

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra radioelektroniky



Detekce podvržené řeči

DIPLOMOVÁ PRÁCE

Autor: Bc. Jiří Šmíd
Vedoucí: Doc. Ing. Petr Pollák, CSc.
Datum: Květen, 2025



ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šmíd** Jméno: **Jiří** Osobní číslo: **491855**
Fakulta/ústav: **Fakulta elektrotechnická**
Zadávající katedra/ústav: **Katedra radioelektroniky**
Studijní program: **Elektronika a komunikace**
Specializace: **Audiovizuální technika a zpracování signálů**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Detekce podvržené řeči na bázi DNN

Název diplomové práce anglicky:

DNN-based Deepfake Speech Detection

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Petr Pollák, CSc. katedra teorie obvodů FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **05.02.2025**

Termín odevzdání diplomové práce: **23.05.2025**

Platnost zadání diplomové práce: **20.09.2026**

doc. Ing. Stanislav Vítěk, Ph.D.
podpis vedoucí(ho) ústavu/katedry

prof. Mgr. Petr Páta, Ph.D.
podpis proděkana(ky) z pověření děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je iřeba uvést v diplomové práci.

Datum převzetí zadání

Podpis studenta



ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	Šmíd	Jméno:	Jiří	Osobní číslo:	491855
Fakulta/ústav:	Fakulta elektrotechnická				
Zadávající katedra/ústav:	Katedra radioelektroniky				
Studijní program:	Elektronika a komunikace				
Specializace:	Audiovizuální technika a zpracování signálů				

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Detekce podvržené řeči na bázi DNN

Název diplomové práce anglicky:

DNN-based Deepfake Speech Detection

Pokyny pro vypracování:

1. Seznamte se s problematikou detekce podvržené řeči s užším zaměřením na způsoby umělého generování podvržené řeči a možnosti její detekce.
2. Proveďte podrobnou rešerší stavu problematiky v dané oblasti včetně dostupných aktuálně používaných systémů pro detekci podvržené řeči na bázi hlubokých neuronových sítí.
3. S dostupnými knihovnami a databázemi reálné a podvržené řeči pro anglický jazyk implementujte vybrané systémy a dosažené výsledky srovnajte s aktuálními výsledky ostatních autorů v dostupné literatuře.
4. Analyzujte jazykové závislosti navrženého systému detekce podvržené řeči. Vyvořte základní soubor reálných a podvržených dat pro češtinu a na jejím základě vytvořte základní verzi klasifikátoru podvržené řeči i pro český jazyk.

Seznam doporučené literatury:

- [1] L. Cuccovillo et al : Open Challenges in Synthetic Speech Detection. In Proc. of IEEE WIFS, Shanghai, China, Dec 2022.
- [2] H. H. Kilinc and F. Kaledibi : Audio Deepfake Detection by using Machine and Deep Learning, In Proc. of WINCOM, Istanbul, Turkiye, 2023.
- [3] O. A. Shaaban, R. Yildirim, A. A. Alguttar : Audio Deepfake Approaches. In IEEE Access, vol. 11, 2023.

PROHLÁŠENÍ

Já, níže podepsaný

Příjmení, jméno studenta: Šmíd Jiří
Osobní číslo: 491855
Název programu: Elektronika a komunikace

prohlašuji, že jsem diplomovou práci s názvem

Detekce podvržené e-mail na bázi DNN

vypracoval samostatně a uvedl veškeré použité informace zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských záverečných prací a Rámocovými pravidly používání umělé inteligence na VUT pro studijní a pedagogické účely v Bc a NM studiu.

Prohlašuji, že jsem v průběhu přípravy a psaní závěrečné práce použil nástroje umělé inteligence. Vygenerovaný obsah jsem ověřil. Stvrdzuj, že jsem si v domě, že za obsah závěrečné práce plně zodpovídám.

V Praze dne 24.02.2025

Bc. Jiří Šmíd

.....
podpis

Poděkování

Rád bych poděkoval Doc. Ing. Petru Pollákovi, CSc. za cenné rady, věcné připo-
mínky a vstřícnost při konzultacích a vypracování diplomové práce.

Bc. Jiří Šmíd

Název:

Detekce podvržené řeči

Autor: Bc. Jiří Šmíd

Studijní program: Elektronika a komunikace

Druh práce: Diplomová práce

Vedoucí práce: Doc. Ing. Petr Pollák, CSc.
Katedra teorie obvodů

Klíčová slova : Podvržená řeč, hlasový deepfake, syntetická řeč, ResNet
OC Softmax, zpracování řeči, český jazyk
detekce podvržené řeči

Key words: Spoofed speech, voice deepfake, synthetic speech, ResNet
OC Softmax, speech processing, Czech language
speech spoof detection

Abstrakt:

Tato práce se zabývá detekcí podvržené řeči v českém jazyce s využitím neuronové sítě ResNet a jednotřídní klasifikace pomocí OC Softmax. Teoretická část představuje problematiku syntetické řeči, zejména pak její možné zneužití, a současné přístupy k detekci tzv. podvržené řeči. Praktická část popisuje návrh a realizaci systému, včetně vytvoření českého datasetu obsahujícího přirozené i syntetické promluvy. Popsány jsou použité vstupní příznaky, architektura sítě a další úpravy. Funkčnost systému je ověřena experimentálně, včetně návrhu nové upravené klasifikační funkce pro hodnocení kvality syntetické řeči. Dosažené výsledky ukazují, že kombinace ResNet a modifikovaného OC Softmaxu účinně detekuje podvrženou řeč i v náročných podmínkách a generalizuje na neznámé systémy i akustické prostředí.

Abstract (English):

This thesis focuses on the detection of spoofed speech in the Czech language using a ResNet-based neural network and one-class classification with OC Softmax. The theoretical part introduces the state-of-the-art of synthetic speech generation, its potential misuse, and current detection approaches of so called spoofed speech. The practical part presents the design and implementation of the detection system, including the creation of a Czech dataset containing both natural and synthetic utterances. It describes the input features, network architecture, and additional modifications. The system's functionality is experimentally verified, including the proposal of a new modified scoring function for evaluating the quality of synthetic speech. Obtained results show that the combination of ResNet and modified OC Softmax effectively detects spoofed speech and generalizes well to unknown systems and acoustic conditions.

Seznam použitých zkratek

ASR	Automatické rozpoznání řečníka (z anglického <i>Automatic Speech Recognition</i>)
ASV	Automatické ověření řečníka (z anglického <i>Automatic Speaker Verification</i>)
CNN	Konvoluční neuronová síť (z anglického <i>Convolutional Neural Network</i>)
DNN	Hluboká neuronová síť (z anglického <i>Deep Neural Network</i>)
FPR	Míra falešně pozitivních výsledků (z anglického <i>False Positive Rate</i>)
FNR	Míra falešně negativních výsledků (z anglického <i>False Negative Rate</i>)
FAR	Míra nesprávně přijatých výsledků (z anglického <i>False Acceptance Rate</i>)
FRR	Míra nesprávně zamítnutých výsledků (z anglického <i>False Rejection Rate</i>)
GMM	Model gausovských směsí (z anglického <i>Gaussian Mixture Model</i>)
LFCC	Lineárně frekvenční kepstrální koeficienty (z anglického <i>Linear Frequency Cepstral Coeficients</i>)
LR	Rychlosť učení (Z anglického <i>Learning Rate</i>)
LSTM	Dlouhá krátkodobá paměť (z anglického <i>Long Short Time Memory</i>).
MFCC	Mel-frekvenční kepstrální koeficienty (z anglického <i>Mel-frequency Cepstral Coeficients</i>)
ResNet	Residuální neuronová síť (z anglického <i>Residual Neural Network</i>)
RNN	Rekurentní neuronová síť (z anglického <i>Recurrent Neural Network</i>)
TTS	Převodník textu na řeč (z anglického <i>Text to Speech</i>)
VC	Hlasová konverze (z anglického <i>Voice Conversion</i>)

