VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCIA MALÍGNYCH DOMÉN S VYUŽITÍM AI

DETECTION OF MALICIOUS DOMAINS USING AI

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE **Bc. FILIP BUČKO**
AUTHOR

VEDOUCÍ PRÁCE **Ing. RADEK HRANICKÝ, Ph.D.**
SUPERVISOR

BRNO 2025

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání diplomové práce

164612

Ústav:	Ústav informačních systémů (UIFS)
Student:	**Bučko Filip, Bc.**
Program:	Informační technologie a umělá inteligence
Specializace:	Kybernetická bezpečnost
Název:	**Detekce maligních domén s využitím AI**
Kategorie:	Bezpečnost
Akademický rok:	2024/25

Zadání:

1. Proveďte rešerši existujícího výzkumu v oblasti detekce maligních doménových jmen (souvisejících s malware, phishingovými kampaněmi apod.).
2. Seznamte se s oblastí umělé inteligence s využitím modelů na bázi deep learning, generativní AI, LLM apod.
3. Navrhněte řešení (model či sadu modelů) pro detekci maligních doménových jmen pomocí umělé inteligence.
4. Navržené řešení implementujte.
5. Experimentálně ověřte funkčnost vytvořeného řešení a porovnejte jej s existujícími přístupy.
6. Zhodnoťte dosažené výsledky a diskutujte možná rozšíření.

Literatura:

- Vaswani, A., Attention is all you need. *Advances in Neural Information Processing Systems, 2017.*
- Hranický, R., Horák, A., Polišenský, J., Jeřábek, K. and Ryšavý, O.: Unmasking the phishermen: phishing domain detection with machine learning and multi-source intelligence. In: Proceedings of the 20th IEEE/IFIP Network Operations and Management Symposium (NOMS), pp. 1-5, IEEE, 2024.
- Foster, David. *Generative deep learning*. " O'Reilly Media, Inc.", 2022.

Při obhajobě semestrální části projektu je požadováno:
Body 1 až 3.

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

Vedoucí práce:	**Hranický Radek, Ing., Ph.D.**
Vedoucí ústavu:	Kolář Dušan, doc. Dr. Ing.
Datum zadání:	1.11.2024
Termín pro odevzdání:	21.5.2025
Datum schválení:	22.10.2024

Abstrakt

Malígne domény predstavujú vážnu hrozbu pre kybernetickú bezpečnosť, keďže sa zneužívajú na šírenie škodlivého kódu, uskutočňovanie phishingových útokov a komunikáciu s napadnutými systémami prostredníctvom domén generovaných algoritmami (DGA). Cieľom tejto práce je navrhnúť a vyhodnotiť modely pre ich detekciu s využitím neurónových sietí postavených na transformerovej architektúre. Modely boli trénované na rôznych typoch vstupných dát, ako sú názvy domén, záznamy RDAP/Whois, údaje z DNS komunikácie a geografické informácie z IP adries. Najvyššie F1 skóre (98,61%) dosiahol model určený na detekciu DGA domén natrénovaný nad doménovými menami. Podobne vysoké hodnoty sa podarilo dosiahnuť aj pri detekcii phishingu (98,39%) a malvéru (95,70%) na základe údajov z RDAP. Navrhované riešenie vykazuje vysokú presnosť, eliminuje potrebu časovo náročného manuálneho spracovania vstupných údajov a umožňuje automatizované učenie z nových dát. Tým sa zvyšuje odolnosť voči meniacim sa útočným technikám a zaručuje spoľahlivé fungovanie aj pri dlhodobom nasadení v praxi.

Abstract

Malicious domains pose a serious threat to cybersecurity, as they are exploited for the distribution of malicious code, execution of phishing attacks, and communication with compromised systems via algorithmically generated domains (DGA). The aim of this thesis is to design and evaluate models for their detection using neural networks based on the transformer architecture. The models were trained on various types of input data, including domain names, RDAP/Whois records, DNS communication data, and geographic information derived from IP addresses. The highest F1 score (98.61%) was achieved by the model designed to detect DGA domains, trained on domain names. Similarly high values were achieved in detecting phishing (98.39%) and malware (95.70%) based on RDAP data. The proposed solution demonstrates high accuracy, eliminates the need for time-consuming manual preprocessing of input data, and enables automated learning from new data. This increases its resilience to evolving attack techniques and ensures reliable performance even during long-term deployment.

Kľúčové slová

Škodlivé domény, Algoritmy generovania domén (DGA), Detekcia malvéru, Detekcia phishingu, Transformery, Generatívna AI, Transferové učenie, Sekvenčné modelovanie, Mechanizmy pozornosti.

Keywords

Malicious domains, Domain Generation Algorithms (DGAs), Malware detection, Phishing detection, Transformer Neural Network, Generative AI, Transfer learning, Sequence modeling, Attention mechanisms.

Citácia

BUČKO, Filip. *Detekcia malígnych domén s využitím AI.* Brno, 2025. Semestrální projekt. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Hranický, Ph.D.

Detekcia malígnych domén s využitím AI

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením Ing. Radka Hranického, Ph.D. a uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Filip Bučko
21. mája 2025

Poďakovanie

Ak by sa ma niekto spýtal, aká je najvýznamnejšia časť diplomovej práce, odpoveďou by nebol Abstrakt, Úvod ani Záver, ale poďakovanie všetkým, ktorí mi pomohli dotiahnuť to až k tomuto bodu. Preto som vďačný všetkým, ktorí ma podporovali počas celého môjho štúdia a počas tvorby tejto diplomovej práce.

V prvom rade patrí veľká vďaka môjmu vedúcemu Radkovi, ktorý sa stal dôležitou súčasťou môjho pôsobenia na fakulte. Jeho trpezlivosť, ochota pomáhať a schopnosť smerovať správnym smerom boli pre mňa veľkou oporou. Nielen pri písaní práce, ale aj pri odbornom a osobnom raste.

Rovnako si vážim podporu svojich kolegov zo skupiny FETA, ktorí mi vždy ochotne pomáhali pri riešení nejasností a technických otázok. Osobitne ďakujem Ondrovi Ondryášovi, Kamilovi Jeřábkovi a Honzovi Políšenskému.

Osobitné poďakovanie patrí mojim rodičom a mojej sestre za ich podporu, dôveru a trpezlivosť. Ďakujem im, že o mne nikdy nepochybovali a verili mi v každej chvíli.

V neposlednom rade by som chcel poďakovať svojim najbližším kamarátom, na ktorých sa môžem vždy spoľahnúť a ktorí ma počas celej cesty neustále podporovali, aby som túto prácu dotiahol do úspešného konca. Ďakujem Maťovi, Terezke, Matúškovi, Šimimu, Kláre, Eri, Jurimu, Kubovi. Som vďačný za to, že mám tých najlepších kamarátov, akých som si mohol kedy priať.

Táto práca je riešená v rámci projektov „Analýza šifrovaného provozu pomocí síťových toků“, č. projektu VJ02010024 podporovaným Ministerstvom vnútra Českej republiky a „Inteligentné informační technologie pro odolnou společnost“ č. projektu FIT-S-23-8209 podporovaný univerzitou VUT v Brne. Poďakovanie tiež adresujem aj sponzorom tejto práce.

Obsah

Zoznam obrázkov

Kapitola 1

Úvod

Internet sa stal neoddeliteľnou súčasťou každodenného života, umožňujúc bezproblémovú komunikáciu, elektronické obchodovanie a neobmedzený prístup k informáciám. Napriek týmto výhodám vytvára sieťová konektivita priaznivé prostredie pre širokú škálu škodlivých aktivít. Medzi tieto aktivity patria aj malígne domény, ktoré sú vytvárané na uskutočňovanie phishingových útokov, šírenie malvéru alebo iných kybernetických hrozieb. Domény, vytvorené s cieľom poškodiť alebo získať citlivé údaje, predstavujú významné riziko v rámci kybernetického priestoru.

Účinná detekcia malígnych domén je nevyhnutná pre zaistenie kybernetickej bezpečnosti. Tradičné prístupy, ako sú čierne listiny, signatúrové metódy alebo heuristiky, poskytujú len obmedzenú ochranu. Tieto metódy zlyhávajú najmä pri identifikácii domén s dynamickým správaním, aké využívajú napríklad algoritmy na generovanie domén (DGA).

Lepšiu presnosť dosahujú moderné prístupy strojového učenia, ktoré využívajú vopred extrahované vlastnosti zo vstupných dát. Medzi najčastejšie využívané prístupy patrí napríklad algoritmus SVM, rozhodovací strom alebo náhodný les. Rovnako sú populárne aj architektúry hlbokého učenia, ako sú rekurentné neurónové siete typu LSTM alebo konvolučné neurónové siete (CNN). Hoci tieto metódy preukázali vysokú efektivitu, ich hlavnou nevýhodou je potreba manuálneho návrhu vstupných vlastností. Tento proces je časovo náročný, závislý od odborných znalostí a často nedokáže pokryť celú variabilitu správania škodlivých domén. Okrem toho majú niektoré architektúry, ako napríklad LSTM, problém so spracovaním dlhších sekvencií a sú menej vhodné na paralelné spracovanie dát, čo znižuje ich praktickú použiteľnosť pri spracovaní veľkých objemov dát.

Transformery do značnej miery tieto obmedzenia prekonávajú. Vďaka mechanizmu pozornosti dokážu zachytiť lokálne aj globálne vzory vo vstupných sekvenciách a efektívne modelovať kontext. Na rozdiel od rekurentných sietí umožňujú spracovanie dát paralelne, čo výrazne zvyšuje ich škálovateľnosť. Zároveň sa učia reprezentovať informácie priamo z pôvodných dát bez potreby manuálneho návrhu vlastností, čo vedie k vyššej automatizácii a flexibilite modelu. Tieto vlastnosti robia z transformerov vhodného kandidáta pre úlohu detekcie škodlivých domén. Skúmaním aplikovateľnosti transformerov v tejto oblasti táto práca neprispieva len k pokroku v oblasti kybernetickej bezpečnosti, ale tiež skúma potenciál týchto architektúr v novom kontexte.

Hlavným cieľom tejto práce je navrhnúť, implementovať a vyhodnotiť model na detekciu škodlivých domén založený na architektúre transformerov. Medzi konkrétne ciele patria:

- analýza taxonómie a správania škodlivých domén,
- prispôsobenie transformerovej architektúry na klasifikáciu domén pri rôznych typoch vstupných dát,
- vyhodnotenie výkonu modelov z hľadiska presnosti, škálovateľnosti a výpočtovej efektívnosti,
- porovnanie výsledkov s tradičnými modelmi strojového a hlbokého učenia,
- experimentálne overenie efektívnosti transformerových modelov pri detekcii jednotlivých kategórií domén.

Tieto ciele sú dosahované návrhom a implementáciou modelov nad viacerými typmi dát vrátane doménových mien, RDAP/Whois záznamov, DNS odpovedí a geografických informácií získaných z IP adries. Modely sú trénované bez potreby manuálneho extrahovania vlastností, s využitím transferového učenia a s dôrazom na výber vhodných tokenizačných stratégií pre jednotlivé kategórie domén.

V rámci experimentov sa podarilo dosiahnuť vysokú presnosť pri všetkých testovaných typoch dát. Detekcia DGA domén na základe názvu domény dosiahla skóre F1 98,61%. V prípade malvérových domén sa modely trénované na RDAP, DNS a geografických dátach pohybovali v rozsahu od 95,70% do 95,93%. Pri phishingových doménach boli najlepšie výsledky dosiahnuté na základe DNS údajov, konkrétne s skóre F1 98,39%. Tieto výsledky potvrdzujú efektívnosť navrhnutého riešenia a jeho pripravenosť na praktické nasadenie. Pre lepšiu orientáciu v obsahu práce je uvedený prehľad jednotlivých kapitol, ktoré odrážajú postupný vývoj riešenia od úvodnej analýzy až po experimentálne vyhodnotenie:

- **Kapitola 2** obsahuje podrobnú analýzu aktuálnych prístupov v rámci detekcie malígnych domén.
- **Kapitola 3** sa zaoberá analýzou malígnych domén, poskytuje hĺbkovú diskusiu o ich taxonómii, vlastnostiach a výzvach pri ich detekcii. použitých pre trénovanie modelu.
- **Kapitola 4** predstavuje teoretické základy neurónových sietí na báze transformerov, podrobne popisuje ich architektúru a analýzu výpočtovej zložitosti.
- **Kapitola 5** prezentuje návrh systému detekcie vrátane architektúry, tokenizácie a stratégie vyhodnocovania.
- **Kapitola 6** dokumentuje implementáciu systému a jeho experimentálnu časť.
- **Kapitola 7** obsahuje experimentálne overenie, vyhodnotenie a porovnanie navrhnutých modelov vrátane interpretácie výsledkov a porovnania s tradičnými prístupmi.
- **Kapitola 8** poskytuje diskusiu nad získanými výsledkami, hodnotí ich význam, identifikuje obmedzenia navrhovaného riešenia a porovnáva výsledky s existujúcimi prístupmi.
- **Kapitola 9** poskytuje zhrnutie získaných poznatkov, zhodnotenie prínosov a návrh ďalšieho smerovania.

Kapitola 2

Analýza aktuálnych prístupov detekcie malígnych domén

Detekcia škodlivých domén je aktívnou oblasťou výskumu vzhľadom na jej kritický význam v kybernetickej bezpečnosti. Medzi malígne domény možno zaradiť phishingové domény, domény distribujúce malvér, domény určené na komunikáciu útočníka s malvérom (napr. domény DGA) a pod. Presnejšia analýza a kategorizácia jednotlivých typov škodlivých domén je popísaná v nasledujúcej kapitole 3. Aktuálne boli navrhnuté rôzne prístupy na identifikáciu malígnych domén, kde tieto prístupy možno rozdeliť na tradičné, pokročilé prístupy využívajúce strojové učenie a prístupy založené na neurónových sieťach. Táto kapitola poskytuje aktuálny prehľad existujúcich štúdií.

2.1 Klasické prístupy

Klasické metódy detekcie škodlivých domén sa spoliehajú na štatistické a podobnostné metriky. Tieto prístupy vypočítavajú podobnosť medzi názvom domény a známymi databázami pomocou metrík, ako sú Levenshteinova vzdialenosť, Jaccardov index a Kullback-Leiblerova divergencia [46, 43, 7]. Konkrétne Zhang et al. [46] vyvinuli systém "DomainWatcher", ktorý využíva Levenshteinovu vzdialenosť na identifikáciu škodlivých domén na základe ich textovej podobnosti so známymi hrozbami, pričom dosiahli presnosť 96 %. Podobne Zhao et al. [48] navrhli prístup založený na segmentácii, pri ktorej sú názvy domén rozdelené na podreťazce a pre tieto segmenty sú vypočítané hodnoty reputácie. Tento systém dosiahol presnosť detekcie 94 % pre domény generované algoritmicky. Napriek jednoduchosti sú tieto metódy výpočtovo náročné pri aplikácii na veľké súbory dát a môžu mať problémy s rozpoznaním nových typov škodlivých domén.

obchádzateľná. hrozbám a môžu viesť k falošným poplachom. Detekcia na základe signatúr je obmedzená na známe hrozby a môže byť ľahko obídená úpravami škodlivého softvéru. Štatistická analýza môže byť výpočtovo náročná a nemusí detekovať všetky hrozby.

2.2 Prístupy s využitím strojového učenia

Strojové učenie sa stalo kľúčovým nástrojom pri odhaľovaní škodlivých webových lokalít, pretože dokáže automaticky rozpoznávať skryté vzory v dátach. Bežne sa využívajú algoritmy učenia s učiteľom ako náhodné lesy [1, 5], rozhodovacie stromy [28] či gradientné posilňovanie [13]. Napríklad systém FANCI [37], založený na náhodnom lese, detegoval

DGA domény s presnosťou 99 %. Selvi et al. [38] dosiahli 98 % presnosť využitím n-gramov a ďalších lexikálnych znakov. Efektívne boli aj modely učenia bez učiteľa, napr. zhlukovanie pomocou K-means, kde bolo skúmané rozdelenie škodlivých domén do zhlukov bez predchádzajúcej anotácie dát [7]. Tieto prístupy však vyžadujú dôkladné inžinierstvo vlastností, čo je náročné pri rýchlo sa meniacich hrozbách. Modely strojového učenia boli úspešne využité aj pri detekcii phishingových a malvérových domén. Sadique et al. [35] dosiahli 87 % presnosť pri detekcii phishingu kombináciou lexikálnych, WHOIS, geografických a príznakov týkajúcich sa infraštruktúry domén. Hason et al. [15] identifikovali robustné znaky pre detekciu phishingových a C&C domén, zatiaľ čo Shi et al. [39] využili kombináciu WHOIS, IP a lexikálnych charakteristík. Palaniappan et al. [31] aplikovali model s DNS a webovými príznakmi na detekciu malvérových domén, pričom na datasete s 20 000 doménami dosiahli presnosť 60 %. Strojové učenie je v porovnaní s klasickými metódami účinnejšie, avšak rýchla evolúcia malígnych domén si vyžaduje pravidelné aktualizácie modelov, čo je časovo náročný proces, najmä kvôli vhodnej identifikácii a následnej extrakcii kľúčových znakov z domén.

2.3 Prístupy využívajúce hlboké učenie

Pokroky v hlbokom učení umožnili vývoj modelov, ktoré dokážu automaticky z dát extrahovať komplexné vzory. Bežne sa používajú konvolučné neurónové siete (CNN) a rekurentné neurónové siete (konkrétne LSTM[1]). Xu et al. [42] využili CNN na analýzu štrukturálnych vzorov v názvoch domén, pričom dosiahli presnosť 98 %. Hybridné modely kombinujúce CNN a LSTM, prezentované v práci od Yang et al. [44], ďalej zlepšili detekčné schopnosti zavedením mechanizmov vlastnej pozornosti na zachytenie kontextových závislostí, pričom dosiahli až 98,9 % presnosť. Ďalší hybridný prístup od Niu et al. [30] použil LSTM optimalizované pomocou Bayesovskej metódy, čím dosiahol presnosť 97 %. Tieto metódy vynikajú presnosťou, ale napr. CNN majú obmedzenú schopnosť rozpoznávať vzdialené vzťahy v sekvenciách, zatiaľ čo LSTM sú menej efektívne pri spracovaní dlhých sekvencií kvôli sekvenčnému spracovaniu dát, ktoré bráni paralelizácii potrebnej pre rozsiahle datasetové aplikácie.

2.4 Prístupy založené na spracovaní prirodzeného jazyka

Metódy založené na NLP[2] spracúvajú doménové mená ako text a aplikujú na domény techniky ako tokenizácia, syntaktická analýza a sémantická reprezentácia. Tieto metódy sú obzvlášť účinné pri detekcii phishingových alebo domén DGA generovaných slovníkom. Napríklad Yang et al. [45] navrhli model sémantickej reprezentácie elementov (SERM) na identifikáciu hybridných domén DGA, pričom dosiahli presnosť 97,48 %. Buber et al. [3] využili techniky NLP na detekciu vizuálnych podobností v phishingových doménach, pričom dosiahli presnosť 97,2 %. Metódy NLP však často narážajú na problémy, keď názvy domén neobsahujú zmysluplné lingvistické štruktúry, napríklad keď sú tvorené náhodnými alebo neštandardnými znakmi, čo obmedzuje ich účinnosť v určitých scenároch.

[1] Long-Short Term Memory Neural Networks
[2] Natural Language Processing

Kapitola 3

Charakteristika malígnych domén: phishing, malvér a DGA

Škodlivé domény sú internetové adresy využívané na vykonávanie nelegálnych alebo škodlivých aktivít. Tieto domény môžu byť vytvorené špecificky na škodlivé účely alebo získané kompromitáciou existujúcich legitímnych domén. Systém DNS[1], ktorý slúži ako základná zložka internetovej infraštruktúry, môže byť zneužitý na podporu týchto aktivít. Škodlivé domény sú navrhnuté tak, aby efektívne podporovali kriminálne operácie a umožnili útočníkom získať citlivé údaje, finančný prospech alebo spôsobiť narušenie informačných systémov. Vďaka rastúcej sofistikovanosti týchto útokov je ich detekcia a neutralizácia stále náročnejšia [49].

3.1 Použitie malígnych domén

Kybernetickí útočníci zneužívajú domény na realizáciu širokej škály útokov, pričom tieto škodlivé adresy slúžia ako kľúčové nástroje na manipuláciu používateľov, zneužívanie zraniteľností a vykonávanie podvodných či škodlivých aktivít. Nasledujúce príklady demonštrujú najčastejšie využitie malígnych domén:

- **Phishingové webové stránky** – Stránky, ktoré imitujú legitímne webové lokality alebo služby s cieľom ukradnúť prihlasovacie údaje, osobné informácie alebo finančné dáta. Využívajú preklepy alebo podobnosti v názvoch (napr. `paypa1.com` namiesto `paypal.com`).
- **Webové stránky na distribúciu malvéru** – Hosťujú škodlivé súbory, ako sú trójske kone, ransomware alebo spyware, pričom sa často tvária ako legitímne zdroje softvéru.
- **Servery Command and Control (C&C)** – Domény, ktoré slúžia na riadenie botnetov alebo iných škodlivých aktivít, pričom umožňujú útočníkom kontrolu nad infikovanými zariadeniami.
- **Cryptojacking** – Domény, ktoré hostujú skripty (napr. JavaScriptové minery), ktoré ťažia kryptomeny priamo v prehliadači používateľa. Tento proces využíva výpočtový výkon zariadenia bez súhlasu používateľa a môže výrazne spomaliť jeho výkon, pričom si útok nevyžaduje stiahnutie malvéru a stačí, aby užívateľ navštívil príslušnú stránku.

[1] Domain Name System

Včasná detekcia a následné blokovanie týchto domén pomáha minimalizovať finančné straty spôsobené krádežou údajov, podvodmi alebo poškodením infraštruktúry. Rovnako prispieva k ochrane citlivých údajov používateľov a organizácií pred neoprávneným prístupom a exfiltráciou. Detekcia škodlivých domén zabezpečuje kontinuitu a stabilitu systémov tým, že prerušuje komunikáciu medzi infikovanými zariadeniami a útočníkmi, čo je zásadné napríklad pri riadení botnetov. Okrem toho pomáha organizáciám zvyšovať dôveryhodnosť a reputáciu elimináciou rizika kompromitácie ich domén alebo infraštruktúry.Efektívna obrana proti škodlivým doménam je zároveň dôležitá pre dodržiavanie legislatívnych a regulačných požiadaviek, ktoré kladú dôraz na ochranu údajov a prevenciu kybernetických útokov, a predstavuje kľúčový krok k zníženiu rizík spojených s modernými kybernetickými hrozbami.

3.2 Taxonómia techník zneužívania domén

Metódy zneužívania doménových mien boli vyvinuté s cieľom narušiť ich integritu a poškodiť reputáciu, pričom hlavné ciele týchto praktík zahŕňajú [14]:

- **Generovanie domén** – Vytvorenie veľkého množstva domén obchádzajúcich detekciu a zároveň pôsobiacich dôveryhodne.
- **Riešenie nedostatku IP adries z pohľadu útočníka** – Prepojenie viacerých domén na jednu IP adresu. Prostredníctvom prepojenia viacerých domén s jedinou IP adresou sa znižuje viditeľnosť jej škodlivej činnosti.
- **Imitácia legitímnych domén** – Úprava názvov domén tak, aby pripomínali legitímne, bez vzbudenia podozrenia.

Na tieto účely sa využívajú techniky ako algoritmy generovania domén (DGA), domain-flux a preklepové útoky (typosquatting). Pochopenie týchto metód je nevyhnutné pre vývoj účinných detekčných systémov. Nasledujúce časti vysvetľujú princípy ich fungovania.

3.2.1 Algoritmy Generovania Domén (DGA)

Algoritmy generovania doménových mien (DGAs) sú mechanizmus, ktorý umožňuje automaticky vytvoriť množstvo doménových mien z počiatočného vstupu alebo počiatočnej podmienky. Kybernetickí útočníci a botnety ich využívajú na generovanie nových domén podľa času alebo dátumu, čím sťažujú ich odstránenie zo zónových súborov a zároveň umožňujú útočníkom vyhnúť sa detekcii a sťažiť tak odhalenie ich aktivít [41]. Algoritmy DGA sa klasifikujú na základe dvoch hlavných aspektov [47, 4, 32]:

1. **Počiatočný parameter (Initial Seed)** – algoritmus pre generovanie domén vyžaduje rôzne vstupné parametre, napríklad:
   - Numerické konštanty – dĺžka domény, pseudo-náhodné čísla.
   - Znakové sady – abecedy, domény najvyššej úrovne (TLD).

   Tieto parametre môžu mať:
   - **Závislosť na čase** – využitie časových vstupov na generovanie domén.

- **Deterministickosť** – schopnosť konzistentne reprodukovať domény pri rovnakých vstupoch.

2. **Schéma generovania (Generation Scheme)** – stanovuje postup vykonávania programu. Medzi najčastejšie schémy generovania patria:
   - **Aritmetická** – využíva výpočtové hodnoty, často odvodené z ASCII alebo tabuľkových hodnôt.
   - **Hashovacia** – používa hashovacie funkcie, ako MD5 alebo SHA256, na generovanie domén.
   - **Zoznam slov** – kombinuje slová z preddefinovaných zoznamov na tvorbu jednoduchších, menej náhodných domén.
   - **Permutačná** – upravuje komponenty danej domény na vytvorenie všetkých možných variácií.

Na základe tejto klasifikácie je možné vidieť rôznorodosť spôsobov, akými sa domény DGA generujú s cieľom vyhnúť sa detekcii.

3.2.2 Domain-Flux

Botnety využívajú techniku domain-flux, ktorá spája veľký počet plne kvalifikovaných doménových mien (FQDN) s jednou IP adresou, čím sa snažia vyhnúť detekcii a zaradeniu na čiernu listinu. Táto taktika poskytuje vysokú mieru adaptability a pomáha útočníkom obísť bezpečnostné opatrenia, čo je obzvlášť účinné pri útokoch typu command-and-control (C&C). Príklady botnetov, ako sú Conficker, Kraken a Torpig, využívajú DNS na dynamické presmerovanie domén, pričom boty postupne prechádzajú medzi doménami, ktoré útočník priebežne registruje [49].

Aby bolo možné zastaviť tieto aktivity, je často potrebné analyzovať a pochopiť algoritmy, ktoré generujú domény (DGA). Autoritatívne DNS servery pritom spracovávajú dotazy iteratívnym spôsobom, čo komplikuje detekciu. Krátka životnosť týchto dynamicky vytvorených domén navyše sťažuje ich monitorovanie a efektívne blokovanie.

3.2.3 Typosquatting

Útočníci využívajú metódu registrácie podobných doménových mien (typosquatting), pri ktorej si zabezpečujú domény s názvami obsahujúcimi drobné typografické úpravy originálnych názvov, ako napríklad `g00gle.com` alebo `gooogle.com` namiesto `google.com`. Táto technika zneužíva chyby používateľov pri zadávaní domén a predstavuje vážne riziká, ako sú podvody, krádež údajov a získavanie dôverných firemných informácií [49]. Varianty tejto techniky sú napríklad:

- **Bitsquatting** – využíva náhodné bitové chyby v hardvéri na vytváranie domén, ktoré sa od originálu líšia iba jedným bitom (napr. `micposoft.com` namiesto `microsoft.com`) [8].
- **Soundsquatting** – registruje domény s názvami, ktoré foneticky pripomínajú legitímne názvy, aby oklamali používateľov pri mylnom použití podobne znejúcich slov (napr. `whetherportal.com` namiesto `weatherportal.com`) [29].

- **Combosquatting** – spája názvy známych značiek s ďalšími termínmi, aby navodil dojem dôveryhodnosti bez zmeny hlavného názvu domény (napr. `apple.com.recover.support`) [21].
- **Homographsquatting** – nahrádza znaky v názvoch domén podobne vyzerajúcimi symbolmi alebo Unicode znakmi, čím ich robí na pohľad nerozoznateľnými od originálu (napr. `fäcebook.com`) [10].
- **Levelsquatting** – vkladá legitímne názvy značiek do subdomén škodlivých domén, pričom zneužíva obmedzené zobrazenie URL na mobilných zariadeniach (napr. `google.com.virus.com`) [9].

Tieto techniky cielene využívajú chyby používateľov a technologické zraniteľnosti, čím sa stávajú účinným nástrojom na realizáciu podvodov a škodlivých aktivít.

3.3 Špecifické vlastnosti škodlivých domén

Pre účinnú detekciu škodlivých domén je nevyhnutné analyzovať ich charakteristické črty. Modely strojového učenia na detekciu škodlivých doménových mien využívajú širokú škálu vlastností a možno ich rozdeliť na štatistické, lingvistické, všeobecné a behaviorálne vlastnosti, pričom každá kategória poskytuje jedinečné poznatky pre detekčný proces. Toto rozdelenie zabezpečuje jasnú štruktúru pre analýzu rôznorodých charakteristík, čo umožňuje komplexné pokrytie a uľahčuje vývoj robustných detekčných systémov. Nižšie je detailný popis týchto kategórií vrátane všeobecných popisov a štrukturálnych príkladov [14, 16, 27].

3.3.1 Štatistické vlastnosti

Štatistické vlastnosti kvantifikujú merateľné aspekty doménových mien, ktoré môžu odhaliť vzorce naznačujúce škodlivé zámery.

- **Dĺžka doménového mena** – Škodlivé domény často prekračujú obvyklú dĺžku, niekedy až 250 znakov, zatiaľ čo bežné domény málokedy presiahnu 75 znakov. Nadmerná dĺžka sa používa na zvýšenie zložitosti alebo vkladanie zavádzajúcich informácií.
  - Príklad (benígne): `example.com` (11 znakov).
  - Príklad (škodlivé): `secure-account-login-verification-updates.com` (49 znakov).
- **Počet subdomén** – Bežné domény obsahujú v priemere tri subdomény. Škodlivé domény často zahŕňajú väčší počet (až 35), aby napodobňovali legitímne subdomény (príliš dlhé domény prehliadač nezobrazí a je zobrazená len legitímne vyzerajúca časť doménového mena) alebo zakryli svoj pôvod.
  - Príklad (benígne): `store.example.com`.
  - Príklad (škodlivé): `secure.update.login.verification.example.com`.
- **Pomer číslic a znakov v doméne** – Legitímne domény zriedkavo obsahujú prechody medzi znakmi a číslicami, okrem výnimočných prípadov značiek alebo identifikátorov produktov. Škodlivé domény často vykazujú nepravidelné prechody na zakrytie svojho účelu.

  - Príklad (benígne): `brand2022.com`.
  - Príklad (škodlivé): `v3r1fy-secure-login78.com`.

- **Počet číslic a neprerušených znakov** – Škodlivé domény často zahŕňajú dlhé číselné sekvencie alebo nepretržité reťazce znakov, zatiaľ čo bežné domény sa týmto vzorom vyhýbajú kvôli čitateľnosti.
  - Príklad (benígne): `product-update.com`.
  - Príklad (škodlivé): `123456-secure-login-update.com`.

3.3.2 Lexikálne vlastnosti

Lingvistické vlastnosti analyzujú sémantickú a syntaktickú skladbu doménových mien, zameriavajúc sa na ich podobnosť s prirodzeným jazykom alebo zmysluplnými štruktúrami.

- **Analýza N-gramov** – Táto technika rozdeľuje doménové mená na malé prekrývajúce sa sekvencie znakov (N-gramy), aby identifikovala vzorce podobné legitímnym slovám alebo náhodným sekvenciám.
  - Príklad (benígne): `bankofamerica.com` → N-gramy: `bank`, `of`, `amer`, `ica`.
  - Príklad (škodlivé): `bnkofamer1ca-login.com` → N-gramy: `bnk`, `of`, `amer`, `1ca`.

- **Podozrivé kľúčové slová** – Niektoré kľúčové slová sa často objavujú v škodlivých doménach na oklamanie používateľov (napr. `login`, `update`, `secure`). Tieto kľúčové slová sú označené ako potenciálne rizikové.
  - Príklad (benígne): `example.com`.
  - Príklad (škodlivé): `secure-login.example.com`.

- **Sekvencie pripomínajúce slová** – Hodnotenie, či doménové mená pripomínajú zmysluplné alebo prirodzené slovné sekvencie, pomáha odhaliť zavádzajúce vzory. Škodlivé domény môžu zahŕňať úmyselné preklepy alebo imitácie.
  - Príklad (benígne): `google.com`.
  - Príklad (škodlivé): `goog1e-login-secure.com`.

3.3.3 Všeobecné vlastnosti

Všeobecné vlastnosti skúmajú širšie atribúty, ako sú domény najvyššej úrovne (TLD), štruktúry subdomén a informácie súvisiace so sieťou.

- **Informácie o TLD** – Niektoré TLDs sú častejšie spojené so škodlivou aktivitou (napr. `.info`, `.xyz`), zatiaľ čo iné, ako `.com`, sú typicky benígne.
  - Príklad (benígne): `example.com`.
  - Príklad (škodlivé): `secure-update.info`.

- **Subdomény** – Škodlivé domény často zneužívajú štruktúry subdomén na napodobnenie legitímnych služieb alebo skrytie svojej identity.
  - Príklad (benígne): `shop.example.com`.
  - Príklad (škodlivé): `login.verify.secure.example.com`.

- **Priradenie IP adries** – Škodlivé domény môžu byť prepojené s IP adresami známymi pre hosťovanie škodlivých aktivít, čo poskytuje ďalšiu vrstvu detekcie.
  - Príklad (benígne): Doména sa mapuje na stabilnú, známu IP adresu.
  - Príklad (škodlivé): Doména sa mapuje na dynamickú alebo blokovanú IP adresu.

3.3.4 Vývoj trendov domén

Škodlivé domény často reagujú na aktuálne trendy alebo krízové situácie, ako sú pandémie či geopolitické udalosti, aby zneužili verejný záujem a zvýšili pravdepodobnosť návštevnosti používateľov. Nasledujúci príklad ilustruje ukážku takejto domény:

- Príklad (benígne): `store.example.com`.
- Príklad (škodlivé): `covid19-vaccine-update.com`.

Kapitola 4

Architektúra Transformerov

Neurónové siete na báze transformerov predstavujú prelomovú technológiu v oblasti umelej inteligencie, najmä v spracovaní prirodzeného jazyka a v získavaní znalostí zo sekvenčných dát. Na rozdiel od tradičných modelov, ktoré spracúvajú informácie postupne, transformery využívajú mechanizmus pozornosti, ktorý umožňuje paralelné spracovanie vstupných dát. Vďaka tomu dosahujú vysokú efektivitu a flexibilitu. V tejto kapitole je predstavený základný princíp fungovania transformerov, popis architektúry a princíp fungovania jednotlivých komponentov architektúry.

4.1 Princíp fungovania transformátorových neurónových sietí

Mechanizmus pozornosti, ktorý transformery využívajú, priniesol zásadnú zmenu v spôsobe spracovania sekvenčných dát a riešenia problémov so vzdialenými súvislosťami. Hlavnou výhodou tejto architektúry je schopnosť efektívne priraďovať váhy rôznym častiam vstupnej sekvencie podľa ich relevantnosti pre konkrétnu úlohu. Vďaka tomu je možné efektívne zachytiť jemné kontextové informácie. Kľúčové kroky v rámci fungovania transformátorov zahŕňajú [19]:

rekurencii či konvolúcii. Hlavnou inováciou je mechanizmus pozornosti, ktorý priraďuje váhy rôznym častiam vstupnej sekvencie na základe ich relevantnosti pre danú úlohu. Vďaka tomu sú schopné efektívne zachytiť dlhodobé závislosti a kontextovú informáciu. Kľúčové kroky v rámci fungovania transformátorov zahŕňajú:

1. **Tokenizáciu a zakódovanie** – konverzia vstupných dát do vektorového priestoru.
2. **Pozičné zakódovanie** – keďže transformátory spracovávajú sekvencie paralelne, k vektorovým reprezentáciám vstupných dát sa pridávajú pozičné zakódovania, aby sa zachovala informácia o poradí tokenov vo vstupnej sekvencii.
3. **Výpočet pozornosti** – model určuje dôležitosť jednotlivých prvkov pre daný kontext.
4. **Plne prepojené vrstvy** – výstup z mechanizmu pozornosti sa spracováva prostredníctvom plne prepojených vrstiev pre ďalšie spracovanie.
5. **Kombinácia vrstiev** – transformátory pozostávajú z viacerých vrstiev mechanizmu pozornosti a plne prepojených vrstiev, čo umožňuje hierarchické učenie reprezentácií.

Vyššie uvedený prehľad slúži na základné oboznámenie s fungovaním transformátorov, pričom jednotlivé komponenty architektúry budú podrobnejšie vysvetlené v nasledujúcej sekcii.

4.2 Prehľad architektúry

Po úvodnom predstavení základnej funkcionality transformerov sa táto sekcia podrobnejšie zaoberá jednotlivými vnútornými mechanizmami. fungovania týchto modelov. V prípade detekcie malígnych domén je detailná znalosť architektúry nevyhnutná pre návrh vhodných úprav a zlepšenie výkonu modelu. Architektúra transformera pozostáva z dvoch hlavných častí [40]:

1. enkodér – spracováva vstupné dáta a vytvára ich vektorovú reprezentáciu, ktorá sa následne používa pri generovaní výstupu,
2. dekodér – generuje výstupnú sekvenciu na základe vstupu z enkodéra a predchádzajúcich výstupov.

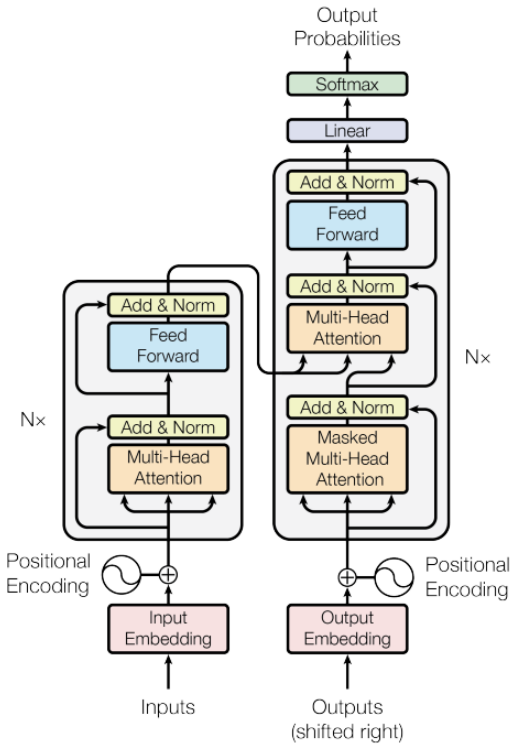
Každá z týchto častí je vytvorená skladaním viacerých vrstiev (napr. 6 alebo 12 vrstiev), ktoré majú podobnú vnútornú štruktúru. Podľa pôvodnej práce [40] sa enkodér skladá z

- **zakódovanie vstupu** (Input Embedding) – konvertuje vstupné tokeny (napr. znaky alebo slová) do vektorovej formy, ktorú vie model spracovať,
- **pozičné zakódovanie** (Positional Encoding) – pridáva informáciu o poradí tokenov v sekvencii, pretože samotná architektúra nemá vnútorný pojem o pozícii,
- **viacvrstvová pozornosť** (Multi-Head Attention) – umožňuje modelu sledovať vzťahy medzi rôznymi časťami vstupnej sekvencie paralelne a z viacerých perspektív,
- **dopredná neuronová sieť** (Feed-Forward Network) – aplikuje nelineárnu transformáciu na každú pozíciu samostatne a zvyšuje výpočtovú kapacitu modelu,
- **reziduálne prepojenia a normalizácia vrstiev** (Add & Norm) – zlepšujú stabilitu trénovania, udržiavajú gradient a umožňujú efektívnejší prenos informácií medzi vrstvami.

Dekodér má podobnú štruktúru, ale navyše obsahuje nasledujúce komponenty [34, 19]:

- **vymaskovaná viacvrstvová pozornosť** (Multi-Head Self-Attention),
- **viacvrstvová pozornosť**, ktorá má ako vstupné dáta výstup enkodéra (kvôli mechanizmu pozornosti zameranej na výstupy enkodéra).

Vo finálnej fáze sa výstupy dekódera prejdú cez lineárnu vrstvu a funkciu softmax a na výstupe dekódera je pravdepodobnostné rozdelenie nad slovnou zásobou. Obrázok 4.1 zobrazuje architektúru transformátoru, ktorá bola prvýkrát predstavená v roku 2017 [40].



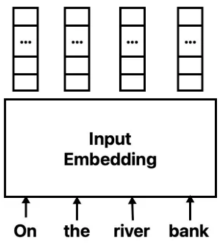
Obr. 4.1: Architektúra transformátorovej neurónovej siete

4.3 Popis jednotlivých súčastí architektúry

V tejto sekcii sú detailnejšie popísané jednotlivé komponenty architektúry. Z popisu v sekcii 4.2 je možné si všimnúť, že komponenty v enkodéri a dekodéri sú takmer identické, len s drobnými úpravami (napr. maskovaná pozornosť a mechanizmus krížovej pozornosti – cross-attention medzi enkodérom a dekodérom).

4.3.1 Zákódovanie vstupu (Input Embedding)

Každý token (napríklad slovo, znak alebo podslovo) je v dátach reprezentovaný číslom. Keďže model nedokáže tieto číselné hodnoty priamo interpretovať ako jazykové vstupy, každému tokenu je priradený vektor reálnych čísel (embedding), ako ilustruje obrázok 4.2.



Obr. 4.2: Ilustrácia transformácie vstupných tokenov na vektory spojitých hodnôt. [26].

Tokeny, ktoré sú si významovo alebo syntakticky podobné (napr. synonymá alebo podobné znaky), dostávajú vektory, ktoré sú si v tomto priestore blízke [34, 12]. Komponenta zakódovania vstupu má teda nasledujúce funkcie:

- Transformuje diskrétne ID tokenov na viacdimenzionálny vektor spojitých hodnôt, kde číslo $d_{model} \in \mathbb{N}$ udáva dimenziu vektora.
- Využíva geometrické vlastnosti vektorového priestoru na zachytenie podobností a rozdielov medzi tokenmi. Tokeny s podobným významom sú v tomto priestore reprezentované vektormi, ktoré sú umiestnené blízko seba.

reprezentáciu pomáha zachytiť sémantické alebo syntaktické vzťahy v spojitom priestore. Formálnu reprezentáciu zakódovania vstupného tokenu x na vektor $e(x)$ je možné zapísať vzorcom 4.1:

$$e(x) = \begin{bmatrix} E_{x,1} \\ E_{x,2} \\ \vdots \\ E_{x,d_{model}} \end{bmatrix} \tag{4.1}$$

kde $E \in \mathbb{R}^{V \times d_{model}}$ je matica vektorových reprezentácií vstupných tokenov a $E_{x,i}$ predstavuje i-tu zložku vektoru na riadku zodpovedajúcom tokenu x.

4.3.2 Pozičné zakódovanie (Positional Encoding)

Na rozdiel od rekurentných neurónových sietí (RNN), transformátorové neurónové siete nečítajú vstupy sekvenčne, a preto nemajú vnútornú informáciu o poradí tokenov. Transformer spracováva všetky tokeny paralelne, preto potrebuje spôsob, ako rozoznať poradie jednotlivých tokenov v sekvencii. Hlavnou úlohou pozičného zakódovania je zachovanie informácie o poradí jednotlivých tokenov, preto k vektorovým reprezentáciám tokenov je pripočítaná hodnota, ktorá túto pozíciu tokenu v sekvencii reprezentuje. Zhrnutie hlavnej funkcionality tejto komponenty je nasledovné:

- Vnáša informáciu o polohe (indexe) každého tokenu v sekvencii.
- Zaručuje, že model identifikuje rovnaké tokeny nachádzajúce sa na rôznych pozíciách (napr. viacnásobné výskyty rovnakého slova vo vete).

