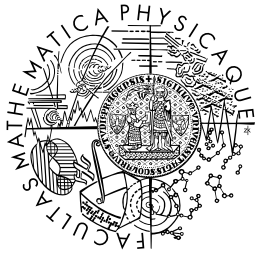


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

DIPLOMOVÁ PRÁCE

Bc. Patrik Romanský

Automatická detekcia fake-news v slovenských textoch

Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. David Mareček, Ph.D.
Studijní program: Informatika
Studijní obor: Umělá inteligence

Praha 2023

Prohlašuji, že jsem tuto diplomovou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Chcel by som vyjadriť veľkú vďaku všetkým, ktorí mi pomohli pri tvorbe tejto diplomovej práce. V prvom rade by som sa chcel poďakovať vedúcemu práce RNDr. Davidovi Marečkovi, Ph.D. za jeho odborné vedenie, rady a podporu počas celej práce. Ďalej by som sa chcel poďakovať Technickej univerzite v Košiciach a tímu Sarnovského za ochotu poskytnúť nám dátovú sadu a tým umožniť experimentovanie v slovenskom jazyku.

Taktiež sa chcem veľmi pekne poďakovať svojej rodine a všetkým milovaným, ktorí mi poskytli obrovskú podporu a podnietili ma k štúdiu na Matematicko-Fyzikálnej Fakulte Univerzity Karlovej, čo viedlo k napísaniu tejto diplomovej práce. Rovnako by som im chcel vyjadriť obrovskú vďaku za nepretržitú oporu a pomoc pri písaní a korektúre tejto práce. Osobitná vďaka patrí mojim najbližším: rodičom a sestre Emke, bez ktorých by som to nezvládol.

Nakoniec, no rozhodne nie s menšou dôležitosťou, by som chcel vyjadriť vďaku všetkým mojim priateľom, ktorí ma sprevádzajú na univerzite. Špeciálne by som sa chcel poďakovať najlepšiemu priateľovi a spolužiakovi Matúšovi, ktorého som spoznal na vysokej škole. Boli sme si navzájom oporou pri štúdiu a spoločne sme prežili najkrajšie chvíle mladosti. Tiež mu patrí veľká vďaka za pomoc s korektúrou tejto diplomovej práce.

Název práce: Automatická detekcia fake-news v slovenských textoch

Autor: Bc. Patrik Romanský

Katedra: Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. David Mareček, Ph.D., Ústav formální a aplikované lingvistiky

Abstrakt: Šírenie fake-news je dlhodobým problémom, ale v posledných rokoch sa stáva ešte výraznejším. Preto sme v tejto práci analyzovali problém ich automatickej detekcie ako úlohu klasifikácie textu. Práca sa od iných, jej podobných štúdií, odlišuje primárne v tom, že sa zameriava na slovenčinu, kde doposiaľ nebola vykonaná takáto rozsiahla sada experimentov. Počas testov sme vytvorili vybalansovaný dataset. Vykonali sme taktiež viac ako 80 experimentov s cieľom nájsť optimálny klasifikátor pre riešenie tohto problému. Ako prvý sme použili predtrénované jazykové modely typu Transformer (BERT, mBERT, RoBERTA, XLM-RoBERTa a SlovakBERT) a pomocou štandardných metrík sme porovnali ich výkonnosť s inými metódami strojového učenia. Pre fine-tuning sme použili aj anglické datasety LIAR a COVID19 FN, na ktorých sme otestovali vplyv témy fake-news a prenos vlastnosti medzi jazykmi. Najlepšie výsledky dosiahol SlovakBERT v kombinácii s tréningom na výlučne slovenskom datasete ($acc = 0{,}9610$).

Klíčová slova: falošné správy, hoax, NLP, SlovakBERT,

Title: Automatic detection of fake-news on Slovak texts

Author: Bc. Patrik Romanský

Department: Institute of Formal and Applied Linguistics

Supervisor: RNDr. David Mareček, Ph.D., Institute of Formal and Applied Linguistics

Abstract: Fake news is a problem in recent years. This study focuses on detecting fake news written in the Slovak language using text classification methods. It is unique because it is the first to conduct such a comprehensive set of experiments on Slovak. During the study, a balanced dataset was created, and over 80 experiments were conducted to find the optimal classifier for the problem. Pre-trained transformer-based language models, including BERT, mBERT, RoBERTA, XLM-RoBERTa, and SlovakBERT, were used in the initial step of the study, and their performance was compared against other machine learning methods using standard metrics. The models were fine-tuned with LIAR and COVID19 FN, English-language datasets, to test the impact of fake news topics and language transfer properties. SlovakBERT combined with training exclusively on Slovak datasets achieved the best results with an ($acc = 0.9610$). This study can contribute to the development of tools to automatically detect fake news in Slovak, aiding in the fight against the spread of false information.

Keywords: fake-news, hoax, NLP, SlovakBERT

Obsah

Úvod

V súčasnosti je veľké množstvo informácií dostupných prostredníctvom tradičných médií a sociálnych sietí. Výhody digitálnej éry a vysokorýchlostného internetu prinášajú množstvo príležitostí, ale zároveň aj riziká. Jedným z najväčších nebezpečenstiev, ktoré môžu vyplynúť z nesprávneho a zavádzajúceho šírenia informácií, sú falošné správy (angl. fake-news). Tieto správy môžu mať vplyv na verejnú mienku a dokonca ovplyvniť výsledky politických volieb a spôsobiť značné škody na povesti jednotlivcov alebo inštitúcií.

S narastajúcim množstvom falošných správ sa stáva čoraz dôležitejšie mať k dispozícii nástroje, ktoré dokážu automaticky detegovať tieto nepravdivé informácie. V súčasnosti existuje množstvo technológií a algoritmov, ktoré sa používajú na detekciu fake-news v rôznych jazykoch. Avšak väčšina týchto riešení sa zameriava na anglický jazyk a pre slovenské jazykové prostredie nie sú úplne vhodné. Preto je potrebné vytvoriť model, ktorý dokáže efektívne detegovať falošné správy v slovenských textoch.

Vzhľadom na vyššie uvedené faktory sa stalo naliehavou úlohou vytvoriť automatické nástroje pre slovenský jazykový korpus, ktoré dokážu odhaliť a klasifikovať falošné správy. V tomto kontexte je veľmi dôležité zabezpečiť, aby sa falošné správy identifikovali a odstránili čo najrýchlejšie. Ak zoberieme do úvahy množstvo článkov, ktoré sa dnes šíria pomocou internetu, nie je dostatok času na manuálne overovanie pravdivosti informácií. Preto je dôležité mať k dispozícii automatické nástroje, ktoré umožňujú efektívne riešenie tohto problému.

V digitálnom veku je šírenie dezinformácií lacnejšie a jednoduchšie. V minulosti iba veľké médiá s dostatočnými zdrojmi boli schopné osloviť veľké publikum. Dnes môže byť každý producentom pre široké masy [1]. Preto sa v tejto práci zameriame na tento moderný fenomén šírenia falošných správ a ich automatickej detekcii pomocou metód strojového učenia.

Cieľ práce

Táto práca skúma problém odhaľovania falošných správ týkajúcich sa pandémie covidu-19 pomocou klasifikácie textu. Cieľom je zistiť, či je možné identifikovať falošné správy výhradne na základe charakteristík použitého jazyka. Zameriava sa na porovnanie úspešnosti rôznych algoritmov strojového učenia pri testovaní na slovenských dátových sadách.

1. Vytvorenie alebo získanie slovenského datasetu, ktorý v sebe bude obsahovať dáta Covid-19 Fake-News.

2. Klasifikácia Covid-19 datasetu s aplikovaním poznatkov z iných jazykov.

3. Porovnanie rôznych klasifikačných nástrojov vzhľadom na ich úspešnosť a časovú náročnosť tréningu.

1. Súvisiace práce

1.1 COVID-19 Fake-News datasety

Pre túto prácu najrelevantnejším datasetom, ktorý je dostupný na internete, je SlovakCovid19 FN dataset, ktorý vytvoril Sarnovský a kolektív [2]. Tento dataset je jedným z mála dostatočne rozsiahlych datasetov pre detekciu falošných správ v slovenskom jazyku. Dátová sada pozostáva z 13 736 článkov, z ktorých je 851 nepravdivých. Dataset bol anotovaný automaticky a následne manuálne 5 hodnotiteľmi. Obsahuje iba textové údaje. Téma falošných správ, ktoré tvoria tento dataset, sa vzťahuje na pandémiu covidu-19, preto aj zahraničné datasety boli vyhľadávané z tejto oblasti.

Podľa informácií, ktoré sme získali, existujú štyri vhodné a verejne dostupné datasety údajov na detekciu falošných správ v angličtine, z toho tri v oblasti covid-19. Datasety sa od seba odlišujú typom zdroja, z akého pochádzajú (sociálne siete, novinové články), dĺžkou textových záznamov, nerovnováhou rozdelenia dát do tried a typom informácie dostupnej pre klasifikáciu (jazykový/sociálny kontext alebo oba typy informácie). Dátové sady uvedené v tejto kapitole sú prehľadne uvedené v tabuľke 1.1.

Dataset s názvom COVID19 FN, ktorý bol vyvinutý Patwom a jeho tímom [3], predstavuje prvý zahraničný zdroj na detekciu falošných správ o covide-19. Obsahuje 5 600 overených tweetov z dôveryhodných zdrojov, ako sú vládne účty, lekárske inštitúcie a spravodajské kanály, ako aj 5 100 manuálne overených nepravdivých tvrdení z rôznych zdrojov, vrátane novinových článkov, tlačových správ a príspevkov na sociálnych sieťach.

FakeCovid je ďalším datasetom určeným pre úlohu klasifikácie covid-19 fake-news [4]. Na rozdiel od predchádzajúceho datasetu je viacjazyčný. Pozostáva z 5182 manuálne kontrolovaných spravodajských článkov zo 40 jazykov, z ktorých 2116 je v angličtine a 47 v nemčine.

Ďalším datasetom pre angličtinu je CoAID [5]. Obsahuje rôzne typy správ, t. j. tvrdenia, spravodajské články a príspevky na sociálnych sieťach. Konkrétne pozostáva z 255 falošných a 3996 skutočných a overených spravodajských článkov viažucich sa na pandémiu, 28 nesprávnych a 454 pravdivých tvrdení o víruse, 926 príspevkov na sociálnych sieťach, ktoré reagujú na tieto články. Príspevky na sociálnych sieťach, t.j. tweety, obsahujú dodatočné informácie o správach, na ktoré reagujú, ako aj údaje o používateľovi, plus interakcie s ostatnými tweetmi. Keďže až 94% dát predstavuje autentické správy, je tiež najnevyváženejším z datasetov. Označenia pravdivosti pre skutočné spravodajské články boli získané na úrovni vydavateľov, t. j. údaje boli zozbierané od spoľahlivých vydavateľov, ako sú ScienceDaily alebo WHO. Falošné články boli zhromaždené pomocou odkazov na nepravdivé články z webových stránok overujúcich fakty. V prípade tvrdení sa na identifikáciu vyhlásení, o ktorých je známe, že sú nepravdivé, použili informácie WHO a ďalšie spoľahlivé zdroje.

Posledným z anglických datasetov, ktorý chceme predstaviť, je LIAR. Dátová sada bola vytvorená výskumníkmi Kalifornskej univerzity v Berkeley a má slúžiť na tréning a hodnotenie modelov, ktoré dokážu rozpoznať falošné správy [6]. Tento dataset sa od ostatných odlišuje témou článkov, ktorou sú voľby v USA.

Názov	Jazyk	Celkovo	Pravdivé	Falošné
Covid19 FN	anglický	10700	5600	5100
FakeCovid	viacjazyčný	5182	4146	1036
CoAID	anglický	4251	3996	255
LIAR	anglický	12836	8318	4518
Slovak-FakeNews	slovenský	13736	12885	851

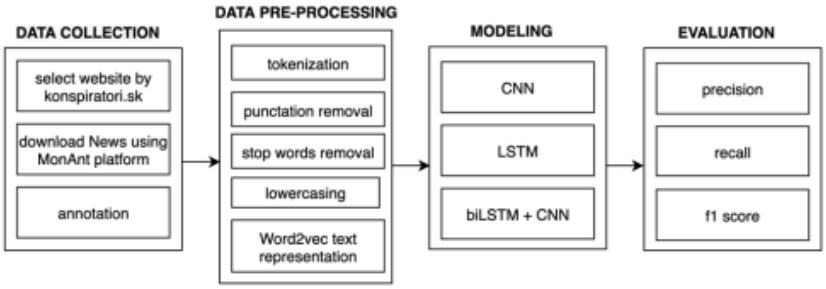
Tabuľka 1.1: Porovnanie datasetov.

Nielen že sú existujúce súbory údajov vzácne, ale sú tiež obmedzené, pokiaľ ide o množstvo textu a typy informácií, ktoré poskytujú, a niekedy sú veľmi nevyvážené, čo spôsobuje skreslenie.

1.2 Detekcia COVID-19 Fake-News

Vzhľadom na povahu dostupných datasetov o covide-19 bola väčšina pokusov o odhalenie falošných správ v tejto doméne založená na jazyku. Prístupom s najlepšími výsledkami v zdieľanej úlohe *COSTRAINT 2021 Workshop on Combating Online Hostile Posts in Regional Language during Emergency Situation* na datasete o falošných správach o covide-19 navrhli Glazkova a kolektív [7], ktorí získali textové reprezentácie pomocou jazykového modelu COVID-Twitter-BERT [8], ktorý bol špeciálne natrénovaný na 160 miliónoch anglických tweetov o víruse a dosiahol na ňom F1 skóre 0,987. O niečo nižší, ale tiež vysoký detekčný výkon (F1 skóre 0,967) sa dosiahol aj na základe reprezentácií zreťazených textov získaných pomocou XLNet [9] a Latent Dirichlet Allocation (LDA) v doprednej neurónovej sieti [10]. Bližší pohľad na ostatné prístupy riešenia zdieľanej úlohy *CONSTRAINT* sú vysvetlené v priložených článkoch. Experimenty vykonané na súbore údajov FakeCovid používali vopred pripravený model BERT na reprezentáciu textového obsahu anglických spravodajských článkov z FakeCovid a získali F1 skóre 0,780 [4]. Justus Mattern a kolektív vytvorili rozsiahly dataset FANG-COVID v nemeckom jazyku, na ktorom uskutočnili množstvo experimentov. Vzhľadom na charakter datasetu, ktorý obsahuje textové dáta aj sociálnu interakciu, boli tieto experimenty zamerané na vplyv použitých vlastností. Najlepšie výsledky dosiahol model BERT plus sociálny kontext [11], ktorý využíva textové dáta aj dodatočné informácie o tvorcovi obsahu. Z výsledkov najdôležitejších vlastností vyplynulo, že oveľa dôležitejší je tvorca správy ako samotný text. To potvrdzuje, že ľudia s istým zmýšľaním majú vyššiu tendenciu šíriť nepravdivé správy. Cui a Lee uskutočnili experimenty so súborom údajov CoAID pomocou rôznych klasifikačných modelov [5], ktoré sa ukázali ako úspešné pri všeobecnej detekcii falošných správ. Najmä niektoré zo všeobecne najvýkonnejších modelov na detekciu falošných správ, ako sú modely CSI [12] a SAME [5], sa ukázali ako menej účinné pri CoAID, pričom dosiahli skóre F1 0,228 a 0,340 v porovnaní s najvýkonnejším modelom dEFEND s F1 skóre 0,581 [13].

Pre slovenčinu sa v oblasti covid-19 fake-news zatiaľ žiaden hlbší výskum nevykonal. Sarnovský a kolektív publikovali dataset a vo svojej publikácii sa venovali základnému výskumu klasifikácie falošných správ [2]. Po vytvorení datasetu sa v experimentálnej časti práce zamerali na modely hlbokých neurónových sietí.



Obr. 1.1: Postup práce Sarnovského a kolektívu [2].

V prvej fáze bol slovenský vstupný dataset rozdelený na trénovaciu a testovaciu množinu. Trénovacia množina obsahovala 9 615 dokumentov, z ktorých 9 000 bolo pravdivých a 615 bolo falošných správ. Z trénovacej množiny boli vyčlenené validačné dáta, ktoré tvorili 10% trénovacej množiny. Na celkové vyhodnotenie experimentov Sarnovský a kolektív použili testovaciu množinu, ktorá sa skladala z 4 121 dokumentov. Z nich bolo 3 901 pravdivých správ a 220 falošných správ.

Na ilustrácii 1.1 vo fáze testovania sa konkrétne porovnávali modely CNN, LSTM (RNN) a biLSTM + CNN, ktoré vykazujú vysokú úspešnosť v riešení podobných úloh. Výsledné porovnanie jednotlivých modelov sa uskutočnilo pomocou metrík Precision, Recall a F1 skóre. Najlepšie výsledky dosiahla sieť biLSTM + CNN pri použití dropout hodnoty 0,1, batch size 32 a optimalizátora Adam. S týmto modelom Sarnovský a kolektív dosiahli najlepšie výsledky pomocou metódy fine-tuning (grid search), ktorý dosiahol Precision 0,9893 a F1 skóre 0,94 na testovacej množine. Detailnejšie výsledky sú uvedené v tabuľke 1.2.

	Precision	Recall	F1 skóre
Regular News	0,99	1,00	0,99
Fake News	0,95	0,82	0,89
Accuracy			0,99
Macro avg	0,97	0,92	0,94
Weighted avg	0,99	0,99	0,99

Tabuľka 1.2: CNN+biLSTM model po optimalizácii [2].

Aj keď výsledky tejto práce znejú sľubne, je dôležité zdôrazniť, že sú veľmi ovplyvnené kvalitou dát, spôsobom čistenia a anotácie. V niektorých aspektoch práce nie je dostatočne presne popísané, ako sa riešili časti textov, ktoré nesúvisia priamo s obsahom článku, ako napríklad úvodné alebo záverečné vety špecifické pre konkrétne médiá. Samotní autori uznávajú, že výsledky sú výrazne ovplyvnené formou dát a nevyváženosťou v dátovej sade. Taktiež odporúčajú ďalší výskum v oblasti detekcie falošných správ.

Napokon aj predošlé zistenia zahraničných štúdií potvrdzujú, že všeobecné modely pre anglický jazyk nemusia fungovať na tejto špecifickej tematike falošných správ, a nedostatok experimentov pre iné jazyky iba zdôrazňujú potrebu výskumu detekcie falošných správ v špecifických kritických situáciách, ako je napríklad pandémia.

2. Fake-news

Táto kapitola poskytuje stručný úvod do problematiky fake-news, ktorou sa táto práca zaoberá.

Obr. 2.1: Vývoj vyhľadávania termínov súvisiacich s témou fake-news pomocou Google search za obdobie od 01. 05. 2014 po súčasnosť. Predošlé obdobie je skoro identické prvej tretine grafu.

Koncept *„Fake-News“* sa začal často používať po voľbách v USA v roku 2016 [14], nárast je pekne viditeľný na obrázku 2.1, ktorý zobrazuje množstvo otázok na Google vyhľadávač k téme fake-news. Dve samostatné vyšetrovania Guardian a BuzzFeed identifikovali najmenej 140 webových stránok produkujúcich falošné správy ohľadom volieb zamerané na občanov USA, pričom všetky boli z malého mesta v Macedónsku.

Najvyššiu hodnotu dosiahol tento fenomén v marci 2020, keď prebiehala pandémia covidu-19. V tejto už aj tak ťažkej dobe sa začali šíriť dezinformácie a fake-news ešte rýchlejším tempom. Predpokladaná príčina spočívala v tom, že zúfalí ľudia v psychicky vypätom období sú náchylnejší na dezinformovanie. Po ustálení situácie s pandémiou sa znižuje množstvo fake-news. Ale ako môžeme vidieť na priebehu grafu, znovu narastajú, čo pravdepodobne bude mať súvis s ďalšou kritickou situáciou, tou je vojna na Ukrajine. Priamy súvis medzi fake-news a kritickými situáciami vo svete nie je zatiaľ dokázaný. Ale fakty nasvedčujú tomu, že sa to v blízkej budúcnosti dokáže.

V kritických situáciách už sú životu nebezpečné, pretože ľudia v psychickom vypätí strácajú kritické myslenie a sú náchylnejší na oklamanie pomocou fake-news. Fake-news robia problémy aj v pokojných dobách. Preto sme sa v tejto práci rozhodli zamerať práve na fake-news z obdobia covidu-19.

2.1 Informačná bublina

Ako sme už spomenuli v úvode, úzko súvisiacim problémom spojeným s termínom fake-news je fenomén informačnej bubliny. Informačná bublina je označenie pre výsledok tzv. personalizovaného obsahu. Pri takejto personalizácii algoritmy webových stránok filtrujú informácie s cieľom zobraziť užívateľovi len tie, ktoré chce vidieť (takže predovšetkým tie, ktoré korešpondujú s jeho názormi) zatiaľ čo od ostatných informácií ho to izoluje. Tieto algoritmy vychádzajú z vyhodnotenia predchádzajúceho správania užívateľa na internete a sú používané napríklad stránkami Facebook, Yahoo News alebo Google Search [15].

Personalizovanie produktov pre užívateľov sa javí ako prospešné. Ale ak dôjde k personalizácii diskusných príspevkov alebo politických, náboženských, sociál-

nych a mnohých iných názorov, môže to viesť k radikalizácii. Hlavne v kontexte, ak si užívateľ neuvedomuje, že informácie, s ktorými prichádza do kontaktu, sú filtrované. Pretože používateľa izoluje od informácií, ktoré by u neho mohli viesť k novému zhodnoteniu problému a väčšej názorovej diverzifikácii. Personalizácia vyhľadávania naopak používateľa ešte hlbšie utvrdzuje v jeho presvedčení. Dôsledkom tohto javu môže byť dezinformovanosť a manipulovateľnosť užívateľov pomocou fake-news [16].

Na záver musíme ešte podotknúť, že situácia sa zlepšovať nebude, ak s tým niečo neurobíme. Ústup klasických médií a nástup šírenia informácií pomocou internetu je nezastaviteľný. Internet prekonal v roku 2018 všetky ostatné mediá v množstve šírenia nových informácií. Preto je v dnešnej dobe otázka správnej a rýchlej detekcie šírenia falošných správ ešte aktuálnejšia ako kedykoľvek predtým.

2.2 Boj Európskej únie proti dezinformáciám

Európska únia si je vedomá sily šírenia dezinformácií, a preto má zriadenú diplomatickú službu East StratCom Task Force. V roku 2015 bol v rámci tejto diplomatickej služby vytvorený tím, ktorý má za úlohu zaoberať sa prebiehajúcimi prokremeľskými dezinformačnými kampaňami a ich vplyvom na politické zriadenia v Európskej únii.

East StratCom Task Force zverejnil analýzu s názvom *„East StratCom Task Force The legal framework to address "fake-news": possible policy actions at the EU level“* [1]. Práca popisuje šírenie dezinformácií a sumarizuje kľúčové zistenia. Vrátane klesajúcej dôvery v internet, nízkej schopnosti jeho užívateľov vyčleniť pravdivé správy hodné ich pozornosti. Ďalšie zistenia boli, že mnohé falošné a manipulatívne správy sú ignorované zodpovednými inštitúciami a niektoré časti z uvedených správ stále existujú na internete. Opätovne sa tak môžu rýchlo šíriť a ovplyvňovať verejnú mienku v Európskej únii. Pretože tým vytvárajú komunikačný šum, že poskytujú protichodné informácie k jednej udalosti. Čo v konečnom dôsledku spôsobuje polarizáciu spoločnosti a zmätok v rozhodovaní.

2.3 Fake-news na Slovensku

Falošné správy, tiež známe ako „fake-news“, sú vážnym problémom na Slovensku rovnako ako aj v mnohých iných krajinách sveta [2].

V Slovenskej republike sa stretávame so zvýšenou aktivitou krajne pravicového politického spektra, ktoré systematicky šíri dezinformácie, najmä v súvislosti s EÚ a NATO. Počas prvej vlny pandémie covidu-19 sa ich úsilie sústredilo najmä na šírenie nepravdivých informácií o výhodách, ktoré majú Rómovia, a o údajnej výraznej starostlivosti štátu o túto komunitu aj počas pandémie. Ich cieľom bolo zvýšiť polarizáciu spoločnosti. Okrem toho sa snažili manipulovať s údajmi o počte nakazených a šírili dezinformácie o rizikách spojených s očkovaním, s cieľom podkopať dôveru v tradičné médiá, ktoré odkrývajú ich krajne pravicové postoje. Toto všetko robili s cieľom získať popularitu a zlepšiť svoj verejný obraz [17].

Samostatnou skupinou sú platení šíritelia falošných správ na Slovensku. Títo ľudia neváhajú šíriť akékoľvek informácie a propagovať ich, ak za to majú zapla-

tené. Zarážajúce je, že niektoré z týchto médií sú označované verejnosťou za relevantné informačné zdroje. Vzhľadom na malú veľkosť Slovenska je táto situácia ešte závažnejšia, pretože je tu relatívne vysoký počet platených šíriteľov falošných správ. Tento jav predstavuje významný problém s negatívnymi dôsledkami pre spoločnosť.

Aj na Slovensku už začínajú projekty zamerané na osvetu k téme falošných správ, ale zatiaľ sú to len začiatky. Automatické systémy na fact-checking zatiaľ neexistujú a fyzickým Facebook fact-checkerom na Slovensku je jedna osoba, ktorá má právo označiť na tejto platforme hoax alebo falošné správy [18].

Aj z toho dôvodu sú potrebné automatické systémy určené na detekciu fake-news pre slovenské texty.

2.4 Definícia deziformácie

Cieľom tejto práce nie je riešiť filozofickú rovinu definície fake-news. Ale pre lepšie pochopenie problému, ktorý sa snažíme vyriešiť pomocou NLP, je vhodná definícia základného pojmu. Podľa definície New York Times sú fake-news správy, ktoré vznikli so zámerom oklamať čitateľa. Na druhej strane tím EÚ, ktorý bojuje proti šíreniu dezinformácií v Európskej únii, má zadefinované pod týmto pojmom správy s jasným zámerom manipulovať verejnú mienku [1].

V tejto práci nebudeme skúmať úmysel, s akým bola správa vytvorená. Či vznikla omylom alebo zámerne. Preto definujeme všetky nepravdivé správy ako fake-news.

3. Detekcia Fake-news

Detekciu falošných správ a dezinformácií šírených na internete je možné rozdeliť na niekoľko podproblémov, ktoré je možné riešiť samostatne alebo ako ich kombináciu. V tejto kapitole popíšeme možnosti automatizácie riešenia jednotlivých podproblémov, ktoré už boli vykonané. Na druhej strane popíšeme aj možnosť využitia strojového učenia a neurónových sietí na riešenie týchto čiastkových problémov. Výsledné klasifikačné skóre by bolo určené ako kombinácia výsledkov riešení týchto čiastkových problémov. Tento prístup minimalizuje riziko falošne pozitívnej klasifikácie.

3.1 Non-NLP metódy

V tejto práci sa zaoberáme detekciou falošných správ vzhľadom na ich textový obsah. Teda problém chápeme ako úlohu textovej klasifikácie. Tento princíp detekcie falošných správ vyplynul z dostupných dát pre slovenský jazyk. Ale hlavná motivácia spočíva v cieli zistiť, či je možné falošné správy detegovať len na základe špecifík použitého jazyka.

Na druhej strane existuje viac prístupov k detekcii falošných správ. Niektoré z princípov ako detegovať falošné správy, ktoré už boli preskúmané, stručne zhrnieme v nasledujúcich podkapitolách.

3.1.1 Overenie zdroja a autora

Pravdivé spravodajské články by mali vždy obsahovať meno autora a zdroj. Väčšina nepravdivých správ tento zdroj neobsahuje. Ale niektoré stránky sa to naučili obchádzať tým, že vytvárajú reťazec mnohých článkov publikovaných na rôznych webových stránkach, aby navodili dojem, že informácie v článku sú dostatočné pokryté zdrojmi. No v skutočnosti reálny zdroj chýba [1].

Automatická kontrola zdrojov by mohla byť vítanou funkciou, ktorá by odhalila pôvod informácie: či pochádza z hodnoverného zdroja, alebo ak by sa v reťazci zdrojov zacyklila, označila by danú informáciu za dôveryhodnú. Výsledne hodnotenie by bolo zahrnuté v celkovom hodnotení všetkých podproblémov. Netreba pri takomto overovaní zdrojov zabudnúť na obrázky a videá, ktoré sú súčasťou článku.

3.1.2 DeepFake

Termín DeepFake sa používa na označenie médií (zvyčajne audio alebo video), ktoré sú manipulované pomocou techník umelej inteligencie, konkrétne algoritmami strojového učenia, na vytvorenie falošnej verzie pôvodného obsahu [1].

Potom jednou z možností, ako odhaliť falošnú správu, je skontrolovať obrázky priložené k článku. Kontrolou sa overí ich súvislosť s textom, ale aj korektnosť jednotlivých priložených obrázkov. Cielene nesprávne zvolený obrázok môže manipulovať rovnako ako klamlivý nadpis alebo časť textu.

Najčastejší spôsob ako určiť, či daný obrázok súvisí s textom, je použitie neurónových sietí založených na architektúre konvolučných neurónových sietí [19].

3.2 Klasifikácia na základe textových dát

Klasifikácia falošných správ založená na textových dátach sa zaoberá triedením a kategorizáciou rôznych textových zdrojov na základe toho, či obsahujú nepravdivé informácie, alebo manipulatívne techniky, ktoré môžu zaviesť čitateľa. Strojové učenie a algoritmy sa používajú na extrakciu príznakov z textu, ktoré následne pomáhajú určiť, či ide o falošnú správu, alebo nie. Existuje viacero prístupov k tejto klasifikácii, napríklad štatistické metódy alebo algoritmy založené na hlbokom učení. V súčasnosti sa vyvíjajú nové nástroje a aplikácie, ktoré umožňujú rýchle a účinné klasifikovanie falošných správ na základe ich obsahu.

V tejto práci sa bližšie zameriame práve na tieto techniky detekcie falošných správ a ich aplikáciu na dataset obsahujúci texty v slovenskom jazyku.

3.2.1 Falošné správy generované pomocou NLP

Falošné správy generované neurónovými sieťami sú v tejto oblasti pomerne novinkou. Generovanie falošných správ pomocou techník spracovania prirodzeného jazyka (NLP) zahŕňa použitie algoritmov strojového učenia a neurónových sietí na vytváranie textu, ktorý je ťažké rozlíšiť od textu napísaného človekom. Tieto algoritmy dokážu analyzovať existujúci text a vytvárať nový text, ktorý sa používa na šírenie falošných alebo neoverených informácií.

Štúdia, ktorú Zellers a kolektív uverejnili, predstavuje model s názvom Grover na tvorbu kontrolovateľných spravodajských článkov [20]. Dokáže vygenerovať celý článok len na základe malého kúsku textu alebo nadpisu. Navyše, ak je vo vstupe uvedená doména spravodajského webu, vygeneruje sa text v štýle týchto novín. Neurónové falošné správy generované týmto modelom boli ľuďmi hodnotené ako dôveryhodnejšie ako správy písané ľuďmi. Podobný výsledok je možné dosiahnuť aj pomocou modelu GPT-2 pri zadaní správneho výrazu (kľúčového slova) [21], aj keď tento model na to nie je priamo určený. Spoločnosť OpenAI už vydala jeho vylepšenú verziu GPT-3, ktorá má aj verejne dostupného chatbota (ChatGPT), čo prácu s týmto modelom ešte uľahčuje. Pomocou tohto verejne dostupného chatbotu vie generovať text aj laik [22].

Generovanie falošných správ pomocou NLP môže byť použité na rôzne účely, napríklad šírenie dezinformácií alebo manipulácia verejnou mienkou. Takéto generovanie falošných správ je o to vážnejšie, že sa môže diať automaticky a plne individuálne sa prispôsobovať každému konečnému čitateľovi na základe jeho preferencií.

4. Vyhodnocovanie výsledkov

Pre porovnanie odlišných metód strojového učenia a vyhodnotenie ich úspešnosti sa používajú rôzne metriky. Značné množstvo metrík, ktoré sú určené pre klasifikačné úlohy, sú založené na hodnotách uvedených v konfúznej matici [23]. Tieto metriky si priblížime v tejto kapitole.

4.1 Konfúzna matica

Zvyčajne sa na prvotnú analýzu výsledkov klasifikačných metód využíva konfúzna matica. Táto matica sa často používa na vypočítanie ďalších komplexnejších metrík, ktoré sú založené na hodnotách v tejto matici.

		Trieda predikcie	
		Positive	**Negative**
Skutočná trieda	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Obr. 4.1: Konfúzna matica pre binárnu úlohu

Na ilustrácii 4.1 je znázornená konfúzna matica pre binárnu klasifikačnú úlohu, t.j. konfúzna matica dvoch tried. Riadky matice reprezentujú skutočné triedy a stĺpce predstavujú predikované triedy príkladov, prípadne môže byť matica transponovaná. Matica znázornená na ilustrácii 4.1, má hlavnú diagonálu, ktorá zodpovedá správnym predikciám, a vedľajšiu diagonálu, ktorá zodpovedá tým nesprávnym.

V prípade, že pozitívne príklady sú označené hodnotou *1* a negatívne príklady hodnotou *0*, kategória *True Positive (TP)* vyjadruje počet pozitívnych príkladov, ktoré boli správne klasifikované. Podobne kategória *True Negative (TN)* vyjadruje počet správne klasifikovaných negatívnych príkladov. Kategória *False Negative (FN)* zahrňuje pozitívne príklady, ktoré boli nesprávne klasifikované ako negatívne príklady. Na druhej strane, kategória *False Positive (FP)* označuje počet príkladov, ktoré sú negatívne, ale boli nesprávne klasifikované ako pozitívne príklady [24].

4.2 Accuracy

Táto metrika vyjadruje pomer počtu správnych predpovedí k celkovému počtu vstupov. Výpočet tejto metriky na základe hodnôt konfúznej matice je matematicky vyjadrený rovnicou 4.1.

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

V prípade mohutne nevybalansovaných klasifikačných tried v datasete nie je vhodné použiť metriku accuracy na vyhodnotenie kvality klasifikačného modelu. V takejto situácii táto metrika ťažko odhaľuje nesprávnu klasifikáciu triedy s menším počtom vstupných vzorov.

4.3 Precision a recall

Podobne ako pri accuracy sa aj hodnota precision vypočíta z konfúznej matice a určuje pomer správne pozitívne predikovaných príkladov ku všetkým, ktoré boli klasifikované ako pozitívne. Tento vzťah sa dá vyjadriť pomocou rovnice 4.2.

$$precision = \frac{TP}{FP + TP} \tag{4.2}$$

Recall vyjadruje percento príkladov danej triedy, ktoré klasifikátor správne identifikuje. Táto metrika indikuje, ako dobre model dokáže rozlišovať jednotlivé triedy. V prípade, že má recall hodnotu 1, klasifikátor dokáže danú triedu identifikovať bezchybne.

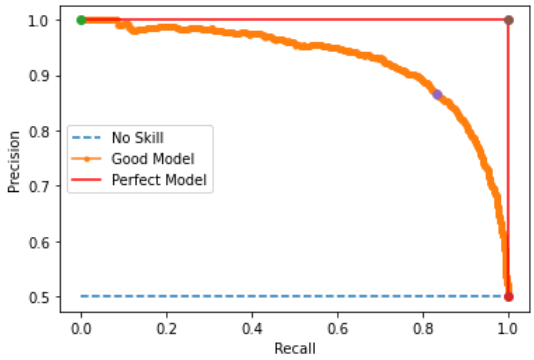
$$recall = \frac{TP}{FN + TP} \tag{4.3}$$

Vysokú hodnotu precision je možné dosiahnuť tiež spôsobom, že klasifikačná metóda vráti iba jeden pozitívny príklad, pričom ešte existuje veľké množstvo pozitívnych, t.j. anotovaných príkladov s hodnotou 1. Takáto situácia je však nežiaduca a hodnota metriky recall bude v tomto prípade veľmi nízka. Na druhej strane opačným extrémom je, keď klasifikátor predikuje všetko ako pozitívne. V takejto situácii je hodnota recall veľmi vysoká, ale typicky potom bude málo zo všetkých pozitívne predikovaných príkladov relevantných, čo tiež nie je štandardný stav. Na druhej strane v takomto prípade bude hodnota precision veľmi nízka. Preto sa tieto metriky najčastejšie vyjadrujú v podobe závislosti hodnoty precision a recall napríklad v podobe precision-recall kriviek. Nevýhodou týchto metrík je, že sú dve a následne porovnanie rôznych klasifikačných metód je veľmi náročné. Preto boli navrhnuté aj iné metriky, ktoré vo svojom výpočte priamo kombinujú tieto dve metriky.

4.3.1 Precision-Recall krivka

Precision-recall krivka (taktiež nazývaná PR krivka) je grafickým nástrojom na vyhodnotenie výkonu klasifikačného modelu, najmä v situácii, že vstupná dátová sada je nevybalansovaná (t.j. jedna trieda je v dátovej sade zastúpená viac ako druhá). Precision-recall krivka je generovaná tým, že postupne meníme prah klasifikácie v modeli, ktorý rozhoduje, kedy označiť príklad za pozitívny a kedy za negatívny. Pri každom zmenšení alebo zväčšení prahu získame iné hodnoty precision a recall a tieto hodnoty sú potom zakreslené do grafu.

V ideálnom prípade sa krivka pohybuje blízko bodu $(1; 1)$, čo indikuje, že model dosahuje vysokú precision aj recall. Čím bližšie je krivka k tomuto bodu, tým lepší je výkon modelu. Na druhej strane, ak krivka ostáva blízko bodu $(0; 0)$, znamená to, že model má zlý výkon.



Obr. 4.2: Precision-recall krivka [25].

Precision-recall krivka je užitočným nástrojom na porovnanie výkonu rôznych modelov, najmä v situáciách, keď je dôležité správne identifikovať menej častú triedu v datasete.

4.4 F1 skóre

Metrika F1 skóre vyjadruje harmonický priemer medzi hodnotami precision a recall. Pre najlepší výsledok dosahuje hodnoty 1, naopak pre najhorší 0 [26]. V prípade klasifikácie viacerých tried sa táto metrika počíta zvlášť a nakoniec je spriemerovaná do $F1_{macro}$.

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \tag{4.4}$$

Ak je $\beta > 1$, potom je kladený väčší dôraz na recall; naopak, ak je $\beta < 1$, je dôraz kladený na precision. Možnosť zmeny dôrazu má veľké množstvo výhod, ale nesie so sebou aj jednu značnú nevýhodu. Ak sú pri vyhodnocovaní rôznych modelov použité odlišné voľné parametre β, potom je vzájomne porovnanie týchto modelov veľmi ťažké.

Táto metrika má hlavnú výhodu v tom, že je kombináciou iných metrík, a tak nemá sklony k rovnakým chybám ako predchádzajúce metriky. Ďalšou výhodou je možnosť zmeny dôrazu na presnosť (precision) alebo úplnosť (recall) podľa toho, ktorá z metrík je pre nás dôležitejšia. Ak chceme priradiť rovnakú váhu obom metrikám, nastavíme parameter β na hodnotu 1.

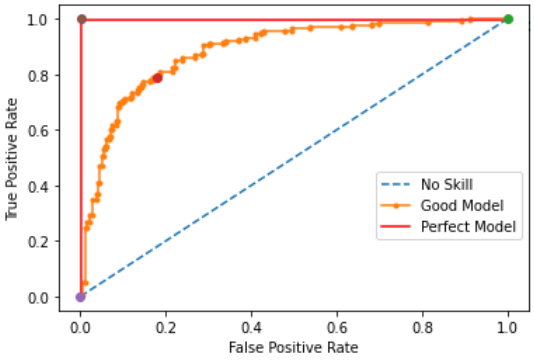
$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.5}$$

4.5 AUC-ROC krivka

Krivka ROC je vytvorená na základe hodnôt *True Positive Rate (TPR)*, ktorá reprezentuje hodnotu na osi y, a *False Positive Rate (FPR)*, ktorá reprezentuje hodnotu na osi x.

$$TPR = \frac{TP}{TP + FN} \tag{4.6}$$

$$FPR = \frac{TN}{FP + TN} \tag{4.7}$$



Obr. 4.3: ROC krivka.

Vyjadrením tejto krivky pomocou jedného čísla je metrika AUC. Názov, ktorý sa používa pri tejto metrike, je skratkou celého názvu v angličtine *„Area Under the Receiver Operating Characteristics“*, taktiež je zaužívaná skratka AUROC.

menej ako 0,5	0,50 až 0,75	0,75 až 0,92	0,92 až 0,97	0,97 až 1,00
nedostatočný	priemerný	dobrý	veľmi dobrý	vynikajúci

Tabuľka 4.1: Približné hodnotenie úspešnosti modelu podľa metriky AUC.

V tabuľke 4.1 sú uvedené intervaly, ktoré vyjadrujú kvalitu modelov na základe hodnoty AUC. Ak má model AUC hodnotu 0,5, naznačuje to, že nemá schopnosť správne klasifikovať príklady a jeho výsledky sú rovnaké, ako by sme pri klasifikácii hádali náhodne medzi pozitívnymi a negatívnymi triedami. V takomto prípade by model nepredstavoval spoľahlivý nástroj pre klasifikáciu. Úspešnosť klasifikátora podľa metriky AUC-ROC sa vyhodnocuje na väčšom množstve behov, pri ktorých sa menia učiace parametre tohto algoritmu. Ak hodnota AUC-ROC je aproximovane 1, indikuje to, že klasifikátor lepšie rozlišuje jednotlivé triedy. Ak je hodnota blízka nule, znamená to, že klasifikátor má najhoršiu schopnosť rozlišovať triedy a predikuje ich v binárnej klasifikačnej úlohe opačne [27].

4.5.1 Porovnanie ROC a precision-recall krivky

ROC (Receiver Operating Characteristic) krivka a precision-recall krivka sú dve rôzne metódy vizualizácie a hodnotenia výkonu klasifikačných modelov. Obidve krivky poskytujú užitočné informácie o schopnostiach modelu, ale môžu byť informatívne v rôznych situáciách.

- ROC krivka by sa mala používať, keď je približne rovnaký počet príkladov pre každú triedu.
- Precision-recall krivka by sa mala používať, keď existuje stredná až veľká nerovnováha medzi triedami.

5. Vektorizácia textu

Existujú rôzne metódy transformácie textových dát do vektorového kódovania. V tejto práci sa budeme venovať transformácii textových dát do numerického priestoru pomocou dvoch z nich: BoW a TF–IDF. Tieto metódy boli využité v základnej sade experimentov a následne pri implementácii hlbokých neurónových sietí v kapitolách 11 a 12. Tieto algoritmy strojového učenia nevedia priamo pracovať s textovým formátom, preto je potrebné upraviť textové dáta do numerického priestoru pred ich spracovaním týmito algoritmami.

5.1 Frekvencia tokenov (BoW)

Veľmi efektívnou a zároveň najjednoduchšou metódou, ktorá sa využíva na transformáciu vstupu, je Bag of words, častejšie uvádzaný pod skratkou BoW. Majme súbor textov, ktorý sa v úlohách spracovania prirodzeného jazyka nazýva korpus. Potom termín frekvencia tokenov môže byť charakterizovaný ako počet jednotlivých tokenov v danom texte. Hlavná myšlienka celej transformácie je veľmi jednoduchá, ale zároveň účinná. Touto myšlienkou je určenie výskytu slov v dokumente, pričom sa ignoruje ich poradie v texte. Vysoký počet niektorého z tokenov značí jeho dôležitosť pri rozhodovaní. Ak aplikujeme tento proces na každý text v korpuse, sú tieto texty reprezentované pomocou vektora čísel s fixnou dĺžkou. Dĺžka tohto vektora je závislá od veľkosti slovníka korpusu.

5.2 TF–IDF

Druhou metódou vektorizácie textu, ktorú si uvedieme, je TF–IDF (z angl. Term Frequency – Inverse Document Frequency). Táto metóda je oveľa komplexnejšia a vo svojej vektorovej reprezentácii zohľadňuje, aké je dané slovo dôležité vzhľadom na celý súbor textov. Transformácia slova na vektor pozostáva z dvoch primárnych častí - frekvencie tokenov (TF) a inverznej dokumentovej frekvencie (IDF). TF miera udáva počet výskytov slova v konkrétnom texte, teda vyjadruje jeho dôležitosť v rámci textu. IDF na druhej strane poskytuje informáciu o tom, ako unikátne alebo špecifické je dané slovo pre isté dokumenty v celom súbore dokumentov. Táto miera znižuje význam častých slov, ktoré sa vyskytujú vo väčšine dokumentov, a zvyšuje význam špecifických slov. Táto informácia sa používa na váženie významu slova v rámci celej kolekcie textov, nie iba v rámci jedného textu. TF-IDF sa vypočíta pomocou vzorca 5.1.

$$TFIDF = TF \times IDF \tag{5.1}$$

Štandardná definícia 5.3 TF je:

$$TF(t,d) = \frac{frekvencia\ tokenu\ t\ v\ dokumente\ d}{celkový\ počet\ tokenov\ v\ dokumente} \tag{5.2}$$

IDF vyjadríme pomocou vzorca 5.3:

$$IDF(t,D) = \log \frac{celkový\ počet\ dokumentov\ v\ datasete}{počet\ dokumentov,\ ktoré\ obsahujú\ token\ t} \tag{5.3}$$

6. Strojové učenie

Táto kapitola bude venovaná klasifikácii textu pomocou rôznych modelov strojového učenia. Klasifikácia textu označuje proces, v ktorom sú vstupné texty kategorizované do vopred definovaných tried. Tento automatický proces spracovania textu sa používa v rôznych oblastiach detekcie, napríklad:

- spamové filtre
- triedenie novinových článkov alebo e-mailov podľa obsahu
- detekcia falošných správ alebo podvodov

Klasifikátor má za úlohu priradiť každý dokument do správnej klasifikačnej triedy. Ešte pred samotným tréningom klasifikačného modelu a jeho otestovaním je nutné v prvom kroku vstupnú dátovú sadu rozdeliť na trénovaciu a testovaciu časť. Až po vykonaní tohto kroku môžeme na trénovacích dátach vybudovať klasifikátor. A následne v ďalšom kroku takto naučený klasifikátor otestovať na množine testovacích dát. Tým, že model je testovaný na nezávislých dátach, získavame objektívnejšiu predstavu o jeho úspešnosti a schopnostiach generalizácie na nové neznáme dáta.

Celkovo je rozdelenie datasetu na trénovaciu, testovaciu a prípadne ešte validačnú množinu súčasťou správnej metodológie strojového učenia, ktorá pomáha vývojárom a vedcom získať spoľahlivé a generalizovateľné modely.

Pri testovaní na overenie úspešnosti modelu používame rôzne metriky ako accuracy, F1 score, precision, recall atď., ktoré sme popísali v kapitole 4.

Cieľom klasifikácie textových dát je rozdeliť množinu m samostatných dokumentov $D_1, D_2,..., D_m$ obsiahnutých v množine textových dát D, ktorú tiež nazývame korpus, do n tried $T_1, T_2, ..., T_n$ na základe ich charakteristík. Klasifikátor sa snaží priradiť každý dokument D_i, pre $i = 1,2, ..., m$, do najvhodnejšej triedy T_j, pre $j = 1, 2, ...,n$, na základe podobnosti dokumentu s touto triedou. Nie je vždy možné jednoznačne priradiť dokument do konkrétnej triedy, ale klasifikátor sa snaží nájsť najlepšiu možnú zhodu, teda najpodobnejšiu triedu dokumentov, ktorým sa daný dokument najviac približuje, aj keď nie stopercentne.

V tejto kapitole sa zameriavame na základné algoritmy strojového učenia, ktoré sme použili na klasifikáciu falošných správ v slovenskom jazyku. Pre túto úlohu sme tieto algoritmy definovali ako klasifikátory. Avšak, prostredníctvom vhodných modifikácií je možné tieto algoritmy adaptovať na výpočet spojitej cieľovej funkcie. Takéto upravené algoritmy potom získavajú schopnosť riešiť regresné problémy. Tým sa vytvára väzba medzi klasifikačnými a regresnými úlohami, a umožňuje sa využitie podobných algoritmických prístupov v oboch prípadoch.

6.1 Naivný Bayesov klasifikátor

Naivný Bayesov klasifikátor je založený na princípe pravdepodobnostného klasifikovania a využíva predpoklad nezávislosti vstupných atribútov medzi sebou.

Oproti iným klasifikačným metódam je jeho hlavnou výhodou, že nepotrebuje veľké množstvo trénovacích dát na naučenie parametrov jednotlivých tried.

Tento model strojového učenia klasifikuje vstup na základe Bayesovej vety, ktorej znenie je nasledujúce:

Majme dva náhodné javy A a B s pravdepodobnosťami $P(A)$ a $P(B)$, kde $P(B) \neq 0$. Potom za predpokladu, že nastal náhodný jav B, je možné podmienenú pravdepodobnosť javu A vyjadriť ako:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{6.1}$$

V predošlom odstavci sme uviedli definíciu Bayesovej vety a v tejto časti sa budeme venovať jej aplikácii v našej konkrétnej úlohe klasifikácie falošných správ na základe textového obsahu. Nech existujú cieľové klasifikačné triedy $T_1, T_2, ..., T_n$, potom úlohou naivného Bayesovho klasifikátora je vypočítať podmienenú pravdepodobnosť, že nový dokument D_{new} s atribútmi (charakteristikami) $d_1, d_2,..., d_l$ patrí do triedy T_j pre $j = 1,2,...,n$.