Obsah

Seznam obrázků	1
Seznam tabulek	3
1 Úvod	5
2 Strojově generovaná řeč	6
2.1 Stručná historie strojově generované řeči	6
2.2 Syntéza řeči	7
2.2.1 Tradiční systémy	7
2.2.2 Systémy založené na ML a NN	8
2.3 Využití strojově generované řeči	10
2.4 Zneužití strojově generované či pozměněné řeči	10
2.4.1 Útoky cílené na konkrétní osobu	11
2.4.2 Podvržená řeč známých a vlivných osob	11
2.4.3 Útoky cílené na ASV systémy	13
3 Detekce podvržené řeči	16
3.1 Systémy detekce	16
3.1.1 Jednodušší systémy	16
3.1.2 Systémy založené na CNN	16
3.1.3 Systémy zachycující časové změny	17
3.1.4 Kombinované systémy	17
3.2 Zvolené struktury	18
3.2.1 Jednoduchý ResNet	18
3.2.2 ResNet + OC Softmax	18
3.3 Dostupné datasety	26
3.3.1 ASV Spoof dataset	26
3.4 Tvorba českého datasetu	29
3.4.1 Použité datasety	29
3.4.2 Použité nástroje	30
3.4.3 Hodnocení kvality generovaných promluv	31
3.4.4 Dataset strojově generované řeči	32
3.5 Volba příznaků	33
3.5.1 Kepstrum	33
3.6 Zvolená metrika pro testování	35
4 Implementace	37
4.1 Implementace prvních experimentů	38
4.1.1 Dataset	38

4.1.2	Model	39
4.1.3	Trénování	40
4.2	Implementace pro hlavní Experimenty	41
4.2.1	Ovládací struktura	42
4.2.2	Úprava nástrojů TTS/VC	44
4.2.3	Implementace výpočtu kepster	44
4.2.4	Implementace klasifikační struktury	44
4.3	Implementace pro Metacentrum	47
5	Experimentální část	49
5.1	Prvotní experimenty	49
5.2	Hlavní experimenty	52
5.2.1	Experiment s ASV Spoof Datasetem	52
5.2.2	Experiment SP SPOOF CS + OC Softmax	52
5.2.3	Experiment 3 - Neviděný útok	53
5.2.4	Experiment 4 - OC Softmax + Margin	54
5.2.5	Experiment 5 - dva neviděné datasety	57
5.2.6	Experiment 6 - Neznámé podmínky	58
5.2.7	Experiment 7 - Neviděné datasety přirozených promluv	59
5.2.8	Experiment 8 - Odstranění c_0	61
5.2.9	Experiment 9 - Různé banky filtrů	62
5.2.10	Experiment 10 - Normalizace	66
5.2.11	Experiment 12 - Důležité koeficienty	69
5.2.12	Experiment 13 - Paralelní větev	69
5.2.13	Další provedené experimenty	72
6	Závěr	74
Bibliografie		75

Seznam obrázků

2.1	Principiální schéma ASV	13
3.1	Základní struktura použité sítě	19
3.2	Základní struktura preaktivacního ResNet bloku	20
3.3	Porovnání projekcí příznaků při využití Softmax, AM Softmax a OC softmax. Přejato z [42]	22
3.4	Znázornění oblastí přirozených a podvržených promluv ve 2D	23
3.5	Ilustrace Softplus funkce. Přejato z [52]	24
3.6	Průběhy ztrátových funkcí pro přirozené a podvržené promluvy pro $r_{real} = 0.9, r_{fake} = 0.5, \alpha = 20$	25
3.7	Ukázka tvarů lineárně frekvenční a mel-frekvenční banky	34
5.1	Znázornění tří bloků Resnetu	50
5.2	Výsledky pro přirozené promluvy a jednotlivé systémy TTS a VC . .	51
5.3	Skóre testovacího datasetu pro 1 neznámý útok - Seed VC	54
5.4	Porovnání průběhů OC Softmax a OC Softmax + margin v závislosti na $\cos(\theta)$ pro $r_{real} = 0.9, r_{fake} = 0.5$ a $\alpha = 20$	56
5.5	Skóre testovacího datasetu pro 1 neznámý útok - Seed VC	57
5.6	Skóre testovacího datasetu pro dva neznámé útoky	58
5.7	Porovnání skóre přirozených promluv testovacího datasetu s CT a T datasety	60
5.8	t-SNE vizualizace výstupních příznakových vektorů. 0 - přirozené promluvy ze známých podmínek, 1-5 podvržená řeč, 21 - T-Dataset .	60
5.9	Porovnání průběhů stejnosměrné složky pro reprezentativní příklady z jednotlivých datasetů	60
5.10	Skóre testovacího datasetu pro odstraněné c_0	61
5.11	Banka filtrů se zužující se šírkou	62
5.12	Skóre testovacího datasetu pro zužující se banku filtrů	63
5.13	Banka filtrů s rostoucí šírkou	63
5.14	Skóre testovacího datasetu pro rozšiřující se banku filtrů	64
5.15	Porovnání skóre přirozených promluv testovacího datasetu pro 300 - 3400 Hz	65

5.16 Skóre testovacího datasetu při standardizaci	67
5.17 Skóre T - datasetu při standardizaci	67
5.18 Porovnání skóre přirozených promluv testovacího datasetu pro min-max normalizaci	69
5.19 Mapa významnosti kepstrálních koeficientů pro přirozené a podvržené promluvy - normalizováno	70
5.20 Distribuce parametrů pro 4 nejdiskriminativní koeficienty pro každý parametr (std, max-min, max-mean, ZCR).	71

*

Seznam tabulek

3.1	Reziduální struktura. Výsledné tvary počítány z v práci dále používané velikosti vstupních příznaků 60 x 750. B odpovídá velikosti dávky	20
3.2	Dostupné datasety. Přejato a přeloženo z [66]	26
3.3	Rozdělení systémů generujících podvrženou řeč, podrobnější popis systémů: [37]	28
3.4	Vygenerované a vybrané promluvy	32
4.1	Základní trénovací parametry ResNet	45
5.1	Základní parametry sítě	51
5.2	Výsledky upravené sítě	51
5.3	Celkové výsledky Experimentu 1	52
5.4	Celkové výsledky Experimentu 2	53
5.5	Error rate (False Positive/False Negative) Experimentu 2 pro jednotlivé kategorie	53
5.6	Celkové výsledky Experimentu 3	53
5.7	Error rate (False Positive/False Negative) Experimentu 3 pro jednotlivé kategorie	54
5.8	Celkové porovnání EER pro výsledky Experimentu 3 a 4	56
5.9	Porovnání ER v jednotlivých kategoriích pro výsledky Experimentu 3 a 4	56
5.10	Celkové porovnání EER pro výsledky Experimentu 4 a 5	57
5.11	Porovnání ER v jednotlivých kategoriích pro výsledky Experimentu 4 a 5	57
5.12	Porovnání FPR pro testovací dataset z Exp 5, neznámých RIR a hudebních zkreslení pro SNR 10 dB a 20 dB	59
5.13	Porovnání EER u testovacího datasetu (Exp. 5), CT a T datasetu	59
5.14	Celkové porovnání EER pro výsledky Experimentu 5 a 8	61
5.15	Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a 8	61
5.16	Celkové porovnání EER s výsledky Experimentu 5	62
5.17	Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5	63

5.18 Celkové porovnání EER s výsledky Experimentu 5	64
5.19 Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5 . .	64
5.20 Celkové porovnání EER s výsledky Experimentu 5	65
5.21 Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5 . .	65
5.22 Celkové porovnání EER pro výsledky Experimentu 5 a normovaných kepstrogramů	66
5.23 Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a normovaných kepstrogramů přes dataset	67
5.24 Celkové porovnání EER pro výsledky Experimentu 5 a normovaných kepstrogramů	68
5.25 Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a normovaných kepstrogramů	68
5.26 Celkové porovnání EER pro výsledky Experimentu 13 a paralelní větve	70
5.27 Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 13 a paralelní větve	71

*

1 Úvod

Historie strojově generované řeči sahá až do 20. let minulého století, kdy byly představeny první elektronické obvody schopné generovat jednotlivé hlásky, později i slova. Využitelná uměle vytvořená řeč se však objevuje až s rozvojem počítačových technologií a statistického modelování řeči. Průlomem v této technologii byly skryté Markovovy modely a v nedávné době neuronové sítě, především na transformerové bázi.