Formálne je možné definovať pozičné zakódovanie ako [40]:

$$PE(p,i) = \begin{cases} \sin\left(\frac{p}{10000^{\frac{i}{d_{model}}}}\right), & \text{ak } i \text{ je párne,} \\ \cos\left(\frac{p}{10000^{\frac{i-1}{d_{model}}}}\right), & \text{ak } i \text{ je nepárne.} \end{cases} \tag{4.2}$$

$$z(p) = e(x_p) + PE(p) \tag{4.3}$$

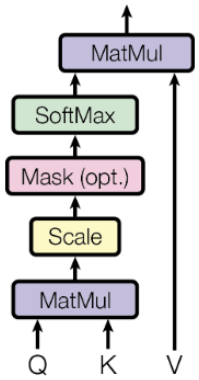
Premenná p vo vzorci reprezentuje index pozície tokenu v sekvencii (pre prvý token $p = 0$, pre druhý token $p = 1, \ldots$). Premenná i predstavuje index dimenzie vo vektorovej reprezentácii tokenu, keďže každá dimenzia vektoru dostane svoju vlastnú frekvenciu, tak rôzne dimenzie oscilujú rôzne rýchlo. Rôzne frekvencie naprieč dimenziami vektora umožňujú zachytiť pozície na rôznych úrovniach rozlíšenia. Vďaka tomu môže model ľahko rozoznať

relatívne vzdialenosti medzi tokenmi (napr. rozdiel 1 pozície, 2 pozícií, 10 pozícií a pod.). Poslednou premennou je d_{model}, ktorá udáva dimenzionalitu vstupných zakódovaných vektorov. Zo vzorca 4.2 vyplýva, že do párnych zložiek vektoru vkladáme sínusové hodnoty a do nepárnych zložiek kosínusové hodnoty. Trigonometrické funkcie sa používajú preto, lebo sú periodické — ich pravidelné "vlny" pomáhajú modelu ľahko rozpoznať, ako ďaleko od seba sa tokeny nachádzajú. V jednotlivých dimenziách sa mení frekvencia oscilácie (daná exponentom a konštantou 10 000), takže model dokáže pokryť rôzne škály vzdialeností medzi slovami [34]. Po vypočítaní vektorov s pozičným kódovaním sú tieto vektory sčítané s príslušnými vstupnými vektormi, čo ukazuje rovnica 4.3. Týmto spôsobom si transformátorové neurónové siete vedia udržať informáciu o poradí aj bez rekurentných vrstiev.

4.3.3 Viacvrstvová pozornosť (Multi-Head Attention)

Modul viacvrstvovej pozornosti je základným prvkom architektúry, ktorý umožňuje modelu efektívne zachytávať kontextuálne vzťahy medzi jednotlivými tokenmi v sekvencii. Tento mechanizmus je schopný paralelne analyzovať rôzne aspekty vzťahov medzi tokenmi, čím zvyšuje výpočtovú efektivitu a kapacitu modelu. Na podrobnejšie objasnenie jeho fungovania je potrebné najprv definovať samotný mechanizmus pozornosti (self-attention), ktorý je implementovaný v každej vrstve (hlave) pozornosti a tvorí základnú stavebnú jednotku pri analýze kontextu.

Mechanizmus pozornosti (Self-Attention) porovnáva každý token (napr. slovo vo vete) so všetkými ostatnými tokenmi v sekvencii, aby určil, ktoré tokeny sú v danej sekvencii preň najpodstatnejšie. Schéma mechanizmu pozornosti je ilustrovaná na obrázku 4.3.



Obr. 4.3: Schéma mechanizmu pozornosti

Formálne možno mechanizmus pozornosti zapísať nasledovne [40]:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{4.4}$$

kde:

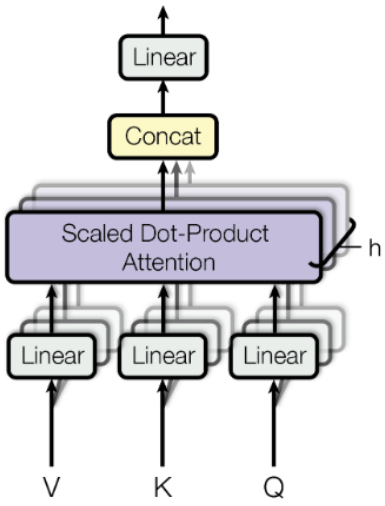
- **Q** (Query) je matica požiadaviek, ktorá reprezentuje požiadavky tokenu na kontext ostatných tokenov v sekvencii,
- **K** (Key) je matica kľúčov, ktorá slúži na charakterizáciu vlastností tokenov a umožňuje hodnotenie ich relevantnosti voči požiadavkám (queries),
- **V** (Value) je matica hodnôt, ktorá obsahuje informácie asociované s jednotlivými tokenmi a ktoré sa následne kombinujú na základe vypočítaných váh,
- $\mathbf{d_k}$ predstavuje dimenziu vektorov, ktorá sa používa na škálovanie, čím sa zabezpečuje numerická stabilita výsledných hodnôt.

Ako už bolo uvedené, mechanizmus pozornosti vytvára projekcie vstupných vektorov v sekvencii do troch rôznych reprezentácií: **Query** (Q), **Key** (K) a **Value** (V). Matica Query predstavuje otázky, ktorými sa token pýta na relevantnosť ostatných tokenov, matica Key charakterizuje vlastnosti tokenov, ktoré odpovedajú na tieto otázky, a matica Value obsahuje prenášané informácie. Tieto projekcie sa definujú pomocou váhových matíc W^Q, W^K, W^V a ich hodnoty sú určené procesom učenia počas tréningu. Matematicky sa tieto projekcie vyjadrujú nasledovne:

$$\begin{aligned} Q &= XW^Q, \\ K &= XW^K, \\ V &= XW^V, \end{aligned}$$

kde X je vstupná matica, ktorá predstavuje vektorovú reprezentáciu slov v sekvencii. Na výpočet vzťahov medzi tokenmi sa používa skalárny súčin medzi Q a K, čím je vyjadrená miera podobnosti medzi tokenmi. Aby sa zabránilo problémom s numerickou stabilitou pri veľkých hodnotách, výsledky sú upravené delením $\sqrt{d_k}$, kde d_k je dimenzia vektorov kľúčov. Následne sa na tieto hodnoty aplikuje funkcia *Softmax*, ktorá normalizuje hodnoty do rozsahu $[0, 1]$ a transformuje tieto podobnosti na pravdepodobnostnú distribúciu, ktorá reprezentuje dôležitosť jednotlivých tokenov. Tieto normalizované hodnoty predstavujú váhy, ktoré označujú, do akej miery sú jednotlivé tokeny relevantné pre konkrétny token. Kombinácia tejto váhovej distribúcie s maticou hodnôt V umožňuje modelu prenášať relevantné kontextové informácie ku každému tokenu.

Viacvrstvová pozornosť (Multi-Head Attention) tento princíp rozširuje tým, že aplikuje mechanizmus pozornosti súbežne vo viacerých vrstvách (hlavách). Každá vrstva pritom môže používať iné váhy, a tak v rôznych podpriestoroch vstupných vektorov sa zameriava na odlišné črty alebo rôzne druhy vzťahov medzi tokenmi. Vďaka tomu model dokáže zachytiť viac rôznorodých kontextových informácií naraz, čo prispieva k lepšiemu porozumeniu celej sekvencie. Architektúra viacvrstvovej pozornosti je ilustrovaná na obrázku 4.4.



Obr. 4.4: Architektúra viacvrstvovej pozornosti

Matematicky je možné zapísať Viacvrstvovú pozornosť nasledovne [40]:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O, \tag{4.5}$$

kde každá jednotlivá hlava head_i je vypočítaná ako:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \tag{4.6}$$

Váhové matice W_i^Q, W_i^K, W_i^V transformujú pôvodné Q, K, V do iného podpriestoru pre i-tú hlavu. Operácia *Concat* následne zreťazí všetky hlavy za sebou. Váhová matica W^O slúži ako výsledná transformácia, ktorá skombinované informácie zo všetkých hláv prevádza do jedného výstupného priestoru. Pre lepšie pochopenie je demonštrovaný mechanizmus pozornosti na nasledujúcom príklade:

> **Predmet: Dôležité upozornenie na podozrivú aktivitu**
> Vážený používateľ,
> zaznamenali sme bezpečnostné riziko na vašom bankovom účte. Prosíme vás, kliknite na odkaz:
> `http://secure-account-checkers.com/bank/login`
> Inak dôjde k zablokovaniu vášho účtu do 24 hodín.
> S pozdravom, Vaša banka

Mechanizmus pozornosti určí pre každý token (napr. *"bankovom"*, *"účet"*, *"kliknite"*, *"secure-account-checkers.com"*, *"zablokovaniu"*) ako veľmi súvisí s ostatnými. Napríklad *„kliknite"* môže upriamiť pozornosť na škodlivú doménu *"secure-account-checkers.com"*, a *"zablokovanie účtu"* zas na urgentnú hrozbu. Takto model dokáže detekovať, že e-mail je podozrivý, pretože obsahuje naliehavý tón a neobvyklú URL adresu.

Viacvrstvová pozornosť spracováva tento e-mail naraz, pričom každá vrstva sa zameriava na odlišný typ informácií. Jedna vrstva môže sledovať len slová spojené s bezpečnosťou, ako napríklad *"riziko"*, *"podozrivú"* či *"zablokovanie"*. Druhá vrstva sa môže zamerať na doménu odkazu a porovnať ju s legitímnymi bankovými doménami, ktoré model pozná. Tretia vrstva môže zisťovať, či text používa nátlakový tón, čo je typický prvok phishingu, napríklad výrazy *"ihneď"* alebo *"do 24 hodín"*. Následne sa tieto informácie spoja, čím model získa ucelený obraz — vidí všetky podozrivé signály naraz, ako sú falošná doména, urgentná výzva alebo nejasné tvrdenia o *"riziku"*. Vďaka tomuto prístupu model dokáže nielen rozpoznať kľúčové varovné tokeny, ako sú konkrétne slová alebo odkazy, ale aj pochopiť ich vzájomné súvislosti.

Zjednodušene povedané, mechanizmus pozornosti zisťuje dôležitosť tokenov voči sebe navzájom, a viacvrstvová pozornosť ho rozkladá do viacerých paralelných pohľadov, aby bol výsledný kontext čo najkomplexnejší.

4.3.4 Dopredná neurónová sieť (Feed-Forward Network)

Hlavnou úlohou doprednej neurónovej siete (FFN) je spracovanie každého tokenu v sekvencii samostatne a nezávisle, čím pridáva ďalšiu vrstvu transformácie k dátam, ktoré už prešli mechanizmom pozornosti. Táto transformácia zohráva kľúčovú úlohu pri reprezentácii a pochopení zložitých vzorcov v údajoch [11].

Každý token je reprezentovaný vektorom čísel, ktoré opisujú jeho vlastnosti, ako význam, pozíciu alebo vzťahy k iným tokenom. Feed-forward sieť aplikuje na tento vektor niekoľko operácií. Prvým krokom je lineárna transformácia reprezentovaná rovnicou 4.7, pri ktorej sa vektor $\mathbf{x}$ vynásobí maticou váh W_1 a pridá sa k nemu bias $\mathbf{b}_1$:

$$\mathbf{z} = \mathbf{x}W_1 + \mathbf{b}_1. \tag{4.7}$$

Táto operácia umožňuje modelu upraviť alebo preusporiadať informácie v pôvodnom vektore, pričom vzniká medzivýsledok $\mathbf{z}$.

Na výstup z lineárnej transformácie sa aplikuje nelineárna aktivačná funkcia, ako napríklad ReLU (Rectified Linear Unit) alebo GELU (Gaussian Error Linear Unit). Nelinearita je nevyhnutná na zachytenie komplexných vzťahov, ktoré by lineárne operácie nedokázali reprezentovať. V prípade ReLU je funkcia definovaná ako:

$$\text{ReLU}(\mathbf{z}) = \max(0, \mathbf{z}). \tag{4.8}$$

Nasleduje ďalšia lineárna transformácia, ktorá projektuje výsledok nelineárnej operácie späť do pôvodného priestoru. Táto operácia je definovaná rovnicou 4.9:

$$\mathbf{y} = \max(0, \mathbf{z})W_2 + \mathbf{b}_2, \tag{4.9}$$

kde W_2 a $\mathbf{b}_2$ predstavujú váhy a bias druhej vrstvy. Výsledný vektor $\mathbf{y}$ predstavuje konečnú reprezentáciu tokenu po spracovaní FFN [34].

Význam FFN spočíva v jej schopnosti zachytávať komplexné vzory, ktoré mechanizmus pozornosti nemusí dostatočne zdôrazniť. Umožňuje zvýrazniť syntaktické a sémantické črty tokenov, ktoré sú dôležité pre pochopenie kontextu. Kombinácia lineárnych transformácií s nelineárnymi funkciami pridáva modelu flexibilitu potrebnú na spracovanie zložitých štruktúr údajov [11].

4.3.5 Reziduálne prepojenia a normalizácia vrstiev

Reziduálne prepojenia a normalizácia vrstiev (Layer Normalization) zohrávajú kľúčovú úlohu pri zvyšovaní stability a efektivity učenia modelu. Tieto mechanizmy umožňujú modelu zachovať a efektívne spracovať dôležité informácie bez toho, aby sa strácali v procese učenia alebo spracovania dát [34, 19].

Reziduálne prepojenie pridáva k výstupu danej vrstvy jej pôvodný vstup. Táto skratka umožňuje modelu porovnať nové transformácie s pôvodnými hodnotami a rozhodnúť sa, či sú tieto transformácie užitočné. Formálne možno reziduálne spojenia vyjadriť rovnicou 4.10:

$$\mathbf{y} = \mathbf{x} + \text{Sublayer}(\mathbf{x}), \tag{4.10}$$

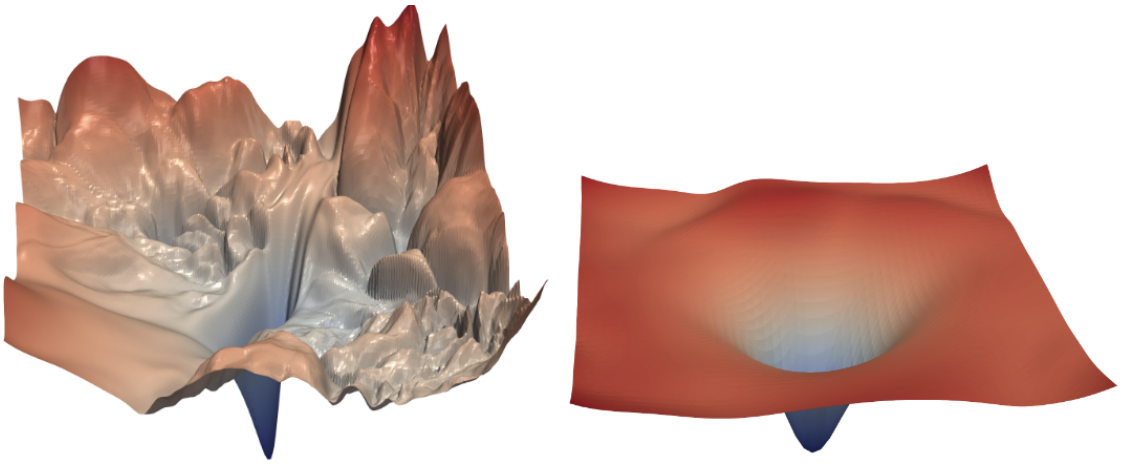
kde $\mathbf{x}$ je vstup do čiastkovej vrstvy (napríklad Viacvrstvová pozornosť alebo FFN) a Sublayer($\mathbf{x}$) je jej výstup. Výsledný vektor $\mathbf{y}$ obsahuje pôvodné aj transformované informácie.

Po pridaní reziduálneho prepojenia sa aplikuje normalizácia vrstiev (Layer Normalization), ktorá zaisťuje, že hodnoty komponentov vektora zostanú stabilné počas učenia. Táto stabilita sa dosahuje úpravou štatistík vektora, konkrétne jeho priemeru a smerodajnej odchýlky, ako je ukázané v rovnici 4.11 [2]:

$$\text{LayerNorm}(\mathbf{y}) = \gamma \cdot \frac{\mathbf{y} - \mu}{\sigma} + \beta, \tag{4.11}$$

kde μ a σ sú priemer a smerodajná odchýlka komponentov vektora $\mathbf{y}$, zatiaľ čo γ a β sú učené parametre, ktoré umožňujú modelu normalizáciu flexibilne prispôsobovať.

Reziduálne prepojenia majú zásadný význam pre stabilitu učenia, pretože zmierňujú problém miznúcich gradientov (vanishing gradients), ktorý je častý pri hlbokých sieťach. Umožňujú modelu „preskočiť“ neadekvátne transformácie a zachovať pôvodné informácie, ak sú transformácie menej užitočné. Normalizácia vrstiev zaisťuje, že hodnoty vo vektoroch sú udržiavané v konzistentnom rozsahu, čo podporuje stabilné a efektívne učenie, čo bolo experimentálne ukázané v práci [24]. Ľavá časť obrázku 4.5 zobrazuje stratovú funkciu pri trénovaní modelu bez použitia normalizácie a spojení, pričom pravá časť zobrazuje použitie normalizácie aj spojení pri trénovaní.



Obr. 4.5: Povrchy stratovej funkcie pre ResNet-56 so zavedenými reziduálnymi spojeniami a bez nich, znázorňujúce rozdiely v ostrosti a plynulosti pomocou metódy normalizácie filtrov.

4.3.6 Komponenty špecifické pre Dekodér

Dekodér obsahuje podobné komponenty ako kodér, ale jeho architektúra je rozšírená o dve špecifické komponenty, ktoré umožňujú dekóderu efektívne spracovávať a generovať sekvencie. Tieto moduly, konkrétne *vymaskovná viacvrstvová pozornosť* a *viacvrstvová pozornosť*, ktoré majú ako vstupné dáta výstup kodéra.

vymaskovná viacvrstvová pozornosť modifikuje klasický mechanizmus pozornosti tým, že maskuje budúce tokeny. Tento proces zabezpečuje, že dekóder pri generovaní sekvencie nemá prístup k tokenom, ktoré ešte neboli vygenerované. Matematicky sa tento prístup realizuje pridaním masky k matici podobností QK^T, kde pre všetky pozície $j > i$ je hodnota masky $-\infty$. Výstupom funkcie softmax sú potom pre vymaskované tokeny hodnoty rovné 0. Maskovanie je nevyhnutné pri autoregresívnych úlohách, ako je generovanie textu, kde každý nový token závisí iba od predchádzajúcich.

Mechanizmus medzi enkodérom a dekodérom (tzv. encoder-decoder attention) umožňuje dekodéru využiť informácie zo vstupnej sekvencie spracovanej enkodérom. Výstupy kodéra slúžia ako kľúče (K) a hodnoty (V), zatiaľ čo dotazy (Q) pochádzajú z dekodéra. Mechanizmus vypočíta váhy pomocou rovnice 4.4: kde výsledné váhy umožňujú dekodéru "zamerať sa" na relevantné časti vstupnej sekvencie. Táto časť je nevyhnutná pri úlohách, ako je preklad textu, kde je dôležité zachovať konzistenciu medzi vstupom a výstupom.

4.4 Analýza časovej zložitosti trénovania

Transformátorové neurónové siete napriek výborným výsledkom v rôznych aplikáciách majú jeden výrazný nedostatok a tým sú vysoké výpočtové nároky, najmä pri dlhých sekvenciách a zložitých modeloch, preto je potrebné ju zohľadniť aj pri návrhu detekčného systému. Pre detailnejšie pochopenie náročnosti tréningu sú definované nasledovné parametre, ktoré ovplyvňujú výpočtovú zložitosť modelu [40]:

- N: Dĺžka vstupnej sekvencie, čo je počet tokenov (napr. znakov) v každom vstupe, často doplnený na maximálnu dĺžku.
- d_{model}: Dimenzia modelu, ktorá určuje veľkosť embeddingov a skrytých stavov.
- L: Počet vrstiev modelu (blokov enkodéra).
- h: Počet hláv pozornosti v mechanizme viacvrstvovej pozornosti.
- d_k: Dimenzia každej vrstvy (hlavy) pozornosti, kde platí $d_{\text{model}} = h \times d_k$.
- d_{ff}: Veľkosť skrytej vrstvy doprednej neurónovej siete, zvyčajne 2–4-násobok d_{model}.
- B: Veľkosť dávky (batch size), teda počet vzoriek spracovaných súčasne počas tréningu.

Hlavné faktory ovplyvňujúce časovú zložitosť modelu sú **mechanizmus viacvrstvovej pozornosti** a **dopredná neurónová sieť**, ktoré sa opakujú v každej vrstve modelu. Celkové náklady na tréning zahŕňajú *dopredný priechod* (forward pass) a *spätný priechod* (backward pass), pričom druhý zvyčajne zdvojnásobuje výpočtové nároky.

4.4.1 Zložitosť mechanizmu pozornosti

Mechanizmus pozornosti umožňuje modelu zachytiť vzájomné vzťahy medzi všetkými tokenmi v sekvencii. Náročnosť tohto mechanizmu vyplýva z nasledujúcich krokov:

1. **Lineárne projekcie** – Každý vstupný token sa premietne do troch rôznych reprezentácií: Queries (**Q**), Keys (**K**) a Values (**V**). Tieto projekcie sa počítajú pre všetky tokeny a všetky hlavy pozornosti (h), pričom každá hlava pracuje s podmnožinou dimenzie d_{model}. Výpočtovú zložitosť týchto projekcií možno zapísať výrazom 4.12:

$$O(N \times d_{\text{model}} \times d_k). \tag{4.12}$$

2. **Zložitosť skalárneho súčinu** – Pozornosť medzi tokenmi sa počíta ako skalárny súčin medzi maticami požiadaviek Q ($N \times d_k$) a maticami kľúčov K ($d_k \times N$), čo má zložitosť:

$$O(N^2 \times d_k). \tag{4.13}$$

   Výsledné skóre pozornosti sa následne vynásobí maticou hodnôt V ($N \times d_k$), čo pridáva ďalšie výpočtové náklady $O(N^2 \times d_k)$. Celkové náklady na všetky h hlavy pozornosti demonštruje výraz 4.14:

$$O(h \times N^2 \times d_k) = O(N^2 \times d_{\text{model}}). \tag{4.14}$$

3. **Projekcia výstupu** – Po výpočte pozornosti pre všetky hlavy sú výstupy zreťazené a transformované späť na dimenziu d_{model}. Náklady na tento krok ukazuje výraz 4.15:

$$O(N \times d_{\text{model}}^2). \tag{4.15}$$

   Celkové náklady na samotný mechanizmus pozornosti v jednej vrstve sú teda:

$$O(N^2 \times d_{\text{model}}) + O(N \times d_{\text{model}}^2). \tag{4.16}$$

4.4.2 Zložitosť doprednej neurónovej siete

Pozičná dopredná sieť spracováva každý token nezávisle dvoma lineárnymi vrstvami oddelenými nelineárnou aktiváciou. Ak je skrytá dimenzia d_{ff}, náklady možno vyjadriť nasledovne:

$$O(N \times d_{\text{model}} \times d_{\text{ff}}) \tag{4.17}$$

pre prvú lineárnu vrstvu,

$$O(N \times d_{\text{ff}}) \tag{4.18}$$

pre nelineárnu aktiváciu a

$$O(N \times d_{\text{ff}} \times d_{\text{model}}) \tag{4.19}$$

pre druhú lineárnu vrstvu. Celková zložitosť pozičnej doprednej siete je:

$$O(N \times d_{\text{model}} \times d_{\text{ff}}). \tag{4.20}$$

4.4.3 Celková zložitosť na vrstvu enkodéra a trénovanie

Sčítaním zložitosti mechanizmu pozornosti (rovnica 4.16) a doprednej neurónovej siete je výsledná celková zložitosť jednej vrstvy zapísaná výrazom 4.14:

$$O(N^2 \times d_{\text{model}}) + O(N \times d_{\text{model}} \times d_{\text{ff}}). \tag{4.21}$$

Ak má model L vrstiev a veľkosť dávky je B, celkové náklady na dopredný priechod sú:

$$O(B \times L \times (N^2 \times d_{\text{model}} + N \times d_{\text{model}} \times d_{\text{ff}})). \tag{4.22}$$

Vrátane spätného priechodu sa celkové náklady na jeden tréningový krok zdvojnásobujú:

$$O(2 \times B \times L \times (N^2 \times d_{\text{model}} + N \times d_{\text{model}} \times d_{\text{ff}})). \tag{4.23}$$

Ako je možné vidieť z vyššie poskytnutých kalkulácií, tréning je výpočtovo náročný kvôli kvadratickej zložitosti, veľkým dimenziám vstupných vektorov a hĺbke architektúry.

Kapitola 5

Konceptuálny návrh riešenia

Nasledujúca kapitola predstavuje návrh a tvorbu transformátorových neurónových sietí určených na efektívnu detekciu škodlivých domén, konkrétne zameraných na domény spojené s malvérom, phishingom a domény generované algoritmami (DGA). V úvode kapitoly je analyzovaná rozsiahla dátová sada zhromaždená na účely detekcie škodlivých a legitímnych domén. Na základe tejto analýzy sa pri trénovaní modelov využívajú:

- doménové mená,
- informácie z RDAP/Whois,
- údaje z DNS záznamov,
- geolokačné informácie.

Pri modelovaní doménových mien sú následne experimentálne porovnané rôzne architektúry predtrénovaných transformátorových neurónových sietí (napr. DistilBERT, ELECTRA, MobileBERT a pod.) s vlastnou navrhnutou architektúrou. Na tejto špecificky navrhnutej architektúre sa testujú aj rôzne stratégie delenia textu na tokeny. Experimenty s doménovými menami slúžia na identifikáciu vhodných architektúr, ktoré sa následne používajú na trénovanie modelov na rozsiahlejších dátach zo záznamov DNS a údajoch z RDAP.

5.1 Prehľad a charakteristika dátových sád

Fundamentálnym aspektom úspešného vývoja výkonných modelov pre detekciu škodlivých doménových mien je existencia rozsiahlej a kvalitnej dátovej sady. Táto práca využíva dátové sady pre nasledujúce kategórie:

- malvér (domény spojené s distribúciou škodlivého softvéru),
- phishing (domény využivané pri phishingových útokoch),
- DGA (algoritmicky generované domény),
- benígne domény.

Dátové sady pre malvérové, phishingové a benígne domény boli zostavené v rámci výskumnej činnosti skupiny FETA a obsahujú rozsiahle informácie pre každú doménu, ako sú DNS záznamy, IP-súvisiace charakteristiky, WHOIS/RDAP dáta, informácie z TLS a

certifikátov a GeoIP dáta. Konkrétne, malvérová dátová sada zahŕňa 100 809 domén z rôznorodých zdrojov, medzi ktoré patria ThreatFox, The Firebog a platforma MISP threat intelligence. Phishingová dátová sada obsahuje 164 425 domén získaných zo služieb PhishTank a OpenPhish. Pre reprezentáciu neškodných domén bola zostavená sada s 368 956 doménami zo služby Cisco Umbrella a 461 338 doménami z reálnej prevádzky siete CESNET. Na zabezpečenie presnosti malvérových a phishingových dát bola ich škodlivosť overená cez VirusTotal, pričom nepotvrdené domény boli odobraté. Podobne, z benígnej dátovej sady boli vylúčené domény označené touto službou ako rizikové. Zber dát pre tieto tri kategórie prebiehal v období od marca 2023 do júla 2024, pričom definitívna verzia dátovej sady bola pripravená v auguste 2024.

Na analýzu domén generovaných algoritmami (DGA) bola použitá dátová sada z databázy DGArchive, spravovanej inštitúciou Fraunhofer FKIE, ktorá zhromažďuje doménové mená generované rôznymi rodinami algoritmov DGA, zachytené počas komunikácie malvéru. Zhrnutie charakteristík jednotlivých dátových sád, vrátane ich zdroja, časového obdobia zberu, veľkosti a počtu vzoriek, je uvedené v tabuľke 5.1.

Kategória	**Zdroj**	**Obdobie**	**Veľkosť dát**	**Počet vzoriek**
Benígne domény 1	CESNET traffic	mar 2023 - júl 2024	4,46 GB	461 388
Benígne domény 2	Cisco Umbrella	mar 2023 - júl 2024	4,23 GB	368 956
Phishingové domény	PhishTank a OpenPhish (filtrované cez VT [1])	mar 2023 - júl 2024	1,56 GB	164 425
Malvérové domény	ThreatFox, The Firebog, MISP (filtrované cez VT)	mar 2023 - júl 2024	755,84 MB	100 809
DGA	Fraunhofer FKIE		71,9 MB	230 070

Tabuľka 5.1: Prehľad dátových sád

5.1.1 Štruktúra dátových sád

Pre poskytnutie kontextu k následnému výberu atribútov pre modelovanie je detailný popis štruktúry použitých dátových sád kľúčový. Dátové sady pre malvérové, phishingové a benígne domény zdieľajú podobnú štruktúru, pričom každý záznam obsahuje nasledujúce polia:

- **domain_name** (`String`): Analyzované doménové meno.
- **url** (`String`): Zdrojová URL adresa, z ktorej bolo doménové meno získané.
- **evaluated_on** (`Date`): Dátum posledného pokusu o zber dát pre dané doménové meno.

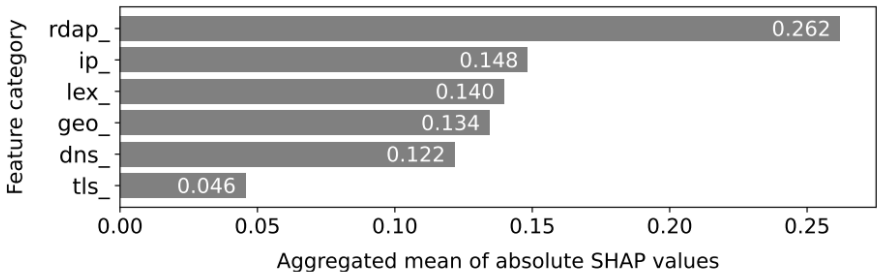
[1] VirusTotal

- **source** (`String`): Identifikátor zdroja, odkiaľ bolo doménové meno získané.
- **sourced_on** (`Date`): Dátum zaradenia doménového mena do dátovej sady.
- **dns** (`Object`): Dáta získané z DNS.
- **rdap** (`Object`): Dáta z RDAP alebo WHOIS záznamov.
- **tls** (`Object`): Dáta z TLS protokolu (nadviazanie spojenia).
- **ip_data** (`Array of Objects`): Pole objektov obsahujúcich IP adresy súvisiace s doménovým menom.

Je dôležité poznamenať, že prítomnosť dostupných dát (`dns`, `rdap`, `tls`, `ip_data`) sa môže v kolekciách líšiť v závislosti od dostupnosti informácií počas zberu dát. V prípade, že sa dané informácie nepodarilo získať (napríklad pri zaniknutých doménach), sú im priradené prázdne hodnoty (*null*).

5.1.2 Výber relevantných atribútov pre modelovanie

Po predstavení dátových sád v predchádzajúcej časti sa táto sekcia zameriava na výber vhodných vstupných atribútov pre trénovanie modelov. Transformátorové neurónové siete, ako bolo podrobne vysvetlené v kapitole 4, sú primárne navrhnuté na spracovanie textových sekvencií, preto je výber zameraný výlučne na textové atribúty, ktoré možno priamo tokenizovať. Tento prístup zároveň umožňuje využiť predtrénované jazykové modely trénované na rozsiahlych textových korpusoch, čím sa výrazne znižuje výpočtová náročnosť, keďže nie je potrebné optimalizovať celú množinu parametrov modelu, ale iba ich podmnožinu (proces známy ako *fine-tuning*).



Obr. 5.1: Agregované SHAP hodnoty pre jednotlivé kategórie vstupných atribútov (model LightGBM).

Na výber konkrétnych typov vstupných údajov bol ďalej zohľadnený aj experimentálny výskum realizovaný v rámci výskumnej skupiny FETA, zameraný na hodnotenie prínosu jednotlivých kategórií vlastností pri klasifikácii domén. Výsledky tejto analýzy, sumarizované na obrázku 5.1, poukazujú na výrazný prínos atribútov získaných z RDAP záznamov, IP informácií (vrátane geolokačných údajov, ktoré sú súčasťou štruktúry `ip_data`), lexikálnych vlastností doménových mien a DNS dát [17]. Naopak, vlastnosti odvodené z TLS komunikácie vykazovali výrazne nižšiu informačnú hodnotu, čo možno pripísať ich obmedzenej variabilite a zníženej diskriminačnej schopnosti. V súčasnosti totiž implementácia šifrovania prestáva byť spoľahlivým rozlišovacím znakom, keďže útočníci bežne využívajú

TLS certifikáty s cieľom zvýšiť dôveryhodnosť škodlivých domén, najmä pri phishingových kampaniach. Vzhľadom na tieto zistenia sa ďalší vývoj modelov sústredí predovšetkým na atribúty s najvyššou mierou prínosu, konkrétne na údaje z RDAP, DNS a IP vrstvy, vrátane textového spracovania doménového mena.

5.2 Architektúra systému detekcie

Na základe predchádzajúcej analýzy a výberu relevantných vstupných atribútov je možnosť systematicky rozdeliť vstupné údaje do štyroch hlavných skupín: doménové meno, informácie z RDAP záznamov, dáta zo záznamov DNS a atribúty súvisiace s IP vrstvou. Na základe týchto typov atribútov a kategórií domén predstavených v časti 5.1 bola zostavená prehľadová tabuľka 5.2, ktorej cieľom je prehľadne zachytiť, ktoré vstupné atribúty sú pri jednotlivých typoch domén k dispozícii na modelovanie. Prehľad dostupnosti atribútov tak zároveň tvorí základ pre návrh architektúry systému, ktorá zohľadňuje odlišnosti v dostupnosti dát pre jednotlivé kategórie domén.

Vstupný atribút	**Malígne domény**			**Benígne domény**
	Malvér	Phishing	DGA	
Doménové meno	✓	✓	✓	✓
RDAP	✓	✓	✗	✓
DNS	✓	✓	✗	✓
IP data	✓	✓	✗	✓

Tabuľka 5.2: Dostupnosť vstupných atribútov pre jednotlivé kategórie domén

Z tabuľky 5.2 vyplýva, že dostupnosť jednotlivých vstupných atribútov sa medzi kategóriami domén výrazne líši. Napríklad pre algoritmicky generované domény sú k dispozícii výhradne doménové mená, zatiaľ čo pri doménach spojených s malvérom alebo phishingom je množina dostupných dát podstatne širšia. Z uvedených rozdielov v dostupnosti dát vyplýva, že je výhodné navrhnúť súbor samostatných klasifikátorov, z ktorých každý je určený na spracovanie konkrétneho typu dát pre zodpovedajúcu kategóriu domén. Výsledné modely sú navrhnuté ako samostatne použiteľné komponenty s dôrazom na ich jednoducho realizovateľnú integráciu. Takto navrhnutá sústava modelov má nasledujúce výhody:

- **Zvýšená interpretovateľnosť modelov**.
- **Robustnosť klasifikácie**.
- **Možnosť paralelného spracovania**.

Interpretovateľnosť klasifikácie je podporená použitím samostatných modelov pre jednotlivé typy vstupných dát, ktoré umožňujú presne určiť, aký prínos má každý konkrétny dátový zdroj v procese detekcie. Tento prínos možno kvantifikovať napríklad pomocou rozdielu vybraných metrík. Nech F_A je výsledné skóre F1 modelu trénovaného výlučne nad dátami typu A, a F_{A+B} ako skóre F1 pri kombinácii dát typu A a B. Potom rozdiel $\Delta_B = F_{A+B} - F_A$ vyjadruje informačný prínos dátového vstupu B, ktorý možno interpretovať ako doplnkovú hodnotu v rámci klasifikačného rozhodovania. Takto získaná miera prínosu zvyšuje transparentnosť rozhodovacieho procesu. V rámci tejto práce predstavuje

typ *A* doménové meno, ku ktorému sú postupne pridávané ďalšie vrstvy ako RDAP, DNS alebo údaje z IP vrstvy.

Robustnosť klasifikácie vychádza zo skutočnosti, že v praxi pre danú doménu nemusia byť vždy dostupné všetky vrstvy vstupných dát. Niektoré domény nemusia obsahovať RDAP záznam, nemusia byť aktívne v DNS alebo nie je možné získať ich IP informácie. V takýchto prípadoch je stále možné vykonať klasifikáciu napríklad na základe samotného doménového mena alebo iných dostupných údajov. Vďaka samostatnému modelovaniu nad jednotlivými typmi dát tak systém zostáva funkčný aj pri neúplných vstupoch, čím sa zvyšuje jeho celková odolnosť voči nedostupným alebo chýbajúcim informáciám.

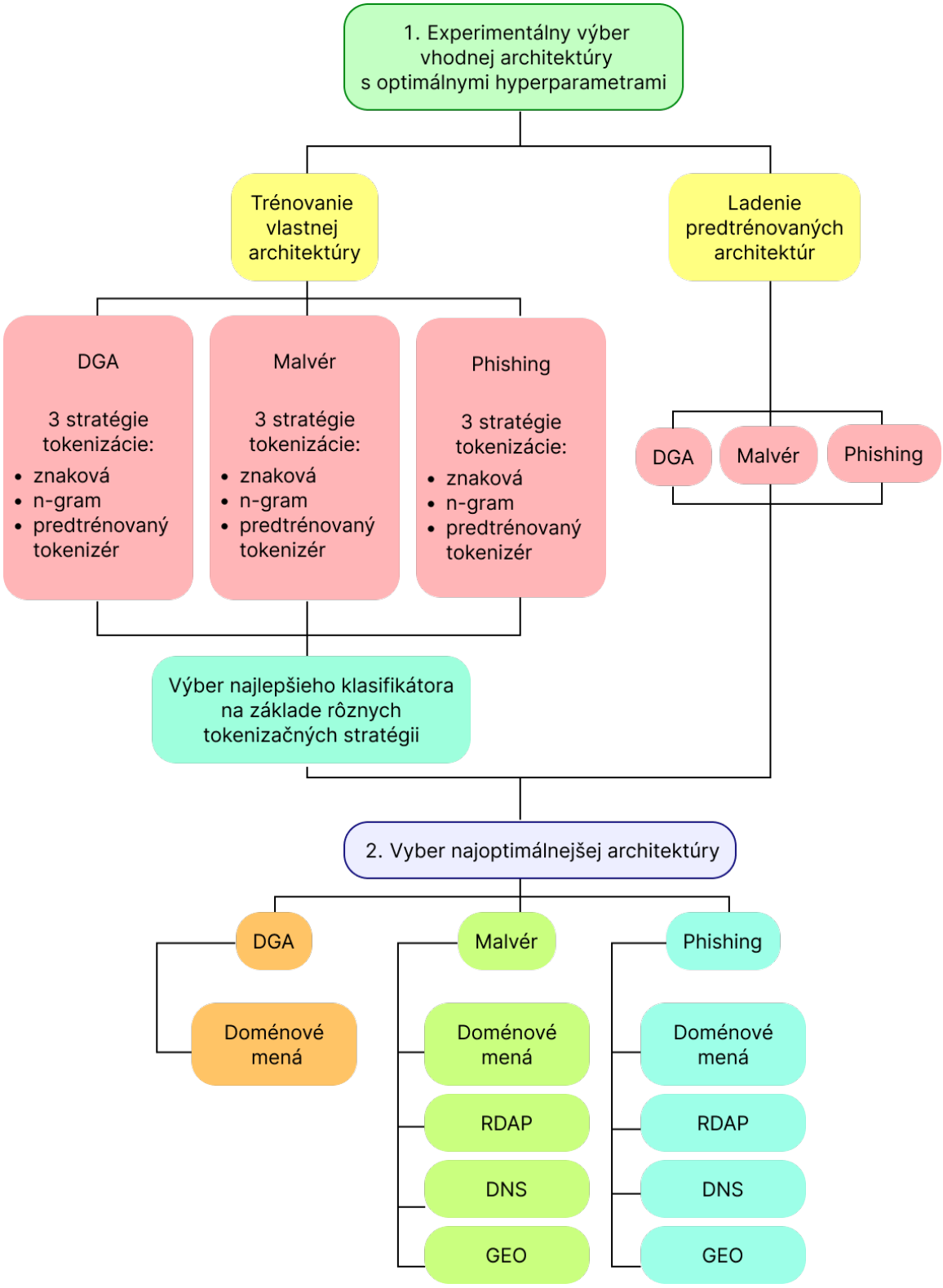
Paralelnosť spracovania predstavuje prirodzený dôsledok návrhu. Všetky klasifikátory môžu vyhodnocovať vstupné dáta súčasne, čím sa znižuje čas potrebný na určenie kategórie domény pri zachovaní presnosti rozhodovania. Táto vlastnosť je obzvlášť dôležitá v prostredí s vysokou mierou priepustnosti alebo v prípadoch, kde je požadovaná nízka latencia systému. Zároveň vzniká priestor pre jednoduchú distribúciu výpočtového zaťaženia, čo zvyšuje škálovateľnosť riešenia a uľahčuje jeho nasadenie v rôznych prevádzkových scenároch.

Obrázok 5.2 znázorňuje experimentálny rámec, ktorý sa v práci použije na systematický výber a vyhodnotenie transformerových architektúr pre úlohu detekcie malígnych domén. Postup bude prebiehať v dvoch hlavných fázach.

V **prvej fáze** bude cieľom nájsť optimálne architektúry pre jednotlivé kategórie domén. Na začiatku budú vybrané tri reprezentatívne architektúry rôznych veľkostí – malá, stredná a veľká – ktoré budú slúžiť ako zástupcovia širšej triedy modelov podľa ich výpočtovej náročnosti. Pre každú z týchto architektúr sa uskutoční ladenie hyperparametrov s cieľom zabezpečiť ich optimálnu konfiguráciu. Tieto optimalizované nastavenia budú následne použité aj pri ďalších architektúrach s podobnou veľkosťou.

Na takto pripravených architektúrach bude následne prebiehať porovnávanie vlastnej architektúry, trénovanej s rôznymi tokenizačnými stratégiami (znaková, n-gramová, predtrénovaný tokenizér), s predtrénovanými transformerovými modelmi. Tento experiment sa uskutoční nezávisle pre tri klasifikačné úlohy: detekciu DGA domén, malvéru a phishingu. Výstupom tejto fázy bude výber najvhodnejšej architektúry pre každú z týchto úloh.

V **druhej fáze** sa vybrané architektúry pre jednotlivé klasifikačné problémy podrobia experimentovaniu s rôznymi vstupnými dátovými kombináciami. Každý model bude trénovaný na rozličných súboroch dát, pričom základom každého vstupu bude vždy doménové meno. K nemu sa budú následne pridávať ďalšie informácie – konkrétne údaje z RDAP, DNS a geolokačné atribúty. Cieľom tejto fázy bude overiť, aký vplyv má doplnenie jednotlivých typov dát na výkonnosť modelu, a identifikovať najinformatívnejšie kombinácie pre spoľahlivú detekciu.



Obr. 5.2: Schéma plánovaného experimentálneho postupu výberu architektúr a testovania ich výkonnosti pri rôznych typoch vstupných dát.

Nasledujúce časti sa podrobne venujú jednotlivým klasifikátorom, pričom pre každú kombináciu kategórie domén a typu vstupných dát je predstavený spôsob spracovania, výber reprezentácie a štruktúra modelu.

5.3 Klasifikácia nad doménovými menami

Doménové meno predstavuje kľúčový atribút pre detekciu škodlivých domén, keďže je takmer vždy prítomné v sieťovej komunikácii. Zároveň nesie vysokú informačnú hodnotu, keďže phishingové domény, ako bolo uvedené v kapitole 3, často imitujú známe značky a vzory, zatiaľ čo DGA domény sa vyznačujú neprirodzenou štruktúrou, zvýšenou entropiou a výskytom netypických znakov. Tieto vlastnosti umožňujú efektívne rozlišovať medzi benígnymi a škodlivými doménami už na základe samotného doménového mena.

Tréning transformerových modelov má kvadratickú výpočtovú zložitosť vzhľadom na dĺžku vstupnej sekvencie (ako bolo popísané v kapitole 4), preto sú doménové mená so svojou kompaktnou tokenovou reprezentáciou vhodným základom pre návrh a ladenie modelov. V rámci týchto experimentov budú nad doménovými menami systematicky porovnávané rôzne architektúry a optimalizované hyperparametre s cieľom dosiahnuť čo najvyššiu presnosť detekcie pre jednotlivé kategórie domén (malvér, phishing, DGA). Výsledkom bude architektúra s najvhodnejšou konfiguráciou pre každý typ domény, ktorá bude následne použitá na trénovanie modelov s rozsiahlejšími vstupmi, ako sú údaje z RDAP, DNS a IP vrstvy.

