamp, Flow Duration, Flow Pkts/s, Tot Fwd Pkts, Fwd Header Len, Flow IAT Mean, Fwd IAT Mean, Fwd IAT Std, Bwd IAT Mean, CWE Flag Count, FIN Flag Cnt, SYN Flag Cnt, RST Flag Cnt, PSH Flag Cnt, ACK Flag Cnt, URG Flag Cnt, ECE Flag Cnt, Pkt Size Avg. Niektoré z nich boli popísané v kapitole číslo 3.1.2, ku ktorým bola pridaná IP adresa a port zdrojovej stanice a IP adresa a port cieľovej stanice.

Druhá úprava je uskutočnená v súbore *flow_session.py*. V ňom je definová trieda *FlowSession*, do ktorej boli definované nové parametre:

- **API_IP** – IP adresa, na ktorej je dostuná služba na detekciu anomálií.
- **API_PORT** – internetový port, na ktorom je služba dostupná.

- **batchSize** – udáva potrebný počet vytvorených súborou, pre zaslanie žiadosti na detekciu anomálií.
- **csvDir** – udáva cestu k zložke, do ktorej budú ukladané CSV súbory.
- **fileID** – udáva identifikačné číslo súboru. Číslo je inkrementované s rastúcim počtom. vytvorených súborov.
- **gid** – systémové identifikačné číslo užívateľskej skupiny.
- **maxLines** – maximálny počet relácií, obsiahnutých v jednom CSV súbore.
- **uid** – systémové identifikačné číslo užívateľa.

Novo definované parametre sú použité v metóde *garbage_collection*, patriacej do rovnakej triedy *FlowSession*. Táto metoda, pri jej zavolaní skontrolovala, či sú vytvorené nové vlastnosti z relácií. Ak tomu tak bolo, všetky nové dáta boli zapísané do jedného CSV súboru. Tento prístup bol nevyhovujúci a preto funkčnosť metódy bolo potrebné pozmeniť. Jedna z uskutočnených zmien je zobrazená vo výpise číslo 3.3.1.

```
if len(self.flows) > self.maxLines:
    path = self.csvDir+str(self.flowID)+".csv"
    output = open(path, "w")
    csv_writer = csv.writer(output)
    for k in keys:
        if self.csv_line >= self.maxLines:
            break
        flow = self.flows.get(k)
        data = flow.get_data()
        if self.csv_line == 0:
            csv_writer.writerow(data.keys())
        csv_writer.writerow(data.values())
        self.csv_line += 1
        del self.flows[k]
    self.filesCreated.append(str(self.flowID))
    self.flowID += 1
    self.csv_line = 0
    output.close()
    os.chmod(path, 0o660)
    os.chown(path,self.uid,self.gid)
```

Výpis 3.3.1: Definícia funkcie na obmedzenie veľkosti CSV súboru.

Ako prvý krok je kontrola počtu zachytených relácií. Počet musí byť väčší ako maximálny hodnota v premennej *maxLines*. Ak je podmienka splnená, je možné

vytvoriť nový súbor a zapísať do neho zhromaždené relácie. Súbor je vytvorený v zložke definovanej premennou *csvDir* a jeho názov je postupne inkrementovaný identifikátor s názvom *flowID*, ku ktorému je pridružená prípona pre CSV súbory. Identifikátor je inkrementovaný až po úplnom naplnení súboru dátami. Hodnoty sú periodicky zapisované po riadkoch, až kým nie je dosiahnutý ich požadovaný počet. Prvý riadok obsahuje názvy všetkých vlastností špecifikovaných v slovníkovej štruktúre *data*. Po názvoch nasledujú odpovedujúce hodnoty daných vlastností jednotlivých relácií, ktoré sú následne odstráňované z celkového zoznamu aby nedošlo k duplikovaní. Po naplnení súboru dátami sú ešte nastavené jeho prístupové oprávnenia. Hodnoty pre prístup jednotlivých skupín je nastavená na 660, čo znamená, že vlastník a skupina má možnosť zápisu a čítania ale ostatný používatelia nemajú k súboru prístup. Po nastavení oprávnení je nastavený vlastník súboru, špecifikovaný premennou *uid* a skupina, do ktorej patrí premennou *gid*.

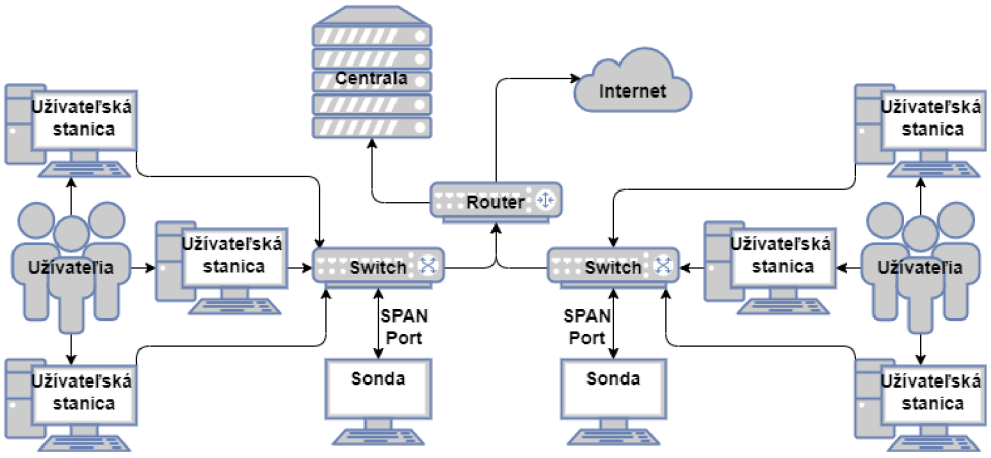
V druhom kroku je kontrolovaný počet vytvorených súborov. Ak je zhodný s nastavenou hodnotou premennej *batchSize*, je zaslaná žiadosť na ich kontolu pre možný výskyt anomálií. Žíadosť je zaslaná na adresu a port štecifikovaný v konštantách *API_IP* a *API_PORT*. K žiadosti sú priložené identifikátory jednotlivých súborov oddelené čiarkami v HTTP parametry s názvom *csv*.

4 Programová implementácia detekcie anomálií

Táto kapitola je zameraná na popis funkčných prvkov celého procesu zachytávania a detekcie anomálií. V prvej časti budú popísané jednotlivé komponenty z hľadiska funkčného zastúpenia v procese a ich vzájomné prepojenia. Dôraz bude kladaný na správanie sa systému ako celku. Po predstavení celej detekčnej architektúri, budú detailnejšie popísane všetky fuknčné časti s pohľadu programovej implementácie, procesu rozhodovanie a využitie komponentov popísaných v kapitole 3. V poslednej časti bude popísaný proces zálohovania a manažmentu ukladacieho priestoru.

4.1 Popis súčastí detekčnej architektúri

Systém detekcie anomálií je navrhnutý ako dvojvrstvový centralizovaný systém, zložený so sieťovej sondy a centrály. Sieťová sonda je zapojená do komunikačnej siete a prijíma pakety všetky pakety, ktoré sú do nej zasielané. Po ich prijatí sú spracovávané podľa patričných relácií a ak ich počet prekročí určitú hranicu, vyextrahujú sa z nich vlastnosti a pošlú sa na analýzu. Počas analýzy sú z vlastností vytvorené obrázky, podľa ktorých sa určí výskyt anomálie v sieti. V prípade, že výsledok je pozitívny, obrázok je zaslaný na dodatočnú analýzu do centráli. V nej je overený výsledok sondy a v prípade, že výsledok sa potvrdí, je pristúpené k detailnejšej identifikácií anomálie.



Obr. 4.1: Ukážka možnosti zapojenia sondy a centrály do sieťovej infraštruktúri.

Na obrázoku číslo 4.1 je znázornená ukážka zapojenia sondy a centráli do sieťovej infraštruktúri. Užívateľské stanice sú pripojené sa prepínač s nastaveným SPAN portom, do ktorého je pripojená sonda. Tým je zabezpečené zrkadlenie všetkej užívateľskej komunikácie do portu s pripojenou sondou. Prepínače sú pripojené so smerovačom, ktorý má prístup do internetu a je prepojený taktiež s centrálou. Takto môže sonda komunikovať s centrálov pri zistení možných výskytov anomálií.

4.2 Návrh sieťovej sondy

Návrh sieťovej sondy spočíval vo výbere hardvérových častí a vytvorení vývojového diagramu detekcie anomálií hlbokov neurónovou sieťov, podľa ktorého bola vytvorená softvérová implementácia. Ako prvé budú popísané vybráné hardvérové časti.

4.2.1 Hardvérové požiadavky sieťovej sondy

Raspberry Pi

Raspberry Pi sú nazývané jednodoskové počítače, vytvárané spoločnosťou *Raspberry Pi Foundation*[1] v Spojenom kráľovstve za účelom vzdelávania ľudí v počítačových vedách a zjednodušenia prístupu k takému to vzdelávaniu. Od jeho vzniku v roku 2012 bol vytvorených mnoho variácií, líšiacich sa veľkosťou, funkcionalitou a výkonom ale ja napriek tomu sú zachované základné požiadavky. Medzi ne patrí nízka cena počítača s možnosťou inštalácie operačné systému Linux a dostupnosť sady pinov všeobecného vstupu/výstupu GPIO (*General Purpose Input/Output*), ktoré umožňujú kontrolovanie elektrických komponentov a vytváranie nástrojov pre internet vecí IoT (*Internet of Things*[2]). Na obrázku číslo 4.2 je zobrazený počítač raspberry pi 4 s dostupnými vstupmi a výstupmi.[29]

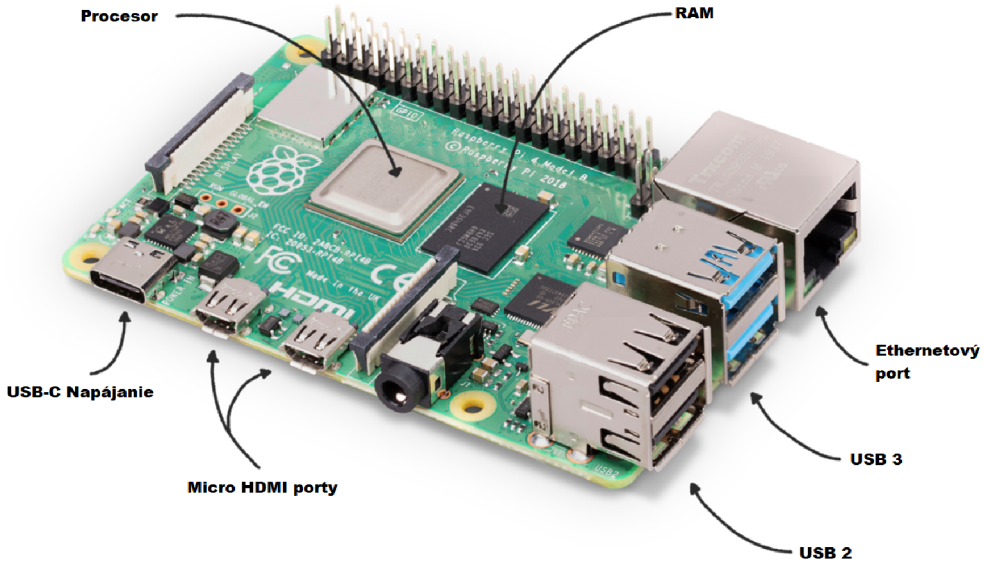
Pre potreby práce bolo vybrané rapberry pi s poradovým číslom modelu 4, ktorý je v čase písania práce ten najnovší vydaný. Počítač je vybavený 8 GB pamäťou RAM, gigovým ethernetovým portom a adaptérom wifi. Preto bude použitý ako výpočetná jednotka pre softvérovú implementáciu sondy.