Dnes využívané nástroje pro generování či konverzi řeči dokáží uživatelům zprostředkovávat službu bez výrazných (slyšitelných) artefaktů, vygenerovaná promluva není monotónní a již se neobjevuje typická trhanost dřívějších systémů. Tyto nástroje dnes nachází široké využití: na telefonních infolinkách, v domácích asistentech, pro namlování voiceoverů videí na sociálních sítích, pro pomoc lidem se zrakovým postižením a pro mnoho dalších.

S rozvojem těchto technologií, jejich dostupností a relativní jednoduchostí jejich používání narostla šance jejich zneužití. Mnoho z nástrojů generování řeči či její konverze z řečníka na řečníka dokáže i z poměrně malého záznamu promluvy člověka jeho hlas poměrně důvěryhodně napodobit. Toho může být zneužito pro různé druhy podvodů, ovlivňování veřejnosti či pro útoky na biometrické systémy, které hlas využívají k identifikaci.

Tato práce si klade za cíl, prozkoumat existující možnosti detekce podvržené řeči používané pro angličtinu či čínštinu a jejich převedení i vylepšení pro český jazyk. Součástí práce je nejen tvorba klasifikátoru, ale i vytvoření databáze podvržené řeči v češtině, která bude využita pro klasifikaci.

2 Strojově generovaná řeč

2.1 Stručná historie strojově generované řeči

První pokusy o vytvoření umělé řeči, resp. umělých hlásek se objevují v 18. století, kdy byly popsány fyziologické rozdíly mezi pěti základními samohláskami. Tyto pokusy se skládaly z pěti rezonátorů, jeden pro každou samohlásku. Později se objevují mechanicko-akustické syntetizátory, které jsou schopné již generovat i základní souhlásky, včetně nasálních i explozivních.

S rozvojem elektrických obvodů ve 20. století přichází na scénu elektronické generátory řeči. Podobně jako mechanické generátory jsou z počátku schopny generovat základní samohlásky (pouze se dvěma formanty), později však i souhlásky. Prvním systémem, který lze považovat za generátor řeči, je VODER (z anglického *Voice Operating Demonstrator*), představený roku 1939. Výstupy tohoto nástroje byly poměrně vzdáleny od přirozené řeči, přesto však prokázal potenciál pro její strojové generování.

V druhé polovině 20. století jsou pak vyvíjeny různé pokročilejší syntezátory založeny na formantové syntéze, kaskádní formantové syntéze a modelování přenosové funkce vokálního traktu pro různé hlásky. Později se přidává i modelování artikulace. První kompletní TTS systém byl představen v roce 1969 a zahrnoval artikulační model i syntaktickou analýzu s pokročilými heuristikami. Později se objevují systémy, které již nejsou založeny pouze na několika parametrech popisujících jednotlivé hlásky (fóny). Systémy využívají databáze difónů (druhá polovina aktuálního fónu a první polovina následujícího), které jsou řazeny za sebe podle zadané sekvence znaků a převedeny na řečový signál např. pomocí LP analýzy [1].

První počítačové modely byly taktéž založeny primárně na formantové syntéze a nastavování přenosových funkcí vokálního traktu (VOCODER). S rozvojem výpočetního výkonu a jejich paměťové kapacity byly nahrány rozsáhlé datasety a použity k výběru parametrů, které lépe popisují fonémový i lingvistický kontext a slouží pro generování řeči s lepší prosodií, důrazy atd. Tyto systémy obsahovaly mnoho parametrů a byly složité na nastavení, proto vzniká nový přístup - statistická parameterická syntéza řeči, která využívá statistiky získané z nahraných promluv. Průlomem v tomto odvětví byly HMM (skryté Markovovy modely), které nepopisují pouze sekvence fonémů, ale i mnoho dalších kontextových informací. S rozvojem technologií strojového učení a metod jako Viterbiho algoritmus či Baum-Welchův algoritmus se na dlouho dobu staly HMM nejpoužívanějším přístupem [2].

V posledních letech, kdy techniky neuronových sítí a strojového učení pokročily

společně s dostupným výpočetním výkonem a je k dispozici dostatečné množství trénovacích dat, je v popředí syntéza řeči pomocí nástrojů strojového učení. Oproti tradičním přístupům, nabízejí tyto nástroje řadu výhod: jednodušší adaptaci na cílového řečníka, možnost vyjádření emocí, přirozenou prosodii a další. Toto je způsobeno schopností neuronových sítí naučit se a využít komplexní vzory a charakteristiky lidské řeči.

2.2 Syntéza řeči

Tato sekce se zabývá stručným popisem základních principiálních struktur pro syntézu řeči tradičními nástroji i E2E systémy založenými na NN. Podrobnější popis systémů není pro tuto práci relevantní.

2.2.1 Tradiční systémy

Tradiční systémy syntézy řeči se obvykle skládají z několika sériově řazených modulů.

První modul obvykle převádí text na fóny s prosodickými značkami (bohatá fonetická transkripce). Skládá se ze tří bloků: analyzátoru textu, fonetického analyzátoru a generátoru prosodie. Analyzátor textu zajišťuje přípravu dat, morfologickou, kontextovou, syntaktickou a prosodickou analýzu (k těm je nutné dodat další nástroje nebo soubory s gramatikou apod.)

Další z modulů převádí fonetickou transkripci na akustický signál, resp. na příznakovou reprezentaci (spectrogram, mel-spectrogram ...). Jako vstup taktéž potřebuje informace o řečníkovi, tj. o rychlosti, intonaci, důrazech aj. Jednodušší z těchto metod je formantová syntéza. Tato syntéza vychází ze zjednodušeného modelu vokálního traktu. Buzení systému odpovídá u řeči vibracím či klidovému stavu hlasivek (pulsy nebo šum), systém sám pak modeluje hlasový trakt (přenosové funkce - formanty). Jedním z používaných systémů je konkatenační syntéza. Ta skládá výslednou řeč z předem nahraných úseků řeči, které se spojují za sebe. Tento typ syntézy vyžaduje rozsáhlou databázi nahraných řečových segmentů pokryvající všechny možné kombinace zvuků v daném jazyce.

Jako poslední může následovat blok, který vygenerované příznaky převádí na řečový signál, pokud není generován samotným systémem.

Tradiční systémy jsou struktury, které vyžadují poměrně velké množství kroků a požadují mnoho externích databází či sad pravidel, a je proto poměrně složité takovýto nástroj vytvořit. U těchto systémů je také obtížné změnit výstupní charakteristiky

řeči, tj. změnit řečníka. Další nevýhodou je jazyková nepřenositelnost, tj. model potřebuje pro každý jazyk jiné sady pravidel a databáze [3].