Pri výbere architektúr určených na klasifikáciu doménových mien bola zohľadnená predovšetkým schopnosť modelov zachytiť štrukturálne a lexikálne vlastnosti krátkych textových sekvencií a zároveň zabezpečiť efektívny priebeh trénovania. Výber preto zahŕňa ľahkú vlastnú architektúru navrhnutú špecificky pre túto úlohu, ako aj reprezentatívnu skupinu predtrénovaných jazykových modelov pokrývajúcich rôzne úrovne zložitosti a trénovacích stratégií. Cieľom je vytvoriť diverzifikovaný súbor transformerových modelov, na základe ktorého bude možné experimentálne posúdiť vhodnosť jednotlivých architektúr pre klasifikáciu rôznych typov malígnych domén.Vybrané architektúry možno rozdeliť do dvoch kategórií: **vlastný návrh architektúry** a **predtrénované modely**.

5.3.1 Vlastný návrh architektúry modelu

Architektúra vlastného modelu pozostáva z nasledujúcich komponentov, ktorých hodnoty boli odôvodnené na základe analýzy dát a odporúčaní v literatúre:

- **Počet kodérových blokov: 2**
  Pre sekvencie s dĺžkou do 50 tokenov postačujú dve vrstvy kodérov, čo potvrdzujú aj experimenty s modelom TinyBERT, kde varianty s 4 vrstvami dosahujú len marginálne zlepšenie oproti 2–3 vrstvám bez výrazného nárastu trénovacieho času [40, 18].

- **Rozmer vstupných vektorových reprezentácií: 128**
  Zníženie dimenzie vstupných vektorov zo štandardných 768 (ako napríklad v modeli BERT-base) na 128 výrazne znižuje pamäťové nároky a výpočtovú záťaž, pričom pre úzko špecializované vstupy, akými sú tokenizované doménové mená, si model zároveň zachováva dostatočnú informačnú kapacitu [22].

- **Počet hláv mechanizmu pozornosti: 4**
  Mechanizmus viacnásobnej pozornosti (multi-head attention) rozdeľuje vnútorné vek-

torové reprezentácie na menšie časti – tzv. hlavy – ktoré paralelne spracúvajú rôzne vzťahy medzi prvkami v sekvencii. Pri zvolenej veľkosti modelu (`d_model` = 128) je rozdelenie do štyroch hláv (každá s rozmerom 32) výhodné, pretože umožňuje súbežné sledovanie viacerých jazykových vzorov bez toho, aby sa nadmerne znižovala vyjadrovacia schopnosť jednotlivých hláv [40, 36].

- **Veľkosť vnútornej (nelineárnej) vrstvy: 256**
  Vnútorná časť každého transformačného bloku pozostáva z dvojice plne prepojených vrstiev, pričom jej šírka bola nastavená na hodnotu 256, teda dvojnásobok veľkosti vstupných vektorov (`d_model` = 128). Tento pomer sa osvedčil ako dostatočný na zachytenie nelineárnych závislostí bez zbytočného zvyšovania počtu parametrov modelu [40, 6].

- **Pravdepodobnosť vypnutia neurónov (Dropout): 0,1**
  Hodnota 0,1 predstavuje osvedčené nastavenie pre menšie modely, ktoré pomáha predchádzať preučeniu bez negatívneho vplyvu na rýchlosť a stabilitu učenia [40].

- **Pooling: globálne priemerovanie**
  Použitie priemerovania naprieč celou sekvenciou nahrádza potrebu špeciálneho klasifikačného tokenu a poskytuje spoľahlivý spôsob zhrnutia informácie pri krátkych vstupoch [23, 20].

Táto konfigurácia vytvára ľahkú a efektívnu architektúru, ktorá plne zachováva kľúčové mechanizmy transformera a zároveň slúži ako spoľahlivý referenčný bod pri porovnávaní s komplexnejšími predtrénovanými modelmi.

5.3.2 Výber predtrénovaných modelov

Predtrénované transformerové modely využívajú jazykové znalosti získané z rozsiahlych textových korpusov, vďaka čomu dokážu efektívne generalizovať aj na nové úlohy s obmedzeným množstvom trénovacích dát. Pri klasifikácii doménových mien, ktoré sú krátke, štruktúrované a často neštandardné z hľadiska prirodzeného jazyka, môžu tieto modely identifikovať opakujúce sa vzory, lexikálne odchýlky a znaky imitácie legitímnych názvov. Cieľom je získať reprezentatívny prehľad o výkonnosti kompaktných, odľahčených, ale aj štandardne veľkých modelov určených na klasifikáciu doménových mien. Konkrétne boli do experimentov zaradené nasledujúce predtrénované modely, ktoré reprezentujú rôzne prístupy z hľadiska zložitosti architektúry, veľkosti modelu a spôsobu predtrénovania:

- **BERT-tiny, BERT-mini, BERT-small, BERT-medium**
  Zjednodušené varianty pôvodného modelu BERT, ktoré sa líšia počtom vrstiev a mechanizmov pozornosti. Umožňujú analyzovať, ako sa zmeny v zložitosti architektúry prejavujú na presnosti klasifikácie.

- **DistilBERT (verzia so zachovaním veľkých písmen aj bez nej)**
  Odľahčený model odvodený od štandardného modelu BERT, ktorý má znížený počet parametrov a vrstiev. Napriek tomu si zachováva väčšinu výkonnosti pôvodného modelu. Používa sa ako východiskový bod na porovnávanie.

- **ALBERT-base**
  Model, ktorý znižuje počet parametrov pomocou zdieľania váh medzi vrstvami a zmenšením rozmeru vstupných reprezentácií. Je navrhnutý s dôrazom na úsporu pamäte a výpočtovej náročnosti pri zachovaní výkonnosti.

- **ELECTRA-small**
  Model využíva odlišný prístup k trénovaniu než tradičné transformerové modely. Namiesto predpovedania zakrytých slov sa učí rozpoznávať, ktoré časti vstupu boli umelo pozmenené. Tento diskriminačný spôsob učenia sa v literatúre osvedčil ako efektívnejší a dosahuje porovnateľnú presnosť pri nižšej výpočtovej náročnosti a kratšom čase trénovania.

- **MobileBERT**
  Model navrhnutý pre nasadenie v prostredí s obmedzenými výpočtovými zdrojmi. Napriek hlbšej štruktúre je optimalizovaný tak, aby mal nízke nároky na výpočtový čas aj pamäť, pričom si zachováva dobrú úroveň výkonnosti.

Takto zostavená skupina modelov umožňuje posúdiť výkonnosť prístupov, ktoré sa líšia nielen architektonickou veľkosťou, ale aj metodikou predtrénovania. Ich porovnanie v jednotných podmienkach trénovania poskytne dôležitý základ pre výber najvhodnejšej architektúry na detekciu škodlivých domén. Model, ktorý dosiahne najvyššiu presnosť v klasifikácii pre danú kategóriu domén (malvér, phishing, DGA), bude následne použitý aj na trénovanie nad rozšírenými vstupmi (RDAP, DNS, IP).

5.3.3 Tokenizačné stratégie pre spracovanie doménových mien

Tokenizácia predstavuje kľúčový krok pri spracovaní textových dát v rámci transformerových modelov, keďže priamo ovplyvňuje výslednú dĺžku vstupnej sekvencie a spôsob, akým model vníma štruktúru vstupu. Vzhľadom na špecifický charakter doménových mien je vhodné preskúmať rôzne stratégie ich rozkladu na tokeny. V rámci experimentov boli vybrané nasledujúce stratégie:

- **Tokenizácia na úrovni znakov (character-level)**
  Každý znak doménového mena sa považuje za samostatný token. Tento prístup zachytáva všetky detaily vrátane znakov ako pomlčky, číslice alebo bodky.
  Príklad: `secure-login123.com` → `['s', 'e', 'c', 'u', 'r', 'e', '-', 'l', 'o', 'g', 'i', 'n', '1', '2', '3', '.', 'c', 'o', 'm']`.

- **Tokenizácia pomocou n-gramov**
  Doménové meno je rozdelené na prekrývajúce sa sekvencie znakov s pevnou dĺžkou (napr. trigramy). Tento spôsob je vhodný na zachytávanie lokálnych štruktúr a opakujúcich sa vzorov.
  Príklad (3-gramy): `secure-login123.com` → `['sec', 'ecu', 'cur', 'ure', 're-', 'e-l', '-lo', 'log', 'ogi', 'gin', 'in1', 'n12', '123', '23.', '3.c', '.co', 'com']`.

- **Tokenizácia predtrénovaným nástrojom modelu DistilBERT**
  Tokenizácia je vykonaná pomocou vstavaného nástroja modelu DistilBERT, ktorý využíva delenie na podslová. Tento prístup slúži ako referenčný bod pre porovnanie s vlastnými metódami.
  Príklad: `secure-login123.com` → `['secure','-','login','123','.','com']`.

Cieľom týchto experimentov je určiť, ktorá z testovaných stratégií vedie k informatívnejšej tokenovej reprezentácii a zároveň zachováva výpočtovú efektivitu modelu pri klasifikácii

doménových mien. V neposlednom rade je dôležité zdôrazniť, že experimentovanie s alternatívnymi tokenizačnými prístupmi bolo v tejto práci realizované výlučne pri použití vlastnej architektúry. Predtrénované jazykové modely sú totiž navrhnuté tak, aby pracovali s konkrétnym typom tokenizácie, na ktorom boli pôvodne trénované. Zmena tokenizačného postupu by narušila konzistenciu medzi vstupnými údajmi a naučenými váhami modelu, čo by viedlo k neplatným výsledkom.

5.3.4 Určenie maximálnej dĺžky vstupnej sekvencie

Maximálna dĺžka vstupnej sekvencie (`max_length`) je dôležitým hyperparametrom transformerových modelov. Pri prekročení tejto hodnoty sú vstupy skracované, čo môže viesť k strate informácií, avšak zvolením príliš veľkej hodnoty dochádza k neefektívnemu využívaniu pamäte a výpočtových zdrojov. Preto je potrebné určiť takú dĺžku, ktorá dostatočne pokrýva typické prípady bez zbytočného predlžovania sekvencie. Spracovanie doménových mien bolo realizované pomocou tokenizéra modelu DistilBERT, keďže tento prístup zabezpečuje jednotnú reprezentáciu vstupov pre vlastnú architektúru aj pre predtrénované modely. Na určenie vhodnej hodnoty `max_length` bola vykonaná analýza distribúcie dĺžok tokenizovaných doménových mien naprieč dátovými sadami pre malvér, phishing a domény DGA. Výsledné štatistiky zobrazuje tabuľka 5.3 a distribúcia rozloženia dĺžok tokenizovaných sekvencií je ilustrovaná grafom 5.3:

Ukazovateľ	**DGA**	**Phishing**	**Malvér**
Priemerná dĺžka (mean)	12,49	13,13	10,94
Medián (median)	11,00	11,00	9,00
Percentilové kvantily			
75. percentil	15	15	12
90. percentil	22	20	17
95. percentil	25	28	22
99. percentil	30	51	29
Maximálna hodnota	68	68	72

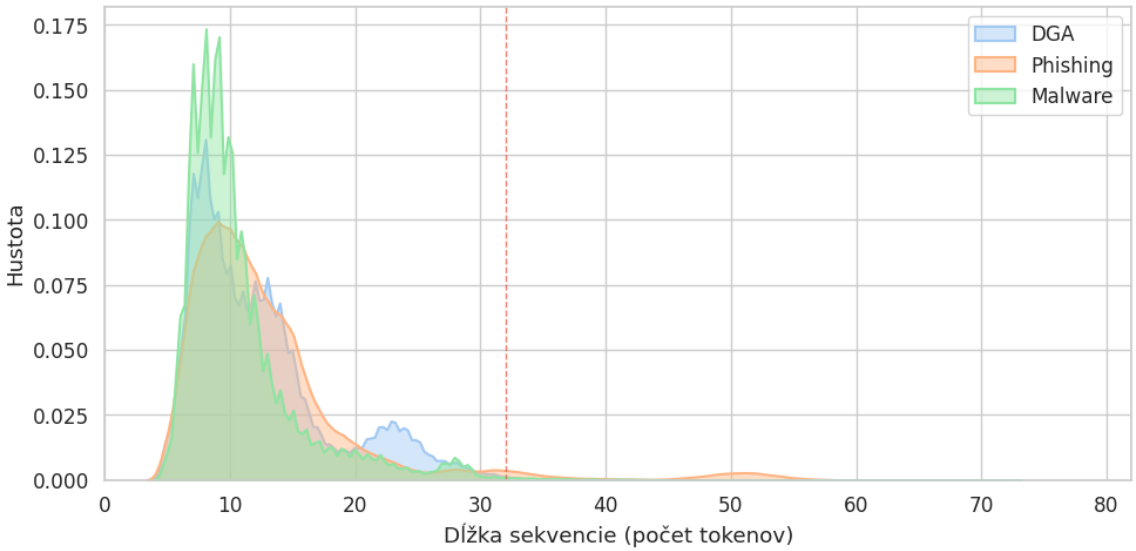
Tabuľka 5.3: Štatistické ukazovatele dĺžok tokenizovaných doménových mien

Na základe analýzy bola maximálna dĺžka vstupnej sekvencie pre doménové mená nastavená na 32 tokenov, ktoré pokrýva viac ako 95% vzoriek bez potreby skracovania. Táto hodnota predstavuje vhodný kompromis medzi pokrytím a výpočtovou efektivitou vo všetkých ďalších experimentoch s doménovými menami.

5.3.5 Optimalizácia hyperparametrov

Cieľom ladenia hyperparametrov je zabezpečiť, aby každá architektúra modelu dosiahla svoj maximálny možný výkon v rámci danej úlohy. Nevhodné nastavenie hyperparametrov môže viesť k nestabilnému trénovaniu, pomalej konvergencii alebo preučeniu modelu. Preto namiesto ručného výberu bol zvolený systematický proces ladenia, ktorý zvyšuje pravdepodobnosť nájdenia vhodných nastavení.

Na ladenie bol použitý nástroj `Ray Tune`, ktorý umožňuje efektívne vyhľadávanie v priestore hyperparametrov pomocou náhodného vzorkovania z vopred definovaných rozdelení.



Obr. 5.3: Pravdepodobnostné rozdelenie dĺžok tokenov v jednotlivých dátových množinách

Tento prístup je výrazne úspornejší v porovnaní s tradičným prístupom, ako je napr. `grid search`, pretože:

- netestuje všetky kombinácie parametrov, ale len ich informatívnu podmnožinu,
- umožňuje paralelné spúšťanie viacerých experimentov,
- a je vhodný aj pre prípady, kde nie je možné vopred odhadnúť optimálne nastavenia.

Z dôvodu neúmernej výpočtovej náročnosti nebolo ladenie hyperparametrov realizované pre všetky architektúry, ale len pre trojicu modelov reprezentujúcich rôzne úrovne architektonickej zložitosti.

- **BERT-tiny** – Zástupca extrémne kompaktných modelov, vhodný pre nasadenie v prostrediach s minimálnymi výpočtovými nárokmi. Jeho ladenie umožňuje zhodnotiť, aký zisk môže priniesť presné doladenie aj pri veľmi malom modeli.
- **BERT-small** – Stredne veľký model, ktorý ponúka kompromis medzi výkonom a efektivitou. Predstavuje typickú voľbu v bežných nasadeniach.
- **DistilBERT-base (uncased)** – Najväčší z vybraných modelov, ktorý si napriek vyššiemu počtu parametrov zachováva rozumnú výpočtovú náročnosť a poskytuje referenčný výkon pre ostatné architektúry.

Týmto výberom bolo možné pokryť široké spektrum architektúr a získať predstavu o význame ladenia naprieč rôznymi modelovými kapacitami. Prehľad optimalizovaných hyperparametrov, vrátane spôsobu výberu ich hodnôt (z definovaných množín alebo pravdepodobnostných rozdelení), a ich významu je uvedený v tabuľke 5.4.

Hyperparameter	Výberové rozmedzie	Význam
Rýchlosť učenia	logaritmicky rovnomerné $\langle 1 \times 10^{-6}, 5 \times 10^{-5} \rangle$	Ovplyvňuje veľkosť aktualizácií váh modelu počas učenia. Nevhodná hodnota môže spôsobiť pomalú alebo nestabilnú konvergenciu.
Veľkosť trénovacej dávky (batch size)	128, 256, 512	Určuje počet vzoriek spracovaných v jednom kroku. Väčšia dávka znižuje variabilitu gradientu a vedie k stabilnejšiemu trénovaniu.
Váhová penalizácia (weight decay)	rovnomerné $\langle 0.0, 0.1 \rangle$	Slúži na regularizáciu modelu, obmedzuje rast váh a znižuje riziko preučenia.
Pomer zahrievania (Warmup ratio)	rovnomerné $\langle 0.0, 0.2 \rangle$	Definuje podiel úvodných krokov trénovania, počas ktorých sa rýchlosť učenia lineárne zvyšuje na cieľovú hodnotu.
Typ plánovača	lineárny, kosínusový	Určuje priebeh zmeny rýchlosti učenia počas trénovania. Ovplyvňuje celkovú stabilitu a rýchlosť konvergencie modelu.

Tabuľka 5.4: Prehľad ladených hyperparametrov a ich význam

5.4 Klasifikácia na základe údajov RDAP

RDAP (Registration Data Access Protocol) je nástupcom protokolu WHOIS, ktorý je navrhnutý na poskytovanie údajov o doménach v štandardizovanom formáte. Umožňuje efektívne získavanie informácií o registrátoroch, dátumoch registrácie, stave domén a kontaktných entitách, čo vytvára vhodný základ pre automatizovanú analýzu. Tieto údaje môžu niesť indikátory podozrivého správania, ako je absencia identifikovateľného registranta, neštandardné hodnoty alebo opakujúce sa technické entity. V rámci tejto sekcie sa analyzujú RDAP údaje pre phishingové a malvérové domény v porovnaní s legitímnymi doménami. Pre lepšiu predstavu tabuľka 5.5 opisuje štruktúru typického RDAP objektu.

Položka RDAP	Dátový typ	Popis
copyright_notice	Reťazec (String)	Oznámenie o autorských právach k údajom RDAP/WHOIS.
dnssec	Boolovská hodnota	Príznak prítomnosti DNSSEC.
entities	Objekt	Obsahuje polia pre rôzne typy entít (napr. abuse, admin, registrant) ako zoznam objektov s detailmi o jednotlivých entitách.
expiration_date	Dátum	Dátum vypršania platnosti domény.
handle	Reťazec (String)	Unikátny identifikátor RDAP záznamu.
last_changed_date	Dátum	Dátum poslednej zmeny údajov o doméne.
name	Reťazec (String)	Názov cieľovej domény, ku ktorej sa záznam vzťahuje.
nameservers	Pole reťazcov	DNS názvy serverov priradených k doméne, podľa RDAP alebo WHOIS.
registration_date	Dátum	Dátum prvej registrácie domény.
status	Pole reťazcov	Stav objektu domény podľa špecifikácie (napr. `active`, `clientHold`).
terms_of_service_url	Reťazec (String)	URL adresa podmienok použitia údajov RDAP.
url	Reťazec (String)	URL zdrojového RDAP záznamu.
whois_server	Reťazec (String)	Adresa WHOIS servera.

Tabuľka 5.5: Prehľad vybraných polí z RDAP objektu a ich popis

Nasledujúci výpis RDAP záznamu phishingovej domény zo spracovanej dátovej sady slúži ako ilustrácia reálnej štruktúry údajov. Na rozdiel od tabuľkovej sumarizácie 5.5, ktorá poskytuje len prehľad názvov a významov jednotlivých polí, tento príklad poukazuje na praktickú zložitosť výstupu – vrátane viacerých úrovní vnorenia, variabilnej dĺžky záznamov a prítomnosti viacerých entít v rámci jedného poľa. Tieto aspekty predstavujú dôležité výzvy pri spracovaní dát a zohrávajú zásadnú úlohu pri návrhu robustného klasifikačného prístupu.

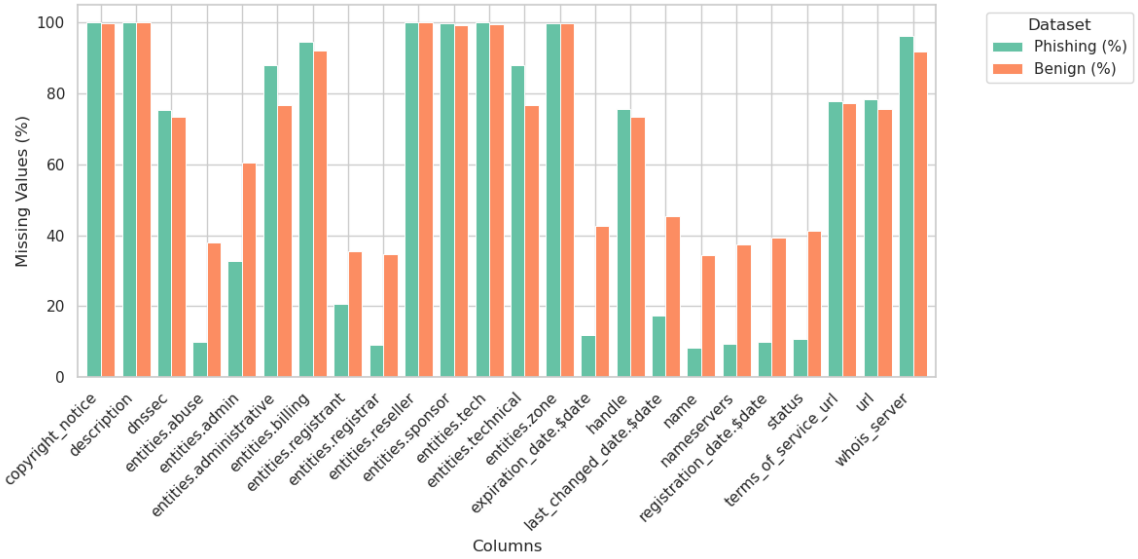
```
rdap{
  'handle': "1BD458F4835D436EB9270B066A9B0B17-LROR",
  'name': "espace-maladie.org",
  'whois_server': "",
  'terms_of_service_url': "https://thenew.org/org-people/about-pir/policies/",
  'copyright_notice': "",
  'description': [],
  'last_changed_date': "2024-04-30T13:58:05.313Z",
  'registration_date': "2024-04-25T13:57:07.929Z",
  'expiration_date': "2025-04-25T13:57:07.929Z",
  'url': "https://rdap.publicinterestregistry.org/rdap/domain/espace-maladie.org",
  'entities': {
    'registrant': [{ "type": "entity" }],
    'technical': [{ "type": "entity" }],
    'administrative': [{ "type": "entity" }],
    'registrar': [{
      'handle': "2",
      'url': "https://rdap.publicinterestregistry.org/rdap/entity/2",
      'type': "entity",
      'name': "Network Solutions, LLC"
    }],
    'abuse': [{
      'handle': "72265CEC0F8444D386CE827213EDBAB4-DONUTS",
      'type': "entity",
      'email': "domain.operations@web.com"
    }]
  },
  'nameservers': [
    "ns1.bluehost.com",
    "ns2.bluehost.com"
  ],
  'status': [
    "client transfer prohibited"
  ],
  'dnssec': false
}
```

Obr. 5.4: Ukážka reálneho RDAP záznamu phishingovej domény

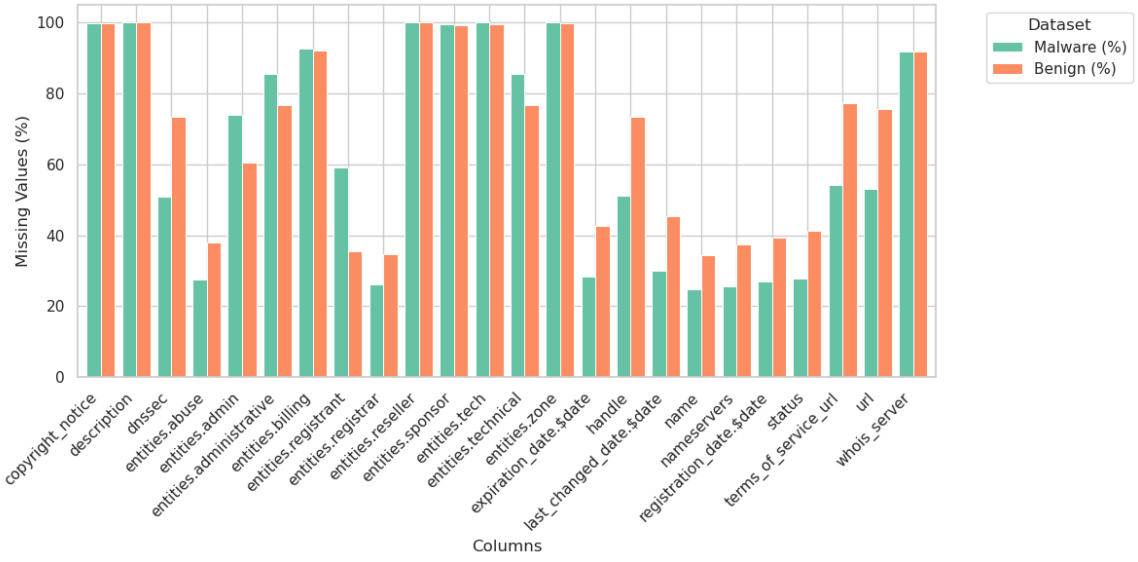
S ohľadom na túto štrukturálnu rozmanitosť je nevyhnutné vykonať podrobnú analýzu dostupných atribútov. Nasledujúca časť sa preto zameriava na vyhodnotenie chýbajúcich údajov, elimináciu málo informatívnych polí a výber takých textových prvkov, ktoré môžu najviac prispieť k efektívnej detekcii škodlivých domén.

5.4.1 Analýza a výber atribútov z RDAP záznamov

RDAP dáta sú pôvodne uložené vo formáte JSON, pričom každá doména je reprezentovaná ako záznam s rôzne štruktúrovanými poľami. Pre účely analýzy sa tieto záznamy transformujú do tabuľkovej reprezentácie, ktorá umožňuje systematicky posudzovať prítomnosť a kvalitu jednotlivých atribútov. Po konverzii sa automaticky identifikujú a odstraňujú stĺpce, ktoré sú úplne prázdne vo všetkých záznamoch, ako aj záznamy bez akejkoľvek využiteľnej hodnoty. Na nasledujúcich grafoch 5.5 a 5.6 je znázornené percento chýbajúcich hodnôt pre jednotlivé atribúty, osobitne pre phishingové a malvérové domény, vždy v porovnaní s benígnymi záznamami.



Obr. 5.5: Percentuálne zastúpenie chýbajúcich hodnôt jednotlivých atribútov – medzi phishingovými a legitímnymi doménami.



Obr. 5.6: Percentuálne zastúpenie chýbajúcich hodnôt pre každý atribút – porovnanie medzi malvérovými a legitímnymi doménami.

Z grafov 5.5 a 5.6 vyplýva, že viaceré atribúty vykazujú vysoký podiel chýbajúcich hodnôt naprieč všetkými kategóriami domén, čo poukazuje na ich obmedzenú dostupnosť v RDAP záznamoch bez ohľadu na typ domény. Zároveň však možno identifikovať aj také atribúty, kde je miera chýbania významne odlišná medzi škodlivými (phishingovými či malvérovými) a legitímnymi doménami. Tieto rozdiely naznačujú, že neprítomnosť údajov môže niesť informačnú hodnotu a môže byť využitá ako diskriminačný znak v klasifikácii. Na systematické overenie tohto predpokladu je použitá štatistická analýza rozdielov chýbajúcich hodnôt, ktorá je rozdelená do dvoch častí:

1. Cohenov koeficient h
2. Chí-kvadrátový test nezávislosti

Cohenov koeficient h predstavuje mieru efektu, ktorá slúži na kvantifikáciu rozdielu medzi dvoma podielmi. V kontexte tejto analýzy umožňuje hodnotiť, do akej miery sa líši výskyt chýbajúcich hodnôt v jednotlivých atribútoch medzi škodlivými a legitímnymi doménami. Jeho výhodou je, že nezohľadňuje len absolútny rozdiel medzi percentami, ale upravuje ho o zakrivenie v rozsahu pravdepodobnosti, čím poskytuje stabilnejšie porovnanie aj pri extrémnych hodnotách. Formálne je definovaný nasledovne:

$$h = 2 \cdot (\arcsin \sqrt{p_1} - \arcsin \sqrt{p_2}),$$

kde p_1 a p_2 predstavujú podiel výskytu (napr. chýbajúcich hodnôt) v dvoch porovnávaných skupinách. Výsledný koeficient h vyjadruje veľkosť efektu – čím vyššia je jeho hodnota, tým väčší rozdiel medzi skupinami. Interpretácia výsledkov vychádza zo štandardného rámca efektových veľkostí podľa Cohena:

- $h < 0{,}2$: zanedbateľný rozdiel,
- $0{,}2 \leq h < 0{,}5$: malý efekt,
- $0{,}5 \leq h < 0{,}8$: stredný efekt,
- $h \geq 0{,}8$: veľký efekt.

Cohenov koeficient sa neviaže na veľkosť vzorky a neposkytuje štatistickú významnosť v zmysle testovania hypotéz, slúži však ako praktický nástroj na identifikáciu atribútov s výrazne odlišným správaním medzi triedami. V tejto práci sa preto využíva ako jedno z rozhodovacích kritérií pri výbere vstupných atribútov, ktoré majú byť ponechané alebo vyradené. Aplikovaná analýza ukázala, že v prípade phishingových domén sa najvýraznejšie rozdiely vyskytujú pri poliach ako stav domény, dátumy registrácie a expirácie, menné servery či kontaktné údaje pre zneužitie. Pri malvérových doménach boli najodlišnejšie atribúty ako registrant, URL adresa, podpora DNSSEC alebo identifikátor domény. Tieto rozdiely môžu reflektovať špecifické technické vlastnosti malígnych domén alebo zámerné obmedzenie dostupnosti registračných údajov. Naopak, pri atribútoch, kde sa miera chýbania medzi triedami výrazne nelíši, možno konštatovať, že prítomnosť alebo absencia hodnoty pravdepodobne nepredstavuje diskriminačný znak.

Chí-kvadrátový test nezávislosti sa používa na overenie, či medzi dvoma kategóriami premenných existuje štatisticky významná súvislosť. V tejto analýze slúži na zisťovanie, či výskyt chýbajúcich hodnôt v danom atribúte závisí od typu domény — teda či sú chýbajúce údaje rozdelené rovnomerne medzi škodlivé a legitímne domény, alebo je medzi nimi systematický rozdiel.

Pre každý atribút sa vytvára kontingenčná tabuľka s dvoma premennými:

- výskyt alebo absencia hodnoty v danom poli (chýba / nechýba),
- typ domény (škodlivá / legitímna).

Na základe tejto tabuľky sa vypočíta chí-kvadrátová štatistika:

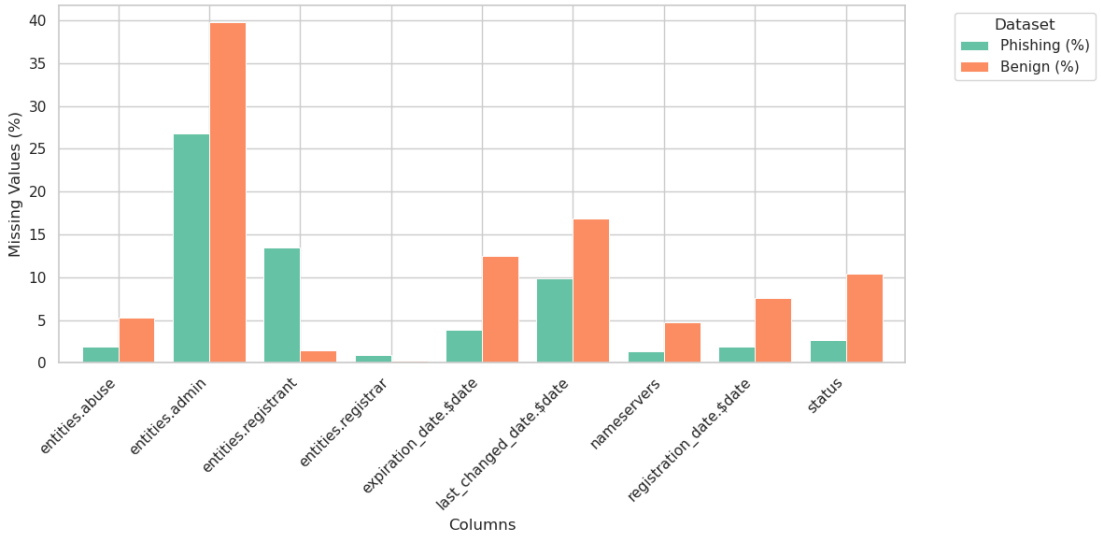
$$\chi^2 = \sum \frac{(O\ - E)^2}{E},$$

kde O je pozorovaná hodnota a E je očakávaná hodnota za predpokladu nezávislosti. Výsledkom testu je p-hodnota, ktorá vyjadruje pravdepodobnosť, že rozdiel medzi skupinami vznikol náhodne.

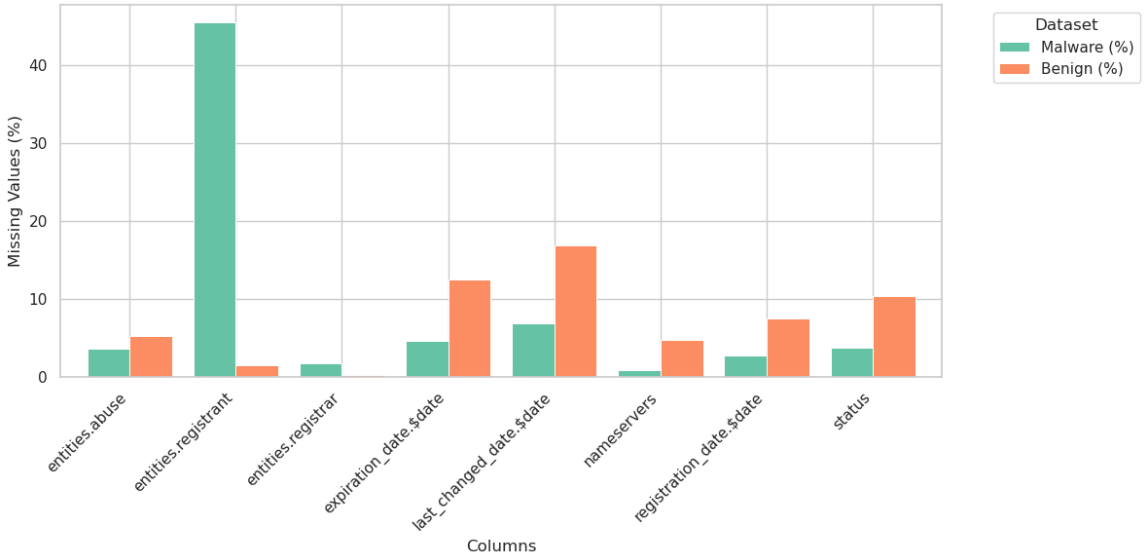
Ak $p < 0{,}05$, rozdiel sa považuje za štatisticky významný, čo naznačuje, že výskyt chýbajúcich hodnôt môže byť spojený s kategóriou domény. Tento test slúži ako doplnkový nástroj k Cohenovmu koeficientu. Zatiaľ čo Cohenov koeficient vyjadruje veľkosť rozdielu, chí-kvadrátový test overuje jeho významnosť vzhľadom na štruktúru údajov. Výsledky testu potvrdili, že prítomnosť chýbajúcich hodnôt vo väčšine atribútov nesie diskriminačný potenciál medzi škodlivými a legitímnymi doménami.

Pri phishingových záznamoch ide najmä o rozdiely v poliach `handle`, registračných dátumoch, `status`, menných serveroch a záznamoch kontaktov pre zneužitie. V prípade malvérových domén sa významné rozdiely objavili napríklad pri `handle`, `url`, `dnssec` alebo `entities.registrant`. Naopak, atribúty ako `whois_server` a `description` nevykazovali štatisticky významné rozdiely. Tieto závery z chí-kvadrátového testu tak potvrdzujú pozorovania získané pomocou Cohenovho koeficientu a poskytujú robustnejší základ pre rozhodovanie o informatívnosti jednotlivých atribútov.

Zohľadnením týchto výsledkov spolu s manuálnou inšpekciou obsahu a rozmanitosti hodnôt v jednotlivých atribútoch s vysokým podielom chýbajúcich údajov bol následne stanovený prah chýbania na úrovni 70%. Tento prah predstavuje praktický kompromis medzi snahou zachovať informatívne znaky a eliminovať atribúty, ktoré poskytujú len minimálny prínos pre klasifikáciu. Polia presahujúce túto hranicu a zároveň nevykazujúce významné rozdiely medzi triedami boli následne označené ako vhodné na vyradenie. Výsledkom tohto viacstupňového filtrovania je finálna množina atribútov, ktoré vykazujú nízku mieru chýbajúcich hodnôt a zároveň disponujú potenciálom na odlíšenie škodlivých domén od legitímnych. Ich zastúpenie v zostávajúcich dátach je znázornené na grafoch 5.7 a 5.8, osobitne pre phishing a malvér.



Obr. 5.7: Miera chýbajúcich hodnôt vo vybraných atribútoch po filtrovaní pre phishingové domény



Obr. 5.8: Miera chýbajúcich hodnôt vo vybraných atribútoch po filtrovaní pre malvérové domény

5.4.2 Konverzia RDAP záznamov do textovej reprezentácie

Vzhľadom na vnorenú a nejednotnú štruktúru RDAP záznamov je nevyhnutné zabezpečiť jednotný formát vstupu pre model. Mnohé atribúty, najmä polia typu `entities`, obsahujú zoznamy objektov s rôznorodými kľúčmi v závislosti od konkrétnej domény. Preto bol najprv vykonaný prieskum štruktúry týchto záznamov s cieľom identifikovať všetky relevantné kľúče, ktoré sa v jednotlivých typoch entít reálne vyskytujú. Na základe toho boli definované jednotné zoznamy „kanonických“ atribútov pre každú entitu (napr. `name`, `email`, `tel`, `handle`), ktoré sa následne systematicky extrahujú a reprezentujú jednotným spôsobom.

Aby bolo možné vstupy spracovať transformerovým modelom, všetky vybrané atribúty sa serializujú do jedného súvislého textového reťazca. Každý záznam je tak transformovaný na sekvenciu textových segmentov, kde jednotlivé bloky reprezentujú hodnoty zvolených polí. Reťazec začína špeciálnym tokenom `[CLS]` a jednotlivé sekcie sú oddelené tokenom `[SEP]`. Vnútorná štruktúra polí je reprezentovaná v tvare `kľúč: hodnota`, pričom viaceré dvojice sú oddeľované znakom `'|'`, a sú tieto bloky oddelené bodkočiarkou. Neprítomné údaje sú označené reťazcom `NA`, čím sa zachováva konzistentná štruktúra naprieč všetkými vstupmi.

Voľba špeciálnych tokenov ako `[CLS]` a `[SEP]` je motivovaná architektúrou transformerových modelov, ako napr. BERT, kde tieto tokeny zohrávajú úlohu pri segmentácii vstupu a inicializácii klasifikácie. Použitie znaku `|` ako vnútorného delimitera zasa prispieva k čitateľnosti a zároveň umožňuje presnejšiu rekonštrukciu štruktúry pri spätnom spracovaní. Výsledný serializovaný vstup je znázornený v príklade 5.1, kde možno vidieť ukážku toho, ako sú jednotlivé segmenty reprezentované a oddelené v súlade s uvedeným formátom.

```
[CLS] domain: pouldersblner.life [SEP] abuse: name: NA | type: entity | email:
    abuse@namecheap.com | url: NA | tel: NA | handle: 10AF0F41201E4614A3B8939BABA4BDF2-
    DONUTS [SEP] registrar: name: NameCheap, Inc. | email: NA | whois_server: NA | url:
    https://rdap.donuts.co/rdap/entity/1068 | tel: NA | handle: 1068 | type: entity [SEP]
    registrant: name: Privacy service provided by Withheld for Privacy ehf | email: NA |
    whois_server: NA | url: NA | rir: NA | tel: NA | handle: NA | type: entity [SEP] admin:
     NA [SEP] nameservers: peaches.ns.cloudflare.com, archer.ns.cloudflare.com [SEP] status
    : client transfer prohibited [SEP]
```

Výpis 5.1: Ukážka serializovaného vstupu po štandardizácii RDAP záznamu

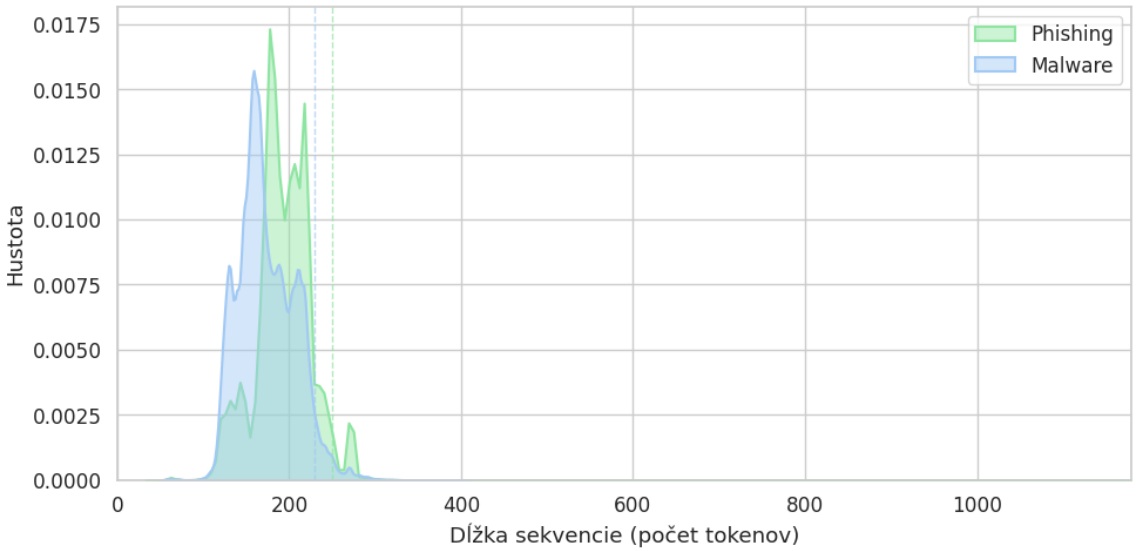
Takto spracované a štandardizované RDAP záznamy predstavujú konzistentný textový vstup, ktorý zachováva štruktúru dôležitých administratívnych a technických údajov o doménach. Získané dáta sú pripravené na vstup do jazykového modelu v podobe jednotných textových reťazcov. Hoci RDAP poskytuje bohaté informácie, jeho praktické využitie je čiastočne obmedzené. Ide o relatívne nový protokol a nie všetky registrové systémy ho implementujú alebo sprístupňujú úplne. Vzhľadom na tieto obmedzenia je dôležité doplniť informácie o ďalší dátový zdroj, ktorý poskytuje pohľad na aktívne fungovanie domén. Nasledujúca časť sa preto venuje analýze údajov z DNS, ktoré prinášajú doplňujúci kontext z hľadiska prevádzkových charakteristík domén.

5.4.3 Určenie maximálnej dĺžky vstupnej sekvencie pre RDAP dáta

Rovnako ako pri doménových menách, aj pri RDAP dátach je potrebné určiť vhodnú maximálnu dĺžku vstupnej sekvencie `max_length`, ktorá zabezpečí pokrytie väčšiny vstupov bez zbytočného predlžovania sekvencie a plytvania výpočtovými prostriedkami. Na tokenizáciu boli použité modely, ktoré zodpovedajú vybraným architektúram. Pre phishingové dáta bol použitý tokenizér `distilbert-base-uncased` a pre malvérové dáta tokenizer `google/electra-base-discriminator`. Distribúcia dĺžok tokenizovaných RDAP vstupov bola analyzovaná osobitne pre phishing a malvér. Výsledky tejto analýzy sú zhrnuté v tabuľke 5.6 a vizuálne znázornené na grafe 5.9.