Neural Compute Stick 2

Neural Compute Stick 2, ďalej len NCS, je malý USB modul od spoločnosti Intel Movidius. Je vytvorený na najnovšej jednotke vizuálneho spracovania Intel Movidius

[1] Viac informácií sú dostupné na: `https://www.raspberrypi.org/about/`.

[2] Viac informácií je možné nájsť na: `https://link.springer.com/content/pdf/10.1007/s12599-015-0383-3.pdf`.



Obr. 4.2: Ukážka počítača raspberry pi s popisom hlavných častí.

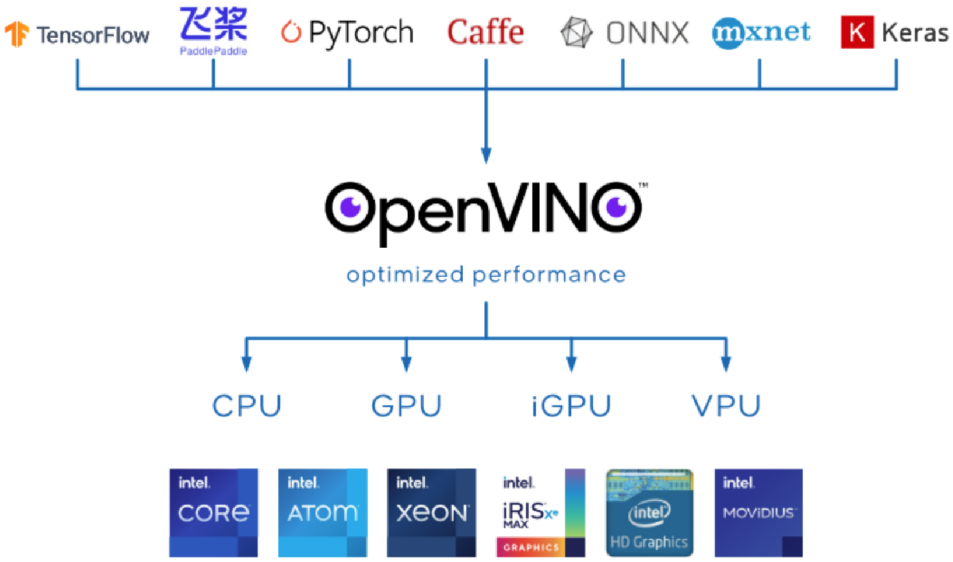
MyRiad X VPU (*visual processing unit*), čo je hardvérový urýchľovač pre hlboké umelé neurónové siete. Modul funguje ako koprocesor pre počítače s operačným systémom Windows, macOS alebo Linux. NCS je taktiež kompatibilné so zariadením raspberry pi a funguje v režime offline, čiže nemusí mať prístup k dodatočnej externej výpočetnej technike.[30]

Jeho výhodov je používanie nástroja OPENVINO (*open visual inference and neural network optimization*). Je to nástroj s otvoreným kódom a má za úlohu optimalizovať a uľahčiť nasadenie modelu nuerónovej siete do prevádzky. Zvýšuje výkon modelov hlbokého učenia pre počítačové videnie, automatizované rospoznanie reči, spracovanie prirodzeného jazyka a ostatné činnosti využívajúce hlbokú neurónovú sieť. V nástoji môžu byť používané predtrénované modely, ktoré boli vytvorené rôznymi populárnymi softvérovými rámcami, ako tensorflow, pytorch, keras a ostatné. Na obrázku číslo 4.3 je možné vidieť všetky podporované softvérové rámce a výpočetné jednotky. Medzi podporovanými výpočetnými jednotkami sú počítačové procesory CPU (*central processing unit*) a grafické karty GPU (*graphics processing unit*) od firmy Intel. Ale taktiež je možné použiť výpočetnú jednotku VPU, ktorá sa nachádza v module NCS.[31]

4.2.2 Softvérové požiadavky sondy

Do vybraného raspberry pi je nutné ako prvý krok nainštalovať operačný systém. Bol vybraný systém s názvom raspbian[3], ktoreho základ je linuxová distribúcia

[3]Viac informácií je možné nájsť na: `https://www.raspbian.org/`.



Obr. 4.3: Ukážka softvérových rámcov a výpočetných jednotiek podporovaných nástrojom openvino.[31]

debian optimalizovaná pre hardvér raspberry pi. Po inštalácií bola uskutočnená základná konfigurácia, ako aktualizácia softvéru, vytvorenie nového používateľa, vytvorenie novej používateľskej skupiny s názvom *sonda*. Pre túto skupinu budú vytvárané CSV súbory z nástroja `CICFlowMeter`. Preto je do skupiny pridaný novo vytvorený používateľ a užívateľ root. Následne je spustená inštalácia potrebných systémových balíkov na vytváranie python virtuálneho prostredia. Kôli tomu, že je vyžadované používať nástroj openvino verzie 2021.2 musí byť používaná verzia programovacieho jazyka python 3.7. Ak táto požiadavka splnená nie je, nástroj v čase písania práce nie je možné nainštalovať.

Po skončení predošlích úkonov sú inštalované nástroje `CICFlowMeter` a openvino. Ako prvá bola inštalovaná upravená verzia nástroja `CICFlowMeter`. Nástroj musí byť spúšťaný s právami root-a, preto s rovnakými právami musí byť aj inštalovaný. Proces začal skontrolovaním dostupnosti potrebných python závislostí a prípadným doplnením, za ktorým nasledovala samotná inštalácia nástroja. Openvino nepotreboval kontrolovať žadné dodatočné python závislosti. Postup inštalácie obsiahlejší a je zdokumentovaný v openvino dokumentácií[4]

Ako posledný krok je vytvorenie python virtuálneho prostredia a doplnenie potrebných závislostí. Preto bolo vytvorené prostredie s názvom *app*, do ktorého boli pridané moduly ako:

[4]Viac informácií je možné nájsť na:`https://docs.openvino.ai/latest/openvino_docs_install_guides_installing_openvino_raspbian.html`.

- **flask** – slúži na vytváranie webových aplikácií
- **pandas** – slúži na manipuláciu a analýzu dát
- **redis** – používa sa pre komunikáciu s úložiskom dátových štruktúr v pamäti s názvom redis
- **torchvision** – slúži na úpravu dát pred vstupom do neurónovej siete
- **scikit-image** – slúži na prácu a manipuláciu s obrázkami

4.2.3 Detekcia anomálií hlbokou neurónovou sieťov

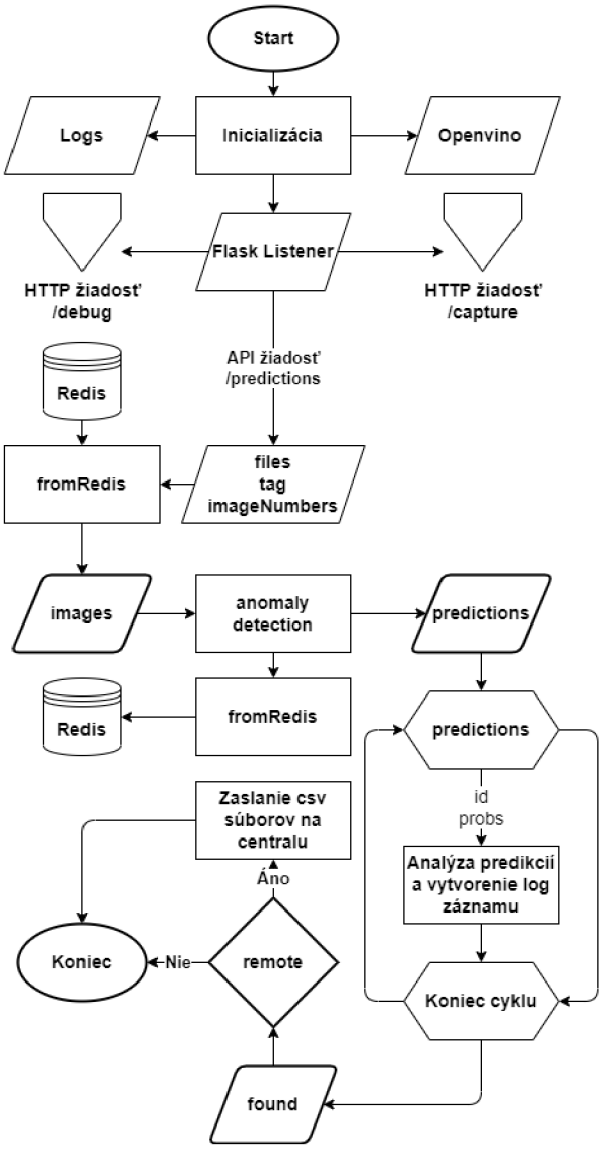
Pre automatickú detekciu anomálií hlbokov neurónovou sieťov bola vytvorená webová aplikácia pomocou python knižnice *flask*[5]. Tá, po jej počiatočnej inicializácií prijíma API[6] žiadosti, ktoré sú spracované a podľa žiadosti je uskutočnená požadovaná akcia. Pre potreby sondy boli vytvorené vo webovej aplikácií tri poskytované služby, na ktoré je možné vytvoriť žiadosť. Sú to */debug*, */capture*, ktoré predpripravujú poskytnuté dáta na predikciu a */predictions* uskutočňujúci predikciu na upravených dátach.

HTTP žiadosť na API */predictions*

Funkciou žiadosti predictions je vytvoriť predikciu hlbokov neurónovou sieťov na vopred upravených dátach. Tie sú určené v parametri žiadosti ako identifikátory oddelené čiarkov. Žiadosť môže byť prijatá s parametrom *csv* alebo *imgs*. Názov parametru udáva počiatočný formát prijatých dát, ktoré môžu byť CSV súbory pre detekciu na živej sieťovej komunikácií alebo obrázky pre testovanie funkčnosti aplikácie.

Na obrázku číslo 4.4 je zobrazený vývojový diagram, znázorňujúci proces inicializácie potrebných závislostí a následnej predikcie. Počas inicializácie je definovaná trieda *Logs*, ktorá je využívaná na vytváranie logovacích záznamov počas behu aplikácie. Po sprístupnení možnosti vytvarania logov je frameworkom *openvino* nahraná natrénovaná neurónová sieť `MobileNetV2`. Použitá neurónová sieť bola trénovaná na dátovej sade obsahujúcej obrázky so styrmi pixelmi pre reláciu. Po dokončení inicializácie je vytvorený log záznam s výsledkom nahrávania modelu a ak proces bol úspešný je možné využívať model na vytváranie predikcií. Ako posledný krok inicializácie je spustenie *flask* webovej aplikácie. Tá je nastavená aby bola dostupná iba na IP adrese 127.0.0.1 na porte 5001. Tým je zabezpečená dostupnosť iba na stanici, na ktorej je aplikácia spustená.