2.2.2 Systémy založené na ML a NN

Systémy založené na NN mohou být end-to-end nebo se mohou skládat z několika částí. Některé z nich umožňují pouze TTS, jiné i převod textu na řeč (STT/ASR - z anglického *Speech To Text, Automatic Speech Recognition*) či převod řeči na řeč (hlasová konverze - převod řeči z jednoho řečníka na jiného, VC - z anglického *Voice Conversion*). Jejich hlavní výhoda spočívá v tom, že ke svému učení a běhu nepotřebují sady pravidel pro daný jazyk. Jistou reprezentaci těchto pravidel (statistický pravděpodobnostní popis) si struktura vybudovala během učení, kdy je na vstup předkládán text (či jeho vektorizovaná podoba) a jako výstup (label) pak odpovídající řečový signál. Níže jsou uvedeny nejpoužívanější nástroje:

Speech T5

Tento model vyvinutý Microsoft Research v sobě kombinuje tři úlohy: TTS, STT a VC a je založen na architektuře transformerů. Jádro je tvořené encoder-decoder strukturou a k dispozici je šest pomocných sítí (*preprocess, postprocess*) pro konkrétní aplikace [4]. Pro TTS je vstupem modelu vektorizovaný text pomocí SpeechT5Processor. Výstupem je pak mel-spektrogram, který lze převést na řečový signál pomocí WaveGlow [5]. Model byl trénován na anglických datasetech, ale lze jej přetrenovat či dotrénovat i pro jiné jazyky. Na Hugging Face jsou k dispozici dvě české mutace. První je kompletně natrénované jádro na českých datasetech [6], druhým pak je model s anglickým jádrem dotrénovaný na české části VoxPopuli datasetu [7],[8]. Experimenty provedené autory prvního z uvedených modelů ukázaly, že takovýto model je schopen velmi dobře rekonstruovat řečníka, resp. jeho příznakové vyjádření (*speaker embedding*) i z přibližně minuty hlasového záznamu [9], [10].

Tacotron 2

Jedná se o autoregresní model vyvinutý Googlem. Podobně jako SpeechT5, nepotřebuje promluvu rozloženou na jednotlivé fonémy, ale postačí mu prostý přepis promluvy (u některých implementací možný i fonémový přepis). Obsahuje mechanismus pozornosti (attention mechanism), který umožňuje lepší synchronicitu mezi textem a výstupem, kterým je mel-spektrogram. Tacotron 2 standardně používá Wave Glow pro syntézu signálů, ale lze využít i jiný systém. Stejně jako předchozí systém i Tacotron umožňuje vložení externích řečových charakteristik cílového řečníka i přenos hlasu řečníka jiného jazyka do jazyka, na který byl systém trénován [11].

FastSpeech

Tento systém vznikl za účelem vyřešení problémů s autoregresivními modely jako Tacotron 2, které jednotlivé segmenty výstupního spektrogramu generují sekvenčně. To přináší nevýhody, jako jsou delší čas generování, možnost kaskádní propagace chyby či občasné problémy se synchronizací (např. selhání pozornostních mechanismů). Vstupem je text a výstupem mel-spektrogram [12].

Kromě zde uvedených modelů jsou hojně užívány i další, např. Glow-TTS a VITS.

2.3 Využití strojově generované řeči

Rozvoj nástrojů i technik strojového učení a umělé inteligence přinesl dosud neuškutečnitelné možnosti nejen v oblasti zpracování řeči, ale i například obrazu, a to i v oblastech syntézy a úprav. Tyto moderní nástroje jsou již schopny velmi věrně generovat řeč, která je téměř nerozlišitelná od přirozené, nebo se jí výrazně podobá a není pro člověka nepřijemná k poslechu. Této schopnosti se využívá v mnoha rozličných aplikacích, zejména v komunikaci mezi strojem a člověkem. Mezi tyto aplikace patří například předčítání textu či obsahu webových stránek pro osoby se zrakovým postižením [13], komunikace zařízení chytrých domácností a mobilních asistentů (Google Home, Alexa aj.), nebo také komunikace hlasových chatbotů, jejichž úkolem je získat informace od zákazníka či mu je poskytnout, než bude spojen s lidským operátorem. Takové využití je poměrně běžné v bankách a podobných institucích [14]. Dalším častým využitím jsou automatické ohlašovací systémy, například na nádražích či letištích.

Mezi další nedávno uvedené aplikace patří také vytváření obsahu, který je pak zveřejňován v televizi či na sociálních sítích na internetu, jelikož tyto systémy nabízejí jednodušší, rychlejší a levnější způsob oproti namlouvání textu lidmi. Další z výhod je možnost mnoha jazykových mutací daného obsahu.

2.4 Zneužití strojově generované či pozměněné řeči

Možnost strojově generovat či pozměnit řeč tak, aby zněla jako řeč konkrétní osoby s sebou přinesla i riziko zneužití. Tato možnost zneužití ještě vzrostla s příchodem systémů, které pro adaptaci na řečníka nepotřebují dlouhý záznam jeho promluv, ale pouze krátkou promluvu [15] [16]. Zpravidla rozlišujeme tři druhy zneužití.

Prvním z nich je cílený telefonní/videotelefonní útok na konkrétní osobu, jehož cílem je oklamat oběť vydáváním se za blízkou osobu (např. kamaráda, nadřízeného, člena rodiny) a přimět ho něco vykonat. Druhým z nich je pak vytvoření deepfaku, nejčastěji i s obrazovou částí, obvykle známé a vlivné osobnosti, který je následně zveřejněn za účelem poškození dané osoby, ovlivnění veřejného mínění či dodání zdání legitimity a pravosti nějakého podvodu. Třetí spočívá v útoku na systémy automatického ověření řečníka (ASV - z anglického *Automatic Speaker Verification*) v bankách i jiných institucích, kde bývají jisté příznaky řeči využity pro biometrické ověření klienta.

Pro první dva typy útoků bývá typické, že neobsahují pouze podvrženou řeč. Mnohdy

bývá přítomno i podvržené video, podvodné emaily a další typy podvodného jednání. Při útocích na společnost mohou být přítomny i botnety sloužící k amplifikaci narativu.

2.4.1 Útoky cílené na konkrétní osobu

Tento typ útoků obvykle cílí na jednotlivce nebo firmy, a to s cílem vylákat peníze. Útočníci se nejčastěji vydávají za vysoké představitele firem a požadují po svých "podřízených či partnerech" převod peněz.

Toto dokládá případ z roku 2019 z Anglie, kdy útočníci uskutečnili telefonát s CEO jedné energetické firmy a vydávali se za CEO její mateřské firmy. Hlas útočníka byl pozměněn či syntetizován tak, aby zněl jako jeho hlas. Útočníkům se podařilo z firmy vylákat 243 000 USD [17].

Další případ se odehrál v roce 2020 v Hong Kongu. Zde se útočníci zavolali manažerovi jedné japonské firmy. Hlas v telefonu zněl jako hlas ředitele mateřské firmy. "Ředitel" informoval manažera o nových akvizicích firmy a nutnosti přeposlat 35 000 000 USD. Pro dodání věrohodnosti byly součástí útoku i podvodné emaily. Z odeslaných peněz se podařilo vystopovat jen asi 400 000 USD [18].

Ne vždy jsou útoky úspěšné. Jedním z těchto neúspěšných pokusů je i případ zneužití identity CEO jedné z největších reklamních agentur WPP Marka Readea. V tomto případě útočníci založili WhatsApp účet, který vypadal jako Readův. Z něj následně sjednali online setkání na Microsoft Teams s vedoucími představiteli firmy, ohledně nových obchodních příležitostí, za účelem odcizení peněz a osobních údajů. V tomto útoku sehrál důležitou roli nástroj hlasové konverze (VC - z anglického *Voice Conversion*) [19].

2.4.2 Podvržená řeč známých a vlivných osob

Prvním z důvodů takového podvodu bývá poškození dobrého jména exponované osoby. Motivace v tomto případě může být osobní, politická, potenciálně i ekonomická. Jedním z příkladů politické motivace je podvržená nahrávka, ve které se údajný předseda strany Progresívne Slovensko Michal Šimečka baví s údajnou novinářkou Monikou Tódovou a řeší spolu, jak ovlivní volby [20][21].