Ukazovateľ	**Phishing**	**Malvér**
Priemerná dĺžka (mean)	192,96	173,38
Medián (median)	192,00	168,00
Percentilové kvantily		
75.,percentil	215	197
90.,percentil	231	217
95.,percentil	245	227
99.,percentil	273	259
Maximálna hodnota	1170	402

Tabuľka 5.6: Štatistické ukazovatele dĺžok tokenizovaných RDAP záznamov



Obr. 5.9: Pravdepodobnostné rozdelenie dĺžok tokenov v RDAP dátach pre phishing a malvér

Na základe tejto analýzy bola maximálna dĺžka sekvencie stanovená na 230 tokenov pre malvér a 250 tokenov pre phishing, čo zabezpečuje pokrytie viac ako 95% vzoriek v oboch kategóriách. Obe hodnoty sú znázornené na grafe 5.6 prerušovanou čiarou.

5.5 Klasifikácia na základe údajov DNS

Údaje z DNS predstavujú kľúčový zdroj informácií o technickom stave a konfigurácii domén. Na rozdiel od registračných údajov získaných prostredníctvom RDAP, ktoré sa zameriavajú na vlastnícke a administratívne väzby, DNS poskytuje informácie o smerovaní domén, využívaných poštových serveroch a nastaveniach autoritatívnych zón. V týchto parametroch sa môžu objavovať aj nepravidelnosti, ktoré signalizujú potenciálne škodlivé správanie, ako napríklad nezvyčajná konfigurácia názvových serverov, absencia očakávaných záznamov alebo netypické hodnoty v hlavičke zóny. DNS tak dopĺňa obraz o doméne z prevádzkovej perspektívy, ktorá môže byť pre detekciu malvéru alebo phishingu zásadná. Pre ďalšiu analýzu je potrebné porozumieť štruktúre DNS záznamov, ktorá je znázornená v tabuľke 5.7. Táto sumarizácia poskytuje prehľad vybraných polí v dátovej množine, ktoré sa následne transformujú do podoby vhodnej pre spracovanie modelom.

Položka DNS	Dátový typ	Popis
A	Pole reťazcov	Zoznam IPv4 adries, na ktoré doména smeruje.
AAAA	Pole reťazcov	Zoznam IPv6 adries priradených doméne.
CNAME	Reťazec	Kanonický alias domény, ak existuje.
MX	Reťazec alebo pole reťazcov	Informácie o poštových serveroch pre doručovanie e-mailov.
NS	Objekt	Mapa názvových serverov a s nimi súvisiacich IP adries.
TXT	Reťazec alebo pole reťazcov	Ľubovoľné textové údaje priradené k doméne, často používané na autentifikáciu.
SOA	Objekt	Údaje o autoritatívnej zóne, ako sú hlavný názvový server, e-mail správcu a parametre aktualizácie.
zone_SOA	Objekt	Alternatívne alebo doplnkové informácie o zónovej konfigurácii.
dnssec	Objekt	Prítomnosť a stav DNSSEC pre jednotlivé typy záznamov.
remarks	Objekt	Doplňujúce informácie, ako napr. prítomnosť DNSKEY alebo podpisov.
sources	Objekt	Počet nezávislých zdrojov, ktoré potvrdili prítomnosť jednotlivých záznamov.
ttls	Objekt	Hodnoty TTL (Time-To-Live) pre každý typ záznamu.

Tabuľka 5.7: Prehľad hlavných DNS položiek a ich význam

Hoci tabuľka 5.7 poskytuje systematický prehľad dostupných DNS polí, samotná štruktúra dát je v skutočnosti komplexnejšia a menej jednotná. Jednotlivé položky sa líšia nielen svojou prítomnosťou, ale aj vnútornou organizáciou. Niektoré polia obsahujú jednoduché reťazce alebo číselné hodnoty, iné predstavujú vnorené objekty či zoznamy s rôznou dĺžkou a štruktúrou. V niektorých prípadoch môžu záznamy úplne chýbať alebo obsahovať iba predvolené hodnoty, čo si vyžaduje špecifický prístup pri ich spracovaní. Pre lepšie pochopenie tejto variability slúži príklad reálneho DNS záznamu phishingovej domény, uvedený v príklade 5.2.

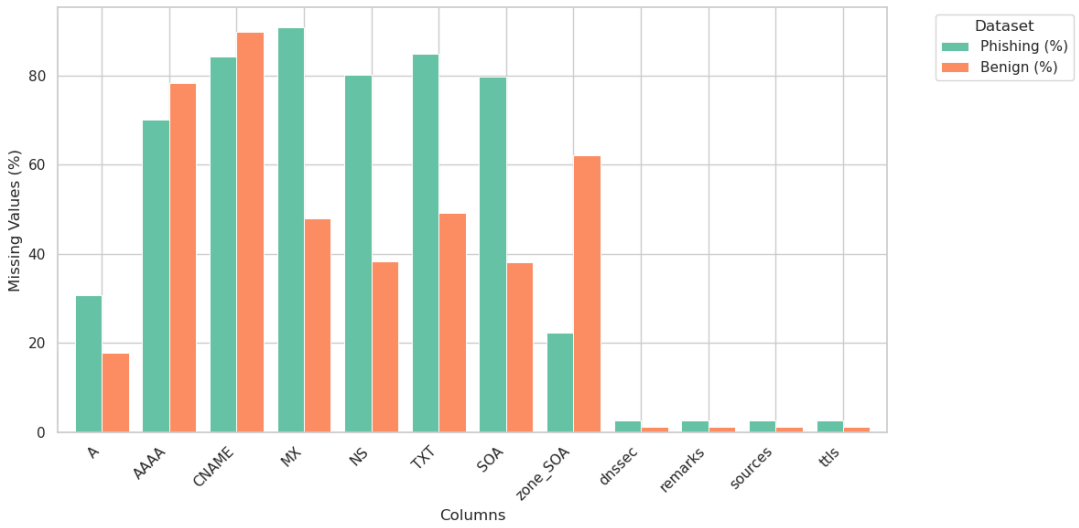
```
dns{
  "A": ["91.202.233.219"],
  "AAAA": null,
  "CNAME": null,
  "MX": null,
  "NS": {
    "ns2.bluehost.com": {
      "related_ips": [
        { "ttl": 3562,"value": "162.159.25.175" }
      ]
    },
    "ns1.bluehost.com": {
      "related_ips": [
        { "ttl": 1288,"value": "162.159.24.80" }
      ]
    }
  },
  "TXT": null,
  "SOA": {
    "primary_ns": "ns1.bluehost.com",
    "resp_mailbox_dname": "root.bluehost.com",
    "serial": 124042510,
    "refresh": 10800,
    "retry": 3600,
    "expire": 604800,
    "min_ttl": 300
  },
  "zone_SOA": null,
  "dnssec": {
    "A": 3,"AAAA": 3,"SOA": 3,"CNAME": 3,
    "MX": 3,"NS": 3,"TXT": 3,"NAPTR": 3
  },
  "remarks": {
    "has_dnskey": false,
    "zone_dnskey_selfsign_ok": false,
    "zone": "espace-maladie.org"
  },
  "sources": {
    "A": 0,"AAAA": 2,"SOA": 0,"CNAME": 2,
    "MX": 2,"NS": 0,"TXT": 2,"NAPTR": 2
  },
  "ttls": {
    "A": 1800,"AAAA": 0,"SOA": 7200,"CNAME": 0,
    "MX": 0,"NS": 7200,"TXT": 0,"NAPTR": 0
  }
}
```


Výpis 5.2: Ukážka reálneho DNS záznamu phishingovej domény

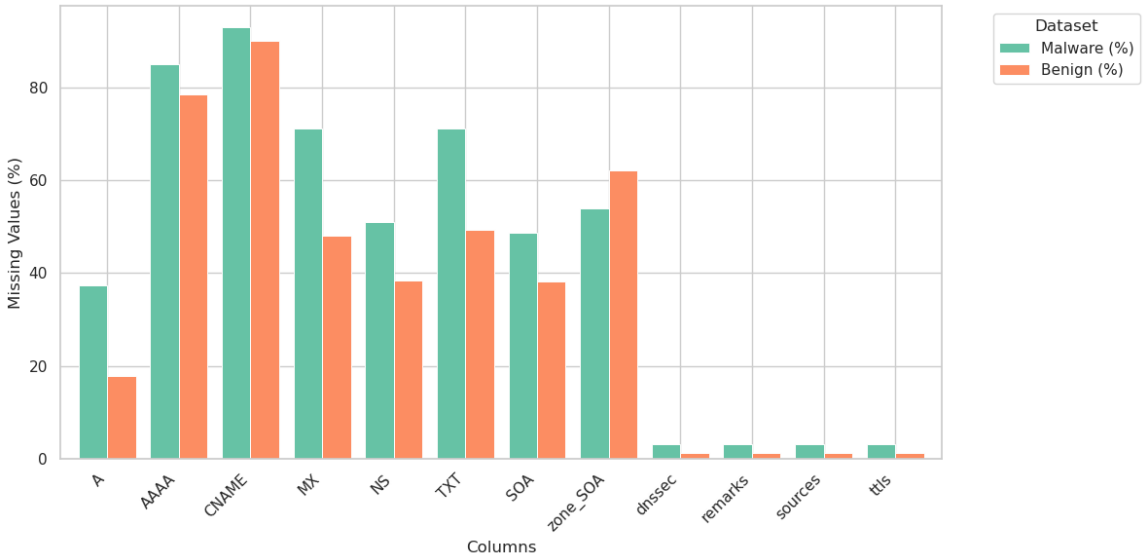
Podobne ako pri údajoch z RDAP, aj DNS záznamy sa vyznačujú výraznou štrukturálnou rozmanitosťou. Niektoré atribúty sa vyskytujú vo väčšine prípadov, iné len ojedinele, pričom ich vnútorná organizácia sa môže výrazne líšiť. Táto variabilita si vyžaduje systematickú analýzu s cieľom identifikovať relevantné a informatívne polia, ktoré sú vhodné na ďalšie spracovanie v rámci klasifikačného modelu.

5.5.1 Analýza a výber atribútov z DNS záznamov

Rovnako ako pri práci s RDAP údajmi, aj v prípade DNS je potrebné transformovať záznamy z pôvodnej hierarchickej štruktúry do tabuľkovej formy, ktorá umožňuje jednotné spracovanie a porovnanie jednotlivých atribútov. Takto pripravené dáta umožňujú systematicky posudzovať prítomnosť a kvalitu polí v škodlivých a legitímnych doménach. V prvom kroku sa identifikujú stĺpce s výlučne chýbajúcimi hodnotami, ako aj záznamy bez akýchkoľvek informatívnych údajov, ktoré sú následne vylúčené z ďalšej analýzy. Na grafoch 5.10 a 5.11 je následne znázornený podiel chýbajúcich hodnôt pre jednotlivé atribúty, osobitne pre phishingové a malvérové domény v porovnaní s legitímnymi.



Obr. 5.10: Percentuálne zastúpenie chýbajúcich hodnôt v jednotlivých DNS atribútoch pri phishingových doménach v porovnaní s legitímnymi.



Obr. 5.11: Percentuálne zastúpenie chýbajúcich hodnôt v jednotlivých DNS atribútoch pri malvérových doménach v porovnaní s legitímnymi.

Z grafov 5.11 a 5.10 možno vyčítať, že najvyšší podiel chýbajúcich údajov sa vyskytuje pri poliach `AAAA`, `CNAME`, `MX` a `TXT`, pričom absencia týchto hodnôt v oboch triedach často presahuje 80%, čo naznačuje, že tieto položky sú všeobecne menej dostupné bez ohľadu na kategóriu domény. Na druhej strane je možné identifikovať aj také atribúty, pri ktorých sa výskyt chýbajúcich hodnôt medzi legitímnymi a škodlivými doménami výrazne odlišuje. Napríklad pole `MX` chýba vo väčšine phishingových záznamov, zatiaľ čo pri benígnych doménach je jeho absencia podstatne nižšia. Výraznejší rozdiel je viditeľný aj pri položkách `NS`, `SOA` a `zone_SOA`, ktoré sú pri legitímnych doménach vyplnené častejšie. Naopak, polia `dnssec`, `remarks`, `sources` a `ttls` sa vyznačujú takmer rovnakým a nízkym podielom chýbajúcich hodnôt vo všetkých sledovaných skupinách, a preto z pohľadu prítomnosti údajov neposkytujú významný rozlišovací signál.

Na kvantitatívne posúdenie rozdielov v absencii hodnôt medzi škodlivými a legitímnymi doménami boli opätovne aplikované štatistické metódy popísané v predchádzajúcej časti 5.4 pri analýze RDAP údajov. Výsledky vyhodnotenia pomocou Cohenovho koeficientu sú nasledujúce:

- **Phishingové domény:** Výrazné rozdiely boli identifikované pri atribútoch `MX`, `NS`, `SOA`, `zone_SOA` a `TXT`, kde hodnota h presahovala hranicu veľkého efektu. Naopak, nízke rozdiely boli pozorované pri poliach ako `AAAA`, `CNAME`, `dnssec`, `remarks`, `sources`, `ttls` a `domain_name`.
- **Malvérové domény:** Najvyššie rozdiely boli zaznamenané pri poliach `MX`, `TXT`, `A`, `NS` a `SOA`, hoci efekt bol spravidla menší ako pri phishingu. Naopak, minimálne rozdiely sa prejavili pri atribútoch `AAAA`, `zone_SOA`, `dnssec`, `remarks`, `sources`, `ttls`, `CNAME` a `domain_name`.

Treba opätovne zdôrazniť, že nízka hodnota rozdielu neznamená nutnosť automatického odstránenia atribútu. Skôr poukazuje na to, že absencia hodnoty sa medzi triedami nevyskytuje systematicky, a preto pri vysokej celkovej miere chýbajúcich údajov možno takéto pole zvážiť na vyradenie bez zásadného oslabenia diskriminačnej schopnosti modelu. Na doplnenie predchádzajúcej analýzy bol opäť aplikovaný chí-kvadrátový test, ktorý už bol využitý pri hodnotení RDAP atribútov. Prehľad výsledkov chí-kvadrátového testu pre jednotlivé atribúty DNS záznamov v oboch škodlivých kategóriách je nasledujúci:

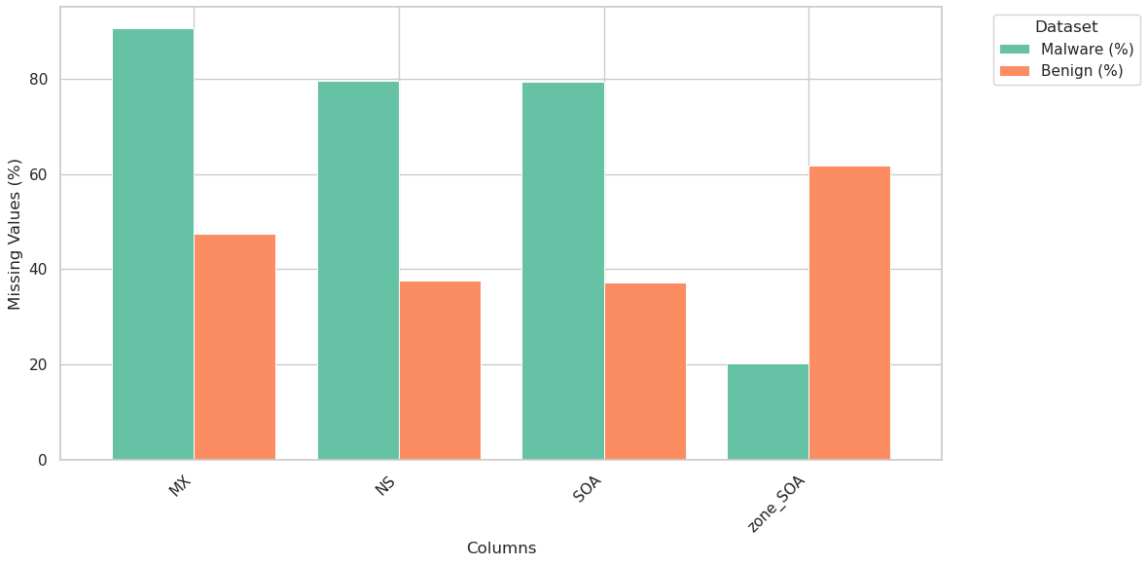
- **Phishingové domény:** Významná štatistická závislosť bola identifikovaná pre všetky sledované atribúty. Najvýraznejšie rozdiely v zastúpení medzi legitímnymi a phishingovými doménami sa prejavili pri poliach `MX`, `NS`, `SOA`, `TXT` a `zone_SOA`.
- **Malvérové domény:** Aj v tejto skupine boli všetky analyzované polia vyhodnotené ako štatisticky významné. Najväčší rozdiel medzi malvérovými a benígnymi záznamami bol taktiež pozorovaný v prípadoch `MX`, `TXT`, `A`, `NS` a `SOA`, čo poukazuje na ich konzistentnú informatívnosť naprieč typmi škodlivého správania.

Na rozdiel od Cohenovho koeficientu, ktorý kvantifikuje veľkosť rozdielu, chí-kvadrátový test sa zameriava na to, či je tento rozdiel štatisticky významný. Zatiaľ čo pri RDAP údajoch niektoré atribúty nevykazovali významnú súvislosť, v prípade DNS záznamov boli všetky polia vyhodnotené ako relevantné. Tieto výsledky naznačujú, že medzi škodlivými a legitímnymi doménami existujú systematické rozdiely v tom, ktoré údaje v DNS záznamoch chýbajú. To poukazuje na potenciál týchto atribútov prispieť k presnejšiemu rozlišovaniu medzi triedami.

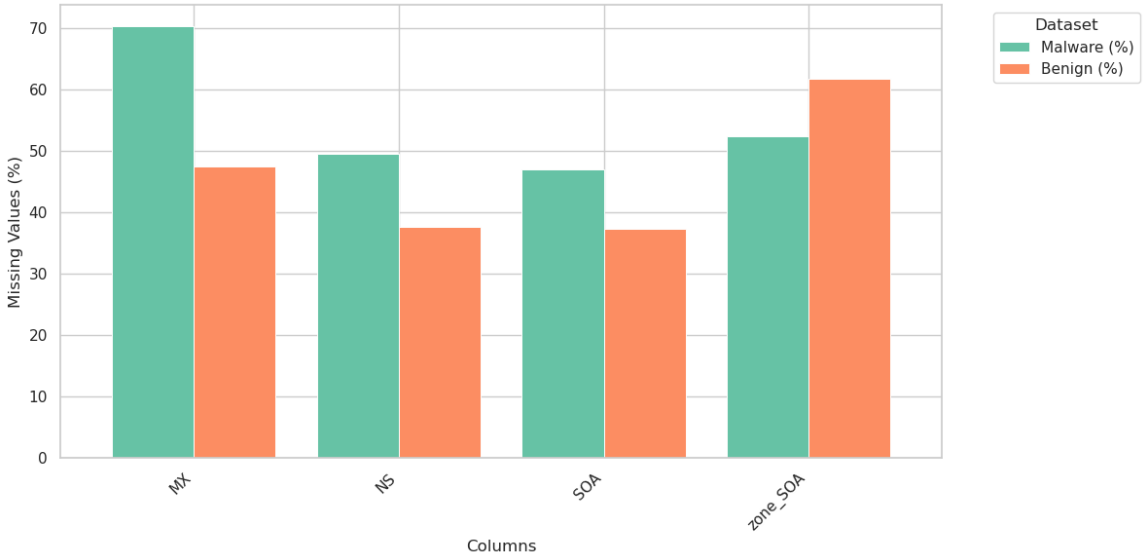
V nadväznosti na tieto zistenia bol vybraný súbor DNS atribútov, ktoré majú potenciál prispieť k účinnej klasifikácii a zároveň sa hodia na textovú reprezentáciu pre jazykový model:

- **domain_name** – názov domény ako hlavný identifikátor,
- **MX** – názvy poštových serverov, ak sú prítomné,
- **NS** – názvy autoritatívnych DNS serverov,
- **SOA** – primárny server a kontaktný e-mail správcu zóny,
- **zone_SOA** – alternatívne údaje o zóne (ak sú k dispozícii),
- **remarks** – doplňujúce textové informácie, ako napríklad názov zóny.

Grafy 5.12 a 5.13 zobrazujú podiel chýbajúcich hodnôt pre tie z uvedených polí, ktoré nevykazovali úplnú prítomnosť v dátach. Polia s úplným pokrytím (napr. `domain_name`) nie sú v grafoch zobrazené, avšak zostávajú súčasťou výslednej reprezentácie. Táto vizualizácia umožňuje lepšie porovnať mieru absencie vybraných atribútov medzi legitímnymi a škodlivými doménami a zároveň potvrdzuje vhodnosť konečného výberu.



Obr. 5.12: Podiel chýbajúcich hodnôt pre vybrané DNS polia pri phishingových a benígnych doménach po odstránení irelevantných atribútov



Obr. 5.13: Podiel chýbajúcich hodnôt pre vybrané DNS polia pri malvérových a benígnych doménach po odstránení irelevantných atribútov

5.5.2 Konverzia DNS záznamov do textovej reprezentácie

Po výbere relevantných polí boli jednotlivé záznamy transformované do jednotného textového formátu, ktorý zodpovedá požiadavkám vstupu do jazykového modelu. Tento proces bol realizovaný prostredníctvom funkcie, ktorá pre každý riadok v dátovom rámci vygeneruje zreťazený text pozostávajúci z vybraných DNS atribútov. Vstupná sekvencia je štruktúrovaná ako zoznam textových dvojíc vo formáte „`meno atribútu: hodnota`“, pričom jednotlivé sekcie sú oddelené špeciálnym tokenom `[SEP]` a celá veta je obalená počiatočným a koncovým tokenom `[CLS]` a `[SEP]`.

Spracovanie jednotlivých atribútov prebieha nasledovne:

- **domain_name:** Doménové meno je konvertované na malé písmená a odstráni sa z neho prefix `www.`, ak je prítomný.
- **MX:** Ak je prítomný, zo štruktúry typu objekt sa extrahujú všetky kľúče, ktoré reprezentujú názvy poštových serverov. Tie sú zreťazené čiarkou.
- **NS:** Ak je hodnota zoznam, jednotlivé položky sa spoja do jedného reťazca oddeleného čiarkami.
- **SOA:** Zo záznamu sa extrahujú iba dve textové polia – názov hlavného servera (`primary_ns`) a e-mail správcu domény (`resp_mailbox_dname`).
- **zone_SOA:** Tento atribút sa spracováva analogicky ako `SOA`.
- **remarks:** Do finálnej reprezentácie sú zahrnuté len tie páry, ktorých hodnoty sú reťazce – napríklad názov zóny. Boolovské alebo prázdne hodnoty sú ignorované.

Ak niektorý z atribútov v konkrétnom zázname chýba alebo nemá vhodnú formu, je nahradený textom `NA`, čím sa zabezpečí konzistentná dĺžka a štruktúra vstupného textu.

Výsledkom prevodu je konzistentná textová reprezentácia každého DNS záznamu, zostavená z vybraných atribútov ako názvové servery, SOA záznamy či doplňujúce poznámky. Tieto prvky sú spojené do jednej štruktúrovanej vety, ktorá následne slúži ako vstup pre tokenizáciu. Výstup tokenizéra potom vstupuje do samotného jazykového modelu. Ukážka takejto textovej transformácie je uvedená v príklade 5.3.

```
[CLS] domain: espace-maladie.org [SEP] MX: NA [SEP] NS: ns1.bluehost.com, ns2.bluehost.com
    [SEP] SOA: primary_ns: ns1.bluehost.com | resp_mailbox_dname: root.bluehost.com [SEP]
    zone_SOA: NA [SEP] remarks: zone: espace-maladie.org [SEP]
```

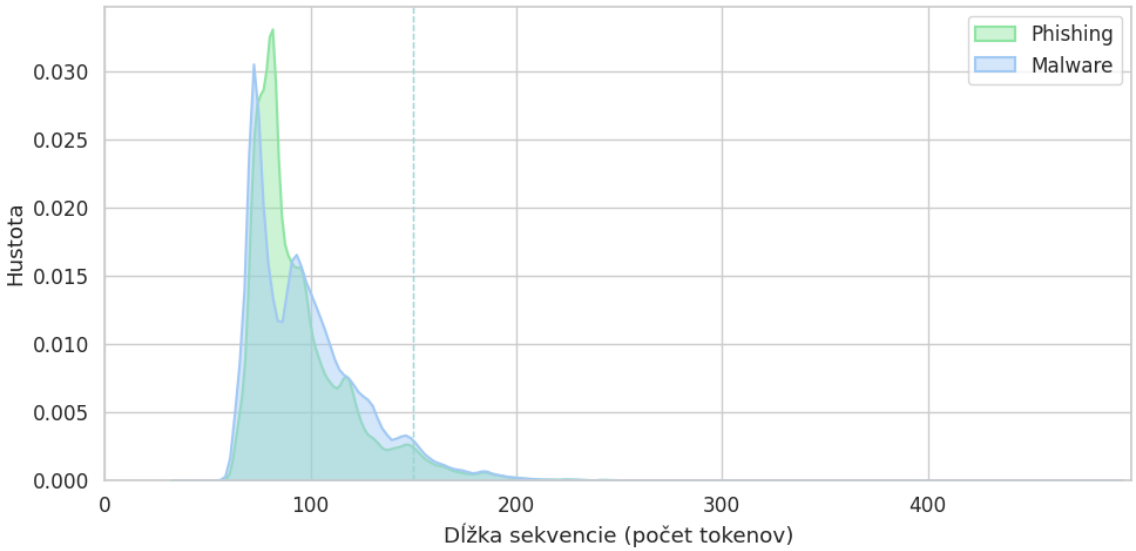
Výpis 5.3: Ukážka textovej reprezentácie DNS záznamu

5.5.3 Určenie maximálnej dĺžky vstupnej sekvencie pre DNS dáta

V súlade s predchádzajúcim prístupom bolo aj pri DNS dátach potrebné stanoviť primeranú maximálnu dĺžku vstupnej sekvencie `max_length`, ktorá zabezpečí vysoké pokrytie bez zbytočnej záťaže na pamäťové a výpočtové prostriedky. Na tokenizáciu boli použité tokenizéry podľa zvolených modelov, pričom pre phishing bol použitý `distilbert-base-uncased` a pre malvér `google/electra-base-discriminator`. Distribúcia dĺžok tokenizovaných sekvencií bola analyzovaná osobitne pre phishing a malvér. Prehľad štatistických ukazovateľov je uvedený v tabuľke 5.8 a zodpovedajúca distribúcia hustoty je znázornená na grafe 5.14.

Ukazovateľ	**Phishing**	**Malvér**
Priemerná dĺžka (mean)	94,69	97,60
Medián (median)	86,00	92,00
Percentilové kvantily		
75.,percentil	104	112
90.,percentil	128	135
95.,percentil	147	150
99.,percentil	182	184
Maximálna hodnota	324	489

Tabuľka 5.8: Štatistické ukazovatele dĺžok tokenizovaných DNS záznamov



Obr. 5.14: Pravdepodobnostné rozdelenie dĺžok tokenov v DNS dátach pre phishing a malvér

Na základe vykonanej analýzy bola maximálna dĺžka sekvencie stanovená na 150 tokenov. Táto hodnota pokrýva viac ako 95% všetkých prípadov a poskytuje rozumný kompromis medzi efektivitou a úplnosťou reprezentácie vstupov. Hodnota je zároveň vyznačená na grafe prerušovanou čiarou.

5.6 Klasifikácia na základe geografických údajov IP adries

Geografická lokalizácia IP adries predstavuje ďalší dôležitý aspekt, ktorý môže napomôcť pri rozlišovaní medzi škodlivými a legitímnymi doménami. Analytické výsledky prezentované v časti 5.1 ukazujú, že práve tieto údaje majú výrazný informačný prínos pre klasifikačný proces. Tieto údaje sú uložené v poli `ip_data`, ktoré obsahuje zoznam všetkých IP adries priradených k doméne. Ku každej z týchto adries sú navyše evidované doplňujúce informácie, medzi ktoré patrí aj geografická lokalizácia. Stručný prehľad štruktúry objektov v rámci poľa `ip_data` uvádza tabuľka 5.9.

Položka	Dátový typ	Popis
`ip`	Reťazec	IP adresa priradená k doméne (typicky záznam typu A alebo AAAA).
`from_record`	Reťazec	Typ DNS záznamu, z ktorého bola IP adresa extrahovaná.
`geo`	Objekt	Geografická poloha adresy: štát, mesto, súradnice a časové pásmo.
`rdap`	Objekt	Informácie o registrátorovi IP adresy podľa RDAP protokolu.
`asn`	Objekt	Údaje o autonómnom systéme, ktorému IP adresa patrí.
`remarks`	Objekt	Doplňujúce technické informácie, napríklad dostupnosť cez ICMP alebo priemerná odozva.

Tabuľka 5.9: Prehľad základných polí v objekte `ip_data`

5.6.1 Analýza a výber atribútov z geografických údajov

Aj keď objekt `ip_data` obsahuje viacero častí s technickými údajmi, do spracovania boli zaradené výhradne geografické informácie z poľa `geo`. Ostatné podobjekty, ako napríklad `ip`, `asn` alebo `remarks`, pozostávajú prevažne z číselných alebo časových atribútov, ktoré nie sú optimálne na spracovanie pomocou jazykových modelov. Okrem toho, údaje z objektu `rdap` boli už samostatne analyzované v rámci experimentu nad registračnými záznamami domén a ich opätovné zahrnutie by znížilo interpretovateľnosť výstupov modelu. Zameranie sa výhradne na geografické údaje tak umožňuje presnejšie vyhodnotiť ich informatívnosť a jednoznačne posúdiť ich prínos pre úlohu klasifikácie domén. Pre lepšie pochopenie štruktúry geografických údajov uložených v objekte `ip_data` slúži nasledovná ukážka 5.4, ktorá znázorňuje výrez dát prislúchajúcich k jednej z IP adries priradenej k analyzovanej doméne.

```
"ip_data": [
  {
    "ip": "142.251.36.97",
    ...
    "geo": {
      "country": "United States",
      "country_code": "US",
      "region": "New York",
      "region_code": "NY",
      "city": "Queens",
      "postal_code": "11414",
      "latitude": 40.66,
      "longitude": -73.839,
      "timezone": "America/New_York",
      "isp": null,
      "org": null
    },
    ...
  },
  {
```

```
    "ip": "2a00:1450:4014:80b::2001",
    ...
    "geo": {
      "country": "Ireland",
      "country_code": "IE",
      "region": null,
      "region_code": null,
      "city": null,
      "postal_code": null,
      "latitude": 53.3471,
      "longitude": -6.2447,
      "timezone": "Europe/Dublin",
      "isp": null,
      "org": null
    },
    ...
  }
]
```

Výpis 5.4: Výberová ukážka geografických údajov z poľa `ip_data`

Ako je vidieť z ukážky výpisu 5.4, geografické údaje predstavujú štruktúrované textové položky, ako sú názvy štátov, miest, regiónov a časových pásiem. Vďaka svojej podobe sú dobre tokenizovateľné a vhodné na spracovanie pomocou jazykového modelu bez potreby dodatočného kódovania alebo numerickej transformácie.

Dátová analýza ukázala, že miera chýbajúcich hodnôt v týchto poliach nie je výrazná, preto sa samostatné grafické vyhodnotenie v tejto časti neuvádza. Tento stav je ovplyvnený aj spôsobom, akým boli dáta získavané. Ako bolo uvedené v práci [17], pre každú IP adresu získanú z DNS záznamov typu `A`, `AAAA` a z rozlíšených `CNAME` záznamov bola vykonaná `ICMP` echo požiadavka (ping) z jedného kontrolného bodu. Následne sa pomocou databáz GeoLite2 City a ASN určilo geografické umiestnenie a príslušnosť k autonómnym systémom.

5.6.2 Konverzia geografických údajov do textovej reprezentácie

Z extrahovaných geografických údajov bola následne zostavená jednotná textová reprezentácia vhodná pre vstup do jazykového modelu. V prípade, že sa v rámci jednej domény vyskytuje viacero IP adries s rovnakým geografickým umiestnením, tieto hodnoty sa zjednocujú a vo výslednom zápise sa premieta len ich rôznorodosť. Počet unikátnych IP adries je zároveň zaznamenaný samostatne ako textový atribút v podobe `ip_count: X`. Každý záznam tak pozostáva zo štruktúrovanej vety, ktorá obsahuje nasledujúce komponenty:

- **názov domény** – názov analyzovanej domény vo formáte reťazca,
- **počet IP adries** – počet unikátnych IP záznamov spojených s danou doménou,
- **krajiny** – zoznam krajín alebo kódov krajín, ktoré boli identifikované,
- **regióny** – zjednotené názvy regiónov, ak sú k dispozícii,
- **mestá** – identifikované mestá, kde sa IP adresy nachádzajú,
- **časové pásma** – pomenované časové zóny, napríklad `Europe/Dublin`.

Tieto časti sú oddelené pomocou tokenov `[SEP]` a celková sekvencia je uzatvorená medzi symbolmi `[CLS]` a `[SEP]`, čím vzniká formát vhodný pre následné spracovanie tokenizérom jazykového modelu. Výsledná podoba takejto reprezentácie je znázornená v príklade 5.5.

```
[CLS] domain: example.org [SEP]
ip_count: 2[SEP]
countries: Ireland, United States [SEP]
regions: New York [SEP]
cities: Queens [SEP]
timezones: America/New_York, Europe/Dublin [SEP]
```

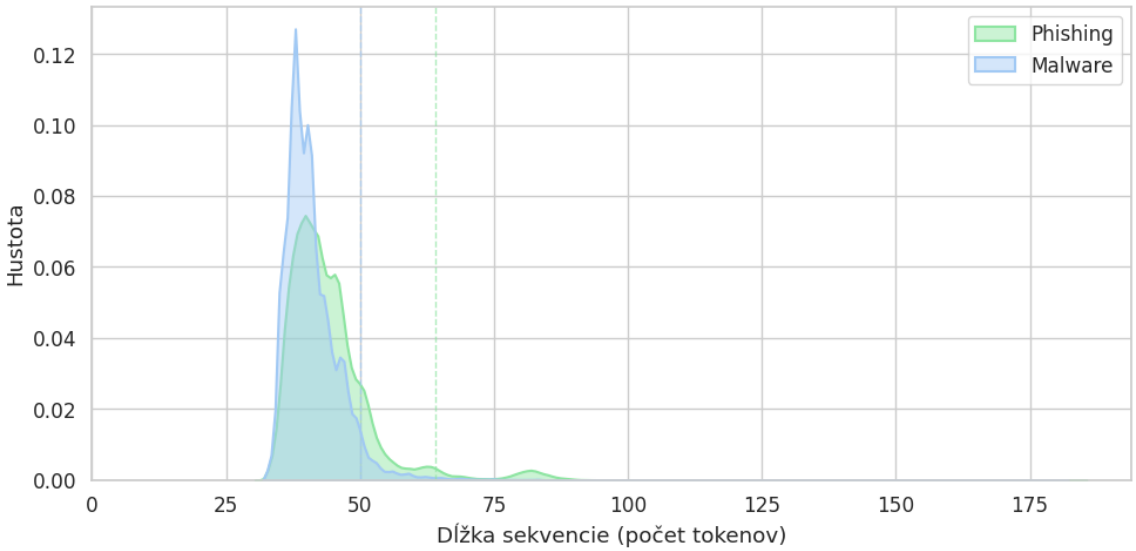
Výpis 5.5: Ukážka textovej reprezentácie geografických údajov domény

5.6.3 Určenie maximálnej dĺžky vstupnej sekvencie pre geografické údaje

Rovnako ako pri predchádzajúcich dátových typoch, bolo aj pri geografických dátach potrebné určiť vhodnú maximálnu dĺžku vstupnej sekvencie `max_length`, ktorá pokrýva väčšinu prípadov a zbytočne nezaťažuje výpočtové prostriedky. Na tokenizáciu boli použité tokenizéry podľa zvolených modelov, pričom pre phishing bol použitý tokenizér `distilbert-base-uncased` a pre malvér tokenizér `google/electra-base-discriminator`. Dĺžky tokenizovaných vstupov boli analyzované samostatne pre phishing a malvér. Súhrn štatistických ukazovateľov je uvedený v tabuľke 5.10, a zodpovedajúce rozdelenie hustoty je zobrazené na grafe 5.15.

Ukazovateľ	**Phishing**	**Malvér**
Priemerná dĺžka (mean)	44,28	41,04
Medián (median)	43,00	40,00
Percentilové kvantily		
75.,percentil	47	43
90.,percentil	52	47
95.,percentil	59	50
99.,percentil	82	60
Maximálna hodnota	184	181

Tabuľka 5.10: Štatistické ukazovatele dĺžok tokenizovaných geografických záznamov



Obr. 5.15: Pravdepodobnostné rozdelenie dĺžok tokenov v geografických dátach pre phishing a malvér

Na základe analýzy bola maximálna dĺžka vstupnej sekvencie nastavená na 50 tokenov pre malvér a 64 tokenov pre phishing, čo zabezpečuje pokrytie 95% všetkých vzoriek v oboch prípadoch. Obidve hodnoty sú opäť znázornené na grafe 5.10.

Kapitola 6

Implementácia

Táto kapitola dokumentuje implementáciu riešenia, ktoré bolo navrhnuté v predchádzajúcej kapitole 5. Cieľom je predstaviť jednotlivé technické komponenty procesu vývoja klasifikačných modelov pre detekciu malígnych domén, ako aj popísať spôsob ich konfigurácie a testovania v rámci experimentov. Riešenie bolo implementované postupne v rámci nasledujúcich etáp:

- príprava a analýza dát,
- návrh a konfigurácia modelov,
- ladenie hyperparametrov,
- trénovanie modelov,
- vyhodnocovanie výsledkov.

V nasledujúcich sekciách je podrobne popísané vývojové prostredie, organizácia zdrojového kódu, ako aj implementácia jednotlivých fáz tohto procesu.

6.1 Vývojové prostredie

Implementácia prebiehala v stabilne nakonfigurovanom prostredí s cieľom zabezpečiť opakovateľnosť experimentov. Táto sekcia poskytuje prehľad o použitom hardvéri, ako aj o softvérových nástrojoch a knižniciach, ktoré tvoria technologický základ celého projektu. Vývojové prostredie možno rozdeliť do dvoch hlavných oblastí:

- **Hardvérové prostriedky** – dostupné výpočtové zdroje, najmä GPU.
- **Softvérová infraštruktúra** – programovací jazyk, knižnice, virtualizácia prostredí.

6.1.1 Hardvérové prostriedky

Tréning a vyhodnocovanie klasifikačných modelov prebiehalo na výpočtovej stanici so systémom Ubuntu 24.04 LTS. Hardvérová konfigurácia zariadenia bola navrhnutá tak, aby umožňovala paralelné spracovanie dát a distribuované trénovanie modelov na báze transformerových architektúr. K dispozícii boli nasledujúce výpočtové prostriedky:

- **Grafické karty:** 2× NVIDIA GeForce RTX 3070
  (8 GB GDDR6 VRAM, architektúra Ampere, 5888 CUDA jadier, maximálna spotreba 220 W).
- **Procesor:** Intel Core i7-11700K
  (8 jadier, 16 vlákien, základná frekvencia 3,60 GHz, maximálna turbo frekvencia 5,00 GHz).
- **Operačná pamäť:** 32 GB DDR4 RAM
  (efektívne dostupná kapacita približne 29 GB).

Táto konfigurácia poskytla dostatočný výpočtový výkon na efektívne trénovanie viacerých modelov súčasne, ako aj na realizáciu rozsiahlejších experimentov s rôznymi konfiguráciami. Všetky experimenty boli spúšťané v režime so zdieľanou GPU[1] pamäťou a rozdelením úloh medzi dve dostupné GPU zariadenia v rámci pracovnej stanice.

6.1.2 Softvérová infraštruktúra

Implementácia riešenia bola realizovaná v programovacom jazyku Python 3.10. Použitie tejto verzie bolo zvolené z dôvodu zabezpečenia plnej kompatibility s kľúčovými knižnicami, ako sú PyTorch a Transformers, ktoré v čase implementácie oficiálne podporovali práve túto verziu.

Z dôvodu konfliktov medzi závislosťami niektorých knižníc boli vytvorené dve samostatné virtuálne prostredia. Jedno prostredie bolo určené na ladenie hyperparametrov, druhé na samotný tréning modelov. Takéto oddelenie umožnilo minimalizovať problémy spojené s nekompatibilitou balíčkov a zároveň zjednodušilo správu experimentov. V nasledujúcom výpise sú uvedené základné knižnice, na ktorých je postavená implementácia celého riešenia:

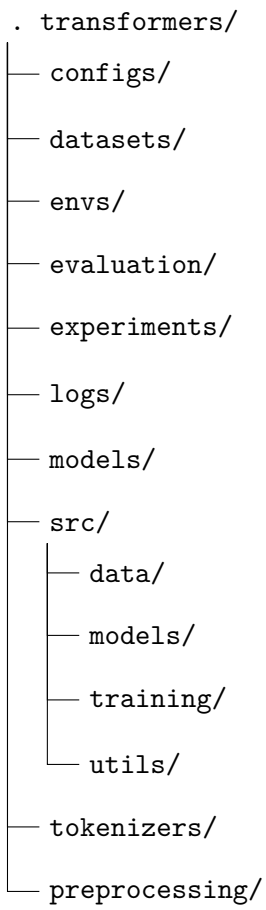
- **PyTorch**[2] – framework pre definíciu neuronových sietí a trénovanie modelov.
- **Transformers (HuggingFace)**[3] – nástroje na načítavanie a jemné doladenie predtrénovaných jazykových modelov.
- **Hydra**[4] – systém pre správu experimentálnych konfigurácií pomocou YAML súborov.
- **scikit-learn**[5] – metriky, rozdelenie datasetov, jednoduché vyhodnocovanie.
- **pandas**[6] – načítavanie, filtrovanie a transformácia dát v tabulárnej forme.
- **tqdm**[7] – vizualizácia priebehu iterácií počas trénovania.

Ďalšie špecializované knižnice, ktoré boli použité, budú uvedené priamo v príslušných sekciách, spolu s opisom ich funkcie a kontextu použitia.

[1] Graphical Processing Unit
[2] `https://pytorch.org`
[3] `https://huggingface.co/docs/transformers`
[4] `https://hydra.cc`
[5] `https://scikit-learn.org`
[6] `https://pandas.pydata.org`
[7] `https://tqdm.github.io`

6.2 Štruktúra repozitára

Zdrojový kód a podporné súbory sú organizované v repozitári, ktorého štruktúra odráža jednotlivé fázy vývoja riešenia. Zámerom bolo dosiahnuť prehľadnosť, modularitu a jednoduchú replikovateľnosť experimentov. Adresárová štruktúra repozitára bola navrhnutá tak, aby zodpovedala jednotlivým krokom vývoja a zároveň bola prehľadne usporiadaná. Pre ilustráciu slúži nasledujúci výpis:



Obr. 6.1: Adresárová štruktúra repozitára

Jednotlivé priečinky v repozitári pokrývajú všetky hlavné časti implementácie: od spracovania dát, cez návrh a trénovanie modelov, až po evidenciu výstupov experimentov. Ich účel je nasledovný:

- **`preprocessing/`** – obsahuje Jupyter notebooky a Python skripty využité na predspracovanie a analýzu vstupných dát. V tejto fáze prebiehalo čistenie, selekcia a transformácia pôvodných `.json` súborov na formát `.csv`.
- **`datasets/`** – obsahuje predspracované a anotované dátové súbory vo formáte `.csv`, ktoré slúžia ako vstup pre trénovanie klasifikačných modelov.
- **`src/data/`** – implementácia nástrojov na načítavanie datasetov a definíciu dátových tried podľa typu vstupu (napr. RDAP, DNS, geografické dáta, doménové mená).

- `tokenizers/` – implementácia tokenizačných stratégií predstavených v kapitole 5, určených na experimentovanie s vlastnou architektúrou.
- `src/models/` – obsahuje moduly zodpovedné za inštanciáciu a konfiguráciu modelových architektúr.
- `src/training/` – implementácia trénovacieho cyklu.
- `experiments/` – priečinok určený na spúšťanie experimentov. Obsahuje hlavný skript na spustenie trénovania a skript na ladenie hyperparametrov.
- `configs/` – YAML konfigurácie pre nastavenie dát, modelov, trénovania a experimentálnych scenárov.
- `logs/` – priečinok s CSV súbormi zaznamenávajúcimi priebeh tréningu.
- `models/` – adresár pre uloženie výstupných modelov. Obsahuje najlepšie a finálne checkpointy z jednotlivých trénovaní.
- `evaluation/` – priečinok určený na vyhodnocovanie natrénovaných modelov. Obsahuje Jupyter notebooky a skripty pre analýzu metrík a vizualizáciu výsledkov.
- `envs/` – definície dvoch samostatných virtuálnych prostredí použitých počas vývoja (viď sekcia 6.1.2).