Túto pravdepodobnosť pre klasifikačnú triedu T_j vypočítame podľa nasledujúceho vzorca:

$$P(T_j|d_1, d_2,..., d_l) = \frac{P(d_1, d_2,..., d_l|T_j)P(T_j)}{P(d_1, d_2,..., d_l)}, \qquad 1 \leq j \leq n \tag{6.2}$$

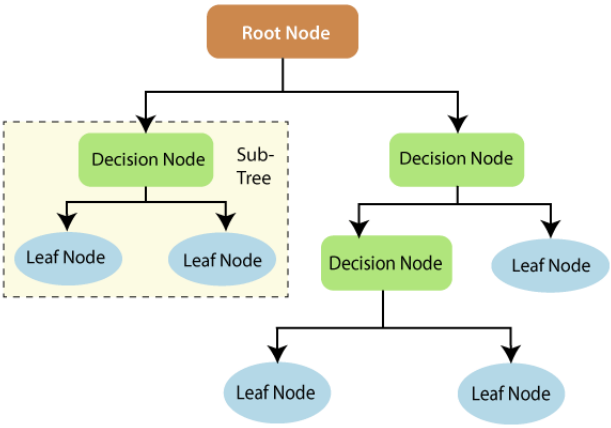
Z predpokladu nezávislosti atribútov medzi sebou potom dostaneme zo vzťahu 6.2 nasledujúci vzorec 6.3 a ten sa už nazýva naivný Bayesov klasifikátor.

$$P(T_j|d_1, d_2,..., d_l) = \frac{\prod_{s=1}^{l} P(d_s|T_j)P(T_j)}{P(d_1, d_2,..., d_l)}, \qquad 1 \leq j \leq n \tag{6.3}$$

Naivný Bayesov klasifikátor, ako sme ho zadefinovali vyššie, môže fungovať iba v teoretickej rovine, no pre jeho aplikáciu v reálnom svete je nutné získať odhady pravdepodobností zo vstupných trénovacích dát, keďže ich v praxi nemáme. Odhady následne slúžia ako základ pre klasifikáciu nových vstupov. Tento klasifikátor sa často používa na spamovú filtráciu v emailovej komunikácii, klasifikáciu článkov a podobne.

6.2 Rozhodovacie stromy

Architektúra rozhodovacích stromov je tvorená hierarchickou štruktúrou pravidiel reprezentovaných pomocou uzlov. Na ilustrácii 6.1 je znázornená štruktúra rozhodovacieho stromu. Každý rozhodovací strom sa skladá z troch typov uzlov: koreňového uzla, vnútorných uzlov a nakoniec listov. Najčastejšia verzia rozhodovacích stromov sa nazýva binárna. Pre tieto stromy platí, že každý vnútorný uzol má práve dvoch nasledovníkov (potomkov). Táto množina rozhodovacích stromov sa rozhoduje na základe prítomnosti alebo naopak absencie jednotlivých znakov v texte, ktorého triedu chceme určiť.



Obr. 6.1: Rozhodovací strom [28].

Počiatočným uzlom každého rozhodovacieho stromu je koreňový uzol, po ktorom nasledujú vnútorné uzly. Pre úlohu klasifikácie textu vnútorné uzly reprezentujú jednotlivé charakteristiky dokumentu. Finálne rozhodnutie musí prejsť cez všetky tieto uzly až k listom. Listy sú uzly v strome, ktoré nemajú potomkov, reprezentujú jednotlivé klasifikačné triedy, do ktorých chceme vstupné dáta priradiť. Vetvy vedúce z vnútorných uzlov označujú rozhodovacie pravidlo, podľa ktorého sme sa v danom uzle rozhodli. Postupným prechodom rozhodovacím stromom od koreňového uzla po listy pre každý dokument v súbore dát docielime jeho klasifikáciu podľa listového uzla. Listový uzol priradí triedy na základe naučených hodnôt na trénovacej množine.

Rozhodovacie stromy používajú kľúčové atribúty (charakteristiky) pre klasifikáciu príkladov do správnej triedy. V našom prípade detekcie fake-news to popíšeme na výbere dokumentu repsektíve textu. V procese klasifikácie je kritické v každom kroku vybrať ten atribút, ktorý najlepšie rozdelí množinu textov v danom uzle. To znamená, že je dôležité rozhodnúť, ktorý atribút sa použije v koreňovom uzle, ktorý vo vnútorných uzloch tak, aby sme docielili čo najlepšiu klasifikáciu daného dokumentu. Giniho index GI a entropia H sú najbežnejšie používané kritériá na výber správneho atribútu pre rozdelenie rozhodovacieho stromu v danom uzle. Pri porovnaní využitia GI verzus entropia v rozhodovacích stromoch ako kritéria delenia uzlov sa vo väčšine aplikuje viac práve Giniho index. Táto skutočnosť je spôsobená štýlom výpočtu entropie, ktorá je založená na logaritmickej funkcii. Výpočet tejto funkcie je náročnejší, a preto sa v praxi viac používa Giniho index [29] pre urýchlenie výpočtov pomocou rozhodovacích stromov. V rámci nášho výskumu sme sa rozhodli využiť Giniho index na výber atribútov pre delenie, s prihliadnutím na uvedené dôvody a predchádzajúce štúdie. V nasledujúcich odstavcoch si podrobnejšie rozoberie výpočet Giniho indexu, aby sme lepšie porozumeli jeho metóde použitia pri výbere atribútov pre delenie.

Vzorec 6.4 pre výpočet Giniho indexu GI v rozhodovacích stromoch používa pravdepodobnosti tried p_i, ktoré určujú podiel príkladov priradených určitej triede v danom listovom uzle k celkového počtu všetkých príkladov v tomto uzle.

$$GI = 1 - \sum_{i=1}^{n} p_i^2, \tag{6.4}$$

Giniho index pre vnútorný uzol sa vypočíta ako súčet hodnôt Giniho indexu pre každý z dcérskych uzlov, vynásobený podielom počtu pozorovaní v danom dcérskom uzle a celkového počtu pozorovaní v materskom uzle. Vzorec 6.5 zjednodušene vyjadruje, ako sa vypočíta celková hodnota Giniho indexu pre daný vnútorný uzol po jeho rozdelení na u nových uzlov:

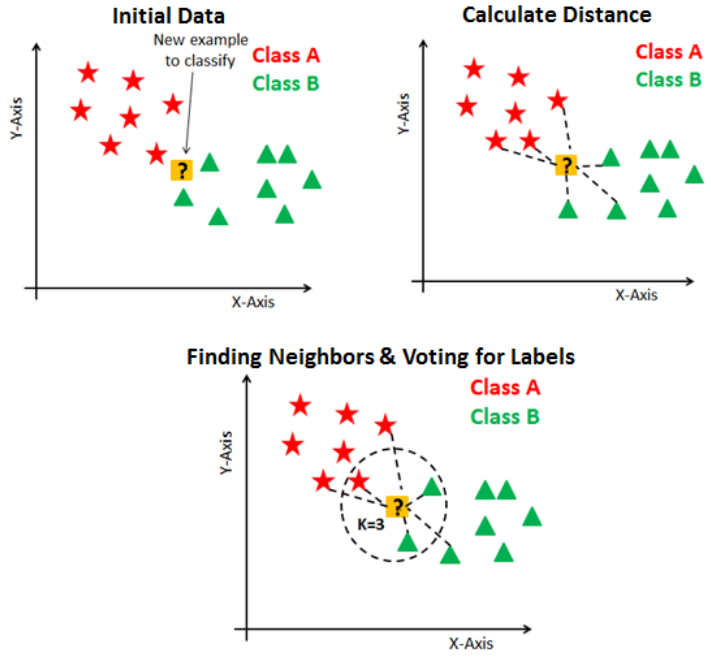
$$GI^* = \sum_{i=1}^{u} \frac{N_i}{N_t} GI(i), \tag{6.5}$$

Vzorec pre celkovú hodnotu Giniho indexu v danom vnútornom uzle zahrňuje počet dcérskych uzlov u, počet pozorovaní N_i v každom z týchto uzlov a celkový počet pozorovaní N_t v materskom uzle.

6.3 k-NN klasifikátor

Algoritmus k-NN, čo znamená k-najbližších susedov, pracuje s predpokladom, že každý príklad z trénovacej dátovej sady reprezentuje bod v priestore dimenzie l. Vstupné príklady sú vyjadrené ako vektory v tomto priestore a každý z trénovacích príkladov je priradený k určitej klasifikačnej triede. Algoritmus k-NN pracuje so vstupným príkladom x, ktorý sa reprezentuje ako l-tica x_i, pričom i značí poradie vstupného príkladu. Trénovacia množina obsahuje dvojice (x_i, w_i), kde w_i označuje triedu, ku ktorej daný príklad patrí.

Pri trénovaní klasifikátora k-NN sa ukladajú vektory a prislúchajúce triedy trénovacích príkladov. Pri klasifikácii sa vyberie k najbližších susedov testovaného príkladu zvyčajne pomocou euklidovskej vzdialenosti a rozhodnutie o triede testovaného príkladu sa robí na základe najčastejšej triedy medzi jeho najbližšími susedmi uloženými v procese trénovania [30]. Parameter k určuje, koľko najbližších susedov sa zapojí do klasifikácie vstupného príkladu x.



Obr. 6.2: Algoritmus k-NN [31].

Klasifikátor k-NN je implementačne pomerne nenáročný a poskytuje napriek tomu značne vysokú presnosť, najmä pre klasifikáciu s malým počtom tried. Jednou z hlavných nevýhod tohto algoritmu je však vysoká výpočtová zložitosť, ktorá vyžaduje porovnanie vstupného príkladu so všetkými príkladmi v trénovacom datasete. Tento proces môže byť náročný, najmä ak je dataset veľký. Existujú rôzne úpravy algoritmu k-NN, ktoré umožňujú jeho prispôsobenie pre rôzne potreby a požiadavky [32].

6.4 Ensemble metódy strojového učenia

Základné metódy strojového učenia dosiahli významné úspechy pri riešení rozmanitých úloh. Avšak, dodatočný výskum odhalil problémy, ako nedostatok dostupných dát, vysoká variabilita dát a preučenie [33]. Ensemble stratégia je jednou z metód, ktoré sa využívajú na riešenie vyššie uvedených problémov a dosahuje pomerne dobré výsledky. Metóda kombinuje výstupy viacerých klasifikátorov namiesto použitia jedného samostatného klasifikátora. Tento prístup zvyšuje presnosť a robustnosť predikcie. Po naučení jednotlivých klasifikátorov v ensemble systémoch je dôležité zvoliť najvhodnejší spôsob spojenia ich výsledkov.

Jednou z techník, ktorú môžeme použiť na spojenie výsledkov, je tzv. soft voting. V tejto metóde každý model v rámci ensemble systému poskytuje pravdepodobnosti pre každú triedu. Pri klasifikácii je potom konečná trieda určená na základe hlasovania, v ktorom sa zohľadňuje priemerne priradená pravdepodobnosť k jednotlivým triedam. Tá trieda, ktorá má najvyššiu priemernú pravdepodobnosť, sa považuje za konečné rozhodnutie. Na druhej strane, majority voting je jednoduchšou technikou hlasovania, kde každý model v ensemble systéme hlasuje pre jednu konkrétnu triedu. V prípade klasifikácie je výsledná trieda určená na základe väčšinového hlasovania. Trieda, ktorá získala najviac hlasov od jednotlivých modelov, je považovaná za konečné rozhodnutie.

Obidve techniky majú svoje výhody aj nevýhody. Soft voting berie do úvahy pravdepodobnostné informácie a môže byť vhodný pre problémy s neistotou v rozhodnutiach. Majority voting je jednoduchší a efektívnejší výpočtovo, avšak nezohľadňuje pravdepodobnostné informácie a môže byť náchylný na nevybalansovanie tried v dátovej sade.

Existuje niekoľko rôznych metód ensemble learningu. Dve základné metódy ensemble strojového učenia sú bagging a boosting.

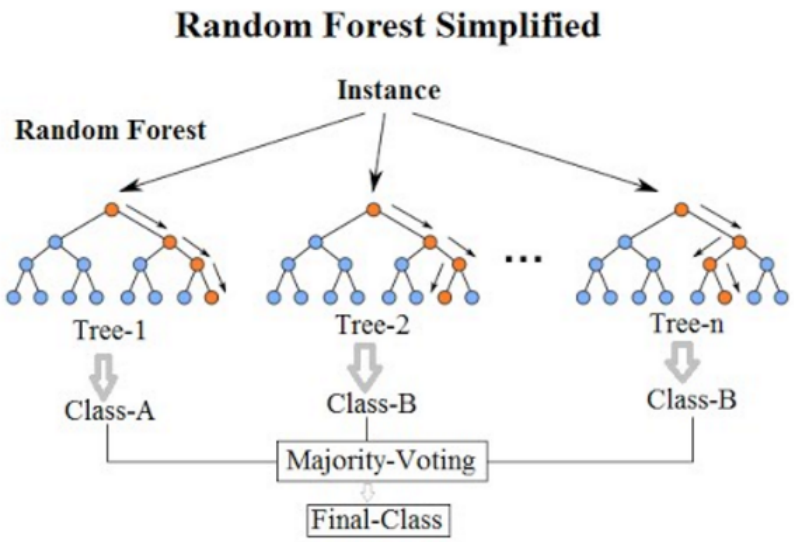
6.4.1 Bagging

Bagging, známy aj ako bootstrap aggregation, je jednou z najstarších metód ensemble systémov, ktoré sa používajú na zlepšenie presnosti algoritmov strojového učenia pre klasifikačné a regresné úlohy. Využíva náhodný výber s opakovaním z pôvodného trénovacieho súboru dát na vytvorenie bootstrap podmnožín, ktoré sa používajú na trénovanie rôznych klasifikátorov.

Finálnym výstupom je rozhodnutie založené na väčšinovom hlasovaní jednotlivých klasifikátorov, pričom konečné rozhodnutie je určené triedou, ktorá získala najväčší počet hlasov. Bagging je veľmi jednoduchý na implementáciu a je účinný najmä v prípade obmedzeného množstva dát [34].

Random Forest

Algoritmus strojového učenia nazývaný Random Forest je aplikáciou metódy bagging s použitím rozhodovacích stromov [35]. Random Forest rieši tento problém vytvorením viacerých stromov, ktoré sú potom kombinované napríklad pomocou štatistického modusu pre klasifikačné úlohy.



Obr. 6.3: Schéma algoritmu náhodný les (random forest) [36].

Vo formálnej definícii sa Random Forest skladá z C_1,...,C_R množiny rozhodovacích stromov. Následne pre každý rozhodovací strom C_i patriaci do náhodného lesa je vytvorená samostatná trénovacia množina, ktorá je vytvorená prevzorkovaním z kompletnej pôvodnej množiny dát. Ide o výbery s náhodným opakovaním príkladov, pričom je definovaná ich veľkosť s. Pri tvorbe stromov sa náhodne vyberajú nielen rôzne podmnožiny vstupných dát, ale aj atribútov, na ktorých základe sa klasifikátor rozhoduje.

Výstup klasifikácie pomocou tohto algoritmu strojového učenia môžeme vyjadriť vzťahom [37].

$$C_{RF} = modus\{C_i(x)\}_i^R, \tag{6.6}$$

kde R označuje počet rozhodovacích stromov v ensemble systéme a $C_i(x)$ je predikcia i-teho stromu pre vstupné dáta x [38].

Na obrázku 6.4 je konkrétna ilustrácia ensemble systému náhodný les, ktorý tvoria tri rozhodovacie stromy. Klasifikácia pomocou tohto konkrétneho modelu strojového učenia sa uskutoční kombináciou výsledkov náhodných stromov metódou majority voting. Finálna klasifikácia príkladu podľa tejto ilustrácie bude do triedy B.

Náhodné lesy sa tešia značnej popularite medzi ensemble systémami. Toto rozšírenie algoritmu rozhovacieho stromu sa vyznačuje jednoduchou implementáciou a veľmi dobre sa vie vysporiadať s problémom preučenia.

6.4.2 Boosting

Boosting je metóda strojového učenia, ktorá vytvára silný model kombinovaním viacerých slabých modelov. Tento algoritmus sa zameriava na zlepšovanie presnosti predikcie tým, že sa postupne učí zo slabých modelov a uprednostňuje tie oblasti, v ktorých tieto modely zlyhávajú.

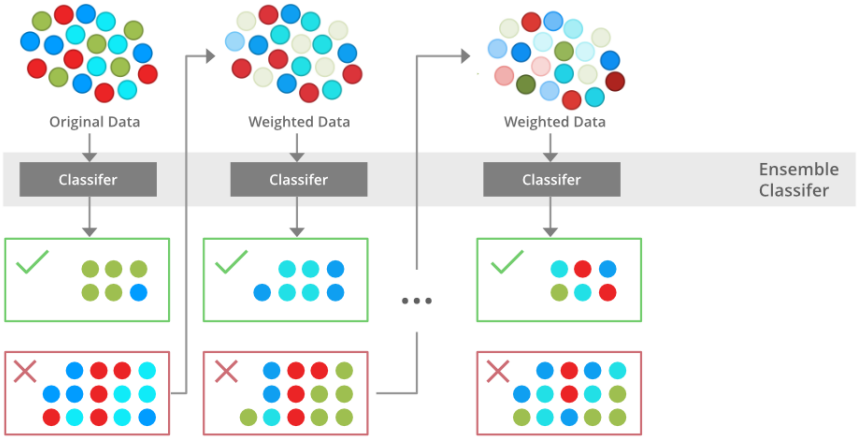
Princíp boostingu spočíva v tom, že sa vytvorí prvý slabý model, ktorý je následne vyhodnotený. Potom sa ďalšie modely vytvárajú tak, aby sa zameriavali na príklady, ktoré predchádzajúce modely zle klasifikovali. Každý ďalší model sa snaží opraviť chyby predchádzajúcich modelov a pridáva sa do celkového ensemble modelu. Počas trénovania sa upravujú váhy jednotlivých inštancií dát s cieľom zamerať sa na príklady, ktoré boli nesprávne klasifikované. Tieto váhy sa postupne prerozdeľujú, čo umožňuje modelu sústrediť sa na najťažšie príklady. Týmto spôsobom dochádza k postupnému zlepšovaniu celkovej presnosti modelu.

Kombinovaním výstupov slabých modelov sa vytvára konečný silný model. Boosting môže byť použitý pre klasifikáciu aj regresiu a je populárnou technikou na zvýšenie presnosti predikcie v rôznych oblastiach, vrátane strojového učenia a dátovej analýzy [39].

XGBoost

XGBoost (eXtreme Gradient Boosting) je vysoko výkonný algoritmus boostingu, ktorý sa používa na klasifikáciu aj regresiu. Bol vyvinutý s cieľom dosiahnuť vysokú presnosť predikcie a rýchlosť trénovania modelu [40].

Algoritmus je založený na koncepte gradientového boostingu, ktorý postupne vytvára súbor slabých rozhodovacích stromov a kombinuje ich do silného modelu. Avšak XGBoost prináša niekoľko inovácií, ktoré mu umožňujú dosiahnuť lepšie výsledky.



Obr. 6.4: Priebeh tréningu algoritmu XGBoost [41].

Jednou z hlavných výhod XGBoost je jeho schopnosť optimalizovať stratovú funkciu pomocou gradientného zostupu. V tomto procese sa váhy jednotlivých stromov prispôsobujú s cieľom minimalizovať stratovú funkciu.

XGBoost využíva aj regularizačné techniky, ako je L1 a L2 regularizácia, ktoré slúžia na prevenciu pretrénovania modelu. Tieto techniky prispievajú k zlepšeniu výkonu a stability modelu tým, že zabezpečujú kontrolu nad komplexitou a váhami jednotlivých príznakov v modeli.

Ďalšou výhodou tohto algoritmu je efektívne spracovanie rôznych dátových formátov a tiež umožňuje paralelné trénovanie modelov, čo zvyšuje rýchlosť a škálovateľnosť. Okrem toho obsahuje optimalizované metódy na manipuláciu s chýbajúcimi hodnotami, správu pamäte a podporuje distribuované výpočty.

Poslednou, ale nie menej dôležitou vlastnosťou je schopnosť poskytovať dôležitosť príznakov (feature importance). Táto schopnosť umožňuje analyzovať vplyv jednotlivých atribútov na predikciu, čím môžeme identifikovať najinformatívnejšie atribúty a zlepšiť úspešnosť modelu.

XGBoost sa stal populárnym algoritmom v rôznych oblastiach. Hlavne vďaka svojej výkonnosti, flexibilite a schopnosti pracovať s veľkými dátovými sadami je XGBoost jedným z najpreferovanejších algoritmov pre predikciu a analýzu dát.

7. Hlboké neurónové siete

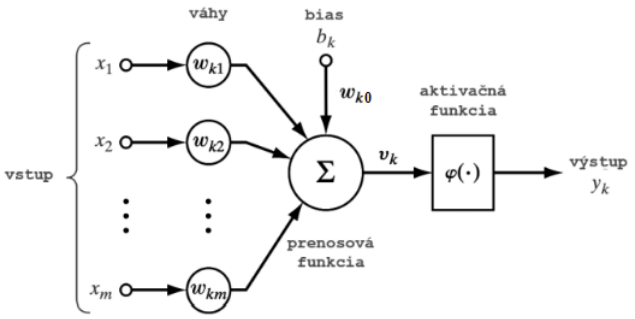
Umelé neurónové siete (skráteným názvom neurónové siete) sú v súčasnosti považované odbornou verejnosťou za jeden z najlepších algoritmov strojového učenia. V dnešnej dobe žnú mnohé úspechy v rôznorodých úlohách [42]. Prinášajú výsledky s veľmi vysokou presnosťou a majú pomerne širokú škálu použitia. Limitované sú iba množinou zdrojových dát, ktoré sú potrebné pre ich správne naučenie. Možnými aplikáciami neurónových sietí sú úlohy počítačového videnia či spracovanie prirodzeného jazyka [43].

V závislosti od použitej architektúry ich niekedy možno označiť aj ako hlboké strojové učenie, ktoré je s narastajúcim výpočtovým výkonom čoraz populárnejšie a komplexnejšie. Na rovnakom princípe ako mnohé iné algoritmy strojového učenia aj neurónové siete fungujú tak, že najskôr sa vytvorí model (architektúra), ktorý sa v ďalšom kroku natrénuje na čo najrozsiahlejšom množstve dát. Na základe vzťahov v týchto známych trénovacích dátach sa model naučí predpovedať výsledok pre nové neznáme príklady z rovnakej domény [42].

Prelom v spracovaní prirodzeného jazyka nastal v roku 2010, keď boli v tejto oblasti použité hlboké neurónové siete [44]. Tento prelom bol možný vďaka dostupnosti dostatočného množstva dát a zvýšeniu výpočtového výkonu. Jedným z prvých výskumníkov, ktorý použil rekurentné neurónové siete, bol Mikolov a jeho kolegovia. Výsledky ich experimentov ukázali, že modely založené na architektúre hlbokých neurónových sietí dosahujú v mnohých úlohách spracovania prirodzeného jazyka lepšie výsledky v porovnaní s tradičnými prístupmi strojového učenia.

7.1 Architektúra neurónových sietí

Neurónové siete sú jedným z matematických modelov strojového učenia, ktoré používajú interné prenosové funkcie na transformáciu vstupných vektorov na výstupné. Ich architektúra a fungovanie sa inšpiruje fungovaním ľudského mozgu [43], preto aj mnohé termíny v tejto oblasti umelej inteligencie sú inšpirované ich anatomickými ekvivalentmi.



Obr. 7.1: Vizualizácia umelého neurónu [43].

Na ilustrácii 7.1 je vyobrazená základná výpočtová jednotka neurónových sietí, ktorá sa nazýva umelý neurón. Podobne ako neurónové siete aj ich výpočtová jednotka je inšpirovaná biologickými nervovými bunkami. Umelý neurón

predstavuje hrubý model biologického neurónu a snaží sa aproximovať jeho správanie. V literatúre je neurón definovaný pomocou váh, bias faktora a aktivačnej funkcie, ktoré sa používajú na transformáciu vstupných dát.

V nasledujúcom odstavci popíšeme zľava doprava ilustráciu na obrázku 7.1, ktorá reprezentuje architektúru umelého neurónu. Vstupné hodnoty sú reprezentované premennými $x_1, ..., x_m$, ktoré vstupujú do neurónu spolu s ich váhami $w_{k1}, ..., w_{km}$ pre výpočet. Na ilustrácii sú označené šípkou, ktorá smeruje do vnútra umelého neurónu.

V umelom neuróne atribút b_k a jemu prislúchajúca váha w_{k0} označujú špeciálny atribút nazývaný bias, ktorého úlohou je dodatočne ovplyvniť výstup aktivačnej funkcie neurónu. Bias je dôležitý, pretože umožňuje neurónovej sieti prispôsobiť sa a zapamätať si rôzne vzory a zložité vzťahy medzi vstupmi a výstupmi. Bias sa využíva na posunutie aktivačnej funkcie neurónu, čím sa mení prahová hodnota, pri ktorej sa neurón aktivuje. Týmto sa zvyšuje flexibilita a schopnosť modelu prispôsobiť sa rôznym dátam. Podobne ako v biologických neurónoch aj v umelých neurónoch existuje vnútorný potenciál označený ako v_k, ktorý sa skladá z váh a vstupov s dodatočným zohľadnením biasu. Vzťah 7.1 je matematickou interpretáciou výpočtu potenciálu neurónu, ako sme ho popísali.

$$v_k = w_{k0} b_k + \sum_{i=1}^{m} w_{ki} x_i \tag{7.1}$$

Výstup umelého neurónu sa získava kombináciou jeho vnútorného potenciálu a aktivačnej funkcie φ, čo je vyjadrené pomocou nasledujúceho vzorca 7.2.

$$y_k = \varphi(v_k) \tag{7.2}$$

Aktivačná funkcia neurónu sa používa na transformáciu výstupu neurónu na hodnotu v určenom rozsahu. Potom sa aplikuje prahovanie na výstupnú hodnotu neurónu pomocou aktivačnej funkcie, kde sa porovnáva s prahovým parametrom. Tento prahový parameter určuje, kedy sa neurón aktivuje a kedy nie.

Pôvodne sa využívala ostrá nelinearita, no neskôr ju nahradili sigmoidálne funkcie ako $\sigma(x)$ alebo $\tanh(x)$, ktoré dokážu lepšie reprezentovať komplexnejšie funkcie a sú jednoducho diferencovateľné [45].

Aktivačné funkcie		
Názov	Obor hodnôt	Funkcia
Ostrá nelinearita	$\{0;\ 1\}$	$\varphi(x) = \begin{cases} 1 & x > 0{,}5 \\ 0 & x \leq 0{,}5 \end{cases}$
ReLu	$\langle 0;\ \infty)$	$\varphi(x) = \max(0,x)$
Logistická sigmoida	$(0;\ 1)$	$\varphi(x) = \frac{1}{(1+e^{-x})}$
Hyperbolický tangens	$(-1;\ 1)$	$\varphi(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

Tabuľka 7.1: Základné aktivačné funkcie neurónových sietí.

7.2 Perceptron

Jeden z najjednoduchších modelov strojového učenia je Perceptron, ktorý tvorí jeden umelý neurón s aktivačnou funkciou typu $\sigma(x)$ alebo prípadne ostrá nelinearita. Tento model dokáže reprezentovať jednoduchý binárny klasifikátor [46], ktorý už v tejto forme predstavuje špecifický algoritmus strojového učenia.

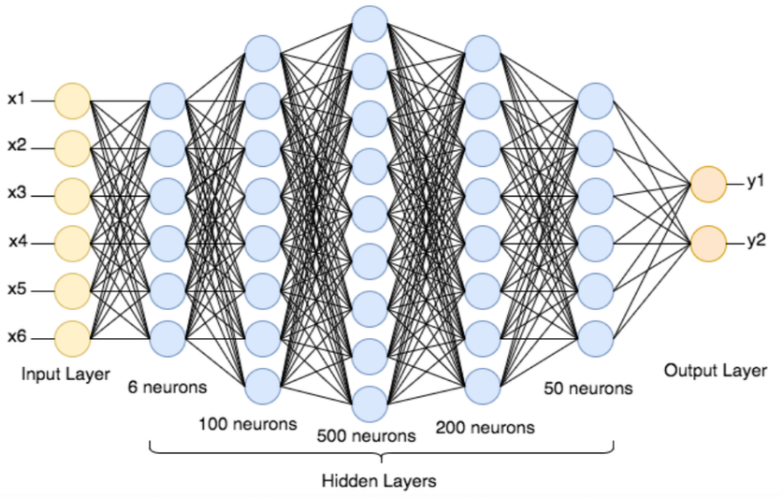
7.3 Neurónová sieť

Perceptron má nevýhodu, že je schopný správne rozdeliť a klasifikovať iba dáta, ktoré sú lineárne separovateľné. To znamená, že v prípade úloh z reálneho sveta, kde sú dáta zvyčajne zložitejšie a nelineárne, nie je dostatočným riešením. Dokonca aj jednoduchá funkcia ako XOR je pre Perceptron problémom, pretože nie je lineárne separovateľná. Aproximácia už tejto jednoduchej funkcie pomocou Perceptronu je nepresná [43].

Použitie väčšej množiny neurónov a ich zapojenie do komplexnejšieho systému umožňuje prekonať obmedzenia Perceptronu. Spôsobom spájania neurónov do usporiadanej štruktúry sa vytvára umelá neurónová sieť.

7.3.1 Topológia neurónovej siete

Neurónová sieť pozostáva z neurónov usporiadaných vo vrstvách. Základná architektúra siete sa skladá zo vstupnej a výstupnej vrstvy, pričom komplexnejšie neurónové siete majú aj skryté vrstvy, ktoré sú uložené medzi vstupnú a výstupnú. Skryté vrstvy predstavujú časti siete, kde sa spracúvajú a transformujú vstupné dáta. Hlavne umožňujú neurónovým sieťam spracovať zložitejšie úlohy, ktoré by nebolo možné riešiť len so vstupnou a výstupnou vrstvou.



Obr. 7.2: Dopredná neurónová sieť [47].

Na ilustrácii 7.2 je vyobrazená topológia, respektíve architektúra takejto siete. Neurónová sieť je tvorená vrstvami, ktoré pozostávajú z neurónov. Pokiaľ informácie prechádzajú cez neuróny jednotlivých vrstiev iba v jednom smere, ide o doprednú neurónovú sieť. Dopredné šírenie signálov medzi neurónmi jednotlivých vrstiev nie je nevyhnutnou podmienkou neurónových sietí. Rekurentné siete

sú schopné navyše obsahovať spätné prepojenia medzi neurónmi a vrstvami, čo umožňuje informáciám cirkulovať a meniť stav siete v čase.

Pojmy ako architektúra, topológia alebo organizačná dynamika siete sa v literatúre používajú na rozdelenie neurónových sietí na rekurentné a dopredné siete. Avšak v oblasti umelej inteligencie zaoberajúcej sa neurónovými sieťami existuje množstvo rôznych prístupov k terminológii, pretože ide o mladú a dynamickú oblasť. Terminológia môže byť rôznorodá a môže sa líšiť v závislosti od zdrojov a literatúry, ktoré sa používajú.

7.3.2 Učenie neurónových sietí

Podobne ako ich biologický vzor aj umelé neurónové siete majú schopnosť učiť sa, čo je ich kľúčovou vlastnosťou. V kontexte strojového učenia to znamená, že sieť môže byť natrénovaná na riešenie špecifického problému [43].

Neurónová sieť sa učí iteratívne na trénovacom súbore dát a jednou z najznámejších metód učenia je algoritmus spätného šírenia chyby (Backpropagation). Po každom prechode súboru dát sa vypočíta gradient chybovej funkcie, ktorý sa určí na základe spätného prechodu sieťou. Gradient sa použije na úpravu váh neurónov tak, aby sa minimalizovala hodnota chybovej funkcie. Chybová funkcia, tiež označovaná ako loss, sa vypočíta podľa vzťahu 7.3 a vyjadruje rozdiel medzi predikovaným a skutočným výstupom neurónovej siete.

$$J(W) = \sum_{i=1}^{n} l(x^{(i)},y^{(i)},W) + \lambda R(W) \tag{7.3}$$

Funkcia $l(x^{(i)},y^{(i)},W)$ určuje chybu pre jeden príklad v trénovacej množine. Potom $J(X,Y,W)$ vyjadruje celkovú chybu pre všetky trénovacie príklady. V druhej časti vzorca je implementovaná regularizácia, ktorá slúži na doplnkovú penalizáciu pre zlepšenie výkonnosti modelu.

Pri trénovaní neurónových sietí hrajú významnú úlohu rôzne hyperparametre, ktoré ovplyvňujú proces učenia [48]. Okrem zmien v štruktúre siete, ako je napríklad inicializácia váh, počet skrytých vrstiev a ich veľkosti, voľba aktivačných a chybových funkcií, existujú aj ďalšie hyperparametre, ktoré majú vplyv na trénovanie neurónovej siete:

- Počet epoch určuje, koľkokrát sa sieť naučí na celú trénovaciu množinu.
- Veľkosť skupiny (angl. batch size) je počet prvkov z trénovacej sady, ktoré sa používajú na úpravu váhy neurónov.
- Miera učenia (angl. learning rate) znamená rýchlosť úpravy váh neurónov po každom kroku trénovania.

V ďalšom odseku popíšeme algoritmus spätného šírenia chyby, ktorý sa využíva pri učení neurónových sietí. Popis pre jednoduchosť uvedieme pre jednu epochu, čo znamená jeden prechod trénovacou sadou. Pri skutočnom tréningu neurónovej siete by bolo nutné niekoľkokrát tento proces zopakovať.

Začiatok učenia neurónových sietí zahŕňa ich počiatočnú inicializáciu. Ak by sme však inicializovali váhy neurónov rovnakými hodnotami, zostali by počas učenia nezmenené. Preto sa používa náhodná inicializácia váh z intervalu $(-e;\ e)$,

pričom e je malé číslo z intervalu $(0;\ 1)$. Existujú aj algoritmy na inteligentnú inicializáciu váh, ktoré vedú k lepšiemu učeniu neurónovej siete.

Po inicializácii neurónovej siete sa trénovacie dáta prechádzajú postupne po jednom príklade x_i. Pri tomto prechode sa vypočíta dopredný prechod siete. Na základe výstupu siete sa potom vypočíta chyba δ, ktorá sa iteratívne počíta späť pre ďalšie skryté vrstvy.

Po každom trénovacom príklade sa vypočítajú gradienty na základe chyby pre každú vrstvu neurónovej siete. Tieto gradienty sú potom použité na úpravu váh siete v rámci metódy učenia, ktorú sme zvolili. Ak sa použije postupné akumulovanie gradientov, úprava váh sa uskutoční po dokončení prechodu cez celý trénovací súbor dát. Ak sa použije online učenie, váhy siete sa upravujú okamžite po každom trénovacom príklade.

7.3.3 Transfer learning

Transfer learning je metóda strojového učenia, ktorá spočíva v použití predtrénovanej neurónovej siete na riešenie novej úlohy. Aby bola táto sieť úspešná a dokázala dobre reprezentovať príznaky, je potrebné, aby bola trénovaná na veľkom súbore vstupných dát. Pod pojmom *„veľký súbor dát“* sa rozumie niekoľko stoviek gigabajtov až terabajtov dát.

Učenie modelu neurónovej siete na takomto rozsiahlom súbore dát môže odhadovo trvať niekoľko týždňov aj farme s grafickými kartami. Avšak jeho následné použitie už je jednoduché a časovo nenáročné vzhľadom na to, že takto natrénované modely sú verejne dostupné.

Pri konvolučných a rekurentných sieťach sa najčastejšie používajú dva prístupy v transfer learningu. Prvým prístupom je fine-tuning, ktorý spočíva v jemnej úprave už natrénovanej siete pre nový problém. Druhým prístupom je použitie siete ako extraktoru príznakov.

Sieť ako extraktor príznakov

Jeden z možných prístupov ako použiť predtrénovaný model neurónovej siete spočíva v tom, že sa ako vstup pre ďalšiu metódu strojového učenia použije výstup niektorej z vrstiev predtrénovanej siete. Vektor príznakov, ktorý je extrahovaný z tejto siete, sa tiež nazýva *„descriptor“*.

Je bežnou praxou využiť výstup z určitej vrstvy predtrénovanej neurónovej siete ako vektor príznakov pre iný model strojového učenia. Tento prístup sa často uplatňuje v úlohách ako klasifikácia obrázkov, pri ktorej môže posledná vrstva konvolučnej siete poskytnúť vstup pre algoritmus k-NN.

Fine-tuning

Druhou veľmi častou metódou transfer learningu je fine-tuning. Základom tejto techniky je použitie predtrénovaného modelu neurónovej siete alebo iba váh vrstiev na riešenie nového problému. Počas fine-tuningu stačí zmeniť iba niektoré vrstvy v sieti, obvykle poslednú alebo posledné dve vrstvy. Váhy neurónovej siete, ktoré sa nepotrebujú meniť, sa zamrazia a to urýchľuje učenie siete na novom súbore dát.

Konkrétnym príkladom použitia techniky fine-tuning je situácia, keď sa využije konvolučná neurónová sieť natrénovaná na datasete ImageNet. Tento model má vo svojich váhach pre jednotlivé vrstvy zakódované poznatky, ktoré sú schopné správne detegovať tvary, farby, prípadne hrany na obrázku. Tieto už naučené poznatky môžu byť následne využité pri riešení nových problémov z oblasti rozpoznania obrazu, akou je napríklad detekcia objektov. Všeobecné znalosti získané z predchádzajúcej trénovacej sady umožňujú rýchlejšie prispôsobenie sa novému problému.

7.4 Konvolučné neurónové siete

Aktuálne sú konvolučné neurónové siete (CNN) jednou z najpopulárnejších foriem neurónových sietí. Svoje najväčšie uplatnenie našli v oblasti počítačového videnia. Okrem toho sa však objavujú aj úspešné aplikácie v spracovaní prirodzeného jazyka a reči, na ktoré v tejto práci nadviažeme.

Primárna oblasť ich využitia nie je prekvapivá vzhľadom na to, že architektúra CNN je inšpirovaná fungovaním vizuálneho centra v mozgu cicavcov, čo znamenalo významný prínos v oblasti strojového učenia [49]. Lokálne prepojenia predstavujú základnú vlastnosť vizuálneho centra mozgu, ktorá bola prebratá do konvolučných neurónových sietí. Lokálne prepojenia umožňujú neurónom v skrytej vrstve pripojiť sa k oblasti neurónov v predchádzajúcej vrstve cez takzvané *„receptive field“*. Neurónovým sieťam to umožňuje lepšie zachytiť lokálne rysy [50]. Táto schopnosť môže byť žiaduca pre niektoré úlohy spracovania prirodzeného jazyka [51] aj napriek tomu, že konvolučné siete boli pôvodne navrhnuté pre spracovanie vizuálneho obsahu. Druhou dôležitou vlastnosťou týchto sietí je n-dimenzionálny rozmer vrstiev. Napríklad pre vizuálne dáta majú tri dimenzie pre každý kanál spektra RGB. Poslednou, ale nie menej dôležitou vlastnosťou je zdieľanie váh neurónov v konvolučnej vrstve v danej hĺbke, čo znamená, že všetky neuróny v tejto vrstve majú rovnaké váhy.

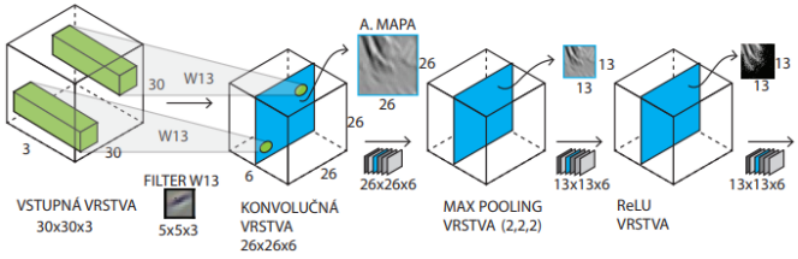
Konvolučné neurónové siete majú výhodu oproti klasickým neurónovým sieťam vďaka vlastnostiam, ktoré sme popísali v predchádzajúcich odstavcoch. Ak by sme sa rozhodli klasickou doprednou sieťou klasifikovať vizuálne dáta, museli by sme vytvoriť veľké množstvo neurónov pre každý pixel obrázka, s obrovským počtom prepojení na nasledujúcu vrstvu. Trénovanie siete s takouto architektúrou by bolo časovo náročne a sieť by mala tendenciu k preučeniu, práve z tohto dôvodu vznikli konvolučné siete, ktorých základnými vlastnosťami sa snažíme tieto problémy potláčať.

Tento typ siete má schopnosť naučiť sa zložité vzorce a vlastnosti zo vstupných dát. V prvých vrstvách siete sa učia detegovať jednoduché prvky ako čiary a hrany. Postupne sa naučia rozpoznávať zložitejšie prvky ako kruhy a viacfarebné prechody. V posledných vrstvách siete sa už učia rozpoznávať veľmi zložité tvary a objekty na vstupnom obrázku.

CNN sa skladajú zo vstupnej vrstvy a blokov, ktoré zahŕňajú vrstvy konvolúcie, poolingu a aktivačnú funkciu ReLu [52]. Pred výstupnou vrstvou siete sa nachádza plne prepojená vrstva neurónov, ktorá vyžaduje zhustenie výstupu z konvolučných blokov do jednej dimenzie. Výstup CNN siete sa častokrát používa ako vstup (descriptor) pre iné metódy strojového učenia v procese zvanom transfer learning.

7.4.1 Konvolučná vrstva

Táto vrstva siete vykonáva konvolučnú operáciu, ktorá spočíva v umiestnení filtra nad vstupnou maticou. Konvolučný filter je umiestnený tak, aby sa jeho horný ľavý roh prekrýval s horným ľavým rohom vstupnej matice. Následne sa hodnoty zo vstupnej matice postupne násobia príslušnými hodnotami z konvolučného filtra a súčet týchto hodnôt sa umiestni na príslušné miesto vo výslednej matici. Filter sa potom posúva po celej vstupnej matici a tento proces sa opakuje, až kým nie sú pokryté všetky prvky vstupu [52]. Na ilustrácii 7.3 môžeme vidieť vizualizáciu fungovania konvolučnej vrstvy na obrazových dátach.

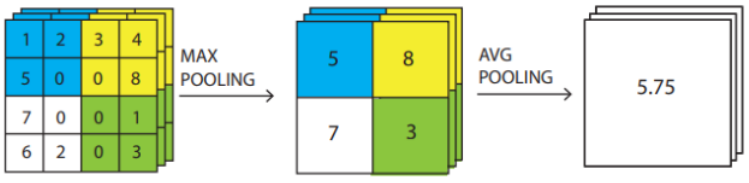


Obr. 7.3: Schematická vizualizácia konvolučnej siete[50].

7.4.2 Pooling vrstva

Pooling vrstva je kľúčovou súčasťou konvolučných sietí, ktorá slúži na zmenšenie priestoru vstupov z predchádzajúcich vrstiev. Tento proces znižuje počet parametrov v nasledujúcich vrstvách a zlepšuje rýchlosť učenia siete. Zároveň pomáha zlepšiť generalizáciu modelu [52].

Pooling vrstva má dva dôležité parametre: veľkosť okna/filtra a posun. Typické hodnoty sú 2 pre veľkosť filtra s rovnako veľkým posunom. Jednotlivé časti vstupnej mapy sa podrobia preddefinovanej operácii a pre každú časť sa vyberie reprezentatívna hodnota na základe zvoleného typu filtra. Maximálna hodnota z oblasti predstavuje MaxPooling vrstvu, zatiaľ čo priemerovanie hodnôt predstavuje AvgPooling.



Obr. 7.4: Operácia MaxPooling a AvgPooling pre data[50].

Pooling vrstva nepoužíva žiadne váhy a slúži iba na transformáciu vstupu na výstup, podobne ako aktivačná vrstva. Na ilustrácii 7.4 je znázornený priebeh tejto operácie.

7.4.3 ReLu vrstva

ReLU vrstva sa používa ako aktivačná vrstva v konvolučných sieťach a je oddelená do samostatnej vrstvy. Táto oddelená aktivačná vrstva umožňuje použiť aktivačnú funkciu na rôzne vrstvy, ako sú konvolučné vrstvy, pooling vrstvy alebo plne prepojené vrstvy. ReLU vrstva je preferovaná pred sigmoidálnymi funkciami pre jej rýchly výpočet a schopnosti riešiť problém miznúceho gradientu. ReLU vrstva sa často používa hneď po konvolučnej alebo plne prepojenej vrstve [52].

7.4.4 Plne prepojená vrstva

V plne prepojenej vrstve v CNN sa neuróny spoja so všetkými výstupmi z predchádzajúcej vrstvy. Váhy neurónov sa učia počas trénovania, aby dokázali identifikovať dôležité vzorce vo vstupných dátach. Takto sa vytvára hierarchia extrakcie príznakov, kde sa každá vrstva snaží identifikovať zložitejšie vzorce v dátach. Plne prepojená vrstva v CNN má za úlohu zlúčiť informácie z predchádzajúcich vrstiev a klasifikovať vstupné dáta do konkrétnych tried. Táto vrstva sa často používa v posledných vrstvách siete, kde sa na základe extrahovaných príznakov vykonáva klasifikácia vstupných príkladov. Teda cieľom plne prepojenej vrstvy je určiť skóre, ktoré indikuje priradenie vstupu do konkrétnej triedy.

7.4.5 Dropout

Vrstva Dropout sa používa v hlbokých neurónových sieťach na reguláciu trénovania a zlepšenie ich výkonnosti. Táto vrstva náhodne vynecháva (deaktivuje) jednotlivé neuróny v predošlej vrstve v procese trénovania. Pri každej iterácii trénovania sa neuróny vyberajú náhodne a deaktivujú sa podľa vopred stanoveného pravdepodobnostného rozdelenia. Tak Dropout zabezpečuje, že jednotlivé neuróny nebudú závislé od prítomnosti alebo výstupu ostatných neurónov. Táto technika má za úlohu znížiť možnosť pretrénovania a zlepšuje schopnosť generalizácie modelu, pretože vynucuje redundanciu v hlbokých sieťach.

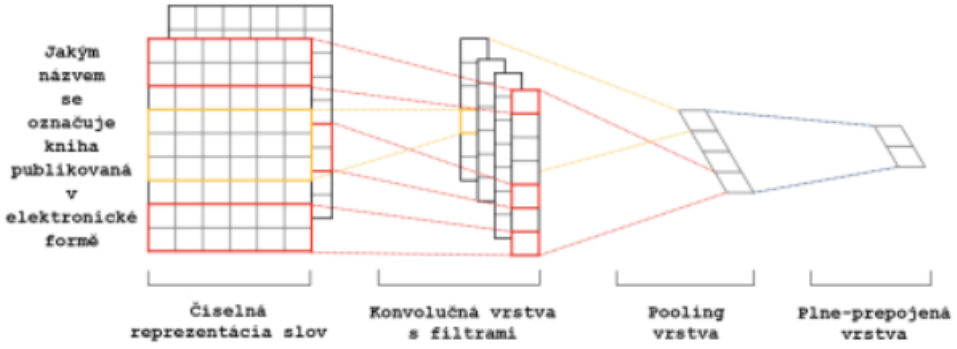
7.4.6 Učenie siete

Trénovanie konvolučných sietí sa v zásade vo väčšine prípadov nelíši od trénovania dopredných sietí. Prevažná množina konvolučných sietí sa učí pomocou algoritmu spätného šírenia chyby, rovnako ako klasické dopredné siete.

Využitie konvolučných neurónových sietí sa rozšírilo aj na spracovanie prirodzeného jazyka, v ktorom sa text reprezentuje ako jednorozmerné pole slov. Na začiatku tohto procesu sa slová z textu transformujú na vektory čísel určitej dimenzie, na ktoré sa následne aplikujú konvolučné filtre prispôsobené pre jednorozmerné vstupy. Výsledné vektory z konvolučných vrstiev sa potom spracujú pomocou ReLu vrstvy a prejdú ďalšou vrstvou. Posledný krok v konvolučnom bloku predstavuje aplikácia pooling vrstvy, ktorá pre každý filter zvlášť vyberie jednu hodnotu z mapy rysov a túto hodnotu považuje za najdôležitejší rys daného filtra. Ak sa používa maximalizačný filter, extrahuje sa maximálna hodnota, pri použití AvgPooling filtra sa vyextrahuje priemerná hodnota [51].

Výstup z konvolučného bloku siete sa ďalej posunie do plne prepojenej vrstvy, ktorej výstupom je distribúcia pravdepodobností priradenia do jednotlivých kla-

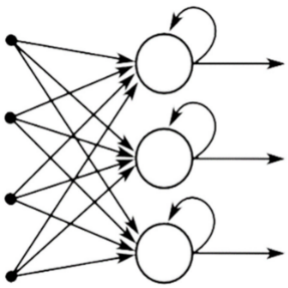
sifikačných tried. Na ilustrácii 7.5 je vizualizácia CNN, ktorá je uvedená aj s príkladom vo forme textu.



Obr. 7.5: Vizualizácia CNN na textových dátach [50].

7.5 Rekurentné neurónové siete

Rekurentná neurónová sieť je typ modelu, ktorý dokáže spracovávať sekvenčné dáta akejkoľvek dĺžky. Termínom sekvenčné dáta označujeme dáta obohatené o časový kontext. Príkladom týchto dát sú napríklad videozáznamy alebo vývoj cien komodít na burze.



Obr. 7.6: Vizualizácia rekurentnej neurónovej siete.

Rekurentné neurónové siete (RNN) sa od klasických dopredných neurónových sietí líšia tým, že v RNN existujú väzby medzi jednotlivými jednotkami, ktoré vytvárajú cykly. Tieto cykly umožňujú sieti vziať do úvahy aj informácie z predchádzajúcich stavov. Schému architektúry RNN možno vidieť na vizualizácii 7.6.

Rekurentné neurónové siete majú výhodu v tom, že umožňujú prácu s vstupnými sekvenciami variabilnej dĺžky, čo je v konvenčných dopredných neurónových sieťach nemožné (môžu pracovať iba so vstupom fixnej dĺžky). Táto schopnosť je kľúčová pre RNN a vychádza z faktu, že váhy siete sú zdieľané pre všetky časti sekvencie, čo umožňuje flexibilnejšie spracovanie vstupných dát. Tento prístup poskytuje výhodu pre trénovacie a testovacie dáta s rozdielnou dĺžkou sekvencií.