Dalším obětí tohoto typu podvrhu se stal Keir Starmer (v té době předseda Labouristů). Podvodná nahrávka byla zveřejněna první den sjezdu strany v Liverpoolu. Na této podvodné nahrávce je zachyceno (resp. vytvořeno), jak Starmer ponižuje své

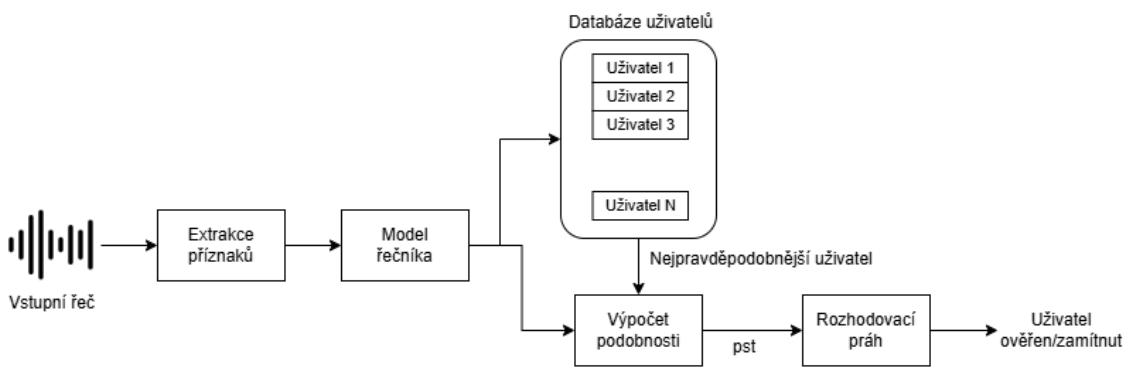
zaměstnance a vyjadřuje se velmi kriticky k Liverpoolu, kde se sjezd konal. Nahrávka byla přehrána více než milionkrát [22].

Dalším z důvodů může být snaha dodat zdání důvěryhodnosti nějakému podvodnému projektu, cílcímu na občany. Jedním z příkladů může být zneužití identity moderátorek CNN Prima News Pavlíny Wolfové a Kateřiny Hošek Štruncové či reportérů Jakuba Kokošky a Ondřeje Němce. V podvržených (deepfake) videích propagují tito novináři různé podvodné investiční projekty slibující veliké zisky za minimální peníze [23]. Dalším z příkladů může být zneužití identity UFC zápasníka Jiřího Procházky. V tomto podvodném videu údajný Procházka nabízí 50 milionů korun prostřednictvím své sázkové aplikace Perfect Game [24]. V obou případech hrála podvržená řeč zásadní roli. Podvod cílený na mnohem větší množství lidí zneužil identitu youtubera MrBeasta, tvůrce cílcího především na mladší diváky (děti a teenageři). V tomto podvodném klipu nabízí údajný MrBeast rozdávání nových Iphonů 15 pouze za 2 dolary [25].

Kvalita videí u takovýchto podvodů mnohdy nebývá valná a člověk znalý velice rychle odhalí, že s videem je něco v nepořádku. Audio bývá často v lepší kvalitě, ale taktéž jsou mnohdy slyšitelné artefakty: řeč nemá přirozené tempo, je monotónější atd. Přes toto všechno si podvody najdou svůj cíl a mnoho lidí je není schopno rozlišit, resp. ani nepřemýšlí, že by se mohlo jednat o podvod. Důvody, proč tomu tak je a jak to zlepšit jsou však tématem pro jiný obor.

2.4.3 Útoky cílené na ASV systémy

ASV jsou biometrické systémy určené k autorizaci uživatele podle jeho hlasu. Skládají se ze dvou fází. První je zadání uživatele do databáze, tedy zaznamenání hlasu a jeho parametrů a jejich uložení. Druhou fází je pak autorizace uživatele při přihlášení do systému. Autorizace se skládá ze dvou kroků: identifikace řečníka (speaker identification) a ověření řečníka (speaker verification). Identifikace spočívá v porovnání hlasu osoby snažící se přihlásit do systému s uloženými uživateli a výběru uživatele s nejpodobnějšími charakteristikami hlasu. Ověření standardně probíhá určením míry shody mezi hlasem osoby snažící se přihlásit a vybraného uživatele z databáze, například pravděpodobnosti, že se opravdu jedná o daného člověka [26].



Obrázek 2.1: Principiální schéma ASV

První podstatnou záležitostí pro fungování ASV systémů je volba příznaků, tj. parametrů popisujících řeč daného člověka. Tyto příznaky by měly zahrnovat fyzikální vlastnosti hlasového traktu i naučené řečové charakteristiky. Dalším požadavkem je textová nezávislost. Mezi rozšířené příznaky patří: melovské kepstrální koeficienty (MFCC) [27], lineárně-frekvenční kepstrální koeficienty (LFCC) [28], v posledních letech také: konstantní Q kepstrální koeficienty (CQCC) [29], gammatónové kepstrální koeficienty (GTCC) [30] či jiné nekepstrální příznaky.

Dalším důležitým bodem je volba modelu řečníka (speaker embedding). Model řečníka se vytváří na základě extrahovaných příznaků při registraci uživatele do systému. Úkolem této části je převod příznaků, které mají různou velikost kvůli délce vstupní promluvy, na reprezentaci pevné délky, popisující řečníka. Promluvy stejného řečníka jsou si blízké. Dříve býval nejrozšířenější model založen na Gausovských směsích (GMM - z anglického *Gaussian Mixture Model*) [31], [32]. Později se začala využívat metoda podpůrných vektorů (SVM - z anglického *Support Vector Machines*) [33] a GMM-UBM (i-vector). V dnešní době jsou rozšířené systémy založené na neuronových sítích např. na DNN (x-vektory) [34].

Posledním krokem je pak samotné ověření řečníka tzv. skórování. V této části jsou spolu porovnávány modely řečníků z databáze a model testovacího řečníka. Výstupem je skóre (pravděpodobnost). Na základě prahu pro skóre je následně rozhodnuto, zda se jedná o daného řečníka z databáze či nikoliv. Skóre může být vypočteno např. jako rozdíl logaritmických pravděpodobností, kosinovou vzdáleností (*cosine similarity*), či pomocí pravděpodobnostní lineární diskriminativní analýzy (PLDA - z anglického *Probabilistic Linear Discriminant Analysis*) [35]. Tomuto kroku také mnohdy předchází redukce dimensionality za pomocí lineární diskriminační analýzy (LDA - z anglického *Linear Discriminant Analysis*).

Podobně jako jiné biometrické systémy, jsou i ASV náchylné na útoky. Útoky lze v základu rozdělit na 3 kategorie: Black box, White box a Gray box. V případě black box útoku nemá útočník žádné apriorní informace o fungování konkrétního ASV systému. Do této kategorie řadíme hardwareové útoky, **útoky s podvrženým hlasem**, a útoky na stabilitu systému. Jedná se o přímé útoky. Opakem je white box, kdy útočník má kompletní znalost a přístup do systému. Tento typ útoku je však málo pravděpodobný, pravděpodobnější je gray box tj. částečná znalost a přístup do systému a subsystémů. Mezi gray box útoky patří i útok s předvybráním oběti, kdy útočník s částečným přístupem do systému extrahuje informace o řečnících a s využitím podobné struktury ASV, jakou je napadená, vybere toho, na kterého se mu bude nejlépe útočit pomocí podvrženého hlasu. Útoků s různou mírou znalosti a přístupu do systému existuje mnoho, pro tuto práci jsou mnohé z nich irelevantní, a jsou zde uvedeny jen pro ilustraci problematiky a nebudou zde dále rozváděny [36].

Black box útoky s podvrženým hlasem lze rozdělit do tří základních typů. Prvním je prostá imitace člověkem. Náchylnost ASV systémů k tomuto útoku je však nejistá. Dalším a nejjednodušším způsobem je nahrání hlasu cílového řečníka a jeho následné přehrání ASV systému. Posledním je využití nástrojů TTS a VC.