Nasledujúce sekcie podrobne popisujú implementáciu jednotlivých častí riešenia v poradí, ktoré zodpovedá ich úlohe v celom vývojovom procese.

6.3 Získavanie a export dát z databázy

Pôvodné dáta boli uložené v dokumentovo orientovanej databáze MongoDB. Každý záznam predstavoval jedno doménové meno spolu s viacerými dátovými vrstvami, ako sú DNS záznamy, údaje z RDAP/WHOIS, informácie o IP adresách a ich geografickej lokalizácii. Tieto záznamy boli rozsiahle a štruktúrované, čo pri hromadnom exporte a načítaní do pamäte (napr. do `pandas.DataFrame`) predstavovalo riziko zahltenia operačnej pamäte.

Z tohto dôvodu boli jednotlivé dátové vrstvy exportované samostatne. Každý výstup obsahoval iba jeden typ dát (napr. DNS, RDAP alebo IP údaje), pričom spoločným identifikátorom zostával atribút `domain_name`, ktorý slúžil na ich vzájomné spárovanie. Pre každú kategóriu domén (malvérové, phishingové, benígne) vznikli samostatné exporty pre každú vrstvu. Výnimku predstavovali DGA domény, kde boli k dispozícii iba samotné doménové mená bez ďalších atribútov. Tie boli z databázy exportované priamo vo formáte CSV. Export z databázy bol realizovaný pomocou nástroja `mongoexport`, ktorý umožňuje selektívne získanie konkrétnych polí z databázy. Vzorový príkaz na export jednej vrstvy údajov pre zvolenú kolekciu je uvedený vo výpise 6.1:

```
mongoexport \
  --uri="mongodb://<USERNAME>:<PASSWORD>@<HOST>:<PORT>/<DB_NAME>" \
  --collection=<COLLECTION_NAME> \
  --type=json \
  --fields=domain_name,<FIELD> \
  --out=<OUTPUT_FILENAME>.json \
  --verbose
```

Výpis 6.1: Vzorový príkaz na export vybranej dátovej vrstvy z MongoDB

Tento príkaz bol prispôsobený pre každú kombináciu doménovej kategórie a dátovej vrstvy, pričom parameter `<FIELD>` nadobúdal hodnoty ako `dns`, `rdap` alebo `ip_data`, a kolekcie boli zvolené podľa konkrétnej kategórie domén. Výsledkom exportu sú samostatné súbory obsahujúce výhradne záznamy jednej kategórie dát. Tieto súbory sú výrazne menšie ako pôvodné záznamy a umožňujú efektívne spracovanie v pamäťovo obmedzenom prostredí. Ich štruktúra je nasledovná:

- **Benígne domény:**
  - `domains_benign.json`,
  - `rdap_benign.json`,
  - `dns_benign.json`,
  - `ip_benign.json`.
- **DGA domény:**
  - `domains_dga.csv`.
- **Phishingové domény:**
  - `rdap_phishing.json`,
  - `dns_phishing.json`,
  - `ip_phishing.json`.
- **Malvérové domény:**
  - `rdap_malware.json`,
  - `dns_malware.json`,
  - `ip_malware.json`.

Takto pripravené exporty predstavujú výkonný a pamäťovo úsporný základ pre ďalšiu dátovú analýzu, výber atribútov a konverziu vstupov do formátu vhodného pre jazykové modely. Nasledujúce sekcie sa venujú spracovaniu obsahu jednotlivých vrstiev a návrhu ich reprezentácie.

6.4 Analýza a predspracovanie dát

Pre potreby trénovania modelu bolo potrebné dôkladne pripraviť všetky dostupné dáta do jednotného a konzistentného formátu. Tento proces zahŕňal rôzne techniky čistenia, transformácie a výberu atribútov podľa typu údajov a ich štruktúry. Cieľom bolo znížiť redundanciu, odstrániť nekonzistencie a zároveň zachovať relevantné informácie pre následnú analýzu. Všetky kroky súvisiace s predspracovaním boli implementované v priečinku `data_processing/`, ktorý obsahuje Jupyter notebooky a Python skripty. Notebooky boli využité najmä v prípadoch, kde bolo potrebné priebežne analyzovať dáta a vizualizovať priebeh spracovania. Naopak, pre úlohy s jasne definovaným priebehom, ktoré bolo vhodné opakovane spúšťať, boli použité samostatné skripty.

6.4.1 Tvorba vyváženého datasetu z doménových mien

Doménové mená tvoria základ každého vstupu pre model, pretože k nim boli v ďalších krokoch pripájané všetky ostatné dátové vrstvy (ako DNS, RDAP či geolokačné informácie) vo forme reťazca. Takto vznikol jednotný vstupný formát vhodný pre spracovanie transformátorovým modelom.

Aby nedošlo k skresleniu modelu v dôsledku nerovnomerného zastúpenia tried, boli pre každú úlohu vytvorené vyvážené trénovacie množiny s rovnakým počtom benígnych a škodlivých domén. Benígne domény boli náhodne vyberané z dvoch nezávislých zdrojov, čím sa zvýšila ich rozmanitosť a znížilo riziko nadmerného prispôsobenia modelu špecifickým vzorom. Škodlivé domény pochádzali z kategórií malvér, phishing a DGA, pričom každá z nich bola spracovaná samostatne. Ako ilustračný príklad možno uviesť prípravu datasetu pre detekciu DGA domén, kde boli náhodne vybrané dve vzorky: 100 000 benígnych domén (50 000 z dátovej sady Cisco a 50 000 z dátovej sady Umbrella) a 100 000 domén generovaných algoritmom DGA. Celý proces výberu, zlúčenia a náhodného premiešania údajov bol realizovaný pomocou skriptu `domain_names_preprocess.py`, ktorý je súčasťou priečinka `preprocessing`. Výstupom je prehľadný CSV súbor pripravený na ďalšie spracovanie modelom.

6.4.2 Spracovanie RDAP a DNS údajov

Pre spracovanie údajov z vrstiev RDAP a DNS boli vytvorené samostatné Jupyter notebooky pre každú kombináciu kategórie domény (malvér, phishing) a typu údajov. Napriek tomu, že jednotlivé datasety sa líšili v štruktúre a miere chýbajúcich hodnôt, princíp spracovania bol jednotný. Notebooky sú umiestnené v priečinku `preprocessing` a zahŕňajú:

- `rdap_phishing_preprocess.ipynb`,
- `rdap_malware_preprocess.ipynb`,
- `dns_phishing_preprocess.ipynb`,
- `dns_malware_preprocess.ipynb`.

Spracovanie údajov prebiehalo v troch hlavných krokoch:

1. **Načítanie a transformácia údajov**
   Vstupné `.json` súbory boli načítané a transformované do tabuľkového formátu pomocou funkcie `pandas.json_normalize`, pričom vnorené štruktúry boli rozbalené

do samostatných stĺpcov. Takto vzniknuté dátové rámce obsahovali jeden riadok na každú doménu a množinu atribútov pripravených na analýzu.

2. **Výber relevantných atribútov**
   Na identifikáciu irelevantných alebo príliš neúplných atribútov sa využili viaceré štatistické prístupy. Rozdiely v miere chýbajúcich hodnôt medzi skupinami sa kvantifikovali pomocou implementovanej funkcie `cohens_h`, založenej na vzorci uvedenom v návrhu riešenia. Závislosť medzi prítomnosťou atribútov a triedou domény bola testovaná pomocou chí-kvadrát testu, implementovaného funkciou `chi2_contingency` z knižnice `scipy.stats`[8] Na základe týchto analýz boli z výsledných dátových rámcov odstránené všetky stĺpce, ktoré neprinášali prínos pre klasifikáciu alebo mali extrémne vysokú mieru chýbajúcich hodnôt.

3. **Konverzia na vstupný formát pre model**
   Významové atribúty (napríklad pri DNS `NS`, `SOA` či `remarks`) boli konvertované do textovej podoby a spoločne s doménovým menom zreťazené do jedného textového reťazca vo formáte prispôsobenom pre transformátorový model. Každý riadok výsledného datasetu tak predstavoval jeden zreťazený vstup vo formáte:

   ```
   [CLS] domain: example.com [SEP] NS: ns1.example.net [SEP] SOA: ...
   ```

   Výsledné datasety boli následne uložené pomocou funkcie `to_csv`. Názvy súborov boli generované podľa schémy `<typ_udajov>_<kategoria>_preprocessed.csv`, napríklad `dns_phishing_preprocessed.csv`.

6.4.3 Spracovanie geolokačných údajov

Na rozdiel od vrstiev RDAP a DNS boli geolokačné údaje spracované jednotným spôsobom bez ohľadu na kategóriu domény. Formát týchto údajov bol konzistentný — každá doména obsahovala atribút `ip_data`, ktorý predstavoval zoznam IP adries s vnorenými geografickými informáciami (napríklad krajina, región, mesto alebo časové pásmo). Keďže tieto údaje pochádzali z externých zdrojov, nevyskytovali sa v nich výrazné štrukturálne rozdiely, a preto nebolo potrebné ich čistiť alebo filtrovať. Spracovanie údajov prebiehalo pomocou skriptu `geo_preprocess.py`, ktorý pozostával z nasledujúcich krokov:

1. **Načítanie a transformácia údajov**
   Vstupné súbory vo formáte `.json` boli načítané po riadkoch ako samostatné objekty, pričom vnorené záznamy boli transformované do tabuľkového formátu rovnakým spôsobom ako pri údajoch RDAP a DNS, teda použitím `pandas.json_normalize`.

2. **Transformácia na vstupný formát pre model**
   Geografické údaje boli extrahované z poľa `ip_data`, ktoré mohlo obsahovať viacero záznamov IP adries pre jednu doménu. Z každého záznamu boli získané informácie ako krajina, mesto alebo časové pásmo. Pri tvorbe vstupného textu boli tieto hodnoty deduplikované — napríklad ak sa rovnaké mesto vyskytovalo pri viacerých adresách, vo výstupe sa objavilo iba raz. Zároveň bol určený atribút `ip_count`, ktorý vyjadroval celkový počet IP adries priradených ku konkrétnej doméne.

   ```
   [CLS] domain: example.com [SEP] ip_count: 2 [SEP] countries: US, DE ...
   ```

[8] Knižnica `scipy.stats` je štandardnou súčasťou ekosystému Python pre vedecké výpočty.

3. **Zlúčenie a vyváženie údajov**
   Rovnako ako pri iných vrstvách boli aj v tomto prípade vytvorené vyvážené datasety, ktoré obsahovali rovnaký počet benígnych a škodlivých domén (malvér a phishing). Vstupné riadky boli náhodne premiešané a následne uložené vo formáte CSV. Názvy výstupných súborov sa držali jednotného pomenovania, napríklad `ip_phishing_preprocessed.csv`.

6.5 Konfiguračný systém a parametrizácia experimentov

Konfiguračný systém slúži na zjednodušenie spúšťania experimentov, zabezpečenie reprodukovateľnosti a pohodlnú správu parametrov počas celého vývoja. Umožňuje definovať experimenty bez nutnosti upravovať zdrojový kód a zároveň podporuje spustenie viacerých experimentov naraz pomocou jedného konfiguračného súboru. Na implementáciu konfiguračného systému bola použitá knižnica `Hydra`, ktorá umožňuje skladanie konfigurácií z viacerých YAML súborov. Do skriptov sa integruje pomocou dekorátora `@hydra.main`, ktorý pri spustení načíta všetky požadované konfigurácie a odovzdá ich ako objekt typu `DictConfig` priamo do hlavnej funkcie.

6.5.1 Hlavné konfiguračné súbory

Všetky konfiguračné súbory sa nachádzajú v priečinku `configs/`. Výsledná konfigurácia vzniká ako kombinácia viacerých čiastkových súborov, ktoré sa prepájajú pomocou sekcie `defaults`. Každý z hlavných súborov slúži ako vstupná konfigurácia pre určitý typ experimentu. Medzi hlavné súbory patria:

- `pretrained_config.yaml`
  - Konfigurácia pre spúšťanie experimentov s predtrénovanými modelmi.
  - Obsahuje odkaz na položku `experiments`, ktorá umožňuje definovať viacero experimentov v rámci jedného súboru.
- `custom_config.yaml`
  - Používa sa pri experimentoch s vlastnou architektúrou a vlastnými tokenizérmi.
  - Definuje parametre špecifické pre vlastný model, ako napríklad spôsob tokenizačnej stratégie.

Príklad obsahu súboru `pretrained_config.yaml` je uvedený vo výpise 6.2.

```yaml
defaults:
  - data: rdap
  - task: malware
  - model: distilbert_base_uncased
  - train: bert_architectures_params
  - experiments: null
  - _self_

data_root: ../datasets
seed: 42
distributed_port: 23456

hydra:
  run:
    dir: .
  job:
    chdir: false
```

Výpis 6.2: Ukážka hlavnej konfigurácie pre predtrénované modely

Z výpisu 6.2 je možné si všimnúť, že tento súbor prepája všetky potrebné komponenty konfigurácie. Hodnoty definované v referencovaných súboroch sa automaticky načítajú a sprístupnia v rámci jedného objektu typu `DictConfig`.

6.5.2 Organizácia čiastkových konfigurácií

Podpriečinky v adresári `configs/` sú usporiadané podľa jednotlivých častí experimentálneho procesu. Každý priečinok obsahuje YAML súbory, ktoré definujú konkrétnu oblasť nastavenia experimentu. Tieto oblasti zodpovedajú hlavným fázam práce s modelmi, ako je príprava dát, definovanie modelovej architektúry alebo konfigurácia trénovania.

- `data/` Obsahuje konfigurácie pre rôzne typy dátových vstupov. Definuje cestu k súboru, typ dát a pomer rozdelenia trénovacích a validačných množín.
- `model/` Obsahuje konfigurácie modelov. Pri predtrénovaných architektúrach sa uvádza iba názov modelu. Pri vlastných modeloch sú definované aj parametre tokenizácie a ďalšie architektonické vlastnosti.
- `task/` Konfigurácie pre klasifikačné úlohy. Definujú napríklad maximálne dĺžky vstupov podľa typu dát.
- `train/` Obsahuje parametre trénovania ako počet epoch, veľkosť trénovacej dávky (batch size), rýchlosť učenia alebo typ plánovača učenia.
- `experiments/` Obsahuje YAML súbory, ktoré definujú konkrétne experimenty. Tieto konfigurácie umožňujú spustenie viacerých modelov s rôznymi nastaveniami bez potreby manuálneho spúšťania každej konfigurácie zvlášť.

Takto navrhnutý konfiguračný systém poskytuje vysokú flexibilitu, prehľadnosť a zároveň podporuje efektívne experimentovanie s rôznymi modelmi a dátovými vstupmi.

6.6 Implementácia architektúr modelov

Na základe návrhu riešenia boli implementované dve hlavné kategórie modelov:

- vlastná transformerová architektúra,
- predtrénované modely z knižnice `HuggingFace`.

Obe kategórie sú navrhnuté tak, aby zdieľali rovnaké rozhranie pre trénovanie, logovanie a hodnotenie. Tým sa zjednodušuje experimentovanie a minimalizuje duplicita v kóde.

6.6.1 Vlastná transformerová architektúra

Vlastný model je implementovaný v súbore `src/models/custom_transformer.py` a pozostáva z nasledujúcich komponentov:

- **Trieda `CustomTransformerClassifier`**
  - Definuje architektúru modelu založenú na transformerovom enkóderi.
  - Obsahuje embedding vrstvu pre tokeny a pozície, viacvrstvový `TransformerEncoder` a výstupný lineárny klasifikátor.
- **Trieda `CustomHFWrapper`**
  - Zabezpečuje kompatibilitu s výstupným rozhraním `SequenceClassifierOutput` z knižnice `HuggingFace`.
  - Umožňuje integráciu vlastného modelu do trénovacieho cyklu bez potreby ďalších úprav.
- **Tokenizér `CustomTokenizer` (`src/tokenizers/custom_tokenizer.py`)**
  - Implementuje tri stratégie tokenizácie označené ako: `char`, `ngram` a `pretrained-tokenizer`.
  - Podporuje tvorbu vlastného slovníka z trénovacích dát.
  - Parametrizuje sa prostredníctvom konfiguračných súborov.
- **Konfigurácia**
  - Hlavná konfigurácia je definovaná v súbore `custom_config.yaml`.
  - Dodatočné parametre ako tokenizačná stratégia, veľkosť n-gramov alebo počet výstupných tried sa definujú v YAML súboroch v priečinku `configs/model/`.
- **Spustenie trénovania**
  - Implementované v `custom_architecture_experiment.py`.
  - Skript načíta konfiguráciu pomocou `Hydra`, pripraví dataset a tokenizér, zostaví model a spustí trénovanie.

6.6.2 Predtrénované transformerové modely

Predtrénované architektúry sú využívané pomocou knižnice `transformers` od `HuggingFace`.

- **Načítanie modelu**
  - Realizuje sa prostredníctvom funkcie `get_transformer_model` v súbore `src/models/transformer_model_factory.py`.
  - Používa rozhranie `AutoModelForSequenceClassification` na základe názvu uvedeného v konfiguračnom súbore.
- **Tokenizácia**
  - Využíva predtrénované tokenizéry cez `AutoTokenizer.from_pretrained(...)`.
  - Nevyžaduje manuálne budovanie slovníka.
- **Konfigurácia**
  - Hlavná konfigurácia je definovaná v súbore `pretrained_config.yaml`.
  - Podobne ako pri vlastnej architektúre sú konfigurácie dodatočných parametrov príslušných predtrénovaných modelov umiestnené v YAML súboroch v priečinku `configs/model/`.
- **Spustenie trénovania**
  - Vykonáva sa pomocou skriptu `experiments/run_experiment.py`.
  - Možné je spustiť buď jeden experiment (`pretrained_config.yaml`), alebo sériu experimentov (napr. `rdap_dns_geo.yaml` a pod.).

6.7 Ladenie hyperparametrov

Ladenie hyperparametrov predstavuje dôležitý krok v procese optimalizácie modelov. V tejto implementácii bola použitá knižnica **Ray Tune**[9], ktorá umožňuje efektívne a paralelné prehľadávanie priestoru hyperparametrov. Na rozdiel od statického prehľadávania priestoru, aké ponúka napríklad *grid search*, Ray Tune implementuje adaptívne algoritmy. Tie na základe priebežne zbieraných metrík upravujú stratégiu výberu hyperparametrov s cieľom efektívne konvergovať k optimálnym nastaveniam.

Hyperparametre, ktoré boli počas ladenia optimalizované, sú uvedené v tabuľke 5.4 v predchádzajúcej kapitole. Vyhľadávanie bolo implementované v priečinku `experiments/`, konkrétne v súbore `hyperparameter_search.py` . Každá konfigurácia bola hodnotená samostatným trénovaním modelu s nasledujúcim nastavením:

- bolo spustených 20 samostatných experimentov (označovaných ako `trials`),
- každý model bol trénovaný počas 3 epoch,
- ako vyhodnocovacia metrika bolo použité **skóre F1**.

[9] `https://docs.ray.io/en/latest/tune/`

Tento počet epoch bol zvolený zámerne nízky. Cieľom bolo rýchlo odhadnúť potenciál danej konfigurácie, pričom predchádzajúce pozorovania ukázali, že modely zvyčajne konvergujú už po niekoľkých iteráciách a zatiaľ nevykazujú známky preučenia. Použitie Ray Tune zároveň umožnilo:

- efektívne využitie viacerých GPU zariadení v paralelnom režime,
- výrazné zrýchlenie procesu ladenia,
- jednoduché prepojenie s `Trainer` API z knižnice `transformers` bez potreby implementácie vlastného trénovacieho cyklu.

6.8 Trénovanie modelov

Implementácia procesu trénovania sa nachádza v súbore `src/training/trainer.py`. Trénovanie modelov je postavené na knižniciach `PyTorch` a `Transformers`, ktoré poskytujú komplexné nástroje na definovanie trénovacieho cyklu, spracovanie vstupov a vyhodnocovanie výkonnosti modelov. V procese trénovania boli využité viaceré prístupy a komponenty s cieľom zabezpečiť efektívne škálovanie, stabilitu učenia a automatizáciu riadenia modelu:

- **Distribuované trénovanie pomocou triedy** `DistributedDataParallel`
  Model je paralelne spustený na viacerých dostupných GPU zariadeniach pomocou triedy `torch.nn.parallel.DistributedDataParallel`. Každá grafická karta pracuje s vlastnou časťou dát a po každej iterácii dochádza k synchronizácii váh modelu. Tento prístup umožňuje efektívne využiť výpočtový výkon viacerých grafických kariet a dosiahnuť lineárne škálovanie bez zníženia presnosti.
- **Rovnomerné rozdelenie dát prostredníctvom triedy** `DistributedSampler`
  Trieda `torch.utils.data.DistributedSampler` zabezpečuje automatické rozdelenie trénovacích aj validačných dát medzi jednotlivé GPU zariadenia tak, aby každé z nich spracovávalo unikátnu podmnožinu bez prekrývania. Tým sa zvyšuje efektivita trénovania a odstraňuje potreba manuálneho zásahu pri príprave dát.
- **Optimalizácia trénovania modelu s využitím optimalizátora** `AdamW`
  Trénovanie modelu je riadené optimalizátorom `AdamW` [25], ktorý je vhodný najmä pri trénovaní transformerových architektúr vďaka svojej schopnosti efektívne regulovať veľkosti váh modelu. Tento prístup pomáha predchádzať nestabilnému správaniu počas trénovania a prispieva k lepšej konvergencii.
- **Dynamické riadenie rýchlosti učenia pomocou plánovača** `get_scheduler`
  Rýchlosť učenia (learning rate) je riadená plánovačom z knižnice `transformers`, ktorý umožňuje definovať zahrievaciu fázu (warm-up) a následné postupné znižovanie rýchlosti učenia. Tento prístup podporuje stabilnejšiu konvergenciu modelu počas trénovania.
- **Vyhodnocovanie výkonu modelu a zaznamenávanie priebežných výsledkov**
  Po každej epoche sa model automaticky vyhodnocuje na validačnej množine. Zaznamenávajú sa metriky ako tréningová a validačná strata, presnosť, precíznosť, recall a skóre F1, ktoré sa ukladajú do CSV súboru. Tieto výsledky poskytujú spätnú väzbu pre analýzu modelu a slúžia ako kritérium pre ukladanie najlepšej verzie modelu.

- **Riadenie trvania trénovania a včasné zastavenie (early stopping)**
  Trénovanie je obmedzené na maximálne **20** epôch. Ak sa validačná metrika nezlepší počas 3 po sebe nasledujúcich epôch, trénovanie sa automaticky ukončí. V prípade zlepšenia sa model uloží ako `*_BEST.pt`. Po skončení trénovania sa bez ohľadu na výkon uloží aj finálna verzia modelu ako `*_FINAL.pt`.

6.9 Implementácia vyhodnocovania výsledkov

Vyhodnocovanie výkonnosti modelov je realizované prostredníctvom dvojice Jupyter notebookov umiestnených v priečinku `evaluation/`:

- `architectures_comparison.ipynb`
  Slúži na porovnanie rôznych architektúr na základe ich uložených váh a výstupných logovacích súborov. Obsahuje vizualizácie priebehu tréningovej a validačnej straty, presnosti a ďalších metrík.

- `training_eval.ipynb`
  Poskytuje detailné vyhodnotenie jedného konkrétneho modelu. Zameriava sa na analýzu nesprávnych predikcií a interpretáciu rozhodnutí modelu pomocou vhodných vizualizácií.

- `confusion_matrices_and_roc.ipynb`
  Generuje matice zámen (confusion matrices) a ROC krivky pre každý model, čím poskytuje pohľad na správanie klasifikátora pri rôznych prahoch rozhodovania.

- `interpretability.ipynb`
  Vizualizuje prínos jednotlivých tokenov na rozhodovanie modelu pomocou techniky *Integrated Gradients*. Umožňuje analyzovať, ktoré časti vstupu mali najväčší vplyv na predikciu.

- `classification_speed.py`
  Skript určený na meranie rýchlosti klasifikácie, ktorý vyhodnocuje priemerný čas spracovania jednej domény modelom. Vhodný na porovnanie efektivity rôznych architektúr z hľadiska výkonu.

- `model_sizes.py`
  Pomocný skript na výpis veľkostí jednotlivých modelov v pamäti. Umožňuje kvantifikovať pamäťové nároky rôznych architektúr.

Notebooky boli zvolené pre svoju interaktivitu a flexibilitu pri analýze výsledkov. Umožňujú rýchlu manipuláciu s dátami, opätovné spúšťanie jednotlivých častí bez potreby celkového pretrénovania a priame doplnenie vizualizácií podľa potreby. Na generovanie grafov je použitá knižnica `matplotlib`. Výsledky týchto vyhodnotení sú prezentované v nasledujúcej kapitole.

Kapitola 7

Experimentálne zhodnotenie a interpretácia výsledkov

Táto kapitola predstavuje výsledky experimentálneho overenia vytvorených modelov určených na detekciu škodlivých domén, pričom priamo nadväzuje na návrh riešenia uvedený v kapitole 5. V úvode kapitoly sú stručne definované použité hodnotiace metriky a rámec hodnotenia výsledkov. Nasleduje prezentácia dvoch hlavných experimentálnych fáz, ktorých cieľom je systematické vyhodnotenie navrhnutého prístupu:

- **Prvá fáza** je zameraná na výber najvhodnejšej transformerovej architektúry pre jednotlivé kategórie škodlivých domén (DGA, malware, phishing). Táto fáza pozostáva z nasledujúcich bodov:
  - **ladenia hyperparametrov** vybraných reprezentatívnych architektúr rôznych veľkostí,
  - **vyhodnotenia rôznych tokenizačných stratégií** pri vlastnej navrhnutej architektúre,
  - **finálneho porovnania** vlastnej architektúry s najlepšou tokenizačnou stratégiou voči súboru predtrénovaných modelov, s dôrazom na výkonnosť a efektívnosť modelov.
- **Druhá fáza** následne využíva optimálne architektúry identifikované v prvej fáze na experimentálne overenie prínosu jednotlivých dátových množín. Konkrétne ide o nasledujúce typy údajov:
  - doménové mená (pre všetky kategórie škodlivých domén),
  - RDAP dáta (pre malvér a phishing),
  - DNS dáta (pre malvér a phishing),
  - geografické údaje získané z IP adries (pre malvér a phishing).

  V rámci tejto fázy sú výsledky interpretované prostredníctvom klasických metrík, priebehu trénovania, grafických metód (Confusion matrix, ROC krivky) a vysvetliteľnosti modelov (Integrated Gradients).

Záver kapitoly tvorí sumarizácia všetkých experimentálnych výsledkov v prehľadnej porovnávacej tabuľke, ktorá okrem základných metrík (napríklad skóre F1) poskytuje aj doplnkové informácie, ako sú veľkosť modelov, trénovací čas a rýchlosť klasifikácie. Na základe týchto zistení sú potom diskutované možné odporúčania a ďalšie smerovanie výskumu.

7.1 Porovnávací rámec

Hodnotiace metriky slúžia na meranie presnosti, spoľahlivosti a efektivity vytvorených modelov. Aby bolo možné spravodlivo porovnať výkonnosť jednotlivých modelov, je potrebné dodržať konzistentné podmienky testovania a hodnotenia. Porovnanie sa uskutočňuje na základe 3 hlavných skupín metrík: výkonnostných, efektívnych a grafických. grafických.

7.1.1 Výkonnostné metriky

Výkonnostné metriky hodnotia presnosť a spoľahlivosť modelov pri predikcii. Tieto metriky poskytujú prehľad o schopnosti modelov správne klasifikovať domény na základe ich charakteristík [50]. Pre hodnotenie výkonnosti modelu budú použité nasledujúce metriky:

- **Presnosť (accuracy):** Miera správne klasifikovaných príkladov v rámci celkového počtu testovacích vzoriek. Vyjadruje celkový výkon modelu, avšak pri nevyvážených dátach (napr. viac legitímnych ako škodlivých domén) môže byť zavádzajúca.

$$\text{Presnosť} = \frac{\text{Počet správnych predikcií}}{\text{Celkový počet vzoriek}}$$

- **Precíznosť (Precision):** Podiel správne identifikovaných škodlivých domén medzi všetkými doménami označenými ako škodlivé. Dôležitá pri minimalizovaní falošne pozitívnych predikcií.

$$\text{Precíznosť} = \frac{\text{Skutočne škodlivé}}{\text{Skutočne škodlivé} + \text{Falošne pozitívne}}$$

- **Úplnosť (Recall):** Podiel správne identifikovaných škodlivých domén medzi všetkými skutočne škodlivými. Táto metrika je kritická pri minimalizovaní falošne negatívnych predikcií.

$$\text{Citlivosť} = \frac{\text{Skutočne škodlivé}}{\text{Skutočne škodlivé} + \text{Falošne negatívne}}$$

- **skóre F1:** Táto metrika hodnotí schopnosť modelu správne klasifikovať rôzne typy škodlivých domén. Harmonický priemer precíznosti a citlivosti, poskytuje vyvážené hodnotenie modelu v prípade nevyvážených tried.

$$\text{skóre F1} = 2 \cdot \frac{\text{Precíznosť} \cdot \text{Citlivosť}}{\text{Precíznosť} + \text{Citlivosť}}$$

7.1.2 Efektívne metriky

Efektívne metriky hodnotia, ako rýchlo a efektívne dokáže model spracovávať a klasifikovať údaje, čo je kľúčové pri nasadzovaní v reálnych systémoch, kde sa spracúva veľké množstvo domén.

- **Čas trénovania:** Meria dobu potrebnú na trénovanie modelu na tréningovom súbore až do dosiahnutia optimálneho výkonu. Tento údaj je dôležitý pri hodnotení praktickosti modelu, najmä ak je potrebné model často aktualizovať na základe nových dát.

- **Rýchlosť inferencie:** Hodnotí čas potrebný na klasifikáciu jednej domény počas prevádzky modelu. Táto metrika je kľúčová v systémoch, ktoré vyžadujú spracovanie veľkých objemov domén v reálnom čase.
- **Veľkosť modelu:** Posudzuje pamäťové a úložiskové požiadavky modelu. Menšie modely sú výhodnejšie pre nasadzovanie v prostrediach s obmedzenými zdrojmi, napríklad na zariadeniach s nízkym výkonom alebo v cloude s optimalizovanými nákladmi.

Tieto metriky umožňujú komplexné hodnotenie modelov, pričom poskytujú pohľad na ich presnosť pri klasifikácii, schopnosť pracovať s rôznymi typmi škodlivých domén a efektivitu v kontexte reálnych systémov. Takéto vyhodnotenie pomáha identifikovať model, ktorý najlepšie spĺňa potreby nasadenia, a zároveň upozorňuje na kompromisy medzi presnosťou a efektivitou.

7.1.3 Grafické metódy hodnotenia modelov

Na lepšie pochopenie správania modelu slúžia vizualizačné nástroje, ktoré ilustrujú jeho rozhodnutia v rôznych situáciách. V experimentoch boli využité najmä:

- **Confusion matrix** – tabuľka zobrazujúca, koľko legitímnych a škodlivých domén bolo správne alebo nesprávne klasifikovaných. Pomáha identifikovať typy chýb, ktoré model robí.
- **ROC krivka (Receiver Operating Characteristic)** – grafické zobrazenie pomeru zachytených škodlivých domén ku falošným poplachom pri rôznych prahoch rozhodovania.

Toto rozdelenie metrík poskytuje komplexný pohľad na kvalitu, efektivitu aj správanie modelov a tvorí základ pre ich porovnanie v ďalších častiach kapitoly.

7.2 Výber optimálnej architektúry

Táto fáza experimentov je zameraná na hľadanie optimálnej transformerovej architektúry pre každú kategóriu škodlivých domén (DGA, malvér, phishing). Vzhľadom na širokú škálu dostupných modelov bolo najskôr realizované ladenie hyperparametrov, aby sa identifikovali najlepšie nastavenia pre reprezentatívne architektúry, líšiace sa svojou veľkosťou (počtom parametrov a vrstiev). Táto optimalizácia bola vykonaná nad dátovou množinou doménových mien, ktoré vďaka kompaktnej forme umožňujú efektívne trénovanie a zároveň poskytujú dostatočnú informačnú hodnotu na spoľahlivú klasifikáciu. Výber týchto architektúr, ako aj metodika ladenia, boli podrobne popísané v návrhu riešenia (viď kapitola 5). Pre pripomenutie, do experimentov boli zaradené nasledovné modely:

- **prajjwal1/bert-tiny** – najmenšia architektúra,
- **bert-small** – stredne veľká architektúra,
- **distilbert-base-uncased** – najväčšia architektúra.

Nasledujúca tabuľka 7.1 sumarizuje najlepšie dosiahnuté výsledky ladenia hyperparametrov pre model `prajjwal1/bert-tiny` naprieč všetkými kategóriami škodlivých domén. Pre

danú konfiguráciu sú uvedené hodnoty metriky F1 a špecifické nastavenia trénovania dosiahnuté po troch epochách.

Hyperparameter	**DGA**	**Malware**	**Phishing**
skóre F1	0.96776	0.84474	0.905569
Rýchlosť učenia	4.749975e-05	4.749975e-05	4.749975e-05
Veľkosť dávky	128	128	128
Weight decay	0.019872	0.019872	0.019872
Warmup ratio	0.0011044	0.0011044	0.0011044
LR Scheduler	lineárny	lineárny	lineárny

Tabuľka 7.1: Optimálne hyperparametre pre `prajjwal1/bert-tiny`

Z tabuľky 7.1 vyplýva, že optimálne nastavenia hyperparametrov boli rovnaké pre všetky tri kategórie škodlivých domén. To naznačuje, že zvolená architektúra dokáže efektívne pracovať s doménovými menami bez potreby špecifického prispôsobovania tréningového procesu pre jednotlivé klasifikačné úlohy. Rozdiely vo výsledkoch skóre `F1` medzi kategóriami teda súvisia s povahou samotných dát než s nevhodným nastavením modelu.

Model `bert-small` reprezentuje stredne veľkú architektúru. Tabuľka nižšie sumarizuje najlepšie dosiahnuté výsledky pre jednotlivé kategórie škodlivých domén vrátane zodpovedajúcich hyperparametrov.

Hyperparameter	**DGA**	**Malware**	**Phishing**
skóre F1	0.981057	0.871832	0.928795
Rýchlosť učenia	3.946213e-05	3.946213e-05	3.946213e-05
Veľkosť dávky	512	512	512
Weight decay	0.09267	0.09267	0.09267
Warmup ratio	0.145455	0.145455	0.145455
LR Scheduler	lineárny	lineárny	lineárny

Tabuľka 7.2: Optimálne hyperparametre pre `bert-small`

Aj pri architektúre `bert-small` sa potvrdilo, že optimálne hyperparametre sa medzi jednotlivými kategóriami škodlivých domén zhodujú. V porovnaní s menšou architektúrou `bert-tiny` však došlo k zlepšeniu skóre F1 vo všetkých troch kategóriách, čo naznačuje, že väčšia kapacita modelu umožňuje efektívnejšie učenie, keďže modely konvergovali rýchlejšie k lepším výsledkom.
Najväčšiu testovanú architektúru predstavuje `distilbert-base-uncased`. V nasledujúcej tabuľke sú uvedené výsledky optimalizácie hyperparametrov pre každú z kategórií domén s dôrazom na dosiahnutý výkon a efektivitu učenia.

Hyperparameter	DGA	Malware	Phishing
skóre F1	0.986028	0.88416	0.93706
Rýchlosť učenia	3.946213e-05	3.946213e-05	3.946213e-05
Veľkosť dávky	512	512	512
Weight decay	0.09267	0.09267	0.09267
Warmup ratio	0.145455	0.145455	0.145455
LR Scheduler	lineárny	lineárny	lineárny

Tabuľka 7.3: Optimálne hyperparametre pre `distilbert-base-uncased`

Aj pri architektúre `distilbert-base-uncased` možno pozorovať úplnú zhodu optimalizovaných hyperparametrov naprieč všetkými tromi kategóriami domén. Tieto hodnoty sú identické ako pri architektúre `bert-small`, čo naznačuje, že stredne veľké a veľké modely vykazujú podobné správanie počas učenia. Zároveň možno vidieť mierne zlepšenie skóre F1, čo poukazuje na potenciál vyššej modelovej kapacity. Skutočnosť, že rovnaké nastavenia fungujú konzistentne medzi rôznymi architektúrami aj úlohami, predstavuje výhodu v podobe zjednodušeného vývoja, keďže odpadá potreba individuálneho ladenia pre každú kombináciu modelu a úlohy.

Na záver možno doplniť, že náročnosť ladenia hyperparametrov sa prejavila aj v potrebnom čase trénovania jednotlivých architektúr. Priemerný čas na jednu kategóriu domén bol približne **24 minút** pre `prajjwal1/bert-tiny`, **1 hodina a 41 minút** pre `bert-small` a až **4 hodiny a 2 minúty** pre `distilbert-base-uncased`. Hoci tieto časy nie sú zanedbateľné, investícia do systematického ladenia je výhodná. V porovnaní s často využívaným prístupom, pri ktorom sa hyperparametre preberajú z iných štúdií alebo sa stanovujú intuitívne, poskytuje táto metóda väčšiu istotu, že konfigurácia je dobre prispôsobená konkrétnemu zadaniu a dátam. Nemožno síce tvrdiť, že nájdené hodnoty sú globálne optimálne, no vzhľadom na obmedzený počet epoch (len 3) a rozumný kompromis medzi rozsahom vyhľadávania a výpočtovou náročnosťou ide o efektívne a kvalitné nastavenia, ktoré budú slúžiť ako základ pre ďalšie experimenty.

7.3 Vyhodnotenie tokenizačných stratégií (vlastná architektúra)

Po optimalizácii hyperparametrov bola analyzovaná účinnosť rozličných tokenizačných stratégií s využitím špecifickej modelovej architektúry. Cieľom tejto časti je identifikovať kombináciu, ktorá zabezpečuje najvyššiu klasifikačnú presnosť pre jednotlivé kategórie škodlivých domén (DGA, malware, phishing). Ako už bolo uvedené v návrhu riešenia (viď kapitolu 5), pri navrhovaní vlastnej architektúry boli zohľadnené nasledujúce tokenizačné stratégie:

- **N-gram tokenizácia (3-gramy)** – segmentácia doménových mien na prekrývajúce sa trojice znakov, zachytávajúce lokálne štruktúry,
- **znaková tokenizácia** – lineárne rozdelenie domén na trojice znakov bez ďalšej sémantickej interpretácie,

- **tokenizácia pomocou predtrénovaného tokenizéra** – využitie tokenizéra pre predtrénované transformerové modely, ktorý odráža segmentačné pravidlá naučené na veľkých textových korpusoch.

Prehľad výsledkov týchto stratégií z hľadiska skóre F1 je uvedený v tabuľke 7.4. Tréning pre každý variant tokenizácie prebiehal počas 20 epoch s konzistentnými hyperparametrami (keďže išlo o malý model, boli použité hyperparametre získané pri ladení architektúry `prajjwal1/bert-tiny`), aby bolo možné zabezpečiť férové porovnanie.

Tokenizačná stratégia	DGA	Malware	Phishing
N-gram (3-gram)	0.92115	0.76286	0.83862
Znaková	0.95683	0.78914	0.81922
Predtrénovaný tokenizér	0.94591	0.82092	0.86831

Tabuľka 7.4: Porovnanie skóre F1 pre rôzne tokenizačné stratégie (vlastná architektúra)

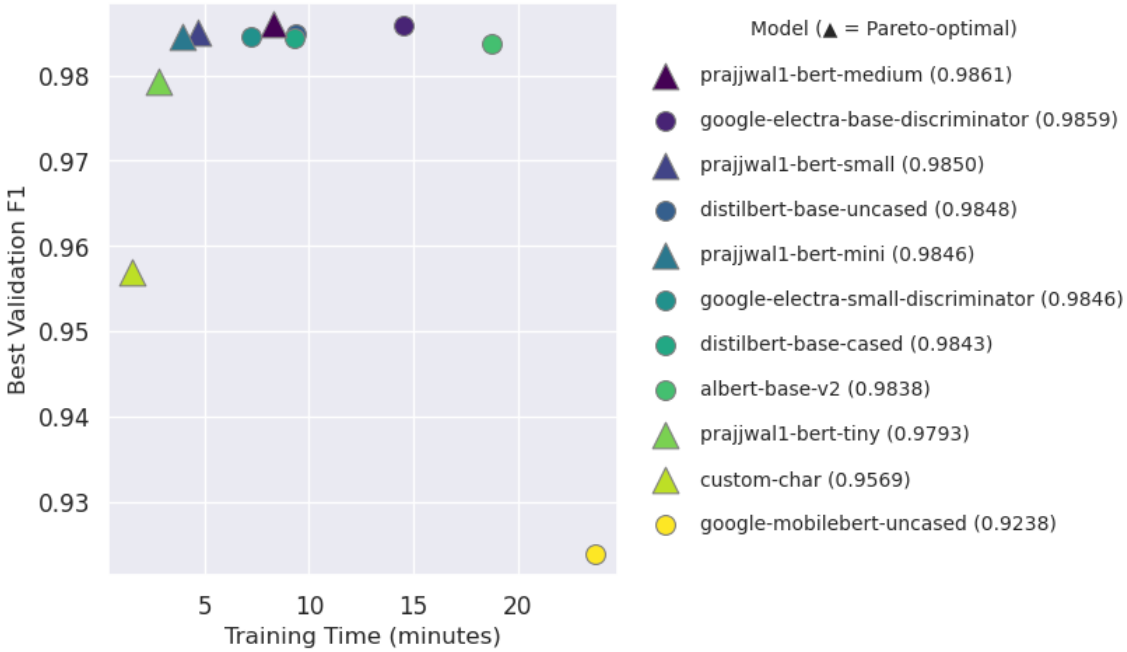
Z výsledkov uvedených v tabuľke 7.4 vyplýva, že optimálna tokenizačná stratégia závisí od konkrétnej kategórie škodlivých domén. Pre DGA domény dosiahla najvyššie skóre F1 znaková tokenizácia, čo možno pripísať schopnosti modelu zachytiť jemné štrukturálne rozdiely na úrovni jednotlivých znakov, ktoré sú pre túto kategóriu typické. V prípade malware domén sa najlepšie osvedčil predtrénovaný tokenizér, pravdepodobne vďaka tomu, že jeho subslovné jednotky lepšie vystihujú časti názvov opakujúce sa v tejto doméne. Pre phishingové domény boli rozdiely medzi stratégiami menej výrazné, pričom mierne najlepší výsledok dosiahol opäť predtrénovaný tokenizér. Výsledky ukazujú, že výber vhodnej tokenizačnej stratégie má významný vplyv na presnosť klasifikácie. Táto analýza umožnila identifikovať najvhodnejší spôsob reprezentácie vstupných dát pre vlastnú architektúru v rámci každej kategórie škodlivých domén, ktorý bol následne použitý v ďalšej fáze porovnania voči predtrénovaným transformerovým modelom.

7.4 Porovnanie vlastnej architektúry voči predtrénovaným modelom

V tejto časti je prezentované finálne porovnanie výkonu najlepšej vlastnej architektúry s optimálnou tokenizačnou stratégiou voči viacerým dostupným predtrénovaným transformerovým architektúram. Porovnanie bolo realizované formou tzv. Pareto grafu, ktorý umožňuje prehľadne identifikovať modely s najlepším kompromisom medzi klasifikačným výkonom a efektivitou (v tomto prípade reprezentovanou časom trénovania). Modely, ktoré sa nachádzajú na tzv. Pareto hranici, predstavujú optimálne voľby vzhľadom na oba porovnávané faktory.

7.4.1 Porovnanie architektúr pre detekciu DGA domén

Pre výber optimálnej architektúry v kategórii DGA domén bolo realizované porovnanie rôznych transformerových modelov z hľadiska ich klasifikačného výkonu (skóre F1) a efektivity (reprezentovanej dĺžkou trénovania v minútach). Výsledky sú vizualizované pomocou Pareto grafu 7.1, ktorý umožňuje prehľadne identifikovať modely poskytujúce najlepší kompromis medzi výkonom a výpočtovými nárokmi.