Rekurentná neurónová sieť je schopná čiastočne si zapamätať predošlý kontext vstupných sekvencií x pomocou skrytého stavu a rekurentných spojení medzi neurónmi alebo vrstvami. To znamená, že neuróny v skrytej vrstve majú dva druhy spojení - okrem spojenia s ďalšími vrstvami majú aj rekurzívne spojenia na seba. V každom kroku sa tieto spojenia používajú na aktualizáciu skrytého stavu, ktorý zodpovedá stavu neurónovej siete v danom čase t a zohľadňuje predošlý kontext.

Stav skrytej vrstvy h_t v RNN sa postupne aktualizuje a uchováva informácie o predchádzajúcich krokoch modelu, teda o kontexte. Tento stav vzniká ako výsledok rekurentného spracovania vstupov a kumuluje sa postupne do skrytého stavu pre ďalšie spracovanie.

Pre ilustráciu: ak máme rekurentnú neurónovú sieť s jedinou skrytou rekurentnou vrstvou, jej dopredný prechod sa skladá z troch krokov. V prvom kroku sa vstupná sekvencia prenesie do skrytého neurónu alebo vrstvy. V druhom kroku sa výstup z tejto skrytej vrstvy prenáša na výstup modelu. V treťom kroku sa výstup z tejto skrytej vrstvy prenáša do ďalšej skrytej vrstvy, kde sa použije ako vstup do ďalšieho kroku v rekurentnom spracovaní.

$$h_t = \sigma((W_{hh}h_{t-1}) + W_{xh}x_t + b_h) \tag{7.4}$$

Vzťah 7.4 vyjadruje stav h_t skrytej vrstvy v čase t, kde W_{xh} označuje spojenia, respektíve váhy zo vstupu do skrytej vrstvy, W_{hy} zo skrytej vrstvy na výstup a W_{hh} rekuretne zo skrytej vrstvy. Potom výstup siete y v čase t sa vypočíta pomocou nasledujúceho vzorca:

$$y = \sigma(W_{hy}h_t) \tag{7.5}$$

Rekurentné neurónové siete priniesli množstvo výhod, ale ich klasická verzia zápasí so závažným problémom, ktorým je miznúci a explodujúci gradient. V angličtine známy pod pojmom *„vanishing and exploding gradient problem“*. Tento problém je spôsobený princípom učenia a architektúrou RNN [53]. V procese učenia modelu sa pri každom spätnom prechode neurónovou sieťou gradient chybovej funkcie vynásobí biasom. Tento bias je zvyčajne menší alebo väčší ako 1 pre väčšinu relevantných príkladov. Po určitom čase trénovania klasických RNN na súbore dát s dlhodobými závislosťami môže dôjsť k problému miznúceho alebo explodujúceho gradientu chybovej funkcie.

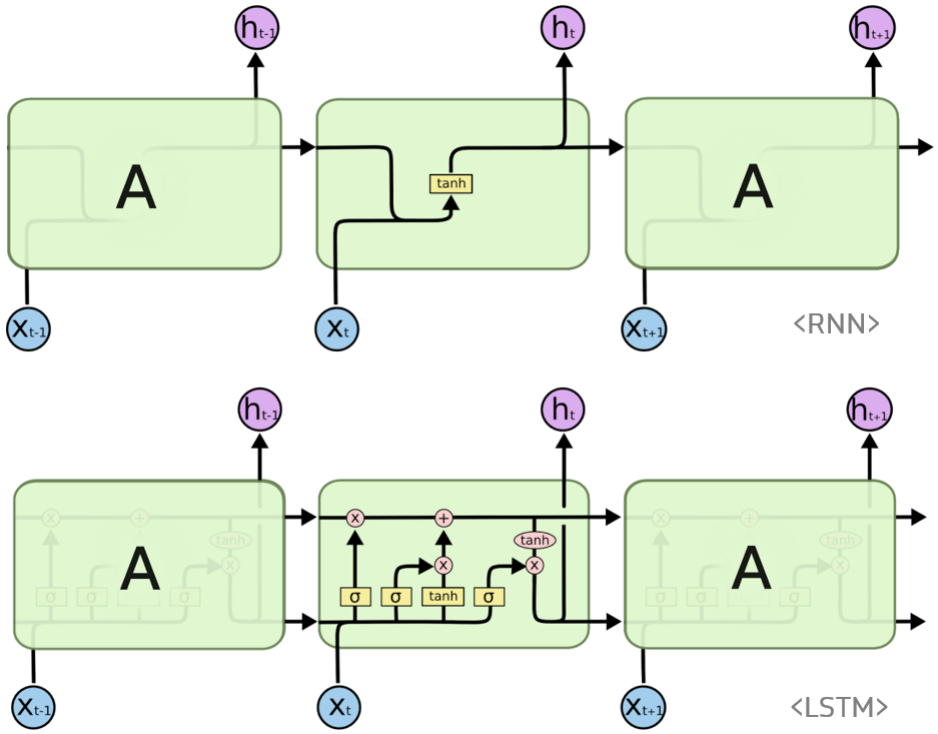
Klasické rekurentné neurónové siete sa v úlohách s dátami, ktoré majú dlhodobé závislosti, ukázali ako neefektívne pre problém miznúceho gradientu. Preto boli vyvinuté vylepšenia klasických rekurentných neurónových sietí, ktoré tento problém eliminujú, napríklad Long Short-Term Memory [53].

7.5.1 LSTM

Modifikácia architektúry rekurentnej neurónovej siete, známa ako Long Short-Term Memory (LSTM), rieši problémy s miznúcim a explodujúcim gradientom, ktoré sa vyskytujú pri učení klasických rekurentých sietí, vďaka čomu sa dokážu efektívne učiť aj na dátach s dlhodobými závislosťami.

Vyriešenie problému miznúceho a explodujúceho gradientu v klasických RNN spočívalo v úprave architektúry samotnej výpočtovej jednotky tejto siete [53]. Pamäťová bunka v LSTM ukladá stav, ktorý sa modifikuje pomocou troch brán - vstupnej, pamäťovej a výstupnej. Každá brána je reprezentovaná sigmoidnou

vrstvou, ktorá určuje, do akej miery má ovplyvniť súčasný stav pamäťovej bunky. Táto bunka je súčasťou reťazovej štruktúry rekurentnej neurónovej siete a je znázornená na obrázku 7.7.



Obr. 7.7: Porovannie RNN a LSTM [54].

V súvislosti s vývojom komplexnejších LSTM sietí pre spracovanie prirodzeného jazyka bolo publikovaných množstvo štúdií, ktoré sa zameriavali na použitie nových učiacich algoritmov a pokročilých architektúr, najmä na metódu Long Short-Term Memory (LSTM), ktorú navrhli Hochtreiter a Schimhuber [53]. Vo svojej štúdii navrhli vytvoriť, ako už sme spomenuli, dodatočné pamäťové brány, ktoré by boli zodpovedné za úpravu vnútorného stavu. Vstupná brána by mala chrániť vnútorný stav pred odchýlkami spôsobenými nerelevantným vstupom, výstupná brána má za úlohu chrániť ostatné jednotky v sieti pred zbytočnými informáciami v pamäti.

7.6 Rekurentné konvolučné sieťe

Rekurentná konvolučná sieť (RCNN) vo svojej architektúre kombinuje vlastnosti CNN a RNN. Cieľom je využiť silné stránky oboch typov sietí na riešenie úloh, ktoré sú vhodné pre jednu alebo druhú architektúru. Napríklad RNN sa hodia na spracovanie časových radov a na zachytávanie dlhodobých závislostí medzi vstupmi a výstupmi, zatiaľ čo CNN sú účinné pri vyhľadávaní vzorov v obrazoch a používajú sa často na úlohy klasifikácie obrazov.

V architektúre neurónových sietí sa kombinujú vlastnosti oboch tak, aby sa dosiahli lepšie výsledky. Liang a Hu popisujú kombinovanú architektúru siete na detekciu objektov v jednej zo svojich prác a podobnú architektúru použili

aj na označenie scény [55]. Vo svojich prácach túto kombinovanú architektúru označujú pojmom RCNN. Nasledujúci citát popisuje ich hlavnú myšlienku:

„A prominent difference is that CNN is typically a feed-forward architecture while in the visual system recurrent connections are abundant. Inspired by this fact, we propose a recurrent CNN (RCNN) for object recognition by incorporating recurrent connections into each convolutional layer [55].“

Základnou stavebnou jednotkou RCNN je rekurentná konvolučná vrstva, ktorej skratka je RCL. Táto vrstva v sebe kombinuje prvky konvolučných a rekurentných neurónových sietí. RCL sa používajú na to, aby sa zachytili dlhodobé závislosti medzi vstupmi a výstupmi v sieti, ako sú schopné urobiť RNN. Ale tiež sú navrhnuté tak, aby využili výkon konvolučných vrstiev na vyhľadávanie vzorov v dátach. Rekuretná konvolučná vrstva dokáže teda vyhľadávať vzory v dátach a zároveň zachytávať dlhodobé súvislosti v nich.

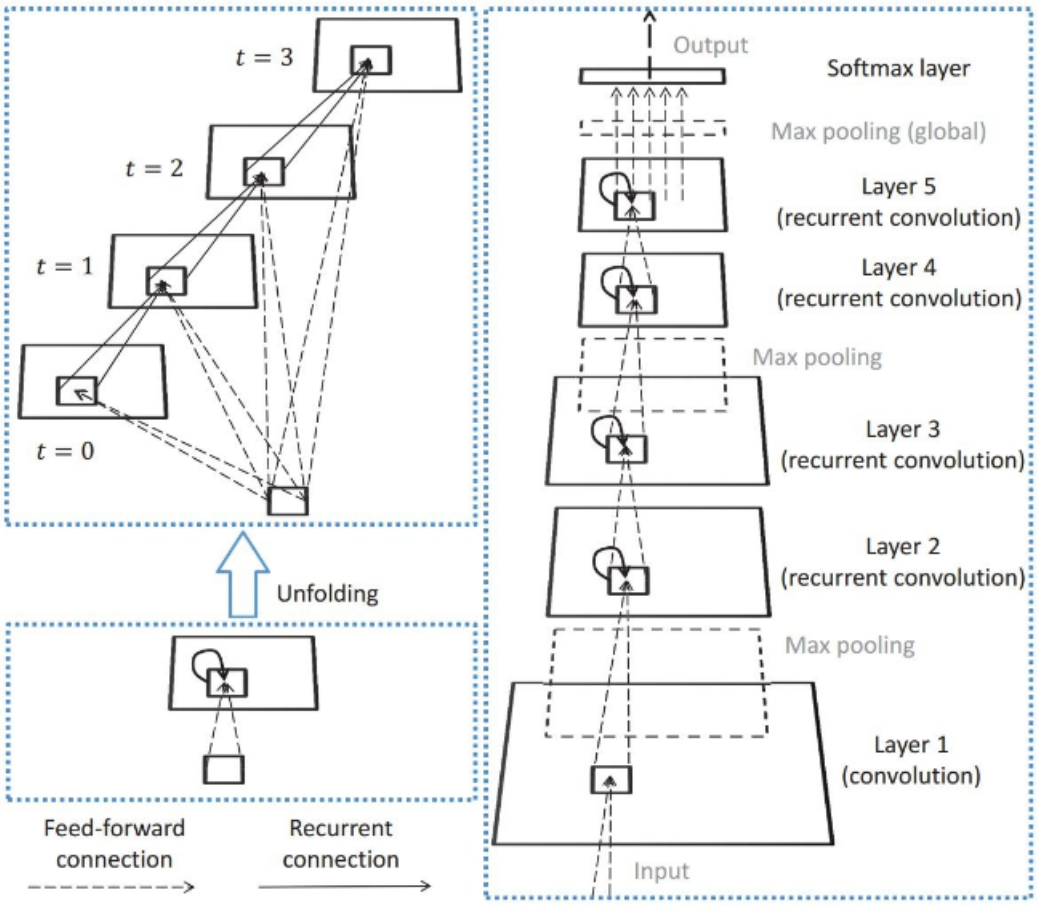
Obr. 7.8: Dôležitosť kontextových informácií pri rozpoznávaní nosa alebo úst [55].

Pri detekcii objektov na vizuálnych dátach záleží na okolí, teda kontexte. Napríklad dôležitosť kontextových informácií pri úlohe rozpoznávania nosa alebo úst spočíva v tom, že tieto časti ľudského tela sa nachádzajú v konkrétnom kontexte v rámci tváre ako celku a v rámci celého obrázka. Kontextové informácie poskytujú dôležité súvislosti, vďaka ktorým môžeme lepšie pochopiť a rozpoznať jednotlivé časti tváre. Napríklad bez znalosti kontextu môže byť ťažké rozpoznať ústa alebo nos v izolácii od ostatných častí tváre a okolia, ako je možné vidieť na obrázku 7.8. Pre klasické CNN je riešenie tejto úlohy problematické. Avšak kľúčovým modulom sú opakujúce sa konvolučné vrstvy (RCL) v RCNN, ktoré do konvolučnej vrstvy zavádzajú opakujúce sa spojenia. S týmito spojeniami sa CNN môže vyvíjať v čase, aj keď vstup je statický a každá jednotka je ovplyvnená iba svojimi susednými jednotkami. Táto vlastnosť integruje kontextové informácie o obrázku, čo je dôležité pre detekciu objektov. Dôležitosť kontextových informácií je znázornená na obrázku 7.8. Na tomto obrázku je veľmi ťažké rozpoznať ústa alebo nos bez kontextu [55].

Na obrázku 7.9 je znázornená schéma RCNN, v ľavej časti ilustrácie je znázornená rekurentná konvolučná vrstva počas troch časových krokov, to smeruje na doprednú neurónovú sieť s hĺbkou štyri. Napravo je zobrazené RCNN použité v práci *Recurrent Convolutional Neural Network for Object Recognition* s jednou

konvolučnou vrstvou, štyrmi RCL, tromi maxpoolingovými vrstvami a jednou softmaxovou vrstvou.

Tréning týchto umelých neurónových sieti prebieha algoritmom *Backpropagation through time (BPTT)*, ktorý sa používa pre učenie rekurentných neurónových sietí (RNN). Tento algoritmus učenia vychádza z pôvodného algoritmu Backpropagation, ktorý sa používa na učenie klasických neurónových sietí, ale muselo dôjsť k modifikáciám, aby bol schopný pracovať s rekurentnými vrstvami. Algoritmus rozdelí vstupný časový rad do jednotlivých krokov a pre každý krok vypočíta chybu medzi skutočným výstupom a očakávaným výstupom. Potom sa chyba „backpropaguje“ cez sieť a používa sa na aktualizáciu váh v rekurentných vrstvách, aby sieť bola schopná lepšie predikovať výstupy aj na základe časovej postupnosti.



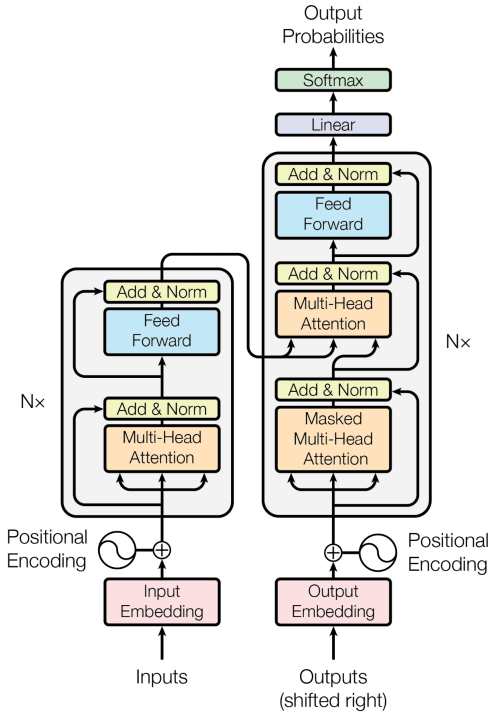
Obr. 7.9: Schematická vizualizácia rekurentnej konvolučnej siete [55].

RCNN podľa výsledkov, ktoré uvádzajú vo svojich prácach Liang a Hu, dosahujú lepšie výsledky ako CNN pri spracovaní vizuálnych dát (hlavne v oblasti detekcie objektov) a porovnateľné alebo mierne lepšie výsledky ako RNN [55]. Ďalšou ich výhodou oproti CNN je, že sú menej náchylné na pretrénovanie. Ďalej uviedli, že tieto siete vo svojej podstate napodobňujú biologické fungovanie mozgu, a preto by sa mal na ne v blízkej budúcnosti klásť väčší dôraz. Na základe týchto výsledkov sme sa rozhodli zakomponovať RCNN do tejto práce a využiť ich na detekciu falošných správ na slovenských textoch.

8. Architektúra Transformer

V roku 2017 Vaswani a jeho kolegovia predstavili model neurónovej siete založenej na architektúre Transformer, ktorý vynikal v prekladateľských úlohách z angličtiny do francúzštiny (WMT 2014 English-to-French) a do nemčiny (WMT 2014 English-to-German). Tento model bol popísaný v článku k štúdii s názvom *„Attention is All You Need“*, ktorý publikovali Vaswani a jeho spoluautori [56].

Transformer je tvorený dvoma primárnymi blokmi - prvým z nich je Encoder, ktorý sa tiež nazýva v slovenskom jazyku kódovač alebo kódovací blok; a druhým blokom je Decoder. Tieto dva bloky sú kľúčovými časťami architektúry Transformer. Podrobná vizualizácia tejto architektúry je zobrazená na ilustrácii 8.1. Architektúra modelu Transformer, ktorá je zobrazená na obrázku 8.1, je tvorená šiestimi kódovacími a dekódovacími blokmi zoradenými za sebou.



Obr. 8.1: Architektúra modelu Transformer [56].

.

Attention

Najdôležitejšou súčasťou modelov neurónových sietí založených na architektúre Transformer je komponent s názvom Attention. Tento komponent nahradil dovtedy používané sequence-to-sequence modely [57].

Sequence-to-sequence model pracuje tak, že sa na začiatku jednotlivo spracujú pomocou Encoder bloku všetky vstupné slová. Táto úplná informácia o vstupe, ktorá má pevne daný rozmer, je následne odovzdaná Decoder bloku, ktorý postupne generuje výstupné slová. Problém však môže nastať pri dlhých vetách alebo textoch, kde nemusí byť možné zakódovať informáciu o všetkých vstup-

ných slovách. Bahdanau a jeho spolupracovníci prišli s novou myšlienkou v oblasti strojového prekladu, ktorá umožňuje modelu vybrať si, na ktoré slová vstupného textu sa má pozerať. Teda sa na ne zamerať pomocou bloku Attention. Týmto spôsobom mohol model kódovať každé vstupné slovo do číselnej reprezentácie a potom použiť Attention blok na výber tých reprezentácií, ktoré sú dôležité pre výpočty. Táto myšlienka bola revolučná a priniesla nový prístup k prekladu textov pomocou neurónových sietí [57].

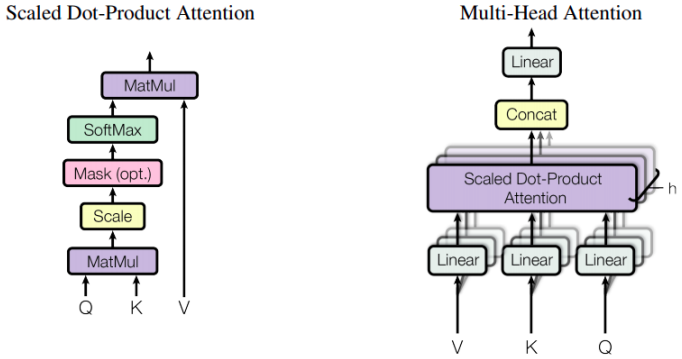
Implementácia Attention vrstvy v transformeroch

Transformery využívajú Attention bloky na to, aby mohli pri spracovaní textov sústrediť svoju pozornosť na relevantné časti. Attention vrstva v transformeroch pracuje s tromi typmi vektorov. Ich názvy sú key (kľúč), value (hodnota) a query (požiadavka).

Výstup z bloku sa počíta ako vážený súčet values, kde váhy zodpovedajú kompatibilite zodpovedajúceho key a query [56]. Pre každé vstupné slovo sa najskôr spočíta dvojica key-value. Následne počas iterácie výpočtu si model vyžiada query ako požiadavku pre daný krok.

V ďalšom kroku je uskutočnené porovnanie query s každým key. Týmto porovnaním získame skóre pre daný key. Na výpočet skóre sa používa skalárny súčin, pretože key aj query sú číselné vektory s rovnakou dimenziou.

Posledným krokom je transformácia skóre na váhy pomocou softmax funkcie. Táto funkcia zabezpečí, že hodnoty skóre budú v intervale $\langle 0;\ 1\rangle$ a budú v súčte dávať 1, ale so zachovaním rovnakých relatívnych rozdielov medzi jednotlivými hodnotami skóre. Nové values získané pomocou softmax funkcie určujú, ako dôležitá je daná value v procese výpočtu výstupu modelu.



Obr. 8.2: Výpočet Attention v transformeri [56]. Ľavá časť ilustruje, ako prebieha výpočet Attention query (Q), key (K) a value (V). Pravá časť obrázka ukazuje, ako funguje Attention s väčším počtom hláv.

Na obrázku 8.2 je ilustrácia výpočtu Attention v architektúre Transformer. V publikácii pomenovali tento princíp výpočtu *Attention Scaled Dot-Product Attention*, pretože využíva skalárny súčin. Proces, ktorý sme opísali vyššie, predstavuje výpočet jednej Attention hlavy v architektúre Transformer. Je pozoruhodné, že jeden blok Attention môže obsahovať viac takýchto hláv. Zvýšenie počtu hláv v Attention bloku pridáva len všetkým týmto výpočtom a vektorom v jednotlivých krokoch o dimenziu viac.

Self-Attention

Nápad na vytvorenie a použitie Attention vrstvy v modeloch neuronových sietí, ktoré slúžia na spracovanie textu, prišiel pri riešení prekladateľských úloh. Prvotná verzia Attention potrebovala pre svoje správne fungovanie údaje zo vstupu modelu aj z jeho výstupu. Bolo potrebné zaistiť, aby model mohol vidieť nielen vstupné slová, ale aj slová, ktoré už sám vygeneroval. Takýto mechanizmus sa odlišuje od mechanizmov Self-Attention používaných v dnešnej architektúre Transformer.

Modely neurónových sieti založené na architektúre Transformer používajú mechanizmus zvaný Self-Attention, ktorý je odlišný od pôvodného Attention. Tento mechanizmus Attention sa vzťahuje na rôzne pozície jedného reťazca slov tak, aby vypočítal reprezentáciu tohto reťazca [56]. Self-Attention v architektúre Transformer si pre každé slovo určí pár key-value a následne v každej iterácii aj query.

Ilustrácia 8.1 znázorňuje Transformer a jeho tri rôzne Attention vrstvy. Medzi Self-Attention vrstvy patrí Attention vrstva v Encoder bloku a v Decoder bloku umiestnená dole. Posledná Attention vrstva, ktorá je znázornená na obrázku vo vnútri Decoder bloku, sa líši od týchto dvoch, pretože sa zameriava na vstupné dáta pre Encoder blok. Informáciu o tom, na ktorú časť vstupu sa má zamerať, však získava zo vstupných dát pre Decoder blok. Túto vrstvu v architektúre Transformer nazývame Encoder-Decoder Attention [58].

Encoder

V neurónových sieťach založených na architektúre Transformer so vstupom do siete ako prvý pracuje Encoder blok. Tento blok sa skladá z dvoch vrstiev, prvou z nich je Self-Attention vrstva. Na ňu nadväzuje druhá časť, ktorou je klasická dopredná neurónová sieť (feed-forward).

Reziduálne spoje sú viditeľné na ilustrácii 8.1 a obklopujú oba komponenty Encoder bloku. Tieto spoje zabezpečujú, že pôvodná vstupná informácia sa propaguje do nasledujúcich blokov. Na konci sa nachádza normalizačná funkcia s názvom layer normalization.

Výstup získaný pomocou aktuálneho Encoder bloku je potom odoslaný na vstup do nasledujúceho Encoder bloku. Ak však ide o posledný Encoder blok, potom je jeho výstup posunutý na vstup do všetkých Decoder blokov.

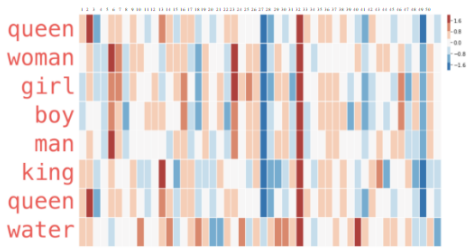
Decoder

Decoder blok má tri časti, ktoré zahŕňajú Self-Attention vrstvu a doprednú neurónovú sieť podobne ako v prípade Encoder bloku. Avšak medzi tieto dve vrstvy je vložená Encoder-Decoder Attention vrstva, ktorej úlohou je vypočítať Attention na základe výstupu Encoder bloku. To znamená, že vstupom pre Decoder blok je doterajší výstup modelu.

Decoder blok sa odlišuje ešte od Encoderu v jednej veci. Vstupný vektor do tohto komponentu transformeru má vždy fixnú dĺžku. To je docielené vyplnením prázdnymi slovami sprava. Následne je nutná ešte úprava prvého Attention bloku tak, aby sa nepozeral na nasledujúce pozície. To je docielené tak, že pred výpočtom softmax funkcie je skóre na týchto doplnených pozíciách prepísané hodnotou $-\infty$, čo spôsobí, že softmax ich ohodnotí 0.

Vstupný embedding

V modeli založenom na architektúre Transformer sa na vstupe nespracovávajú priamo slová alebo vety, ale vstupný text sa najskôr rozdelí na tokeny, ktoré môžu reprezentovať celé slová, ich časti alebo aj jednotlivé znaky. Tokeny sú potom zakódované pomocou kódovacieho algoritmu, ktorý ich prevedie na prirodzené čísla. Jednou z možností kódovania tokenov je ich identifikačné číslo v kódovacej tabuľke.



Obr. 8.3: Príklad GloVe vstupného embeddingu na anglických slovách [59].

Využitie vstupného embeddingu na reprezentáciu slov umožňuje, aby slová s podobným významom mali podobné vektory. Táto podobnosť je dosiahnutá tým, že každé slovo je reprezentované vektorom numerických hodnôt. Model siete sa naučí túto konkrétnu reprezentáciu slov počas trénovania.

Pri použití architektúry Transformer pre spracovanie textu je potrebné najskôr v prvom kroku ešte upraviť tento kód jednotlivých tokenov pomocou one-hot kódovania, pretože model neprijíma kód tokenu priamo, nevie s ním pracovať. One-hot kódovanie funguje tak, že pre danú veľkosť slovníka vytvorí vektor s nulami na všetkých pozíciách a na pozícii, ktorá zodpovedá kódu tokenu v slovníku, je umiestnená hodnota jeden. Pre príklad, ak chceme zakódovať token s kódom 2 a veľkosť slovníka je 5, výsledný vektor bude mať tvar [0 0 1 0 0].

Vstupný embedding je vlastne tabuľka, ktorá každému tokenu priradí jeho konkrétny vektor hodnôt. Technicky je implementovaný ako matica s dimenziami $s \times d$, kde „s“ reprezentuje veľkosť kódovacieho slovníka a „d“ je dimenzionalita modelu, t.j. požadovaná veľkosť vstupu. V prípade BERT-base modelu to je matica 30522×768. Daný token teda bude ohodnotený 768 rôznymi zložkami, ktoré sa model naučil pre dané slovo pri tréningu.

Na ilustrácii 8.3 je príklad vstupného embeddingu GloVe. Ak pozornejšie preskúmame tento vstupný embedding, môžeme vidieť, že všetky podstatné mená sa zhodujú v 31. zložke, ktorá je reprezentovaná bordovou farbou. S najväčšou pravdepodobnosťou táto značka nesie informáciu o slovnom druhu vstupného slova. Za povšimnutie stojí aj značka číslo 26, všetky životné podstatné mená majú v tejto zložke tmavomodrú hodnotu, iba water (voda) sa od týchto slov odlišuje. Z toho môžeme usúdiť, že táto časť nesie informáciu o životnosti objektu, ktorý slovo popisuje. Ďalšie informácie, ktoré môžeme vyčítať z vizualizácie, sú slová rovnakého pohlavia, napríklad man (muž) a boy (chlapec) majú podobné niektoré úseky, na druhej strane slová ako king (kráľ) a queen (kráľovná) sú podobné v iných úsekoch.

Pozičné kódovanie

Ďalšou dôležitou súčasťou architektúry Transformer je pozičné kódovanie, ktoré umožňuje modelu brať do úvahy aj poradie jednotlivých vstupných tokenov. To je veľmi dôležité, pretože v modeloch neurónových sietí založených na architektúre Transformer sa nepoužívajú konvolučné ani rekurentné vrstvy, ktoré by zabezpečili poradie tokenov. Pozičné kódovanie sa pripočítava k vstupnému embeddingu a zabezpečuje, aby pozícia každého tokenu bola zakódovaná do modelu. Bez tohto kódovania by model nedokázal rozlišovať poradie slov vo vete. Táto myšlienka bola rovnako prvýkrát predstavená v článku *„Attention is All You Need“*, ktorý publikovali Vaswani a kolektív v roku 2017 [56].

Každá pozícia vo vstupnom texte má svoj jedinečný vektor, ktorý sa vytvára pomocou pozičného kódovania. Pri výpočte tohto vektora sa využívajú goniometrické funkcie sínus a kosínus. Pozičné kódovanie je kľúčové pre zachytenie poradia slov vo vete, ktoré by inak nebolo zohľadnené v architektúre Transformer.

Pre párne pozície sa používa funkcia sínus a pre nepárne kosínus. Vzorec na výpočet i-tej zložky slova na pozícii *pos* je definovaný pomocou tejto funkcie:

$$PE_{pos,2_i} = \sin\left(\frac{pos}{10000^{\frac{2_i}{d_{model}}}}\right), \tag{8.1}$$

$$PE_{pos,2_{i+1}} = \cos\left(\frac{pos}{10000^{\frac{2_i}{d_{model}}}}\right), \tag{8.2}$$

kde parameter d_{model} vyjadruje dimenzionalitu modelu.

Táto funkcia je špeciálne navrhnutá tak, aby sa model mohol naučiť pracovať s relatívnou pozíciou vstupných tokenov. Pretože každý pevný posun o k v pozícii sa dá vyjadriť ako lineárna kombinácia PE_{pos+k} a PE_{pos}, kde PE_{pos} je pozičné kódovanie pre pozíciu *pos*.

Výstup

Architektúra Transformer modelu zahŕňa výstupnú vrstvu, ktorá sa nachádza na konci Decoder bloku a pozostáva z plne prepojenej vrstvy neurónov. Táto vrstva produkuje skóre pre každý token. Na výstup Decoder bloku sa pripája softmax vrstva, ktorá prevedie skóre na pravdepodobnosti pre každý token. Tento výstup je finálnym výstupom jednej iterácie architektúry Transformer. Potom sa tento výstup použije ako vstup Decoder bloku na generovanie ďalšieho tokenu v ďalšom kroku.

8.1 BERT

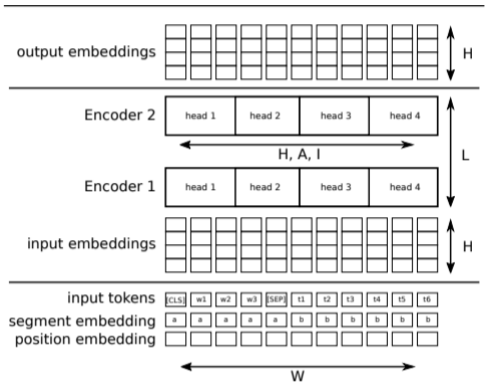
V tejto časti bližšie predstavíme BERT, hlavný model, ktorý budeme používať v tejto práci. BERT je skratka pre *Bidirectional Encoder Representations from Transformers*, je to jazykový model, ktorý využíva aplikáciu obojsmerne učeného transformeru (mechanizmus pozornosti). Tento model je založený na architektúre Transformer a bol navrhnutý Devlinom a kol. v roku 2018 [60].

BERT je založený na myšlienke transferového učenia. Najprv sa tento model musí natrénovať na veľkej množine počiatočných dát, ale potom sa dá relatívne rýchlo doladiť a prispôsobiť na riešenie širokej škály úloh. Aj keď je proces trénovania modelu náročný, je nutné ho vykonať len raz a predtrénovaný model BERT je verejne dostupný.

8.1.1 Architektúra modelu

Architektúra modelu BERT je založená na Encoder bloku z Transformeru. V nasledujúcej časti zadefinujeme jednotlivé parametre BERT modelu, ktoré korešpondujú s veľkosťami BERT-base modelu.

- V – veľkosť slovnej zásoby (30 000)
- W – maximálna dĺžka vstupnej sekvencie (512)
- L – počet vrstiev Encoderu (12)
- H – hidden size, rozmer key, value, query pre Attention v Encoderi (768)
- I – intermediate size, rozmer doprednej neurónovej siete Encodera (3072)
- A – počet Attention hláv (12)



Obr. 8.4: Schéma architektúry modelu BERT vrátane parametrov [60].

8.1.2 Tokenizátor

BERT používa tokenizér WordPiece [61]. WordPiece je jednoduchý tokenizér, ktorý používa greedy algoritmus na rozdelenie každého slova na tokeny. Jeho slovná zásoba má dve časti: začiatočné tokeny (celé slová alebo predpony) a nasledovné tokeny (prípony a infixy). Napríklad slovo *flying* je rozdelené na tokeny *fly* a *ing*. Vzhľadom na to, že WordPiece je interný nástroj spoločnosti Google, vydali iba proces tokenizácie a nie proces tvorby slovnej zásoby. Z tohto dôvodu boli vývojári nútení pre nové modely vytvoriť svoje vlastné súbory so slovnou zásobou.

8.1.3 Vstup

Vstupné tokeny sú najprv zakódované pomocou one-hot kódovania a potom premietnuté do vektorov dimenzie H, ktorých počet je rovný počtu Attention hláv (12). Tento vstupný embedding je totožný ako v architektúre Transformer, inicializované sú náhodne a naučené počas predtrénovania.

Prvý vstupný token je vždy [CLS]. Pre kontextualitu modelu môže jeho embedding obsahovať informácie o celej vstupnej sekvencii. Embedding [CLS] sa zvyčajne používa v klasifikačných úlohách.

Ďalšou dôležitou informáciou je správna identifikácia pozície jednotlivých tokenov v sekvencii. BERT využíva pozičné kódovanie ako paralelný vstup k tokenu. Detailnejšie informácie o pozičnom kódovaní sú popísané v kapitole 8.

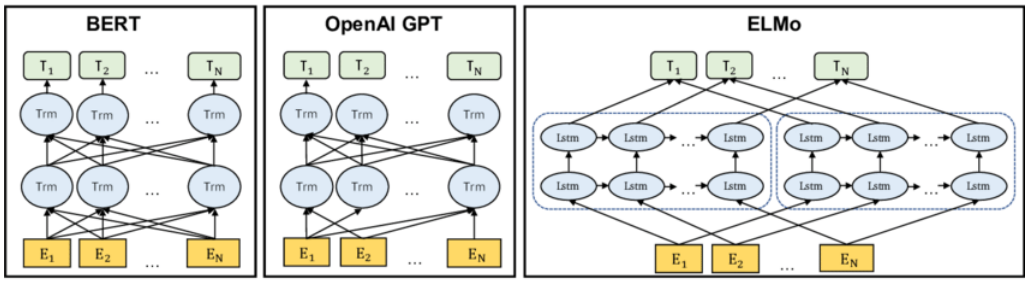
Niektoré úlohy NLP, napríklad odpovedanie na otázky, vyžadujú, aby bol model schopný akceptovať viacero sekvencií. Jedným z príkladov takého vstupu je kombinácia (<„Otázka“, „Odpoveď“>). Architektúra BERT to rieši dvoma spôsobmi. Prvým je použitie špeciálneho tokenu [SEP] na oddelenie sekvencií. Druhým je embedding segmentov. Embedding segmentu je paralelný vstup k tokenom a ukazuje, do ktorej sekvencie token patrí. Tento embedding sa učí počas trénovania modelu.

Výsledný embedding je súčtom všetkých troch vyššie uvedených - vstupného, pozičného a segmentového embeddingu. Ten sa následne posunie na vstup do Encoder bloku. Každý výsledný vektor je stále iba výsledkom pre jeden vstupný token. Potom embedding celého vstupu je dimenzia (W, H).

8.1.4 Encoder vrstvy

Každá z vrstiev Encodera vykonáva multi-head Attention na každom tokene rovnakým spôsobom ako v transformeri. Tento proces vytvára kontextový embedding. Existuje L vrstiev Encodera, každá z nich má A hláv, ktoré zdieľajú spolu H neurónov. Každá hlava má H/A neurónov (dimenzie query, key a value) v prípade BERT-base je to 64. Dopredná neurónová sieť, ktorá je v Encoder bloku, má veľkosť každej Attention vrstvy rovnú hodnote I.

Rovnako ako pri transformeri sa na všetky vstupné embeddingy aplikujú rovnaké váhy. Rozmer tensora, ktorý vystupuje z Encoder bloku, je dimenzia (W, H).



Obr. 8.5: Porovnanie architektúr BERT, GPT A ELMo [60].

Na rozdiel od ELMo, ktorá má zreteľné prechody zľava doprava a sprava doľava, alebo GPT [62], ktorá má iba ľavý kontext, má BERT reálny obojsmerný kontext.

8.1.5 Predtrénovanie

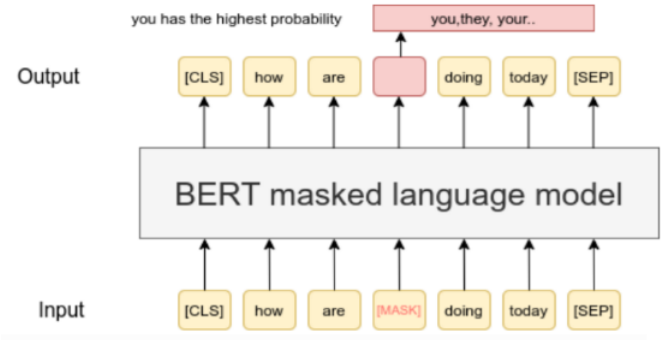
Autori navrhli dve samostatné úlohy na predtrénovanie, ktoré sa dajú automaticky vygenerovať z korpusu. Týmito predtrénovanými úlohami sú Masked Language Modelling a Next Sentence Prediction. Následné prispôsobenie modelu pre iné úlohy sa vykonáva pripojením ďalších vrstiev na základnú časť modelu.

Masked Language Modeling

Prvá úloha, na ktorú bol model BERT predtrénovaný, je predikcia slova na základe kontextu vo vstupnej sekvencii. Pre tento účel sa využila technika *Masked language model (MLM)*, pri ktorej sa 15% tokenov zo vstupu náhodne zamaskuje. Pre správne určenie kontextu musí sekvencia tokenov začínať špeciálnym tokenom [CLS] a končiť špeciálnym tokenom [SEP].

V prípade, že sa token/slovo maskuje, slovo sa nahradí iným slovom v 10% prípadov, v ďalších 10% zostáva nezmenené a v 80% prípadov je slovo nahradené špeciálnym maskovacím tokenom [MASK].

Cieľom modelu je správne predpovedať pôvodný token iba na základe obojsmerného kontextu. V diagrame na obrázku 8.6 je tento postup zobrazený graficky.



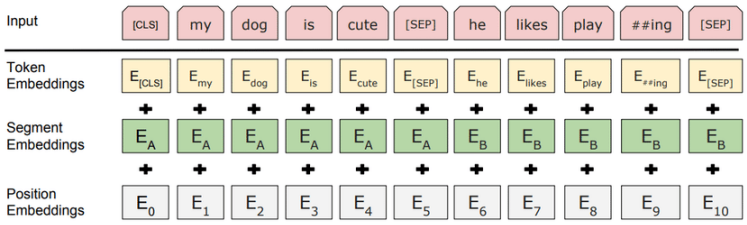
Obr. 8.6: Masked Language Modeling [63].

Next Sentence Prediction

Druhou hlavnou úlohou, ktorú BERT rieši, je úloha NSP. Je zvlášť užitočná pre všetky iné úlohy založené na porozumení vzťahu medzi dvoma vetami (napr. odpovede na otázky alebo odvodenie z prirodzeného jazyka).

Túto úlohu môžeme binarizovať tak, že vzhľadom na dvojicu viet A a B trénujeme model na pochopenie toho, či ide o po sebe idúce vety, alebo nie. Pri danej dvojici viet je cieľom tejto úlohy rozhodnúť, či druhá veta bola v pôvodnom texte hneď po prvej. Na zodpovedanie tejto úlohy model používa kontextové vloženie tokenu [CLS] (ktorý je vždy prvým vstupným tokenom) v binárnom klasifikátore.

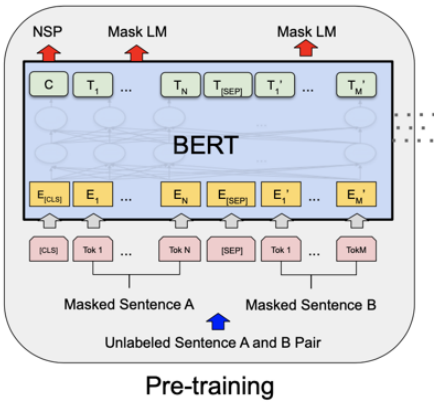
Napríklad v ideálnom prípade model neklasifikuje vety *„The child plays in the park..“* a *„The catcher in the rye is a great book.“* ako po sebe idúce vety.



Obr. 8.7: Reprezentácia vstupného embeddingu pre model BERT. Vyjadrená ako súčet tokena, segmentu a pozičného kódovania [60].

Pre obe úlohy používa BERT separátnu stratovú funkciu v podobe cross-entropy. V predtréningu je výsledná stratová funkcia (loss) vyjadrená jednoducho pomocou lineárnej kombinácie týchto dvoch. Aby sme však tieto úlohy splnili, musíme mierne upraviť vstupné embeddingy generované napríklad pomocou vopred trénovaného WordPiece tokenizéra. Robíme to pridaním embeddingu pozície (z dôvodu vysvetleného v predchádzajúcich častiach) a embeddingu segmentu do každého embeddingu tokenu. Toto sa robí s cieľom poskytnúť modelu BERT informáciu, či token patrí do prvej alebo druhej vety pre úlohu NSP.

Obrázok 8.7 zobrazuje vstupnú reprezentáciu, ktorá vstupuje do BERT modelu. Okrem toho dodávame vstupom dva špeciálne tokeny: token [CLS] označujúci začiatok sekvencie a token [SEP], ktorý označuje koniec vety, t. j. umiestnený medzi párom viet a na konci celej sekvencie. Finálny embedding tokenu [CLS], skratka pre klasifikačný token, ako už naznačuje jeho názov, sa používa na konečnú klasifikáciu v rámci klasifikačnej úlohy. Experimenty Clarka a kol. (2019) ukazujú [64], že väčšina informácií, t. j. entropia, obsiahnutá vo výstupnej sekvencii, je zahrnutá v tomto embeddingu tokenu, ktorý preto možno považovať za určitý druh embeddingu viet. Takže združovacia vrstva na obrázku 8.7 nemá byť zamýšľaná ako zvyčajná združovacia vrstva (ako je maximálne alebo priemerné združovanie), ale skôr ako vrstva, ktorá združuje, t. j. extrahuje, iba finálny embedding [CLS]. To sa potom odovzdá plne pripojenej vrstve, ktorá po aktivácii klasifikuje výstup pomocou sigmoidnej funkcie v našom kontexte viacerých značiek.



Obr. 8.8: Reprezentácia predtréningovej metódy pre model BERT s celou jeho štruktúrou ako [CLS] a [SEP] token, vrátane MLM a NSP [60].

Obrázok 8.8 sumarizuje predtréningový postup so všetkými jeho hlavnými charakteristikami. Vo svojej pôvodnej verzii je BERT trénovaný na BooksCorpus (asi 800 miliónov slov) a anglickej Wikipédii (asi 2500 miliónov slov). Trénovací proces zabral približne 40 epoch.

V čase, keď bol tento model predstavený, BERT prekonal všetky už existujúce LM v rôznych úlohách a stal sa de facto najmodernejším modelom NLP. Viac podrobností o modeli BERT možno nájsť v práci [60], ktorú publikovali Devlin a kolektív v roku (2018).

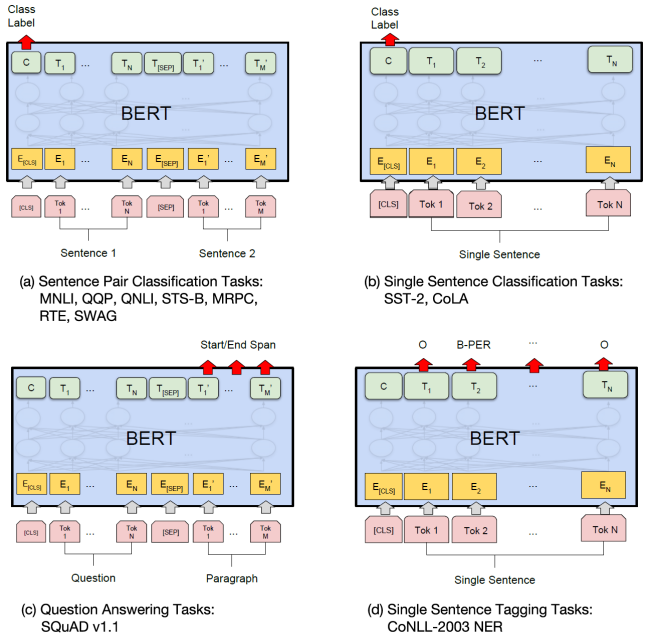
8.1.6 Fine-tuning

Fine-tuning je proces prispôsobenia predtrénovaného modelu BERT na inú úlohu, ako na ktorú bol predtrénovaný.

Prvým krokom je úprava vstupu úlohy tak, aby vyhovovala vstupom modelu BERT a trochu pripomínala predtréningové úlohy. To je možné uskutočniť v mnohých úlohách, pretože BERT podporuje viacero vstupných sekvencií. Avšak pre svoju obojsmernú povahu a pevnú vstupnú dĺžku sekvencií nie je BERT vhodný na úlohy generovania textu (zatiaľ čo GPT je [62]).

Druhým krokom je pridanie ďalšej vrstvy (vrstiev) na koniec posledného bloku. Tieto novo pridané vrstvy by mali byť schopné odpovedať na novo zadanú úlohu.

Posledným krokom je tréning takto upraveného modelu BERT na nových dátach, ktoré sú špecifické pre novo zadanú úlohu. Počas tohto procesu sa trénujú všetky váhy modelu. Príklady možných úloh, ktoré sa takýmto spôsobom dajú riešiť, zahŕňajú všetky úlohy GLUE [65], klasifikáciu textu, sekvenčné označovanie, parafrázovanie, zodpovedanie otázok a mnohé ďalšie.



Obr. 8.9: Fine-tuning modelu BERT na oboch úlohách. [60]

Proces fine-tuningu je oveľa rýchlejší ako tréning, Devlin a kol. uvádzajú až 300-násobné zrýchlenie. Svoj experiment porovnávali na 16 TPU počas 4 dní versus 1 TPU po dobu 1 hodiny [60].

8.2 Modifikácie modelu BERT

BERT má veľa rôznych variantov a v tejto časti spomenieme tie najpodstatnejšie, ktoré v tejto práci budú implementované a porovnané v rade experimentov, vzhľadom na ich úspešnosť pri detekcii falošných správ v slovenskom datasete. Modifikácie modelu BERT, založené na architektúre Transformer predstavené v tejto podkapitole, dosiahli vynikajúce výsledky v rôznych úlohách spracovania prirodzeného jazyka (NLP), vrátane klasifikácie textu, rozpoznávania pomenovaných entít, strojového prekladu, analýzy sentimentu a zodpovedania otázok. Sú široko prijímané a slúžia ako základ pre mnohé aplikácie a výskumné pokroky v oblasti NLP.

8.2.1 mBERT

Viacjazyčný mBERT je rozšírením pôvodného modelu BERT. V súčasnosti sú dostupné dve verzie, prvá je *cased*, ktorá má WordPieces vrátane veľkých a malých písmen. Druhá verzia modelu je *uncased*, ktorá má všetky podslová písané malými písmenami a bez diakritiky.

Kľúčovou myšlienkou mBERT-u je vytvorenie jedného modelu, ktorý dokáže pracovať s viacerými jazykmi. To ho odlišuje od tradičných modelov špecifických pre jeden jazyk, mBERT je trénovaný súčasne na rôznych jazykoch, čo mu umožňuje zachytiť spoločné jazykové vzory naprieč rôznymi jazykmi. Vo vete mBERT chápe, že slovo *„dog“* a slovo *„pes“* majú rovnaký význam bez toho, aby v príkladoch v tréningu videl konkrétny príklad alebo informácie na ich rozoznanie/spojenie. Táto reprezentácia viacjazyčného jazyka v rovnakom zdieľanom priestore nám umožňuje vykonávať takzvaný medzijazyčný prenos. To z neho robí cenný nástroj pre viacjazyčné spracovanie prirodzeného jazyka (NLP).

Prekvapivé je, že aj pre jazyky, ktoré majú len malé percento trénovacích príkladov, to v skutočnosti funguje veľmi dobre. Pre objasnenie, len jedno percento tréningových príkladov je vo viacjazyčnom BERT-e pre slovenský jazyk.

mBERT je postavený na rovnakej architektúre ako BERT, ktorá pozostáva z viacvrstvového bidirekcionálneho kódovacieho transformeru. Pretrénovanie modelu mBERT prebehlo rovnakým spôsobom ako pri modeli BERT z kapitoly 8.1, z ktorého vychádza. Avšak na rozdiel od BERT-u, ktorý bol trénovaný na anglickom korpuse, je mBERT trénovaný na textoch z 104 rôznych jazykov. Tieto rôznorodé trénovacie dáta umožnili mBERT-u naučiť sa reprezentácie cez jazyky a prenášať znalosti medzi jazykmi.

Schopnosť prenášať znalosti medzi jazykmi je jednou z hlavných výhod. Táto metóda prenosu učenia sa ukazuje ako prospešná v prípadoch, keď je k dispozícii iba obmedzené množstvo anotovaných dát v cieľovom jazyku. Využitím predtrénovaného modelu mBERT-u môžu výskumníci a vývojári iba jemnými upravami prispôsobiť model na špecifické NLP úlohy v konkrétnom jazyku bez nutnosti tréningu celého modelu v danom jazyku.