[5]Bližšie informácie sú k dispozícií na: `https://flask.palletsprojects.com/en/2.0.x`.
[6]Rozhranie pre programovanie aplikácií *(Application Programming Interface)*.



Obr. 4.4: Vývojový diagram API žiadosti */predictions*.

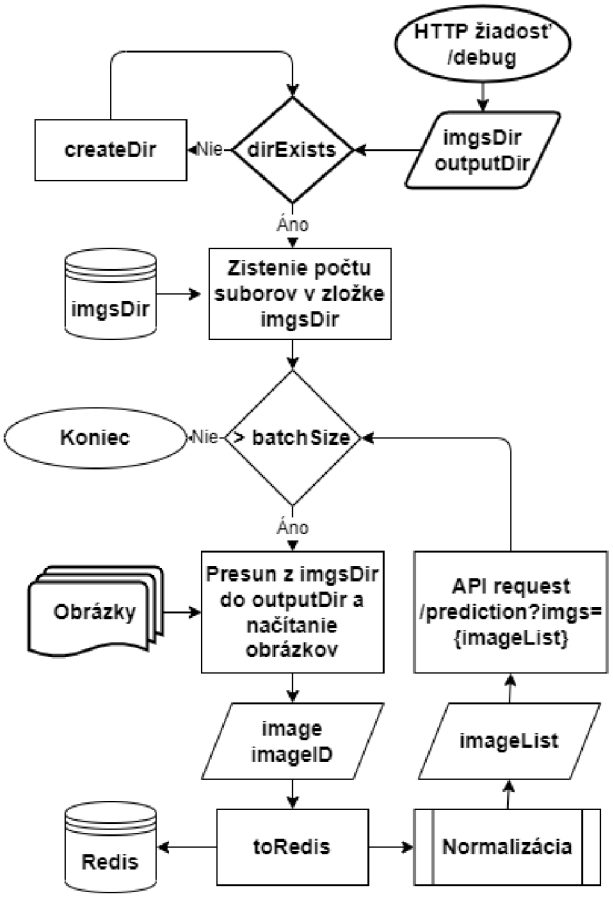
Po inicializácií je možné vytvárať žiadosti na cestu debug (popísanú v časti 4.2.3), capture (popísanú v časti 4.2.3) a predictions. Po prijatí žiadostí na cestu prediction je spracovaný pridružený parameter s prijatou hodnotou. Tá je predaná funkcií *startDetection* na spracovanie. Z hodnoty sú extrahované identifikátori obrázkov, ktoré slúžia na získanie dát z programu `Redis`[7]. Tie sú použité ako vstup do vopred inicializovaného modelu vo frameworku *openvino*. Výstupom z modelu je pole s vypočítanými hodnotami. Tie však nie sú smerodajné, preto použitím funkcie softmax sú upravené na pravdepodobnostné hodnoty, ktorých súčet je rovný jednej. Maximálna pravdepodobnosť je z poľa vyňatá pre určenie výsledkej kategórie

[7]Bližšie informácie sú k dispozícií na: `https://redis.io/docs/about/`.

vstupného obrázku. Výsledok je zaznamenaný do logu ako benign ak má kategória hodnotu 0 alebo ako anomaly ak má hodnota kategóriu 1. Výsledok je možné zaslať na dodatočnú kontrolu na centrálu. To je dosiahnuté zaslaním patričných CSV súborov, vytvorením API žiadosti na webovú aplikáciu, ktorá sa nachádza na centrále. Posledný úkon je vymazanie použítých dát z programu `Redis`, použitím ich identifikátorov, rovnako ako pri ich získavaní.

HTTP žiadosť na API */debug*

Funkciou žiadosti debug je overiť funkčnosť programu so sadou vopred predpripravených obrázkov. Tie musia byť najskôr upravené do požadovanej podoby, aby bolo možné uskutočniť predikciu hlbokov neurónovou sieťov.



Obr. 4.5: Vývojový diagram API žiadosti */debug*.

Na obrázku číslo 4.5 je zobrazený vývojový diagram, na ktorom sú znázornené uskutočňované kroky úpravy obrázkov. Proces je započatý prijatím žiadosti bez dodatočných argumentov. Nasledovaný overením existencie zložiek s obrázkami *imgsDir* a zložka *outputDir*, do ktorej sú skontrolované obrázky presunuté. Ak niektorá

nie je dostupná, v zápätí je vytvorená. Ak sa v zložke *imgsDir* vyskytujú obrázky, je získaný ich počet. Ak je ich počet väčší alebo rovný veľkosti hodnoty *batchSize*, obrázky o danom počte sú vybrané na spracovanie. Najskôr sú presunuté do zložky *outputDir*, z ktorej je načítané, normalizované a je im preradený unikátny identifikátor. Normalizované obrázky s ich identifikátormi sú vložené do programu `Redis`. Týmto úkonom je odstránené spomalie spôsobené ukládaním a nasledným načítaním obrázkov z disku pri ich predávaní. `Redis` ukladá dáta do pamäte RAM, čím je zaistený rýchleší prístum k uloženým dátam. Po uložení je vytvorená API žiadosť na vytvorenie predikcie, ktorá zahrňuje desať identifikátorov oddelených čiarkov v parametri *imgs*.

Postup presunu obrázkov, normalizácie, priradzovania identifikátorov a ukladania do programu `Redis` je opakovaný až kým v zložke imgsDir sa nenachádzajú žiadne obrázky alebo je ich počet menší ako hodnota *batchSize*.

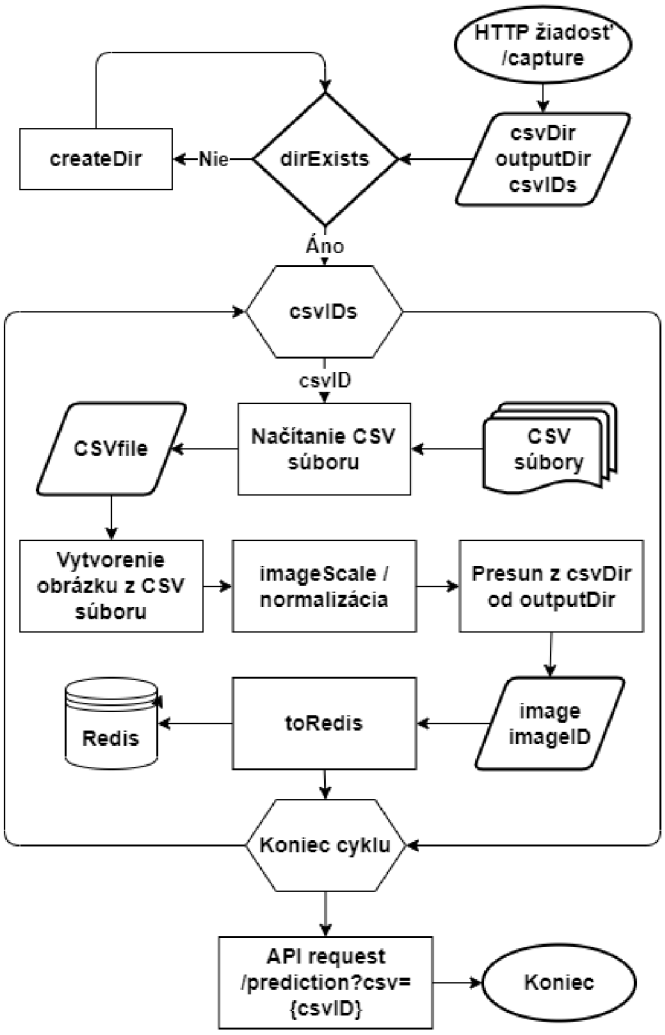
HTTP žiadosť na API */capture*

Funkciou žiadosti capture je príjímať dáta na predikciu, vytvorené zachytávaním živej sieťovej komunikácie. Na túto službu zasiela žiadosti aj program `CicFlowMeter` s desiatimi identifikatormi CSV súborov v parametri *csv*.

Na obrázku číslo 4.6 je zobrazený vývojový diagram, na ktorom je znázornený proces úpravy a transformácie CSV súborov na obrázky. Rovnako ako pri žiadosti debug je po prijatí žiadosti skontrolovaná existencia zložiek *csvDir* a *outputDir*. V zložke *csvDir* sú ukladané CSV súbory programom `CicFlowMeter`. Ak je všetko v poriadku, je započatý proces transformácie. Z prijatého parametru sú vyextrahované identifikátory súborov, podľa ktorých sú načítané korešpodujúce súbory zo zložky. Procesom transformácie sú z CSV dát vytvorené obrázky, ktoré kôli optimalizácií vstup do neurónovej siete musia byť taktiež aj normalizované. Použité súbory sú presunuté do zložky *outputDir* a vytvorené obrázky sú vložené do programu `Redis` s rovnakým identifikátorom, aký má príslušný CSV súbor. Po dokončení transformácie je vytvorená API žiadosť na vytvorenie predikcie, ktorá zahrňuje desať identifikátorov oddelených čiarkov v parametri *csv*.

4.2.4 Programová štruktúra sieťovej sondy

Vytvorený program je zložený z viacerých častí, nazývaných moduly a jedného hlavné spustiteľného súboru. Dôvodom, prečo bolo rozdelenie programu na menšie časti je zvýšenie prehliadnosti programu a zlúčenie častí programu do skupín, ktoré plnia rovnakú alebo podobnú funkcionalitu. Tým je zabezpečená možnosť jednoduchej úpravy v prípade nutnosti zmeny niektorej časti programu a redukcia možnosti pre vznik chyby počas implementácie.



Obr. 4.6: Vývojový diagram API žiadosti */capture*.

Takto vytvorený program je tvorený súbormi, ktoré sú:

- **app.py** – je hlavný spustiteľný súbor. Jeho úlohov je inicializovať všetky hlavné programové súčasti ako logovanie, komunikáciu s programom `Redis`, *flask* webovú aplikáciu a model neurónovej siete pomocou frameworku *openvino*. Zároveň sú vytvorené API cesty s definovanými funkciami, ktoré budú vykonané pri ich zavolaní s požadovanými parametrami.
- **config.py** – v tomto súbore sú definované konštantné parametre, používané počas celého behu programu. Nastavuje sa nimi správanie programu, ako zakázanie zasielanie súborov na dodatočnú kontrolu na centrálu, povoliť rozšírený výstup v kozole alebo priloženie do logových záznamov čas trvanie učitých udalostí. Následne sú určené názvy používaných zložiek, názov súborov s modelom pre *openvino*, veľkosť dávky, rozmery a zloženie obrázkov po transformácií, IP

adresy všetkých používaných služieb spolu s ich číslami portov a nastavenie logov. Formát logových zaznamov je zložený z časového razítka, názvu zariadenia, stupeň vážnosti záznamu a správa, ktorá je špecifická pre jednotlivé udalosti ale ku každej správe je pridružená IP adresa zariadenia. Nakoniec je definovaný slovník, mapujúci kategóriu analyzovanéj komunikácie na poradové číslo a naopak.