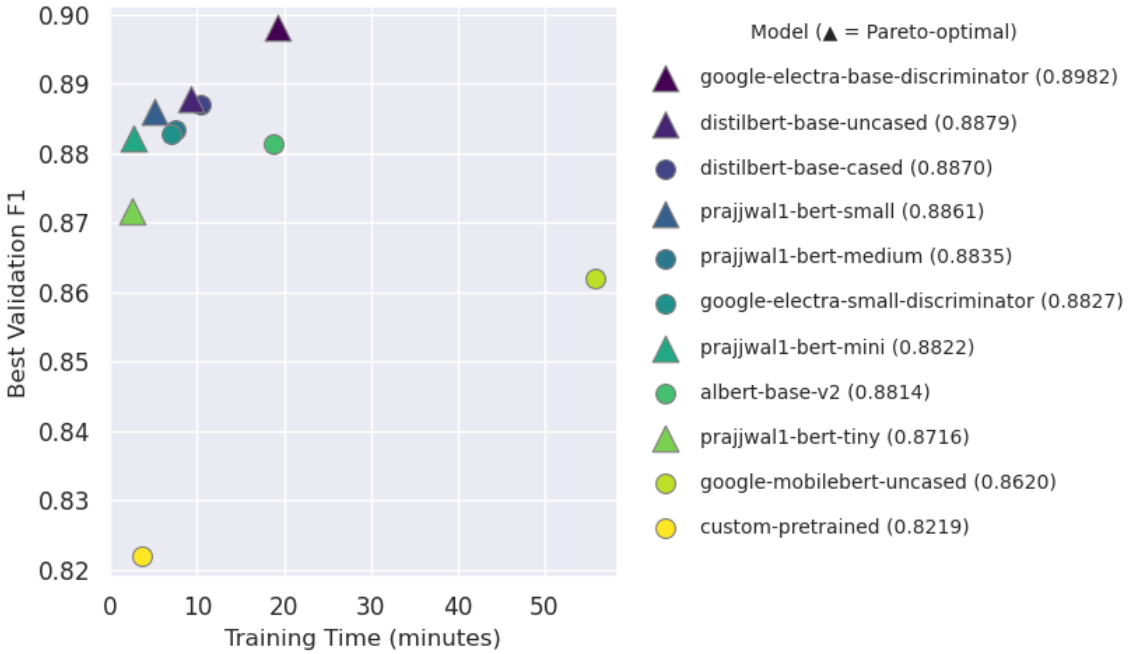


Obr. 7.1: Porovnanie modelov pre kategóriu DGA: skóre F1 vs. čas trénovania

Ako je z grafu zrejmé, najlepšiu rovnováhu medzi presnosťou a efektivitou dosiahli modely `prajjwal1/bert-medium` a `prajjwal1/bert-small`, ktoré predstavujú optimálny kompromis medzi presnosťou klasifikácie a trénovacím časom. Vlastná architektúra s najlepšou tokenizačnou stratégiou sa výkonom priblížila k týmto modelom, ale nie dostatočne. Z hľadiska efektivity sa však ukázala ako konkurencieschopná. Za najslabší model v tejto kategórii možno označiť `google-mobilebert-uncased`, ktorý dosiahol najnižšie skóre F1 napriek výrazne vyššej dobe trénovania. Na základe analýzy výkonu a výpočtovej náročnosti bol pre klasifikáciu DGA domén vybraný model `prajjwal1/bert-medium`.

7.4.2 Porovnanie architektúr pre detekciu malvérových domén

Rovnako ako v prípade DGA domén, aj pre kategóriu malvérových domén bolo uskutočnené porovnanie transformerových architektúr z hľadiska presnosti klasifikácie a trénovacej efektivity. Výsledky sú zobrazené v nasledujúcom grafe 7.2.

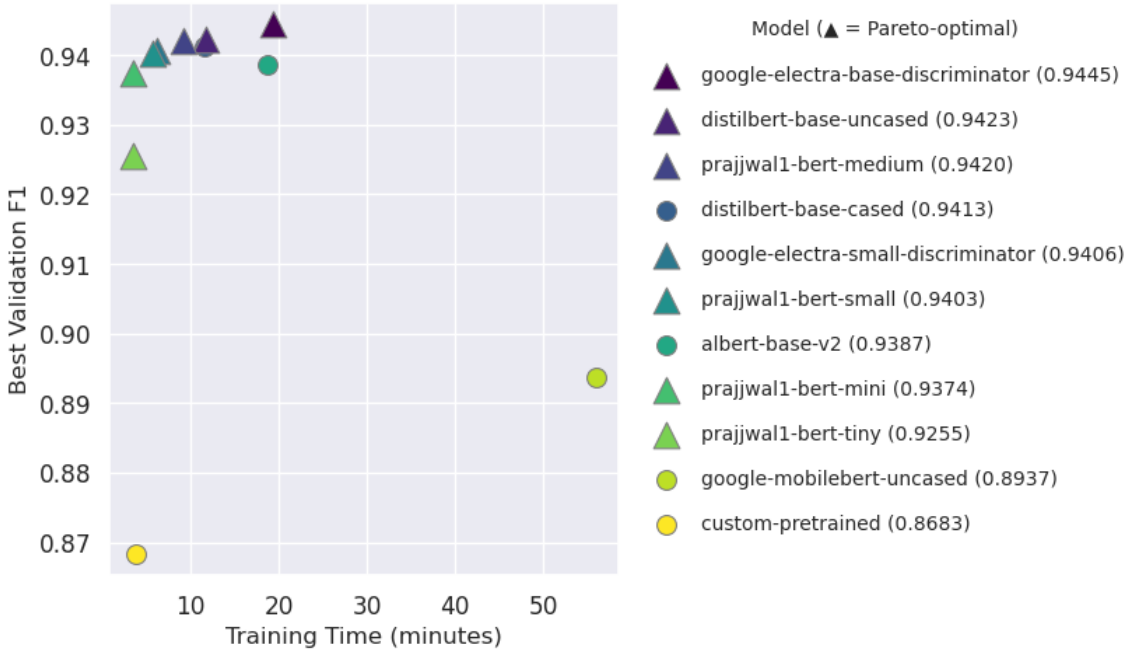


Obr. 7.2: Porovnanie modelov pre kategóriu Malware: skóre F1 vs. čas trénovania

Z grafu 7.2 možno identifikovať viacero modelov, ktoré dosahujú priaznivý pomer medzi klasifikačným výkonom a dĺžkou trénovania. Najlepšie výsledky preukázal model `electra-base-discriminator`, ktorý dosiahol výrazne vyššie skóre F1 než ostatné architektúry pri stále akceptovateľnej dĺžke trénovania. Vlastná architektúra s najlepšou tokenizačnou stratégiou dosiahla nižšie klasifikačné skóre ako najvýkonnejšie modely, no prekonala model `google-mobilebert-uncased` a zároveň si zachovala porovnateľný čas trénovania. Najmenej efektívnym modelom z predtrénovaných architektúr bol `google-mobilebert-uncased`, ktorý pri dlhšom trénovaní nedosiahol adekvátne výsledky. Na základe analýzy výkonnostných a časových charakteristík bol pre ďalšiu fázu zvolený model `electra-base-discriminator`.

7.4.3 Porovnanie architektúr pre detekciu phishingových domén

Pre kategóriu phishingových domén bolo opäť realizované porovnanie transformerových modelov z hľadiska ich klasifikačnej presnosti a trénovacej efektivity. Výsledky sú zobrazené na nasledujúcom grafe:



Obr. 7.3: Porovnanie modelov pre kategóriu Phishing: skóre F1 vs. čas trénovania

Z grafu vyplýva, že najvyššiu presnosť dosiahli modely `google-electra-base-discriminator` a `distilbert-base-uncased`, pričom druhý menovaný poskytol porovnateľný výkon pri nižšej výpočtovej náročnosti. Vlastná architektúra dosiahla najnižšie skóre F1 spomedzi všetkých hodnotených modelov. Na základe týchto výsledkov bol ako optimálna architektúra pre detekciu phishingových domén zvolený model `distilbert-base-uncased`, keďže rozdiel vo výkone oproti najpresnejšiemu modelu bol minimálny, avšak vykazoval takmer dvojnásobne nižšiu časovú náročnosť na trénovanie. Na záver prvej fázy experimentov boli identifikované najvhodnejšie modelové architektúry pre každú z hodnotených kategórií škodlivých domén. Výber vychádzal z kombinovaného hodnotenia klasifikačnej presnosti a výpočtovej efektivity modelov. Konkrétne:

- **DGA**: `distilbert-base-uncased`
- **Malware**: `google-electra-base-discriminator`
- **Phishing**: `distilbert-base-uncased`

Tieto architektúry boli následne použité ako základ pre druhú fázu experimentov, ktorá sa zameriava na vyhodnotenie vplyvu rôznych typov vstupných údajov na detekčnú výkonnosť modelov. V nasledujúcej časti je podrobne analyzovaný prínos jednotlivých dátových vrstiev (RDAP, DNS, geografické údaje) nad rámec doménových mien.

7.5 Vyhodnotenie modelov nad doménovými menami

V tejto časti začína druhá fáza vyhodnocovania modelov, ktorá sa sústreďuje na posúdenie ich výkonnosti pri klasifikácii na základe rôznych typov vstupných údajov. Úvodná fáza tejto analýzy je zameraná na modely trénované výlučne na doménových menách, a to pre

tri kategórie škodlivých domén: algoritmicky generované domény (DGA), domény spojené s malvérom a phishingové domény. Pre každý model bude vykonané hodnotenie podľa jednotnej schémy, ktorá zahŕňa:

- klasifikačné metriky,
- vizualizáciu priebehu trénovania,
- grafické metódy (confusion matrix a ROC krivka),
- interpretáciu rozhodovania modelu pomocou techniky Integrated Gradients.

Tento rámec umožňuje komplexné a konzistentné porovnanie jednotlivých modelov z hľadiska presnosti, správania počas učenia aj interpretovateľnosti výsledkov.

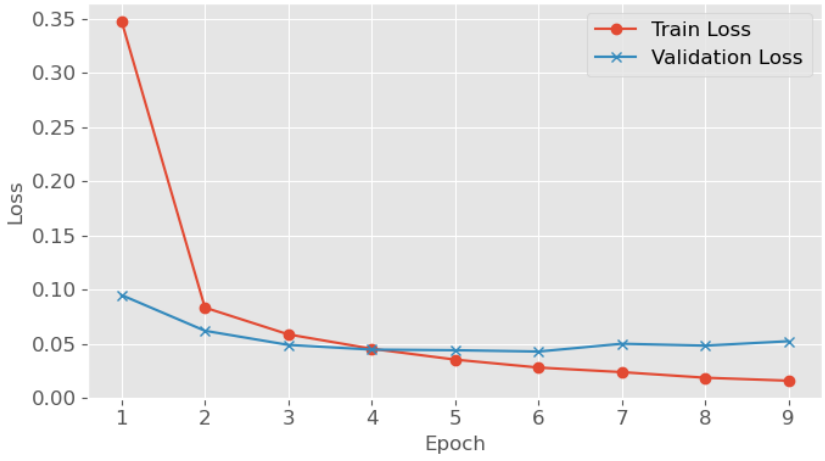
7.5.1 Klasifikátor DGA domén

Ako ukázali výsledky predchádzajúcej analýzy, pre klasifikáciu DGA domén bol vybraný model `prajjwal1/bert-medium`. Model dosiahol nasledujúce hodnoty klasifikačných metrík:

- **skóre F1:** 98,61%,
- **presnosť (accuracy):** 98,53%,
- **precíznosť (precision):** 98,39%,
- **úplnosť (recall):** 98,82%.

Dosiahnuté hodnoty naznačujú, že model si udržiava vysokú úroveň presnosti aj spoľahlivosti. Hodnota recall presahujúca 98% poukazuje na schopnosť modelu zachytiť väčšinu škodlivých domén, zatiaľ čo vysoká precision svedčí o tom, že len veľmi malé množstvo legitímnych domén bolo nesprávne označené ako DGA. Vyvážené F1-score potvrdzuje, že model dobre zvláda kompromis medzi týmito dvoma aspektmi.

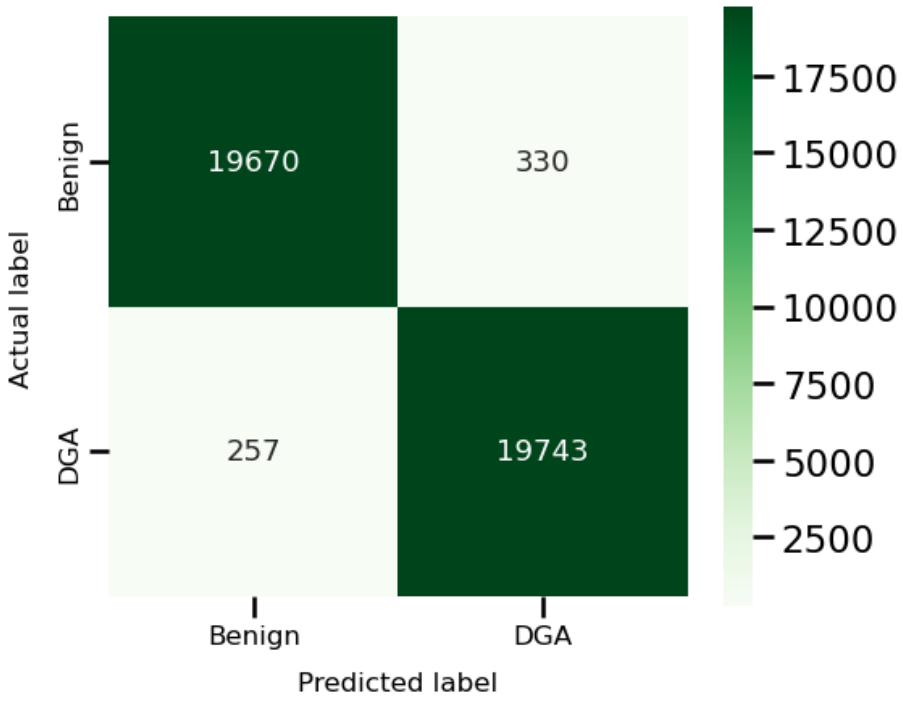
Hodnotenie číselných metrík poskytuje dôležitý, no len čiastočný pohľad na správanie modelu. Pre komplexnejšie zhodnotenie jeho spoľahlivosti je vhodné ďalej analyzovať priebeh učenia a overiť, či model nevykazuje známky pretrénovania alebo iných nežiaducich javov počas optimalizácie, čo ukazuje obrázok 7.4.



Obr. 7.4: Priebeh trénovacej a validačnej straty pri klasifikácii DGA domén

Z grafu 7.4 možno vyčítať, že model `prajjwal1/bert-medium` dosahuje pri klasifikácii DGA domén rýchlu a stabilnú konvergenciu. Trénovacia strata klesá strmo už počas prvých troch epochách, čo naznačuje, že model sa dokáže efektívne naučiť základné vzory prítomné v doménových menách. Súčasne s tým sa znižuje aj validačná strata, pričom medzi oboma krivkami zostáva len malý rozdiel, čo svedčí o dobrej schopnosti modelu generalizovať na nevidené dáta. Od šiestej epochy sa validačná strata prestáva znižovať a následne mierne narastá, zatiaľ čo trénovacia strata ďalej pozvoľna klesá. Tento trend naznačuje klesajúci prínos ďalšieho trénovania a potenciálne riziko pretrénovania. Z tohto dôvodu bol pri učení modelu použitý mechanizmus early stopping, ktorý sledoval vývoj validačnej straty a po troch epochách bez zlepšenia automaticky ukončil trénovanie.

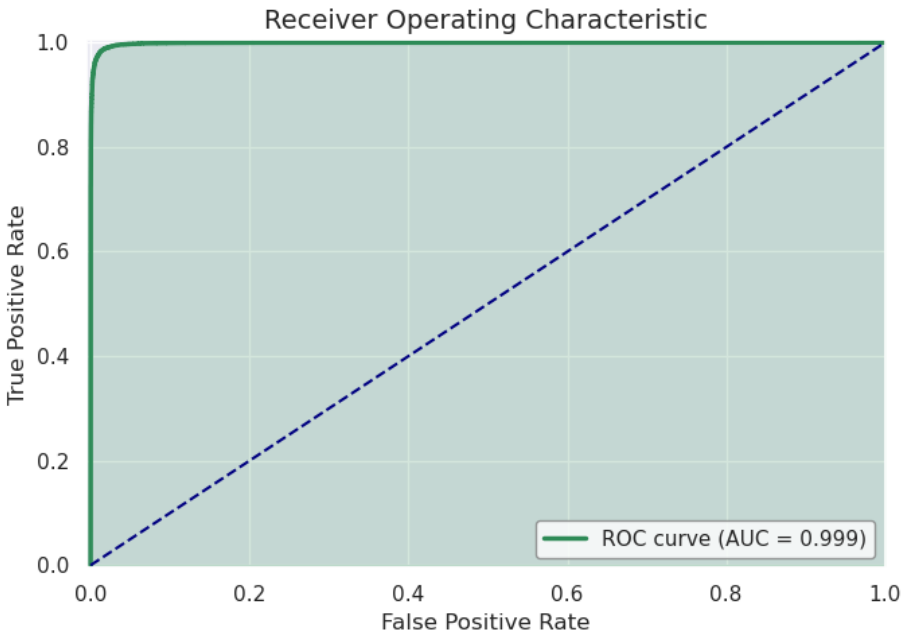
Priebeh trénovania potvrdil stabilitu modelu, ale pre podrobnejšie posúdenie jeho správania pri predikcii je vhodné analyzovať, ako často dochádza k správnej a nesprávnej klasifikácii jednotlivých prípadov. Confusion matrix, zobrazená na obrázku 7.5, poskytuje prehľad o tom, či model robí chyby skôr pri identifikácii legitímnych alebo škodlivých domén, a dopĺňa tak informácie získané z klasifikačných metrík.



Obr. 7.5: Confusion matrix pre klasifikáciu DGA domén

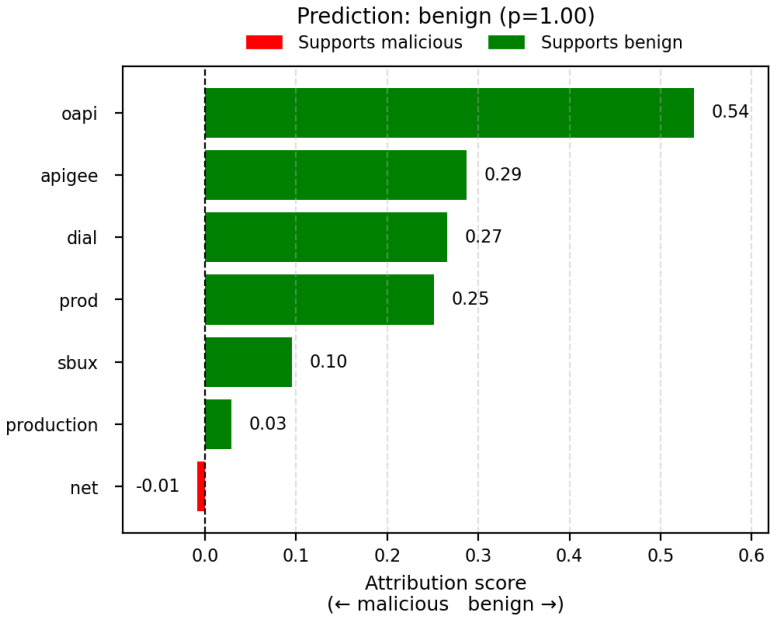
Z matice 7.5 vyplýva, že model dosahuje veľmi nízku mieru chýb na oboch stranách. Z 20 000 legitímnych domén bolo nesprávne označených len 330 prípadov, zatiaľ čo z 20 000 DGA domén uniklo detekcii iba 257 prípadov. Takéto rozloženie chýb potvrdzuje vysokú spoľahlivosť modelu pri označovaní legitímnych domén aj jeho schopnosť precízne odhaliť DGA domény bez výrazného skreslenia žiadnej z tried.

Spoľahlivosť modelu ďalej potvrdzuje aj ROC krivka na obr. 7.6, ktorá má takmer ideálny priebeh. Krivka rýchlo stúpa k hodnote 1 na osi zachytených pozitívnych prípadov, pričom si udržiava nízku mieru nesprávnych klasifikácií legitímnych domén. Tento tvar krivky potvrdzuje, že model veľmi dobre rozlišuje medzi DGA a legitímnymi doménami.

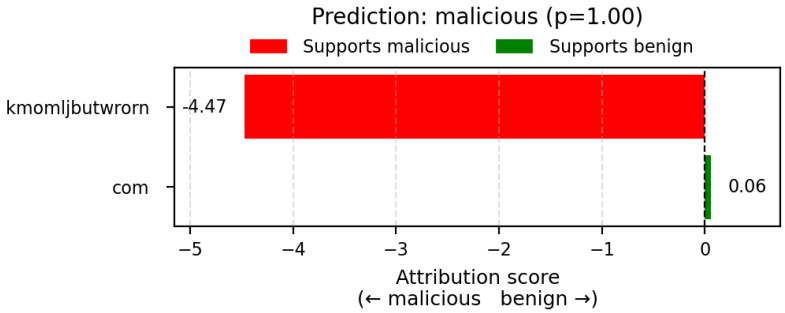


Obr. 7.6: ROC krivka pre klasifikáciu DGA domén

Pre ilustráciu interpretovateľnosti rozhodnutí modelu boli náhodne vybrané dve doménové mená z testovacej množiny. Jedna doména bola benignná a druhá patrila do kategórie DGA. Obrázky 7.7 a 7.8 znázorňujú atribučné skóre jednotlivých tokenov získané metódou Integrated Gradients. Zobrazené tokeny boli pred vizualizáciou upravené tak, aby predstavovali celé slová. Odstránili sa špeciálne znaky, spojili sa rozdelené časti slov a vynechali sa jednopísmenové výskyty. Zelené stĺpce podporujú predikciu ako benignnú, červené stĺpce ako škodlivú.



Obr. 7.7: Integrated Gradients pre náhodne vybranú benignnú doménu



Obr. 7.8: Integrated Gradients pre náhodne vybranú DGA doménu

Na obrázku 7.7 vidieť, že tokeny ako `"oap"`, `"apigee"` či `"dial"` majú najvyššie kladné skóre, čím najviac prispievajú k označeniu domény za legitímnu. Iba token `"net"` zaznamenáva mierne záporné skóre, no jeho vplyv je zanedbateľný. Naopak na obrázku 7.8 dominuje silné záporné skóre tokenu `"kmomljbutworn"`, ktoré jednoznačne ťahá predikciu smerom k DGA. Token `"com"` má mierne kladné skóre, čo naznačuje, že prítomnosť bežnej TLD sama osebe modelu nestačí na označenie domény za benignnú. Týmto spôsobom možno získať detailný pohľad na to, ktoré časti vstupu ovplyvňujú rozhodnutie modelu najvýraznejšie.

Model `prajjwal1/bert-medium` dosahuje pri detekcii DGA domén vysokú presnosť aj spoľahlivosť. Hodnota recall presahujúca 98% ukazuje, že model dokáže zachytiť takmer všetky škodlivé domény, zatiaľ čo vysoká precision potvrdzuje, že len malé množstvo legitímnych domén bolo označené nesprávne. Vyvážené skóre F1 naznačuje dobrý kompromis medzi citlivosťou a presnosťou. Priebeh trénovania ukázal stabilnú konvergenciu bez známok pretrénovania, čo potvrdzuje aj použitie mechanizmu early stopping. Confusion matrix odhalila veľmi nízky podiel chybných klasifikácií a ROC krivka potvrdila schopnosť modelu spoľahlivo rozlišovať medzi legitímnymi a DGA doménami. Interpretácia pomocou metódy Integrated Gradients navyše ukázala, že model pri rozhodovaní vyhodnocuje zmysluplné časti doménového mena, čo zvyšuje dôveru v jeho výstupy.

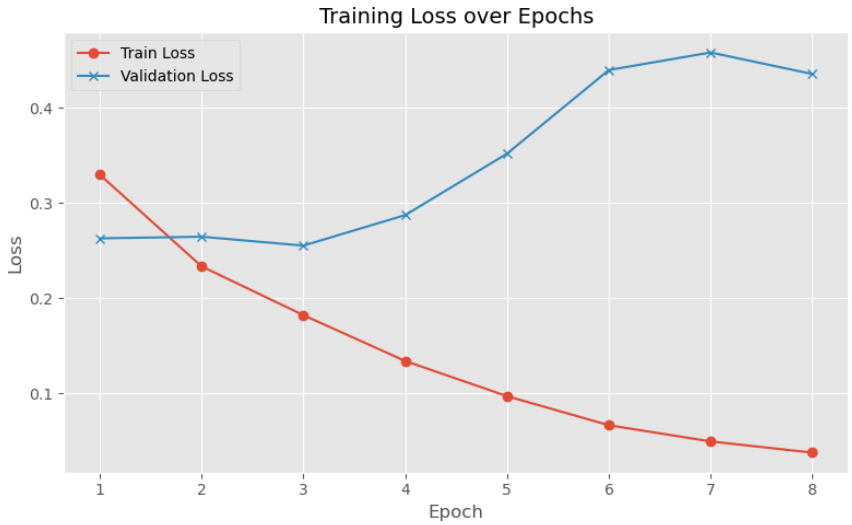
7.5.2 Klasifikátor malvérových domén

Pre klasifikáciu malvérových domén bol na základe predchádzajúcich experimentov zvolený model `google/electra-base-discriminator`. Model dosiahol nasledujúce hodnoty klasifikačných metrík:

- **skóre F1:** 89,71%,
- **presnosť (accuracy):** 89,52%,
- **precíznosť (precision):** 88,39%,
- **úplnosť (recall):** 91,06%.

Výsledky naznačujú, že model si pri detekcii malvérových domén udržiava solídny výkon. Hodnota recall nad 91% poukazuje na schopnosť modelu zachytiť veľkú väčšinu škodlivých prípadov, zatiaľ čo precision okolo 88% znamená, že menšia časť legitímnych domén bola klasifikovaná chybne. Vyvážené skóre F1 na úrovni 89,71% potvrdzuje, že model zvláda kompromis medzi týmito dvoma aspektmi pomerne dobre.

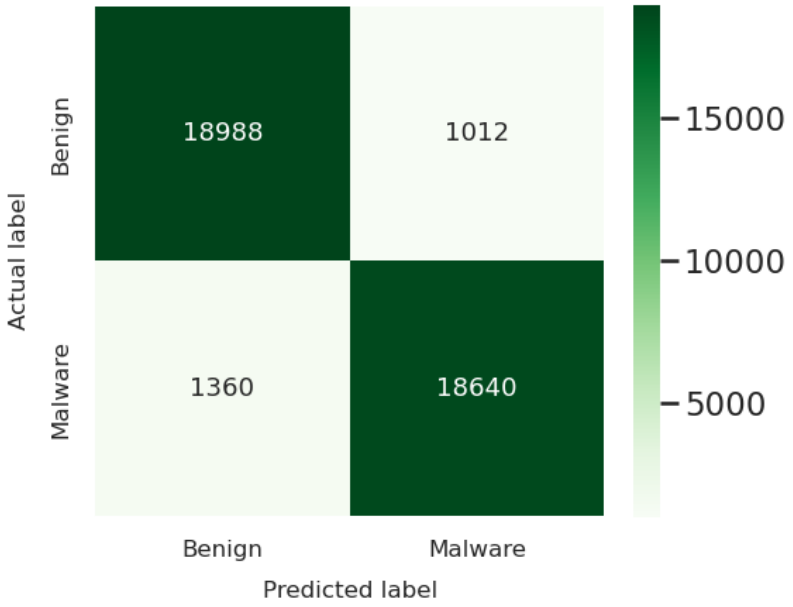
Pre lepšie pochopenie správania modelu počas učenia bol analyzovaný priebeh trénovacej a validačnej straty, znázornený na obr. 7.9.



Obr. 7.9: Priebeh trénovacej a validačnej straty pri klasifikácii malvérových domén

Z grafu možno pozorovať, že model síce vykazuje stabilný pokles trénovacej straty, avšak validačná strata od piatej epochy narastá, čo poukazuje na pretrénovanie. Trénovanie pokračovalo až do ôsmej epochy, pričom strata na validačnej množine dosiahla svoje maximum v siedmej epoche. Táto dynamika naznačuje, že výkon modelu na trénovacích dátach už prestával korelovať s výkonom na nevidených dátach. Tento jav je potrebné zohľadniť pri výbere vhodného momentu na zastavenie trénovania, napríklad pomocou mechanizmu early stopping.

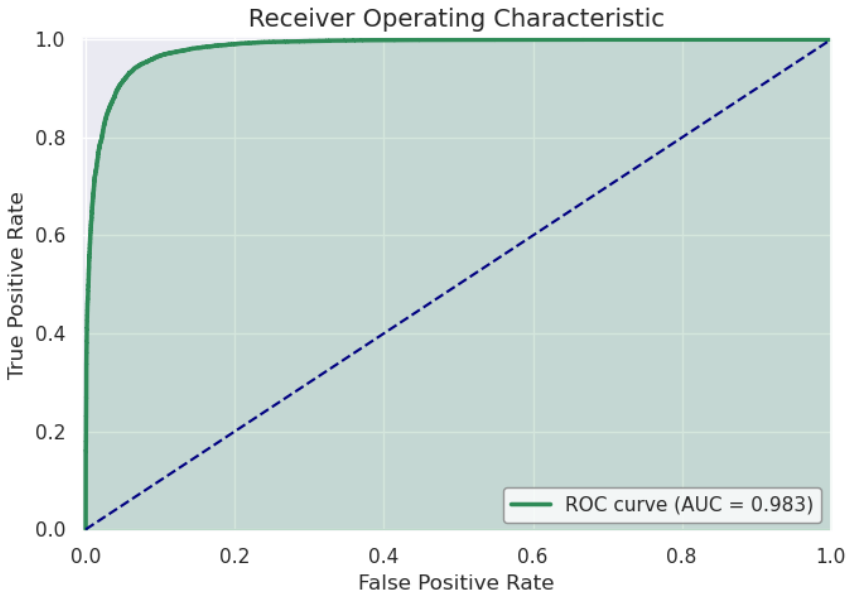
Doplnkový pohľad na kvalitu modelu ponúka confusion matrix na obr. 7.10, ktorá ukazuje rozloženie správnych a nesprávnych klasifikácií medzi legitímnymi a škodlivými doménami.



Obr. 7.10: Confusion matrix pre klasifikáciu malvérových domén

Z obrázku vyplýva, že z celkového počtu 20 000 legitímnych domén bolo chybne klasifikovaných 1 012 ako škodlivé, zatiaľ čo z 20 000 malvérových domén nebolo správne identifikovaných 1 360. To znamená, že falošne pozitívne prípady tvoria približne 5% a falošne negatívne okolo 6,8%. Napriek týmto chybám je celkové rozloženie predikcií vyvážené a model si zachováva stále pomerne vysokú spoľahlivosť.

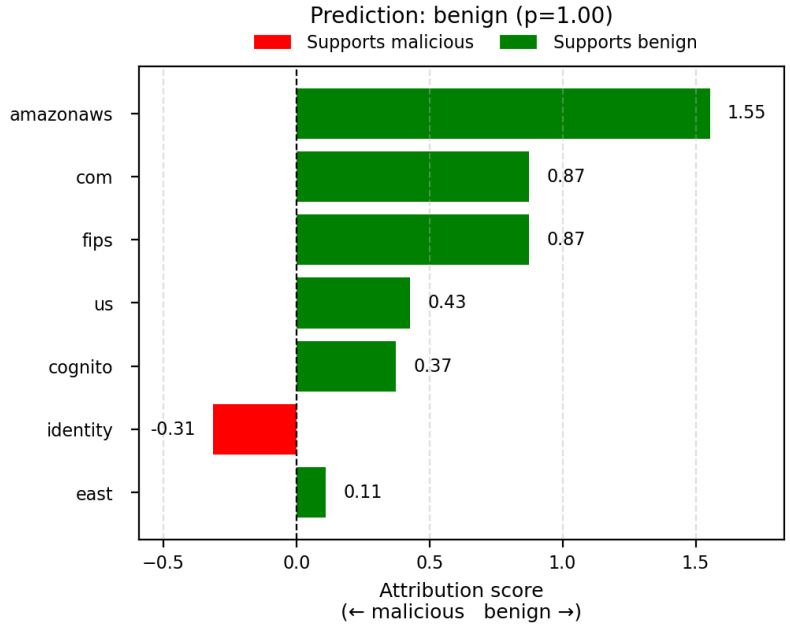
Spoľahlivosť modelu podporuje aj tvar ROC krivky na obr. 7.11, ktorá ukazuje vysoký podiel zachytených škodlivých domén pri relatívne nízkej miere nesprávnych označení legitímnych.



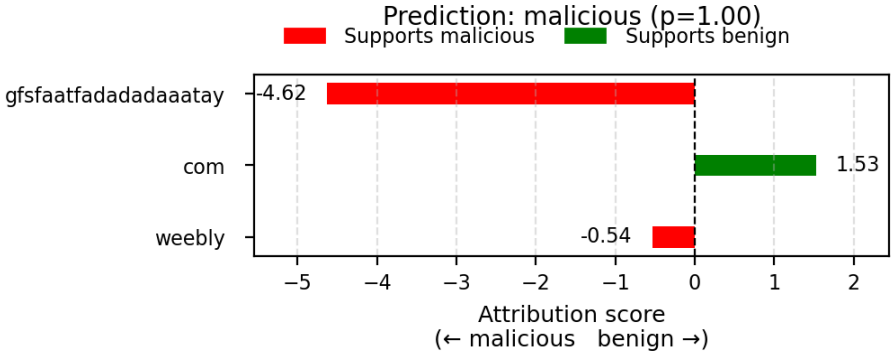
Obr. 7.11: ROC krivka pre klasifikáciu malvérových domén

Krivka má typický konkávny tvar, pričom sa rýchlo približuje k ľavému hornému rohu. To potvrdzuje, že model dokáže dobre rozlišovať medzi triedami aj pri rôznych nastaveniach prahových hodnôt.

Na záver boli opäť analyzované atribučné skóre pomocou metódy Integrated Gradients. Zobrazené tokeny boli pred vizualizáciou upravené tak, aby predstavovali celé slová. Odstránili sa špeciálne znaky, spojili fragmenty a ignorovali jednopísmenové výskyty.



Obr. 7.12: Integrated Gradients pre náhodne vybranú benignnú doménu (malware klasifikátor)



Obr. 7.13: Integrated Gradients pre náhodne vybranú škodlivú doménu (malware klasifikátor)

Na obrázku 7.12 vidieť, že model považuje tokeny `"amazonaws"`, `"com"` a `"fips"` za silné indikátory legitímnych domén. Záporné skóre tokenu `"identity"` je minimálne a neovplyvňuje rozhodnutie výrazne. Naopak, na obrázku 7.13 dominuje token `"gfsaatfadadadaaatay"`,

ktorý výrazne prispieva k označeniu domény ako škodlivej. Tento prípad ukazuje, že model dokáže identifikovať podozrivé znaky v doméne a reagovať na ne primerane.

Celkovo možno konštatovať, že model pre klasifikáciu malvérových domén dosahuje spoľahlivý výkon, pričom efektívne vyvažuje citlivosť aj presnosť. Napriek určitému pretrénovaniu na validačnej množine si model zachováva dobré rozlišovacie schopnosti, čo potvrdzujú numerické metriky, vizualizácie chýb aj interpretácia rozhodnutí.

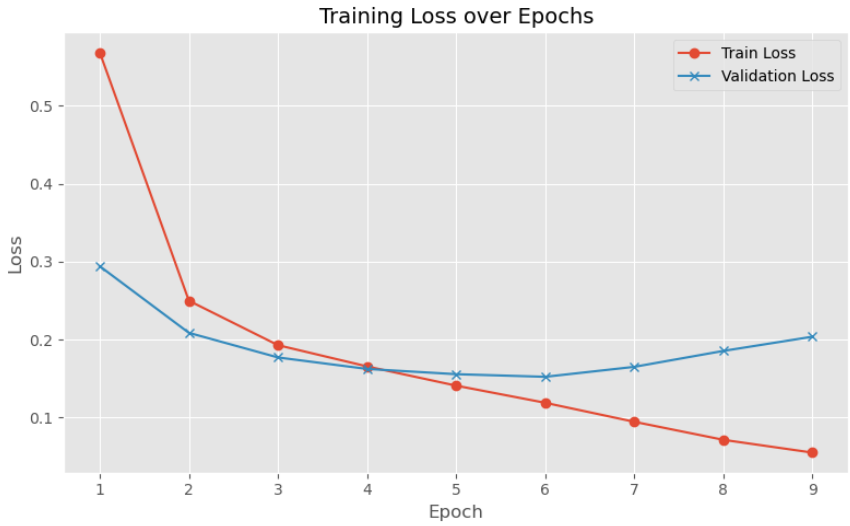
7.5.3 Klasifikátor phishingových domén

Na základe experimentálnej analýzy bol pre klasifikáciu phishingových domén vybraný model `distilbert-base-uncased`. Model dosiahol nasledujúce hodnoty klasifikačných metrík:

- **skóre F1:** 94,17%,
- **presnosť (accuracy):** 94,18%,
- **precíznosť (precision):** 94,80%,
- **úplnosť (recall):** 93,54%.

Model vykazuje veľmi silný výkon s vysokou úrovňou presnosti a vyváženou citlivosťou. Hodnota recall nad 93 % potvrdzuje schopnosť detekovať väčšinu phishingových domén, zatiaľ čo vysoká precision (takmer 95 %) svedčí o tom, že len veľmi malé množstvo legitímnych domén bolo označených nesprávne. Vyrovnané skóre F1 podčiarkuje, že model spoľahlivo zvláda kompromis medzi oboma aspektmi.

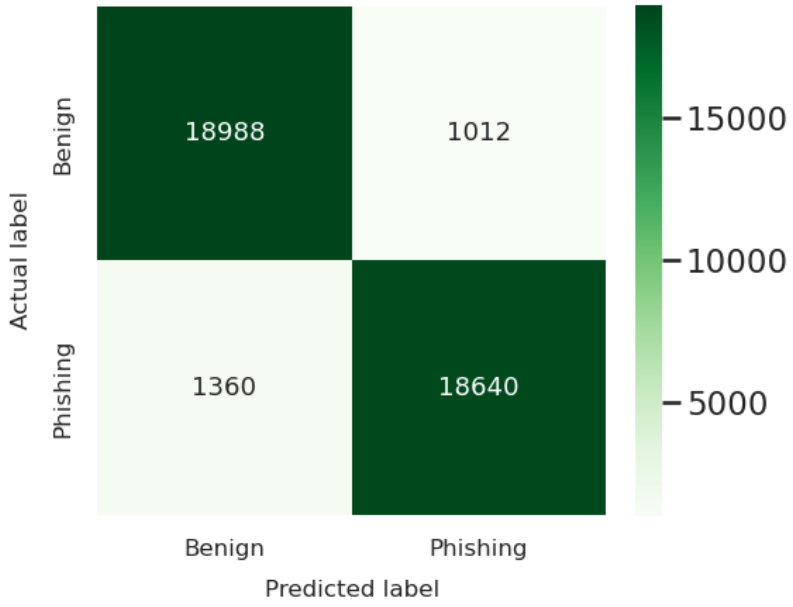
Dynamiku učenia znázorňuje graf na obr. 7.14, ktorý zobrazuje priebeh trénovacej a validačnej straty počas jednotlivých epoch.



Obr. 7.14: Priebeh trénovacej a validačnej straty pri klasifikácii phishingových domén

Z grafu možno vyčítať, že trénovacia aj validačná strata klesajú súčasne počas úvodných fáz učenia, pričom od štvrtej epochy sa rozdiel medzi nimi stabilizuje. Od siedmej epochy začína validačná strata mierne rásť, zatiaľ čo trénovacia naďalej klesá. Tento priebeh naznačuje začínajúce pretrénovanie, no rozdiel medzi stratami ostáva pomerne malý. Model si tak aj napriek dlhšiemu trénovaniu udržiava dobrú schopnosť generalizácie.

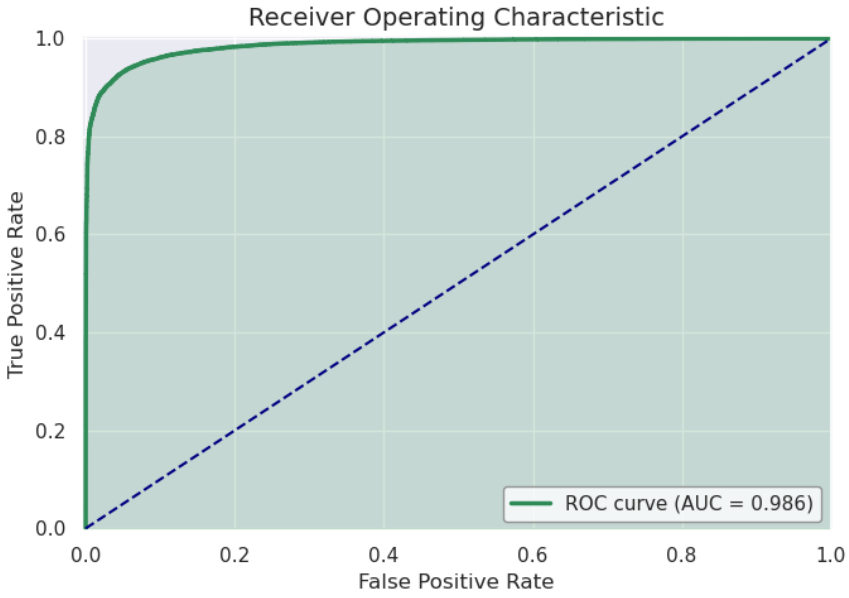
Confusion matrix na obr. 7.15 poskytuje detailný pohľad na rozloženie klasifikácií.



Obr. 7.15: Confusion matrix pre klasifikáciu phishingových domén

Z 20 000 legitímnych domén bolo chybne klasifikovaných 1 012, zatiaľ čo z 20 000 phishingových domén uniklo detekcii 1 360. Celková distribúcia chýb je vyvážená a ukazuje, že model neskláňa svoje rozhodovanie výrazne v prospech jednej z tried. Vzhľadom na charakter úlohy je mierne vyšší dôraz na recall akceptovateľný.

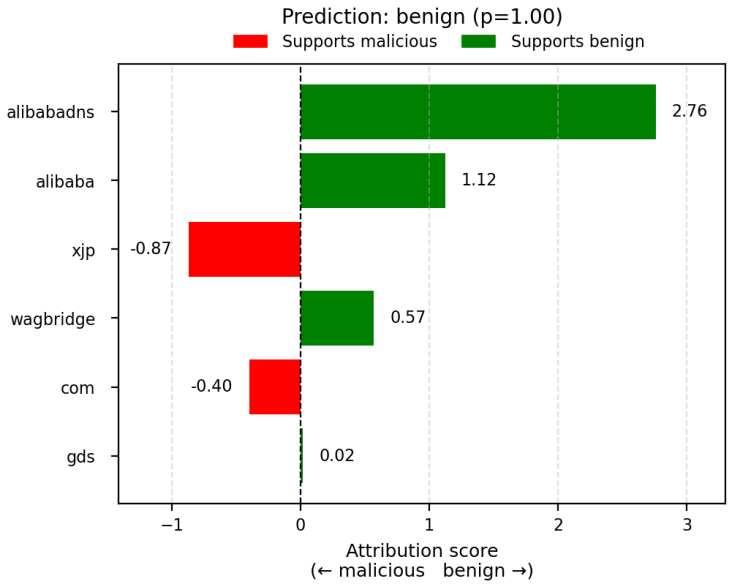
Na obr. 7.16 je zobrazená ROC krivka, ktorá zachytáva výkonnosť modelu naprieč rôznymi prahmi klasifikácie.