8.2.2 RoBERTa

Tento model neurónovej siete založený na architektúre Transformer optimalizuje a zrobustňuje pôvodný model BERT (Bidirectional Encoder Representations from Transformers). Prvýkrát bol predstavený v roku 2019 (Liu a kolektív) ako

modifikácia architektúry BERT s cieľom riešiť niektoré obmedzenia tohto modelu a zlepšiť jeho výkon [66]. Model RoBERTa je založený na rovnakej architektúre ako BERT z kapitoly 8.1 s drobnými rozdielmi, ktoré majú viesť k zlepšeniu výkonu modelu. Hlavné rozdiely medzi modelmi BERT a RoBERTa spočívajú v cieľoch predtrénovania a použitých technikách. Pri tréningu modelu RoBERTa sa využil väčší korpus trénovacích dát a tréning modelu prebiehal dlhšie v porovnaní s pôvodným modelom BERT.

Hlavné rozdiely medzi RoBERTa a BERT:

- Dĺžka viet: BERT zahadzuje dlhšie vety počas predtrénovania, zatiaľ čo RoBERTa používa metódu posúvania okna, aby zvládla dlhšie textové sekvencie, čo vedie k lepšej reprezentácii dlhých textov.
- Dĺžka trvania: Tréning modelu RoBERTa bol dlhší a s väčším počtom iterácií, čo vedie k hlbšiemu a generalizovanejšiemu reprezentovaniu jazyka.
- Trénovacie dáta: RoBERTa je predtrénovaná na oveľa väčšom korpuse v porovnaní s modelom BERT, vrátane dátových sád ako *Books1*, *Books2*, *CC-News*, *OpenWebText* a ďalších.
- Ciele predtrénovania: Pri tréningu modelu RoBERTa bola použitá iba metóda Masked Language Modeling (MLM). Cieľom MLM je náhodne zakrývať niektoré tokeny vstupného textu a trénovať model na predikciu zakrytých tokenov.

  Na rozdiel od modelu BERT pri tréningu nebola použitá metóda NSP popísaná v sekcii 8.1, ktorá bola v pôvodnom modeli BERT použitá ako dôležitá súčasť tréningu [60]. Neskoršie experimenty autorov modelu RoBERTa ukázali, že odstránenie tejto úlohy z trénovania modelu môže viesť k zlepšeniu výsledkov. Preto nebola použitá v návrhu modelu RoBERTa.

8.2.3 XLM-RoBERTa

XLM-Roberta je viacjazyčnou verziou modelu RoBERTa. Rovnako ako v modeli mBERT je kľúčová myšlienka vytvoriť jeden model, ktorý dokáže pracovať s viacerými jazykmi. Vytvorenie tohto modelu a proces jeho tvorby je motivovaný modelom mBERT s cieľom postaviť robustenejšiu a väčšiu verziu tohto modelu. Tento model je postavený na rovnakej architektúre a tréningu ako RoBERTa, ktorá pozostáva z viacvrstvového bidirekcionálneho kódovacieho transformeru s tým rozdielom, že je predtrénovaná na jazykovom korpuse, ktorý tvorí 100 rôznych jazykov.

8.2.4 SlovakBERT

Tento model vytvoril KInIT v spolupráci s Gerulata Technologies s cieľom zlepšiť automatické spracovanie slovenských textov. SlovakBERT je založený na architektúre Transformer, konkrétne na type modelu RoBERTa a bol trénovaný na veľkom množstve dát v slovenskom jazyku [67].

SlovakBERT je prvým takýmto modelom určeným pre slovenský jazyk. Tento model bol vytvorený koncom roka 2021, a publikovaný v článku *„SlovakBERT:*

Slovak Masked Language Model“ [67]. SlovakBERT analyzujeme podrobnejšie vzhľadom na jeho výhody, ktoré vyplývajú z jeho príslušnosti k slovenskému jazyku.

Architektúra modelu

Architektúra je založená na modeli RoBERTa [66], ktorá je už v základe robustnejšia ako model BERT. V tabuľke 8.1 je uvedený detailný popis tejto architektúry. Ako tokenizér bol použitý BPE [68] s veľkosťou abecedy 50264.

SlovakBERT	
Architecture	RoBERTa
Num. layers	12
Num. attention head	12
Hidden size	768
Num. parameters	110M
Languages	1
Training dataset size (tokens)	4.6B
Slovak dataset size (tokens)	4.6B
Vocabulary size	50K
Universal Dependencies train set tokenization	
Average token length (chars)	3.23
Average word length (tokens)	1.43
Effective vocabulary	16.6K
Effective vocabulary (%)	33.05

Tabuľka 8.1: Architektúra modelu SlovakBERT ako bola uverejnená v publikácií *„SlovakBERT: Slovak Masked Language Model“* [67].

Tréning modelu

Učenie modelu prebiehalo na hardvéri tvorenom 4 NVIDIA A100 GPUs, a zabralo aproximovane 248 hodín. Model bol trénovaný približne 300000 trénovacích krokov s batch size 512, čo je približne 70 epoch. Každá epocha pozostáva z 4277 trénovacích krokov. Ako optimalizér bol použitý optimalizačný algoritmus *Adam* s hodnotou učiaceho koeficientu 5×10^{-4}. Regularizácia učenia bola docielená Dropout = 0.1 a weight decay ($\lambda = 0.01$). Dĺžka vstupných sekvencií bola limitovaná na 512 tokenov so snahou zachovať čo najviac sekvencií v plnom formáte.

Trénovanie modelu SlovakBERT nebolo jednoduchou úlohou. Samotný tréning vyžadoval takmer dva týždne výpočtov na serveri s výkonným hardvérom. Ak by sme to porovnali s bežným počítačom s grafickou kartou strednej triedy, trénovanie by trvalo roky, s bežným pracovným notebookom dokonca aj desaťročia.

Trénovacie dáta

Ako trénovací súbor dát bola použitá kombinácia dostupných dátových súborov v slovenskom jazyku a dataset získaný web-crawlingom slovenských webových stránok. Dostupné súbory dát v slovenskom jazyku, ktoré boli použité pre tréning, sú: *Wikipedia* (326 MB textu), *Open Subtitles* (415 MB) a *OSCAR 2019 dataset* (4,6 GB).

Tieto dátové sady boli rozšírené o dáta, ktoré boli získané prehľadaním webových stránok s top doménou *.sk*. Na jednotlivé webové stránky bola aplikovaná detekcia jazyka a boli z nej extrahované iba názov a čistý textový obsah bez HTML značiek. Veľkosť takto získaných dát bola 17,4 GB. Text bol potom spracovaný nasledujúcimi krokmi:

- Adresy URL a e-mailové adresy boli nahradené špeciálnymi symbolmi.
- Opakujúce sa interpunkčné znamienka boli redukované, t.j. ak existovali sekvencie tvorené rovnakým interpunkčným znamienkom, potom boli zredukované iba na jedno znamienko (napr. !!!! na !).
- Markdown syntax bola odstránená.
- Nealfanumerické znaky boli odstránené.

Výsledný dátový súbor bol rozdelený na vety a boli z neho odstránené duplicity. Finálny dataset tvorí 181,6 milióna unikátnych viet. Tieto dáta tvoria obraz toho, čo model považuje za slovenčinu. Celkovo má konečný dátový súbor veľkosť 19,35 GB.

Evaluácia modelu

Výsledný model bol otestovaný výskumníkmi z KInIT na rozličných úlohách z oblasti spracovania prirodzeného jazyka. SlovakBERT dosahuje výborné výsledky v gramatickej analýze, sémantickej analýze, rozpoznávaní sentimentov či klasifikácii dokumentov.

Výsledky jednotlivých experimentov sú dostupné vo verejne dostupnom článku „SlovakBERT: Slovak Masked Language Model“ [67]. V súčasnosti je SlovakBERT jedným z najlepších modelov pre spracovanie slovenského jazyka. Natrénovaný model je verejne dostupný, môže ho teda NLP komunita využiť na riešenie rôznych problémov spracovania prirodzeného jazyka v slovenčine.

9. Fake-news datasety

Získanie dostatočne veľkej dátovej sady pre klasifikáciu falošných správ a dezinformácií je veľmi náročný proces, aj v dnešnej dobe. Týmto náročným krokom získania dátovej sady to však iba začína, jej následné spracovanie a klasifikovanie je ešte komplikovanejšie.

V súčasnosti existuje iba obmedzené množstvo dátových sád, ktoré sú k dispozícii pre tento fenomén v porovnaní s ostatnými klasifikačnými problémami. To je spôsobené náročnosťou tvorby datasetov pokrývajúcich problém fake-news. Problém pri tvorbe týchto datasetov spočíva v tom, že každý článok v dátovej sade je nutné manuálne overiť a vyhodnotiť jeho pravdivostnú hodnotu. Jedným z dôvodov takejto zdĺhavej kontroly je absencia dostatočne kvalitných automatických systémov na detekciu falošných správ, ktoré by mohli byť použité na prvotnú automatickú anotáciu.

Každý text anotuje viac anotátorov, aby sa zabránilo chybovosti a zaujatosti jednotlivca. Tento postup, ktorým sa vytvárajú datasety zaoberajúce sa témou fake-news, je náročný na čas aj ľudské zdroje (anotátorov). S tým je spojený ďalší problém: pre kvalitnú anotáciu je potrebné väčšie množstvo anotátorov, čo sa týka najmä menej rozšírených jazykov, kde týchto anotátorov môže byť nedostatok. Príkladom je aj slovenský jazyk, ktorý má jedného oficiálneho anotátora pre obsah príspevkov na Facebooku (feed-checker).

Z týchto dôvodov sa väčšina prác zaoberajúcich sa fake-news venovala jazykom s väčším jazykovým korpusom, teda s dostatkom dát, ako sú angličtina, nemčina a francúzština. Pre slovenský jazyk sa nám podarilo získať momentálne jednu z prvých neverejných dátových sád, na ktorej experimentovali Sarnovský s kolektívom [2]. Vzhľadom na nedostatok kvalitných dát v slovenskom jazyku sa pokúsime v experimentálnej časti tejto práce využiť aj cudzojazyčné datasety pokrývajúce tému fake-news, s cieľom transformovať metódy úspešné na jazykoch s dostatkom dát na slovenčinu.

V nasledujúcich podkapitolách predstavíme dátové sady, ktoré boli použité v tejto práci. Zameriavame sa na ich pôvod, jazyk, základné štatistiky a vlastnosti jednotlivých textov. Datasety sú rozdelené do dvoch skupín: anglickej a slovenskej podľa jazyka, v ktorom sú texty v dátových sadách.

9.1 Anglické dátové sady

Pre anglický jazyk existuje niekoľko dátových sád, pre naše účely sme vybrali dve z nich. Prvým je dataset s názvom LIAR, ktorý je známy v NLP komunite. Ako druhý sme zvoli COVID19 FN. Dôvodom voľby tohto datasetu je charakter slovenskej dátovej sady, ktorú sa nám podarilo získať. Slovenská dátová sada je špecificky zameraná na články covid-19 fake-news. Chceme teda porovnať aj vplyv témy článkov a úspešnosť transferu medzi jazykmi. Z tohto dôvodu boli zvolené práve tieto dva datasety.

9.1.1 LIAR

LIAR je dátová sada pozostávajúca z výrokov a novinových článkov, ktoré sú anotované. Dátová sada bola vytvorená výskumníkmi Kalifornskej univerzity v Berkeley a má slúžiť na tréning a hodnotenie modelov, ktoré dokážu rozpoznať falošné správy [6]. V dobe vytvorenia bol najväčším datasetom svojho druhu.

Túto dátovú sadu sme si zvolili pre jej značnú popularitu v zahraničnej komunite NLP pri detekcii fake-news. Hlavným dôvodom však je téma, ktorou sa zaoberajú jednotlivé texty. LIAR pokrýva najmä politické témy, ktoré súviseli s voľbami v USA v roku 2016. Články v datasete LIAR majú diametrálne odlišnú tému ako tie obsiahnuté v slovenskom datasete (covid-19). Tento fakt nám dáva priestor na overenie hypotéz, že fake-news majú podobný formát v rôznych témach. Prípadne túto hypotézu pre slovenský jazyk vyvrátiť.

	Veľkosť
Trénovaca množina	10 269
Validačná množina	1284
Testovacia množina	1283
Avg. dĺžka textov (počet tokenov)	17,9

Tabuľka 9.1: LIAR štatistiky dátovej sady [6].

Dátová sada obsahuje celkovo 12,8 tisíc manuálne anotovaných novinových článkov z rôznych zdrojov, vrátane politických blogov, stránok overujúcich fakty a hlavných zdrojov správ. Podrobná správa a analýza jednotlivých článkov bola vykonaná pomocou *politifact.com*, kde každý článok analyzovali a overili jeho zdroj. Následne v druhom kroku bol obsah článku analyzovaný editorom, ktorý vyhodnotil pravdivostnú hodnotu obsahu. Každý článok je označený jednou zo šiestich kategórií: *pants-fire*, *false*, *barely-true*, *half-true*, *mostly-true* a *true*.

- *true*: Tvrdenie je pravdivé (presné) a nič významné v ňom nechýba.
- *mostly-true*: Tvrdenie je pravdivé, ale potrebuje dodatočné objasnenie alebo ďalšie informácie.
- *half-true*: Tvrdenie je čiastočne pravdivé, ale vynecháva dôležité detaily alebo veci vytrháva z kontextu.
- *barely-true*: Tvrdenie obsahuje prvok pravdy, ale ignoruje kritické skutočnosti, ktoré by dali na obsah iný pohľad.
- *false*: Tvrdenie je nepravdivé.
- *pants-fire*: Tvrdenie nie je pravdivé a obsahuje absurdný záver. V článku označené ako: *„Liar, Liar, Pants on Fire!“*

Dataset vo formáte, ako bol zverejnený, je rozdelený do troch disjunktných množín, trénovacej, validačnej a testovacej množiny. Obsahuje tiež ďalšie metadáta pre každý článok, ako je zdroj a dátum publikovania. V tabuľkách 9.1 a 9.2 je možne vidieť distribúciu príkladov do jednotlivých množín.

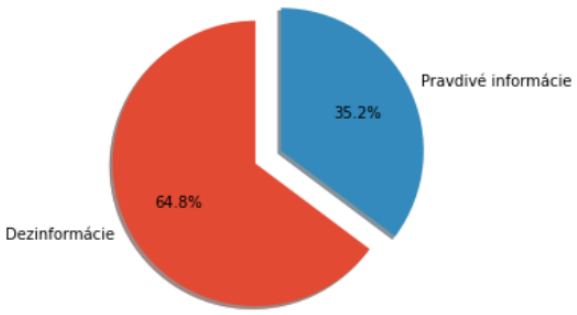
Autori článkov	Počet
Demokrati	4150
Republikáni	5687
FB príspevky	2185

Tabuľka 9.2: Tri najpočetnejšie skupiny autorov v dátovej sade LIAR, ktoré boli uverejnené v publikácii *„Liar, Liar Pants on Fire“: A New Benchmark Dataset for Fake News Detection* [6].

Predspracovanie dátovej sady

Pre účely tejto práce je nutné upraviť pôvodnú dátovú sadu LIAR. Z charakteru klasifikačných tried v cieľovom slovenskom datasete je nevyhnutné upraviť klasifikačné triedy aj v datasete LIAR. Z pôvodných 6 tried dôjde k redukcii na binárny príznak, kde 1 značí falošnú správu a 0 pravdivú. Pôvodné príznaky tried sme nanovo rozložili nasledovne: klasifikačné triedy *„mostly-true“* a *„true“* sme spojili do novej triedy, ktorá charakterizuje pravdivú množinu s príznakom 0. Ostatné klasifikačné triedy z pôvodného datasetu sú v novej dátovej sade chápané ako nepravdivé t.j. 1.

V dátovej sade došlo ešte k jednej zmene, pôvodné rozloženie článkov do troch množín train, valid a test sme zmenili. V novej dátovej sade train a valid tvoria jednu trénovaciu množinu a pôvodná testmnožina sa používa na validáciu. Tento krok sme vykonali z dôvodu zväčšenia trénovacej množiny, vzhľadom na to, že evaluácia sa bude vzťahovať na slovenský dataset. Po tejto úprave je rozdelenie do tried 64,8% nepravdivých správ a 35,2% pravdivých, ako je možné vidieť na obrázku 9.1.

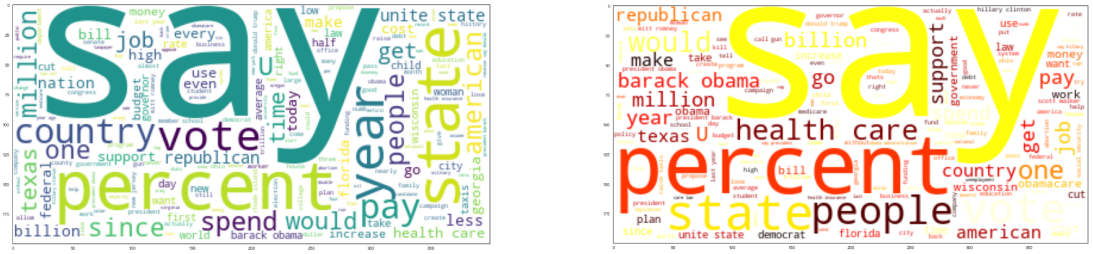


Obr. 9.1: Distribúcia do tried upraveného LIAR.

Vizualizácia dátovej sady

V tejto časti sú vizualizácie LIAR datasetu pre oboznámenie sa s charakterom dát, ktoré ho tvoria. Na nasledujúcich vizualizáciách sú zobrazené word clouds pre jednotlivé cieľové množiny. Pri tomto zobrazení veľkosť slova odzrkadľuje jeho

význam v jednotlivých textoch, t.j. veľkosť tokenov odzrkadľuje ich frekvenciu v dátach.



(a) Pravdivé články

(b) Falošné články

Obr. 9.2: LIAR: word cloud pre pravdivé a falošné články.

Počas vizualizácie sme vykonali aj množstvo testov zaoberajúcich sa frekvenciou slov v jednotlivých množinách a dospeli sme k záveru, že slovná zásoba pravdivých a falošných článkov v dátovej sade LIAR obsahuje minimálne rozdiely. Vykonanie týchto testov neprinieslo žiadnu zásadnú informáciu oproti vizualizáciám word cloud na obrázku 9.2. Tieto testy len potvrdili frekvencie slov, ako sme ich vizualizovali.

V týchto textoch dominujú slovné spojenia z oblasti volieb, politiky a súvisiacich problémov, ktoré je možné vidieť na vizualizácii 9.2. Výsledok je očakávaný, pretože LIAR dataset sa zameriava práve na túto oblasť falošných správ.

9.1.2 COVID19 FN

COVID19 FN je ďalšou dátovou sadou, ktorou sa budeme zaoberať v tejto práci [3]. Tento dataset bol zvolený z dôvodu rovnakej témy článkov ako má slovenský dataset, t.j. články spojené s pandémiou covidu-19. V experimentálnej časti tejto práce porovnávame výsledky s modelmi natrénovanými na datasete LIAR, ktorý sme predstavili v sekcii 9.1.1. Od tohto porovnania očakávame potvrdenie alebo vyvrátenie hypotézy o dôležitosti témy článkov pri detekcii fake-news. COVID19 FN je dataset obsahujúci správy zamerané na ochorenie covid-19, ktorý pôvodne vytvorili Patwa a kolektív [3]. Na overenie a kontrolu faktov použili podobné stránky ako tvorcovia datasetu LIAR, vrátane *Politifact.com*, *NewsChecker*, *Boomlive* a z nástrojov napríklad *Google fact-check-explorer* a *IFCN chatbot*. Táto dátová sada pozostáva z tweetov, článkov a príspevkov na sociálnych sieťach. Pri tvorbe datasetu najskôr identifikovali správy súvisiace s pandémiou covid-19. Následne manuálne detegovali jazyk správ, do datasetu boli zahrnuté iba správy, ktoré boli písané v anglickom jazyku. Následne rozdelili manuálne správy na pravdivé a falošné.

Pravdivé správy boli zozbierané z overených zdrojov, ktoré sa zaoberali ochorením covid-19: stránky a účty vládnych organizácií, zdravotnícke organizácie a noviny. Konkrétne boli zozbierané zo 14 rôznych zdrojov, ako sú *World Health Organization (WHO)*, *Centers for Disease Control and Prevention (CDC)*, *Covid India Seva*, *IndianCouncil of Medical Research (ICMR)* a ďalšie. Každá správa bola manuálne anotovaná človekom, ktorý určil jej pravdivosť a relevantnosť k téme covid-19. Falošné správy sú zozbierané zo špekulatívnych zdrojov,

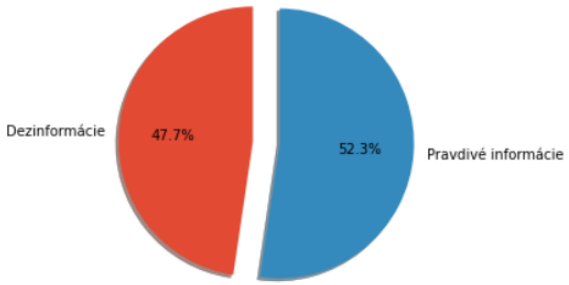
ktoré boli verifikované ako fake-news. Rovnako ako pravdivé správy bolo manuálne overené, či daná správa súvisí s covidom-19. Anotácia pravdivostnej hodnoty prebehla manuálne, t.j. hodnotiteľom bol človek.

Dátová sada vo forme, ako bola publikovaná, je rozdelená do troch disjunktných množín a to trénovacej, validačnej a testovacej. Distribúcia správ do jednotlivých množín je zobrazená v tabuľke 9.3.

	Veľkosť
Trénovacia množina	6420
Validačná množina	2140
Testovacia množina	2140
Avg. dĺžka textov (počet tokenov)	27,05

Tabuľka 9.3: COVID19 FN: štatistiky dátovej sady [3].

V tejto dátovej sade nie je potrebná úprava klasifikačných tried, pretože sa zhodujú s triedami v slovenskom datasete. Dataset je rozdelený do 3 podmnožín, ktoré sú vybalansované rovnako ako celý dataset. Tento fakt veľmi pomáha učeniu modelov, keď nie je nutné dodatočné vyvažovanie rozloženia tried.



Obr. 9.3: Distribúcia do tried COVID19 FN.

Predspracovanie dátovej sady

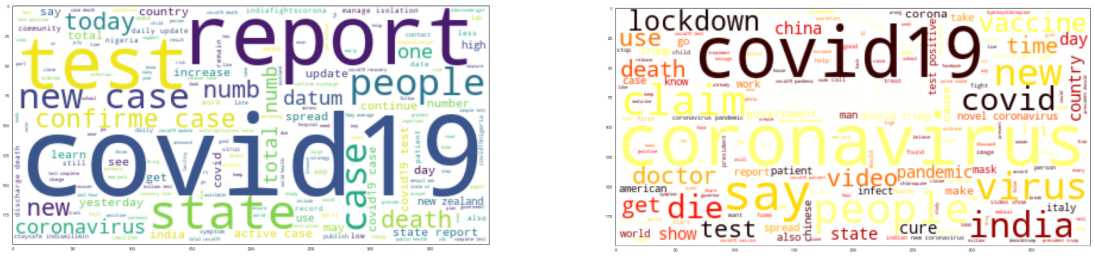
V dátovej sade sme rovnako ako pri LIAR datasete zmenili pôvodné rozloženie článkov z troch množín na dve. V novej dátovej sade train a valid tvoria jednu trénovaciu množinu a pôvodná testovacia množina sa používa na validáciu. Tento krok sme vykonali z rovnakých dôvodov ako v dátovej sade LIAR.

Dataset pozostáva primárne z príspevkov na sociálnych sieťach, obsahuje aj nealfanumerické znaky, ako sú napríklad rôzne emotikony a sprievodné znaky. Tieto znaky bolo nutné zo správ vymazať z dôvodu, že veľké lingvistické modely s nimi nevedia pracovať a iba by to prinieslo šum do dát.

Vizualizácia dátovej sady

V tejto časti sú vizualizácie dátovej sady COVID19 FN pre oboznámenie sa s charakterom dát, ktoré ho tvoria. Analyzovali sme tiež dataset na úrovni tokenov, pričom 10 najčastejších tokenov po odstránení stopwords je:

- Falošné správy: coronavirus, covid19, people, will, new, trump, says, video, vaccine, virus
- Pravdivé správy: covid19, cases, new, tests, number, total, people, reported, confirmed, states
- Kombinované správy: covid19, cases, coronavirus, new, people, tests, number, will, deaths, total



(a) Pravdivé články

(b) Falošné články

Obr. 9.4: COVID19 FN: word cloud pre pravdivé a falošné články.

Ilustrácia 9.4 vizualizuje frekvenciu slov pre jednotlivé klasifikačné triedy pomocou metódy word clouds. Z word cloudov a najfrekventovanejších slov môžeme vidieť, že existuje významné prekrytie dôležitých slov medzi falošnými a reálnymi správami. Tento jav je spôsobený hlavne tým, že obe spadajú do jednej témy pandémie covidu-19.

Ak sa bližšie pozrieme na výpis najfrekventovanejších slov v jednotlivých podmnožinách, môžeme si všimnúť jemné odchýlky. V prípade pravdivých článkov je skladba najfrekventovanejších slov zameraná na štatistiky a dáta súvisiace s ochorením covid-19, testovaním obyvateľstva a počtom potvrdených prípadov v jednotlivých štátoch. Tieto články charakterizuje poskytovanie všeobecných informácií o priebehu pandémie covidu-19 bez konkrétneho zamerania na určité témy. Falošné správy sa zameriavajú na konkrétne témy a udalosti súvisiace s pandémiou covidu-19, napríklad výroky Donalda Trumpa, dezinformácie o vakcínach a mnoho ďalších, ktoré sú podložené odkazmi na tvrdenia alebo videozáznamy. Tieto dodatočné odkazy sú nami odvodené z vyššieho výskytu slov „say“, „video“, ktoré sa vo falošných článkoch vyskytujú častejšie ako informácie o počte nakazených.

Na základe týchto zistení sa ukazuje, že falošné a pravdivé správy sa odlišujú aj v slovnej zásobe, ktorá ich tvorí. Ale je dôležité si uvedomiť, že tieto výsledky sú založené na určitom špecifickom datasete. Frekvencia slov môže byť ovplyvnená rôznymi faktormi, ako je zameranie zdrojov, časové obdobie a použité metódy spracovania dát. Pre presnejšie a komplexnejšie vyhodnotenie by bolo potrebné analyzovať väčší súbor dát a zohľadniť aj kontext a sémantiku viet a správ.

9.2 Slovenské dátové sady

Verejné dátové sady určené pre úlohu klasifikácie fake-news v slovenskom jazyku momentálne nie sú dostupné. Existuje iba malé množstvo dátových sád

a k tomu zatiaľ žiaden repozitár nie je verejne dostupný. Z tohto dôvodu sa tejto problematike venuje minimum vedcov z oblasti NLP na Slovensku. Tento stav nie je spôsobený absenciou fake-news na Slovensku, ale problémom s anotáciou textov potrebných pre vytvorenie datasetu.

Nám sa v tejto práci podarilo získať dataset vytvorený Sarnovským a kolektívom [2], ktorý nie je aktuálne verejne dostupný. Za poskytnutie tohto datasetu sme mu veľmi vďačný, položil tým základný kameň celej tejto práce. Na základe dát z tohto datasetu sme vytvorili v tejto práci nový menší dataset, ktorý je popísaný v sekcii 9.2.2.

Tvorcovia majú v pláne v budúcnosti publikovať celý dataset, čo by mohlo rozbehnúť automatickú klasifikáciu fake-news v slovenskom jazyku.

9.2.1 SlovakCovid19 FN

Uvedená dátová sada je prvým slovenským datasetom, ktorý v tejto práci predstavíme. SlovakCovid19 FN vytvoril Sarnovský a kolektív na Technickej univerzite v Košiciach [2]. Táto dátová sada nie je verejne dostupná a je nutné o ňu požiadať. Nám bola poskytnutá dátová sada v už rozšírenej verzii od vydania publikácie, ale v neočistenej forme, aby sme na nej mohli vykonať vlastné čistenie, keďže tím z TU v Košiciach má v pláne svoj dataset publikovať v samostatnom článku zameranom na predspracovanie a čistenie tejto dátovej sady.

SlovakCovid19 FN je dataset, ktorý je jedným z prvých slovenských datasetov pre úlohu klasifikácie fake-news. Obsahuje správy týkajúce sa ochorenie covid-19 zozbierané z rôznych slovenských webov. Overenie pravdivostnej hodnoty článkov najskôr prebehlo automaticky, keď boli jednotlivé zdrojové weby ohodnotené podľa relevantnosti na *Konspiratori.sk*. Na základe tohto ohodnotenia boli články z jednotlivých webov rozdelené do skupín na pravdivé a nepravdivé. Táto pôvodná myšlienka sa ale ukázala ako nesprávna. Pretože aj stránky, ktoré šíria dezinformácie, občas zverejňujú pravdivú správu. Táto skutočnosť priniesla do dát množstvo nesprávne ohodnotených článkov.

Z tohto dôvodu bol zavedený druhý stupeň automatického overenia pravdivostnej hodnoty článkov, ktorý sa zameriaval na určenie pôvodného zdroja správy. Táto metóda v prvom kroku overila, či sa v správe nachádza reťazec zdrojov, z ktorého publikovaná správa čerpala. Je bežnou praxou, že sa v správach často nachádzajú informácie o zdroji, z ktorého bola správa čerpaná, napríklad *SITA*, *AP*, *HSP*, *RusVesna* a iné. Na základe týchto informácií sa pridelilo ďalšie dodatočné skóre zohľadňujúce pravdepodobnosť šírenia falošných správ zo zdrojových médií. Týmito dvoma automatickými metódami boli dáta ohodnotené v prvom kroku.

Paralelne s týmto postupom sa jednotlivé články hodnotili aj manuálne prostredníctvom 5 anotátorov. Týmito anotátormi boli pre potreby ohodnotenia dátovej sady študenti na Technickej univerzite v Košiciach, ktorí sa podieľali na tvorbe dátovej sady. Jednotlivé články hodnotili na základe pravdivosti obsahu textov, ktoré dané články tvorili. Všetky články v dátovej sade boli ohodnotené každým anotátorom z dôvodu zvýšenia presnosti ohodnotenia pravdivostnej hodnoty. Výsledná anotácia datasetu bola vytvorená spojením manuálnej a automatickej anotácie.

Dataset je tvorený pravdivostnou hodnotou a obsahom článku. Obsah jednot-

livých článkov v dátovej sade tak ako nám bola poskytnutá netvoria čisté textové dáta, ale podľa zdroja aj rôzne druhy šumu. Medzi ne môžeme zaradiť HTML tagy, URL odkazy na multimediálny obsah, záhlavia stránok, metadáta webu atď. Tento šum však nie je pre všetky články v dátovej sade konzistentný. Z tohto dôvodu tieto dodatočné informácie (metadáta) nie je možné použiť pri výslednej klasifikácii falošných správ.

Aby sme mohli tento dataset použiť pre klasifikáciu pomocou NLP, bolo nutné ho od týchto dodatočných informácii očistiť.

Predspracovanie dátovej sady

Vzhľadom na to, že sme dostali dátovú sadu v jej pôvodnej forme, ktorá obsahovala veľké množstvo šumu a dáta z webových stránok po fáze dolovania dát (data mining), bolo nevyhnutné najprv vyčistiť tento dataset od netextových dát, ktoré by mohli ovplyvniť klasifikáciu na základe textových údajov. Aby sme mohli porovnať naše výsledky s prácou Sarnovského, v prvom kroku sme z dátovej sady odstránili metadáta, ktoré boli zmienené v ich práci ako tie, ktoré odstránili. Naším zámerom bolo zvoliť základnú metódu čistenia tak, aby sme sa priblížili k ich experimentu a mohli sme následne porovnať ich výsledky. Informácie, ktoré Sarnovský a kolektív odstránili a uviedli ich v publikácii, sú nasledovné [2]:

- Adresy URL hypertextové odkazy a e-mailové adresy boli odstránené.
- Opakujúce sa interpunkčné znamienka boli redukované, t.j. ak existovali sekvencie tvorené rovnakým interpunkčným znamienkom, potom boli zredukované iba na jedno znamienko (napr. !!!! na !).
- Markdown syntax bola odstránená.
- Záhlavia článkov.
- Nealfanumerické znaky boli odstránené.

Počas predspracovania textových dát sme však zistili, že niektoré články obsahovali zaujímavé dodatočné informácie, ktoré zostali v tele článkov aj po fáze čistenia. Tieto textové metadáta, ktoré boli stiahnuté spolu s obsahom článkov, považujeme za nežiaduce, pretože pridávajú šum a skresľujú priestor dátového súboru. Tieto dodatočné informácie by mohli viesť k zaujatým rozhodnutiam založeným práve na týchto metadátach a nie na skutočnom obsahu článkov. Príkladmi takýchto dodatočných informácií, ktoré sa vyskytovali iba v podmnožine článkov, sú:

„Ak vás článok obohatil o ďalší uhol pohľadu, podporte ľubovoľnou čiastkou slobodu slova. Ďakujeme. CHCEM PODPORIŤ“

„Vyhlásenie: Názory autora sa nemusia zhodovať s názormi vydavateľstva Sofian, s.r.o. Zodpovednosť za obsah tohto článku nesie výhradne jeho autor. Vydavateľstvo Sofian, s.r.o. nie je zodpovedné za akékoľvek prípadné nepresné či nesprávne informácie v tomto článku. Sofian, s.r.o. dáva súhlas na zdieľanie našich pôvodných článkov na ďalších nekomerčných internetových stránkach, ak nebude

zmenený ich text a názov. Pri zdieľanom článku musí byť uverejnený zdroj a autor. Ak chcete články z nášho webu publikovať v tlači či inými formami, vrátane komerčných internetových stránok, kontaktujte redakciu na zemavek@zemavek.sk.“

„UPOZORNENIE Vážení čitatelia – diskutéri. Podľa zákonov Slovenskej republiky sme povinní na požiadanie orgánov činných v trestnom konaní poskytnúť im všetky informácie zozbierané o vás systémom (IP adresu, mail, vaše príspevky atď.) Prosíme vás preto, aby ste do diskusie na našej stránke nevkladali také komentáre, ktoré by mohli naplniť skutkovú podstatu niektorého trestného činu uvedeného v Trestnom zákone. Najmä, aby ste nezverejňovali príspevky rasistické, podnecujúce k násiliu alebo nenávisti na základe pohlavia, rasy, farby pleti, jazyka, viery a náboženstva, politického či iného zmýšľania, národného alebo sociálneho pôvodu, príslušnosti k národnosti alebo k etnickej skupine a podobne.“

Po bližšom preskúmaní týchto dodatočných informácii sme dospeli k zisteniu, že všetky tieto články pochádzajú z jedného zdroja *Hlavné správy*, ktorý totožné texty (ako sú uvedené v príkladoch vyššie) obsahuje aj v dnešnej dobe na svojich webových stránkach. Väčšina článkov z toho zdroja v sebe obsahovala na konci text, ktorým vyzývali na finančnú podporu tohto „nezávislého“ média v boji proti vláde a úradom. Tento text dopĺňa informáciu o obsahu, ktorý dané médium šíri, no napriek tomu sme ho z datasetu odstránili. Pre *Hlavné správy* to nekončí iba pri tejto jednej zaujímavosti. Ďalšou bola informácia v päte webovej stránky, ktorá bola pri web crawlingu stiahnutá. V tejto časti webovej stránky sa zriekli všetkých následkov v prípade šírenia nimi publikovaných článkov a informovali, ako ich vláda a iné nezávislé médiá na Slovensku chcú „umlčať“.

Dôležité je zdôrazniť, že tieto dodatočné textové informácie, ktoré boli stiahnuté spolu s obsahom článkov, sa nachádzali iba v podmnožine článkov patriacich výhradne do triedy falošných správ. Tieto informácie boli prítomné asi v polovici článkov z tejto množiny. Vzhľadom na príslušnosť k triede falošných správ by klasifikátor, ak by sme tieto textové dáta neodstránili, bol náchylný klasifikovať tieto správy ako falošné na základe tejto časti textu a nie na základe ich skutočného obsahu. V pôvodnej práci Sarnovského a jeho kolektívu nie je uvedené, ako sa s týmito informáciami vysporiadali a či ich ponechali alebo odstránili z datasetu. My sme sa rozhodli odstrániť všetky tieto „metadáta“, keďže cieľom tejto práce je klasifikácia fake-news na základe textového obsahu článkov.

Poslednou zaujímavosťou, ktorá sa ukázala v dátach, bola informácia o autoroch niektorých konšpiračných teórií. Počet takýchto článkov bol v jednotkách, ale zaujímavé bolo, že všetky články podpísané týmto autorom boli fake-news. Informácie o autorovi boli vo väčšine prípadov vymazané manuálne z textových dát, aby sa výsledný NLP klasifikátor nezameriaval iba na tieto konkrétne časti textov, ale aby sa zameral na obsah daného článku.

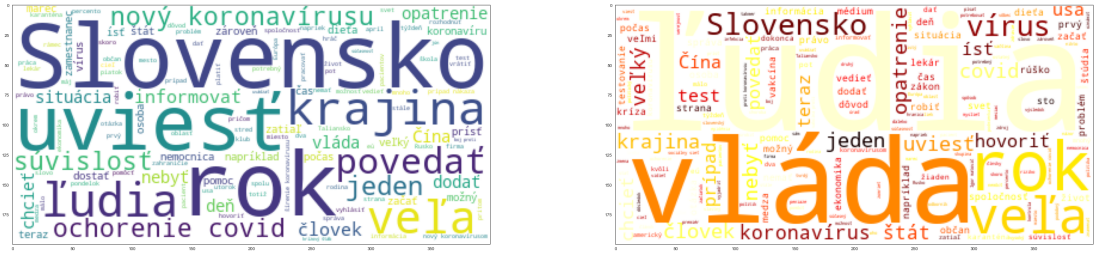
Pomocou týchto zistení by sa však dali skonštruovať ďalšie umelé metadáta k textovým a postupne vybudovať zložitejší dataset, ktorý by obsahoval textové dáta aj kontext k nim v podobe metadát. Následne by sa tento dataset dal použiť na natrénovanie ensemble systémov alebo iných kombinovaných klasifikátorov. Ale vzhľadom na malé množstvo dát s týmito informáciami to pre tento dataset nebolo použiteľné. Výsledkom tohto predspracovania bol slovenský dataset obsahujúci iba relevantné textové dáta.

Štatistiky	
Falošné články	851
Pravdivé články	12885

Tabuľka 9.4: SlovakCovid19 FN : štatistiky dátovej sady.

Vizualizácia dátovej sady

V tejto časti sú vizualizácie dátovej sady SlovakCovid19 FN pre oboznámenie sa s charakterom dát, ktoré ho tvoria.



(a) Pravdivé články

(b) Falošné články

Obr. 9.5: SlovakCovid19 FN FN: word cloud pre pravdivé a falošné články.

	Pravdivé	Falošné	Všetky
nový	12399	754	13153
koronavírusu	10806	808	11614
rok	9124	942	10066
opatrenie	8961	859	9820
ľudia	8590	1169	9759
prípad	8618	671	9289
vláda	7681	1234	8915
krajina	7720	855	8575
slovensko	7106	704	7810
situácia	6921	476	7397

Tabuľka 9.5: SlovakCovid19 FN : 10 najfrekventovanejších slov v dátovej sade.

Na základe ilustrácie 9.5 a tabuľky 9.5, ktoré zobrazujú frekvencie slov pre slovenský dataset, je možné pozorovať vzory vo výskyte slov pre jednotlivé triedy. Podrobnejšia analýza je uvedená v prílohe A.3 v podobe histogramov, ktoré poskytujú detailnejšie štatistiky. Rovnako ako v anglickej verzii existuje významný prekryv dôležitých slov medzi jednotlivými triedami. Možno vyčítať informácie o prebiehajúcej pandémii covidu-19, pre ktorú rezonujú výrazy v každom z obrázkov. Podobnosť celého datasetu k pravdivým článkom je spôsobená prevahou týchto dát v datasete.

Ak sa podrobnejšie pozrieme na zoznam najčastejších slov v jednotlivých skupinách, môžeme si všimnúť malé rozdiely. V prípade pravdivých článkov sa často

vyskytujú slová zamerané na štatistiky a dáta týkajúce sa ochorenia covidu-19, testovania populácie a počtu potvrdených prípadov v jednotlivých krajinách aj na Slovensku. Tieto dáta majú podobný informatívny charakter ako v predošlom anglickom datasete zaoberajúcom sa fake-news o covide-19. Na druhej strane sa falošné správy sústreďujú na konkrétne témy a udalosti súvisiace s pandémiou covidu-19, napríklad vládne opatrenia, dezinformácie o vakcínach a rúškach a ďalšie. V prípade falošných správ sa na popredné pozície dostávajú slová „vláda“ a „opatrenia“, ktoré prekonávajú aj samotné slovo „koronavírus“. Táto štatistika potvrdzuje výsledky word clouds, že falošné správy sa viac zameriavajú na politickú situáciu. V týchto článkoch viac rezonujú pojmy spojené s vládou, organizáciami a opatreniami. Akosi sa strácajú informácie o prebiehajúcej pandémii a viac sa články sústredia na štátne orgány a politiku.

Na základe týchto pozorovaní je zjavné, že falošné a pravdivé správy sa odlišujú aj vo svojej slovnej zásobe a oblasti záujmu. Je však dôležité uvedomiť si, že tieto výsledky sú založené na konkrétnom datasete. Pre presnejšie a komplexnejšie vyhodnotenie by bolo potrebné analyzovať väčší súbor dát a brať do úvahy aj kontext správ, v ktorom vznikli.

9.2.2 DownSampled SlovakCovid19 FN

K vytvoreniu tohto datasetu sme pristúpili potom, ako sme sa oboznámili s pôvodným datasetom SlovakCovid19 FN, ktorý nám poskytli Sarnovský a kolektív [2]. Nový dataset je vytvorený ako základná slovenská dátová sada pre klasifikáciu fake-news v téme pandémie covidu-19 pomocou metód NLP. Týmto datasetom nemáme cieľ nahradiť pôvodný dataset, práve naopak, má ho dopĺňať. Práca s ním je jednoduchšia, keďže je už očistený od šumu (HTML značky, záhlavia stránok, URL, odkazy na multimediálne dáta a mnohé iné). Táto korektúra zmenšuje časovú náročnosť predspracovania textu pri jeho budúcich využitiach.

Cieľom tohto datasetu je oboznámenie sa s problematikou fake-news na Slovensku. Reprezentuje úvodný krok pre prácu s pôvodným datasetom SlovakCovid19 FN, s ktorým je práca ťažšia, ale po získaní skúseností na nami vytvorenom datasete je možné informácie získané z neho uplatniť aj na riešenie problému v pôvodnej verzii datasetu. Poslednou dôležitou informáciou, ktorú musíme spomenúť, je, že dataset sme vytvorili vybalansovaný z dôvodu lepšieho porovnávania s vybalansovanými cudzojazyčnými datasetmi, ktoré sú verejne dostupné. Na nami vytvorenom datasete sme vykonali množstvo testov, ktoré sú uvedené v kapitolách o experimentoch.

Tvorba dátového súboru

V procese tvorby datasetu sme zvažovali rôzne prístupy na jeho vytvorenie. Vykonali sme niekoľko experimentov, v ktorých sme testovali rôzne prístup, aby sme sa vysporiadali s nevyváženými triedami v pôvodnej dátovej sade. Ako prvý nápad sme zvážili využitie zabudovanej funkcionality väčšiny modelov, ktorá automaticky vyvažuje dáta počas trénovacieho procesu. Takto natrénované modely sme následne otestovali na nevyváženej testovacej množine z pôvodnej dátovej sady. Testy úspešnosti sme porovnali od základných algoritmov strojového učenia spomenutých v kapitole 6, 7 až po modely založené na architektúre Transfor-

mer z kapitoly 8. Pri testovaní na nevybalasovanej testovacej množine pôvodného očisteného datasetu vykazovali tieto modely vysokú falošnú pozitivitu.

V nasledujúcom kroku sme sa venovali úpravám datasetu pomocou metód upsamplingu a downsamplingu na vyrovnanie tried. Upsampling spočíva v zväčšení počtu príkladov v minoritnej triede, zatiaľ čo downsampling znamená redukciu počtu príkladov v majoritnej triede na rovnaký počet ako v minoritnej triede. Pri porovnávaní týchto prístupov sme zistili, že downsampling je lepšou možnosťou. Pri použití metódy upsamplingu a náhodnom výbere s opakovaním sme zistili, že aby jednotlivé modely boli schopné detegovať všetky fake-news v upsampled datasete, musia sa spoliehať aj na vlastnosti, ktoré fake-news obsahujú minimálne. To bolo spôsobené väčším počtom takýchto článkov v upsampled datasete. Lenže to spôsobilo, že sa zvýšil prekryv spoločných vlastností medzi fake-news a pravdivými článkami, čo viedlo k zníženej úspešnosti modelu. Po porovnaní so zmenšenou verziou, kde sme manuálne overili pravdivostnú hodnotu dát a nevytvárali príklady automaticky, sme zistili, že metóda upsamplingu nedosahuje také dobré výsledky. Preto sme sa rozhodli pracovať so zmenšenou verziou datasetu, ktorú podrobne opíšeme v nasledujúcej časti.

V prvom kroku bol tento dataset očistený od netextových dát, rovnako ako pri predspracovaní textových dát pre dataset SlovakCovid19 FN v sekcii 9.2.1. Po tomto očistení nasledovala samotná tvorba datasetu. Prvým krokom bola extrakcia článkov, ktoré sú nepravdivé. Extrahovali sme všetky fake-news články, ktoré ostali po očistení z pôvodného datasetu. Žiaden článok sme nevynechali z dôvodu, že v pôvodnom datasete tvorili približne iba 6% t.j. asi 800 článkov. V novom datasete sme chceli zachovať čo najväčší počet článkov, z tohto dôvodu sme ponechali všetky fake-news články z pôvodného datasetu, ktoré ostali po očistení od šumu (počet klesol zhruba o 100).

Druhým krokom tvorby datasetu bolo vybratie rovnako veľkej množiny pravdivých článkov z pôvodného datasetu. To sme urobili nasledujúcim spôsobom: vybrali sme prvú náhodnú množinu pravdivých článkov, túto množinu sme celú manuálne prešli a overili jej pravdivostnú hodnotu. V tomto kroku boli niektoré nevhodné články odstránené. Takže ich bolo nutné nahradiť novou náhodnou množinou z dát, ktoré ostali ešte v datasete. Tento krok sme opakovali 3-krát. Overenie pravdivostnej hodnoty pôvodných článkov robili traja ľudia. Táto fáza zabrala približne 1 hodinu denne po dobu 3 týždňov.

Po dokončení tohto kroku sme vytvorili nový DownSampled SlovakCovid19 FN dataset, ktorý je tvorený iba textovými dátami a prešiel dodatočnou manuálnou anotáciou. DownSampled SlovakCovid19 FN je vybalansovaný dataset, ktorý je tvorený 1466 článkami s tematikou covidu-19.

10. Non-NLP detekcia fake-news

V tejto práci sa primárne zaoberáme detekciou falošných správ na základe ich textového obsahu. Tento problém chápeme ako úlohu textovej klasifikácie a väčšina experimentov, ktoré sme vykonali a budú popísané v nasledujúcich kapitolách, sa zaoberá práve touto problematikou. Princíp detekcie falošných správ vyplynul z dostupných dát pre slovenský jazyk, ktoré sa nám podarilo získať. Základná sada experimentov sa predsa len zameriava na nelingvistické metódy detekcie fake-news v slovenskom jazyku, ktoré sme popísali v kapitole 3. Dataset, ktorý sme získali pre slovenský jazyk, nie je primárne koncipovaný pre nelingvistické metódy detekcie falošných správ. Preto neobsahuje štruktúrované informácie o zdrojových stránkach, autorovi a prílohách (obrázky, odkazy, zdroje atď.).

Vzhľadom na to, že sme ho získali v jeho zdrojovej a neočistenej forme, zdrojový súbor dát sme museli predspracovať a očistiť od niektorých netextových dát, ako sú HTML tagy alebo nealfanumerické znaky typu emotikony a mnohé iné. Rovnako sme odstraňovali metadáta, ktoré boli uložené spolu s textovým obsahom webovej stránky. To sme museli uskutočniť z dôvodu, že tieto dáta boli iba v podmnožine textových súborov a mohli by skresliť výsledky lingvistických metód strojového učenia.

Počas čistenia dát a preprocessingu sme pre podmnožinu textov extrahovali informácie, ktoré sú v zaujímavom vzťahu k niektorým stránkam, autorom alebo špecifickým webovým odkazom, ktoré indikujú vyšší výskyt fake-news. Potom sme na základe nich potvrdili niektoré hypotézy o nelingvistických metódach detekcie falošných správ aj na podmnožine datasetu pre slovenský jazyk. Výsledky týchto zistení uvádzame v tejto kapitole.

10.1 Autor textu

V rade experimentov rôznych zahraničných štúdií bola popísaná súvislosť medzi falošnými správami a ich autorom, pretože autor je typicky zodpovedný za tvorbu a šírenie nepravdivých alebo zavádzajúcich správ. Autori, ktorí majú zlý úmysel, môžu zámerne vytvárať fake-news s cieľom zmanipulovať verejnú mienku, ovplyvniť voľby alebo podnietiť konflikty. Takíto autori sa môžu snažiť vyvolávať emócie, ako sú strach, hnev alebo podozrenie, aby dosiahli svoje ciele.

Okrem toho sa však môže stať, že aj dobre zameraní autori môžu neúmyselne šíriť fake-news. To môže nastať, ak autori neoverujú informácie, ktoré zdieľajú, alebo ak dôverujú zdrojom, ktoré sú nespoľahlivé alebo neoverené.