- **filePreprocessing.py** – funkciou modulu je spracovanie vstupných dát od požadovaného tvaru pred vstupov do neurónovej siete. Preto obsahuje dve funkcie, ktoré sú určené na spracovanie vstupných dat v obrázkovom formáte alebo vo formáte CSV. Koncové úkony majú dané funkcie pomerne rovnaké a to uloženie upravených dát a zaslanie žiadosti na predikciu spoločne s ich identifikátormi.
- **imageTransformation.py** – modul slúži na transformáciu CSV súborov na obrázky. Obsahuje triedu *Transformation* s tromi verejnými funkciami, ktoré sa líšia počtom výstupných obrázkov a prípadných súbežne vytváraných súborov. Pre potreby sondy je využívaná funkcia, vytvárajúca jeden obrázok zo vstupného CSV súboru. V prípade potreby je možné vytvoriť z jedného CSV súboru viac ako jeden obrázok ak je v súbore dostatočný počet dát. Na tento účel je využívaná odlišná funkcia, ktorá je určená na tvorbu skupiny obrázkov. Posledná vstupná funkcia je navrhnutá za účelom vytvárania dátovej sady, používanej na trénovanie neurónovej siete. Pre úslešné využívanie funkcie je nutné vytvoriť dodatočnú vlastnosť, v ktorej sú jednotlivé zachytené relácie osnačené. Značka vyjadruje účel danej relácie, poprípade konkretizuje použítý útok.
- **netRB.py** – v module je definovaná trieda *Classify*, používaná na vytváranie predikcií. Pri inicializácií objektu je pomocou frameworku *openvino* načítaný model, špecifikovaný v súbore *config.py* spolu s ostatnými konštantami ako veľkosť dávky, názov zariadenia na akceleráciu predikcií a názvy používaných zložiek. Trieda obsahuje jednu verejnú funkciu a dve privátne funkcie. Verejná funkcia je volaná pre vytvorenie predikcií uložených v programe `Redis`. Prvá definovaná funkcia je používaná na úpravu vytvorených predikcií a následné zaznamenanie výskytov. Druhá privátna funkcia je tvorená algoritmom softmax, používanej na úpravu hodnôt predikcií.
- **utilities.py** – je modul, v ktorom je definovaná trieda s názvom *Logs* a pomocné funkcie. Trieda *Logs* nastavuje všetky súvislosti, späté s vytváraním logových záznamov. Ostatné funkcie, ktoré sa nachádzajú v module sú používané na vytváranie zložiek, premenovanie súborov, získavanie a vymazávanie dát z programu Redis a funkcia na vytvorenie API žiadosti na centrálu s priložením CSV súborov.

4.2.5 Správa súborov sieťovou sondou

Je predpokladané, že sonda bude zachytávať sieťovú komunikáciu a z nej vytvárať CSV súbory po pomerne dlhé časové obdobie. Preto sa môže vyskytnúť prípad zahltenia uloženého priestoru a tým negatívne ovlivniť výkon sondy. Preto boli vytvorené dva skripty, ktoré sú určené na zabránenie výskytu tento udalosti.

Prvým skript obsahuje príkaz na zálohovanie novo vytvorených súborov do vopred definovaného úložného miesta. Miesto, do ktorého sú súbory zálohované sa nachádza v centrále. Práve pre tento účel bol vybraný program *rsync*[8], využívaný na synchronizáciu súborov medzi dvomi stanicami. V základnom móde sú dáta prenášané komunikačným kanálom nešifrovane. To je vyriešené použitím služby ssh[9], ktorá vytváranú komunikáciu šifruje a autentizuje sondu voči centrále. Pri použití overovania pomocou mena a hesla nebolo možné proces automatizovať, preto bol zvolený prístup autentizácie pomocou certifikátu. Z toho dôvodu bol na strane sondy vygenerovaný verejný a súkromný kľuč a verejný kľúč bol zaslaný a uložený do centráli. Tým je umožnená automatická autentizácia za pomoci privátneho kĺúča. Takto vytvorený skript je programom *cron*[10] spúšťaný každý deň o polnoci.

Druhý skript má za úlohu zabrániť zahĺteniu úložného priestoru. Pri štarte sú požadované tri parametre: cesta k sledovanej zložke, možné percentuálne obsadenie celkového úložného priestoru a počet súborov, odstránených počas jedného cyklu. Po štarte je získaná celková veľkosť úložného priestoru a veľkosť súborov, ktoré sa nachádzajú v sledovanej zložke. Ak veľkosť súborov má väčšie percentuálne obsadenie ako to, ktoré bolo zadané vo vstupnom parametri, súbory s najstarším časovým razítkom začnú byť odstraňované. Počet odstraňovaných súborov bol zadaný vo vstupnom parametri. Následne je overené percentuálne obsadenie súborov v zložke, rovnako ako v predchádzajúcom kroku. Ak veľkosť prevyšuje požadované percentuálne obsadenie je započaté odstráňovanie súborov. Odstráňovanie pokračuje dokým obsadenie nie je pod požadovanou hranicou a ak ju prekročí, skript je ukončený. Spúšťanie skriptu je automatizované taktiež programom *cron* kažký deň, 30 minút po vykonaní zálohy.

4.3 Návrh centrály

Návrh centrály spočíval vo výbere hardvérovej časti, ktorá podporuje akceleráciu výpočtu predikcií hlbokov neurónovou sieťov. Po výbere bol upravený vývojový diagram sieťovej sondy, aby proces odpovedal požadovanej funkcionalite centrály

[8]Bližšie informácie sú k dispozícií na: `https://linux.die.net/man/1/rsync`.

[9]Bližšie informácie sú k dispozícií na: `https://www.ssh.com/academy/ssh`.

[10]Bližšie informácie sú k dispozícií na: `https://man7.org/linux/man-pages/man5/crontab.5.html`.

a podľa uskutočnených zmien upraviť softvérovú implementáciu. Ako prvá bude popísaná hardvérová časť centráli.

4.3.1 Hardvérové a softvérové požiadavky centrály

Rovnako ako pri sieťovej sonde aj centrále je nutné použiť externé zariadenie, ktoré urýchli výpočet predikcií. Presne preto bola použitá grafická karta NVIDIA GTX 1050. Táto grafická karta podporuje proprietárnu technológiu `CUDA` (*Compute Unified Device Architecture*), ktorá je používaná na urýchlenie paralelnych výpočtov.

`CUDA` je paralelná výpočtová platforma a model API, ktorý vyvinula spoločnosť Nvidia. Pomocou `CUDA` možno využiť výkon grafických procesorov Nvidia na vykonávanie všeobecných výpočtových úloh, ako je násobenie matíc a vykonávanie operácií lineárnej algebry, namiesto vykonávania len grafických výpočtov. Pomocou CUDA sú urýchlené výpočtovo náročné aplikácie, ktoré využívajú výpočetnú silu GPU na spracovanie paralelizovateľných úkonov. Takýto prístup k riešeniu problémov bol prijatý v mnohých oblastiach, ktoré využadujú vysoký výpočetný výkon pri výpočtoch s pohyblivou rádovou čiarkov. Príklady takýchto odvetvý sú: pri práci s počasím, dátová veda a analýza, hlboké učenie a strojové učenie, obrana a spravodajstvo a množstvo iných.[32]

Po výbere GPU nasledovala inštalácia potrebných závislostí. V prvom rade bolo nutné nainštalovať ovládač grafickej karty a nástroj `CUDA`, ktorý je pridávaný osobytne. Po overení správneho postupu, viditeľnov existenciou ovládača a nástroja `CUDA` bol nainštalovaný programovací jazyk *python* so všetkými potrebnými knižnicami. Najpodstatnejšie z nich sú:

- **flask** — slúži na vytváranie webových aplikácií
- **pandas** — slúži na manipuláciu a analýzu dát
- **pillow** – slúži na prácu a manipuláciu s obrázkami
- **torch** – je framework pre prácu s modelmi neurónových sietí
- **torchvision** — slúži na úpravu dát pred vstupom do neurónovej siete

4.3.2 Detekcia anomálií hlbokou neurónovou sieťov

Rovnako ako pre sieťovú sondu, tak aj pre centrálu bola vytvorená webová aplikácia pomocou python knižnice *flask*. Tá po počiatočnej inicializácií čaká na zaslané HTTP žiadosti na API aplikácie. Na rozdiel od sieťovej sondy, ktorá prijímala žiadosti iba na vnútornú smyčku, aplikácia centráli je dostupna jej verejnej adrese na porte 5000. Pre potreby centráli boli vytvorené 3 poskytované služby vo forme API.

Služba dostupne na ceste /csvPredict je primárne volaná sieťovov sondov pre dodatočnú predikciu na CSV súboroch. Na cestu /imgPredict sú zasielané predpripravené obrázky pre overenie funkčnosti aplikácie a na cestu /predictions je vytváraná žiadosť lokálne na vytvorenie predikcií z prijatých CSV súborov.