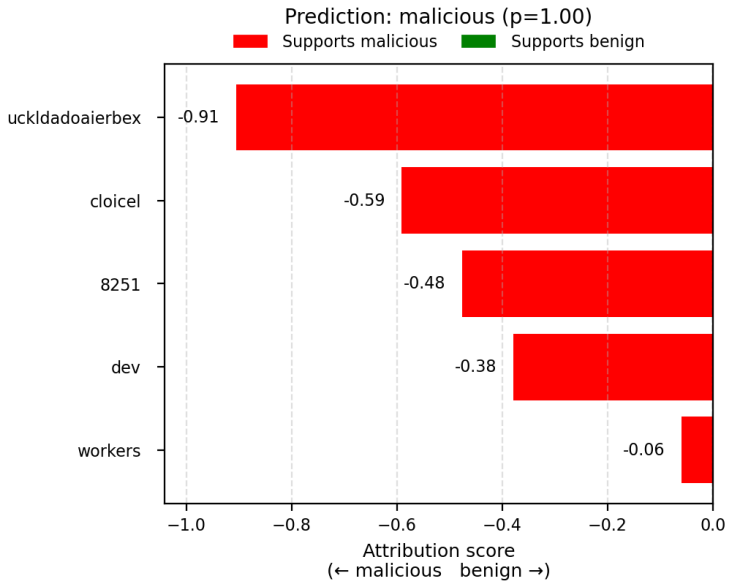
Obr. 7.16: ROC krivka pre klasifikáciu phishingových domén

Krivka má výborný tvar s rýchlym nástupom k hornej hrane grafu, čo potvrdzuje silnú diskriminačnú schopnosť modelu. Aj pri menších posunoch prahu zostáva zachytených veľké množstvo phishingových domén s nízkym výskytom falošných poplachov.

Na záver boli na dvoch vzorkách z testovacej množiny pomocou metódy Integrated Gradients vizualizované najvplyvnejšie tokeny. Tokeny boli pred zobrazením upravené – boli odstránené špeciálne znaky, zlúčené fragmenty slov a ignorované jednopísmenové výskyty.



Obr. 7.17: Integrated Gradients pre náhodne vybranú benignnú doménu (phishing klasifikátor)



Obr. 7.18: Integrated Gradients pre náhodne vybranú phishingovú doménu (phishing klasifikátor)

Na obr. 7.17 je možné vidieť, že model označil tokeny `"alibabadns"`, `"alibaba"` a `"wagbridge"` ako najvýraznejšie pozitívne faktory. Negatívne skóre tokenu `"xjp"` je síce silnejšie, no nedostatočné na zvrátenie klasifikácie. Naopak, phishingová doména na obr. 7.18 obsahuje viaceré tokeny s výrazným negatívnym vplyvom, ako `"uckldadaoierbex"` či `"cloicel"`, ktoré model vyhodnotil ako rozhodujúce faktory pre škodlivú klasifikáciu.

Zhrnutím možno povedať, že model pre klasifikáciu phishingových domén dosahuje výborné výsledky. Dokáže spoľahlivo identifikovať phishingové domény s minimálnym počtom falošných pozitívnych rozhodnutí. Výsledky metrík, priebeh učenia, distribúcia chýb aj interpretácia rozhodnutí naznačujú, že ide o robustný a prakticky použiteľný model.

7.6 Vyhodnotenie modelov nad RDAP dátami

V tejto časti sa hodnotí výkonnosť modelov trénovaných na rozšírených vstupoch, ktoré okrem doménových mien obsahujú aj štruktúrované informácie z RDAP protokolu. Tieto údaje zahŕňajú napríklad názvy registrátorov, stav domény, technické kontakty alebo špecifiká registračných záznamov. Cieľom je overiť, či pridanie týchto informácií zvyšuje presnosť detekcie škodlivých domén v porovnaní s klasifikátormi založenými výhradne na názve domény.

7.6.1 Klasifikátor malvérových domén (RDAP)

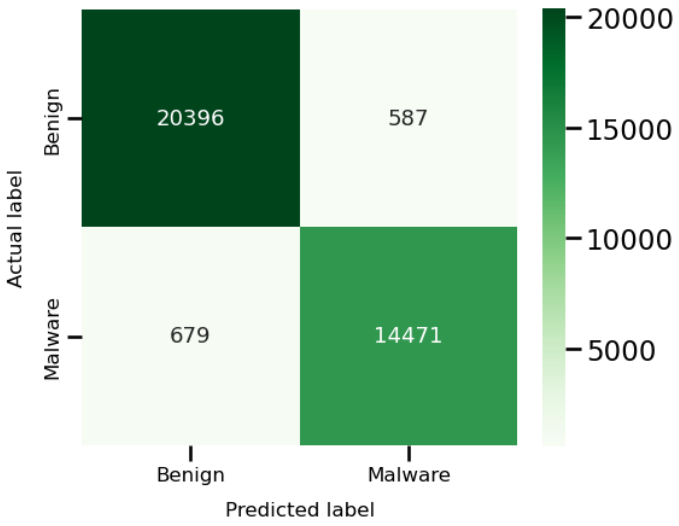
Model trénovaný na kombinácii doménových mien a RDAP údajov dosiahol nasledujúce hodnoty klasifikačných metrík:

- **skóre F1:** 95,70%,
- **presnosť (accuracy):** 96,42%,
- **precíznosť (precision):** 96,15%,
- **úplnosť (recall):** 95,25%.

Z pohľadu klasifikačných metrík ide o významné zlepšenie oproti modelu založenému len na doménových menách. Vyššia precision znamená, že legitímne domény sú klasifikované správne vo väčšom počte prípadov, zatiaľ čo zvýšený recall dokazuje lepšiu schopnosť detekovať škodlivé domény. Model tak dokáže efektívnejšie pracovať s doplnkovými informáciami o pôvode a registrácii domény. Vývoj trénovacej a validačnej straty počas učenia zobrazuje graf na obr. 7.19.

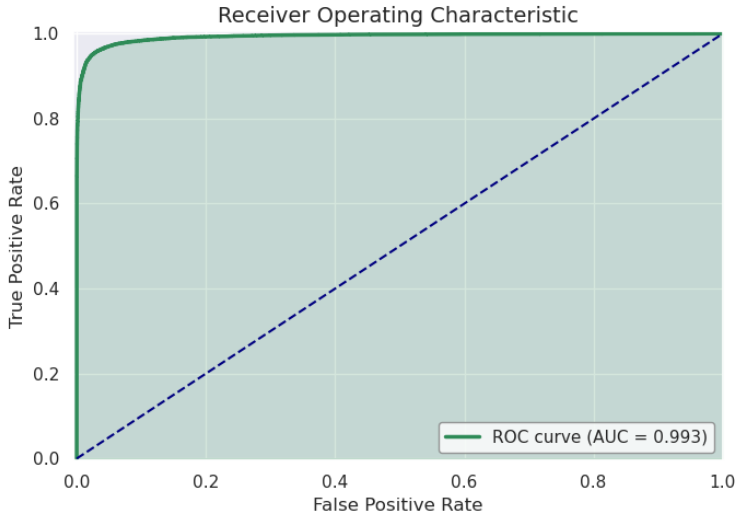
Obr. 7.19: Priebeh trénovacej a validačnej straty pri klasifikácii malvérových domén (RDAP)

Z grafu možno pozorovať rýchly pokles trénovacej straty počas prvých troch epoch, zatiaľ čo validačná strata sa znižuje pomalšie a od štvrtej epochy vykazuje kolísavý priebeh. Od piatej epochy začína rásť, čo naznačuje začínajúce pretrénovanie. Model bol preto trénovaný s použitím mechanizmu early stopping, ktorý trénovanie ukončil po strate zlepšenia na validačnej množine. Confusion matrix na obr. 7.20 dopĺňa numerické metriky konkrétnym rozložením správnych a chybných predikcií.



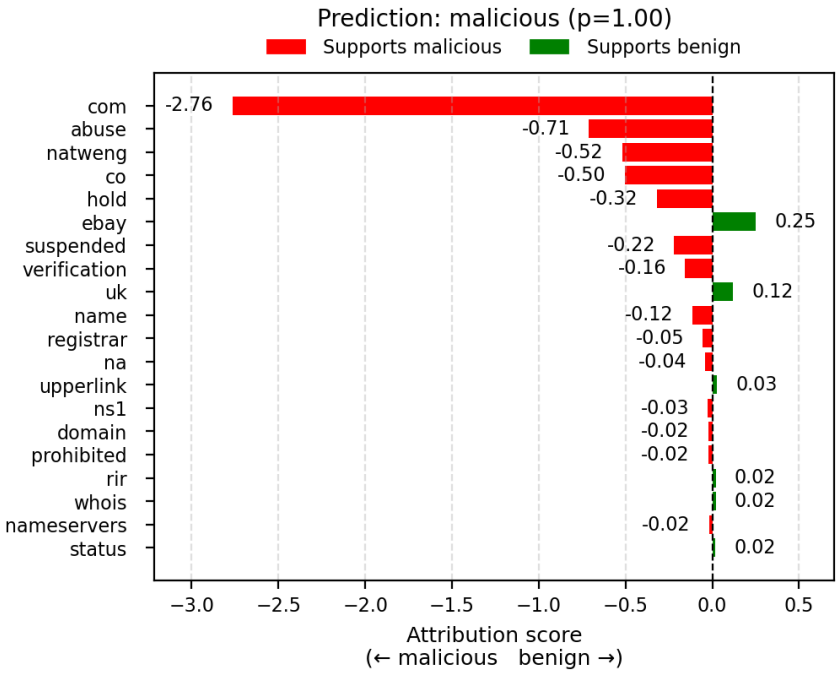
Obr. 7.20: Confusion matrix pre klasifikáciu malvérových domén (RDAP)

Model správne identifikoval 14 471 z 15 150 škodlivých domén, pričom omylom označil 587 legitímnych domén ako škodlivé. Tieto čísla potvrdzujú vysokú presnosť klasifikácie a zároveň ukazujú, že doplnenie RDAP údajov prispelo k zníženiu miery falošne pozitívnych aj falošne negatívnych predikcií. Spoľahlivosť modelu potvrdzuje aj ROC krivka na obr. 7.21, ktorá ukazuje veľmi vysokú diskriminačnú schopnosť modelu medzi škodlivými a legitímnymi doménami.



Obr. 7.21: ROC krivka pre klasifikáciu malvérových domén (RDAP)

Krivka rýchlo stúpa k hodnote 1 a udržiava si malú mieru falošných klasifikácií, čo svedčí o vysokej spoľahlivosti modelu aj pri zmene rozhodovacieho prahu. Na záver bola pomocou metódy Integrated Gradients analyzovaná jedna náhodne vybraná škodlivá doména, aby bolo možné získať detailnejší pohľad na rozhodovanie modelu. Zobrazené tokeny boli upravené tak, aby zahŕňali iba významové celky – odstránené boli špeciálne znaky, zlučované fragmenty a vynechané jednopísmenové výskyty.



Obr. 7.22: Integrated Gradients pre škodlivú doménu pri klasifikácii s RDAP údajmi

Na obr. 7.22 vidieť, že najvyšší negatívny vplyv na klasifikáciu mali tokeny ako `"com"`, `äbuse"`, `"natweng"` alebo `"hold"`, ktoré sa často objavujú v kontexte pozastavených alebo

podozrivých domén. Pozitívne príspevky ako `ëbay"` alebo `"verification"` síce figurujú v interpretácii, no ich celkový vplyv na predikciu je nízky. Výsledok poukazuje na to, že model dokáže správne identifikovať relevantné znaky v RDAP zázname, ktoré sú typické pre škodlivé domény.
Celkovo možno konštatovať, že pridanie RDAP údajov zlepšilo výkonnosť modelu pri detekcii malvérových domén. Model si zachováva vysokú presnosť a spoľahlivosť, dobre generalizuje a vykazuje robustné správanie aj pri rozšírenom vstupe.

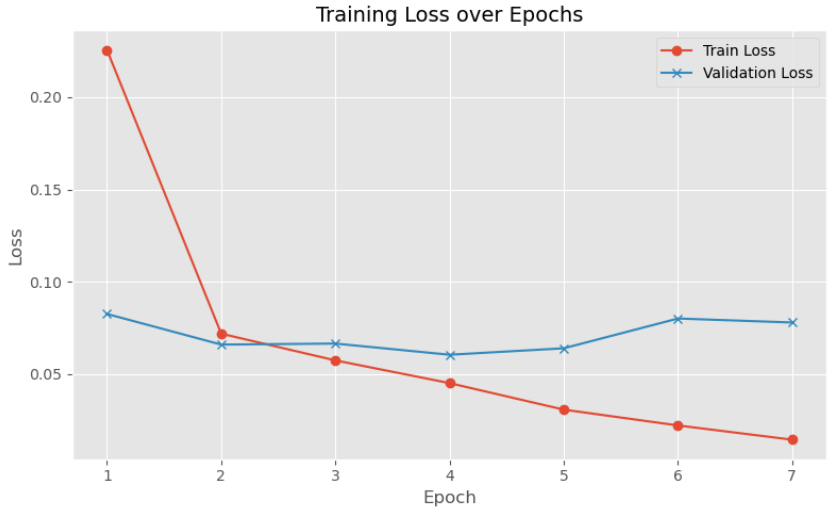
7.6.2 Klasifikátor phishingových domén (RDAP)

Model trénovaný na kombinácii doménových mien a informácií z RDAP záznamov dosiahol nasledujúce výsledky:

- **presnosť (accuracy):** 98,12%,
- **precíznosť (precision):** 99,24%,
- **úplnosť (recall):** 97,55%,
- **skóre F1:** 98,39%.

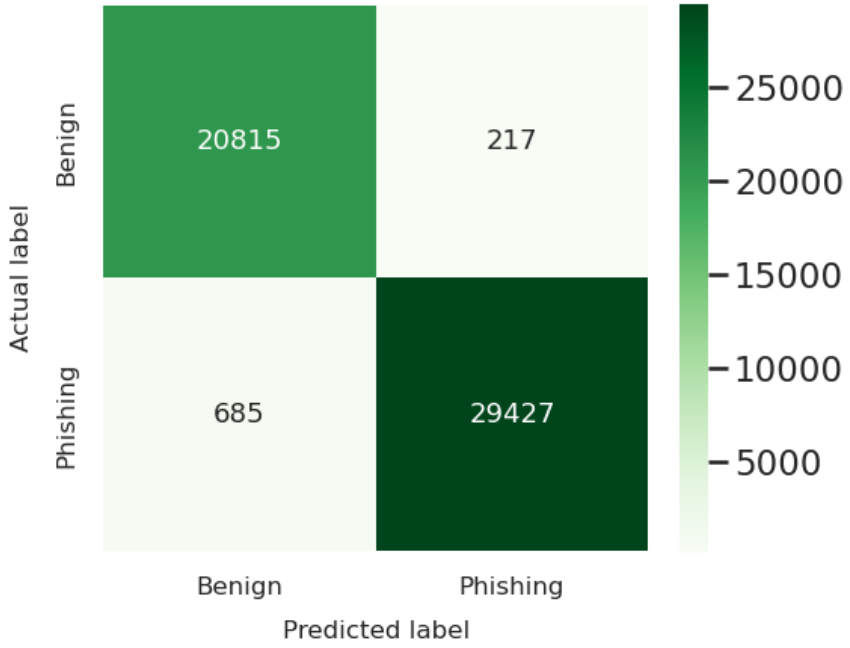
Tieto výsledky naznačujú vysokú spoľahlivosť modelu pri identifikácii phishing domén. Recall takmer 98% poukazuje na schopnosť zachytiť väčšinu škodlivých záznamov, zatiaľ čo vysoká precision potvrdzuje, že iba malé množstvo legitímnych domén bolo nesprávne označené. Vysoké skóre F1 odráža dobrý kompromis medzi oboma metrikami.

Na overenie stability modelu počas trénovania bol sledovaný priebeh trénovacej a validačnej straty (obr. 7.23). Model dosiahol rýchlu konvergenciu, pričom validačná strata sa po tretej epoche stabilizovala. Od piatej epochy však dochádza k miernemu zhoršeniu na validačných dátach, čo môže naznačovať začínajúce pretrénovanie. Tréning preto využíval mechanizmus early stopping, ktorý ukončil optimalizáciu po stagnácii validačnej straty.



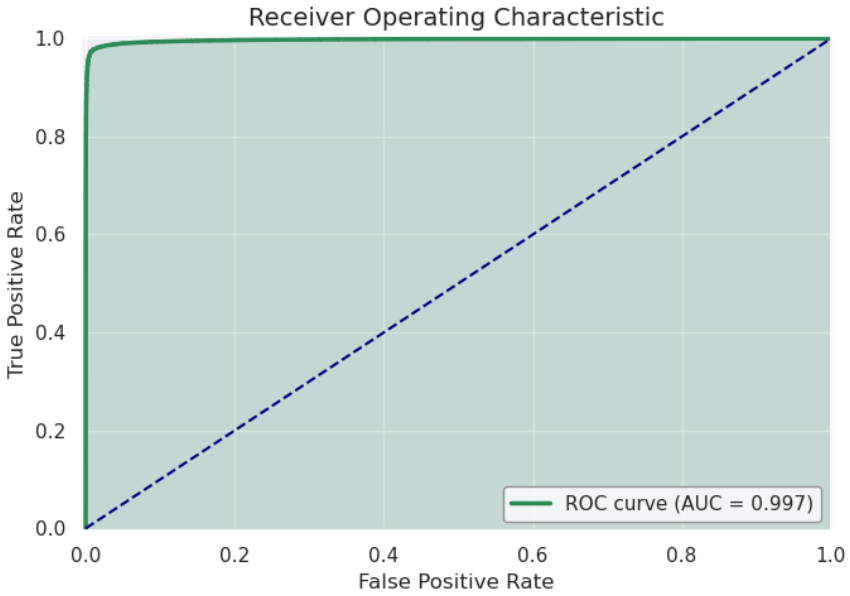
Obr. 7.23: Priebeh trénovacej a validačnej straty pri klasifikácii phishing domén (RDAP)

Spoľahlivosť modelu potvrdzuje aj confusion matrix (obr. 7.24). Nesprávne bolo klasifikovaných len 217 legitímnych domén ako phishing a 685 phishing domén uniklo detekcii. Vzhľadom na veľkosť testovacej množiny ide o veľmi priaznivý výsledok.



Obr. 7.24: Confusion matrix pre klasifikáciu phishing domén (RDAP)

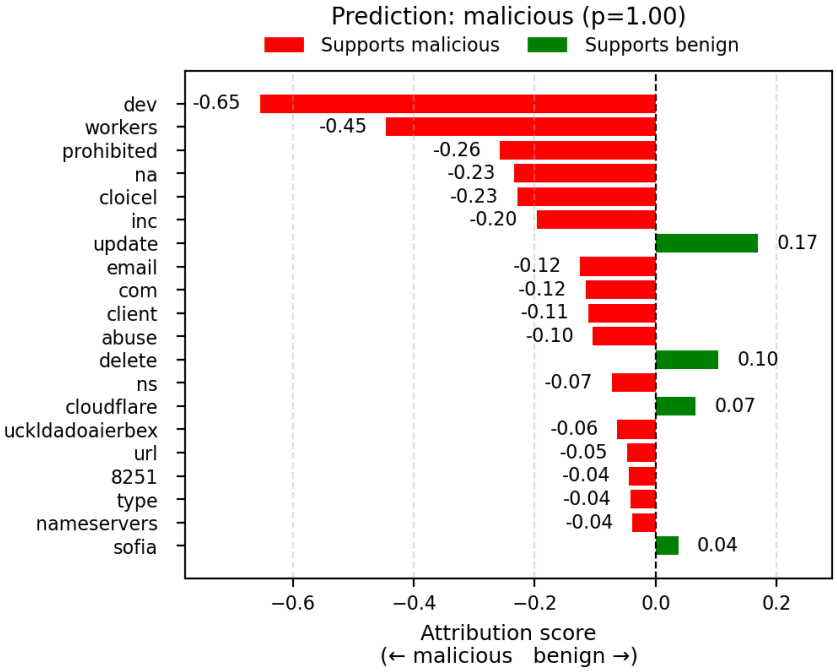
Dobrú schopnosť oddeľovať legitímne a phishing domény potvrdzuje aj tvar ROC krivky (obr. 7.25). Krivka sa blíži ideálnemu tvaru, čo svedčí o vysokom rozlišovacom výkone modelu naprieč rôznymi prahmi rozhodovania.



Obr. 7.25: ROC krivka pre klasifikáciu phishing domén (RDAP)

Interpretácia pomocou metódy Integrated Gradients na obr. 7.26 ukazuje, ktoré prvky záznamu prispievajú k predikcii phishingu. Medzi najviac prispievajúce tokeny patria napríklad `"dev"`, `"workers"` alebo `"prohibited"`, ktoré sa často vyskytujú v súvislosti s podoz-

rivými doménami. Naopak, tokeny ako `"update"` alebo `"cloudflare"` majú mierne kladné skóre a môžu indikovať benígnosť.



Obr. 7.26: Interpretabilita predikcie pre phishing doménu – Integrated Gradients

Model postavený nad rozšírenými RDAP dátami tak dokazuje, že kombinácia textových reprezentácií doménového mena a registrátorských údajov výrazne pomáha pri spoľahlivej detekcii phishingu.

7.7 Vyhodnotenie modelov nad DNS dátami

V tejto časti sú vyhodnotené modely trénované nad obohatenými vstupmi, ktoré zahŕňajú informácie zo záznamov DNS. Tieto dáta ponúkajú pohľad na infraštruktúru súvisiacu s doménovým menom, čo môže pomôcť pri odhaľovaní anomálií a vzorcov typických pre škodlivé domény.

7.7.1 Klasifikácia malvérových domén (DNS)

Model vytrénovaný na DNS dátach dosiahol nasledujúce klasifikačné výsledky:

- **presnosť (accuracy):** 95,98%,
- **precíznosť (precision):** 96,56%,
- **úplnosť (recall):** 95,30%,
- **skóre F1:** 95,93%.

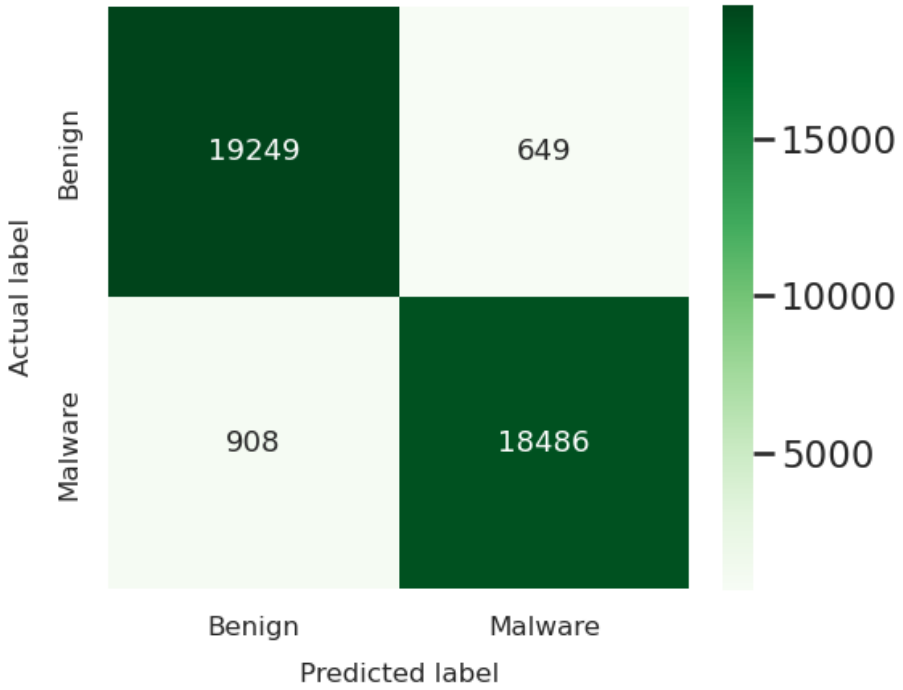
Výsledky poukazujú na veľmi dobrú vyváženosť medzi citlivosťou a presnosťou modelu. Vysoká presnosť znamená, že väčšina detegovaných domén bola skutočne škodlivá, zatiaľ čo

hodnota recall nad 95% potvrdzuje schopnosť zachytiť väčšinu všetkých škodlivých prípadov. Kombinovaná F1-hodnota blízka 96% poukazuje na spoľahlivé rozhodovanie aj v náročnejších prípadoch.

Pre lepšie porozumenie správania modelu bol analyzovaný priebeh trénovania (obr. 7.27). Z grafu vyplýva, že model konverguje stabilne, avšak po druhej epoche možno pozorovať náhly výkyv validačnej straty. Po tejto anomálii sa však strata vracia do normálu a vývoj ďalej prebieha vyrovnane. Použitý mechanizmus early stopping zabezpečil, že model nebol pretrénovaný, a trénovanie sa ukončilo po siedmej epoche.

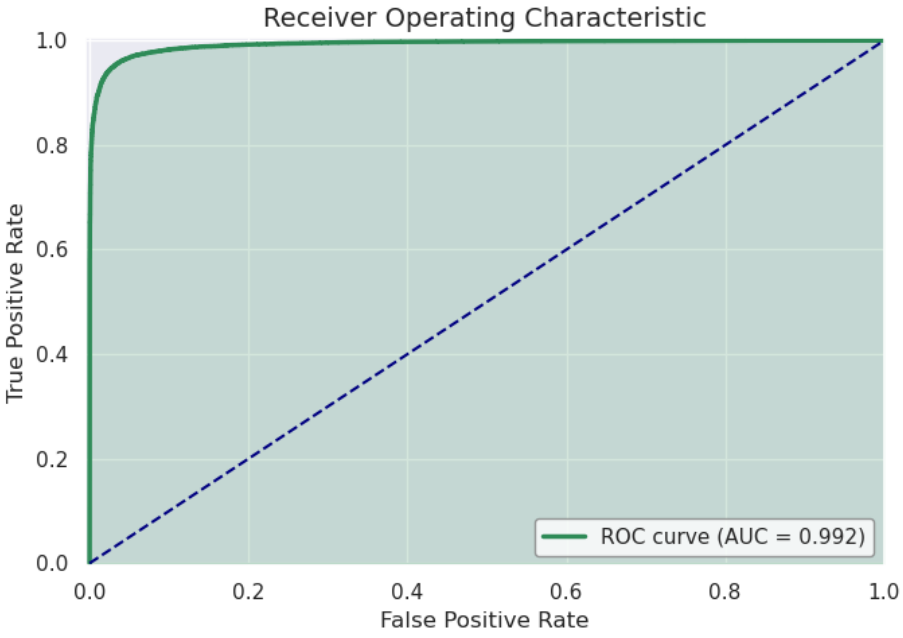
Obr. 7.27: Priebeh trénovacej a validačnej straty pri detekcii malvéru z DNS dát

Spoľahlivosť modelu potvrdzuje aj confusion matrix na obr. 7.28, kde vidieť veľmi nízky počet chybných predikcií. Nesprávne bolo klasifikovaných len 649 legitímnych a 908 škodlivých domén, čo predstavuje len malé percento vzhľadom na veľkosť testovacej množiny.



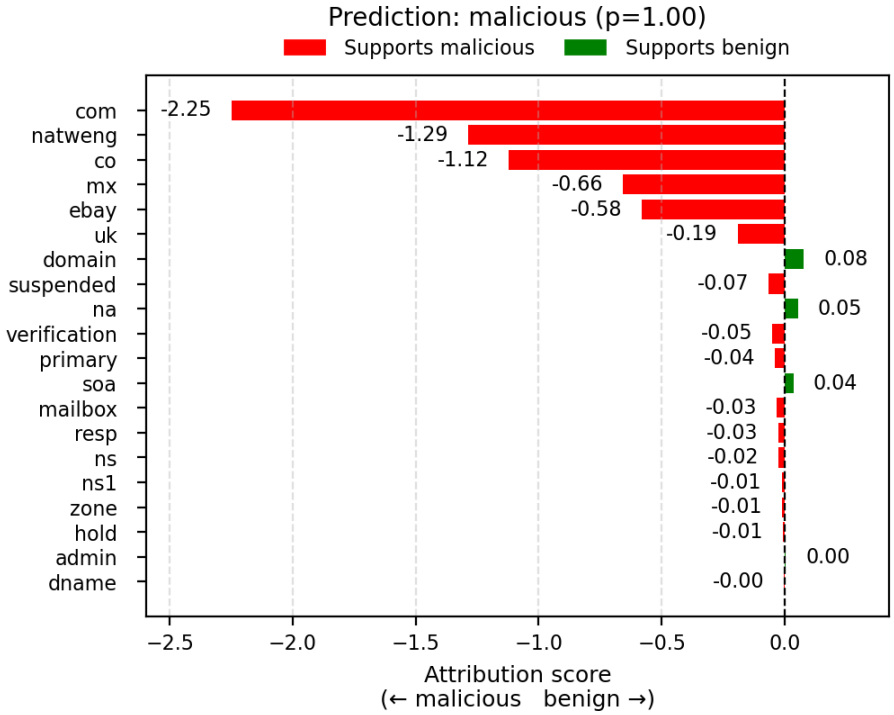
Obr. 7.28: Confusion matrix pre detekciu malvéru na základe DNS dát

Podľa ROC krivky na obr. 7.29 je model schopný veľmi dobre oddeľovať škodlivé a legitímne domény. Krivka sa drží blízko ľavého horného rohu, čo znamená vysoký podiel správne klasifikovaných pozitívnych prípadov pri nízkej chybovosti.



Obr. 7.29: ROC krivka pre klasifikáciu malvéru z DNS dát

Interpretácia pomocou metódy Integrated Gradients bola vykonaná na náhodne vybranej škodlivej doméne. Na obrázku 7.30 sú zobrazené atribučné skóre jednotlivých tokenov. Najsilnejší vplyv na škodlivú predikciu mali tokeny ako `"com"`, `"natweng"`, `"co"` alebo `"mx"`, ktoré sa často objavujú v infraštruktúre podozrivých alebo zneužitých domén. Zelené prvky podporujú klasifikáciu ako legitímnu, červené ako škodlivú.



Obr. 7.30: Vizualizácia atribučných skóre pomocou Integrated Gradients pre škodlivú doménu

7.7.2 Klasifikátor phishingových domén (DNS)

Model určený na klasifikáciu phishingových domén nad DNS dátami dosiahol veľmi presvedčivé výsledky. Základné klasifikačné metriky sú nasledovné:

- **presnosť (accuracy):** 98,00%,
- **precíznosť (precision):** 98,44%,
- **úplnosť (recall):** 97,56%,
- **skóre F1:** 98,00%.

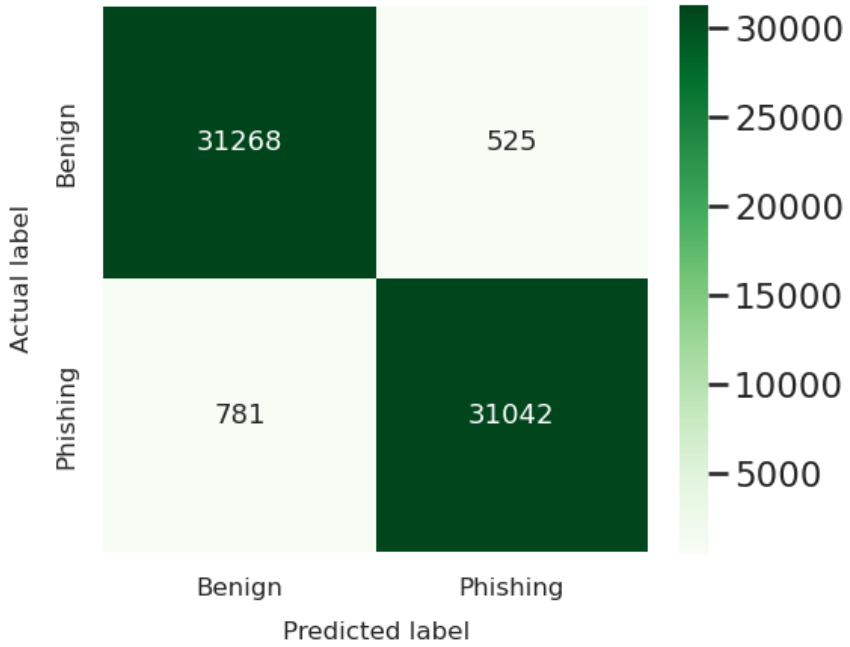
Tieto výsledky naznačujú vysokú presnosť pri detekcii phishingových domén aj spoľahlivé odlíšenie legitímnych prípadov. Vysoká presnosť potvrdzuje, že väčšina označených phishingových domén bola skutočne škodlivá, zatiaľ čo hodnota recall blízka 98% naznačuje, že len málo škodlivých domén uniklo detekcii. Výsledné skóre F1 ukazuje na vyvážený výkon medzi oboma aspektmi.

Pre lepšie porozumenie správania modelu bol analyzovaný priebeh učenia, znázornený na obrázku 7.31. Trénovacia strata sa počas prvých epoch výrazne znižuje a neskôr stabilne

klesá. Validačná strata nasleduje podobný trend, pričom zostáva nízka až do posledných epoch, kde dochádza k miernemu nárastu. Tento priebeh ukazuje, že model sa učil stabilne a bez výrazného pretrénovania.

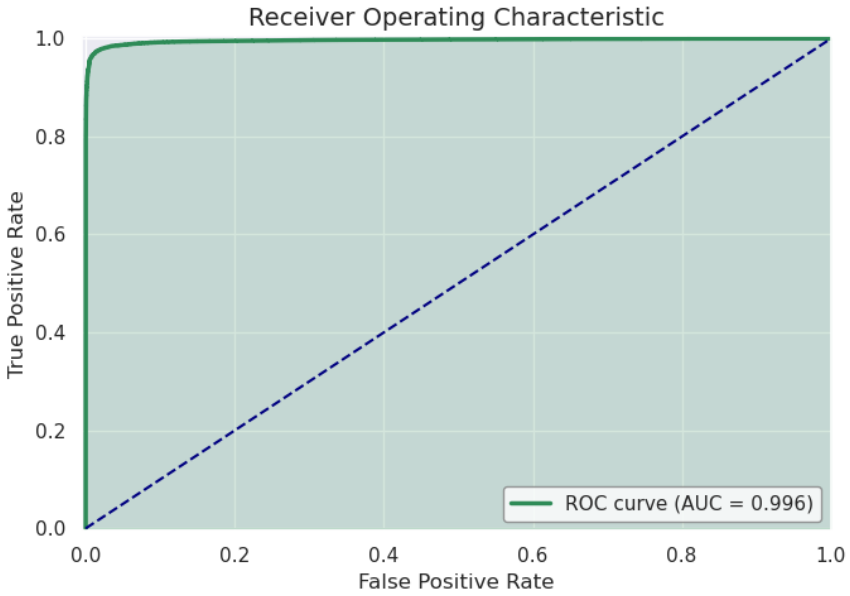
Obr. 7.31: Priebeh trénovacej a validačnej straty pri klasifikácii phishingových domén (DNS)

Spoľahlivosť modelu potvrdzuje aj confusion matrix na obrázku 7.32, z ktorej vyplýva, že nesprávne označených legitímnych domén bolo len 525 a phishingových 781 z celkového počtu viac ako 60 000 prípadov. Toto rozloženie chýb ukazuje rovnomerný výkon v oboch triedach.



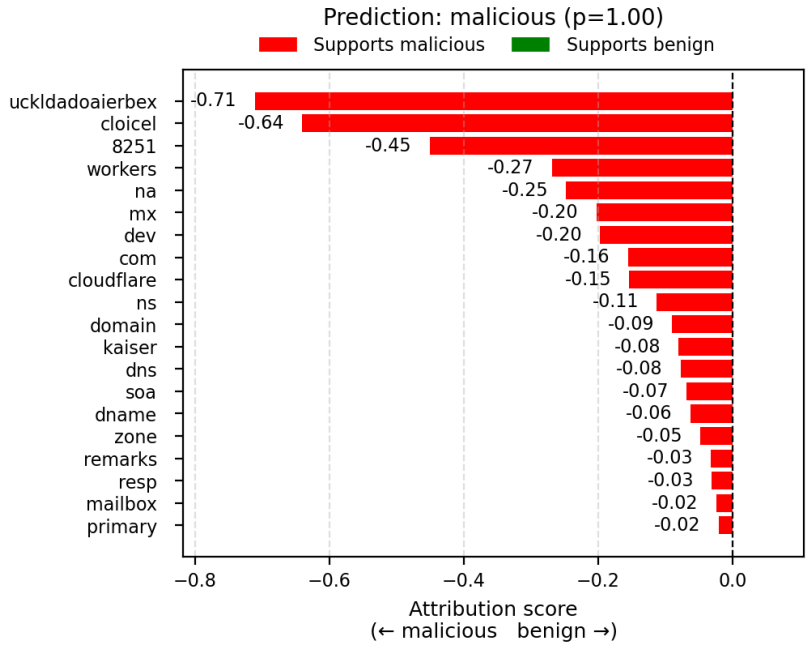
Obr. 7.32: Confusion matrix pre klasifikáciu phishingových domén (DNS)

Výkon modelu pri oddeľovaní phishingových a legitímnych domén potvrdzuje aj tvar ROC krivky na obrázku 7.33. Krivka naznačuje, že model udržiava vysokú mieru správne zachytených škodlivých prípadov aj pri nízkej miere falošných poplachov.



Obr. 7.33: ROC krivka pre klasifikáciu phishingových domén (DNS)

Pre doplnenie interpretovateľnosti bola vizualizovaná jedna náhodne vybraná phishingová doména z testovacej množiny pomocou metódy Integrated Gradients. Obrázok 7.34 znázorňuje, ktoré tokeny mali najväčší vplyv na rozhodnutie modelu. Medzi kľúčové znaky, ktoré prispeli k predikcii škodlivosti, patria napríklad `"uckldadoaierbex"`, `cloicel` a `8251`. Ich vysoké záporné atribučné skóre naznačuje jednoznačný posun predikcie smerom k phishingovej triede.



Obr. 7.34: Integrated Gradients pre náhodne vybranú phishingovú doménu (DNS)

Získané výsledky ukazujú, že model postavený nad DNS dátami je veľmi účinný nástroj na detekciu phishingových domén. Vysoké skóre metrík, stabilné učenie a interpretovateľné rozhodnutia potvrdzujú jeho spoľahlivosť aj v praktických scenároch.

7.8 Vyhodnotenie modelov nad geografickými dátami

V tejto časti boli doménové mená rozšírené o doplnkové informácie geografického charakteru, ako napríklad lokalita, región, štát alebo sieťová infraštruktúra súvisiaca s pôvodom domény. Cieľom je overiť, či takéto obohatenie vstupu zvyšuje presnosť modelu pri detekcii škodlivých aktivít. Nasledujúce výsledky ukazujú výkonnosť klasifikačných modelov trénovaných na týchto rozšírených dátach.

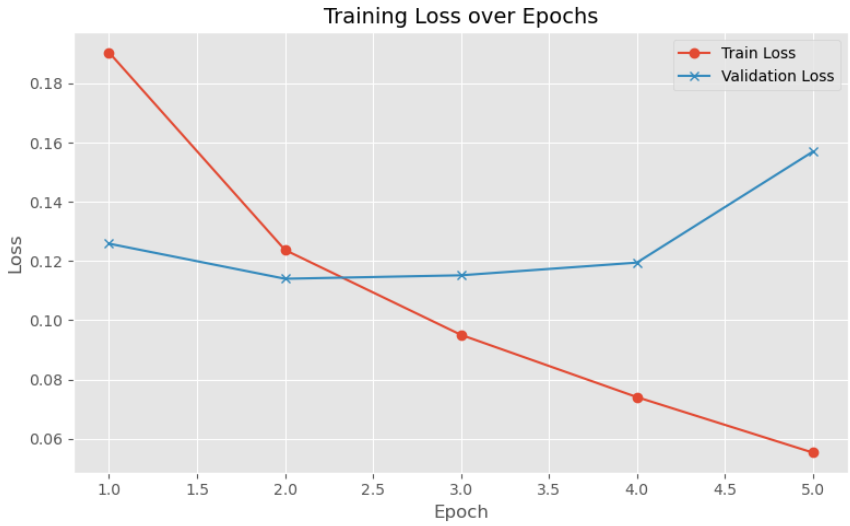
7.8.1 Klasifikátor malvérových domén (GEO)

Model dosiahol nasledujúce hodnoty klasifikačných metrík:

- **presnosť (accuracy):** 95,74%,
- **precíznosť (precision):** 95,23%,
- **úplnosť (recall):** 96,37%,
- **skóre F1:** 95,79%.

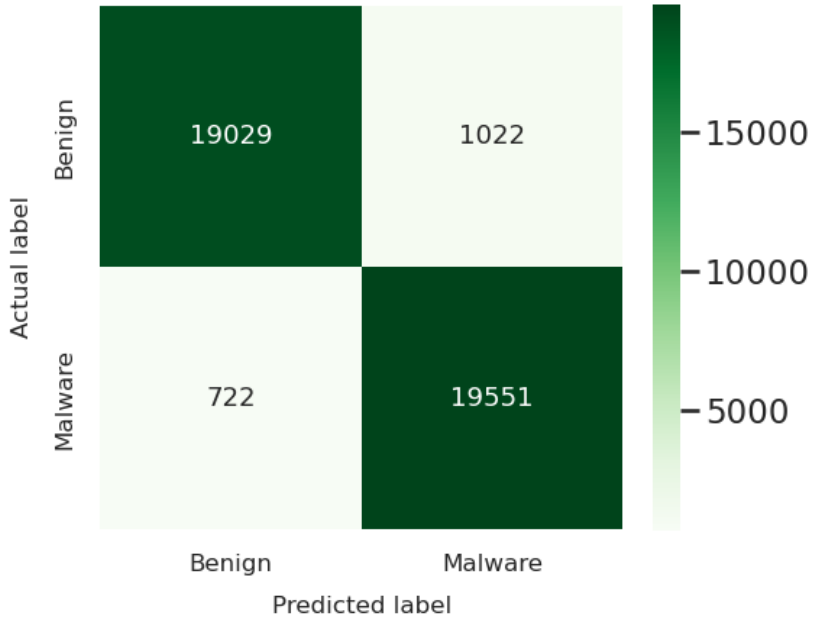
Model si udržiava vysokú úroveň výkonnosti, pričom recall aj precision sú vyvážené a blízke 96,%. Vysoká hodnota recall svedčí o tom, že väčšina škodlivých domén bola správne detegovaná, zatiaľ čo vysoká precision potvrdzuje, že model len zriedkavo označil legitímnu doménu ako škodlivú.

Pre detailnejšie posúdenie priebehu učenia a stability modelu bol sledovaný vývoj trénovacej a validačnej straty (obr. 7.35). Trénovacia strata konzistentne klesá, zatiaľ čo validačná po počiatočnom poklese začne rásť po štvrtej epoche. Tento priebeh môže naznačovať jemné pretrénovanie, ktoré však bolo kontrolované použitím mechanizmu early stopping.



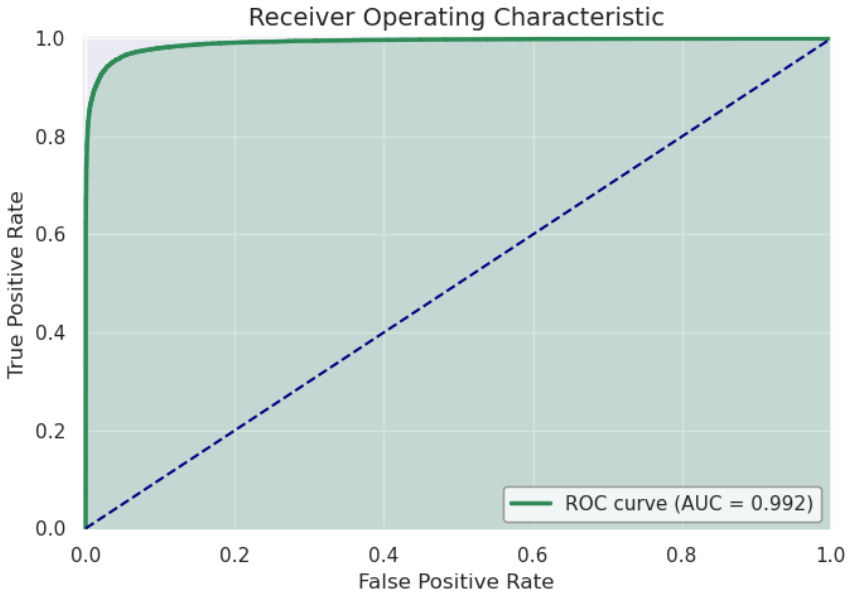
Obr. 7.35: Priebeh trénovacej a validačnej straty pri klasifikácii malvéru (GEO)

Spoľahlivosť modelu potvrdzuje aj confusion matrix (obr. 7.36), z ktorej vyplýva, že z 20 000 škodlivých domén bolo nesprávne klasifikovaných len 722 prípadov a z 20 000 legitímnych domén len 1022. Model teda zvláda klasifikáciu oboch tried s vysokou presnosťou.