Týchto autorov spája jedno, že v ich tvorbe sa vyskytuje väčšie percento falošných správ. Fakt, že autor, ktorý publikoval fake-news, má vyššiu tendenciu publikovať tento typ správ aj v budúcnosti, potvrdilo množstvo zahraničných štúdií. Potvrdenie tejto hypotézy sme očakávali aj pre slovenský jazyk, ale výsledky nás mierne prekvapili. Aj keď pri spätnom vyhodnotení a zamyslení sa nad nimi a nad veľkosťou slovenského novinového priestoru je pravdepodobné, že za väčšinu fake-news na Slovensku môže hŕstka novinárov, ktorí ich šíria.

Po očistení textových dát od HTML tagov a metadát sme uskutočnili rad vizualizačných experimentov s dátami, aby sme lepšie pochopili slovnú skladbu týchto textov. Pri zobrazení najfrekventovanejších slov pomocou word cloudu

a následnej kalkulácii štatistík výskytu týchto slov sme získali zaujímavé zistenia na podmnožine dátovej sady, ktorú tvoria falošné správy. V tejto množine textov sa na popredných miestach ocitli aj mená, ktoré boli po bližšom výskume vyhodnotené ako mená autorov týchto článkov. Jedným z nich je aj *Jaroslav Zajac*. Po preskúmaní článkov spojených s týmto menom sa ukázalo, že *Jaroslav Zajac* je autorom 129 článkov, z nich 124 je klasifikovaných ako fake-news. A to z celkového počtu 851 falošných správ, ktoré tvoria dátovú sadu. To znamená že každá ôsma správa, ktorá je klasifikovaná ako falošná, pochádza od tohto autora.

Ďalšími autormi, ktorých by sme mohli zaradiť do tejto skupiny „šíriteľov" falošných správ, sú *Jana Tutková*, *Patrik Sloboda* a napríklad aj *Ján Lakota*. Napríklad *Jana Tutková* má publikované 3 články, *Patrik Sloboda* ich má 8 a posledným autorom, ktorého sme spomenuli, je *Ján Lakota* s 5 článkami. Títo autori majú iba jednotky článkov v našej dátovej sade, ale všetky ich články sú klasifikované ako fake-news. Týmto ich nechceme oficiálne označiť za šíriteľov fake-news, ale v našej dátovej sade majú vyššie percento fake-news a v nej by takto označení boli. Oficiálne označenie je ale možné až na základe väčšieho množstvá dát, no ani vtedy sa to nedá tvrdiť so 100% istotou.

Na druhej strane stoja autori *Ivan Lehotský* a *Tatiana Stará*, u ktorých nemáme zaznamenaný v našom datasete žiaden falošný článok. Od prvého menovaného autora máme v datasete 52 článkov a u druhej autorky dokonca úctyhodných 142 pravdivých článkov.

Výsledkom týchto zistení pre náš dataset je potvrdenie hypotézy, že niektorí autori majú vyššiu tendenciu publikovania falošných správ - či už úmyselne, alebo nie.

10.2 Zdroj textu

Ako sme uviedli v kapitole 3, automatická kontrola zdrojov by mohla byť efektívnou možnosťou detekcie falošných správ. Tento spôsob by odhalil pôvod informácie, teda či pochádza z hodnoverného zdroja. Ak by sa v reťazci zdrojov zacyklila, označila by danú informáciu za nedôveryhodnú.

Zdroj	Pravdivé články	Fake-news
NaturalNews.com	0	5
Badatel.net	1	29
rt.com	6	31

Tabuľka 10.1: Zdroje fake-news.

Podobne ako v predošlom prípade detekcie falošných správ na základe autora ani v tomto prípade nemáme v datasete uvedené zdrojové údaje pri všetkých článkoch ani samostatne, ani v rámci ich textovej formy. V tomto prípade je ťažké určiť, čo je zdroj pre daný text, pretože ak nie je uvedený v záhlaví alebo niekde v určitej časti textu, ale iba ako *URL*, je to automaticky ťažšie detegovateľné. Avšak pri vizualizácii datasetov sa nám podarilo taktiež odhaliť menšiu množinu zdrojov, ktoré sú zodpovedné za šírenie falošných správ ako *NaturalNews.com*, *Badatel.net*, *rt.com*. Z týchto zdrojov čerpajú články, ktoré sú vo väčšine prípadov klasifikované ako fake-news.

Ako je možné vidieť v tabuľke 10.1, falošné správy sa často opierajú o weby, ktoré majú cieľ navodiť dojem pravdivej správy. Zistili sme, že falošné články v našej dátovej sade sa často tvária ako nezávislé médiá alebo vyjadrujú opozíciu voči politickým entitám a orgánom. Ich cieľom je vytvoriť dojem, že sú spoľahlivým zdrojom informácií. Pri preskúmaní falošných článkov v dátovej sade sa ukázalo, že obsahujú zvýšené množstvo slovných spojení ako *nezávislý*, *sloboda slova*, *podpora slobody slova* a iné, ktoré majú u čitateľa podvedome navodiť pocit dôvery v toto médium, aby v ďalšom kroku takto navodenú dôveru využili na šírenie falošných správ.

Preto je pri detekcii falošných správ rovnako dôležité overiť relevantnosť autora a aj zdroje, o ktoré sa napísaný článok opiera.

10.3 Špecifický textový obsah v metadátach

Pri skúmaní textov v dátovej sade sme zistili ešte jednu zaujímavú vlastnosť. Približne polovica falošných správ v datasete obsahovala v appendixe minimálne jeden z týchto reťazcov: *„Ak vás článok...“*, *„Vyhlásenie: Názory autora sa nemusia zhodovať...“* alebo *„Sme pod neustálym tlakomMnohým kruhom nevyhovuje existencia Hlavných správPodporte jediné väčšie nezávislé médium na Slovensku“*.

Bližším skúmaním týchto textov sme prišli k zisteniu, že všetky pochádzajú z jedného zdroja, ktorým boli *Hlavné správy*. Aj keď v sebe článok priamo neodkazoval na tento web, pri tvorbe datasetu sa spolu s hlavným telom článku stiahli aj tieto dodatočné informácie z päty webu. Následne pri spracovávaní sme našli 100-percentnú zhodu týchto reťazcov s webom *Hlavné správy*, rovnako ako ich umiestnenie v rámci článkov, ktoré na ňom publikujú. Detailnejší popis je uvedený v podkapitole 9.2, ktorá sa zaoberá predspracovaním slovenského datasetu.

Články, ktoré obsahovali aspoň jeden z týchto reťazcov, boli všetky klasifikované ako fake-news. Touto zaujímavou vlastnosťou článkov sa nepriamo ukázalo, aká dôležitá je identifikácia zdroja stránky.

10.4 Zhrnutie

Na základe týchto zistení sa potvrdilo, že pri tvorbe komplexného systému na detekciu falošných správ pre menšie jazykové korpusy je rovnako dôležitá identifikácia falošných správ na základe autora, zdroja a priložených médií ako detekcia pomocou textového obsahu konkrétneho článku. Preto by mal byť tento systém tvorený ako hybridný model, ktorý v sebe spája viacero uhlov pohľadov na falošné správy a hodnotí ich na základe celkového pohľadu na problematiku fake-news. Výsledné klasifikačné skóre by bolo určené ako kombinácia výsledkov riešení týchto čiastkových problémov. Tento prístup minimalizuje riziko falošne pozitívnej/negatívnej klasifikácie.

11. Základná sada experimentov

V prvej sade experimentov sa zameriame na základné algoritmy strojového učenia, ktoré sme uviedli v kapitole 6. Cieľom týchto experimentov je nastaviť baseline pre komplexnejšie metódy a experimenty pre každý zo slovenských datasetov z kapitoly 9.

Každá podkapitola sa zaoberá rovnakou sadou experimentov vykonaných iba na inom slovenskom datasete. Pre všetky dátové sady sú vykonané tieto experimenty: V prvom experimente sú porovnané základné metódy strojového učenia pomocou krížovej validácie. V nasledujúcom experimente sme pomocou optimalizácie hyperparametrov našli optimálnu voľbu parametrov pre najlepší klasifikátor z prvého experimentu. Posledný experiment je venovaný ensemble systémom, v ktorých je predpoklad zlepšenia oproti samostatným klasifikátorom z predošlých experimentov.

11.1 Príprava vstupných dát

Vzhľadom na rôznorodosť formátov spracovávaných dát je nevyhnutné vykonať normalizáciu textu. Normalizácia textu je proces úpravy vstupného textu tak, aby bol vhodný pre automatické strojové spracovanie. V rámci tejto normalizácie sa vstup štandardizuje do jednotného formátu. V našom prípade sme vykonali nasledujúce kroky normalizácie textu pre základnú sadu experimentov:

- Zmena formátu písmen: Všetky písmená sa môžu upraviť na malé alebo veľké, nezáleží na akú veľkosť, štandardizácia veľkosti textu napomáha, aby rovnaké slová, ktoré sa od seba odlišujú iba veľkosťou písma, znamenali pre model to isté. V tejto práci sme štandardizovali veľkosť písma na malé.
- Tokenizácia: Proces, ktorý rozdeľuje text na jednotlivé skupiny znakov, oddelené medzerami alebo inými symbolmi. Tieto skupiny znakov sa nazývajú tokeny a môžu reprezentovať slová alebo frázy [69].
- Odstránenie interpunkcie: Interpunkčné znamienka ako bodky, čiarky alebo dvojbodky sme odstránili, aby sa zjednotil formát textu. Interpunkcia nemá žiaden špecifický význam pre základné modely pri následnej analýze textov, no jej odstránenie nám pomôže skrátiť dĺžku textov.
- Odstránenie diakritiky: Diakritické znamienka (napr. dĺžne, mäkčene) na písmenách sme taktiež odstránili, aby sa zjednotila forma slov a výrazov.
- Odstránenie stopwords: Slová ako „a“, „alebo“, „na“, „pri“, „s“ atď. sme taktiež odstránili, aby sa zredukoval počet slov v texte a zlepšila sa kvalita analýzy [69].
- Lemmatizácia: Proces je založený na odstraňovaní flexie slova pomocou rôznych pravidiel a tým upraviť slovo na základný tvar, tzn. slovo „bežať“ sa zjednotí s tvarmi „bežím“, „bežal“, „bežal by som“ na základný tvar „bežať“. No s dôrazom na to, že výsledkom lemmatizácie je skutočné slovo t.j. lemma. Lemma je základný tvar slova, slovníkový tvar. To túto metódu odlišuje napríklad od izolácie koreňa slova, kde výsledkom nie je slovo [70].

Tento proces štandardizácie dát, ktorý v sebe zahŕňa odstránenie diakritiky, stopwords, interpunkcie a ďalších znakov jazyka, sme využili na prípravu dát pre základné modely strojového učenia. Normalizácia vstupu je veľmi dôležitým krokom pre ďalšie procesy úpravy dát. Predspracovanie textu pred použitím metód strojového učenia pomáha zjednodušiť a štandardizovať textové dáta, odstrániť zbytočné informácie a zlepšiť kvalitu a interpretáciu textového vstupu. To vedie k lepšiemu výkonu a efektívnosti modelov strojového učenia a analýze textu. Tento proces sa v angličtine označuje ako *„dimensionality reduction“* (redukcia dimenzie). Je to technika, ktorá sa používa na zmenšenie počtu rozmerov alebo atribútov v sade údajov. V našom prípade sa snažíme znížiť počet tokenov (slov) v slovnej zásobe. Cieľom je uchovať čo najviac informácií z pôvodných textov, no zároveň zjednodušiť ich štruktúru a zložitosť.

Po tomto kroku získavame text očistený od šumu a nepotrebných slov, z ktorých nie je možné získať dodatočné informácie. Tieto metódy majú za cieľ zlepšiť spracovanie textov a zabezpečiť, aby NLP modely lepšie porozumeli a efektívne pracovali s textovými dátami. Avšak dôležitosť a vhodnosť týchto metód závisí od konkrétnej NLP úlohy a používaného datasetu. Preto sú tieto metódy použité iba v základnej sade experimentov a v sade experimentov zameraných na hlboké neurónové siete [70].

11.2 Dataset DownSampled SlovakCovid19 FN

Prvým datasetom, na ktorom sme vykonali základnú sadu experimentov, je DownSampled SlovakCovid19 FN. Tento dataset bol nami vytvorený z dôvodu veľkého nevybalansovania pôvodného datasetu a má slúžiť na vytvorenie základných výsledkov a oboznámenie so slovenskými dátami, ktoré nie sú ovplyvnené nevybalansovanou dátovou sadou. Detailnejšie informácie o tvorbe tohto datasetu sú uvedené v podkapitole 9.2.2. Pre tieto experimenty bol dataset predspracovaný, ako je uvedené v podkapitole 9.2.2 a v úvode tejto kapitoly 11.1.

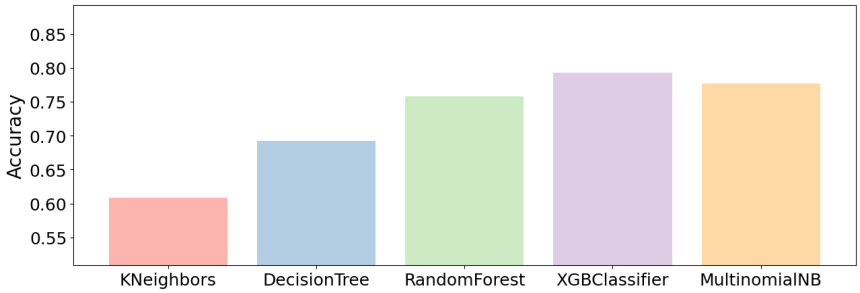
11.2.1 Porovnanie algoritmov strojového učenia

Prvým experimentom v základnej sade je porovnanie jednoduchých modelov strojového učenia pomocou krížovej validácie na celej dátovej sade. V rámci tohto experimentu sme si vybrali nasledujúce algoritmy strojového učenia na porovnanie: rozhodovací strom, náhodný les, k-NN (k-nearest neighbors), naivný Bayesov klasifikátor a XGB klasifikátor. Algoritmy strojového učenia, ktoré sme vymenovali, boli použité v ich základnom nastavení bez špecifickej úpravy ich učiacich parametrov.

Vyhodnotenie sme realizovali pomocou 10-fold krížovej validácie, ktorá bola zvolená z dôvodu spoľahlivejšieho odhadu výkonnosti modelov v porovnaní s jednoduchým rozdelením dát na trénovaciu a testovaciu množinu. Porovnanie na celej dátovej sade bolo realizované s cieľom minimalizovať vplyv náhodných fluktuácií v dátach na výsledky a má poskytnúť robustnejšie porovnanie medzi modelmi. Jednotlivé algoritmy strojového učenia sme porovnali na základe nasledujúcich metrík: F1 skóre, Accuracy a AUC.

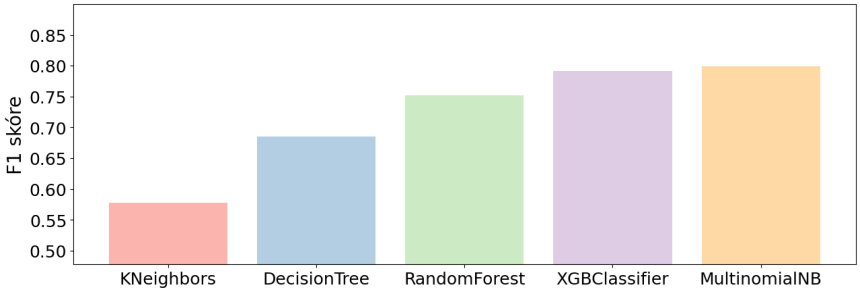
Vzhľadom na to, že základné algoritmy strojového učenia nie sú schopné pracovať s textovým vstupom, bolo nutné vstupný dataset transformovať do nume-

rickej reprezentácie v procese, ktorý sa nazýva vektorizácia textu. Pre základnú sadu experimentov sme zvolili vektorizáciu textu pomocou metódy BoW v spojení s odstránením stopwords z textov, čoho výsledkom bola slovná zásoba s veľkosťou 70000. V nasledujúcich častiach tejto podkapitoly uvedieme výsledky experimentov a ich interpretáciu. Sumarizácia výsledkov tejto sady experimentov je v prílohe A.4 v tabuľke A.1. Ako prvú uvedieme vizualizáciu úspešností jednotlivých algoritmov pomocou prehľadných stĺpcových grafov pre nami zvolené metriky. Na záver všetky výsledky uvedieme v tabuľke, ktorá v sebe bude obsahovať presné hodnoty, ktoré dosiahli jednotlivé algoritmy.



Obr. 11.1: Porovnanie klasifikátorov na základe metriky: Accuracy.

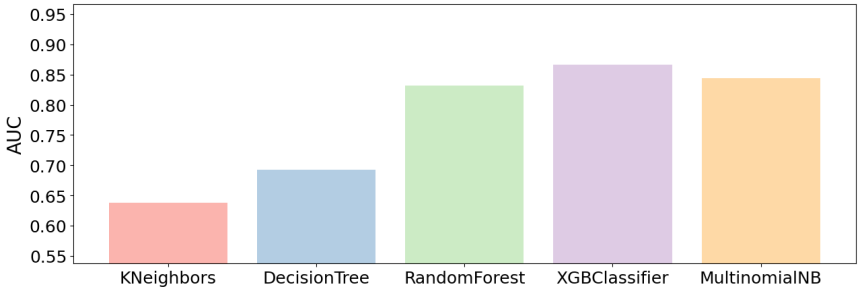
Na vizualizácii 11.1 je zobrazený graf porovnania klasifikátorov na základe priemernej hodnoty metriky accuracy cez všetky behy krížovej validácie jednotlivých algoritmov strojového učenia, ktoré sme testovali. Podľa tejto metriky sa na prvých troch miestach umiestnili algoritmy: XGB klasifikátor, naivný Bayesov klasifikátor a náhodný les. Prvé dva menované dosiahli podobné skóre, pričom tretí už za prvým značne zaostáva. Náhodný les je horší ako XGB klasifikátor približne o 0,04 (4%). Ďalšie dva algoritmy strojového učenia už nedosahujú kvality prvých troch spomenutých. Posledný dosahuje zhruba accuracy 0,61 na danom datasete. Prvý menovaný XGB klasifikátor dosiahol priemernú hodnotu skóre $acc = 0{,}793$. Teda najhorší algoritmus dosahuje o 0,18 (18%) nižšiu accuracy ako ten najlepší. Táto metrika ukazuje, že vhodná voľba algoritmu strojového učenia pre riešenie problému detekcie fake-news je zásadná.



Obr. 11.2: Porovnanie klasifikátorov na základe metriky: F1 skóre.

Druhá vizualizácia 11.2 je venovaná porovnaniu algoritmov pomocou metriky F1 skóre. Podobne ako pri predošlej metrike sa na prvých miestach umiestnili naivný Bayesov klasifikátor, XGB klasifikátor a náhodný les. Len v tomto prípade si prvé miesto prvé dva algoritmy vymenili. Ale rozdiel v ich úspešnosti je zanedbateľný. Podobnosť medzi vyhodnotením metrík accuracy a F1 skóre je

spôsobená vybalansovaným datasetom, teda aj výsledky ostatných algoritmov sú skoro totožné ako pri predošlej metrike.



Obr. 11.3: Porovnanie klasifikátorov na základe metriky: AUC.

Posledná metrika, na ktorú sa zameriame pri základnej sade experimentov, je AUC. Porovnaním jednotlivých algoritmov strojového učenia na základe tejto metriky dostávame poradie, ktoré je totožné so zoradením podľa metriky accuracy. Na obrázku 11.3 je znázornený stĺpcový graf úspešnosti na základe tejto metriky pre zvolené metódy strojového učenia.

Tromi najúspešnejšími algoritmami sú XGB klasifikátor, naivný Bayesov klasifikátor a náhodný les. Najúspešnejší algoritmus v tomto prípade je znovu XGB klasifikátor, ktorý aj v zahraničných štúdiách dosahuje dobré výsledky spomedzi základných algoritmov strojového učenia pri riešení lingvistického problému, ktorý spočíva v detekcii fake-news.

Na základe týchto vizualizácií sme vo výslednej tabuľke uviedli presné hodnoty iba prvých troch najúspešnejších algoritmov, keďže ďalšie dva algoritmy dosahovali približne o 1/4 horšie skóre podľa nami zvolených metrík, preto ich pre prehľadnosť vo výslednej tabuľke nezobrazujeme.

Klasifikátor	Accuracy	F1 skóre	AUC
XGBClassifier	0,793	0,791	0,866
MultinomialNB	0,777	0,799	0,844
RandomForestClassifier	0,759	0,752	0,832

Tabuľka 11.1: Porovnanie základných klasifikátorov.

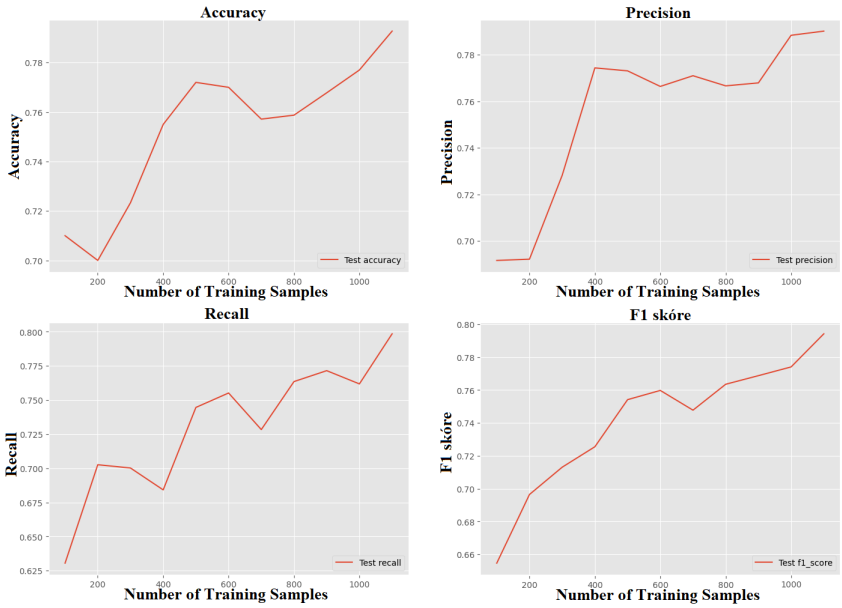
Tabuľka 11.1 obsahuje výsledky v zvolených metrikách troch najlepších klasifikátorov zo základnej sady algoritmov strojového učenia pre DownSampled SlovakCovid19 FN dataset. Z výsledkov experimentov v tabuľke 11.1 môžeme vyčítať, že krížová validácia základných algoritmov pre vyvážený a očistený dataset DownSampled SlovakCovid19 FN dosahuje pomerne dobré výsledky a tie môžeme brať ako dobrý odrazový mostík pre ďalšie experimenty.

Aj jednoduché algoritmy vedia určiť správnu triedu a to môžeme pokladať pri takto náročnej úlohe klasifikácie fake-news a pri našom obmedzenom množstve vstupných dát za úspech. Avšak je dôležité zdôrazniť, že výsledky experimentu sú výrazne ovplyvnené kvalitou dát a spôsobom ich spracovania v krížovej validácii. Vzhľadom na to, že tieto jednoduché algoritmy strojového učenia sú náchylnejšie na kvalitu dát, sa vyhodnotenie krížovou validáciou uskutočnilo na kompletnej

dátovej sade s cieľom znížiť vplyv rôznorodosti príkladov v dátovej sade pri náhodnom výbere. Priznávame, že tento fakt by mohol mať vplyv na porovnanie s výsledkami nasledujúcich komplexnejších algoritmov v prospech základnej sady, pretože v ďalších experimentoch rozdelíme celú dátovú sadu na trénovaciu, validačnú a testovaciu množinu. Na druhej strane: tento experiment je oddelený od ostatných fáz experimentov a slúžil iba na všeobecné porovnanie základnej sady algoritmov medzi sebou v rámci tejto skupiny, hoci základom pre optimalizáciu v ďalšom kroku boli algoritmy strojového učenia vybrané na základe výsledkov tohto experimentu. V nasledujúcej fáze optimalizácie boli tieto algoritmy opäť trénované od základu na oddelenej trénovacej množine a následne vyhodnotené na nezávislej testovacej množine. Tento prístup bol zvolený s cieľom správneho porovnania výkonu týchto algoritmov s komplexnejšími modelmi. Týmto spôsobom sme sa snažili zabezpečiť, že algoritmy prechádzajú celým procesom trénovania a vyhodnocovania na nezávislých množinách, čo by malo zaručiť korektné porovnanie s komplexnejšími modelmi na rovnakých disjunktných množinách.

11.2.2 Vplyv veľkosti trénovacej množiny na úspešnosť klasifikácie

V tejto časti základnej sady experimentov sme skúmali, ako veľkosť trénovacej množiny ovplyvňuje úspešnosť klasifikácie falošných správ. V prvom kroku bola vstupná dátová sada rozdelená do troch množín (trénovacia, validačná a testovacia) v pomere 6:2:2. Rozdelenie prebehlo so zachovaním rozloženia tried v jednotlivých množinách. V tomto experimente sme použili iba trénovaciu a validačnú množinu. Testovacia množina bola vytvorená, ale nebola použitá, aby sme zachovali nezávislosť pre nasledujúce experimenty.



Obr. 11.4: Vplyv veľkosti trénovacej množiny na skúmané metriky.

Grafy na ilustrácii 11.4 znázorňujú vplyv veľkosti trénovacieho datasetu na výslednú úspešnosť klasifikátora. Vizualizácia bola vytvorená pomocou XGB klasifikátora iba na kombinácii trénovacej a validačnej množiny. Vyhodnotenie v každom

bode sa vždy konalo pomocou 5-foldovej krížovej validácie na podmnožine danej veľkosti. Táto podmnožina bola vytvorená z kombinácie trénovacej a validačnej množiny náhodným výberom príkladov. Ukázalo sa, že zväčšovanie veľkosti trénovacej množiny pre úlohu klasifikácie fake-news má pozitívny vplyv na výkonnosť modelu vo všetkých skúmaných metrikách. Hoci nastal mierny prepad vo všetkých metrikách okolo hodnoty 600, ale je iba dočasný a nemá dlhodobý vplyv. Za výsledok tohto experimentu môžeme brať zistenie, že ak chceme dosiahnuť lepšie výsledky pre klasifikáciu fake-news, jednou z možností je práca na veľkosti trénovacích datasetov, ktoré sú v dnešnej dobe pre slovenský jazykový korpus nedostatočné.

11.2.3 Optimalizácia hyperparametrov

Na základe výsledkov predošlých experimentov sme sa v tomto zamerali na optimalizáciu hyperparametrov pre najlepšie tri metódy z prvého experimentu v tejto sade. V procese optimalizácie hyperparametrov jednotlivých algoritmov sme ich testovali v kombinácii s dvoma vektorizačnými metódami BoW a TD-IDF z kapitoly 5. Po optimalizácii hyperparametrov algoritmus XGBClassifier v kombinácii s TD-IDF metódou vektorizácie vstupu dosahoval najlepšie zlepšenie oproti zvyšným dvom algoritmom. Preto výsledky optimalizácie tohto algoritmu uvedieme v tejto sekcii v kombinácii s vektorizačnou metódou TD-IDF.

```
tvec = TfidfVectorizer(
   max_features= 3000,
   ngram_range= (1, 3),
   max_df= 0.65,
   stop_words= stopwords
)
```

Python implementácia 11.1: Opt. hyper. vektorizácia textu.

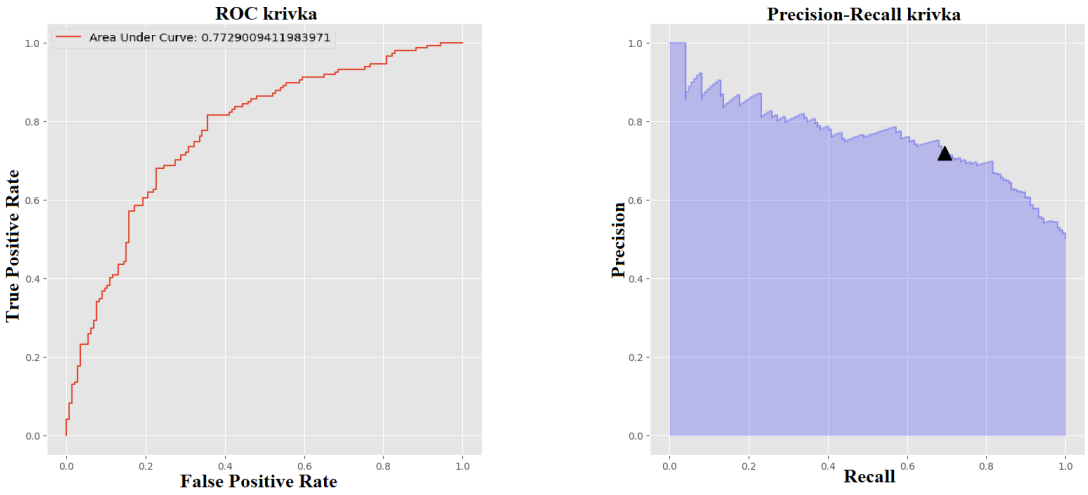
Na optimalizáciu hyperparametrov sme použili Tree Parzen Estimator. Algoritmus používa kombináciu stromových tried na vyhodnotenie hyperparameterov a parzenovských odhadov na modelovanie pravdepodobností. Priestor hyperparameterov je rozdelený na dva susedné stromy - prior a posterior. Prior strom modeluje pravdepodobnostnú distribúciu neinformovaných hodnôt hyperparameterov, zatiaľ čo posterior strom modeluje distribúciu hodnôt, ktoré dosiahli lepšie výsledky. V každej iterácii algoritmus generuje novú sadu hyperparameterov na základe prior stromu a vyhodnocuje ich pomocou validačnej množiny. Na základe výsledkov aktualizuje posterior strom a pokračuje v ďalších iteráciách. Cieľom je minimalizovať chybu výpočtu hodnôt hyperparameterov pomocou parzenovského odhadu pravdepodobnosti. Po 15 iteráciách tejto metódy na kombinácii trénovacej a validačnej množiny bola objavená táto množina parametrov ako optimálna:

```
params = {
   'learning_rate': 0.225,
   'max_depth': 5,
   'min_child_weight': 1.0,
   'n estimators': 600,
   'normalize': 0,
   'scale': 1,
   'subsample': 0.9
   }
```

Python implementácia 11.2: Opt. hyper. XGBoost.

Po optimalizácii hyperparametrov sme vykonali testovanie modelu XGBoost na nezávislej testovacej množine, ktorá bola vyčlenená z datasetu DownSampled SlovakCovid19-FN v prvom kroku, s výsledkami: $acc = 0{,}7065$ a $F1 = 0{,}7062$.

Na ilustrácii 11.5a je znázornená ROC krivka, ktorá sa spočítala na základe výsledkov klasifikátora XGB na testovacej množine. Ideálna ROC krivka by mala najskôr kolmo stúpať hore a až potom zvyšovať mieru falošnej pozitivity. Ak sa teda pozrieme detailnejšie na obrázok 11.5a, ktorý zobrazuje ROC krivku pre náš baseline klasifikátor XGB, môžeme z neho vidieť, že tento model má od ideálneho prípadu ešte ďaleko. I keď je tam značný logaritmický nárast, ešte stále by mohol byť prudší. Na druhej strane náhodný klasifikátor by mal ROC krivku v oblasti diagonály, takže už aj tento jednoduchý klasifikátor, ktorý sme si zvolili za baseline model, prekonáva náhodný výber. S ROC krivkou súvisí ešte jedná veličina - AUC, ktorá popisuje ROC krivku. Nami dosiahnutá hodnota pomocou XGB klasifikátoru bola 0,7729, čo radí kvalitatívne tento klasifikátor ako dobrý model.



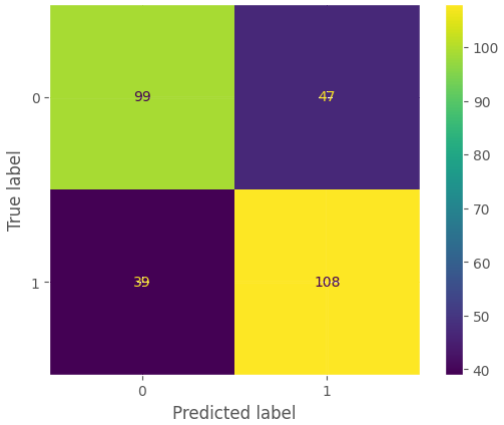
(a) ROC krivka.

(b) Precision-recall krivka.

Obr. 11.5: Baseline model: ROC a precision-recall krivka.

Ďalšou krivkou, ktorá sa používa na vyhodnotenie kvality testovaného klasifikátora, je precision-recall krivka. Krivka je vyobrazená pre nami testovaný klasifikátor XGB na obrázku 11.5b. Pre túto metriku sme sa rozhodli vzhľadom na porovnanie s nevybalansovaným datasetom. V tomto experimente, keď sú klasifikované triedy pravdivých článkov a fake-news vyrovnané, nemá o toľko vyššiu výpovednú hodnotu ako ROC krivka. Jej sila sa ukáže až pri nevybalansovanom prípade dát. Výsledná krivka má očakávaný charakter priebehu. Tento klasifikátor dosahuje mierne nadpriemerné hodnoty, ako sme to od baseline modelu očakávali. Optimálna hodnota precision oproti recall (threshold) je znázornená ako bod (0,75; 0,7) na obrázku 11.5b. Ešte však nedosahuje optimálne výsledky precision oproti recall.

Z konfúznej matice modelu XGB klasifikátora môžeme vyčítať, že klasifikuje nesprávne každú štvrtú falošnú správu. A na každé dve správne klasifikované falošné správy nesprávne zaradí do tejto množiny aj jeden pravdivý článok.



Obr. 11.6: Baseline model: konfúzna matica.

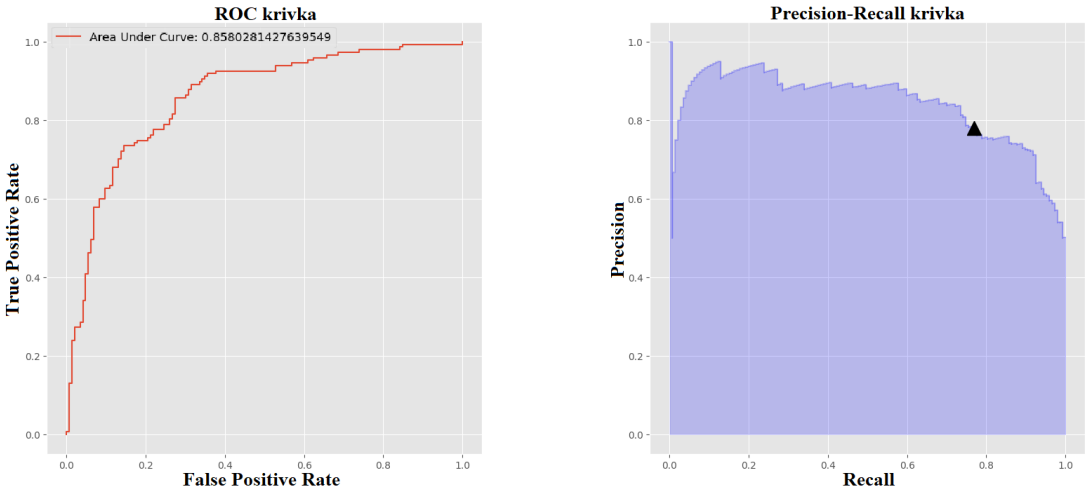
11.2.4 Ensemble metóda

Posledný experiment, ktorý sme vykonali v základnej sade na DownSampled SlovakCovid119 FN datasete, bol zameraný na otestovanie ensemble metódy a jej prínosu pri klasifikácii fake-news. Pre tento test ensemble metód sme si zvolili voting classifier (soft voting), ktorý na základe predošlých experimentov v sebe kombinuje základné algoritmy strojového učenia: náhodný les, naivný Bayesov klasifikátor a XGB klasifikátor.

Klasifikátor XGB ma zvolené hyperparametre na základe výsledkov predošlého experimentu optimalizácie tohto klasifikátora. Zvyšné dva klasifikátory majú základné nastavenie parametrov. Motivácia k nastaveniu týchto algoritmov týmto štýlom mala cieľ ukázať, aký prínos má samotná ensemble metóda. V tomto experimente chceme ukázať, že aj rozšírenie najlepšieho predošlého modelu o menej kvalitné modely (základné nastavenie parametrov) bude viesť k zlepšeniu výslednej klasifikácie tohto systému. Keďže aj v tomto teste metód strojového učenia pracujeme s rovnakými algoritmami, bola nevyhnutná vektorizácia textov, t.j. transformácia textových dát do ich vektorovej podoby. Pre tento experiment sme naďalej pokračovali vo vektorizácii vstupu pomocou TF-IDF s rovnakým nastavením parametrov ako v predošlom experimente. Výsledná hodnota klasifikácie je vyjadrená ako priemerná hodnota klasifikácií modelov, ktoré tvoria tento ensemble systém, tzv. soft voting, ktorý je detailnejšie popísaný v podkapitole 6.4. V nasledujúcej časti popíšeme dosiahnuté výsledky tohto klasifikátora na DownSampled SlovakCovid19 FN datasete. Trénovanie prebehlo na nezávislej trénovacej množine, ktorú sme vytvorili v predošlom experimente, a následné overenie prebehlo na testovacej sade tejto množiny.

Na grafe 11.7a môžeme vidieť ROC krivku pre testovaný ensemble systém. Ak priebeh tejto krivky porovnáme s baseline klasifikátorom z predošlého experimentu 11.5a, je zjavné, že krivka ensemble systému ma rýchlejší rast, čo interpretujeme ako vyššiu kvalitu modelu, pretože hneď od začiatku nezvyšuje úmerne falošnú pozitivitu. Oproti baseline modelu má bližšie k optimálnemu klasifikátoru na základe ROC krivky. Pre porovnanie s baseline modelom použijeme ešte AUC hodnotu, ktorá súvisí priamo s ROC krivkou. Táto hodnota pre ROC krivku ensemble systému dosahuje hodnotu 0.8580, kdežto baseline XGB klasifikátor dosiahol hodnotu 0,7729. Ensemble systém sa touto hodnotou dostal už medzi veľmi

dobré klasifikátory, ale baseline model bol svojím skóre AUC zaradený iba medzi dobré metódy klasifikácie fake-news.

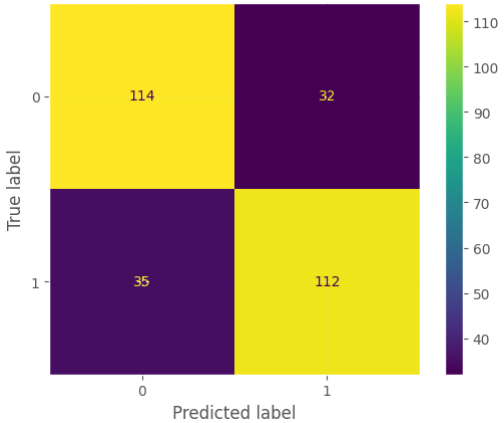


(a) ROC krivka.

(b) Precision-recall krivka.

Obr. 11.7: Ensemble model: ROC a precision-recall krivka.

Ďalšia metrika, na ktorej základe môžeme ensemble systém porovnať s baseline modelom, je precision-recall krivka. Túto metriku môžeme vidieť na vizualizácii 11.7b. Rovnako ako pri teste baseline modelu na vybalansovanom datasete táto metrika porovnania precision vs. recall veľa nepovie. Ale ak túto krivku porovnáme s precision recall krivkou pre baseline klasifikátor XGB, môžeme postrehnúť, že charakter priebehu krivky má bližšie k ideálnemu prípadu, keď ide o krivku smerujúcu k bodu (1; 1). Hoci je v úvode značný prepad precision až k hodnote 0,5, ale tento vzor je typický pre väčšinu týchto kriviek a je známy ako *„sawtooth pattern“*. Bod vyznačený na ilustrácii 11.7b je aktuálny threshold, ktorý je pre tento model približne v bode (0,8; 0,8) a v porovnaní s baseline modelom, ktorý dosahoval optimálnu hodnotu v bode (0,75; 0,7), predstavuje tiež mierne zlepšenie.



Obr. 11.8: Ensemble model: konfúzna matica.

Rovnako ako metrika ROC aj precision vs. recall potvrdzuje zlepšenie baseline modelu doplnením o ďalšie dva základné modely strojového učenia a ich následné spojenie do ensemble systému. Na obrázku 11.8 je vizualizácia konfúznej matice ensemble modelu. Na základe tejto konfúznej matice sme spočítali $acc = 0{,}7713$ a $F1 = 0{,}7713$. Výsledné metriky sú približne o 0,07 (7%) vyššie ako pri baseline modeli. Toto zlepšenie je markantné a len potvrdzuje výsledky zlepšenia popísané pomocou ROC a precision vs. recall kriviek. Ensemble systém dosahuje lepšie výsledky oproti baseline modelu z dôvodu, že kombinácia viacerých klasifikátorov pridáva robustnosť tomuto modelu a znižuje zlú klasifikáciu jednotlivých tried.

11.2.5 Zhrnutie

Cieľom týchto experimentov bolo oboznámenie s datasetom, jeho následné spracovanie pomocou metód strojového učenia a vytvorenie baseline modelu. V druhom experimente tejto základnej sady sme vytvorili baseline model pomocou XGB klasifikátora, ktorý dosiahol najlepšie výsledky z porovnávaných algoritmov pre DownSampled SlovakCovid19 FN dataset. Hodnota základných metrík pomocou baseline modelu je stanovená na $acc = 0{,}7065$ a $F1 = 0{,}7062$.

Tieto hodnoty budeme pri ďalších experimentoch brať ako referenčné a na základe nich porovnávať prínos komplexnejších modelov pri detekcii fake-news na slovenskom datasete. V poslednom experimente sa nám podarilo vytvoriť systém klasifikátorov zo základných modelov strojového učenia, ktoré prekonali baseline model v každej metrike a tým potvrdili, že nielen zlepšovanie dátovej sady vedie k lepším výsledkom, ale aj použitie robustnejších a komplexnejších systémov pre detekciu fake-news.

11.3 Dataset SlovakCovid19 FN

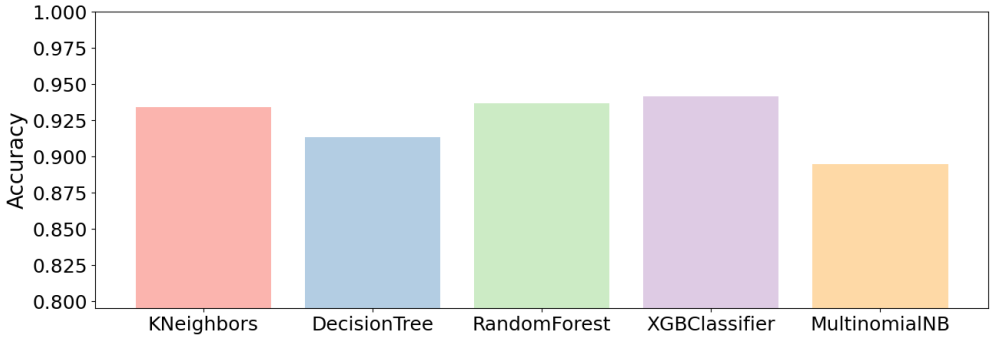
SlovakCovid19 FN je druhým slovenským datasetom, na ktorom sme vykonali základnú sadu experimentov. Tento dataset je pôvodným datasetom, ktorý sme získali od Technickej univerzity v Košiciach. Dataset je značne nevybalansovaný v prospech pravdivých článkov v pomere 93:7. Na základe tohto datasetu bola vytvorená zmenšená verzia s vybalansovanými množinami tried. Základná sada experimentov pomocou jednoduchých algoritmov strojového učenia bola na zmenšenom datasete predstavená v predošlej podkapitole 11.2. Jej úlohou je slúžiť ako referenčná vzorka pre celý pôvodný dataset.

V tejto podkapitole vykonáme rovnakú sadu experimentov ako na zmenšenej verzii datasetu a porovnáme vplyv zväčšenia dátovej sady aj zmenu rozloženia počtu príkladov v jednotlivých triedach.

11.3.1 Porovnanie algoritmov strojového učenia

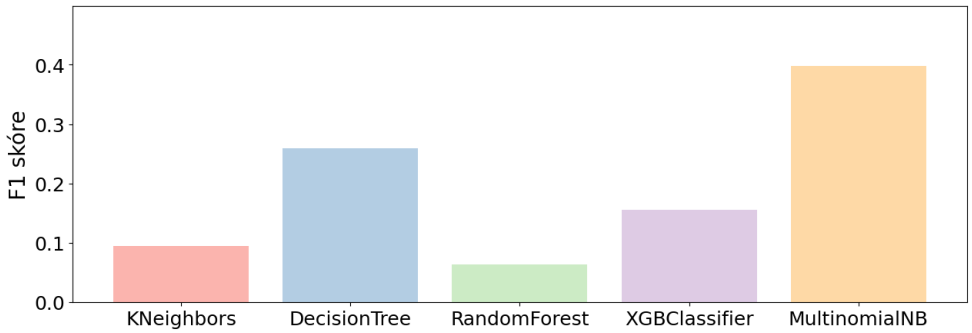
V tomto experimente sme porovnali medzi sebou rovnaké algoritmy strojového učenia ako v podkapitole 11.2. Pre ujasnenie porovnávanými algoritmami sú: rozhodovací strom, náhodný les, k-NN, naivný Bayesov klasifikátor a XGB klasifikátor. Úspešnosť jednotlivých algoritmov strojového učenia sme pre porovnanie s predošlými výsledkami uskutočnili pomocou rovnakých metrík, t.j. F1 skóre, accuracy, AUC. Vyhodnotenie sa uskutočnilo rovnako pomocou 10-foldovej

krížovej validácie na celej dátovej sade. Obdobne ako v predošlých experimentoch základné modely strojového učenia nie sú schopné spracovať textový vstup, aj v tomto prípade bolo nutné previesť transformáciu vstupu v procese vektorizácie textu. Pre zachovanie korektnosti porovnania experimentov na rôznych dátových sadách aj v tomto prípade prebehla vektorizácia pomocou algoritmu BoW s rovnakými parametrami ako v experimente 11.2. Taktiež parametre jednotlivých algoritmov strojového učenia sme ponechali v ich základnom nastavení ako pri predošlom experimente. Najprv ukážeme výsledky experimentov pomocou prehľadných stĺpcových grafov pre jednotlivé algoritmy a metriky. Nakoniec zhrnieme všetky výsledky v tabuľke s presnými hodnotami.



Obr. 11.9: Porovnanie klasifikátorov na základe metriky: accuracy.

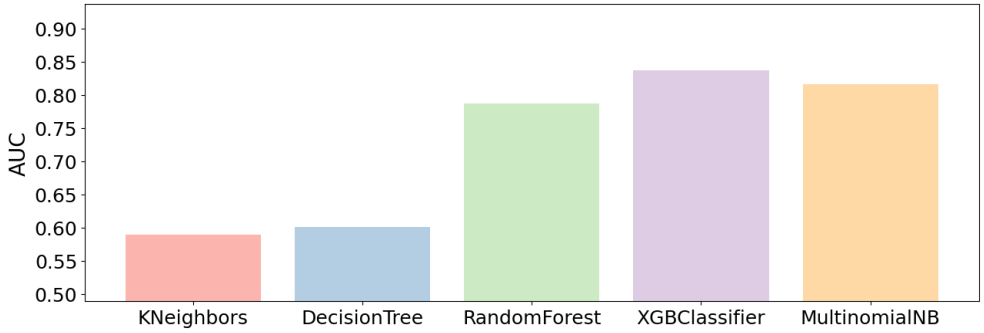
Na grafe 11.9 sú porovnané klasifikátory na základe ich priemernej accuracy získanej z krížovej validácie. Ide o výsledky testovania jednotlivých algoritmov strojového učenia na celom slovenskom datasete. Podľa metriky accuracy cez všetky behy krížovej validácie sa na prvých troch miestach v poradí umiestnili XGB klasifikátor, náhodný les a k-NN. Všetky testované algoritmy dosiahli výsledky s rozdielom v presnosti približne 0,05 (5%). Tento výsledok pre všetky algoritmy bol spôsobený výraznou nerovnováhou rozloženia tried v datasete. A aj keď sa zdá, že na tomto datasete dosahujú jednotlivé algoritmy lepšie výsledky ako na predošlom, nie je to tak. Skôr je to naopak a celá skupina testovaných algoritmov dopadla rovnako zle. Tento výsledok nás neprekvapil, pretože dátová sada je značne nevybalansovaná a je ju ťažké klasifikovať. Pre porovnanie: algoritmus, ktorý by volil dominujúcu triedu, by dosiahol accuracy 0,9381, preto všetky algoritmy, ktoré dosiahli nižšiu hodnotu, môžeme pokladať za neúspešné, pretože sú horšie ako tento triviálny spôsob.



Obr. 11.10: Porovnanie klasifikátorov na základe metriky: F1 skóre.

Na ilustrácii 11.10 sú výsledky jednotlivých algoritmov na základe metriky F1 skóre. Vizualizáciou tejto metriky pomocou stĺpcového grafu pre jednotlivé algoritmy sa ukázalo, že pre takto nevyvážený dataset je pre porovnanie modelov metrika accuracy nepostačujúca. Ak zohľadníme precision vs. recall v metrike F1 skóre. Zistíme, že algoritmy sa správajú podpriemerne a dosahujú ani nie 0,5 hodnotu F1 skóre. Zhoršenie výsledkov v metrike F1 skóre oproti predošlému experimentu (vybalansovaná dátová sada) sme predpokladali, keďže táto dátová sade je markantne nevybalansovaná.

Pri tejto metrike sa plne ukázal problém celého datasetu - jeho nevyváženosť, ktorá bude najväčším problémom aj pri nasledujúcich experimentoch.



Obr. 11.11: Porovnanie klasifikátorov na základe metriky: AUC.