Dále se útoky dělí na útoky virtuálního přístupu (LA - z anglického *Logical Access*) a fyzického přístupu (PA - z anglického *Physical Access*). V LA scénáři je počítáno s tím, že útočník nemusí pro zadání hlasu do ASV systému využít mikrofon, postačí mu soubor s nahrávkou hlasu ať už reálného nebo syntetického. Toto bývá obvykle v případech že ASV systém je vzdálený a ke komunikaci s ním je využito digitálního přenosového kanálu (telefon, VOIP). V PA scénáři musí útočník podvrženou nahrávku přehrát do mikrofonu ASV systému či do mikrofonu, který je na vstupu digitálního přenosového kanálu, jestliže se do něj není schopen dostat přímo s digitální nahrávkou. V tomto případě je nutné uvažovat charakteristiky mikrofonu a reproduktoru, ze kterého je podvrh přehráván, i prostředí. Tyto útoky jsou špatně detekovatelné, k jejich provedení není třeba technických znalostí a jsou jednoduché

na realizaci [37].

Samotné ASV systémy jsou schopné část útoků odfiltrovat a rovnou dané pokusy zamítnout. Toho je dosaženo již ve fázi trénování ASV systému a přidávání uživatelů. Systému jsou poskytnuty různé falešné nahrávky z různých nástrojů generování či úpravy hlasu i od různých mluvčích a jsou v systému označeny jako falešné. Z toho plyne, že pokud bude minimální vzdálenost modelu testovacího mluvčího a modelu jedné z podvržených ukázek, na jaké byl model trénován, dojde k zamítnutí. Toto řešení ovšem není optimální, jelikož je schopné detektovat útoky pouze pro některé typy hlasů či konkrétní typy útoků, na které byl systém trénován, což je v době rozšíření různých přístupů a metod strojově generované řeči nedostačující v důsledku nedostatečné generalizace.

3 Detekce podvržené řeči

Jak bylo zmíněno v předchozí kapitole, podvržená řeč představuje nebezpečný jev, který může způsobit značné finanční, politické či jiné škody, a je proto klíčové umět ji odlišit od řeči přirozené.

3.1 Systémy detekce

Detekce podvržené řeči je poměrně náročný úkol, jelikož detekční systémy musí být schopny pracovat s velikou škálou mluvčích a mnoha druhů útoků, které jsou pro systém obvykle neznámé, tj. systém musí umět dobře generalizovat. Systémy pracující pouze s manuálním nastavováním mezí pro příznaky se již ukázaly jako nedostačující. V současnosti se pro detekci využívají výhradně systémy strojového učení.

3.1.1 Jednodušší systémy

Jednodušší systémy jsou založeny na gaussovských směsích (GMM), kde jsou příznaky (např. LFCC, MFCC) použity pro natrénování dvou modelů. Jeden z modelů je naučen z příznaků přirozené řeči, druhý pak z příznaků podvržené řeči. Příznaky testovaného signálu jsou následně prezentovány oběma modelům, jejichž výstupem bývá logaritmická pravděpodobnost. Tyto pravděpodobnosti se porovnají a model s vyšší hodnotou je považován za odpovídající testovanému signálu [38].

Z jednoduchých klasifikačních algoritmů se také v malé míře využívají i SVM (z anglického *Support Vector Machines*), konkrétně ve variantě One Class SVM. Při trénování klasifikační hranice je zde kladen důraz na učení se pouze z jedné klasifikační třídy - přirozené řeči. Vše, co nezapadá do této kategorie, je označeno za kategorii druhou. Toto představuje rozdíl oproti tradičním SVM, které rozhodovací hranici budují rovnoměrně pro obě třídy [39].

3.1.2 Systémy založené na CNN

Další kategorií systémů představují modely založené na konvolučních neuronových sítích (CNN). Jejich cílem je nalezení prostorových rozdílů mezi reálnými a podvrženými promluvami. Časové změny v příznacích jsou v těchto systémech převedeny na prostorové. Silnou stránkou konvolučních vrstev je schopnost nalézt reprezentativní příznaky ze vstupních dat, které umožní klasifikaci. Vstupy do těchto modelů bývají nejčastěji spektrogramy, kepstrogramy (např. LFCC, MFCC) nebo Q-kepstrální koeficienty. Velmi často se rovněž používají příznaky získané pomocí modelu Wav2Vec

2.0 [40]. Nejpoužívanější z této architektury jsou struktury ResNet. Tyto potenciálně hluboké neuronové sítě těží z přemostění mezi jednotlivými bloky, což vede ke snížení rizika vymizení či exploze gradientu a sít tak může být hlubší. Modely ResNet mohou mít různé architektury. Může se jednat o varianty sítí původně navržených pro klasifikaci obrazu, jako jsou ResNet-18, ResNet-34, ResNet-50 a další [41] [42]. Také se může jednat o menší a jednodušší struktury obsahující pouze několik residiuálních bloků [43].

3.1.3 Systémy zachycující časové změny

Jiná kategorie systémů nepřevádí časové změny na prostorové, ale využívá svých vnitřních pamětí pro zachycení časových změn. Mezi tyto sítě patří zejména modely založené na rekurentních neuronových sítích (RNN), především pak na jejich variantě LSTM. LSTM dokážou oproti RNN zachytit delší časový kontext a nejsou tak náchylné k problému mizícího gradientu. Příklady využití RNN a LSTM viz [44], [45].

3.1.4 Kombinované systémy

Dalším z možných přístupů je kombinace více kategorií do jednoho systému. Jednou z častých kombinací je spojení LCNN a LSTM tedy lehké konvoluční neuronové sítě a LSTM. Tato struktura může být výhodná díky kombinaci předností obou přístupů. LCNN slouží pro extrakci vysokoreprezentativních příznaků ze vstupních příznaků (např. spektrogramů) a LSTM v nich následně detekuje časové změny [46].

Kromě uvedených systémů byly prováděny experimenty i s dalšími strukturami a jejich kombinacemi. Většina těchto systémů pracuje s manuálně vypočítanými příznaky, jako jsou spektra, kepstra a další, některé systémy však pracují přímo s audiosignálem - tzv. Raw-wave systémy, které fungují end-to-end a extrakce relevantních příznaků probíhá uvnitř sítě [47].

3.2 Zvolené struktury

Tato sekce popisuje dvě zvolené architektury ResNet pro detekci podvržené řeči. První z použitých architektur je menší, jednodušší a byla zvolena pro úvodní experimenty, jejichž cílem bylo seznámit autora práce se základními principy fungování CNN a ResNet, s dostupnými knihovnami, nástroji a postupy trénování, testování i optimalizace. Druhá struktura je komplexnější a tvoří jádro práce. Obě zvolené struktury předpokládají jednokanálový dvoudimenzionální vstup zachycující příznaky i jejich časové změny, např. spektrogram či kepstrogram.

3.2.1 Jednoduchý ResNet

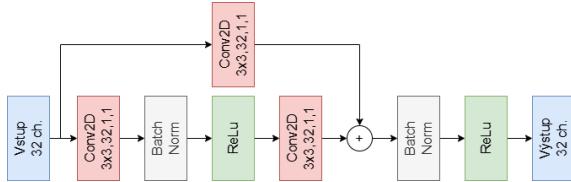
Tento jednoduchý ResNet se v základní konfiguraci skládá ze vstupní konvoluční vrstvy, dvou reziduálních bloků, dropout vrstvy a plně propojené vrstvy s 800 neurony. Výstupem jsou pak dva neurony pro klasifikaci, z jejichž hodnot je počítána pravděpodobnost příslušnosti k dané třídě (přirozené/podvržené). Struktura reziduálního bloku je znázorněna na obrázku 3.1b. Normalizace přes dávku (*Batch Normalisation*) pomáhá se stabilizací trénování a přispívá k vyšší efektivitě, jelikož redukuje kovariacioní posun. Dropout vrstvy, které při trénování náhodně vypouštějí zadané procento spojů, napomáhají síti s generalizací. Aktivační funkce ReLU přispívá k zavedení nonlinearity do modelu. Konvoluční vrstva v přemostění pouze srovnává počet kanálů a dimenze tak, aby bylo možné obě větve sečíst. Pro takovouto redukci dimenzionality existuje několik přístupů.