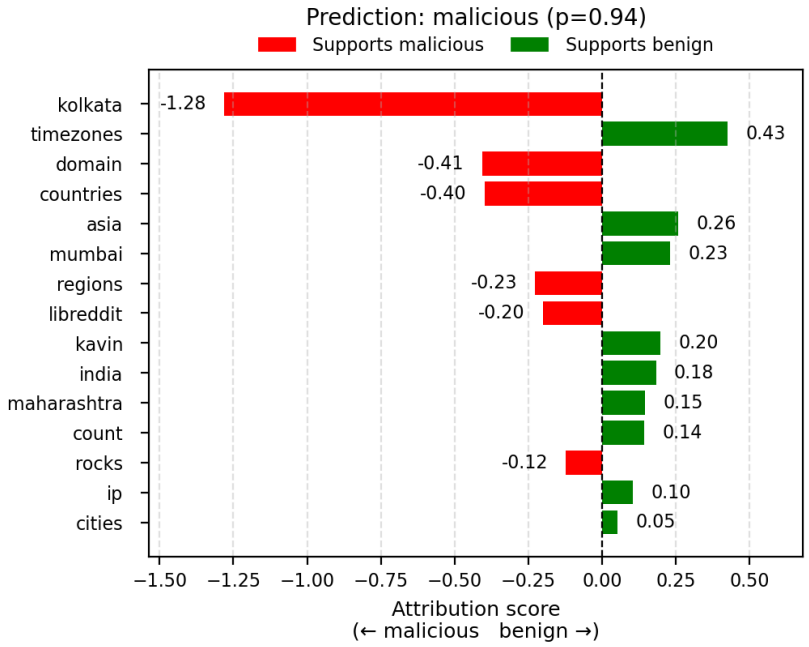
Obr. 7.36: Confusion matrix pre klasifikáciu malvéru (GEO)

Schopnosť modelu rozlišovať medzi legitímnymi a škodlivými doménami potvrdzuje aj ROC krivka (obr. 7.37), ktorá vykazuje veľmi priaznivý priebeh s minimálnym podielom falošných poplachov a rýchlym nárastom zachytených škodlivých prípadov.



Obr. 7.37: ROC krivka pre klasifikáciu malvéru (GEO)

Na záver bola vybraná jedna škodlivá doména pre ilustráciu interpretovateľnosti pomocou metódy Integrated Gradients. Obrázok 7.38 zobrazuje tokeny, ktoré najviac ovplyvnili predikciu. Červené stĺpce značne prispievajú k označeniu domény ako škodlivej, zatiaľ čo zelené ju podporujú ako legitímnu.



Obr. 7.38: Interpretácia rozhodnutia modelu pre škodlivú doménu (GEO)

Medzi najvýznamnejšie záporné tokeny patria `"kolkata"`, `"domain"` a `"countries"`, ktoré model považoval za indikátory škodlivého obsahu. Na druhej strane tokeny ako `"timezones"` alebo `äsia"` mali opačný vplyv. Tieto pozorovania potvrdzujú, že model rozlišuje medzi rôznymi geografickými výrazmi a ich súvislosťou so škodlivým správaním.

7.8.2 Klasifikátor phishingových domén (GEO)

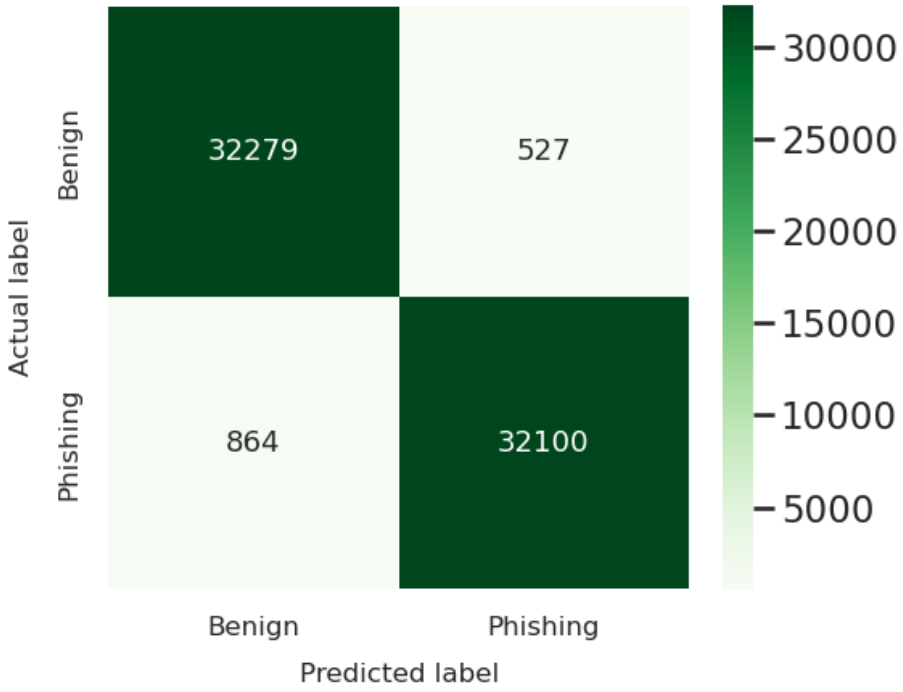
Model trénovaný nad geografickými dátami dosiahol nasledujúce výsledky:

- **presnosť (accuracy):** 97,95%,
- **precíznosť (precision):** 98,40%,
- **úplnosť (recall):** 97,47%,
- **skóre F1:** 97,93%.

Tieto hodnoty potvrdzujú, že model si zachováva vysokú úroveň spoľahlivosti a vyváženosti. Vysoká presnosť pri oboch triedach znamená, že model má nízky výskyt falošných označení, pričom recall blízky 98% ukazuje, že väčšina phishingových domén bola úspešne detegovaná. Vyvážené skore F1 odráža dobrý kompromis medzi detekciou a minimalizáciou falošných poplachov.

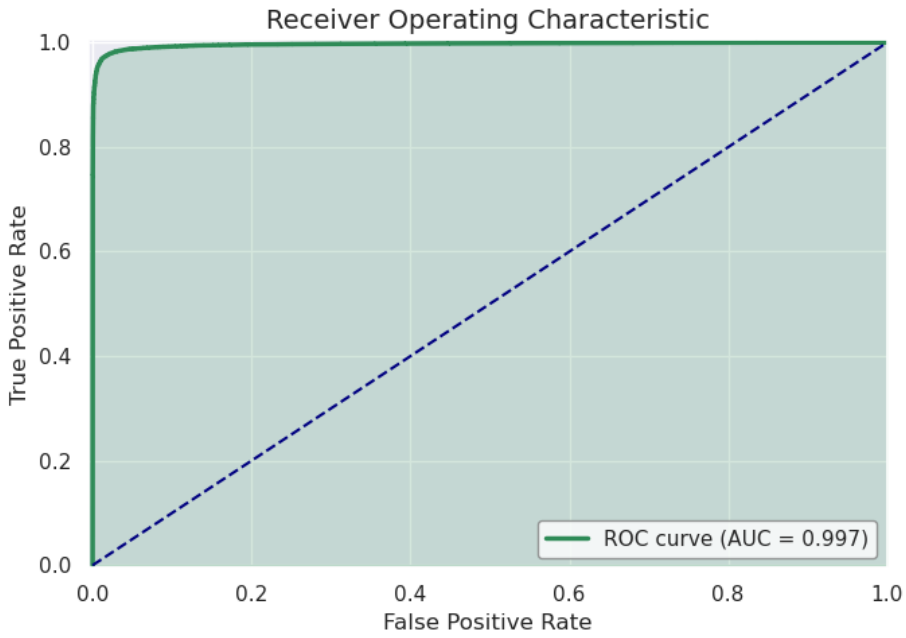
Obr. 7.39: Priebeh trénovacej a validačnej straty pri klasifikácii phishing domén (GEO)

Na obrázku 7.39 je vidieť, že model konverguje rýchlo a stabilne. Po počiatočnom poklese sa validačná strata ustáli a neskôr mierne stúpa, zatiaľ čo trénovacia strata naďalej klesá. To môže naznačovať mierne pretrénovanie, ktoré bolo ošetrené použitím mechanizmu early stopping.



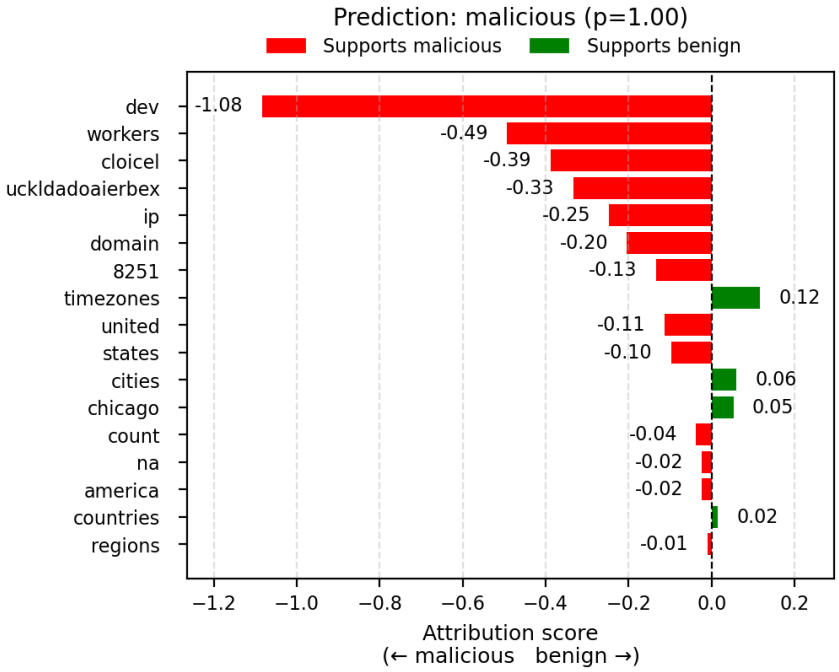
Obr. 7.40: Confusion matrix pre klasifikáciu phishing domén (GEO)

Z confusion matrix 7.40 vyplýva, že model urobil len 527 chybných klasifikácií pri legitímnych doménach a 864 pri phishingových. Tieto hodnoty sú vzhľadom na veľkosť testovacej množiny veľmi nízke a potvrdzujú konzistentnú výkonnosť modelu.



Obr. 7.41: ROC krivka pre klasifikáciu phishing domén (GEO)

Aj ROC krivka na obrázku 7.41 potvrdzuje schopnosť modelu spoľahlivo odlišovať medzi legitímnymi a phishingovými doménami. Krivka má strmý nástup pri veľmi nízkej miere falošných pozitív, čo je ideálny stav pri detekcii bezpečnostných hrozieb.



Obr. 7.42: Interpretácia predikcie pre phishing doménu pomocou metódy Integrated Gradients (GEO)

Na obrázku 7.42 je vizualizované atribučné skóre pre náhodne vybranú phishingovú doménu. Najviac k predikcii ako škodlivá prispeli tokeny `"dev"`, `"workers"`, `"cloicel"` alebo `"uckldadoaierbex"`. Niektoré slová ako `"timezone"`, `ünited"` alebo `"cities"` naopak mierne podporovali benignnú klasifikáciu, no ich váha bola výrazne nižšia. Tieto výsledky naznačujú, že model sa pri rozhodovaní opiera o kontextovo významné časti vstupu, čo zvyšuje dôveru v jeho správanie.

7.9 Zhrnutie vyhodnotenia všetkých modelov

Táto sekcia sumarizuje výsledky všetkých testovaných modelov nad rôznymi typmi dát a kategóriami škodlivých domén. Prezentované sú dosiahnuté hodnoty skóre F1 a výpočty *information gain*, ktoré vyjadrujú zlepšenie výkonu oproti klasifikácii založenej len na samotných doménových menách. Výsledky sú uvedené v tabuľkách 7.5 a 7.6.

Typ dát	**DGA**	**Malware**	**Phishing**
Doménové meno	**98.61 %**	89.71 %	94.17 %
RDAP	–	95.70 %	98.39 %
DNS	–	95.93 %	98.00 %
GEO	–	95.79 %	97.93 %

Tabuľka 7.5: Skóre F1 modelov podľa typu dát a kategórie domény

Typ dát	**DGA**	**Malware**	**Phishing**
RDAP	–	+6.00	+4.22
DNS	–	+6.22	+3.83
GEO	–	+6.08	+3.76

Tabuľka 7.6: Information gain oproti doménovým menám (v percentuálnych bodoch)

Z výsledkov je zrejmé, že pridanie kontextuálnych dát výrazne zlepšuje schopnosť modelov detegovať škodlivé domény. Pri detekcii **malware** domén dosiahli všetky dátové typy podobné zvýšenie presnosti, pričom najvyšší prínos malo DNS (+6.22 percentuálneho bodu). V prípade **phishingu** boli rozdiely mierne nižšie, ale stále významné – najväčší prínos bol zaznamenaný pri RDAP dátach (+4.22). Tieto výsledky potvrdzujú, že externé informácie o doméne (napríklad záznamy o infraštruktúre, geografii či vlastníkoch) poskytujú modelu kľúčové signály, ktoré mu umožňujú lepšie rozhodovať aj pri neznámych alebo maskovaných doménach.

Okrem klasifikačnej presnosti bol vykonaný aj experiment zameraný na meranie latencie klasifikácie jednej domény. Testovanie prebehlo na grafickej karte popísanej v kapitole o implementácii. Tabuľka 7.7 sumarizuje priemerný čas spracovania jednej vzorky a veľkosť jednotlivých modelov.

Model	**Latencia (ms)**	**Veľkosť (MB)**
Domain-DGA-BERT-medium	0.392	157.89
Domain-Phishing-Distil	0.483	255.46
Domain-Malware-ELECTRA	0.707	417.74
GEO-Phishing-Distil	0.816	255.46
GEO-Malware-ELECTRA	1.235	417.74
DNS-Phishing-Distil	1.637	255.46
RDAP-Phishing-Distil	2.749	255.46
DNS-Malware-ELECTRA	3.568	417.74
RDAP-Malware-ELECTRA	5.334	417.74

Tabuľka 7.7: Porovnanie latencie a veľkosti jednotlivých modelov

Z pohľadu efektivity sú najrýchlejšie modely, ktoré pracujú iba s doménovými menami. Pridanie ďalších dátových vstupov vedie k nárastu latencie, pričom najpomalšie sú modely spracúvajúce RDAP dáta. Tie poskytujú najviac informácií, ale zároveň sú aj výpočtovo najnáročnejšie. Veľkosť modelov zodpovedá použitej architektúre. Modely postavené na architektúre ELECTRA sú väčšie ako modely využívajúce DistilBERT, čo sa prejavuje vo vyšších požiadavkách na výpočtové zdroje. Z tabuľky je zároveň možné pozorovať súvislosť medzi veľkosťou modelu a rýchlosťou klasifikácie. S rastúcou veľkosťou modelu dochádza k poklesu rýchlosti spracovania jednej domény.

Kapitola 8

Diskusia

V tejto časti práce sú interpretované výsledky experimentov s cieľom objasniť ich význam a širšie súvislosti. Diskusia zahŕňa hlavné pozorovania, porovnanie s existujúcimi prístupmi, vybrané zaujímavosti z priebehu analýzy, odporúčania pre detekciu škodlivých domén, ako aj zhodnotenie prínosu dosiahnutých výsledkov. Zistenia sú posudzované z hľadiska ich praktickej využiteľnosti aj v kontexte stanovených cieľov práce.

8.1 Hlavné zistenia a pozorovania

Výsledky experimentov potvrdili, že názvy domén predstavujú významný zdroj informácií pri detekcii škodlivých domén, predovšetkým pri algoritmicky generovaných a phishingových prípadoch. V prípade DGA domén modely spoľahlivo identifikovali neprirodzené kombinácie znakov, zatiaľ čo pri phishingových doménach boli úspešné v rozpoznávaní výrazov imitujúcich známe služby, ako napríklad `"login"` či `"verify"`. Pri malvérových doménach samotný názov často nestačil. Výrazné zlepšenie detekcie nastalo po doplnení registračných, technických a sieťových údajov. Ukázalo sa, že faktory ako krátkodobá registrácia, anonymizovaný registrátor alebo opakované využívanie rovnakých serverov boli významnými indikátormi škodlivosti.

V rámci jednej architektúry sa pre všetky klasifikačné úlohy osvedčili rovnaké tréningové parametre, čo súvisí s jednotnou štruktúrou vstupov. Väčšie modely dosahovali najlepšiu presnosť, stredne veľké architektúry však ponúkli priaznivý kompromis medzi výkonom a výpočtovými nárokmi. Menšie modely poskytli dobrú spoľahlivosť a sú vhodné na nasadenie v prostrediach s obmedzenými zdrojmi.

Významný vplyv mala aj zvolená tokenizačná stratégia. Pri DGA doménach bola najvhodnejšia tokenizácia na úrovni znakov, zatiaľ čo pri phishingových a malvérových prípadoch boli efektívnejšie tokenizéry rozdeľujúce názvy podľa jazykových vzorov.

Modely konvergovali k vysokému výkonu už po niekoľkých tréningových cykloch. Rozhodnutia vychádzali z relevantných častí vstupu – v prípade DGA išlo o náhodné reťazce, pri phishingu o slová ako `"paypal"` či `"account"` a pri RDAP údajoch napríklad o stav domény alebo anonymizovaného registrátora. Rýchla konvergencia však viedla aj k rýchlemu pretrénovaniu, ak nebolo použité predčasné ukončenie učenia. Táto technika sa ukázala ako nevyhnutná na zachovanie schopnosti modelu generalizovať na nové dáta.

8.2 Zhodnotenie výsledkov v kontexte súčasného výskumu

Porovnanie s existujúcimi prácami ukazuje, že navrhované modely dosahujú konkurencieschopné výsledky. V oblasti detekcie algoritmicky generovaných domén (DGA) boli v úvode kapitoly 2 prezentované prístupy ako FANCI od Schüppena et al.[37], ktorý využíval náhodný les a dosiahol presnosť 99%, alebo práca Selviho et al.[38], kde model náhodného lesa založený na maskovaných n-gramoch dosiahol presnosť 98%. Pri detekcii phishingových domén možno výsledky porovnať s prácou Spotting the Hook[17], kde najvyššie skóre F1 dosiahol model LightGBM (0,976), pričom miera falošne pozitívnych klasifikácií bola veľmi nízka. V oblasti detekcie domén súvisiacich s malvérom možno ako referenčnú uviesť diplomovú prácu Jána Polišenského [33], kde po kombinácii viacerých typov údajov dosiahol model LightGBM skóre F1 nad 0,98. V našich experimentoch sa podarilo dosiahnuť porovnateľné hodnoty už pri použití jednotlivých dátových podmnožín, ako sú záznamy RDAP alebo DNS, čo potvrdzuje účinnosť testovaných modelov pri práci s konkrétnymi dátovými zdrojmi. Porovnanie s uvedenými štúdiami je zároveň významné tým, že využívali identickú dátovú sadu, čo umožňuje presnejšie vyhodnotiť vplyv konkrétnych algoritmov a metodiky na výslednú úspešnosť detekcie.

8.3 Praktická uplatniteľnosť a obmedzenia riešenia

Z pohľadu praktického nasadenia sa ako kľúčové ukázalo najmä to, že modely založené na názvoch domén dosahovali veľmi nízku latenciu klasifikácie. Napríklad modely Domain-DGA-BERT-medium alebo Domain-Phishing-Distil klasifikovali s priemernou dobou pod 0,5 milisekundy. Tieto výsledky potvrdzujú, že takéto modely sú vhodné pre online spracovanie dát v reálnom čase, kde sú výpočtové zdroje alebo reakčný čas kritickým faktorom. Naopak, pri modeloch využívajúcich doplnkové údaje ako RDAP, DNS alebo geolokačné informácie sa prejavila výrazne vyššia latencia. Napríklad model RDAP-Malware-ELECTRA dosahoval priemerný čas klasifikácie viac ako 5 milisekúnd. Hoci túto latenciu do určitej miery ovplyvňuje architektúra modelu, v praktickom nasadení ju ešte viac zvyšuje potreba tieto údaje dodatočne získať z externých zdrojov po tom, čo je doména zachytená. Tento krok môže celý proces klasifikácie výrazne spomaliť a z toho dôvodu sú tieto modely vhodnejšie pre offline spracovanie, kde nie je požiadavka na okamžitú reakciu. Môžu byť efektívne nasadené napríklad pri dávkovej analýze v rámci Threat Intelligence platforiem, pri retrospektívnej detekcii alebo ako podpora analytikov v bezpečnostných operačných centrách. Modely pracujúce výhradne s názvami domén sa tak javia ako vhodné riešenie pre systémy, kde je prioritou rýchla odozva, ako sú IDS, firewall alebo proxy servery.

8.4 Etické aspekty riešenia

Všetky klasifikátory uvedené v práci sú vytvorené na základe verejne dostupných dát, ako sú názvy domén, informácie z DNS, RDAP alebo geografické údaje o infraštruktúre. Nepoužívajú sa žiadne dôverné informácie o konkrétnych používateľoch, organizáciách ani iných citlivých entitách. Takýto prístup zaručuje súlad s princípmi ochrany súkromia a minimalizuje etické riziká spojené so zberom a spracovaním údajov.

Zároveň je potrebné zohľadniť riziká spojené s nesprávnou detekciou, najmä falošne pozitívnymi klasifikáciami, pri ktorých môže byť legitímna doména nesprávne označená ako škodlivá. Takéto rozhodnutie môže mať praktické dôsledky, ako je nedostupnosť služby, na-

rušenie prevádzky alebo poškodenie reputácie subjektu, ktorému doména patrí. Na zníženie týchto rizík sa javí ako vhodné kombinovať výstupy viacerých klasifikačných modelov trénovaných nad rôznymi dátovými zdrojmi. Ich výsledky môže následne spracovávať nadriadený rozhodovací modul, napríklad model založený na gradientne zosilňovaných rozhodovacích stromoch, ktorý umožní efektívne vyhodnotiť konfliktné výstupy a zvýšiť robustnosť systému bez nutnosti manuálneho zásahu.

8.5 Možné vylepšenia a rozšírenia

Transformerové architektúry sa v úlohe detekcie malígnych domén osvedčili a ponúkajú priestor na ďalšie zlepšenia v presnosti, robustnosti aj efektivite. Z hľadiska vstupov je možné rozšíriť modely o kombinované spracovanie viacerých dátových zdrojov, ako sú DNS, RDAP a geolokačné údaje, v jednej architektúre. Prínosom môže byť aj zavedenie priebežného učenia, ktoré by umožnilo rýchlu adaptáciu modelov na nové typy domén bez potreby úplného pretrénovania.

Ďalej je vhodné optimalizovať výpočtové nároky modelov pre nasadenie v prostredí, kde je dôležitý vysoký výkon a nízka latencia klasifikácie, napríklad pri spracovaní veľkých objemov sieťovej prevádzky v reálnom čase. Súčasne je potrebné testovať odolnosť voči neúplným alebo zašumeným dátam, ako aj voči adversariálnym vstupom, ktoré môžu cielene ovplyvniť výstup modelu. Význam má tiež experimentovanie s architektúrou modelov, najmä využitie hybridných prístupov, ktoré kombinujú transformery s konvolučnými alebo inými typmi neurónových sietí. Dôležitou oblasťou ostáva interpretovateľnosť výstupov, keďže spoľahlivé a zrozumiteľné vysvetlenie rozhodnutia modelu je kľúčové pre jeho dôveryhodné nasadenie v praxi.

Kapitola 9

Záver

Cieľom tejto diplomovej práce bolo navrhnúť, implementovať a experimentálne vyhodnotiť riešenie na detekciu malígnych domén. V práci sú pod pojmom malígne domény rozlišované tri hlavné kategórie: DGA, malvér a phishing. Pre každú z týchto kategórií boli k dispozícii rôzne typy dát, konkrétne názvy domén, RDAP záznamy, dáta z DNS a geografické informácie. Výnimku tvorili DGA domény, pri ktorých boli dostupné len názvy doménových mien.

Na základe analýzy problému bol navrhnutý modulárny systém pozostávajúci z viacerých samostatných klasifikátorov. Pre každú kombináciu úlohy a typu dát bol vytvorený vlastný model, čo umožňuje ich flexibilné využitie, kombinovanie, a zároveň presnú interpretáciu prínosu jednotlivých dátových zdrojov.

Pred trénovaním modelov bola vykonaná rozsiahla analýza a čistenie dát, zameraná na identifikáciu relevantných atribútov, odstránenie chýbajúcich hodnôt a prípravu vstupov v textovej podobe. Experimentálny proces prebiehal v dvoch hlavných fázach. V prvej fáze prebiehalo vyhľadávanie optimálnych architektúr a nastavenie hyperparametrov, najmä nad dátami s doménovými menami, ktoré preukázali vysokú informačnú hodnotu. V druhej fáze boli vybrané architektúry trénované na všetkých dostupných množinách dát. Celkovo bolo navrhnutých a vyhodnotených 9 klasifikátorov, pričom v rámci procesu ladenia bolo natrénovaných až 33 rôznych architektúr.

Medzi najvýznamnejšie výsledky patrí skóre F1 98,61% pri detekcii DGA domén len na základe doménových mien. V prípade malvéru sa najlepšie výsledky dosiahli s DNS dátami (95,93%), zatiaľ čo pri phishingu poskytli najvyššiu presnosť RDAP dáta (98,39%). Tieto výsledky ukazujú, že pre detekciu malvéru sú rozhodujúce znaky v sieťovej prevádzke, zatiaľ čo pri phishingových doménach zohrávajú významnú úlohu registračné údaje.

Dosiahnuté výsledky prevyšujú väčšinu súčasných výskumov, ktoré často využívajú manuálne navrhnuté vlastnosti. V tejto práci sa však klasifikátory učili priamo z dát bez potreby manuálnej extrakcie vlastností (feature-less approach). Tento prístup znižuje závislosť od heuristík, zvyšuje flexibilitu a umožňuje rýchlu adaptáciu na nové typy útokov.

Neustále sa vyvíjajúce hrozby si vyžadujú priebežné zlepšovanie detekcie. Ďalšími krokmi sú testovanie odolnosti modelov voči zámerným manipuláciám, nasadenie v reálnom prostredí a prípadná hardvérová optimalizácia (napríklad pomocou FPGA). Význam má aj kombinovanie rôznych dátových zdrojov, skúmanie hybridných architektúr a posilnenie interpretovateľnosti modelov tak, aby poskytovali zrozumiteľné vysvetlenia svojich rozhodnutí pre sieťových správcov.

Literatúra

[1] Almashhadani, A. O.; Kaiiali, M.; Carlin, D. a Sezer, S. MaldomDetector: A system for detecting algorithmically generated domain names with machine learning. *Computers & Security*. Elsevier, 2020, zv. 93, s. 101787.

[2] Ba, J. L.; Kiros, J. R. a Hinton, G. E. *Layer Normalization*. 2016. Dostupné z: `https://arxiv.org/abs/1607.06450`.

[3] Buber, E.; Diri, B. a Sahingoz, O. K. NLP based phishing attack detection from URLs. In: Springer. *Intelligent Systems Design and Applications: 17th International Conference on Intelligent Systems Design and Applications (ISDA 2017) held in Delhi, India, December 14-16, 2017*. 2018, s. 608–618.

[4] Bučko, F. *Klasifikácia doménových mien generovaných algoritmami DGA*. Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Hranický, Ph.D.

[5] Cersosimo, M. a Lara, A. Detecting malicious domains using the splunk machine learning toolkit. In: IEEE. *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. 2022, s. 1–6.

[6] Clark, K.; Luong, M.-T.; Le, Q. V. a Manning, C. D. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In: ICLR. *International Conference on Learning Representations*. 2020.

[7] Cucchiarelli, A.; Morbidoni, C.; Spalazzi, L. a Baldi, M. Algorithmically generated malicious domain names detection based on n-grams features. *Expert Systems with Applications*. Elsevier, 2021, zv. 170, s. 114551.

[8] Dinaburg, A. Bitsquatting: DNS Hijacking Without Exploitation. In: *Black Hat USA Conference*. 2011. Dostupné z: `https://media.blackhat.com/bh-us-11/Dinaburg/BH_US_11_Dinaburg_Bitsquatting_Slides.pdf`. Presented at Black Hat USA 2011, Las Vegas, NV, USA.

[9] Du, K.; Yang, H.; Li, Z.; Duan, H.; Hao, S. et al. TL; DR hazard: a comprehensive study of levelsquatting scams. In: Springer. *Security and Privacy in Communication Networks: 15th EAI International Conference, SecureComm 2019, Orlando, FL, USA, October 23–25, 2019, Proceedings, Part II 15*. 2019, s. 3–25.

[10] Fasllija, E.; Enişer, H. F. a Prünster, B. Phish-Hook: Detecting phishing certificates using certificate transparency logs. In: Springer. *Security and Privacy in Communication Networks: 15th EAI International Conference, SecureComm 2019, Orlando, FL, USA, October 23–25, 2019, Proceedings, Part II 15*. 2019, s. 320–334.

[11] Geva, M.; Caciularu, A.; Wang, K. R. a Goldberg, Y. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *ArXiv preprint arXiv:2203.14680*, 2022.

[12] Giacaglia, G. How Transformers Work: The Neural Network used by OpenAI and DeepMind. *Towards Data Science*, March 2019. Dostupné z: `https://towardsdatascience.com/how-transformers-work-605ddf35d10f`. Accessed on January 11, 2025.

[13] GP, A. a Gladston, A. A machine learning framework for domain generating algorithm based malware detection. *Security and Privacy*. Wiley Online Library, 2020, zv. 3, č. 6, s. e127.

[14] Hamroun, C.; Amamou, A.; Haddadou, K.; Haroun, H. a Pujolle, G. A review on lexical based malicious domain name detection methods. *Annals of Telecommunications*. Springer, 2024, s. 1–17.

[15] Hason, N.; Dvir, A. a Hajaj, C. Robust malicious domain detection. In: Springer. *Cyber Security Cryptography and Machine Learning: Fourth International Symposium, CSCML 2020, July 2–3, 2020, Proceedings 4*. 2020, s. 45–61.

[16] Hranický, R.; Horák, A.; Polišenský, J.; Jeřábek, K. a Ryšavý, O. Unmasking the phishermen: phishing domain detection with machine learning and multi-source intelligence. In: IEEE. *NOMS 2024-2024 IEEE Network Operations and Management Symposium*. 2024, s. 1–5.

[17] Hranický, R.; Horák, A.; Polišenský, J.; Ondryáš, O.; Jeřábek, K. et al. Spotting the Hook: Leveraging Domain Data for Advanced Phishing Detection. In: *2024 20th International Conference on Network and Service Management (CNSM)*. 2024, s. 1–7.

[18] Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X. et al. TinyBERT: Distilling BERT for Natural Language Understanding. In: Association for Computational Linguistics. *Findings of the Association for Computational Linguistics: EMNLP 2020*. 2020, s. 4163–4174.

[19] Jurafsky, D. *Speech and language processing*. Prentice-Hall, 2000.

[20] Kim, S. a Lee, G. URLTran: Efficient Tokenization for URL Classification. In: ACM. *Proceedings of the 2021 Workshop on Cybersecurity and AI*. 2021, s. 45–55.

[21] Kintis, P.; Miramirkhani, N.; Lever, C.; Chen, Y.; Romero Gómez, R. et al. Hiding in plain sight: A longitudinal study of combosquatting abuse. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017, s. 569–586.

[22] Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P. et al. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In: ICLR. *International Conference on Learning Representations*. 2020.

[23] Lee, H.; Park, M. a Choi, J. TransURL: Transformer-Based URL Classification for Security Threat Detection. In: IEEE. *Proceedings of the 2022 IEEE Symposium on Security and Privacy*. 2022, s. 789–804.

[24] Li, H.; Xu, Z.; Taylor, G.; Studer, C. a Goldstein, T. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 2018, zv. 31.

[25] Loshchilov, I. a Hutter, F. Decoupled Weight Decay Regularization. In: *International Conference on Learning Representations (ICLR)*. 2019. Dostupné z: https://openreview.net/forum?id=Bkg6RiCqY7.

[26] Maklin, C. *Transformers Explained* online. Dostupné z: https://towardsdatascience.com/transformers-explained-445e5fcbbe59. [cit. 2025-01-11]. Published on August 22, 2022.

[27] Mankar, N. P.; Sakunde, P. E.; Zurange, S.; Date, A.; Borate, V. et al. Comparative Evaluation of Machine Learning Models for Malicious URL Detection. In: *2024 MIT Art, Design and Technology School of Computing International Conference (MITADTSoCiCon)*. 2024, s. 1–7.

[28] Mvula, P. K.; Branco, P.; Jourdan, G.-V. a Viktor, H. L. COVID-19 malicious domain names classification. *Expert Systems with Applications*. Elsevier, 2022, zv. 204, s. 117553.

[29] Nikiforakis, N.; Van Acker, S.; Meert, W.; Desmet, L.; Piessens, F. et al. Bitsquatting: Exploiting bit-flips for fun, or profit? In: *Proceedings of the 22nd international conference on World Wide Web*. 2013, s. 989–998.

[30] Niu, Y.; Guan, M.; Yuan, W.; Chen, Y.; Chen, L. et al. A Bayesian optimization-based LSTM model for DGA domain name identification approach. In: IOP Publishing. *Journal of Physics: Conference Series*. 2022, sv. 2303, č. 1, s. 012015.

[31] Palaniappan, G.; Sangeetha, S.; Rajendran, B.; Goyal, S.; Bindhumadhava, B. et al. Malicious domain detection using machine learning on domain name features, host-based features and web-based features. *Procedia Computer Science*. Elsevier, 2020, zv. 171, s. 654–661.

[32] Plohmann, D.; Yakdan, K.; Klatt, M.; Bader, J. a Gerhards Padilla, E. A comprehensive measurement study of domain generating malware. In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, s. 263–278.

[33] Polišenský, J. *Porovnání klasifikačních metod pro účely detekce maligních domén*. Brno, 2024. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Hranický, Ph.D.

[34] Rothman, D. a Gulli, A. *Transformers for Natural Language Processing: Build, train, and fine-tune deep neural network architectures for NLP with Python, Hugging Face, and OpenAI's GPT-3, ChatGPT, and GPT-4*. 2. vyd. Packt Publishing, 2022. ISBN 978-1803247335.

[35] Sadique, F.; Kaul, R.; Badsha, S. a Sengupta, S. An automated framework for real-time phishing URL detection. In: IEEE. *10th CCWC Conference*. 2020, s. 0335–0341.

[36] Sanh, V.; Debut, L.; Chaumond, J. a Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv preprint arXiv:1910.01108*, 2019.

[37] Schüppen, S.; Teubert, D.; Herrmann, P. a Meyer, U. {FANCI}: Feature-based automated {NXDomain} classification and intelligence. In: *27th USENIX Security Symposium (USENIX Security 18)*. 2018, s. 1165–1181.

[38] Selvi, J.; Rodríguez, R. J. a Soria Olivas, E. Detection of algorithmically generated malicious domain names using masked N-grams. *Expert systems with applications*. Elsevier, 2019, zv. 124, s. 156–163.

[39] Shi, Y.; Chen, G. a Li, J. Malicious domain name detection based on extreme machine learning. *Neural Processing Letters*. Springer, 2018, zv. 48, s. 1347–1357.

[40] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L. et al. Attention is all you need. In: Neural Information Processing Systems Foundation. *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017, s. 5998–6017. NIPS'17. ISBN 9781510860964.

[41] Vinayakumar, R.; Soman, K. a Poornachandran, P. Detecting malicious domain names using deep learning approaches at scale. *Journal of Intelligent & Fuzzy Systems*. Ios Press, 2018, zv. 34, č. 3, s. 1355–1367.

[42] Xu, C.; Shen, J. a Du, X. Detection method of domain names generated by DGAs based on semantic representation and deep neural network. *Computers & Security*. Elsevier, 2019, zv. 85, s. 77–88.

[43] Yadav, S.; Reddy, A. K. K.; Reddy, A. N. a Ranjan, S. Detecting algorithmically generated malicious domain names. In: *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. 2010, s. 48–61.

[44] Yang, L.; Liu, G.; Dai, Y.; Wang, J. a Zhai, J. Detecting stealthy domain generation algorithms using heterogeneous deep neural network framework. *Ieee Access*. IEEE, 2020, zv. 8, s. 82876–82889.

[45] Yang, L.; Liu, G.; Wang, J.; Zhai, J. a Dai, Y. A semantic element representation model for malicious domain name detection. *Journal of Information Security and Applications*. Elsevier, 2022, zv. 66, s. 103148.

[46] Zhang, P.; Liu, T.; Zhang, Y.; Ya, J.; Shi, J. et al. Domain watcher: detecting malicious domains based on local and global textual features. *Procedia Computer Science*. Elsevier, 2017, zv. 108, s. 2408–2412.

[47] Zhang, Y.; Wu, Y. a Jin, S. Which DGA Family does A Malicious Domain Name Belong To. In: IEEE. *2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC)*. 2020, s. 53–60.

[48] Zhao, H.; Chang, Z.; Bao, G. a Zeng, X. Malicious Domain Names Detection Algorithm Based on N-Gram. *Journal of Computer Networks and Communications*. Wiley Online Library, 2019, zv. 2019, č. 1, s. 4612474.

[49] Zhauniarovich, Y.; Khalil, I.; Yu, T. a Dacier, M. A survey on malicious domains detection through DNS data analysis. *ACM Computing Surveys (CSUR)*. ACM New York, NY, USA, 2018, zv. 51, č. 4, s. 1–36.

[50] Zheng, A. *Evaluating machine learning models: a beginner's guide to key concepts and pitfalls*. O'Reilly Media, 2015.

Príloha A

Obsah priloženého pamäťového média

Štruktúra adresárov na priloženom pamäťovom médiu je rozdelená nasledovne:

- **dp** – Tento adresár obsahuje finálnu verziu diplomovej práce vo formáte `PDF`, ako aj všetky zdrojové súbory v jazyku LaTeX, ktoré boli použité na jej zostavenie.
- **transformers** – V tomto priečinku sa nachádza kompletná implementácia riešenia. Obsahuje všetky skripty na trénovanie a vyhodnocovanie modelov, uložené modely, potrebné konfiguračné súbory a súvisiace dáta. Súčasťou adresára sú aj klasifikátory a experimenty realizované počas vývoja. Detailná štrukúra bola popísana v kaptile 6
- **datasets** – Adresár obsahuje všetky dátové sady použité v projekte. Nachádzajú sa tu pôvodné (nespracované) domény generované algoritmami DGA a záznamy legitímnych domén, ako aj predspracované verzie týchto dát, ktoré boli následne použité pri trénovaní modelov.

Príloha B

Manuál

V rámci tejto prílohy je popísaná inštalácia všetkých potrebných závislostí pre úspešné spustenie programu. Následne je uvedený odporúčaný postup vykonania experimentov a trénovania modelov. V prípade akýchkoľvek komplikácií počas inštalácie alebo spúšťania programu je detailnejší návod uvedený v súbore README.md v koreňovom adresári projektu. Na spustenie projektu je potrebné disponovať výpočtovým hardvérom s podporou CUDA – GPU kompatibilnou s technológiou NVIDIA CUDA. Celý inštalačný a spúšťací postup je navrhnutý pre prostredie Conda, ktoré umožňuje oddelenie závislostí a zabezpečuje reprodukovateľnosť experimentov.

B.1 Inštalácia

Projekt využíva dve samostatné prostredia `Conda` – jedno určené pre trénovanie modelov na GPU (`gpu`) a druhé pre proces hľadania hyperparametrov (`hpsearch`). Konfigurácie týchto prostredí sú uložené v adresári `envs/` vo forme súborov `gpu.yaml` a `hpsearch.yaml`.

1. Inštalácia prostredí z YAML súborov

V koreňovom adresári projektu je možné vytvoriť potrebné prostredia pomocou nasledujúcich príkazov:

```
conda env create -f envs/gpu.yaml
conda env create -f envs/hpsearch.yaml
```

Týmto sa vytvoria dve izolované prostredia s názvami `gpu` a `hpsearch`, ktoré obsahujú všetky potrebné knižnice a závislosti podľa definície v príslušných YAML súboroch.

2. Aktivácia prostredia

Po vytvorení nie je prostredie automaticky aktivované. Pred spustením ktoréhokoľvek zo skriptov je preto potrebné manuálne aktivovať požadované prostredie:

```
conda activate gpu        # pre trénovanie modelov
conda activate hpsearch   # pre spúšťanie ladenia parametrov
```

3. Overenie inštalácie

Po aktivácii prostredia je možné overiť zoznam nainštalovaných balíkov pomocou príkazu:

```
conda list
```

B.2 Spúšťanie experimentov

Po úspešnej inštalácii prostredí je možné spúšťať jednotlivé skripty určené na trénovanie modelov a vykonávanie experimentov. Každý skript je navrhnutý pre konkrétny účel a viaže sa na jedno z dvoch prostredí `Conda`.

1. Priradenie skriptov k prostrediam

- **Prostredie `gpu`** – určené na trénovanie modelov:
  - `experiments/custom_architecture_experiment.py`
  - `experiments/pretrained_experiment.py`
- **Prostredie `hpsearch`** – určené na hľadanie hyperparametrov:
  - `experiments/hyperparameter_search.py`

2. Tréning s predtrénovanými modelmi

Pre spustenie experimentov využívajúcich predtrénované modely (napr. BERT, DistilBERT, Electra) slúži skript:

```
python -m experiments.run_experiment
```

Tento príkaz načíta predvolenú konfiguráciu zo súboru `configs/pretrained_config.yaml`, ktorá definuje výber dát, typ úlohy, architektúru modelu a parametre trénovania. Parametre je možné upravovať priamo z príkazového riadku pomocou syntaxe knižnice `Hydra`. Napríklad zmenu veľkosti dávky a výber iného modelu je možné nastaviť nasledovne:

```
python -m experiments.run_experiment train.batch_size=512 \
                                     model=google_electra_base
```

Pre spustenie konkrétnej experimentálnej konfigurácie je možné použiť parameter `-m`. Napríklad:

```
python -m experiments.run_experiment -m experiments=rdap_dns_geo
```

3. Tréning vlastnej architektúry

Pre trénovanie vlastnej transformerovej architektúry je určený skript:

```
python -m experiments.custom_architecture_experiment
```

Konfigurácia sa nachádza v súbore `configs/custom_config.yaml`. Spôsob tokenizácie je možné zvoliť cez parametre príkazového riadku ako v predchádzajúcom prípade.

4. Hľadanie hyper-parametrov

Automatické ladenie hyperparametrov sa vykonáva pomocou skriptu:

```
python -m experiments.hyperparameter_search
```

Skript je navrhnutý ako samostatný skript bez využitia externých konfiguračných systémov. Pred spustením je možné upraviť skript v nasledujúcich dvoch bodoch:

- **Výber kategórie domén**, nad ktorou sa vykonáva ladenie (DGA, malvér, phishing):

  Hodnota premennej `data_path` sa upravuje podľa toho, kde sa v konkrétnom systéme nachádza priečinok `datasets`, ktorý obsahuje vstupné CSV súbory s doménami.

  ```
  data_path = "datasets/dga/dga_preprocessed.csv"

  # data_path = "datasets/malware/malware_preprocessed.csv"
  # data_path = "datasets/phishing/phishing_preprocessed.csv"
  ```

- **Voľba modelovej architektúry**, ktorá sa má optimalizovať:

  Architektúra sa určuje cez hodnotu premennej `MODEL_NAME`:

  ```
  MODEL_NAME = "prajjwal1/bert-medium"

  # MODEL_NAME = "prajjwal1/bert-tiny"
  # MODEL_NAME = "distilbert-base-uncased"
  ```

Týmto spôsobom je možné spúšťať všetky časti projektu – od základného trénovania modelov cez experimentovanie s vlastnými architektúrami až po systematické ladenie parametrov.