V rámci základnej sady experimentov sme sa zamerali aj na metriku AUC. Na obrázku 11.11 je znázornený stĺpcový graf úspešnosti pre jednotlivé algoritmy podľa tejto metriky. Tromi najúspešnejšími algoritmami sú XGB klasifikátor, naivný Bayesov klasifikátor a náhodný les. Najúspešnejší algoritmus v tomto prípade je znovu XGB klasifikátor, ktorý v tejto sade experimentov dokazuje najstabilnejšie výsledky. Je však dôležité si uvedomiť, že pri hodnotení úspešnosti pomocou metriky AUC je potrebné brať do úvahy aj nevyváženosť dátovej sady. Pri vyhodnocovaní výsledkov tejto metriky je dôležité byť opatrný a zohľadňovať aj ďalšie metriky a charakteristiky dátovej sady, na ktorej boli algoritmy testované. Je potrebné mať na pamäti, že metrika AUC sama osebe nemusí dostatočne zohľadňovať nevyváženosť dát a môže viesť k skresleniu výsledkov. Interpretácia výsledkov by mala byť vykonávaná s ohľadom na tieto faktory.

Podľa grafov zobrazených na obrázkoch 11.9, 11.10 a 11.11 sú najlepšie výsledky dosiahnuté algoritmami XGBClassifiar, MultinomialNB a RandomForest. Musíme však podotknúť, že najlepšiu stabilitu spomedzi nich mal aj v tomto prípade XGBoost klasifikátor, ktorý sa vo všetkých metrikách umiestnil v prvej trojici najlepších. Zvyšné dva algoritmy aspoň v jednej z metrík z tejto trojice vypadli. MultinomialNB dosiahol najlepšie výsledky pre metriku F1 skóre, ktorá by mala v kontexte markantného nevybalansovania dátovej sady podávať najlepšie informácie o výkonnosti jednotlivých algoritmov strojového učenia. Pri vyhodnocovaní úspešnosti tretieho najlepšieho algoritmu sme museli zobrať do úvahy širší kontext úlohy. Po detailnejšej analýze sme dospeli k záveru, že tretí najlepší algoritmus je náhodný les. Tento algoritmus strojového učenia bol v prvej trojici pri dvoch porovnávaných metrikách (accuracy a AUC). Ale v metrike F1 skóre, ktorá by sa mala najlepšie vysporiadať s nevyváženosťou dát, bol prekonaný rozhodovacím stromom. Pri konečnom zaradení do prvej trojice sme zobrali do úvahy aj

jeho výsledky v predošlej sade experimentov na vyváženej dátovej sade, rovnako aj fakt, že náhodný les by mal oproti rozhodovaciemu stromu lepšie generalizovať výsledky pri optimalizácii v ďalšom kroku. Finálne rozhodnutie pre vybratie náhodného lesa sme uskutočnili na základe jeho výsledku v metrike accuracy, kde sa iba jemu a XGBoost podarilo prekonať hranicu $acc = 0{,}9381$, ktorú dosahuje jednoduchý klasifikátor voliaci dominujúcu triedu.

Klasifikátor	Accuracy	F1 skóre	AUC
XGBClassifier	0,941	0,155	0,837
MultinomialNB	0,895	0,394	0,817
RandomForestClassifier	0,937	0,064	0,787

Tabuľka 11.2: Porovnanie základných klasifikátorov.

Tabuľka 11.2 zobrazuje výsledky základnej sady experimentov na dátovej sade SlovakCovid19 FN. Pre lepšiu prehľadnosť sme v tabuľke uviedli presné hodnoty iba pre prvú trojicu najúspešnejších algoritmov. V tabuľke sú uvedené hodnoty presnosti (accuracy), F1 skóre a AUC pre tieto tri klasifikátory, aby sme mohli porovnať ich výkon a vybrať najlepší pre ďalšie experimenty. Ostatné klasifikátory boli vynechané z tabuľky, pretože dosiahli horšie výsledky v porovnaní s tromi najúspešnejšími algoritmami.

Z výsledkov experimentu v tabuľke 11.2 je zjavné, že táto úloha pre celý dataset, ktorý je markantne nevyvážený, je len veľmi ťažko riešitelná. Keďže je dataset nevyvážený v pomere 93:7, úspešnosť v metrike accuracy menšia ako 0,9381 znamená, že aj klasifikátor, ktorý vyberie iba dominantnú triedu, je lepší. Túto úspešnosť dosiahli iba 2 klasifikátory, ale tie v metrike F1 skóre prepadli a nedosahujú dostatočné výsledky. Znamená to, že klasifikujú do triedy pravdivých článkov takmer celý dataset až na pár výnimiek.

Tento výsledok je ale očakávaný, vzhľadom na náročnosť detekcie fake-news a tiež vstupným dátam, ktoré sme mali v tomto experimente. Experimentom sme sa chceli oboznámiť s celým pôvodným datasetom a vytvoriť baseline hranice pre ďalšie experimenty, ktoré na výsledkoch týchto klasifikátorov a experimentov budú stavať. Sumarizácia výsledkov tejto sady experimentov je v prílohe A.4 v tabuľke A.1

11.3.2 Vplyv veľkosti trénovacej množiny na úspešnosť klasifikácie

V prvom kroku sa vstupná dátová sada rozdelila do troch množín (trénovacia, validačná a testovacia) v pomere 6:2:2. To znamená, že 60% pôvodných dát tvorí trénovaciu množinu, a zvyšných 40% je rovnomerne rozdelených medzi validačnú a testovaciu množinu. Toto rozdelenie sa uskutočnilo tak, aby sa zachovalo rozloženie tried v jednotlivých množinách.

V druhej fáze experimentu sme podobne ako pri zmenšenej verzii datasetu experimentovali s veľkosťou trénovacej množiny. Tento experiment bol vykonaný za rovnakých podmienok ako experiment v podkapitole 11.2.2 iba na SlovakCovid19 FN dátovej sade.

Obr. 11.12: Vplyv veľkosti trénovacej množiny na skúmané metriky.

Ilustrácia 11.12 obsahuje grafy, ktoré ukazujú, ako veľkosť trénovacieho datasetu ovplyvňuje úspešnosť klasifikátora. Rozšírenie trénovacej množiny vedie k zlepšeniu vývoja všetkých sledovaných metrík. Podobne ako v pri predošlom datasete sme zaznamenali prepad všetkých metrík, avšak tento prepad bol dočasný a nemal dlhodobý vplyv. Rovnako to bolo aj v predošlej verzii experimentu pre zmenšenú verziu datasetu, tento stav je pravdepodobne spôsobený rôznorodosťou príkladov v novej pridanej množine fake-news. Dá sa preto predpokladať, že oscilácie v metrikách sú spôsobené nevyváženou formou datasetu a rôznorodosťou príkladov (textov) v dátovej sade.

Ako záver experimentu môžeme brať zistenia, že pre dosiahnutie lepších výsledkov pri klasifikácii fake-news je nutné pracovať na vyváženosti a veľkosti trénovacích datasetov, ktoré sú pre slovenský jazyk momentálne nedostatočné. Tieto zistenia sú uskutočnené na základe experimentov, ktoré sme vykonali na konkrétnej dátovej sade. Je dôležité pripomenúť, že výsledky tohto experimentu sú silne ovplyvnené kvalitou týchto dát, na ktorých bol experiment vykonaný, a spôsobom, akým boli tieto dáta spracované v krížovej validácii. Pre definitívne potvrdenie týchto zistený je nutné vypracovať detailnejšiu analýzu problematiky a vykonanie ďalších testov na iných dátových sadách s cieľom potvrdiť alebo vyvrátiť tieto zistenia, ktoré boli vytvorené na základe tejto sady experimentov.

11.3.3 Optimalizácia hyperparametrov

Podobne ako v podkapitole 11.2.3, sme v tomto experimente skúmali optimalizáciu hyperparametrov pre najlepšie 3 metódy z predošlého experimentu. V rámci optimalizácie hyperparametrov sme zistili, že algoritmus XGBClassifier má najlepšie výsledky v tejto verzii datasetu aj v zmenšenej verzii. Preto sa v tejto časti zameriame na výsledky optimalizácie tohto algoritmu. V rámci experimentu sme využili TF-IDF vektorizáciu textu s rovnakými parametrami pre optimalizáciu hyperparameterov ako pri zmenšenej verzii datasetu v podkapitole 11.2.3.

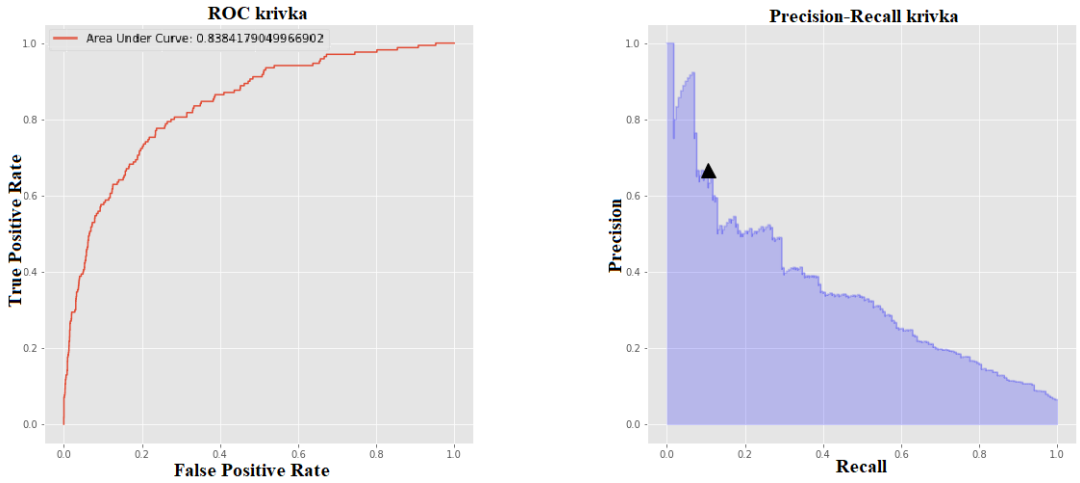
Rovnaká voľba parametrov bola zvolená s cieľom zabezpečiť správne porovnanie optimalizovaných algoritmov.

Na optimalizáciu parametrov sme použili Tree Parzen Estimator. Ako sme uviedli, dataset SlovakCovid19 FN bol rozdelený v pomere 6:2:2 na trénovaciu, validačnú a testovaciu množinu. Po vykonaní 15 iterácií optimalizačného algoritmu na kombinácii trénovacej a validačnej množiny sme identifikovali tieto optimálne hodnoty pre hyperparametre modelu:

```
params = {
    'learning_rate': 0.1,
    'max_depth': 5,
    'min_child_weight': 4.0,
    'n_estimators': 150,
    'normalize': 0,
    'scale': 1,
    'subsample': 0.8
}
```

Python implementácia 11.3: Opt. hyper. XGBoost.

Na základe optimalizácie parametrov sme opätovne otestovali XGBClassifier na nezávislej testovacej množine (SlovakCovid19 FN) s výsledkami $acc = 0{,}9407$ a $F1 = 0{,}5831$. Po optimalizácii parametrov accuracy modelu neklesla ale F1 skóre zlepšilo svoju hodnotu o 0,4, čo predstavuje zlepšenie o 40% v porovnaní s neoptimalizovanou verziou algoritmu.



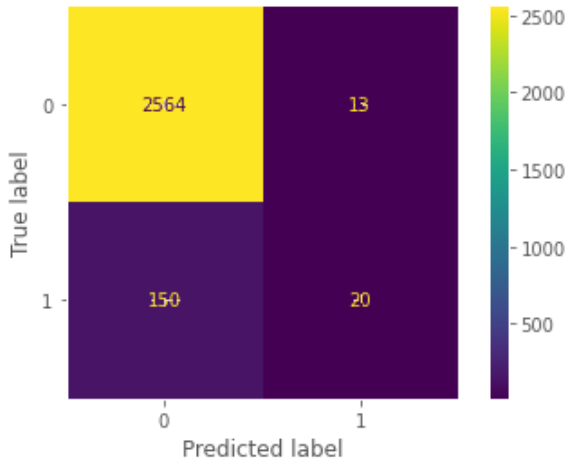
(a) ROC krivka.

(b) Precision-recall krivka.

Obr. 11.13: Baseline model: ROC a precision-recall krivka.

Ilustrácia 11.13a zobrazuje ROC krivku, ktorá vyjadruje vzájomný vzťah medzi TPR a FPR klasifikátora XGB na SlovakCovid19 FN datasete. Aj keď ROC krivka má pre celý slovenský dataset podobný priebeh ako pre jeho zmenšenú verziu, musíme tento výsledok brať s rezervou, keďže táto metrika má značné problémy s nevybalansovanými dátami. Hodnota AUC pre klasifikátor XGBoost na SlovakCovid19 FN datasete bola 0,8384, čo z kvalitatívneho hľadiska radí tento klasifikátor medzi veľmi dobré modely. Keďže táto hodnota úzko súvisí s ROC krivkou, je potrebné brať do úvahy, že hodnoty AUC a ROC krivky môžu byť ovplyvnené nevyváženosťou dát, a preto k nim treba pristupovať s rezervou.

Na vyhodnotenie úspešnosti XGBoost na SlovakCovid19 FN datasete sme taktiež použili precision-recall krivku. Bola zvolená z dôvodu nevyvážených tried v datasete. Obrázok 11.13b zobrazuje túto krivku pre náš klasifikátor na zvolenom datasete. Charakter precision-recall krivky pre klasifikátor XGBoost na našom datasete SlovakCovid19 FN je podobný ako u iných klasifikátorov používaných na podobné úlohy. Krivka sa začína pri hodnote precision blízkej 1, avšak postupne klesá s narastajúcou hodnotou recall. Klesanie je očakávané zo vzájomného vzťahu precision a recall. Z výrazného poklesu krivky hneď na začiatku je možné usúdiť, že existuje vysoký podiel falošne pozitívnych príkladov. Tento fakt môže byť spôsobený veľkou podobnosťou medzi pravdivými a falošnými článkami, ktorú základné modely strojového učenia nedokážu rozlíšiť. Tejto skutočnosti nepomáha nevyváženosťou datasetu, kde na jednu falošnú správu pripadá zhruba 15 pravdivých článkov. Z výsledkov predpokladáme, že v tomto kontexte distribúcie tried je pre klasifikátor ťažké nájsť špecifické vzory typické len pre fake-news, ktoré sa nevyskytujú v takto prevažujúcej pravdivej triede článkov a nastáva veľký prekryv vlastností týchto správ. Celkovo však precision-recall krivka poskytuje ucelený pohľad na kvalitu klasifikácie v rámci celej dátovej sady.

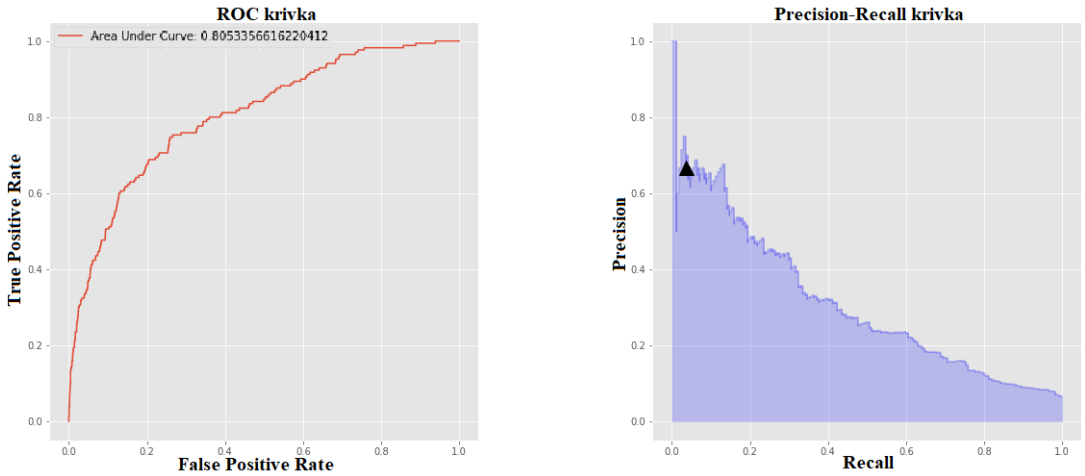


Obr. 11.14: Baseline model: konfúzna matica.

Podľa konfúznej matice klasifikátora XGBClassifier môžeme vidieť, že model správne klasifikuje približne každú siedmu falošnú správu, avšak na každú správne klasifikovanú falošnú správu zaradí nesprávne aj jeden pravdivý článok. Tento výsledok bude použitý ako baseline pre verziu celého slovenského datasetu v nasledujúcich experimentoch.

11.3.4 Ensemble metóda

Posledný experiment, ktorý sme vykonali v základnej sade na SlovakCovid19 FN datasete, bol zameraný na otestovanie ensemble metódy a jej prínosu pri klasifkácii fake-news v nevyváženom datasete. Zvolili sme rovnaký postup ako pri zmenšenej verzii datasetu, bol použitý XGBoost v kombinácii s algoritmom MultinomialNB a náhodný les. Trénovanie systému klasifikátorov prebehlo na trénovacej množine a následné vyhodnotenie na nezávislej testovacej množine, ktorá bola vytvorená na začiatku tohto experimentu.



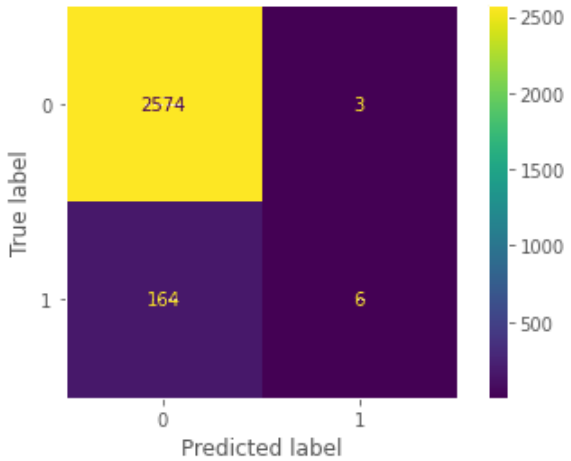
(a) ROC krivka.

(b) Precision-recall krivka.

Obr. 11.15: Ensemble model: ROC a precision-recall krivka.

V grafe 11.15a je zobrazená ROC krivka pre testovaný ensemble systém. Hoci sa na prvý pohľad zdá, že má dobrý charakter priebehu, podobný ako v prípade zmenšeného datasetu, musíme vziať do úvahy extrémnu nevyváženosť datasetu. Vzhľadom na extrémnu nevyváženosť datasetu nemôžeme považovať ROC krivku za spoľahlivý ukazovateľ úspešnosti testovaného ensemble systému. Preto jej hodnoty nezodpovedajú skutočnej úspešnosti systému.

Na druhej časti ilustrácie 11.15 na grafe 11.15b je zobrazená precision-recall krivka. Z jej prudkého poklesu môžeme dedukovať, že hneď od samotného začiatku spojením viacerých klasifikátorov pomocou soft metódy výberu predikujeme vyššie percento falošne pozitívnych prípadov.



Obr. 11.16: Ensemble model: konfúzna matica.

Predpokladáme, že zvýšená chybovosť systému je spôsobená faktom, že dve chybné klasifikácie prehlasovali jednu pozitívnu. Tento fakt indikuje, že klasifikátory chybne klasifikujú rôzne časti testovacej množiny a ich samostatná chybovosť sa v tejto skupine ešte znásobuje. V experimente sa ukázalo, že použitie ensemble metódy, ktorá kombinuje viacero jednoduchých klasifikátorov bez zohľadnenia nevyváženosti dátovej sady, vedie k nižšej presnosti modelov v porovnaní s použitím

jednotlivých klasifikátorov samostatne. Tento výsledok bol získaný na datasete s výraznou nevyváženosťou a platí pre tento konkrétny prípad.

Z obrázku 11.16 môžeme vidieť, že kombinovanie viacerých klasifikátorov neprodukuje uspokojivé výsledky. Každý jednoduchý klasifikátor robí chyby v rôznych častiach testovacej sady, ktoré sa prejavujú aj v konečnej klasifikácii. To vedie k tomu, že model predikuje väčšinu príkladov ako pravdivé, ale v skutočnosti má vysokú chybovosť. Tieto výsledky naznačujú, že použitie ensemble modelov nie je vždy vhodné a môže dokonca viesť k zhoršeniu presnosti klasifikácie.

11.3.5 Záver experimentu

Cieľom týchto experimentov bolo oboznámenie s celým slovenským datasetom, jeho následné spracovanie pomocou metód strojového učenia a vytvorenie baseline modelu. V druhom experimente tejto sady sme vytvorili baseline pomocou modelu XGBClassifier, ktorý dosiahol najlepšie výsledky z porovnávaných algoritmov. Úspešnosť baseline modelu bola stanovená na $acc = 0{,}9407$ a $F1 = 0{,}5831$.

V poslednom experimente tejto sady sme sa rovnako ako pri zmenšenej verzii datasetu snažili vytvoriť ensemble systém pre klasifikáciu fake-news. Musíme však zhodnotiť, že spojenie jednoduchých klasifikátorov bez nejakého vyváženia datasetu nemá cenu. Aspoň v kontexte nami testovanej dátovej sady. Ako výsledok tohto experimentu môžeme konštatovať zistenie, že ak chceme na klasifikáciu fake-news použiť jednoduchšie modely, je nutné zlepšiť kvalitu datasetu. Inak je nutné použiť komplexnejšie metódy strojového učenia.

11.4 Zhrnutie

Ako výsledok celej základnej sady experimentov môžeme brať nasledujúce zistenia: Pre kvalitnú detekciu falošných správ je nutné zlepšiť trénovací dataset, ktorý musí byť viac vybalansovaný a obsahovať väčšie množstvo dát. Skvalitnenie dátovej sady, ako je možné vidieť pri porovnaní dátových sád v tejto kapitole, vedie k veľkému zlepšeniu klasifikácie jednotlivými algoritmami. Na druhej strane je zmenšený dataset už na hrane svojou veľkosťou a bolo by vhodné, pri zachovaní vyváženosti jeho tried, rozšíriť ho na väčšiu množinu textov.

Ak bude splnená prvá podmienka skvalitnenia datasetu pre detekciu falošných správ, potom - ako ukázali experimenty v tejto sade - aj jednoduché algoritmy dokážu detegovať fake-news na slovenských textoch. Ukázalo sa tiež, že zvýšením komplexity modelov sa táto úspešnosť zvyšuje. Preto sme sa rozhodli v nasledujúcich experimentoch použiť komplexnejšie metódy strojového učenia a porovnať ich s dosiahnutými výsledkami. Očakávame zlepšenie úspešnosti, aj keď za cenu väčšej výpočtovej náročnosti.

12. Experimenty: Neurónové siete

V tejto sade experimentov sa zameriame na úspešnosť hlbokých neurónových sietí pri klasifikácii falošných správ v slovenskom jazyku, ktoré sme zadefinovali v kapitole 7. Konkrétne typy architektúr neurónových sietí, ktoré boli zvolené pre túto sadu experimentov, boli vybrané na základe ich nadpriemerných výsledkov pri klasifikácii textov. Odôvodnenie sme uviedli v kapitole 7, ktorá sa vzťahuje na neurónové siete. V prvom kroku budeme testovať výkonnosť jednotlivých neurónových sietí na každom zo slovenských datasetov a porovnáme ich úspešnosť na jednotlivých datasetoch. V druhom kroku porovnáme výkonnosť jednotlivých sietí medzi oboma dátovými sadami, aby sme videli, ako dobre sa každá sieť vysporiadala s každým z datasetov.

V rámci základnej sady experimentov sme rozdelili pôvodné dátové sady na trénovaciu, validačnú a testovaciu množinu v pomere 6:2:2. V tomto experimente sme trénovali model na rovnakej trénovacej množine ako v predchádzajúcom experimente. Následne sme vyhodnotili model na nezávislej testovacej množine, ktorá bola taktiež rovnaká ako v predchádzajúcom experimente.

Tréning neurónových sietí

Všetky neurónové siete boli trénované s týmito parametrami, ak nebude pre daný experiment špecifikované inak.

Parametre: Tréning trval 20/50/100 epoch (pre downsampled aj hodnota 500) s batch size 32, bol použitý optimizer Adam s predvolenou hodnotou learning rate. Ako metrika behu tréningu bola zvolená presnosť (accuracy) a chybová funkcia binárna cross entropia (binárny klasifikačný problém).

Evaluácia

Pre každý dataset sme vykonali viacero experimentov s rôznymi nastaveniami hyperparametrov, ktoré sme následne porovnali a vyhodnotili. Výsledky sme prezentovali v prehľadnej tabuľke a pomocou konfúznej matice s podrobným popisom výsledkov. Cieľom týchto experimentov je zistiť, ako dobre sa hlboké neurónové siete dokážu naučiť klasifikovať slovenské textové dáta a aký vplyv majú rôzne parametre na výkonnosť tejto siete. Týmto postupom sme chceli zabezpečiť korektnosť porovnania úspešnosti algoritmov na jednotlivých dátových sadách naprieč všetkými sadami experimentov.

Rekurentné neurónové siete

Prvá neurónová sieť, ktorú sme zvolili na základe jej nadpriemernej úspešnosti pri klasifikácii textových dát, bola LSTM rekurentná neurónová sieť. Tento typ siete je veľmi populárny pri klasifikácii textových dát s vysokou úspešnosťou, preto sme ju zaradili do tohto experimentu.

Architektúra siete

Konkrétne sme použili LSTM sieť s nasledujúcou architektúrou, ktorá bola implementovaná v jazyku Python pomocou knižnice *tensorflow*:

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm (LSTM)                  (None, 1024)              16486400
 batch_normalization (BatchN  (None, 1024)              4096
 ormalization)
 dense_0 (Dense)              (None, 512)               524800
 dense_1 (Dense)              (None, 256)               131328
 dense_2 (Dense)              (None, 128)               32896
 dense_3 (Dense)              (None, 64)                8256
 dense_4 (Dense)              (None, 32)                2080
 batch_normalization_1        (None, 32)                128
(BatchNormalization)
 dropout (Dropout)            (None, 32)                0
 dense_5 (Dense)              (None, 1)                 33
=================================================================
Total params: 17,190,017
Trainable params: 17,187,905
Non-trainable params: 2,112
_________________________________________________________________
```

Python implementácia 12.1: LSTM.

Konvolučná neurónová sieť

Ďalšou neurónovou sieťou, ktorú sme v experimentoch použili, bola konvolučná neurónová sieť. Tento typ siete sa často používa na spracovanie obrazových dát, ale ukázalo sa, že je tiež veľmi úspešná pri klasifikácii textových dát.

Architektúra siete

V rámci tohto experimentu sme použili implementáciu konvolučnej neurónovej siete (CNN) z kapitoly 7 o hlbokých neurónových sieťach, konkrétne z časti 7.4 o CNN. Implementáciu neurónovej siete sme naprogramovali v jazyku Python a použili sme túto architektúru:

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 spatial_dropout1d (SpatialD  (None, 3000, 1)           0
 ropout1D)
 conv1d (Conv1D)              (None, 2996, 128)         768
 dropout_1 (Dropout)          (None, 2996, 128)         0
 conv1d_1 (Conv1D)            (None, 2992, 64)          41024
 dropout_2 (Dropout)          (None, 2992, 64)          0
 max_pooling1d (MaxPooling1D)(None, 1496, 64)           0

 flatten (Flatten)            (None, 95744)             0
 dense_1 (Dense)              (None, 512)               49021440
 dense_2 (Dense)              (None, 256)               131328
 dropout_3 (Dropout)          (None, 256)               0
```

```
 dense_3 (Dense)              (None, 128)               32896
 dense_4 (Dense)              (None, 64)                8256
 dense_5 (Dense)              (None, 32)                2080
 batch_normalization (Batc    (None, 32)                128
 hNormalization)
 dropout_4 (Dropout)          (None, 32)                0
 dense_6 (Dense)              (None, 1)                 33
=================================================================
Total params: 49,237,953
Trainable params: 49,237,889
Non-trainable params: 64
_________________________________________________________________
```

Python implementácia 12.2: CNN.

Rekurentné konvolučné siete

Posledným typom neurónovej siete, ktorý sme použili v tomto experimente, je rekurentno-konvolučná neurónovú sieť. Táto sieť kombinuje prvky rekurentnej neurónovej siete a konvolučnej neurónovej siete. Tento typ siete sme bližšie popísali v kapitole 7 o hlbokých neurónových sieťach, konkrétne v sekcii 7.6 o RCNN sieťach, v ktorej je aj porovnanie s predošlými dvoma typmi neurónových sietí pri klasifikácii obrázkov. Tento typ siete ešte nie je veľmi otestovaný experimentmi, ale my sme sa ju rozhodli použiť, pretože v sebe kombinuje vlastnosti dvoch sietí, ktoré majú veľmi dobré výsledky pri klasifikácii textov. Spolu by teda mohli vytvoriť ešte silnejší systém pre klasifikáciu falošných správ.

Architektúra siete

Podobne ako pri predchádzajúcich dvoch typoch neurónových sietí sme sa pre implementáciu rozhodli použiť jazyk Python a knižnicu *tensorflow*.

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 spatial_dropout1d_1 (Spatia (None, 3000, 1)           0
 lDropout1D)
 conv1d_1 (Conv1D)           (None, 3000, 64)          384
 rcl (RCL)                   (None, 3000, 32)          16000
 dropout_1 (Dropout)         (None, 3000, 32)          0
 rcl_1 (RCL)                 (None, 3000, 32)          12928
 max_pooling1d_1 (MaxPooling (None, 1500, 32)          0
 1D)
 dropout_2 (Dropout)         (None, 1500, 32)          0
 rcl_2 (RCL)                 (None, 1500, 32)          12928
 dropout_3 (Dropout)         (None, 1500, 32)          0
 rcl_3 (RCL)                 (None, 1500, 32)          12928
 max_pooling1d_2 (MaxPooling (None, 94, 32)            0
 1D)
 flatten_1 (Flatten)         (None, 3008)              0
 dense_1 (Dense)             (None, 512)               1540608
 dense_2 (Dense)             (None, 256)               131328
 dropout_4 (Dropout)         (None, 256)               0
 dense_3 (Dense)             (None, 128)               32896
```

```
 dense_4 (Dense)              (None, 64)                8256
 dense_5 (Dense)              (None, 32)                2080
 batch_normalization_1 (Bat   (None, 32)                128
 chNormalization)
 dropout_5 (Dropout)          (None, 32)                0
 dense_7 (Dense)              (None, 1)                 33
=================================================================
Total params: 1,770,497
Trainable params: 1,769,409
Non-trainable params: 1,088
_________________________________________________________________
```

Python implementácia 12.3: RCNN.

12.1 Dataset DownSampled SlovakCovid19 FN

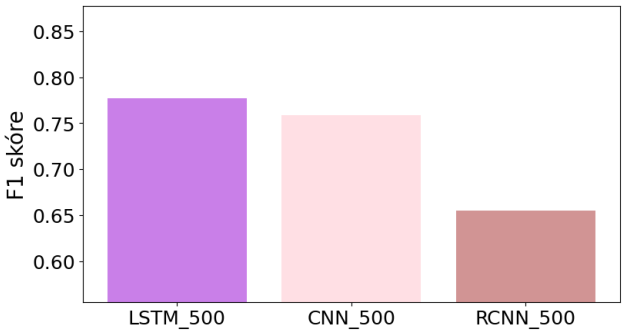
V tejto časti sme sa zamerali na klasifikáciu slovenských článkov o pandémii covidu-19 pomocou rôznych typov neurónových sietí. Pre tento experiment sme použili DownSampled SlovakCovid19 FN dataset, ktorý sme popísali v kapitole 9 o datasetoch. Dataset sme vytvorili zmenšením a vybalansovaním distribúcie klasifikačných tried v pôvodnom slovenskom datasete.

Cieľom tohto experimentu bolo zistiť, ktorý typ neurónovej siete bude mať najvyššiu úspešnosť pri klasifikácii týchto článkov a či bude schopný rozpoznať falošné správy o covide-19 v slovenských textových dátach.

Experiment	epochs	accuracy	F1 skóre
LSTM	20	0,7406	0,7369
LSTM	50	0,7235	0,7172
LSTM	100	0,7611	0,7611
LSTM	500	0,7782	0,7771
CNN	20	0,7235	0,7232
CNN	50	0,7474	0,7466
CNN	100	0,7372	0,7371
CNN	500	0,7611	0,7587
RCNN	20	0,4812	0,3565
RCNN	50	0,6485	0,6447
RCNN	100	0,6348	0,6347
RCNN	500	0,6553	0,6553

Tabuľka 12.1: Experiment: Neurónové siete - DownSampled SlovakCovid19 FN.

V tabuľke 12.1 sú uvedené výsledky pre všetky experimenty vykonané pomocou hlbokých neurónových sieti na dátovej sade DownSampled SlovakCovid19 FN. Tabuľka obsahuje hodnoty metrík accuracy a F1 skóre pre každý model neurónovej siete a počet epoch, ktorých kombináciu sme testovali. Metriky boli zvolené vzhľadom na výsledky predošlých experimentov, aby boli medzi sebou porovnateľné. Na obrázku 12.1 je zobrazený stĺpcový graf, ktorý porovnáva najlepšie modely pre každý typ neurónovej siete s ostatnými metódami v rámci tejto sady experimentov.



Obr. 12.1: Experiment: Neurónové siete - Downsampled SlovakCovid19 FN.

Ako sme uviedli v úvode tejto kapitoly, použili sme tri rôzne typy neurónových sietí: LSTM, CNN a RCNN, s rôznymi hodnotami epoch. Z hodnôt v tabuľke môžeme dedukovať, že najlepšie výsledky pri klasifikácii falošných správ o covide-19 v slovenských textových dátach sme dosiahli s LSTM sieťou s najväčším počtom epoch, t.j. rovných 500. Táto sieť dosiahla accuracy 0,7782 a F1 skóre 0,7771. Tesne za touto sieťou nasleduje CNN, rovnako s 500 epochami, ktorej úspešnosť bola accuracy 0,7611 a F1 skóre 0,7587. RCNN má výsledky o poznanie horšie. Najlepší výsledok podobne ako predošlé dve siete dosiahla s maximálnym počtom epoch. Avšak jej výsledná úspešnosť bola v porovnaní s ostatnými neurónovými sieťami, ktoré sme v tomto experimente testovali, o zhruba 0,1 (10%) nižšia pre obe sledované metriky. Úspešnosť RCNN bola nasledovná: accuracy 0,6553 a F1 skóre 0,6553. Úspešnosť jednotlivých sieti je možno vizuálne porovnať na ilustrácii 12.1, na ktorej je zobrazený stĺpcový graf metriky F1 skóre.

Všeobecne môžeme vidieť, že pre každý typ siete sme dosiahli najlepšie výsledky s väčším počtom epoch. Pri testovaní vyšších počtov iterácii neurónové siete začali trpieť problémom preučenia a ich výkonnosť klesla. LSTM a CNN siete sa javia z hľadiska úspešnosti ako lepšie v porovnaní s RCNN sieťou. Na druhej strane v prípade RCNN došlo k preučeniu až s vyšším počtom epoch, ktorý sa však iba vyrovnal predošlým dvom neurónovým sieťam.

Porovnali sme výsledky neurónových sietí s výsledkami baseline modelu, ktorý dosiahol presnosť $acc = 0{,}7065$ a hodnotu F1 skóre $= 0{,}7062$ na danej dátovej sade. Súčasná kombinácia LSTM a CNN sietí prekonala tento model o 0,07 (7%), čo predstavuje signifikantné zlepšenie v klasifikácii falošných správ. V porovnaní s pôvodným modelom sa výrazne zlepšila presnosť detekcie. Baseline model dokáže rozpoznať iba tri zo štyroch falošných správ, zatiaľ čo LSTM model má úspešnosť až päť správ zo šiestich. RCNN ako jediná neurónová sieť tento model neprekonala počas 500 epoch, ktoré sú uvedené vo výsledkoch. Aby dosiahla porovnateľnú úspešnosť s ostatnými neurónovými sieťami, musela byť táto sieť trénovaná s väčším počtom epoch. Aj keď to bolo výpočtovo náročné, stále nedokázala prekonať ostatné neurónové siete v presnosti klasifikácie. Tento fakt - predpokladáme - je spôsobený komplexnejšou konštrukciou tohto modelu, ktorý je kombináciou predošlých dvoch architektúr. Tento model na natrénovanie váh potrebuje rozsiahlejší dataset alebo vyšší počet epoch, čo sa ukázalo jeho zlepšovaním pomocou počtu iterácií, ktoré sú uvedené v tabuľke 12.1.

Z experimentu vyplývajú nasledujúce zistenia: Komplexnejšie modely strojového učenia dosahujú už vo svojom základnom nastavení lepšie výsledky ako

optimalizované klasifikátory z predošlých testov. Predpokladáme, že ďalší výskum a optimalizácia týchto modelov by mala viesť ešte k zvýšeniu úspešnosti neurónových sietí. Vzhľadom na to, že už tieto základné NN prekonali baseline model, posunieme sa v tejto práci v ďalších experimentoch k ešte komplexnejším modelom typu Transformer, u ktorých predpokladáme ešte markantnejšie zlepšenie úspešnosti. Z tohto experimentu môžeme tiež vyčítať, že slovenský dataset v takejto kvalite (vybalansovaný) nie je dostatočne rozsiahly a je potrebné jeho rozširovanie a zlepšovanie, aby mohla byť realizovaná automatická detekcia falošných správ v slovenských dátach.

12.2 Dataset SlovakCovid19 FN

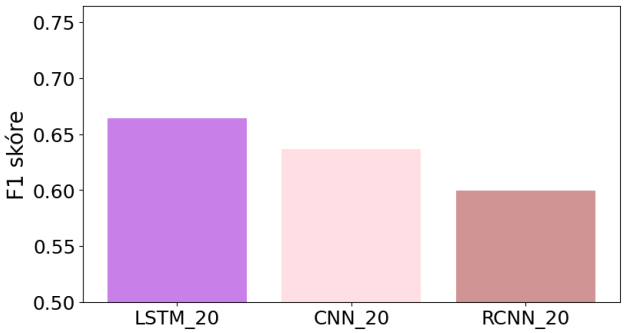
Rovnako ako v predošlom experimente aj v tomto sme sa zamerali na klasifikáciu falošných správ pomocou rôznych typov neurónových sietí na slovenských textových dátach. Pre experiment sme použili celý pôvodný slovenský dataset pre klasifikáciu fake-news, ktorý bol rozdelený na rovnakú trénovaciu, validačnú a testovaciu množinu ako v predošlom experimente 11.3.

Cieľom experimentu bolo rovnako porovnať úspešnosť rôznych typov neurónových sietí pri klasifikácii falošných správ na rozsiahlejších slovenských textových dátach a zistiť, ktorý typ siete dosahuje najlepšie výsledky. Druhým cieľom bolo porovnať vplyv kvality datasetu na úspešnosť neurónových sietí. Na to nám poslúžia výsledky z predošlého experimentu, ktorý bol vykonaný na vybalansovanej verzii tohto datasetu.

Experiment	epochs	accuracy	F1 skóre
LSTM	20	0,9370	0,6643
LSTM	50	0,9356	0,6248
LSTM	100	0,9327	0,6359
CNN	20	0,9388	0,6369
CNN	50	0,9327	0,5713
CNN	100	0,9403	0,6151
RCNN	20	0,8897	0,5998
RCNN	50	0,9276	0,5877
RCNN	100	0,9301	0,5454

Tabuľka 12.2: Experiment: Neurónové siete - SlovakCovid19 FN.

Vzhľadom na to, že sme vykonali podobný experiment na zmenšenom datasete, uvádzame výsledky v rovnakom formáte ako v predchádzajúcej časti. V tabuľke 12.2 sú uvedené výsledky pre všetky experimenty s hlbokými neurónovými sieťami na SlovakCovid19 FN datasete. Podobne na ilustrácii 12.2 je stĺpcový graf F1 skóre pre najlepšie modely, ktoré sme testovali na SlovakCovid19 FN dátovej sade. Z týchto výsledkov by sa mohlo na prvý pohľad zdať, že všetky testované modely dosiahli vysokú presnosť klasifikácie, ale v skutočnosti to skresľuje nevybalansovanosť datasetu. Úspešnosť $acc = 0{,}9381$ dosahuje aj klasifikátor, ktorý volí dominantnú triedu.



Obr. 12.2: Experiment: Neurónové siete - SlovakCovid19 FN.

Ak sa pozrieme na výsledky už s touto informáciou, potom dostatočne kvalitný výsledok dosahuje iba jeden typ neurónovej siete - CNN. Ak sa však pozrieme na F1 skóre, LSTM sieť dosahuje lepšie výsledky. To znamená, že deteguje viac falošných správ, ale za cenu vyššej falošnej pozitivity. V porovnaní s predchádzajúcimi experimentami na zmenšenom datasete sa ukázalo, že neurónové siete majú v základnom nastavení mierne lepšie výsledky ako baseline. Avšak rozdiel už nie je taký výrazný ako v predchádzajúcich experimentoch. Porovnanie výsledkov z tejto sady testov ukazuje, že komplexnejšie strojové učenie prináša zlepšenia, hoci nie také značné ako v prípade zmenšenej verzie datasetu. Tento výsledok potvrdzuje hypotézu, že zlepšovanie klasifikácie falošných správ závisí od zlepšovania modelov aj datasetu. Preto sme sa rozhodli vytvoriť DownSampled SlovakCovid19 FN dataset, ktorý sme predstavili v predošlých experimentoch v kapitole 9.

12.3 Zhrnutie

Za výsledky tejto sady experimentov považujeme potvrdenie hypotéz, ktoré sú zo zahraničných štúdií, ale aj z našich predošlých experimentov. Použitie komplexnejších, t.j. zložitejších metód strojového učenia vedie k zlepšeniu úspešnosti predikcie falošných správ. Tento poznatok a zistenie sme predpokladali, preto v ďalších experimentoch budeme pokračovať v tomto trende a použijeme ešte komplexnejšie metódy detekcie fake-news. V tomto experimente sa ukázala ešte viac dôležitosť kvalitných dát oproti základnej sade testov. Tento test zdôraznil dôležitosť kvalitných dát pre ich využitie pre automatickú detekciu fake-news v slovenskom jazyku. Ukázali sme, že slovenský dataset v takejto kvalite nie je dostatočný a je potrebné jeho vybalansovanie a rozširovanie, ak chceme pokročiť v riešení tejto problematiky. Vybalansovanie nie je ten najzávažnejší problém, ale spolu s malým množstvom falošných správ spôsobuje problémy pri učení klasifikátorov. Preto je potrebné vybudovanie kvalitnejšej dátovej sady, aby sa následne zlepšila automatická detekcia falošných správ.

Treba podotknúť, že výsledky experimentov sú ovplyvnené zvolenou metodikou: downsampling dátovej sady, ktorá ukázala na začiatku lepšie výsledky celým spektrom algoritmov. Na druhej strane sme zvažovali aj metódu upsamplingu, ale na základe výsledkov sme ju nakoniec vylúčili. Napriek tomu si uvedomujeme, že v novej téme detekcie falošných správ by bolo vhodné vykonať ďalšiu hlbšiu analýzu spracovania dát, čo by pravdepodobne prinieslo ďalšie poznatky.

13. Experimenty: Transformers

V tomto experimente porovnáme modely založené na architektúre Transformer pri klasifikácii falošných správ v slovenskom jazyku, ktoré sme zadefinovali v kapitole 8. V prvom experimente budeme testovať výkonnosť rôznych architektúr typu Transformer na každej zo slovenských dátových sád a porovnáme ich úspešnosť. V tomto kroku prebehne fine-tuning modelov na slovenskej trénovacej množine a následne ich úspešnosť vyhodnotíme na testovacej množine, ktorá prislúcha k danej trénovacej množine. Druhý experiment je založený na kombinácii fine-tuningu modelov na anglických dátových sadách týkajúcich sa rôznych tém falošných správ a následnej evaluácie na rovnakej slovenskej testovacej sade, ktorá bola použitá v predchádzajúcich experimentoch. V poslednom experimente otestujem spojenie slovenského trénovacieho datasetu s anglickými datasetmi. V tomto experimente najskôr prebehne fine-tuning na anglickom datasete a následne v ďalšom kroku na slovenskom trénovacom datasete. Evaluácia úspešnosti modelov prebehne na rovnakej slovenskej testovacej sade dát, akú sme použili v predchádzajúcom prípade. Výsledky jednotlivých experimentov uvedieme v prehľadnej tabuľke a v stĺpcovom grafe pre najúspšnejšie modely.

Tokenizácia vstupných textov prebehla s nasledovným nastavením parametrov:

```
# Tokenizer parametre
   add_special_tokens = True
   max_length = 512
    pad_to_max_length = True
   return_attention_mask = True
```

Python implementácia 13.1: Tokenizer: parametre

Parametre fine-tuningu sú pre jednotlivé modely nastavené na rovnakú hodnotu s cieľom porovnať výkonnosť týchto modelov medzi sebou.

```
# Model parametre
  batch_size=8
  epochs=20
  loss = SparseCategoricalCrossentropy(from_logits=True)
  metric = SparseCategoricalAccuracy('accuracy')
  optimizer = Adam(learning_rate=2e-5,epsilon=1e-08)
```

Python implementácia 13.2: Model: parametre

Všetky modely v tejto sade experimentov sú implementované v jazyku Python pomocou knižnice *tensorflow*. Konkrétne typy architektúr Transformer, ktoré boli zvolené pre túto sadu experimentov, sú nasledovné:

BERT

Ako prvý typ architektúry Transformer sme si zvolili model BERT. V rámci testov je zvolená anglická verzia tohto modelu, preto je nutné slovenské dátové sady pre tento model automaticky preložiť do anglického jazyka. Automatický preklad sme vykonali pomocou knižnice *googletrans* v jazyku Python. Týmto krokom chceme overiť a otestovať prenositeľnosť poznávacích znakov fake-news medzi jazykmi.

Architektúra

Konkrétne bola zvolená base verzia architektúry modelu BERT. Tento typ architektúry Transformer sme tiež zvolili z dôvodu porovnania s modelom SlovakBERT, ktorý má rovnaký počet parametrov.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bert (TFBertMainLayer)       multiple                  109482240
dropout (Dropout)            multiple                  0
classifier (Dense)           multiple                  1538
=================================================================
Total params: 109,483,778
Trainable params: 109,483,778
Non-trainable params: 0
_________________________________________________________________
```

Python implementácia 13.3: BERT-base-uncased

mBERT

Ďalším modelom v našej sade je multilingválny BERT, ktorý sme vybrali z dôvodu možnosti vyhnúť sa nutnosti prekladu textov pri použití predténovaných cudzojazyčných modelov s architektúrou Transformer.

V rámci testovania tohto modelu budeme pracovať s textami, ktoré sú preložené, a aj s tými, ktoré sú v pôvodnej forme. Je tiež dôležité poznamenať, že multilingválny BERT disponuje širokou škálou jazykových zdrojov v porovnaní s jednojazyčnými modelmi. To nám umožní pracovať s textami v rôznych jazykoch.

Architektúra

Podobne ako v predošlom prípade modelu BERT sme zvolili base verziu, z dôvodu korektnosti porovnania so slovenskou verziou modelu BERT.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bert (TFBertMainLayer)       multiple                  109482240
dropout (Dropout)            multiple                  0
classifier (Dense)           multiple                  1538
=================================================================
Total params: 109,483,778
Trainable params: 109,483,778
Non-trainable params: 0
_________________________________________________________________
```

Python implementácia 13.4: mBERT-base-uncased

RoBERTa

RoBERTa je ďalšou verziou modelu založeného na architektúre Transformer. Tento model vznikol na základe rozšírenia pôvodnej verzie BERT o nové tréno-

vacie techniky, ktoré umožnili vylepšiť jeho výkon v rôznych jazykových adaptáciách. RoBERTa model dosahuje lepšie výsledky v porovnaní s BERT-om naprieč rôznymi jazykovými verziami a môže byť použitý na širokú škálu jazykových úloh.

Architektúra

RoBERTa-base je založená na rovnakej architektúre ako BERT-base. Avšak v porovnaní s BERT-om má RoBERTa-base viac trénovacích parametrov, čím sa dosahuje lepší výkon.

RoBERTa-base má viac než 120 miliónov trénovacích parametrov a dosahuje výrazne lepšie výsledky v rôznych jazykových úlohách ako BERT-base. Táto verzia RoBERTa bola trénovaná na viac ako 160 GB textových dát v rôznych jazykoch a dosahuje lepší výkon aj v prípade, že trénovacie dáta sú nevyvážené.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
roberta (TFRobertaMainLayer) multiple                  124055040
classifier (TFRobertaClassi  multiple                  592130
ficationHead)
=================================================================
Total params: 124,647,170
Trainable params: 124,647,170
Non-trainable params: 0
_________________________________________________________________
```

Python implementácia 13.5: RoBERTa- base

XLM-RoBERTa

XLM-RoBERTa je založená rovnako ako predošlé modely na architektúre Transformer, ale s niektorými vylepšeniami, ktoré umožňujú prekladať text medzi rôznymi jazykmi. XLM-RoBERTa je teda rozšírením RoBERTa o možnosť práce s viacerými jazykmi súčasne. XLM-RoBERTa má viac ako 277 miliónov trénovacích parametrov, čo je oveľa viac ako RoBERTa-base. Táto verzia bola trénovaná na ešte väčšom objeme textových dát z viacerých jazykov ako predošlé modely. Preto je schopná dosiahnuť ešte lepšie výsledky v multilingválnych úlohách.

Architektúra

```
_________________________________________________________________
Layer (type)                   Output Shape            Param #
=================================================================
roberta (TFXLMRobertaMainLayer)  multiple           277453056
classifier (TFXLMRobertaCla      multiple            592130
ssificationHead)
=================================================================
Total params: 278,045,186
Trainable params: 278,045,186
Non-trainable params: 0
_________________________________________________________________
```

Python implementácia 13.6: XLM-RoBERTa-base

SlovakBERT

SlovakBERT je posledným modelom v tejto sade experimentov, ktorý je založený na architektúre Transformer. Tento model má potenciál výrazne zjednodušiť a zrýchliť prácu s jazykovými dátami v slovenskom jazyku a pomôcť tak aj vývoju jazykových technológií v Slovenskej republike.