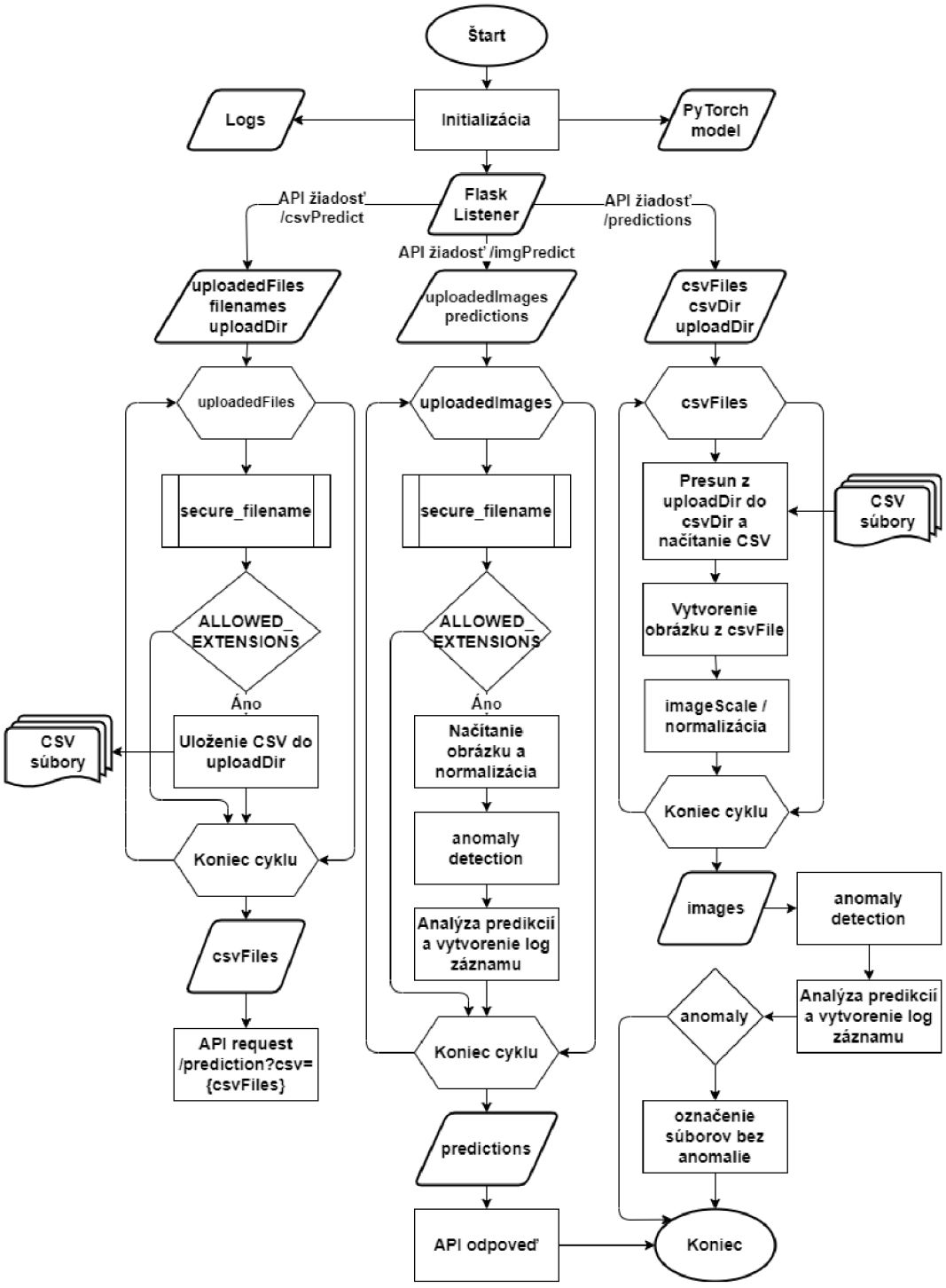
Na obrázku číslo 4.7 je znázornený vývojový diagram webovej aplikácie, znázorňujúci počiatočnú inicializáciu a procesy, ktoré sú uskutočňované po prijatí HTTP žiadostí na jednu z API. Rovnako ako pri inicializácii aplikácie sieťovej sondy je najprv vytvorený spôsob vytvárania logov. Za ním nasleduje nahratie naučeného modelu hlbokej nurónovej siete a jeho sprístupnenie pre možnosť vytvárania predikcií. Ako posledný úkon pred započatím používania je spustenie samotnej webovej aplikácie s preddefinovanými cestami. Po tomto kroku ak všetko prebehlo úspešne, je možné začať spracovávať prichádzajúce HTTP žiadosti.

HTTP žiadosť na API */csvPredict*

Funkciou žiadosti csvPredict je prijatie a uloženie prichádzajúcich súborov. Tie sú zasielané sondami na dodatočnú kontrolu. Bohužial na centrálu možu zasielať súbory aj iné entity ako sú sondy, z dôvodu prístupnosti centráli aj z inej siete ako je vnútorná smyčka. Práve preto je nutné overiť, či sa jedná o CSV súbory a nepovolené formáty zahadzovať. To je dosiahnuté prvotným očistením názvu súboru pomocou funkcie *secure_filename*, ktorá vráti vstupný názov bez nepovolených skupín znakov a medzier. Z názvu je následne extrahovaná prípona súboru a porovnaná so zoznamom povolených prípon. Ak sa vyskytne zhoda, súbor je uložený do zložky s nahrávanými súbormi. A ak sa zhoda nevyskytne, súbor je z bezpečnostých dôvodov zahodený. Po spracovaní všetkych prijatých súborov je vytvorená HTTP žiadosť na API predictions s názvami uložených súborov oddelených čiarkov v parametri *csv*.

HTTP žiadosť na API */imgPredict*

Funkciou žiadosti imgPredict je prijatie obrázkov, slúžiacich na otestovanie funkčnosti aplikácie. Rovnako ako v kapitole číslo 4.3.2 sú názvy prijímaných súborov skontrolované. Povolené prípony súborov sú *jpeg* a *png*. V prípade, že sa prípona nezhoduje z povolenými, súbor je zahodený. Ak sa vyskytne zhoda, obrázok je načítaný a normalizovaný, čím je pripravený na vstup do modelu. Ešte pred samotným vstupom sú normalizované dáta obrázku vložené do pamäte grafickej karty, čo je vyžadované pri využívaní nástrojá `CUDA`. Ak je podmienka splnená, za pomoci modelu je vytvorená predikcia, ktorá je následne funkciou *softmax* prepočítaná do pravdepodobnostných hodnôt. Tie sú zoradené od najväčšej po najmenšiu a vyňatá je prvá hodnota v poradí spolu s príslušnou kategóriou. Číslo jednotlivých kategórií je následne namapované na odpovedajúci názov kategórie a spolu s pravdepodobnostnou



Obr. 4.7: Vývojový diagram webovej aplikácie spúšťanej na centrále.

hodnotou je vytvorené logový záznam. Do logu je taktiež k spomínaným hodnotám pridaný aj názov súboru pre lepšiu spätnú dohľadateľnosť záznamu. Tento proces je opakovaný na všetkých prijatých obrázkoch, z ktorých sú všetky upravené predikcie zoskupené a zaslané ako HTTP odpoveď na pririjatú žiadosť.

HTTP žiadosť na API */predictions*

Funkciou žiadosti predictions je trasformácia CSV súborov na obrázky a vytvorenie predikcií. Táto žiadosť je volaná primárne cez vnútornú smyšku po prijatí a uložení CSV súborov v centrále. Prijatá žiadosť musí obsahovať parameter *csv* s identifikátormi súborov, ktoré sú určené na spracovanie. Ako prvý krok je transformácia CSV súborov na obrázky. To je uskutočnené načítaním jednotlivých súborov a pomocou transformácie relácií na pixely sú vytvárané skupiny obrázkov. Je predpokladané, že súbory pochádzajú zo sondy, na ktorej bol vytvorený iba jeden obrázok na súbor. Na centrále to neplatí. Z dôvodu používania viacerých vlastností relácie má jedna relácia dvojnásobné pixelové zastúpenie. Preto na centrále je vytváraný dvojnásobný počet obrázkov ako na sonde z rovnakého počtu CSV súborov. Po transformácií je vytvorená sada obrázkov normalizovaná a pripravená na vstup do modelu. Vytváranie predikcií je skro rovnaké ako v kapitole číslo 4.3.2. Líšia sa iba posledným procesom, v ktorom nie je zasielaná odpoveď na žiadosť ale niektoré súbory sú označované ako falošne pozitívny. To nastane v prípade, že súbor v súbore nie je odhalená anomália.

4.3.3 Programová štruktúra centrály

Vytvorený program je zložený z viacerých častí, nazývaných moduly a jedného hlavné spustiteľného súboru. Základom bola programová štruktúra sondy popísaná v kapitole číslo 4.2.4. Počet a účel jednotlivých modulov zmenený nebol. Funkčná stránka niektorých z modulov bola vyhovujúca pre potreby centrály, preto do ich obsahu zasahované nebolo. Na druhú stranu, zásah do obsahu súborov vyžadujúcich zmenu má buď minoritný charaktrer alebo bolo nutné zmeniť kompletný obsah upravovanej funkcie.

Upravované súbory a zmeny v ich obsahu sú:

- **app.py** – je hlavný spustiteľný súbor. Jeho úlohov je inicializovať všetky programové závyslosti využiané počas behu programu. Ako prvé je spustené vytváranie logavých záznamov, nasledovaný sprístupnením modelu neurónej siete na vytváranie predikcií pomocou frameworku *pytorch*. Ak všetko prebehlo úspešne je definovaná *flask* webová aplikácia. Do konfigurácie je vložená cesta zložky, do ktorej bude aplikácia ukladať prijaté súbory. Vytvorené sú aj spomínané API cesty s ich korešpodujúcimi funkciami.
- **config.py** – v tomto súbore sú definované konštantné parametre, používané počas celého behu programu. Pre potreby centrály boli pridané zoznamy prípon, ktoré sú povolené pre prijímané súbory. Boli odstránené konštanty, používané pre prácu s frameworkom *openvino* a boli nahradené konštantami pre

framework *pytorch*. Tie definujú názov použitého predtrénovaného modelu a zariadenie, v ktorom bude predikcia uskutočňovaná. Ako posledné zmeny boli uskutočnené v hodnotách konštánt. Počet pixelov na reláciu bol zmenený z 4 na 8, dostupnosť webovej aplikácia bola zmená zo spätnej smyčky na hodnotu 0.0.0.0[11], port webovej aplikácie z 5001 na 5000 a názvu zariadenia pre logové záznamy.

- **filePreprocessing.py** – funkciou modulu je spracovanie vstupných dát od požadovaného tvaru pred vstupov do neurónovej siete. Zmenou v súbore je odstránenie predprípravy obrázkov a úprava predprípravy CSV súborov. Tá spočíva vo využití odlišnej funkcie na transformáciu CSV súborov. Pre sondu bol vytváraný jeden obrázok zo CSV súboru. Ale kôli zmene počtu pixelov na reláciu nie je počet vytváraných obrázkov konkretizovaný ale sú vytvárané do tej doby, kým je nenastane koniec súboru. Tento postup zabezpečí vytovrenie dvoch obrázkov, namiesto jedného a ak je hodnota počtu pixelov na reláciu zmenená, počet obrázkov je takisto zmený.
- **netServer.py** – základ module bol súbor netRB.py, ktorý bol premenovaný aby lepšie vystihoval jeho funkcionalitu. V module je definovaná trieda Classify, používaná na vytváranie predikcií. Pri inicializácií objektu je skontrolovaná dostupnosť zariadenia, podporujúceho nástroj `CUDA`. Ak dostupné nie je, pre výpočet bude použitý procesor zariadenia. Po nastavení výpočetného zariadenia je načítaný model frameworkom *pytorch* spolu s predtrénovanými váhami a je vložený do používaného zariadenia. Počas procesu inicializácie sú vytvárané logové záznamy spoločne s dobou, potrebnov na načítanie modelu. Trieda taktiež obsahuje jednu verejnú a jednu privátnu metódu. Verejná metóda prijíma ako vstupný parameter obrázky, ktoré sú vložené do používaného zariadenia. Nasleduje proces predicie, po ktorom je zavolaná privátna funkcia na analýzu výstupných dát z procesu. Dáta sú upravené funkciou softmax a je hodnota s najvyššou pravdepodobnosťou je vyňatá spolu s korešpondujúcou kategóriou. Karegória je namapovaná na jej názov podľa slovníka v súbore *config.py* a vytvoria sa logavé záznamy. Výstupom z metódy je pravdepodobnosť a názov kategórie.

[11] Hodnota 0.0.0.0 znamená, že webová aplikácia je dostupná všetkým staniciam, ktoré sú schopné sa spojiť s centrálov.