Počet bloků i velikost konvolučních jader byly v průběhu experimentů s touto strukturou měněny. Struktura je autorská s inspirací v [43].

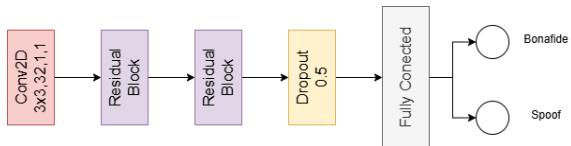
3.2.2 ResNet + OC Softmax

Po zevrubné rešerši existujících modelů byla pro experimenty zvolena struktura ResNet + OC-Softmax [42]. Ta se skládá ze dvou základních komponent. ResNet slouží jako extraktor příznaků zatímco OC-Softmax zajišťuje skórování a klasifikaci.

Využitý ResNet vychází ze struktury ResNet 18. Na vstupu se předpokládají jednokanálové časově-příznakové reprezentace, např. kepstrogram ve formátu [B 1 F T], kde B je velikost dávky (batch size), T označuje čas (pořadí rámců), F značí frekvence (koeffice) a 1 odpovídá jednomu kanálu. Základní struktura sítě je následující:



(a) Schéma základního reziduálního bloku. Konvoluci vrstvy: Jádro 3x3, 32 kanálů, Stride 1, Padding 1. Konvoluce v přemostění není v tomto nastavení nutná, později poslouží pro srovnání dimenzí. Možno nahradit převzorkováním.



(b) Schéma základní sítě

Obrázek 3.1: Základní struktura použité sítě

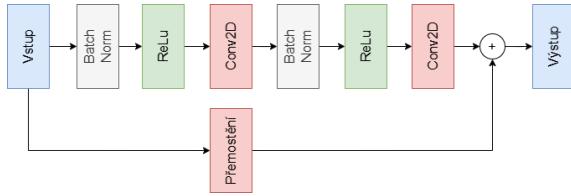
1. Vstupní konvoluce
2. Residuální struktura s preaktivací
3. Koncová konvoluce - Bottleneck
4. Self-Attention Pooling Layer
5. Příznaková vrstva

Vstupní konvoluce

Vstupní 2D konvoluce s parametry: kernel = (9, 3), stride = (3, 1), padding = (1, 1) tvoří 16 kanálů a jejím hlavním účelem je snížení počtu kepstrálních koeficientů, tedy podvzorkování ve frekvenční ose.

Residuální struktura s preaktivací

Základními stavebními kameny jsou residuální bloky s preaktivací. Oproti standardním residuálním blokům představeným v [41], jsou v preaktivacích blocích aplikovány normalizace přes dávku (Batch Normalisation) a ReLU aktivační funkce před konvolucí, což zajišťuje lepší tok gradientu sítí, pomáhá s regularizací a zrychluje učení sítě [48]. Výsledný blok pak má strukturu zobrazenou na obrázku 3.2. Přemostění může mít formu identity, či 1x1 konvoluce v případě nutnosti úpravy velikosti.



Obrázek 3.2: Základní struktura preaktivitačního ResNet bloku

Samotná reziduální struktura se skládá ze čtyř vrstev, kde každá obsahuje 2 bloky, jak je znázorněno v tabulce 3.1. Velikost konvolučního jádra je pro všechny bloky 3 x 3.

Vrstva	Počet kanálů	Stride	Výstupní tvar
1	64	1	(B, 64, 18, 750)
2	128	2	(B, 128, 9, 375)
3	256	2	(B, 256, 5, 188)
4	512	2	(B, 512, 3, 94)

Tabulka 3.1: Reziduální struktura. Výsledné tvary počítány z v práci dále používané velikosti vstupních příznaků 60 x 750. B odpovídá velikosti dávky

Koncová konvoluce - Bottleneck

Tato vrstva je navržená pro redukci dimenziality a kolabuje "frekvenční" rovinu na velikost 1, pro libovolnou velikost vstupních příznaků. Tato dimenze je následně odstraněna (`squeeze()`). Výstupní počet kanálů je 256. Vrstva slouží jako úzké hrdlo (bottleneck) sítě, který redukuje dimenzialitu dat před self - attention vrstvou, což snižuje výpočetní náročnost, zlepšuje generalizaci a umožňuje vrstvě se soustředit pouze na časové změny. Pro velikost příznaků: (60, 750) <-> (B, 1, 60, 750) -> Reziduální struktura -> (B, 512, 3, 94) -> Bottleneck -> (B, 256, 94) = (B, C, T')

Self Attention Pooling

V této části dochází nejprve k permutaci vektoru z (B, C, T') na (B, T', C), tj. každý časový okamžik je reprezentován vektorem velikosti C (256) a je možné jej zapsat jako $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{T'}] \in \mathbb{R}^{T' \times C}$, kde x_t odpovídá jednotlivým vektorům z R^C v čase t .

Dále je zaveden trénovatelný vektor $\mathbf{w} \in \mathbb{R}^C$, který slouží pro výpočet skóre e_t v každém časovém okamžiku.

$$e_t = \tanh(\mathbf{x}_t^\top \mathbf{w}) \quad (3.1)$$

Vektor \mathbf{w} slouží pro výběr důležitých příznaků - důležitější příznaky ve všech časech obdrží vyšší hodnotu w_c , a tím se více projeví v e_t . Hyperbolický tangens slouží

pro limitaci velkých hodnot, výsledné hodnoty jsou pak z intervalu [-1, 1] a zavádí nelinearitu do výpočtu.

Z e_t jsou následně spočteny váhy α_t pomocí softmax funkce:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{s=1}^{T'} \exp(e_s)}, \quad (3.2)$$

která převádí původní skóre na pravděpodobnosti a zajišťuje jejich umístění v intervalu [0,1] a $\sum_{t=1}^{T'} \alpha_t = 1$. Důležitějším časovým okamžikům je přiřazena větší váha, méně důležitým pak menší.

Tyto váhy jsou využity k výpočtu váženého průměru a vážené směrodatné odchylky pro jednotlivé koeficienty (kanály) přes čas:

$$\mu = \sum_{t=1}^{T'} \alpha_t \mathbf{x}_t \quad (3.3)$$

$$\sigma = \sqrt{\sum_{t=1}^T \alpha_t (\mathbf{x}_t - \mu)^2 + \epsilon}, \quad (3.4)$$

kde ϵ je malá konstanta pro numerickou stabilitu. Takto je možné pomocí dvou vektorů reprezentovat hodnoty a časové změny v jednotlivých kanálech. Tato reprezentace nemá mnoho dimenzí a dokáže rozlišit, které z kanálů jsou stabilní a které v důležitých časových úsecích obsahují nekonzistentní vzory, či jiné irregularity, které mohou být způsobeny nástroji TTS nebo VC. Výsledné vektory jsou následně spojeny a výstupní vektor má tvar (B, 2C). Struktura byla přejata z [49].

Příznaková vrstva

Poslední částí struktury je plně propojená lineární vrstva s 512 neurony, která má za cíl redukci dimensionality, pokud by byla bottleneck vrstva jinak nastavena, a zároveň aplikuje projekci výstupu z pooling vrstvy do výstupního příznakového vektoru. Toto je velmi důležité, jelikož takováto projekce může podchytit kombinace jednotlivých μ_c a σ_c , které mohou mít větší diskriminační schopnost než jednotlivé koeficienty.