Vzhľadom na tento potenciál sme sa ho rozhodli zaradiť do sady experimentov a porovnať jeho úspešnosť s cudzojazyčnými verziami prípadne multilingválnymi. Cieľom je otestovať, či modely pre konkrétne jazyky aj v dnešnej dobe multilingválnych modelov (prípadne cudzojazyčných modelov v spojení s automatickým prekladom pre jazyky, ktoré nepodporujú multilingválne modely), poskytujú konkurenčnú výhodu pri spracovaní prirodzeného jazyka. Konkrétne v našom prípade pri detekcii falošných správ v slovenskom jazyku.

Architektúra

Architektúra SlovakBERT modelu je rovnako ako v predošlých modeloch rozšírená iba o klasifikačnú vrstvu.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bert (TFBertMainLayer)       multiple                  109482240
dropout (Dropout)            multiple                  0
classifier (Dense)           multiple                  1538
=================================================================
Total params: 109,483,778
Trainable params: 109,483,778
Non-trainable params: 0
_________________________________________________________________
```

Python implementácia 13.7: SlovakBERT

13.1 DownSampled SlovakCovid19 FN

Fine-tuning modelov a ich evaluácia prebehne na DownSampled SlovakCovid19 FN datasete, ktorý sme vytvorili ako súčasť tejto práce. Nami vytvorený dataset v tomto experimente rozdelíme v pomere 6:2:2 na trénovaciu, validačnú a testovaciu množinu.

Následne pre anglický model BERT je nutné každú z týchto množín preložiť do anglického jazyka, na čo sme využili automatický preklad v Pythone pomocou knižnice *googletrans*. Model mBERT je multilingválna verzia BERT-u, ktorú v tomto experimente otestujeme na pôvodných slovenských dátach aj na preloženej verzii. Nasledujúce dva modely založené na architektúre Transformer, ktoré sme predstavili v kapitole 8 a ich implementáciu sme uviedli v úvode tejto kapitoly, sú RoBERTa a XLM-RoBERTa. Podobne ako pri predošlých dvoch zahraničných modeloch ich otestujeme na preloženej verzii slovenského zmenšeného datasetu. V prípade multilingválnej verzie RoBERTa na oboch variantoch ako pri mBERT. Posledným modelom, ktorý sa zúčastní tohto experimentu, je prvý slovenský model založený na architektúre Transformer, SlovakBERT.

Pre multilingválny mBERT/XLM-RoBERTa použijeme fine-tuning v dvoch samostatných verziách. Označme prvú verziu mBERT/XLM-RoBERTa, ktorá bude prispôsobená na datasete v slovenskom jazyku. Potom mBERT*/XLM-RoBERTa* je verzia modelu prispôsobená na preloženom slovenskom datasete do anglického jazyka. O automatický preklad do angličtiny sa postará Python knižnica *googletrans*.

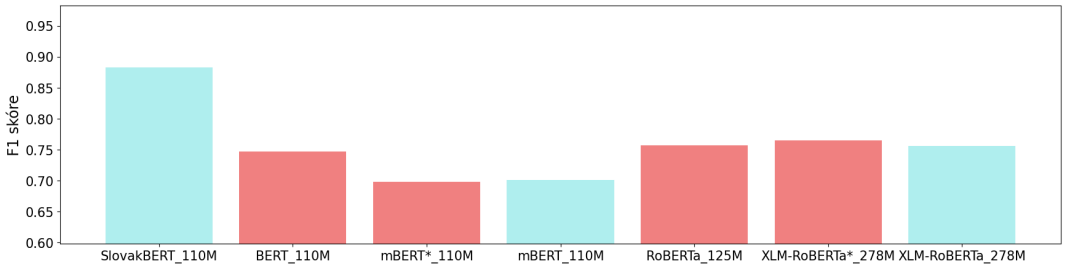
Výsledky

Výsledky pre modely založené na architektúre Transformer sú uvedené v tabuľke 13.1 a na ilustrácii 13.1, na ktorej sú jednotlivé modely zoradené podľa počtu parametrov a rovnaká farba znamená, že boli trénované a testované v rovnakom jazyku (modely testované v angličtine majú červenú farbu, v slovenčine modrú). Hviezdička pri multilingválnych modeloch mBERT*/XLM-RoBERTa* značí, že tento experiment bol vykonaný na preloženom slovenskom datasete.

Experiment	epochs	precision	recall	accuracy	f1-score
Dátová sada preložená do angličtiny.					
BERT	11	0,7836	0,7143	0,7577	0,7473
mBERT*	10	0,8108	0,6122	0,7338	0,6977
RoBERTa	15	0,7730	0,7415	0,7611	0,7569
XLM-RoBERTa*	16	0,7899	0,7415	0,7713	0,7649
Dátová sada v slovenčine.					
mBERT	13	0,7661	0,6463	0,7235	0,7011
XLM-RoBERTa	20	0,7868	0,7279	0,7645	0,7562
SlovakBERT	14	0,8491	0,9184	0,8771	0,8824

Tabuľka 13.1: Experiment: Transformer Downsampled SlovakCovid19 FN.

Výsledky v tabuľke 13.1 pre jednotlivé modely sú prezentované pomocou metrík precision, recall a F1-score, ktoré sú bežne používané na hodnotenie výkonu klasifikačných modelov. Z tabuľky vyplýva, že najlepším modelom pre tento dataset je SlovakBERT, ktorý dosiahol najvyššie hodnoty pre všetky sledované metriky. Tento model dosiahol precision 0,8491, recall 0,9184 a F1-score 0,8824. Na druhom mieste sa umiestnil model XLM-RoBERTa*, ktorý dosiahol precision 0,7899, recall 0,7415 a F1-score 0,7649. Tretie miesto patrí modelu RoBERTa s hodnotou F1-score 0,7569.



Obr. 13.1: Experiment: F1 skóre modelov založených na architektúre Transformer na Downsampled SlovakCovid19 FN.

Celkovo možno zhrnúť, že pre tento dataset dosiahol SlovakBERT najlepšie výsledky a je preto najlepším modelom pre klasifikáciu textu týkajúceho sa covidu-19 v slovenskom jazyku. Ak sa pozrieme na výsledky jednotlivých modelov bližšie, prichádzame k novým zisteniam. V nasledujúcej časti sa pozrieme na výsledky týchto modelov bližšie a zhodnotíme úspešnosť aj s odôvodnením.

Na prvom mieste sa umiestnil SlovakBERT, veľký lingvistický model určený pre slovenský jazyk. Aj napriek menšiemu počtu parametrov dokázal SlovakBERT zvíťaziť nad modelmi RoBERTa a jeho multilingválnou verziou, vďaka tomu, že je prvým modelom s architektúrou Transformer určeným pre slovenský jazyk a využil plne pochopenie slovenských viet oproti konkurencii.

Týmto umiestnením sa potvrdilo, že aj v dnešnej dobe sú špeciálne modely pre jednotlivé jazyky potrebné a umožňujú kvalitnejšie predikcie v danom jazyku. Tento fakt je umocnený ešte kvalitou dát, s ktorými pri riešení daného lingvistického problému pracujeme. V tomto prípade ide o malý slovenský dataset, v ktorom zahraničné modely nemôžu prekonať fine-tuning slovenského modelu pre slovenský jazyk, ktorý je už v základe pripravený na prácu s textami v tomto jazyku, a tak si vie lepšie poradiť s jednotlivými úskaliami slovenského jazyka pri riešení rôznych lingvistických úloh.

Za týmto modelom sa umiestnili všetky tri verzie RoBERTa modelu, ktorých rozdiely boli minimálne. Toto umiestnenie oproti modelu BERT/mBERT nás neprekvapuje. Je spôsobené väčším počtom parametrov pre tieto modely oproti základnej verzii BERT a väčšou množinou dát, na ktorej boli predtrénované. Tieto tri modely dosahujú skoro totožné výsledky. Ukazuje sa, že pre tento multilingválny model nehrá rolu, v ktorom z podporovaných jazykov je dataset. Taktiež sa ukazuje, že rozsiahlejší model kompenzuje stratu v preklade. Táto strata je však markantná oproti natívnej verzii SlovakBERT.

Modely BERT sa umiestnili na konci rebríčka. Konkrétne anglická verzia BERT, hneď za RoBERTa modelmi. Táto strata je spôsobená (ako sme už spomenuli) menším počtom parametrov modelu ako aj nutnosťou prekladu textov. Tento proces môže viesť k strate niektorých informácií obsiahnutých v pôvodnom texte alebo k ich nahradeniu sémanticky podobnými verziami. V kontexte detekcie falošných správ môže dôjsť k stratám malých odchýlok, ktoré odlišujú pravdivé správy od falošných.

Vyššia úspešnosť modelu BERT je spôsobená tým, že multilingválna verzia mBERT resp. mBERT* je natrénovaná na dátach, z ktorých napríklad tvorili články v slovenskom jazyku približne jedno percento. Pôvodný BERT je natrénovaný čisto na anglických textoch a nástroje na preklad textov sú v dnešnej dobe zväčša natrénovane na duálnych korpusoch, nie na multilingválnych dátach, nevedia teda prekladať medzi ľubovoľnou dvojicou jazykov, ale zas obmedzenie prekladu na dva jazyky zvyšuje jeho presnosť.

Na konci nášho rebríčka sa teda umiestnil multilingválny mBERT. Tento model dopláca na svoju robustnosť, hoci vie pracovať s textovými dátami vo viac ako 100 jazykoch. Dvojica miniexperimentov, pozostávajúca zo súťaže medzi dvoma verziami multiligualného BERT-u, potvrdila, že prekladom sa strácajú niektoré vlastnosti pôvodného textu. Verzia mBERT*, ktorá bola použitá s preloženým slovenským datasetom (rovnako ako BERT stráca oproti SlovakBERT), má nižšiu úspešnosť ako mBERT, ktorý bol použitý priamo na slovenskom texte. Tento výsledok iba potvrdil, že použitie textových dát v pôvodnom jazyku zvyšuje

úspešnosť modelu. V pôvodnom jazyku textu sú všetky odchýlky, ktoré odlišujú fake-news od pravdivých správ, kdežto pri preklade sa niektoré z týchto informácií z textu vytratia a jemné rozdiely medzi niektorými fake-news a pravdivými správami sa zotrú.

Výsledky tohto experimentu ukazujú, že v prípade, že pre daný jazyk neexistuje verzia modelu BERT, je možné použiť multilingválne alebo zahraničné verzie veľkých lingvistických modelov. V prípade, že je daný jazyk podporovaný niektorým z multilingválnych modelov, je možné ich použiť priamo. Avšak v prípade, že je nutný preklad textov, môže byť tento proces náročný a aj úspešnosť môže byť nižšia v porovnaní s natívnymi modelmi.

13.2 Fine-tuning na anglických datasetoch

Táto skupina experimentov je zameraná na fine-tuning modelov na jednom z anglických datasetov a iba následná evaluácia prebehne na slovenskom testovacom datasete, t.j. na identických článkoch ako v predošlom experimente. Pre modely BERT, RoBERTa, mBERT* a XLM-RoBERTa* na preloženej verzii pomocou automatického prekladu v Pythone knižnicou *googletrans*. Evaluácia pre mBERT respektíve pre XLM-RoBERTa prebehne na pôvodnom neprelože-nom datasete. Pre slovenskú verziu SlovakBERT modelu sú anglické datasety LIAR a COVID19 FN preložené do slovenského jazyka.

13.2.1 LIAR

V tomto čiastkovom experimente prebehol fine-tuning modelov na anglickom datasete LIAR z kapitoly 9.1.1. Cieľom tohto experimentu je overiť hypotézu vplyvu témy fake-news na výslednú detekciu.

Experiment	epochs	precision	recall	accuracy	f1-score
Slovenská dátová sada preložená do angličtiny.					
BERT	12	0,5017	1,0000	0,5017	0,6682
mBERT*	14	0,5017	1,0000	0,5017	0,6682
RoBERTa	4	0,5368	0,8435	0,5563	0,6561
XLM-RoBERTa*	20	0,5017	1,0000	0,5017	0,6682
Dátové sady v pôvodných jazykoch.					
mBERT	14	0,5017	1,0000	0,5017	0,6682
XLM-RoBERTa	20	0,5017	1,0000	0,5017	0,6682
Dátová sada LIAR preložená do slovenčiny.					
SlovakBERT	20	0,5017	1,0000	0,5017	0,6682

Tabuľka 13.2: Experiment: Fine-tuning na anglickom datasete LIAR.

Ak sa pozrieme na výsledky v tabuľke 13.2 a porovnáme ich s výsledkami z prvého experimentu uvedenými v tabuľke 13.1, zistíme, že fine-tuning iba na anglickom datasete LIAR bez fine-tuningu na slovenských dátach nepostačuje. A takto natrénované modely sú slabšie ako najhorší model v predošlom experimente 13.1. Takmer všetky modely predikovali každú správu ako falošnú.

V porovnaní s predošlým experimentom 13.1, kde sme porovnávali model RoBERTa s rovnakou architektúrou, sme zistili, že v aktuálnom experimente dosiahol tento model nižšiu úspešnosť o 0,09 (9%). Zhoršenie výkonu nie je spôsobené jazykom, v ktorom sú textové dáta, ako sme ukázali v predošlom experimente, kde sme preukázali, že pri rovnakej téme dát je možné chybu prekladu takmer zanedbať. Preto môžeme text považovať za napísaný v preloženom jazyku. Vzhľadom na výsledky v tabuľke 13.1, kde sme zistili, že pre RoBERTa nezáleží na jazyku dát, musí zvýšenie chyby v aktuálnom experimente súvisieť s odlišnou témou článkov v trénovacom a testovacom datasete. Multilingválne modely sú na tom ešte horšie. Tie sa vo fáze fine-tuningu adaptovali nielen na tému dát, ale aj na anglický jazyk. Následkom toho neboli na slovenskom testovacom datasete schopné správnej klasifikácie. Nezhoda témy sa potvrdila najvýraznejšie na slovenskom modeli SlovakBERT, ktorého úspešnosť klesla o viac ako 0,2 (20%). Horšie je, že model predikuje prakticky celú množinu ako falošnú a teda zhoršenie je rovné klasifikátoru, ktorý predikuje iba falošné správy.

Na tomto experimente sa ukázalo, že fake-news s rôznou témou majú iný textový charakter pre nami vybrané dátové sady. Ukazuje sa, že sa model nedá natrénovať iba na jednej doméne fake-news a zovšeobecniť ho na detekciu v akejkoľvek téme. Na druhej strane, výsledky popísané v tejto časti boli získané prostredníctvom experimentov, ktoré sme vykonali na nami zvolených dátových sadách. Pre overenie poznatkov v tejto novej oblasti detekcie falošných správ by bolo vhodné uskutočniť ďalšiu analýzu na ďalších dátových sadách.

13.2.2 COVID19 FN

V tomto teste prebehol fine-tuning modelov na anglickom datasete COVID19 FN z kapitoly 9.1.2. Keďže sme v predošlom experimente ukázali, že téma článkov má obrovský vplyv na štruktúru textov a teda aj na následnú klasifikáciu. V nadväznosti na to, sme sa rozhodli týmto experimentom overiť, či falošne správy ohľadom rovnakej témy z rôznych kútov sveta možno spájať do spoločných datasetov, pretože majú spoločný charakter nosnej správy.

Experiment	epochs	precision	recall	accuracy	f1-score
Slovenská dátová sada preložená do angličtiny.					
BERT	13	0,5296	0,9728	0,5529	0,6859
mBERT*	15	0,5603	0,8844	0,5939	0,6860
RoBERTa	6	0,5477	0,8979	0,5768	0,6804
XLM-RoBERTa*	9	0,5524	0,9320	0,5870	0,6937
Dátové sady v pôvodných jazykoch.					
mBERT	15	0,5017	1,000	0,5017	0,6682
XLM-RoBERTa	9	0,5017	1,0000	0,5017	0,6682
Dátová sada COVID19 FN preložená do slovenčiny.					
SlovakBERT	8	0,5991	0,8844	0,6451	0,7143

Tabuľka 13.3: Experiment: Fine-tuning na anglickom datasete COVID19 FN.

Ak porovnáme výsledky dosiahnuté modelmi, na ktorých prebehol fine-tuning pomocou COVID19 FN v tomto experimente, s výsledkami tých, ktoré boli pris-

pôsobené v predošlom experimente pomocou LIAR datasetu, tak sú tieto výsledky pre väčšinu modelov podobné. Ukazuje sa, že každá krajina má určité špecifiká pri písaní fake-news. Rovnaká téma článkov pomohla týmto modelom zvýšiť úspešnosť, ale iba miernym spôsobom. Predpokladáme, že mierne zlepšenie v rámci rovnakej témy fake-news naprieč jazykmi je spôsobené tým, že v rámci rovnakej témy sa určité fake-news šíria v svojich jazykových mutáciách naprieč rôznymi jazykmi. Takéto fake-news musia byť teda všeobecne platné kdekoľvek na svete a nebudú sa vzťahovať na lokálnu situáciu. Napríklad, pre pandémiu covidu-19 to sú falošné správy o vakcinácii, úmrtnosti a liečbe.

Týmto experimentom sme ukázali, že fine-tuning modelu na detekciu fake-news iba na datasete, ktorý neobsahuje špecifiká danej lokálnej komunity, je nepostačujúci. Modely, ktoré sú prispôsobené priamo na dátach z tejto komunity (oblasti), majú výsledky kvalitnejšie a k tomu odpadá aj nutnosť prekladu.

13.3 Spojenie datasetov

V poslednom experimente plynulo nadviažeme na predošlé výsledky. Skočili sme fine-tuningom modelov SlovakBERT, BERT, mBERT, mBERT*, respektíve RoBERTa, XLM-RoBERTa a XLM-RoBERTa* na anglických datasetoch LIAR a COVID19 FN. Takto prispôsobené modely pre úlohu fake-news sme použili na klasifikáciu fake-news, ale zo Slovenska, aby sme ukázali, že kontext danej správy sa prekladom nemení. Tento experiment však môže pokračovať aj ďalším krokom, a tým je opätovný fine-tuning, ale tentokrát na slovenskom trénovacom datasete. Až po fine-tuningu na tomto datasete uskutočníme evaluáciu na nezávislej slovenskej testovacej množine.

13.3.1 LIAR

Fine-tuning u modelov v tejto časti najskôr prebehol na LIAR datasete, následne opätovný fine-tuning ale už na slovenskom trénovacom datasete. Výsledky po evaluácii na testovacom datasete sú uvedené v tabuľke 13.4.

Experiment	epochs	precision	recall	accuracy	f1-score
Slovenská dátová sada preložená do angličtiny.					
BERT	2	0,7698	0,7279	0,7543	0,7483
mBERT*	6	0,8333	0,0340	0,5119	0,0653
RoBERTa	13	0,8000	0,6803	0,7543	0,7353
XLM-RoBERTa*	20	0,5017	1,0000	0,5017	0,6682
Dátové sady v pôvodných jazykoch.					
mBERT	20	-	0,0000	0,4983	-
XLM-RoBERTa	20	0,5017	1,0000	0,5017	0,6682
Dátová sada LIAR preložená do slovenčiny.					
SlovakBERT	20	-	0,0000	0,4983	-

Tabuľka 13.4: Experiment: Fine-tuning (LIAR)

Pre modely mBERT* a XLM-RoBERTa* prebehne fine-tuning najskôr na anglickom datasete LIAR a následne na preloženom trénovacom slovenskom datasete

DownSampled SlovakCovid19 FN. Taktiež evaluácia prebehla na preloženej testovacej slovenskej sade. Pre SlovakBERT sme LIAR dataset preložili do slovenského jazyka, podobne ako v predošlom teste.

Oproti fine-tuningu iba na LIAR datasete a následnom vyhodnotení už na slovenskom testovacom datasete došlo k zlepšeniu v anglických modeloch, respektíve multilingválnych modeloch na preložených dátach. Toto zlepšenie bolo očakávané, keďže sme modely ešte dotrénovali na slovenských trénovacích dátach. Neprekonali však modely, ktoré boli trénované iba na slovenskom datasete. Táto skutočnosť je znovu spôsobená rôznou témou článkov, lebo LIAR správy skresľujú výsledok a prinášajú šum do modelu. Tento šum má za následok vyššiu falošnú pozitivitu.

V tomto experimente sme použili multilingválne modely, ktoré sme nezaťažili prekladom slovenského datasetu do angličtiny. Tieto modely sme najskôr fine-tunovali na anglickom LIAR datasete a následne sme ich trénovali na slovenskom datasete. Avšak v porovnaní s predchádzajúcim experimentom sme zaznamenali ešte väčšie zhoršenie úspešnosti modelov na nepreloženom slovenskom datasete. Zhoršenie je spôsobené spôsobom fine-tuningu, kde si v prvom kroku najskôr modely prispôsobili svoje váhy a rozhodnutia anglickému jazyku. V druhom kroku už boli ale donútené pracovať so slovenskou sadou, na ktorej prebehlo aj vyhodnotenie. Týmto striedaním jazykov sa zanieslo do fine-tuningu ešte väčšie množstvo šumu. Ako sa ukázalo týmto testom, je vhodnejšie trénovacie dáta automaticky preložiť do jedného z jazykov a následne na nich spustiť fine-tuning. Takto natrénované modely poskytujú lepšie výsledky.

V tomto experimente sa ukázalo, že SlovakBERT nedokázal správne klasifikovať testovacie správy a označil ich všetky ako pravdivé. Toto zlyhanie bolo zapríčinené rozdielnou témou trénovacích dát a aj chybou prekladu anglického datasetu do slovenského jazyka.

Z tohto experimentu vyplýva, že aj keď je známe, že väčší objem dát môže vylepšiť výkon modelu, v prípade fake-news je kľúčové zohľadniť tému testovaných falošných správ a tiež dôležitosť jazykových špecifík cieľového jazyka, v ktorom sú texty napísané.

13.3.2 COVID19 FN

Fine-tuning modelov v tejto časti, podobne ako pre LIAR, najskôr prebehol na COVID19 FN datasete a následne opätovný fine-tuning, ale už na slovenskom trénovacom datasete. Výsledky po evaluácii na testovacom datasete sú uvedené v tabuľke 13.5. Podobne ako v predošlom experimente aj v tomto pre multilingválne modely prebehol test s preloženými datasetmi do anglického jazyka aj na kombinácii angličtiny a slovenčiny podľa toho, v akom jazyku bol daný dataset pôvodne. Pre slovenský model SlovakBERT sme opätovne preložili COVID19 FN do slovenského jazyka ako v predošlom prípade LIAR datasetu.

Tento test zavŕšuje sadu experimentov pomocou modelov založených na architektúre Transformer na DownSampled SlovakCoid19 FN dataset. Tento test sme založili na rovnakom princípe ako predošlý na LIAR datasete len s tým cieľom, že ich chceme porovnať navzájom a zistiť, či rovnaká téma medzi datasetmi v iných jazykoch môžeme pomôcť modelom pri detekcii falošných správ. Ako vidíme v tabuľke 13.5, rovnaká téma cudzojazyčných datestov pomohla oproti

LIAR experimentu. Došlo k veľkému zlepšeniu medzi zahraničným datasetom s rovnakou a odlišnou témou. Všetky modely okrem toho dosiahli zlepšenie oproti tréningu iba na zahraničných dátach. To znamená, že aj malý dataset v cieľovom jazyku s rovnakou témou falošných správ dokáže pomôcť zlepšiť predikciu týchto modelov, ktoré sú trénované na zahraničných dátach. Tento zahraničný dataset však musí byť v rovnakej tematickej oblasti ako cieľový. Toto zistenie vyplynulo z porovnania testov LIAR a COVID19 FN datasetov.

Experiment	epochs	precision	recall	accuracy	f1-score
Slovenská dátová sada preložená do angličtiny.					
BERT	19	0,7450	0,7551	0,7474	0,7500
mBERT*	10	0,7928	0,5986	0,7201	0,6822
RoBERTa	15	0,7410	0,8367	0,7713	0,7859
XLM-RoBERTa*	12	0,7260	0,7211	0,7235	0,7235
Dátové sady v pôvodných jazykoch.					
mBERT	9	0,8062	0,7075	0,7679	0,7536
XLM-RoBERTa	4	0,7313	0,6667	0,7099	0,6975
Dátová sada LIAR preložená do slovenčiny.					
SlovakBERT	4	0,8095	0,8095	0,8089	0,8095

Tabuľka 13.5: Experiment: Fine-tuning (COVID19 FN)

13.3.3 Záver

Čisto slovenský model, ktorý bol natrénovaný iba na slovenskom datasete, dosiahol najlepšie výsledky. Predpokladáme, že tento výsledok je spôsobený špecificitou fake-news pre Slovensko. Táto špecificita je daná malým množstvom médií, ktoré šíria na Slovensku falošné správy, a tým sa odlišujeme od zahraničia, kde je týchto zdrojov väčšie množstvo a je možné zovšeobecniť detekciu falošných správ. Tieto zistenia je možné interpretovať nasledovne, ak pre daný jazykový korpus existuje predtrénovaný model, je optimálne ho použiť na detekciu falošných správ. Ak tento model pre daný jazykový korpus neexistuje, potom je lepšie použiť metódu automatického prekladu v spojení s modelom, ktorý je určený pre daný preložený jazyk. Najmenej vhodnou alternatívou je použité multilingválnych modelov napríklad typu mBERT v danom jazyku, ak ho podporuje, alebo automaticky preložiť textu do podporovaného jazyka. Ukazuje sa, že nie je vhodné na fine-tuning použiť inú tematickú oblasť, akej sa týka cieľová oblasť fake-news. Podobne aj kombinácia viacerých rôznych jazykových korpusov zanáša šum do modelov.

Je dôležité zdôrazniť, že výsledky týchto experimentov sú ovplyvnené výberom konkrétnych datasetov. V oblasti detekcie falošných správ, a najmä pri kombinovaní datasetov z rôznych jazykov a krajín, by bolo vhodné uskutočniť ďalšiu dôkladnú analýzu. Cieľom by bolo overiť blízkosť medzi jednotlivými jazykmi a krajinami v kontexte falošných správ. Táto analýza by nám pravdepodobne poskytla ďalšie poznatky o tom, ako efektívne kombinovať datasety v rôznych jazykoch s cieľom zlepšiť výslednú detekciu.

13.4 SlovakCovid19 FN

Poslednou sadou experimentov v tejto diplomovej práci, ktoré sme vykonali pomocou modelov založených na architektúre Transformer, bola ich implementácia pre celý slovenský dataset na základe výsledkov predošlých testov. Ako sa ukázalo v predošlých experimentoch, najvyššiu úspešnosť mala slovenská verzia modelu BERT, ktorá bola natrénovaná čisto na slovenskom datasete. Preto sme sa v poslednom teste rozhodli otestovať práve túto kombináciu modelu a datasetu a vyhodnotiť ich úspešnosť. Na tejto kombinácii modelu a datasetu sme vykonali množstvo experimentov s rôznymi modifikáciami ako modelu tak aj trénovania, ktoré sú súčasťou tejto práce v priložených súboroch. Modifikácia modelu SlovakBERT prebiehala pripojením ďalších vrstiev za tento model. Týmito vrstvami boli CNN, LSTM, RCL, plne prepojené vrstvy, dropout vrstva a rôzne iné možnosti. Najúspešnejšiu verziu tohto modelu uvedieme v tejto časti kapitoly aj s výsledkami, ktoré dosiahol na tomto datasete. Podobne ako v predošlých testoch bol celý dataset rozdelený na tri časti v pomere 6:2:2 (train, valid, test).

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_ids (InputLayer)       [(None, 512)]             0

attention_mask (InputLayer)  [(None, 512)]             0

roberta (TFRobertaMainLayer) TFBaseModelOutputWi  124644864
                             thPoolingAndCrossAt
                             tentions(last_hidde
                             n_state=(None, 512,
                              768),
                              pooler_output=(Non
                             e, 768),
                              past_key_values=No
                             ne, hidden_states=N
                             one, attentions=Non
                             e, cross_attentions
                             =None)

spatial_dropout1d (SpatialDrop  (None, 512, 768)    0
out1D)

global_average_pooling1d (Glob  (None, 768)         0
alAveragePooling1D)

dense (Dense)                (None, 128)           98432
dense_1 (Dense)              (None, 64)            8256
dense_2 (Dense)              (None, 32)            2080
dense_3 (Dense)              (None, 16)            528
dropout_37 (Dropout)         (None, 16)            0
dense_4 (Dense)              (None, 2)             34
=================================================================
Total params: 124,754,194
Trainable params: 124,754,194
Non-trainable params: 0
_________________________________________________________________
```

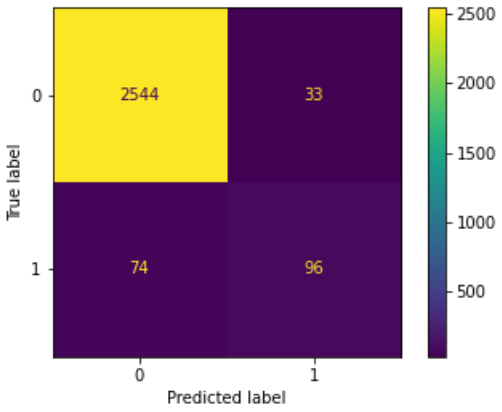
Python implementácia 13.8: SlovakBERT - SlovakCovid19 FN

Výslednú architektúru, ktorá dosiahla najlepšie výsledky pri detekcii fake-news na celom slovenskom datasete, sme pre model SlovakBERT implementovali v jazyku Python pomocou knižnice *tensorflow*. SlovakBERT rozšírený o tieto klasifikačné vrstvy dosiahol výsledky, ktoré sú uvedené v tabuľke 13.6 a na ilustrácii konfúznej matice 13.2.

Experiment	epoch	precision	recall	accuracy	f1-score
SlovakBERT	20	0,7442	0,5647	0,9610	0,6421

Tabuľka 13.6: Experiment: SlovakBERT (SlovakCovid19 FN).

Finálna verzia SlovakBERT modelu v tejto implementácii dosiahla najlepšie výsledky na celom slovenskom datasete spomedzi všetkých modelov založených na architektúre Transformer. Jej úspešnosť je najlepšia spomedzi všetkých spôsobov automatickej detekcie falošných správ na slovenských textoch, ktoré sme v tejto práci testovali na tomto datasete. Prekonáva základné metódy strojového učenia aj hlboké neurónové siete vo všetkých nami sledovaných metrikách. Podobne dosahuje najlepšie výsledky z množstva experimentov založených na architektúre Transformer, ktoré sme taktiež vykonali na tomto datasete.



Obr. 13.2: Experiment: SlovakBERT (SlovakCovid19 FN).

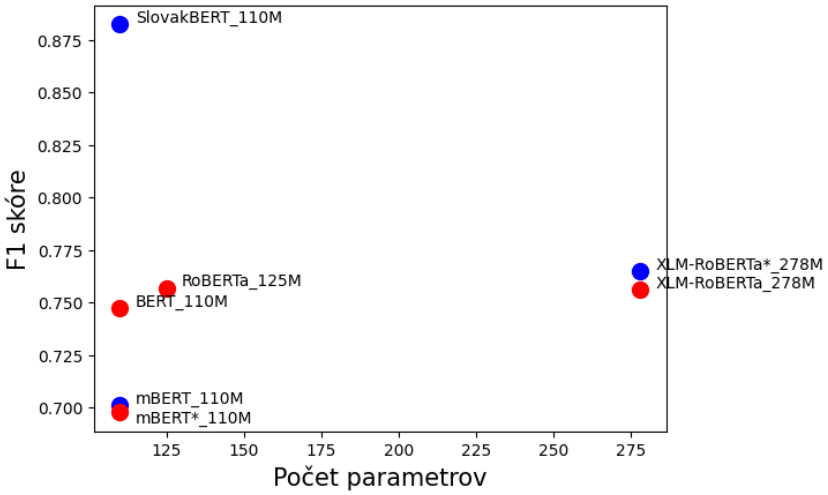
Z konfúznej matice tohto modelu môžeme vidieť veľkú nevyváženosť v celom datasete a možný potenciál zlepšenia výsledkov tohto modelu na základe predošlých experimentov pri zlepšení celého slovenského datasetu.

13.5 Analýza vplyvu jazyka

V rámci tejto práce sme využili viaceré dátové sady v rôznych jazykoch na trénovanie modelov a vyhodnotenie ich úspešnosti pri detekcii falošných správ na slovenskej testovacej dátovej sade. Konkrétne sme zahrnuli nasledujúce jazyky: angličtina a slovenčina. Rovnako sme použili rôzne jazykové mutácie modelov založených na architektúre Transformer v kombinácii s prekladom, prvý natívny model pre slovenský jazyk, ako aj viacjazyčné modely. Touto rôznou kombináciou jazykov a modelov sme chceli overiť, či vplyv štýlu, akým sú písané falošné správy v danom jazyku, má nejaký významný dosah na ich výslednú detekciu.

Cieľom tejto analýzy je lepšie porozumieť, či sa jazykové odchýlky medzi falošnými a pravdivými správami prejavujú aj na lexikálnej úrovni a či je možné preložené texty brať za rovnocenné. V tejto časti analýzy sa zameriame na vplyv prekladu konkrétne na dátovej sade DownSampled SlovakCovid19 FN.

Na ilustrácii 13.3 je stĺpcový graf znázorňujúci úspešnosť modelov založených na architektúre Transformer, ktoré boli trénované a vyhodnotené na DownSampled SlovakCovid19 FN dátovej sade v pôvodnej a preloženej forme. Modely v grafe sú zoradené podľa počtu parametrov a rovnaká farba znamená, že boli trénované a testované v rovnakom jazyku (modely testované v angličtine majú červenú farbu, v slovenčine modrú).



Obr. 13.3: Analýza vplyvu jazyka.

Pri detailnejšom rozbore grafu si môžeme všimnúť, že aj keď počet parametrov ovplyvňuje konečnú úspešnosť jednotlivých modelov, dôležitejším faktorom je skutočnosť, že model je určený pre zdrojový slovenský jazyk, v ktorom boli falošné správy napísané. Napríklad SlovakBERT s 110M parametrami dosahuje výrazne lepšie výsledky než XLM-RoBERTa s asi o 160M viac parametrami, ktorý bol druhým najúspešnejším modelom, keďže je priamo navrhnutý pre slovenský jazyk a nie je potrebné použiť preklad. Pracuje priamo s textami v ich pôvodnej forme so všetkými lexikálnymi a gramatickými formami z pôvodného textu. Ukazuje sa, že prekladom textov do angličtiny sa z dátovej sady odstraňujú jemné odchýlky medzi falošnými a pravdivými správami v ich lexikálnej a gramatickej štruktúre. Podobne, ak sa pozrieme na porovnanie čisto slovenského modelu s multilingválnymi modelmi použitými na nepreloženom datasete, tak sa ukazuje výhoda tohto modelu. Tieto multilingválne modely nemajú takú schopnosť zachytiť úplne najmenšie detaily slovenského jazyka. Predpokladáme, že to je spôsobené malou trénovacou množinou v slovenskom jazyku vo fáze ich predtrénovania. Tejto skutočnosti nasvedčuje aj fakt, že multilingválny model XLM-RoBERTa, ktorý je predtrénovaný na väčšej slovenskej množine ako mBERT dáva lepšie výsledky. Predpokladáme, že to je spôsobené tréningom na tejto väčšej trénovacej množine a tým spôsobenému lepšiemu pochopeniu súvislosti v slovenských textoch. Z našich výsledkov vyplýva, že pri detekcii falošných správ je dôležité zohľadniť jazykový kontext. Modely trénované na korpuse v rovnakom jazyku ako testovacia (cieľová) množina fake-news majú tendenciu dosahovať lepšie výsledky v porov-

naní s modelmi trénovanými na dátových sadách v odlišných jazykoch. Ukazuje sa, že preklad nie je vhodným spôsobom pre detekciu na základe textu, pretože zanáša a odstraňuje jazykové odchýlky akými sú písané fake-news. Rovnako multilingválne modely v dnešnej dobe ešte nevedia plne nahradiť veľké jazykové modely pre konkrétne jazyky, ktoré prinášajú veľkú konkurenčnú výhodu.

Tieto zistenia naznačujú, že výber správneho modelu v danom jazyku je kľúčovým faktorom pri detekcii falošných správ. Existuje niekoľko faktorov, ktoré môžu vysvetľovať tento jav. Prvým je lexikálna a gramatická štruktúra jazyka. Každý jazyk má svoje vlastné charakteristiky, slovnú zásobu a gramatické pravidlá, čo môže ovplyvniť používanie jazyka v komunikácii. Teda štýl, ktorým sú písané falošné správy, môže niesť určité špecifické znaky. Napríklad, niektoré jazyky môžu mať viac synoným pre kľúčové slová často používané vo falošných správach, čo môže spôsobiť komplikácie pri ich detekcii ak texty prekladáme, alebo ak modely nedokážu tieto rozdiely v slovách vyhodnotiť.

Analýza vplyvu jazyka dátovej sady na detekciu falošných správ ukázala, že jazyk má významný vplyv na úspešnosť modelov. Modely trénované na dátových sadách v rovnakom jazyku dosahujú lepšie výsledky v porovnaní s modelmi trénovanými na dátových sadách v odlišných jazykoch. Pri výbere dátových sád a trénovaní modelov je preto dôležité zohľadniť jazykový kontext a špecifiká danej populácie. Predpokladáme, že tieto zistenia môžu byť nápomocné pri vývoji efektívnych modelov na detekciu falošných správ v rôznych jazykových prostrediach. Na druhú stranu, tieto výsledky a analýzy sú uskutočnené na základe experimentov, ktoré sme vykonali v tejto práci. Vzhľadom na naše obmedzenia sme si vedomí nutnosti ďalšieho dodatočného skúmania tejto problematiky fake-news pre slovenský jazyk, aby sa potvrdili tieto hypotézy a výsledky.

13.6 Analýza vplyvu témy fake-news

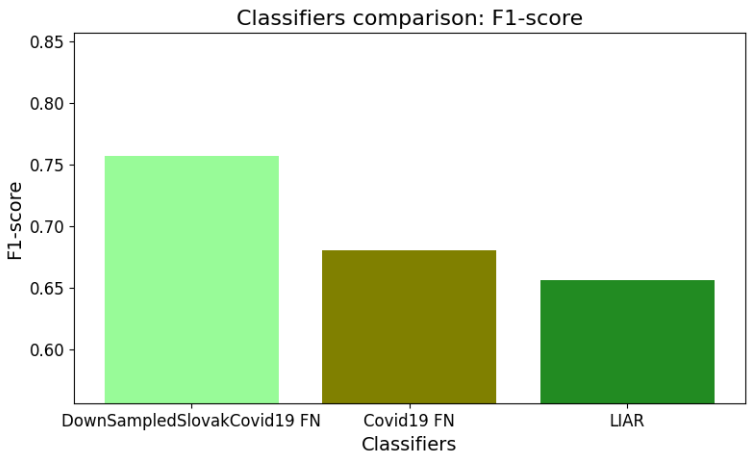
Táto analýza sa zameriava na súhrn výsledkov predchádzajúcich experimentov a skúma vplyv tém článkov na konečnú úspešnosť detekcie falošných správ. Cieľom tejto analýzy je preskúmať, či falošné správy majú spoločné charakteristiky, alebo sú odlišné v závislosti od tematickej oblasti ku ktorej sa vzťahujú. Pri trénovaní sme využili tri rôzne datasety: LIAR, DownSampled SlovakCovid19 FN a COVID19 FN. Testovanie sa uskutočnilo na nezávislej testovacej množine vyčlenenej z DownSampled SlovakCovid19 FN dátovej sady. Touto metodikou vyhodnotenia, respektíve testovania, na dátových sadách s rôznou témou sme chceli porovnať vplyv podobnosti témy v trénovacom a testovacom datasete na výslednú detekciu, či majú falošné správy z rôznych oblastí podobné charakteristiky, alebo sú tieto množiny fake-news rozdielne.

Pre otestovanie tejto hypotézy sme použili kombináciou rôznych trénovacích množín s rôznou témou článkov a overiť ich vplyv na výslednú detekciu. Prvou dátovou sadou bola LIAR, ktorá je kontextovo zaradená do politickej sféry v USA, na druhú stranu Covid19 FN je anglický dataset v problematike pandémie t.j. tematicky ma blízko k slovenskému datasetu, iba je to zahraničná verzia dátovej sady zaoberajúca sa pandémiou covid-19. Pre vylúčenie jazykovej bariéry sme v tejto analýze ako referenčnú vzorku zobrali preloženú trénovaciu množinu slovenského datasetu do angličtiny, ktorá by mala typom správ zodpovedať najpresnejšie preloženej testovacej množine. V tejto analýze sa detail-

nejšie pozrieme na model RoBERTa, na ktorom porovnáme úspešnosť vzhľadom k tréningu na rôznych dátových sadách. Porovnanie vykonáme na tomto modeli vzhľadom k jeho najstabilnejším výsledkom naprieč všetkými kombináciami datasetov z predošlých experimentov. Správanie ostatných testovaných modelov je analogické, iba s horšími výsledkami.

V tejto analýze sa zameriame na vplyv tréningu na jednom datasete a následné otestovanie na slovenskom datasete s rôznou tematikou. Okrem toho skúmame aj alternatívny prístup, ktorý kombinuje trénovanie na dvoch datasetoch a hodnotí ich úspešnosť na rovnakej testovacej množine. Týmto druhým krokom by sme chceli overiť a otestovať možnosti spájania rôznojazyčných datasetov do väčších celkov, ktoré by následne mohli byť použité na trénovanie komplexnejších a efektívnejších detektorov fake-news.

V prvom kroku sa detailnejšie pozrieme na jednoduchý fine-tuning na jednej z dátových sád LIAR, COVID19 FN a DownSampled SlovakCovid19 FN. Na ilustrácii 13.4 sú výsledky úspešnosti modelu RoBERTa po fine-tuningu na jednom z týchto datasetov a následnej evaluácii na slovenskej testovacej množine preloženej do angličtiny.



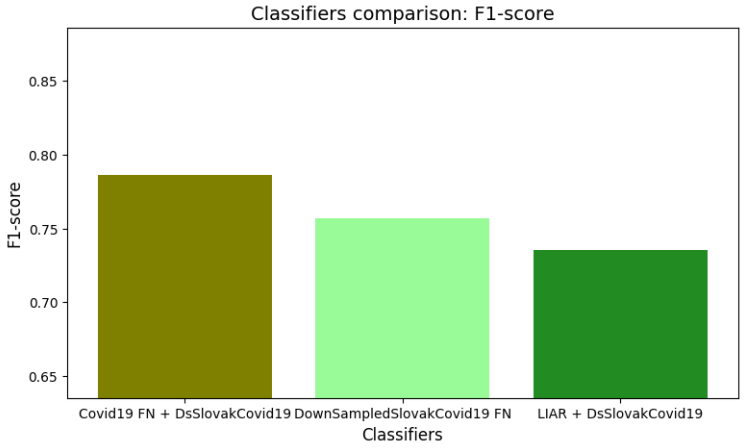
Obr. 13.4: Analýza vplyvu témy.

Na základe výsledkov experimentov vykonaných v tejto práci sa ukazuje, že tematické oblasti fake-news sa medzi sebou odlišujú. Model RoBERTa natrénovaný iba na LIAR datasete dosiahol najnižšiu úspešnosť na testovacom slovenskom datasete. Predpokladáme, že tento výsledok je s najväčšou pravdepodobnosťou spôsobený rôznou tematickou oblasťou, ktorá je v daných článkoch spracovávaná. LIAR je politickým datasetom zaoberajúcim sa fake-news šíriacimi sa pri voľbách v USA 2016. Na druhú stranu, testovacia slovenská dátová sade v sebe zahŕňa hlavne články z oblasti pandémie covid-19 na Slovensku. Medzi týmito datasetmi je veľký rozdiel a preto predpokladáme túto zníženú úspešnosť.

Druhý dataset, ktorý sme analyzovali bol anglický dataset COVID19 FN. Výber tohto datasetu, ktorý sme popísali v predošlých kapitolách, bol na základe jeho tematickej blízkosti k získanému slovenskému datasetu. Ak sa pozrieme na výsledky modelov natrénovaných na tomto datasete, ukazuje sa, že dosahujú lepšie výsledky ako modely trénované iba na datasete LIAR. Ukazuje sa, že rovnaká téma článkov fake-news v rôznych datasetoch vedie k miernemu zlepšeniu úspešnosti oproti článkom, ktoré pojednávajú všeobecne o fake-news, alebo sú s rôznou

tematikou. Domnievame sa, že je to spôsobené šírením falošných správ rovnakého typu, respektíve rovnakej správy v jej rôznych jazykových mutáciách v rámci tejto témy, ktorú sú schopné takéto modely detegovať naprieč rôznymi jazykmi.

V tomto prípade sa ukázala dôležitosť kultúrneho kontextu. Falošné správy môžu byť ovplyvnené špecifikami jednotlivých kultúr/lokalít, spoločenských a politických udalostí, čo sa môže prejaviť v texte správ. Modely trénované na dátových sadách v jednej kultúre môžu byť citlivejšie na špecifiká tejto kultúry a lepšie zachytávať charakteristiky falošných správ v tomto kontexte. Tento fakt potvrdzujú aj nami vykonané experimenty, kedy najlepšie výsledky dosiahol model ktorý bol trénovaný na trénovacej množine slovenského datasetu. Tento výsledok je s najväčšou pravdepodobnosťou spôsobený tým, že táto trénovacia množine v sebe zahŕňa všetky špecifiká tejto jazykovej sady bez ohľadu na jazykovú formu, keďže sme tento dataset preložili a tým zotreli rozdiely v jazyku medzi falošnými a pravdivými správami. Je tiež dôležité mať na pamäti, že výber dátových sád pre trénovanie modelov by mal zohľadňovať kultúrne a jazykové špecifiká danej populácie a cieľového jazyka. Ak je cieľom detegovať falošné správy v konkrétnom jazykovom kontexte, je vhodné použiť trénovacie dáta v tom istom jazyku.



Obr. 13.5: Analýza vplyvu témy (spojenie dátových sád).

Na základe našich zistení z prvého kroku analýzy sme dospeli k záveru, že čisto zahraničné dátové sady použité pri trénovaní všeobecného modelu na detekciu falošných správ neprodukujú uspokojivé výsledky. Preto sme sa rozhodli pokračovať v našom výskume a skúmať možnosť kombinácie zahraničných dátových sád s našou slovenskou trénovacou množinou. Cieľom je nájsť efektívnejší prístup k detekcii falošných správ, ktorý využíva kombináciu rôznych zdrojov informácií. Výsledky úspešnosti na nezávislej testovacej slovenskej množine sú zobrazené pre model RoBERTa podobne ako v predošlom prípade vo vizualizácii 13.5. Ako referenčná vzorka je uvedený tréning iba na slovenskej trénovacej množine (výsledok totožný ako na predošlej vizualizácii 13.4).

Na základe našich experimentov sa ukázalo, že spojenie dátových sád s rôznou tematickou oblasťou a taktiež z rozdielnych kultúr (lokalít) nie je správnou cestou. Ukazuje sa, že spojenie zahraničného a slovenského datasetu z rovnakej tematickej oblasti pandémie covid-19 viedlo u modelu RoBERTa k prekonaniu úspešnosti modelu natrénovaného iba čisto na slovenskej dátovej sade. Tento výsledok by mohol byť interpretovaný ako jedna z možnosti tvorby dátových sád pre

menšie jazykové korpusy. Ukazuje sa, že použitie väčšej zahraničnej dátovej sady doplnenej o malú dátovú sadu z konkrétnej lokality (kultúry) vedie k zlepšeniu oproti použitiu iba tejto lokálnej verzie. Predpokladáme, že rozšírenie zahraničnej dátovej sady z rovnakej oblasti fake-news, ktorá prispieva k všeobecným faktom o fake-news, o túto lokálnu dátovú sadu prináša dôležité dodatočné kľúčové informácie. Tieto informácie sa vzťahujú k lokálnym špecifikám falošných správ, ktoré sa šíria v tejto komunite. Je nutné brať na zreteľ, že kombinovanie je vhodné iba ak nie je dostupný veľký jazykový model pre daný jazyk. Ak takýto model existuje, jeho výsledky sú lepšie ako v prípade spojených datasetov.

Je dôležité zdôrazniť, že výsledky tejto analýzy sú ovplyvnené výberom konkrétnych datasetov a vykonanými experimentami. Výsledky sú značne limitované dostupnými dátovými sadami v slovenskom jazyku, u ktorých nie je možné priamo v slovenskom jazyku porovnať vplyv témy. Bolo by vhodné v budúcich prácach otestovať vplyv rôznej témy falošných správ iba v rámci slovenského jazyka. Tým by sa vylúčilo ako možný vplyv pri porovnávaní lokalita t.j. špecifické lokálne charakteristiky šírených správ. V oblasti detekcie falošných správ, a najmä pri kombinovaní datasetov z rôznych tém a oblastí, by bolo vhodné uskutočniť ďalšiu dôkladnú analýzu. Cieľom by bolo overiť vplyv príslušnosti k určitej kultúre a krajine v kontexte falošných správ. Táto analýza by nám pravdepodobne poskytla ďalšie poznatky o tom, ako efektívne kombinovať datasety v rôznych témach a jazykoch s cieľom zlepšiť výslednú detekciu.

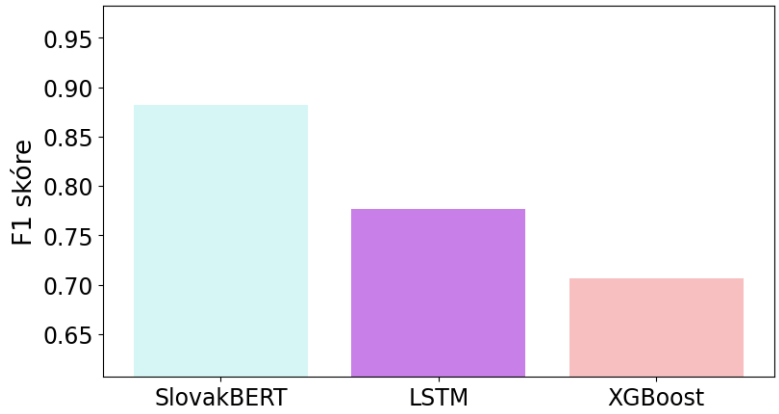
13.7 Analýza vplyvu komplexity modelov

Poslednú analýzu výsledkov, ktoré sme vykonali v tejto práci, venujeme vplyvu komplexity testovaného modelu na konečnú úspešnosť. V rámci našich experimentov sme testovali rôzne typy modelov strojového učenia, ako napríklad základné modely, hlboké neurónové siete a modely založené na architektúre Transformer.