5 Testovanie návrhu v experimentálnom pracovisku

Táto kapitola je zameraná na spôsob testovania vytvorenej sieťovej sondy spoločne s centrálov. V prvej časti bude popísané experimentálne pracovisko s pohľadu dostupných staníc a sieťovej infraštruktúri. Do popísanej infraštruktúri bude zaimplementovaná sieťová sonda komunikujúca s centrálov. V ďalšej časti budú vybrané testovacie útoky a ich praktické prevedenie. Vytvorená sieťová komunikácia bude sieťovou sondov vyhodnotená a zaslaná na dodatočne overenie na centrálu. A v poslednej časti budú vyhodnotené získané výsledky testovania.

5.1 Popis experimentálneho pracoviska

Pre potreby testovanie sieťovej sondy bolo potrebné použiť experimentálne pracovisko, ktoré obsahuje rôznorodé počítačové systémy a možnosti spúšťať počítačové útoky na vybrané zariadenia. Na tento účel bolo poskytnuté virtuálne laboratórne prostredie, vytvorené virtualizačným softvérom *VMware ESXi*[1]. Výhoda pri využívaní virtuálneho prostredia bola vo vytváraní aktuálnych snímkov zariadení. To znamená zachytenie aktuálneho stavu stanice, do ktorého je možné stanicu navrátiť pri výskyte nepriaznivej situácie.

V tabuľke číslo 5.1 sa nachádza technická charakteristika všetkých staníc v experimentálnom pracovisku. Každej zo staníc je priradené doménové meno, statická IP adresa s maskov siete. Podľa účelu alebo služby, ktorú jednotlivé stanice poskytujú, im boli prideľované fyzicke zdroje. Tie sú pre každú stanicu odlišné ale Windows Domain Controller, Windows Server Exchange, Apache Web Server a Ubuntu Elasticsearch majú k dispozícia najviac fyzických zdrojov. Je to preto, že sú vyťažované ostatnými stanicami alebo budú vyťažované v budúcnosti.

V tabuľke číslo 5.2 je zobrazená sieťová infraštruktúra experimentálneho pracoviska. Sieť je rozdelená na tri VLAN (*virtual local area network*) zóny a to DMZ (demilitarized zone), private a public. Každá z nich má priradený vlastný virtuálny prepínač, starajúci sa o smerovanie komunikácie. V každej zóne je umiestnená klientská stanica s operačným systémom *kali linux*[2]. Tieto stanice slúžia na simuláciu kybernetických útokov v situácií sa sa útočník nachádza vo verejnej siete, DMZ zóne alebo získal prístup do privátnej časti siete. Umiestnenie ostatných klientských

[1] Bližšie informácie sú k dispozícií na: `https://www.vmware.com/products/esxi-and-esx.html`.

[2] Bližšie informácie sú k dispozícií na: `https://www.kali.org/docs/introduction/what-is-kali-linux/`.

staníc a serverov má štandartnú formu, čo značí, že webový server je umiestnený v DMZ zóne a ostatné klientské stanice spolu s serveri spolu s Windows radičom domény sa nachádzajú v privátnej časti.

Tab. 5.1: Technická charakteristika experimentálneho pracovika.

Názov	Doménové meno	Sieťová adresa	Maska siete	VLAN	HDD [GB]	RAM [GB]	CPU
Windows Domain Controller	regina.vmware.fekt.cz	10.50.64.2	255.255.192.0	Private	50	12	2
Windows Server Exchange	eva.vmware.fekt.cz	10.50.64.3	255.255.192.0	Private	50	10	2
Apache Web Server	mia.vmware.fekt.cz	10.50.0.2	255.255.192.0	DMZ	30	10	2
Kali Linux	katka.vmware.fekt.cz	10.50.128.2	255.255.192.0	Public	20	4	1
Kali Linux	terka.vmware.fekt.cz	10.50.65.4	255.255.192.0	Private	20	4	1
Kali Linux	klara.vmware.fekt.cz	10.50.1.2	255.255.192.0	DMZ	20	4	1
Windows Endpoint	renata.vmware.fekt.cz	10.50.65.2	255.255.192.0	Private	30	4	1
Windows Endpoint	radka.vmware.fekt.cz	10.50.65.3	255.255.192.0	Private	30	6	2
Ubuntu Elasticsearch	lada.vmware.fekt.cz	10.50.64.4	255.255.192.0	Private	50	10	2
CentOS Linux	miri.vmware.fekt.cz	10.50.64.5	255.255.192.0	Private	20	4	2
CentOS Linux	lenka.vmware.fekt.cz	10.50.64.6	255.255.192.0	Private	10	2	1

Tab. 5.2: Sieťová infraštruktúra experimentálneho pracovika.

VLAN zone	VLAN-ID	Subnet
DMZ	10	10.50.0.0/18
Private	20	10.50.64.0/18
Public	30	10.50.128.0/18

DMZ	**Addresses**
Critical Hosts	10.50.0.2 - 10.50.0.254
Endpoints	10.50.1.2 - 10.50.1.254
Other	10.50.2.2 - 10.50.63.254

Private	**Addresses**
Critical Hosts	10.50.64.2 - 10.50.64.254
Endpoints	10.50.65.2 - 10.50.65.254
Other	10.50.66.2 - 10.50.127.254

Public	**Addresses**
Other	10.50.128.2 - 10.50.255.254

Sonda bola pripojená do fyzického rozhrania v serveri, na ktorom sa nachádza virtualizované experimentálne pracovisko. Na to aby bola sonda funkčná, musí byť sieťová komunikácia zrkadlená a posielaná na sieťové rozhrania, do ktorého je pripojená. Virtualizačným softvérom *VMware ESXi* nepodporuje zrkadlenie komunikácie

z vnútornej siete do vonkajších zariadení. Preto bol virtuálny prepínač zóny private nastavený do promiskuitného módu. To spôsobilo zasielanie každého prichádzajúceho paketu všetkým staniciam v zóne. Následne bola vytvorená užívateľská stanica s doménovým menom lenka majúca dve sieťové rozhrania. Prvé bolo pripojené do zóny private a druhý bol pripojený do nového virtuálneho prepínača, do ktorého je taktiež pripojené aj fyzické rozhranie so sondou. Tá však neprijíma potrebnú sieťovú komunikáciu. Preto na užívateľskej stanici lenka bolo prvé sieťové rozhranie nastavené do promiskuitného módu, čím zaniklo zahadzovanie paketov určenýhc iným staniciam. Následne aby boli pakety preposlané sonde, musia byť zapuzdrené do novej IP hlavičky. Preto bol medzi sondou a lenkou vytvorený sieťový tunel, do ktorého boli pakety smerované. Ako posledný krok bolo overenie funkčnosti realizovaného riešenia zapnutím programu tcpdump a preskúmať do sondy prichádzajúce pakety. Ich IP adresy pochádzali z privátnej VLAN zóny, čím bolo docielené zrkadnenie paketov do vonkajších zariadení.

Centrála nie je súčasťov virtualizovaného experimentálneho pracoviska. Pre jej účely bolo nutné nastaviť dedikovaný server, do ktorého bola pripojená grafická karta nvidia gtx 1050 s podporov `CUDA`. Tým bol zabezpečený akcelerovaný výpočet predikcií neurónovou sieťou. A pre sprístupnenie centráli sonde, je server prístupný z verejnej siete.

5.2 Zvolené testovacie útoky

Pre potreby testovania boli zo zoznamu kybernetických útokov, použitých na vytvorenie dátovej sady `CSE-CIC-IDS2018`, vybrané tri. Tie budú spustené v experimentálnom pracovisku zo stanice s doménovým menom terka. Nimi vytvorená sieťová komunikácia bude spracovaná a vyhodnotená sondou. Tá následne spracované CSV súbory pošle centrále na dodatočné vyhodnotenie.

Ako prvý bol zvolený útok skenovaním portov (*port scan attack*[3]). Ten je štandartne využívaný útočníkmi pri vstupe do neznámej siete na získanie informácií o dostupných zariadení a slúžbách, ktoré sú stanicami poskytované. Preto je žiadúce zachytiť túto nežiadanú aktivitu v komunikačnej sieti. Nástroj, použitý na skenovanie portov je nazývaný *nmap*[4]. Ten bol spúšťaný na štyri fázy. V prvej fáze bola overovaná dostupnosť prvých 1024 portov. V druhej fáze boli zisťované aj verzie služieb. V tretej faze boli získavané všetky informácie, ktoré sa dali zo skenovanej stanice získať ale stále bol rozsah portov obmedzený na 1024. V poslednej fáze bol

[3]Bližšie informácie sú k dispozícií na: `https://www.extrahop.com/resources/attacks/malicious-port-scanning/`.

[4]Bližšie informácie sú k dispozícií na: `https://nmap.org/docs.html`.

skenovaných celý rozsah, čiže všetkých 65 535 portov bez dodatočných modifikátorov. Pre definíciu staníc, ktoré majú byť skenované bol vytvorený textový súbor. Ten obsahoval IP adresy všetkych staníc v tabuľke číslo 5.1 okrem stanice, z ktorej bol útok spúšťaný. Dôvod, prečo nebol použitý celý rozsah podsiete je ten, že pred kažkým dotazovaním sa stanice je použitý ARP protokol na získanie MAC adresy. Ak stanica dostupná nie je, odpoveď nedorazí a postup je opakovaný pre ďalšiu IP adresu. Táto komunikácia bude nástrojom `cicflowmeter` preskočená, z dôvodu absencie TCP alebo UDP hlavičky.

Druhý útok bol zvolený slovníkový útok na protokol *SSH*. Na jeho realizáciu bol použitý nástroj *hydra*[5] a slovník s názvom *rockyou.txt*. Po spustení budú vytvárané spojenia s cieľovou stanicou a postupným výberom hesiel zo slovníka budú overované. To bude pokračovať až do doby, kým nie je uskutočnené úspešné prihlásenie. Ako cieľ útoku bola zvolená stanica s doménovým meno lenka a doba, po ktorú bol útok spustený bola stanovená na 30 minút.

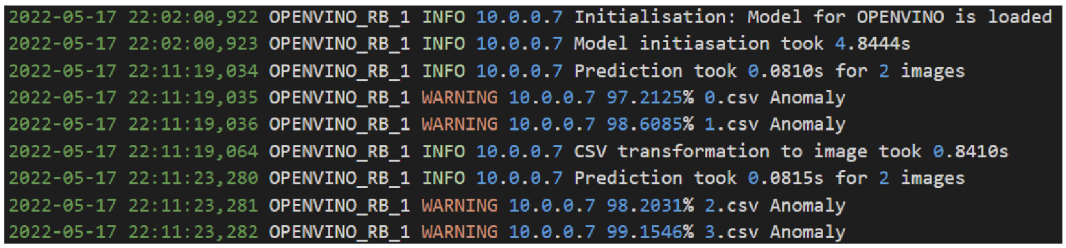
Tretím útokom bol DOS (*denial of service*) na webový server s názvom *slowLoris*[6]. Útok spočiva vo vytváraní a udržiavaní určeného počtu spojení s cieľovým webovým serverom až do doby, kým nie je schopný vytvoriť nové. Tým je zabezpečené odopretie prístupu ostatným užívateľom, čim je služba zneprístupnená. Ako cieľ útoku bol zvolený webový server, dostupný na stanici s doménovým menom mia. Pred štartom útoku bolo stanový maximálny počet spojení na 1000 a doba útoku na 30 minút.

5.3 Hodnotenie výsledkov testovania návrhu

V tabuľke číslo 5.3 sú uvedené útoky s percentuálnou úspešnosťov predikcií sondy a centrály. Úspešnosť predikcií bola vypočítaná z logových záznamov, vytváraných počas priebehu testovania. Na obrázku číslo 5.1 je zobrazená ukážka záznamu z log súboru vytvoreného sondou. Na začiatku je záznam z inicializácie neurónovej siete, nasledovaný záznamom uskutočnených predikcií. Taktiež boli zaznamenávané doby trvania transformácie CSV súborov na obrázky a doby trvania uskutočnovanej predikcie. CSV súbory, ktoré sa v zázname spracované, obsahujú zachytený skenovanie portov.

Zo zvolených útokov bol sondov zachytený iba prvý útok, ktorý bol úspešne vyhodnotený za anomáliu so 100 % úspešnosťou. Ostatné útoky boli klasifikované za normálnu komunikáciu, čo mohlo byť spôsobené nedostatkom komunikácie bežných užívateľov.

[5]Bližšie informácie sú k dispozícií na: `https://github.com/vanhauser-thc/thc-hydra`.
[6]Bližšie informácie sú k dispozícií na: `https://github.com/gkbrk/slowloris`.



Obr. 5.1: Ukážka logového záznamu vytvoréneho sondov.

Zo sondy boli CSV súbory zaslané na centrálu pre dodatočnú kontrolu. Tá pri skenovaní portov dokázala klasifikovať 50 % CSV záznamov ako infiltrácia alebo DOS útok. Lepší výsledok bol dosiahnutý pri útoku *slowLoris*, ktorý bol klasifikovaný ako DOS útok s pravdepodobnosťov 80 %. Posledný slovníkový SSH útok bol klasifikovaný ako normálna komunikácia, rovnako ako sondou. Jednov z príčin takého to výsledku môže byť nedostatočná sieťová komunikácia vytváraná užívateľmi. Taktiež mnoho extrahovaných vlastností je založených na časovej zložke prenášaných paketov, ktoré sú ovplinované agresívnosťov útoku. Zmena v rýchlosti vysielania žiadosti oproti použitej dátovej sade mohla spôsobyť podobnosť s normálnou komunikáciou. Možné korekcie tohoto problému sú uvedené v kapitole číslo 6.

Tab. 5.3: Tabuľka s výsledkami tesovacích útokov

Názov útoku	**Úspešnosť sondy** [%]	**Úspešnosť centrály** [%]	**Cieľová stanica**	**Trvanie útoku**
Skenovanie portov (*port scanning*)	100	50	Všetkých 10 staníc	8
Slovníkový SSH útok (*dictionary SSH attack*)	0	0	lenka	30
Slowloris	0	80	mia	30

Počas testovania bolo pozorované aj hardvérové vyťaženie sondy. Maximálne vyťaženie procesora dosiahlo hodnoty 50 % a maximálne využitie pamäte RAM bolo 8 %. Z výsledkov je zrejmé, že generovaná sieťová komunikácia v experimentálnom pracovisku by mohla byť zvýšená.

6 Rozširujúce návrhy

V tejto kapitole budú navrhnuté rozšírenia práce a dôvody, prečo prečo ich uskutočniť. V prvom rade bude zameraná pozornosť na dátovu sadu, použitú na trénovanie modelov. A to z možnosti rozšírenia jej veľkosti o dáta, zachytené pri prevádzke sondy a expanziu množstva extrahovaných vlastností sieťovej komunikácie. Následne budú predstavené možnosti hlásenia zachytených anomálií užívateľom a spôsobu vykonávanie korekčných úkonov pri nesprávnom odhade. Posledný rozširujúci návrh, bude zameraný na zjednodušenie nasadenia sondy do prevádzy a jej dodatočnej správy.

Možnosti úpravy dátovej sady

Jedným z dôvodov, prečo nebol slovníkový SSH útok zachytený može byť nerovnomerné zloženie dátovej sady. Tento trend bolo možné pozorovať aj na maticiach zámien, v ktorej mala normálna komunikácia oveľa väčšie zastúpenie ako všetky anomálie dohromady. Tento problém môže byť adresovaný doplnením dátovej sady o dátový prenos tak, aby pomer medzi normálnou komunikáciou a anomáliami sa približoval jedna ku jednej. To môže byť dosiahnuté dvomi spôsobmi. Prvým je generovanie útokov v experimentálnom pracovisku zároveň z obyčajnou komunikáciou, vytváranov bežným správaním sa užívateľov počítačovej siete. Zo zachytených dát extrahovať vlastnosti a ku každej relácií priradiť značku. Druhým prístupom je zachytávanie komunikácie pomocou sondy, tá automaticky extrahuje vlastnosti a vytvorý rovnako veľké bloky dát. Ku kažkému bloku bode priradená ina jedna značka prenosu, podľa času vytvorenia.

Druhý spôsob úpravy dátovej sady je zvýšenie počtu extrahovaných vlastností. Tým sa navýší množstvo informácií, ktoré má neurónová sieť k dispozícií ale zvýši sa tým aj harvérové vyťaženie sondy. Preto tento prístup vyžaduje dodatočné testovanie.

Zavedenie hlásenia anomálií a korekčných pravidiel

Posledným úkonom spracovanie dát je vytvorenie logových záznamov. Tento prístup nie je vhodný pre praktické nasadenie sondy a centráli a preto je potrebné doplniť dodatočné kroky na spracovanie vytvorených predikcií. Prvý krok je použitie programu *filebeat*[1]. Ten slúži na zhromažďovanie logových súborov zo staníc na

[1]Bližšie informácie sú k dispozícií na:`https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-overview.html`.

jedno miesto pre ich analýzu a zálohu. Takto je zaistená správa logov bez nutnosti manuálnej kontorly každej stanice. Druhým krokom je zasielanie oznámení o zistenej anomálnej činnosti v sieti. To môže byť zabezpečené viacerými spôsobmi. Najjednoduchšie riešenie je zasielanie e-mailovej správy s informáciami o zistení. Alebo je možné využiť pokročilejšie riešenia tretích strán ako sú ako *slack* a *pagerduty*. Tie poskytujú vytvorenie skupiny, do ktorej sú zasielané informácií o zisteniach a umožňujú rýchlu komunikáciu medzi jej členmi.

Niektoré vytvárané predikcie nemusia byť správne, preto zavedením možnosti trénovania neurónovej siete za behu programu je možné docieliť jeho postupné ladenie. Trénovanie by však bolo spúštané manuálnym výberom zle predikovaných súborov alebo označením časového úseku, počas ktorého k chybe nastalo. Avšak s týmto prístupom je možné presnosť predikcií znížiť, preto je nutné vytvárať periodické zálohy.

Zjednodušenie správy sondy

Nasadenie sondy do prevádzky a následné uskutočňovanie aktualizácie systému je časovo náročna záležitosť. Preto je navrhované využitie automatizačného programu *ansible*[2], ktorý je používaný na zjednodušenie správy systémov. Jeho základom sú tzv. playbook-y, obsahujúce postupnosť úkonov, ktoré budú vykonávané. Na cieľovej stanici musí byť dostupná služba SSH, cez ktorú sa *ansible* pripojí a vykoná vopred definované úkony. Týmto spôsobom môžu byť nastavované paralélne viaceré zariadenia súčasne. To platí aj pre vykonávanie aktualizácií.

[2]Bližšie informácie sú k dispozícií na:`https://www.ansible.com/resources/get-started?hsLang=en-us`.

Záver

Táto práca bola zameraná na aplikovanie konvolučnej neurónovej siete na detekciu anomálií v počítačových sieťach pomocou grafickej reprezentácie prevádzky. V teoretickej časti boli vysvetlené základné techniky zachytávania dátového prenosu za účelom analýzy. Následne boli popísané umelé neurónové siete a to z pohľadu ich stavby a funkčnosti. Vysvetlené boli aj metriky, používané na hodnotenie presnosti učených modelov a ako posledné boli popísané tri generalizované hlboké neurónové siete. Hlavná váha bola kladená na ich zloženie vrstiev a výhod, ktoré sú daným prístupom dosiahnuté. Po teoretickej časti nasledovala rešerš moderných vedeckých prístupov, zameraných na detekciu anomálií hlbokými neurónovými sieťami. Vybrané boli štyti práce, z ktorých bola vytvorená stručná sumarizácia prístupu k problematike a dosiahnutých výsledkov. Vyzdvyhnuté boli taktiež aj použité dátové sady, spôsob ich tranformácia na obrázky a použité modeli neurónovných sietí.

Praktická časť bola rozdelená do časti prípravy komponentov, časti programovej implementácie a časti testovania návrhu. V časti prípravy komponentov boli popísané prístupy, aplikované na zvolenú dátovú sadu `CSE-CIC-IDS2018`, z ktorej boli vytvorené tri sady obrázkov. Každá zo sád obsahuje rozdielný počet extrahovaných vlastností z relácie, použitých na tvorbu obrázkov, čo zapríčiňilo ich odlišné počty prvkov. Tieto sady boli použíté na trénovanie modelov `MobileNetv2`, `MobileNetv3` a `Denset`. Celkovo bolo natrénovaných päť modelov, z ktorých najlepšej presnosti 97,9 % dosiahol `MobileNetv3` s použitou dátovou sadou, vytvorenou tretím prístupom. Tento prístup zahŕňal najväčší počet extrahovaných vlastností z jednej relácie. Preto tento model bude použitý v centrále a v sonde bude použitý `MobileNetv2`. Pri tomto modeli najlepšiej presnosti 96,7 % bolo dosiahnuté použitím dátovej sady, vytvorenou druhým prístupom. Posledným komponentom bol nástroj `CicFlowMeter`, ktorý bol upravený aby vytvárané záznamy sieťovej komunikácie do súborov o konštantnej veľkosti a vytváral žiadosti na ich preverenie na výskyt anomálií.

V úvode časti programovej implementácie bolo vytvorené možné zapojenie detekčnej architektúri a stanice, z ktorej je zložená. To bolo nasledované detailným popisom sondy a centrály. Ako prvá bola predstavená sonda a jej hardvérové a softvérové požiadavky. Následne bola predstavená funkcionalita a režimy predikcie na vopred pripravených obrázkoch alebo dát zo zachytenej komunikácie. Popis sondy bol završený predstavením programovej štruktúri a podporných nástrojov na kontrolu dotupného mieta na disku a zálohovanie vytvorených záznamov. Popis centráli bol mal rovnaký charakter ako u sondy.

V poslednej časti bolo popísané testovanie vytvoreného návrhu v experimentálnom pracovisku. Pracovisko pozostávalo z virtualizačného nástroja *VMware ESXi*, v ktorom sa nachádzalo jedenásť strojov. K týmto strojom bolo cez fyzické rozhranie

pripojená sonda a privátna komunikácia tvorená vo experimentálnom pracovisku bola zrkadlená tak, aby ju sonda zachytávala a spracovávala. Na testovacie účely bol zvolý útok skenovania portov, ktorý bol zachytený sondou aj centrálou, DOS útok slowLoris zachytený iba centrálou a slovníkový SSH útok, ktorý nebol zachytený ani jedným zariadením. Po vyhodnotení výsledkov testovania boli navrhnuté rozšírenia práce, zamerané na expanziu dátovej sady, hlásenie vyskytnutých udalostí a automatizácie správy zariadení.