V analýze vplyvu komplexity modelov sme sa zameriavali na hodnotenie efektivity rôznych typov modelov pri detekcii falošných správ. Naším cieľom bolo preskúmať, ako zvyšovanie komplexity modelu ovplyvňuje jeho schopnosť identifikovať a klasifikovať falošné správy. Predpokladali sme, že zvyšovanie komplexity modelu môže priniesť výhody v podobe lepšej schopnosti zachytiť sofistikovanejšie formy falošných správ a dosiahnuť vyššiu úspešnosť.

V našich experimentoch sme začali s jednoduchšími modelmi strojového učenia, ako napríklad rozhodovací strom, náhodný les, k-NN (k-nearest neighbors), naivný Bayesov klasifikátor a XGB klasifikátor. Tieto modely sú schopné zvládnuť základné úlohy, ale predpokladáme, že majú obmedzenú schopnosť rozpoznať sofistikovanejšie formy falošných správ. Neskôr sme preskúmali hlboké neurónové siete, ktoré predstavujú komplexnejšie modely s viacerými vrstvami a väčším počtom neurónov. Tieto siete majú vyššiu kapacitu na zachytenie zložitejších vzťahov v dátach a dosiahli vyššiu úspešnosť pri detekcii falošných správ. Avšak, je dôležité brať do úvahy, že dátová sada s ktorou sme pracovali má svoje nedostatky, ktoré sme už popísali pri jednotlivých experimentoch a neurónové siete môžu byť náchylnejšie na tieto vplyvy. Nakoniec sme skúmali modely založené na architektúre Transformer, ktoré využívajú mechanizmus pozornosti na spracovanie textových sekvencií. Tieto modely majú vysokú kapacitu na zachytenie jemných nuáns falošných správ a môžu sa ukázať ako účinné pri detekcii falošš-

ných správ v reálnom čase. Na ilustrácii 13.6 sú uvedené výsledky formou stĺpcového grafu pre najlepšie modely z každej sady experimentov na DownSampled SlovakCovid19 FN datasete. Z ilustrácie a na základe experimentov popísaných v predošlých kapitolách môžeme vyvodiť zistenie, že so zvyšujúcou komplexitou modelov dochádza k výraznému zlepšeniu ich úspešnosti. Treba však zdôrazniť, že zvyšovanie komplexity modelov prináša aj určité výzvy. Vyššia komplexita môže znamenať väčšiu náchylnosť na overfitting, čo môže viesť k zhoršeniu výkonu modelu na nových, neznámych dátach. Navyše, komplexnejšie modely vyžadujú väčšie množstvo trénovacích dát na efektívne učenie, čo môže byť obmedzujúcim faktorom, ak tieto dáta nie sú k dispozícii. Okrem toho, náročnejšie modely môžu vyžadovať aj väčšie výpočtové prostriedky a dlhšiu dobu trénovania.



Obr. 13.6: Analýza vplyvu komplexity modelov.

Napriek tomu, že naše výsledky poukazujú na úspešnosť komplexnejších modelov pri detekcii falošných správ, je dôležité pokračovať v skúmaní a porovnávaní rôznych prístupov a techník. Rozvoj efektívnych modelov na detekciu falošných správ je neustály proces, ktorý vyžaduje kontinuálne zlepšovanie a aktualizáciu na základe nových poznatkov a výziev, ktoré sa vyskytujú vo fake-news prostredí.

Celkovo, aj keď sme získali určité zistenia o vplyve komplexity modelov na úspešnosť detekcie fake-news, je stále potrebné ďalšie štúdium tejto problematiky pre slovenský jazyk a rozličné jazykové prostredia. Tieto dodatočné analýzy a experimenty nám pomôžu poskytnúť presnejšie a robustnejšie modely pre boj proti šíreniu falošných správ.

13.8 Porovnanie s predošlými experimentami

V tejto podkapitole porovnáme výsledky našej práce s predchádzajúcimi experimentami, ktoré uskutočnili Sarnovský a kolektív na zdrojovej dátovej sade, ktorú si špecificky upravili. Je dôležité poznamenať, že naša práca je jednou z prvých, ktorá sa zaoberá detekciou falošných správ v slovenčine na takto rozsiahlej sade experimentov. Predchádzajúce štúdie buď pracovali s inými jazykmi, najčastejšie s angličtinou, alebo sa obmedzovali na vytvorenie datasetu a základné experimenty. V tejto časti porovnáme naše výsledky s výsledkami Sarnovského a kolektívu, ktoré sme popísali v kapitole 1.2. V tejto predošlej práci [2] sa zaoberali klasifikáciou falošných správ v slovenskom jazyku, pričom ich práca spočívala vo

vytvorení dátovej sady a následnej základnej sady experimentov, v ktorej používali tradičné modely hlbokých neurónových sietí, ako sú CNN, LSTM (RNN) a biLSTM + CNN. Ich najlepší model, biLSTM + CNN s použitím metódy fine-tuning, dosiahol vysoké accuracy 0,9893 a F1 skóre 0,9400 na testovacej množine. Tieto výsledky boli dosiahnuté na celej dátovej sade, ktorú pred jej vyhodnotením predspracovali. Ich práca je podobná našej; zakladá si na rovnakom zdrojovom datasete po fáze data mining. Od tejto fázy získania dát sa spracovanie datasetov odlišuje. V práci Sarnovský a kolektív nepopisujú všetky kroky predspracovania textov, takže sme dodržali iba tie, ktoré popísali, a ostatné sme vytvorili na základe dát a ich detailnejšieho preskúmania, ktoré sme popísali v kapitole 9. Hlavný rozdiel týchto prác spočíva v sade experimentov. Práca Sarnovského a kolektívu sa zameriava hlbšie vytvorením datasetu a okrajovo sa zameriava na základnú sadu experimentov, v našej práci to je naopak.

Táto práca sa zaoberá komplexnejšou sadou experimentov, kedy sme použili modely od základných algoritmov až po veľké jazykové modely založené na architektúre Transformer. S cieľom detailnejšie spracovať dátovú sadu a určiť výhody na nevýhody tejto dátovej sady. Porovnali sme výsledky na celej dátovej sade predošlého modelu Sarnovského a kolektívu s našimi výsledkami a zistili sme, že ich model dosiahol lepšie výsledky. Naším najlepším modelom pre celú dátovú sadu SlovakCovid19 FN je SlovakBERT natrénovaný na slovenskej trénovacej množine, ktorý dosiahol výsledky accuracy 0,9610 a F1 skóre 0,642. Tieto výsledky sú horšie ako výsledky z predošlej práce. Tento rozdiel môže byť spôsobený rôznymi faktormi, ako sú odlišná architektúra modelov, odlišné rozdelenie datasetu na trénovaciu, validačnú a testovaciu množinu. My však predpokladáme, že je spôsobená hlavne rôznou formou predspracovania dát.

V kapitole 9 a jej podkapitole 9.2.1 sme popísali proces vytvárania a čistenia nášho dátového setu a zistili sme, že niektoré články obsahujú dodatočný text, ktorý nesúvisí s obsahom článku. Dôležité je zdôrazniť, že tieto dodatočné textové informácie sa nachádzali iba v podmnožine článkov patriacich výhradne do triedy falošných správ. V skutočnosti boli tieto informácie prítomné asi v polovici článkov z tejto množiny. Keby sme tieto textové dáta neodstránili, klasifikátor by bol náchylný na klasifikáciu týchto správ ako falošných na základe tejto časti textu a nie na základe ich skutočného obsahu. V pôvodnej práci Sarnovského a jeho kolektívu nie je uvedené, ako sa s týmito informáciami vysporiadali, či ich ponechali alebo odstránili z dátovej sady. V našej práci sme sa rozhodli odstrániť všetky tieto „metadáta“, keďže cieľom nášho výskumu bola klasifikácia fake-news na základe textového obsahu článkov.

S ohľadom na nami dosiahnuté výsledky a porovnania s výsledkami Sarnovského a jeho kolektívu v ich práci [2] sa domnievame, že v ich práci tieto doplňujúce textové dáta z článkov neodstránili. Toto by mohlo viesť k skresleniu výsledkov a zlepšeniu ich úspešnosti v porovnaní s našou súpravou experimentov. Avšak my sa domnievame, že k správnej metodike spracovania dát patrí odstránenie týchto dodatočných informácii z textov, aby bolo výsledne vyhodnotenie korektné. V práci Sarnovského a jeho kolektívu sa priznáva, že výsledky ich práce sú ovplyvnené kvalitou dát, spôsobom čistenia a anotácie. Niektoré aspekty práce nedostatočne popisujú ako sa riešili časti textu, ktoré nesúvisia priamo s obsahom článku, ako napríklad úvodné alebo záverečné vety špecifické pre konkrétne médiá. Autori tiež uznávajú, že výsledky sú výrazne ovplyvnené formou dát a ne-

vyváženosťou v dátovej sade. Vzhľadom na tieto nedostatky odporúčajú ďalší výskum v oblasti detekcie falošných správ.

Na základe porovnania výsledkov s predchádzajúcimi experimentami a našich zistení ohľadom predspracovania dátovej sady možno konštatovať, že naša práca prináša nové poznatky a výsledky v oblasti detekcie falošných správ v slovenskom jazyku. Hoci sme nedosiahli lepšie výsledky ako práca Sarnovského a kolektívu, naša práca prispieva k rozšíreniu poznatkov a predstavuje nový prístup k tejto problematike. Dôležité je poznamenať, že každá práca má svoje vlastné špecifiká a výsledky môžu byť ovplyvnené rôznymi faktormi. Budúce výskumy by mohli skúmať možnosti kombinácie rôznych prístupov a techník s cieľom dosiahnuť ešte lepšie výsledky v detekcii falošných správ v slovenskom jazyku.

13.9 Zhrnutie

Posledná sada experimentov sa zamerala na veľké lingvistické modely založené na architektúre Transformer z kapitoly 8. Výkonnosť jednotlivých modelov rovnako ako v predošlých testoch bola porovnaná na SlovakCovid19 FN datasete, podobne ako na jeho zmenčenej verzii. Pre fine-tuning boli použité aj zahraničné dátové sady, konkrétne LIAR a COVID19 FN v anglickom jazyku. Touto sadou testov sme chceli overiť úspešnosť modelov založených na architektúre BERT aj vplyv rôznorodosti datasetov, ktoré sa líšili v téme falošných správ, jazyku, prípadne vplyv veľkosti datasetu. Pre overenie týchto kombinácií sme v sérii testov vykonali takmer 50 experimentov s rôznymi modelmi v kombinácii s voľbou parametrov a datasetom. Výsledkom týchto experimentov sú zistenia, ktorých zhrnutie uvedieme v nasledujúcich odstavcoch.

Prvým zistením a zároveň najzásadnejším, ktoré vyplynulo z množstva našich experimentov, je sila natívnych modelov. Ukázalo sa, že aj keď multilingválne modely prípadne cudzojazyčné spojené s automatickým prekladom poskytujú dobrú alternatívu v prípade nedostupnosti natívneho modelu, ale tieto modely v prípade existencie modelu založeného na architektúre BERT (prípadne iný model založený na architektúre Transformer) priamo pre cieľový jazyk nedokážu tento model prekonať a nedosahujú ani zďaleka také dobré výsledky.

Druhým zistením, ale nie menej dôležitým, je vplyv témy falošných správ na štruktúru textových dát. Ukázalo sa, že tematicky rôzne falošné správy nezdieľajú také množstvo lingvistických znakov. To znamená, že pre predikciu fake-news na základe textových dát je nutné pri tréningu zachovať tematiku falošných správ na základe testovacej množiny. Obzvlášť pri zahraničných zdrojoch v spojení s prekladom. Inak je úspešnosť takého modelu zreteľne znížená.

Posledným výsledkom týchto experimentov je vplyv kvality datasetu na celkovú predikciu klasifikátora. Vplyv dátovej sady na celkovú úspešnosť modelu je markantná a ukazuje sa, že kvalita datasetov pre detekciu falošných správ pre slovenský jazyk je nedostatočná.

Zhrnutím našich poznatkov je nutnosť skvalitnenia trénovacích dát pre detekciu fake-news na Slovensku prípadne pre iné jazykové korpusy a zároveň použitie, respektíve vytvorenie predtrénovaných veľkých jazykových modelov pre konkrétny jazyk. Tieto dve zistenia môžu pomôcť zlepšiť automatickú detekciu falošných správ na Slovensku aj vo svete. Je však dobré podotknúť, že výsledky týchto experimentov sú ovplyvnené výberom konkrétnych datasetov a ich metodikou.

14. Obmedzenia práce

Naša práca má určité obmedzenia, pretože mnohé rozhodnutia pri návrhu boli významne ovplyvnené získanou zdrojovou dátovou sadou pre slovenský jazyk a následnou zvolenou metodikou spracovania tejto dátovej sady. Obmedzenia tejto práce sa teda dajú rozdeliť do dvoch častí: V prvom kroku sa budeme venovať dátovej sade a jej problémom a následne rozoberiem možné obmedzenia a nevýhody zvolenej metódy detekcie.

V porovnaní s inými jazykmi, ako napríklad angličtina, pre slovenský jazyk neexistuje žiadna verejné dostupná dátová sada týkajúca sa detekcie fake-news. To komplikuje prácu a vytvorenie automatických detektorov fake-news pre slovenčinu. Ďalšou komplikáciou, ktorá vyplýva z tohto problému je, že nasledujúcu štúdiu sme museli prispôsobiť forme dátovej sady, ktorú sa nám podarilo získať. Pre iné jazyky, ako angličtina, kde je takýchto dátových sád väčšie množstvo, je možné zvoliť dátovú sadu na základe metodiky, ktorú chceme otestovať. V našom prípade sme museli upraviť metódu detekcie falošných sprav na základe dát, ktoré nám boli poskytnuté.

Získaná dátová sada, ktorá nám bola poskytnutá Sarnovským a kolektívom, v sebe nesie aj množstvo obmedzení, ktoré spomínajú autori už v svojej práci [2]. Aj keď výsledky ich štúdie znejú sľubne, sami priznávajú, že je dôležité zdôrazniť, že sú veľmi ovplyvnené kvalitou dát, spôsobom čistenia a anotácie, ktoré môžu mať za následok skreslenie finálnych výsledkov. My sme v našej práci zistili ešte ďalšie obmedzenia tejto dátovej sady, ktoré nie sú detailnejšie popísané v predošlej práci, ktoré by mohli mať rovnako vplyv na výslednú klasifikáciu. Tejto problematike sme sa venovali v kapitole 9, konkrétne v časti 9.2.1. Jedným z príkladov je napríklad ako sa riešili časti textov, ktoré nesúvisia priamo s obsahom článku, ako napríklad úvodné alebo záverečné vety špecifické pre konkrétne médiá. Samotní autori odporúčajú ďalší výskum v oblasti detekcie falošných správ, ku ktorému sa aj po výsledkoch našej práce prikláňame.

Obmedzenia dátových sád by sa dali sumarizovať týmito charakteristikami. Počet a množstvo dátových sád pre slovenský jazyk je nedostatočné. Tieto dátové sady nepokrývajú celé spektrum rôznych tematických oblasti, ktorých sa môžu fake-news týkať. Dátové sady v slovenskom jazyku po vykonaní našich experimentov sú dosť ovplyvnené špecifikami lingvistických vlastností tohto jazyka, ako napríklad ohýbanie slov a bohatosť slovníka, ktoré predstavujú výzvy pri vytváraní efektívnych algoritmov detekcie falošných správ, ktoré by dokázali správne identifikovať a interpretovať rôzne formy falošných informácií. Taktiež, v slovenskom jazyku sa môžu vyskytovať sémantické nuansy, ktoré vyžadujú hlbšie porozumenie kontextu a kultúry, aby bolo možné efektívne identifikovať falošné správy. Niektoré falošné správy môžu využívať zaužívané konotácie, iróniu alebo špecifické slovné spojenia, čo predstavuje výzvu pre automatizované systémy detekcie.

Taktiež, pri detekcii falošných správ je dôležité mať dostupné spoľahlivé a overené referenčné zdroje, ktoré slúžia na porovnávanie a overovanie pravdivosti informácií. Posledným obmedzením spojeným so slovenskými datasetmi je nedostatok akreditovaných anotátorov. To má za následok nedostatok nezávislých a spoľahlivých faktických databáz, o ktoré by sa mohli opierať algoritmy detekcie falošných správ.

Na základe získaného datasetu sme zvolili metódu detekcie falošných správ založenú na klasifikácii textu na základe obsahu. Táto metóda detekcie falošných správ so sebou prináša aj obmedzenia a nevýhody. V článku *„Fake News Detection via NLP is Vulnerable to Adversarial Attacks“* [71], ktorý publikoval Zhixuan Zhou a kolektív detailnejšie rozoberajú problémy modelov založených na klasifikácii textu pri detekcii falošných správ. V tomto článku hodnotili detektor falošných správ Fakebox vzhľadom na nepriateľské útoky, vrátane skresľovania faktov. Experimenty ukázali, že útok vážne narušuje model. Autori tejto štúdie predpokladajú, že podobné modely založené výlučne na jazykových charakteristikách budú v reálnom svete oveľa menej efektívne a budú zraniteľné voči manipuláciám. Tento druh útoku je oveľa jemnejší (decentnejší), pretože nezmení celkový štýl písania novinových článkov a má potenciál vyhnúť sa detekcii podobnosti. Zároveň dospeli k záveru, že pri nedostatočne dobre napísaných skutočných článkoch alebo určitých témach, ktoré sa často považujú za zdroje falošných správ, sa zvyšuje miera falošne pozitívnych hodnotení. Táto situácia môže negatívne ovplyvniť entuziazmus amatérskych novinárov, keďže existuje potenciál pre nesprávnu klasifikáciu nedostatočne napísaných, avšak skutočných správ.

Autori tejto štúdie predpokladajú, že do modelov detekcie falošných správ je nevyhnutné integrovať porovnávanie a kontrolu faktov z viacerých zdrojov, aby sme skutočne vedeli odhaliť dezinformácie a nie len rozhodovať na základe klasifikácie textov. Jedným z možných spôsobom získavania faktov o nových udalostiach je využitie *crowdsourced knowledge grafu*, ktorý je popísaný v tejto štúdii. Tento graf je dynamicky aktualizovaný miestnymi a dobre informovanými ľuďmi. Zbierané aktuálne informácie sa potom môžu porovnať s informáciami extrahovanými z novinových článkov a pomôcť generovať označenie pravdivosti [71].

Podobnou štúdiou, ktorá sa tiež zaoberá robustnosťou metód detekcie falošných správ je *„All Your Fake Detector Are Belong to Us: Evaluating Adversarial Robustness of Fake-News Detectors Under Black-Box Settings“* [72], ktorá rovnako ako predošlá štúdia porovnáva odolnosť rôznych detektorov voči nepriateľským útokom. Na tento účel použili štyri rôzne architektúry - viacvrstvový perceptron (MLP), konvolučné neurónové siete (CNN), rekurentné neurónové siete (RNN) a nedávno navrhnutý hybridný detektor falošných správ CNN-RNN - a viacero datasetov - Kaggle dataset falošných správ, ISOT dataset a LIAR dataset. Otestovali rôzne formy útokov a zistili, že tieto útoky majú výrazný vplyv na neurónové siete, ktoré sa zameriavajú na detekciu falošných správ iba na základe klasifikácie textu. Na druhú stranu, vo svojej štúdii popisujú aj protiopatrenia. Napríklad pri útoku Text-Bugger spôsobujú útočníci perturbácie, často v podobe pravopisných chýb, ktoré môžeme jednoducho odhaliť pomocou kontroly pravopisu. Alternatívne je možné úmyselne náhodne perturbovať trénovaciu sadu, tak aby obsahovala pravopisné chyby ako metódu augmentácie údajov na zvýšenie odolnosti modelu NLP.

Taktiež pozorovali, že pre čisté/nezasiahnuté vstupy je konečné rozhodnutie v súlade s mnohými prispievajúcimi slovami, na rozdiel od prípadu s nepriateľskými vstupmi, kde sa konečné rozhodnutie zvyčajne pripisuje len niekoľkým z mnohých vstupných slov. Tento poznatok môže byť využitý na účinné protiopatrenia voči hrozbe nepriateľských (adversarial) príkladov.

Experimenty v tejto štúdii naznačujú vývoj budúcich obranných mechanizmov založených na architektúrach RNN. Tiež zistili, že dlhšie vstupné príklady (texty)

umožňujú detektoru naučiť sa lepšie reprezentácie, čo zvyšuje presnosť a odolnosť detektorov falošných správ. Okrem toho, použitie silných techník regularizácie, napríklad L2-regularizácia, môže výrazne zvýšiť odolnosť detektorov falošných správ, aj keď s poklesom presnosti na pôvodných vstupoch. Avšak takýto pokles presnosti detektorov môže byť vyvážený pomocou silných techník augmentácie údajov.

Na záver možno konštatovať, že detekcia falošných správ je komplexná úloha a vyžaduje viac ako iba základnú klasifikáciu textu. Integrácia kontroly pravopisu, použitie silných techník regularizácie, či kontrola, ktoré slová prispievajú k výslednej klasifikácii by mali viesť k zlepšeniu týchto metód. Na druhú stranu sa ukazuje, že nástroje detekcie fake-news založené na metódach klasifikácie textu môžu byť zneužité pri tvorbe nových automatických nástrojov, ktoré budú mať za cieľ oklamanie týchto detektorov a šírenie nedetekovateľných falošných správ. Preto je vhodná integrácia týchto metód s porovnávaním faktov z rôznych zdrojov a využitie ďalších metód, ako je *crowdsourced knowledge graf*, môže prispieť k lepšej identifikácii a interpretácii falošných správ. Je potrebné pokračovať vo výskume a vyvíjať nové techniky a metódy, ktoré budú schopné efektívne detegovať a odhaľovať falošné správy v rôznych jazykoch a kontextoch.

Záver

V tejto práci sme sa zaoberali problematikou automatickej detekcie falošných správ (fake-news) v slovenčine z pohľadu klasifikácie textu. Fake-news, teda úmyselne nepravdivé alebo skreslené informácie, sa v poslednej dobe na internete vyskytujú čoraz častejšie, a preto je dôležité naučiť sa ich automaticky detegovať. Práca je jedinečná v tom, že spravodajské články boli spracované v slovenskom jazyku, pre ktorý ešte takto rozsiahla sada experimentov nebola vykonaná. Väčšina doterajších štúdii, ktoré sa zaoberali touto problematiku, bola buď určená pre cudzojazyčné dátové sady, typicky pre angličtinu, alebo štúdie určené pre slovenčinu iba popisovali vytvorenie datasetu so základnou sadou experimentov.

Naším hlavným cieľom bolo vytvorenie nástroja, ktorý by dokázal detegovať falošné správy na slovenských textoch nielen z novinových článkoch, ale aj v príspevkoch na sociálnych sieťach. Okrem toho sme si stanovili viacero čiastkových krokov, ktoré nám mali pomôcť tento cieľ dosiahnuť. Jedným z nich bolo vytvorenie vlastného datasetu pre detekciu falošných správ v slovenskom jazyku. Následným krokom bolo vytvorenie a otestovanie základných modelov pre rôzne metódy strojového učenia s cieľom poskytnúť ucelený pohľad na automatickú detekciu falošných správ v slovenskom jazyku. Tieto kroky boli dôležité a ich splnenie nám umožnilo lepšie porozumieť procesu automatickej detekcie fake-news a taktiež položilo základ pre ďalšie výskumy v tejto oblasti.

Na začiatku práce ešte pred tvorbou klasifikátora bolo nevyhnutné nájsť dostatočne veľký dataset pre trénovanie a testovanie detekčných modelov v slovenskom jazyku. Získanie tohto datasetu bolo výzvou, pretože v tom čase nebola verejne dostupná žiadna slovenská dátová sada. Nakoniec sa nám podarilo získať základnú dátovú sadu od tímu Sarnovského na Technickej univerzite v Košiciach. Umožnil nám týmto krokom prístup k prvému slovenskému datasetu na detekciu falošných správ v tematike covidu-19. Na základe tejto tematiky slovenskej dátovej sady sme vybrali vo vzťahu k nej aj ostatné cudzojazyčné datasety na detekciu falošných správ. Konkrétne sme vybrali anglickú dátovú sadu COVID19 FN, ktorá má rovnakú tematiku článkov ako slovenský dataset. Druhým anglickým datasetom, ktorý sme v tejto práci použili, je LIAR, ktorý v sebe obsahuje články z politickej oblasti. Táto dátová sada bola vybraná na otestovanie spoločných vlastností fake-news s rôznou témou článkov.

Vykonali sme viac ako osemdesiat experimentov na zistenie vhodnej kombinácie predspracovania textu a modelu strojového učenia. Taktiež sme vykonali množstvo testov s cudzojazyčnými (anglickými) datasetmi v kombinácii so slovenskou dátovou sadou, na ktorých sme overili prenositeľnosť znakov falošných správ medzi jazykmi, respektíve témami. Po niekoľkých experimentoch s rôznymi modelmi strojového učenia sme zistili, že náš pôvodný dataset bol nevyvážený a len ťažko riešiteľný pre základné modely. Preto sme sa rozhodli vytvoriť vlastnú zmenšenú verziu datasetu s vyváženými triedami, ktorá by mohla slúžiť ako referenčný benchmark pre naše ďalšie experimenty a ako trénovacia sada pre rôzne modely. Tento nový dataset bol vytvorený z pôvodného datasetu s manuálne overenými anotáciami, ktorý sme zmenšili a vyvážili a v poslednom kroku sme ho použili ako referenčný bod pre všetky naše experimenty.

Následne sme na oboch slovenských datasetoch otestovali základnú sadu expe-

rimentov. V nej sme vykonali testy pomocou jednoduchých modelov strojového učenia ako Bayesov model, rozhodovací strom, náhodný les, k-nn a XGBoost. Na základe výsledkov jednotlivých metód strojového učenia sme vytvorili optimalizovaný baseline model pomocou XGBoost, ktorý dosiahol najlepšie výsledky $F1_{downsampled} = 0{,}7065$ pre zmenšenú verziu datasetu a $F1 = 0{,}5831$ pre celý pôvodný dataset. V ďalšom kroku sme otestovali komplexnejšie modely založené na architektúre neurónových sietí. Konkrétne CNN, RNN a RCNN, ktorých výsledky sme porovnali s baseline modelom, ale aj medzi sebou. Všetky neurónové siete v porovnaní so základnou sadou experimentov dosiahli lepšie výsledky. Najkvalitnejšie výsledky pre oba slovenské datasety dosiahla neurónová sieť LSTM. Na druhej strane už v tomto experimente sa ukázalo, že celý slovenský dataset by potreboval kvalitnejšie dáta z dôvodu nízkeho obsahu falošných správ.

Najrozsiahlejšia časť práce sa zaoberala vykonaním experimentov s modelmi založenými na architektúre Transformer. Modely založené na tejto architektúre, ktoré sme otestovali, boli BERT, mBERT, RoBERTa, XLM-RoBERTa a SlovakBERT. Tieto modely sme otestoval na oboch slovenských datasetoch aj na kombinácii s anglickými verziami datasetov. Závery zo sady experimentov poukázali na to, že modely s komplexnejšou architektúrou sa dokážu lepšie vysporiadať so stále nedostatočnou slovenskou dátovou sadou. Zároveň sa však ukázala potreba vytvárania kvalitnejších datasetov.

Kombinácia slovenských datasetov s datasetmi v iných jazykoch s cieľom rozšírenia obsahu textov má svoje obmedzenia. Predpokladáme, že hlavným dôvodom je samotný slovenský dataset, ktorý má značné obmedzenia. Ďalšími možnými dôvodmi, ku ktorým sme dospeli na základe vykonaných experimentov, sú napríklad: že pri písaní falošných správ sa prejavuje špecifickosť slovenského jazyka, ktorá sa stráca pri preklade do iných jazykov. Tento efekt síce nie je až taký výrazný v porovnaní s multilingválnymi modelmi, no s prvým slovenským modelom založeným na architektúre Transformer je stále významný. Rovnako spracovanie multilingválnymi modelmi nevie dostatočne podchytiť túto jemnú odchýlku. Toto zistenie potvrdilo nutnosť aj naďalej vytvárať špecifické modely pre každý konkrétny jazykový korpus.

Pri testovaní kombinácie slovenských a zahraničných dátových sád sme ďalej zistili, že vplyv témy článkov na trénovanie a testovanie modelov je významný. Zistili sme, že trénovanie modelov na jednej oblasti falošných správ a ich následné použitie v inej oblasti nie je možné. Možno predpokladať, že podobné témy alebo dokonca rovnaký jazyk môžu byť zahrnuté do jedného datasetu. Avšak datasety s falošnými správami pre rôzne jazyky a s rôznymi témami nie sú zlučiteľné a prispievajú k veľkému množstvu odlišností a šumu počas učenia modelov.

V rámci poslednej sady experimentov boli testované rôzne modely založené na architektúre Transformer a ich výsledky boli zhodnotené. Zistilo sa, že modely BERT a mBERT dosahovali všeobecne nižšiu úspešnosť ako ostatné testované modely. Tento fakt sa môže vysvetliť nižším počtom parametrov a špecifikami slovenského jazyka. Modely RoBERTa a XLM-RoBERTa dosiahli lepšie výsledky ako BERT a mBERT, ale najúspešnejším modelom sa stal SlovakBERT natrénovaný iba na slovenskej dátovej sade, ktorý má rovnaký počet parametrov ako BERT. Tento model v kombinácii so slovenským trénovacím datasetom dosiahol najlepšie výsledky na testovacej množine pre obe slovenské datasety, pri zmenšenom datasete dosiahol hodnoty $F1_{downsampled} = 0{,}8824$ (o 18% viac)

a $acc_{downsampled} = 0{,}8771$, zatiaľ čo na celej sade dosiahol prekvapivo dobré výsledky $F1 = 0{,}6421$ a $acc = 0{,}9610$ (acc o 2% viac ako baseline model a F1 skóre zhruba o 8%).

SlovakBERT model trénovaný na čisto slovenskom datasete dosiahol pozoruhodné výsledky v detekcii fake-news. Jednou z jeho výhod v porovnaní s cudzojazyčnými modelmi je, že nie je potrebné prekladať slovenské texty, čo eliminuje šum spôsobený prekladom a zlepšuje presnosť klasifikácie. Ale je nutné počítať s tým, že ak takýto model neexistuje pre jazyk, jeho pretrénovanie je výpočtovo náročné. Ak takýto model existuje, potom nie je nutné prekladať texty a zvyšuje to presnosť modelu aj výslednú rýchlosť klasifikácie. Je možné argumentovať, že použitie multilingválnych modelov eliminuje potrebu prekladu textov, čím sa ušetrí čas a úsilie pri detekcii falošných správ. Tieto modely sú však zvyčajne trénované na korpusoch obsahujúcich viacero jazykov a množina dát pre konkrétny jazyk, na ktorý sa chceme zamerať, môže byť minimálna. Toto má častokrát za následok zníženú presnosť týchto modelov pri detekcii falošných správ.

V tejto práci sme položili základný kameň pre detekciu falošných správ v slovenskom jazyku, na ktorej výsledkoch sa v budúcnosti dá stavať. Je dôležité uvedomiť si, že výsledky tejto práce sú ovplyvnené zvolenou metodikou a použitými dátovými sadami. Uznávame, že v novej téme detekcie falošných správ by bolo vhodné vykonať ďalšiu hlbšiu analýzu od spracovania dát až po vyhodnotenie výsledkov, čo by pravdepodobne prinieslo ďalšie poznatky. Dúfame, že výsledky práce poslúžia ako inšpirácia pre ďalší výskum v oblasti klasifikácie textu v oblasti fake-news.

Budúca práca

Detekcia falošných správ je v dnešnej dobe čoraz väčší a aktuálnejší problém, ktorý je nutné stále riešiť. V nadväznosti na túto prácu by bolo možné pokračovať rôznymi smermi. Jedným z možných smerovaní pre zlepšenie klasifikácie falošných správ je vytvorenie kvalitnejších a rozsiahlejších dátových sád z rôznych tematických oblastí v slovenskom jazyku.

Ďalšou možnosťou je použitie na transfer a fine-tuning iné jazyky ako angličtina, napríklad čeština, poľština alebo iné slovanské jazyky, ktoré sú tomuto jazyku bližšie, čo by mohlo viesť k zlepšeniu a zastúpeniu jazykov so spoločnými znakmi. Tento prístup by mohol viesť k zlepšeniu detekcie falošných správ a zahrnutiu jazykových špecifík spoločných pre tieto jazyky.

Iným smerom môže byť kombinácia nelingvistických a lingvistických metód pre detekciu falošných správ. Napríklad kombinácia textu a multimediálneho obsahu článku pre zlepšenie detekcie. Prípadne použitie metadát, s čím súvisí vytvorenie datasetu, ktorý v sebe zahŕňa aj tieto informácie.

Celkovo existuje viacero prístupov a kombinácií, ktoré sa dajú použiť na zlepšenie detekcie falošných správ. Neustále sa vylepšujúce technológie a prístupy môžu pomôcť v boji proti šíreniu dezinformácií a zabezpečiť lepšiu dôveryhodnosť správ a informácií.

Zoznam použitej literatúry

[1] Andrea Renda (CEPS Centre for European Policy Studies and College of Europe). The legal framework to address “fake news”: possible policy actions at the eu level, 2018.

[2] Martin Sarnovský, Viera Maslej-Krešňáková, and Klaudia Ivancová. Fake News Detection Related to the COVID-19 in Slovak Language Using Deep Learning Methods. *Acta Polytechnica Hungarica*, 19(2):43–57, 2022.

[3] Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. Fighting an infodemic: COVID-19 fake news dataset. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 21–29. Springer International Publishing, 2021.

[4] Gautam Kishore Shahi and Durgesh Nandini. FakeCovid - A multilingual cross-domain fact check news dataset for COVID-19. *CoRR*, abs/2006.11343, 2020.

[5] Limeng Cui and Dongwon Lee. CoAID: COVID-19 healthcare misinformation dataset. *CoRR*, abs/2006.00885, 2020.

[6] William Yang Wang. "Liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

[7] Anna Glazkova, Maksim Glazkov, and Timofey Trifonov. g2tmn at constraint@AAAI2021: Exploiting CT-BERT and ensembling learning for COVID-19 fake news detection. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 116–127. Springer International Publishing, 2021.

[8] Martin Müller, Marcel Salathé, and Per E Kummervold. COVID-Twitter-BERT: A Natural Language Processing Model to Analyse COVID-19 Content on Twitter, 2020.

[9] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2019.

[10] Zekeriya Anil Guven, Banu Diri, and Tolgahan Cakaloglu. Comparison of topic modeling methods for type detection of turkish news. In *2019 4th International Conference on Computer Science and Engineering (UBMK)*. IEEE, sep 2019.

[11] Justus Mattern, Yu Qiao, Elma Kerz, Daniel Wiechmann, and Markus Strohmaier. FANG-COVID: A new large-scale benchmark dataset for fake news detection in German. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 78–91, Dominican Republic, November 2021. Association for Computational Linguistics.

[12] Natali Ruchansky, Sungyong Seo, and Yan Liu. CSI. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, nov 2017.

[13] Limeng Cui, Kai Shu, Suhang Wang, Dongwon Lee, and Huan Liu. defend: A system for explainable fake news detection. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2961–2964, 2019.

[14] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36, 2017.

[15] Dominic Spohr. Fake news and ideological polarization: Filter bubbles and selective exposure on social media. *Business information review*, 34(3):150–160, 2017.

[16] Dominic DiFranzo and Kristine Gloria-Garcia. Filter bubbles and fake news. *XRDS: Crossroads, The ACM Magazine for Students*, 23(3):32–35, 2017.

[17] Gabor Hulko. Fake news and social media: Regulation and administrative measures in slovakia. *In Medias Res*, page 297, 2021.

[18] Ivan Janiga. Jediný slovenský fact-checker pre facebook: Všetkých dezinformátorov spája, že chcú získať čo najviac followerov. `https://www.attelier.sk/fact-checker-robert-barca-rozhovor/`, 2021. [Online; accessed 10-Dec-2022].

[19] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[20] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. *Advances in neural information processing systems*, 32, 2019.

[21] David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Generating sentiment-preserving fake online reviews using neural language models and their human-and machine-based detection. In *International Conference on Advanced Information Networking and Applications*, pages 1341–1354. Springer, 2020.

[22] Munazza Zaib, Quan Z Sheng, and Wei Emma Zhang. A short survey of pre-trained language models for conversational ai-a new age in nlp. In *Proceedings of the Australasian Computer Science Week Multiconference*, pages 1–4, 2020.

[23] Christopher D Manning. *Introduction to information retrieval*. Syngress Publishing,, 2008.

[24] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[25] Complete guide to understanding precision and recall curves. https://analyticsindiamag.com/complete-guide-to-understanding-precision-and-recall-curves/, 2021. [Online; accessed 20-Jun-2023].

[26] Yutaka Sasaki et al. The truth of the f-measure. *Teach tutor mater*, 1(5):1–5, 2007.

[27] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[28] tutorialandexample. Decision trees in machine learning. https://www.tutorialandexample.com/decision-trees, 2019. [Online; accessed 10-Feb-2023].

[29] Klára Komprdová et al. *Rozhodovací stromy a lesy*. Akademické nakladatelství CERM, 2012.

[30] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[31] datacamp. https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn, 2013. [Online; accessed 10-Feb-2023].

[32] Jan Švehla. Za nového člověka! sociální experiment v poválečném československu na příkladu emila zátopka. 2021.

[33] Robi Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45, 2006.

[34] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[35] Tom Dietterich. Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, 27(3):326–327, 1995.

[36] wikipedia. Random forest. https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png. [Online; accessed 10-Feb-2023].

[37] Tin Kam Ho. Recognition of handwritten digits by combining independent learning vector quantizations. In *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pages 818–821. IEEE, 1993.

[38] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[39] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[40] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[41] Xgboost. `https://www.geeksforgeeks.org/xgboost/`, 2021. [Online; accessed 20-Jun-2023].

[42] Ing. Juraj Muráň. Úvod do neurónových sietí - ako fungujú neurónové siete. `https://umelainteligencia.sk/uvod-do-neuronovych-sieti/`, 2013. [Online; accessed 20-Feb-2023].

[43] Simon Haykin. *Neural networks and learning machines, 3/E*. Pearson Education India, 2009.

[44] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

[45] George Cybenko. Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:183–192, 1989.

[46] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[47] Meriem Bahi and Mohamed Batouche. Deep learning for ligand-based virtual screening in drug discovery. pages 1–5, 10 2018.

[48] Pranoy Radhakrishnan. What are hyperparameters? and how to tune the hyperparameters in a deep neural network. *Towards Data Science*, 18, 2017.

[49] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[50] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[51] Wen-tau Yih, Kristina Toutanova, John C Platt, and Christopher Meek. Learning discriminative projections for text similarity measures. In *Proceedings of the fifteenth conference on computational natural language learning*, pages 247–256, 2011.

[52] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[53] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[54] Leila Kalkhoran, Shima Tabibian, and Elaheh Homayounvala. Detecting persian speaker-independent voice commands based on lstm and ontology in communicating with the smart home appliances. *Artificial Intelligence Review*, 11 2022.

[55] Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3367–3375, 2015.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[57] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[58] Jay Alammar. The illustrated transformer. *The Illustrated Transformer–Jay Alammar–Visualizing Machine Learning One Concept at a Time*, 27, 2018.

[59] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[60] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[61] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[62] Radford Alec, Narasimhan Karthik, Salimans Tim, and Sutskever Ilya. Improving language understanding with unsupervised learning. *Citado*, 17:1–12, 2018.

[63] Abhilash Jain, Aku Ruohe, Stig-Arne Grönroos, and Mikko Kurimo. Finnish language modeling with deep transformer models. *arXiv preprint arXiv:2003.11562*, 2020.

[64] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does bert look at? an analysis of bert's attention, 2019.

[65] Petr Zelina. Pretraining and evaluation of czech albert language model.

[66] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[67] Matúš Pikuliak, Štefan Grivalský, Martin Konôpka, Miroslav Blšták, Martin Tamajka, Viktor Bachratý, Marián Šimko, Pavol Balážik, Michal Trnka, and Filip Uhlárik. Slovakbert: Slovak masked language model, 2021.

[68] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[69] Jose Camacho-Collados and Mohammad Taher Pilehvar. On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis. *arXiv preprint arXiv:1707.01780*, 2017.

[70] Vimala Balakrishnan and Ethel Lloyd-Yemoh. Stemming and lemmatization: A comparison of retrieval performances. 2014.

[71] Zhixuan Zhou, Huankang Guan, Meghana Moorthy Bhat, and Justin Hsu. Fake news detection via nlp is vulnerable to adversarial attacks. *arXiv preprint arXiv:1901.09657*, 2019.

[72] Hassan Ali, Muhammad Suleman Khan, Amer AlGhadhban, Meshari Alazmi, Ahmad Alzamil, Khaled Al-Utaibi, and Junaid Qadir. All your fake detector are belong to us: evaluating adversarial robustness of fake-news detectors under black-box settings. *IEEE Access*, 9:81678–81692, 2021.

Zoznam obrázkov

Zoznam tabuliek

A. Prílohy

A.1 Technická špecifikácia použitých zariadení

V tejto práci sa na prípravu dát a vykonanie experimentov využívala platforma *Google Colab*, ktorá poskytuje cloudové výpočtové prostredie. Na výpočty bola využitá grafická karta NVIDIA A100-SXM s dostupnou operačnou pamäťou (RAM) 89,6 gigabajtov. Presné technické špecifikácie grafickej karty sú uvedené na obrázku A.1.

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.85.12    Driver Version: 525.85.12    CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-SXM...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   33C    P0    46W / 400W |      0MiB / 40960MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Obr. A.1: Google Colab výpočtová jednotka.

A.2 Použité technológie pri implementácii

V tejto prílohe sa budeme zoberať popisom technológií, ktoré sme použili pri implementácií tejto diplomovej práce, ktorá sa zaoberá detekciou falošných správ. Uvedieme niekoľko nástrojov a technológií, ktoré sme použili, aby sme zabezpečili efektívne a presné spracovanie a analýzu slovenských textov.

Jedným z týchto nástrojov bol *Jupyter Notebook*, ktorý poskytol interaktívne programovacie prostredie pre jazyk Python 3.9.0. Python sme si vybrali pre jeho vysokú popularitu, jednoduchosť použitia a široké možnosti knižníc a nástrojov. Okrem toho sme využili aj *Google Colab*, cloudové vývojové prostredie s výkonnými výpočtovými zdrojmi, ktoré nám umožnilo efektívne rýchlejšie iterovanie pri vývoji.

Pri implementácii sme využili aj množstvo knižníc jazyka Python, ktoré sú uvedené v nasledujúcej časti:

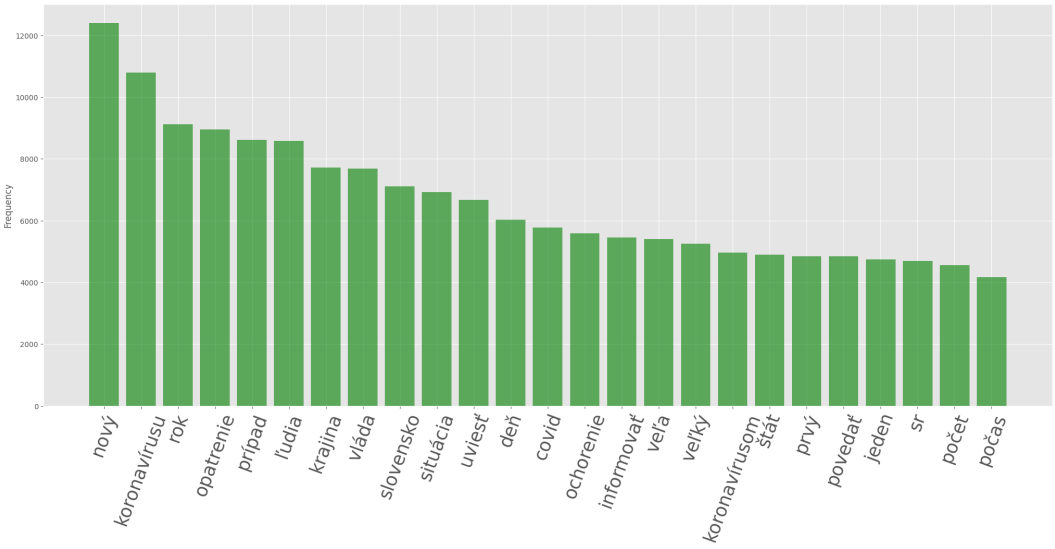
1. Na spracovanie a čistenie dát sa použili knižnice:
    - *'os'* na manipuláciu s adresárovým systémom a načítanie dát z rôznych formátov (napr. .csv, .txt, .json)

- *'re'* na spracovanie a extrakciu informácií z textov
- *'string'* na prácu s reťazcami a znakmi
- *'nltk'* a *'simplelema'* na tokenizáciu, odstránenie stopwords a stemming slov

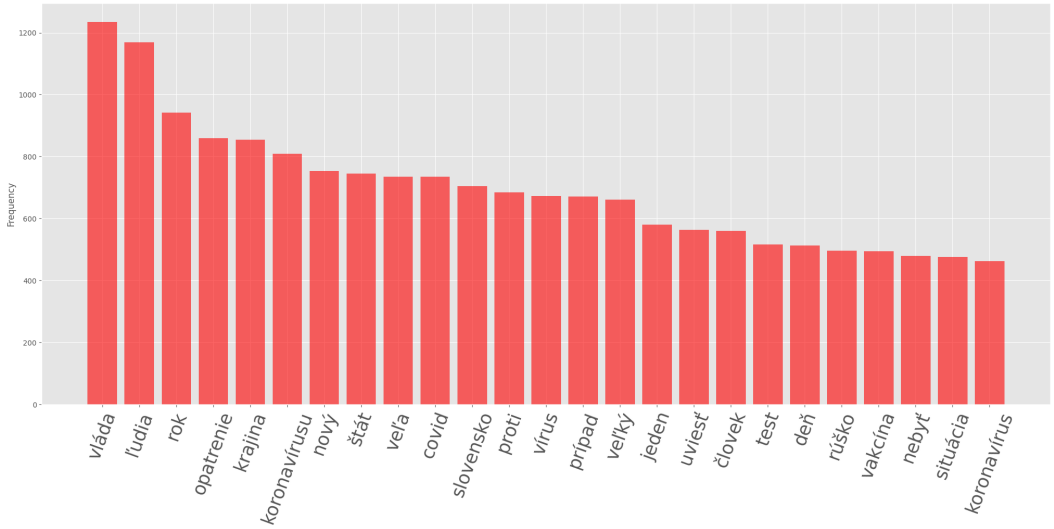
2. Vektorizácia textu sa konala pomocou knižníc:
   - *'sklearn'* na vytvorenie vektorových reprezentácií textov
   - *'transformers'* na vytvorenie vektorových reprezentácií textov pomocou predtrénovaných jazykových modelov (napr. BERT, RoBERTa, SlovakBERT)
3. Trénovanie modelov strojového učenia:
   - *'sklearn'* na trénovanie klasifikačných modelov (napr. DecisionTreeClassifier, RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier, SVC, LinearSVC, MLPClassifier, MultinomialNB, KNeighborsClassifier, XGBClassifier)
   - *'tensorflow'* a *'keras'* na vytvorenie hlbokých neurónových sieti
   - *'transformers'* pomocou, ktorého boli vytvorené veľké predtrénované jazykové modely (napr. BERT, RoBERTa, SlovakBERT)
   - *'hyperopt'* na optimalizáciu hyperparametrov modelov
   - *'callback'* na monitorovanie a zlepšovanie výkonnosti modelov počas trénovania
4. Vyhodnocovanie a vizualizácia výsledkov:
   - Použitie knižnice *'sklearn'* na vyhodnocovanie výkonnosti modelov pomocou metrík ako presnosť, recall, F1-skóre a ROC krivka
   - Knižnica *'matplotlib'* bola použitá na vizualizáciu dát a výsledkov
   - Použitie knižníc *'pandas'* a *'numpy'* na spracovanie a manipuláciu so vstupnými dátami
   - *'argparse'* pre reprezentácie parametrov výpočtov
   - *'pickle'* na uloženie modelov strojového učenia a dátových sád
5. Preklad textov do rôznych jazykov za použitia knižnice:
   - *'googletrans'* vo verzii *3.1.0a0*

Celková technická špecifikácia zahŕňa viacero knižníc na spracovanie, vektorizáciu a trénovanie klasifikačných modelov a zároveň na vizualizáciu ich výsledkov. Použitie týchto knižníc umožnilo efektívne riešenie problému detekcie falošných správ v tejto práci.

A.3 Najfrekventovanejšie slová v slovenskej dátovej sade



Obr. A.2: 25 najfrekventovanejších slov v pravdivých článkoch.



Obr. A.3: 25 najfrekventovanejších slov vo falošných článkoch.

A.4 Súhrn výsledkov základných experimentov

Klasifikátor	Accuracy	F1 skóre	AUC
DownSampled SlovakCovid19 FN dátová sada.			
XGBClassifier	0,793	0,791	0,866
MultinomialNB	0,777	0,799	0,844
RandomForestClassifier	0,759	0,752	0,832
DecisionTreeClassifier	0,692	0,685	0,692
KNeighborsClassifier	0,608	0,578	0,638
Baseline	0,707	0,706	0,773
Ensamble systém	0,771	0,771	0,858
SlovakCovid19 FN dátová sada.			
XGBClassifier	0,941	0,155	0,837
MultinomialNB	0,895	0,394	0,817
RandomForestClassifier	0,937	0,064	0,787
DecisionTreeClassifier	0,913	0,259	0,602
KNeighborsClassifier	0,895	0,095	0,589
Baseline	0,941	0,583	0,838
Ensamble systém	0,939	0,519	0,805

Tabuľka A.1: Porovnanie základných klasifikátorov.