Literatúra

[1] QIAN, M.;CoONG, S.;BAOJIANG, C.;XIAOHUI, J.: *A novel model for anomaly detection in network traffic based on kernel support vector machine* [online]. Computer & Science Volume 104, 5.2021, [cit. 12.12.2021]. Dostupné z URL:<https://www.sciencedirect.com/science/article/pii/S0167404821000390>.

[2] SIMROSS-WATTENBERG, F.;ASENSIO-PE'REZ, I., J.;CASASECA-DE-LA-HIGUERA, P.; MARTI'N-FERNA'NDEZ, M.;DIMITRIADIS, A., I.: *Anomaly Detection in Network Traffic Based on Statistical Inference and Stable Modeling* [online]. IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 8, 7/8.2021, [cit. 12.12.2021]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5714699>.

[3] GRIMMICK, R.: *Packet Capture: What is it and What You Need to Know* [online]. Varonis, posledná aktualizácia 4.8.2021, [cit. 12.12.2021]. Dostupné z URL: <https://www.varonis.com/blog/packet-capture/>.

[4] BEALE, J.; CASWELL, B.: *Snort Intrusion Detection 2.0*. 2003, [cit. 12.12.2021].

[5] WINPCAP: *WinPcap Documentation* [online]. [cit. 12.12.2021]. Dostupné z URL: <https://www.winpcap.org/docs/docs_412/html/main.html>.

[6] NMAP.ORG: *Npcap: Nmap Project's packet sniffing library for Windows* [online]. [cit. 12.12.2021]. Dostupné z URL: <https://nmap.org/npcap/guide/>.

[7] COMPTIA: *What Is Wireshark and How Is It Used?* [online]. [cit. 12.12.2021]. Dostupné z URL: <https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it>.

[8] TCPDUMP: *Man page of TCPDUMP* [online]. posledná aktualizácia 19.11.2021, [cit. 12.12.2021]. Dostupné z URL: <https://www.tcpdump.org/manpages/tcpdump.1.html>.

[9] Canadian Institute for Cybersecurity: *CICFlowMeter (formerly ISCXFlowMeter)* [online]. [cit. 12.12.2021]. Dostupné z URL: <https://www.unb.ca/cic/research/applications.html>.

[10] IBM Cloud Education: *Neural Networks* [online]. posledná aktualizácia 17.8.2020, [cit. 12. 12. 2021]. Dostupné z URL: <https://www.ibm.com/cloud/learn/neural-networks>.

[11] GUPTA, V.: *Understanding Feedforward Neural Networks* [online]. posledná aktualizácia 9.8.2017, [cit. 12.12.2021]. Dostupné z URL: <https://learnopencv.com/understanding-feedforward-neural-networks/>.

[12] MCCULLUM, N.: *Deep Learning Neural Networks Explained in Plain English* [online]. freeCodeCamp, posledná aktualizácia 28.7.2020, [cit. 12.12.2021]. Dostupné z URL: <https://www.freecodecamp.org/news/deep-learning-neural-networks-explained-in-plain-english/>.

[13] BAHETI, P.: *The Essential Guide to Neural Network Architectures* [online]. Microsoft, posledná aktualizácia 29.11.2021, [cit. 12.12.2021]. Dostupné z URL: <https://www.v7labs.com/blog/neural-network-architectures-guide#what-are-neural-networks>.

[14] PAI, A.: *CNN vs. RNN vs. ANN – Analyzing 3 Types of Neural Networks in Deep Learning* [online]. posledná aktualizácia 17.2.2020, [cit. 12.12.2021]. Dostupné z URL: <>.

[15] SAZLI, H., M.: *A brief review of feed-forward neural networks* [online]. Ankara University, Faculty of Engineering, Department of Electronics Engineering, Ankara, TURKEY, posledná aktualizácia 6.2.2006, [cit. 12.12.2021]. Dostupné z URL: <https://www.researchgate.net/publication/228394623_A_brief_review_of_feed-forward_neural_networks>.

[16] IBM Cloud Education: *Convolutional Neural Networks* [online]. posledná aktualizácia 20.8.2020, [cit. 12.12.2021]. Dostupné z URL: <https://www.ibm.com/cloud/learn/convolutional-neural-networks>.

[17] IBM Cloud Education: *Recurrent Neural Networks* [online]. posledná aktualizácia 14.9.2020, [cit. 12.12.2021]. Dostupné z URL: <https://www.ibm.com/cloud/learn/recurrent-neural-networks>.

[18] SAEED, M.: *An Introduction To Recurrent Neural Networks And The Math That Powers Them* [online]. posledná aktualizácia 24.9.2021, [cit. 12.12.2021]. Dostupné z URL: <>.

[19] NICHOLSON, C.: *Evaluation Metrics for Machine Learning – Accuracy, Precision, Recall, and F1 Defined* [online]. [cit. 12.12.2021]. Dostupné z URL: <https://wiki.pathmind.com/accuracy-precision-recall-f1>.

[20] SANDLER, M.;HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.: *MobileNetV2: Inverted Residuals and Linear Bottlenecks* [online]. posledná aktualizácia 21.3.2019, [cit. 22.05.2021]. Dostupné z URL: <https://arxiv.org/pdf/1801.04381.pdf>.

[21] Google AI, Google Brain: *Searching for MobileNetV3* [online]. posledná aktualizácia 20.11.2019, [cit. 22.05.2021]. Dostupné z URL: <https://arxiv.org/pdf/1905.02244.pdf>.

[22] HUANG, A.; LIU, Z.; MAATEN, L.; WEINBERGER, K.Q.: *Densely Connected Convolutional Networks* [online]. posledná aktualizácia 28.01.2018, [cit. 22.05.2021]. Dostupné z URL: <https://arxiv.org/pdf/1608.06993.pdf>.

[23] PERE, C.: *What are Loss Functions?* [online]. posledná aktualizácia 17.6.2020, [cit. 12.12.2021]. Dostupné z URL: <https://towardsdatascience.com/what-is-loss-function-1e2605aeb904>.

[24] BROWNLEE, J.: *What is a Confusion Matrix in Machine Learning* [online]. posledná aktualizácia 18.11.2016, [cit. 12.12.2021]. Dostupné z URL: <https://machinelearningmastery.com/confusion-matrix-machine-learning/>.

[25] LAM, J.: *Machine Learning based Anomaly Detection for 5G Networks* [online]. Macquarie University, Robert Abbas, Macquarie University, 7.3.2020, [cit. 12.12.2021]. Dostupné z URL: <https://arxiv.org/pdf/2003.03474.pdf>.

[26] TAHERI, S.;SALEM, M.; YUAN, S.,J.: *Leveraging Image Representation of Network Traffic Data and Transfer Learning in Botnet Detection* [online]. Department of Electrical and Computer Engineering, University of Central Florida, Orlando, USA, 27.1.2018, [cit. 12.12.2021]. Dostupné z URL: <https://pdfs.semanticscholar.org/aa23/7db17ca455e2020f0152ae10257448ab7ed4.pdf>.

[27] WANG, W.;ZHU, M.: *Malware Traffic Classification Using Convolutional Neural Network for Representation Learning* [online]. Department of Automation, University of Science and Technology of China Hefei, China, 2017, [cit. 12.12.2021]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7899588&tag=1>.

[28] LIU, Z.;DI, X.;DING, Q.; LIU, W.; QI. H.; LI, J.; YANG, H.: *NADS-RA: Network Anomaly Detection Scheme Based on Feature Representation and Data Augmentation* [online]. School of Computer Science and Technology, Changchun University of Science and Technology, Changchun, China, 25.11.2020, [cit. 12.12.2021]. Dostupné z URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9269968>.

[29] LOYSE, G.: *raspberry-pi Documentation* [online]. posledná aktualizácia 16.11.2017, [cit. 22.05.2022]. Dostupné z URL: <https://readthedocs.org/projects/raspberry-pi-intro/downloads/pdf/latest/>.

[30] INTEL: *Intel® Neural Compute Stick 2* [online]. [cit. 22.05.2022]. Dostupné z URL: <>.

[31] INTEL: *Intel® Distribution of OpenVINO Toolkit* [online]. [cit. 22.05.2022]. Dostupné z URL: <https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html>.

[32] NVIDIA: *CUDA C++ Programming Guide* [online]. posledná aktualizácia 05.2022, [cit. 22.05.2022]. Dostupné z URL: <https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf>.

[33] RAI, Jatin. *ArcSight – A better insight security solution.* [online]. [cit. 20. 3. 2022]. Dostupné z URL: <https://www.trustradius.com/reviews/arcsight-enterprise-security-manager-formerly-hp-arcsight-2019-12-09-13-03-18>.

Zoznam symbolov a skratiek

API	Application Programming Interface
CSV	Comma-separated values
DDoS	Distributed Denial of Service
DoS	Denial of Service
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IMAP	Internet Message Access Protocol
NCS	Neural Compute Stick
PCAP	Packet Capture
POP3	Post Office Protocol
ReLU	Rectified Linear Unit
RGB	Red Green Blue
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell

Zoznam príloh

A Obsah elektronickej prílohy

V elektronickej prílohe sú priložené súbory so zdrojovými kódmi aplikacií, použitých pri tvorbe tejto práce. Súčasťou sú aj upravené súbory nástroja cicflowmeter a obrázky na testovanie. S prílohe nie sú dodané natrénované modeli.

```
.1 / ..... koreňový adresár priloženého archívu
├── centrala ..... adresár s programom centrály
│   ├── model ..... adresár na vkladanie modelov
│   ├── app.py ..... spustiteľná python aplikácia
│   ├── config.py
│   ├── filePreprocessing.py
│   ├── imageTransformation.py
│   ├── netServer.py
│   ├── README.md ..... postup inštalácie a spustenia
│   ├── requirements.txt
│   ├── uploadScript.py
│   └── utilities.py
├── cicflowmeter ..... adresár s upravenými časťami programu CicFlowMeter
│   └── src
│       └── cicflowmeter
│           ├── flow.py
│           └── flow_session.py
├── obrazky ..... adresár s testovacími obrázkami
│   ├── 2_pristup ..... obrázky vytvorené 2. prístupom
│   │   └── <cislo>.png ..... 10 obrázkov
│   └── 3_pristup ..... obrázky vytvorené 3. prístupom
│       └── <cislo>.png ..... 10 obrázkov
└── sonda ..... adresár s programom sondy
    ├── model ..... adresár na vkladanie modelov
    ├── app.py ..... spustiteľná python aplikácia
    ├── config.py
    ├── filePreprocessing.py
    ├── imageTransformation.py
    ├── netRB.py
    ├── README.md ..... postup inštalácie a spustenia
    ├── requirements.txt
    ├── spaceRelease.sh ..... skript na kontrolu miesta na disku
    ├── uploadScript.py
    └── utilities.py
```