OC Softmax

Standardní klasifikační struktury NN obsahují za poslední vrstvou ještě klasifikační dopřednou vrstvu s počtem neuronů odpovídajících počtu klasifikačních kategorií, v tomto případě dva: pro přirozené a podvržené. Tento přístup, stejně jako jiné metody založené na binární klasifikaci, selhává v generalizaci na neviděné typy útoků či nahrávací podmínky. Toto je nejspíše způsobeno tím, že binární klasifikátory předpokládají stejné distribuce příznakových vektorů u přirozených a podvržených

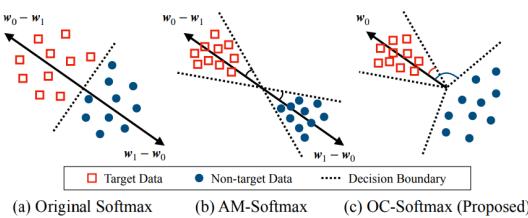
promluv v trénovacím i testovacím datasetu. U přirozených promluv lze tento předpoklad považovat za přibližně platný, pro podvržené promluvy nikoliv. Z tohoto důvodu se jako možné řešení jeví jednotříďová klasifikace (*one-class classification*).

Zde se vychází z předpokladu, že jedna z klasifikačních skupin (přirozené promluvy) má blízkou (ideálně stejnou) distribuci v obou datasetech. Cílem je zachytit tuto distribuci a nastavit její ohraničení tak, aby vzorky z ostatních (viděných i neviděných) skupin ležely mimo tuto hranici. Tento přístup byl aplikován pro detekci podvrženého obrazu [50] i pro detekci podvržené řeči [39] s využitím jiných struktur i klasifikačních funkcí.

U tradičních binárních klasifikátorů založených na neuronových sítích s využitím softmax funkce jsou výstupní příznaky tlačeny na opačné strany a mezi nimi je dána pevná hranice. Takovýto přístup je velice náchylný na neviděně typy útoků.

Vylepšení přináší AM softmax [51], která nevyužívá jednu pevně danou hranici, ale zavádí úhlový prostor, ve kterém se mohou dané klasifikační kategorie vyskytovat. Tyto prostory jsou naproti sobě a jsou oba dva stejně úhlově velké. Jednotlivé příznakové vektory jsou tlačeny do jedné či druhé oblasti, mezi nimiž je prostor. Jelikož trénování probíhá na datech z obou klasifikačních kategorií a oba prostory jsou stejně úhlově velké, může tento přístup taktéž vést ke zhoršené generalizaci.

OC Softmax oproti tomu zavádí dvě velikosti úhlových prostorů - jeden pro cílovou kategorii (přirozené promluvy), druhý pro necílovou (podvržené promluvy). Pro výpočet jsou využity pouze váhy z první kategorie. Tím vzniká poměrně kompaktní prostor pro přirozené promluvy a širší prostor pro podvržené promluvy, vedoucí k lepší generalizaci na neznámé útoky.



Obrázek 3.3: Porovnání projekcí příznaků při využití Softmax, AM Softmax a OC softmax.
Přejato z [42]

V praxi se toho dosahuje následujícím způsobem. Pro zjednodušení uvažujme dávku o velikosti jedna, výstupem z ResNetu je vektor $\mathbf{x} \in \mathbb{R}^{512}$ s libovolným směrem a velikostí. Na tento vektor je aplikována L2 normalizace, která zajistí, že $\|\mathbf{x}\| = 1$. Vstupní vektory jsou tak mapovány na jednotkovou hypersféru. Při porovnávání s centrálním vektorem, který určuje střed oblasti přirozené mluvy, není tedy třeba

počítat s velikostí a směrem vektorů, ale stačí porovnávat pouze jejich směry, jelikož středový vektor $\mathbf{w} \in \mathbb{R}^{512}$ je také normován. Středový vektor je na začátku trénování inicializován pomocí Kaimingova uniformního (He-uniformního) rozložení a jeho hodnoty jsou trénovatelným parametrem. Tento vektor je v každé iteraci normován, aby zůstal na jednotkové hypersféře.

Samotné porovnání úhlů je provedeno pomocí kosinové podobnosti, což je velmi výhodné pro jednotkovou hypersféru, jelikož její výpočet

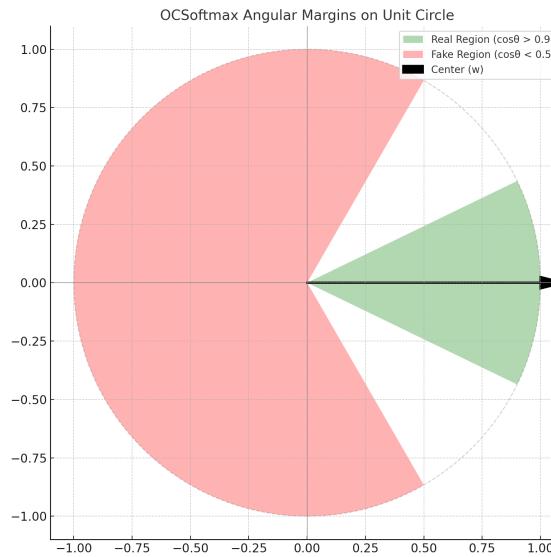
$$\cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{w}^\top}{\|\mathbf{x}\| \|\mathbf{w}^\top\|} \quad (3.5)$$

přechází pouze na

$$\cos(\theta) = \mathbf{x} \cdot \mathbf{w}^\top. \quad (3.6)$$

Kosinová podobnost nabývá hodnot od -1 pro vektory opačných směrů do 1 pro vektory ve stejném směru.

Dalšími vstupními parametry OC Softmax jsou hodnoty úhlů pro jednotlivé oblasti. Pro přirozené promluvy je zadán maximální úhel, pro podvržené pak minimální úhel. Tyto hodnoty jsou v kódu zadány jako kosinové podobnosti. Ilustrace ve 2D prostoru pro $r_{real} = \cos(\theta_{real}) = 0.9$, $r_{fake} = \cos(\theta_{fake}) = 0.5$ a pro vektor \mathbf{w} ve směru [1, 0] je znázorněna na obrázku 3.4.



Obrázek 3.4: Znázornění oblastí přirozených a podvržených promluv ve 2D

Při trénování je s ohledem na pevně dané úhlové meze počítáno zvlášť pro přirozené a zvlášť pro podvržené promluvy. Přirozené promluvy by měly být tlačeny do oblasti přirozených promluv, tj. do zadaného úhlu od středového vektoru, tak aby $\cos(\theta) \rightarrow 1$

resp. $\cos(\theta) \in [r_{real}, 1]$. Toho je dosaženo pomocí:

$$\text{score} = r_{real} - \cos(\theta) \quad (3.7)$$

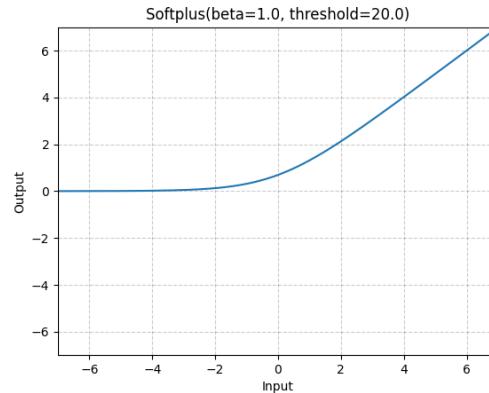
Jestliže se $\cos(\theta) \rightarrow r_{real}$, či dokonce $\cos(\theta) > r_{real}$ bude skóre blízké nule (či záporné) a následná penalizace bude nižší.

Podvržené promluvy by měly být tlačeny do oblasti podvržených promluv, tak aby $\cos(\theta) \in [-1, r_{fake}]$. Toho je obdobně jako v předchozím případě dosaženo pomocí:

$$\text{score} = \cos(\theta) - r_{fake}, \quad (3.8)$$

kde r_{fake} slouží jako vrchní mez. Cílem je opět kladné skóre blízké nule nebo záporné.

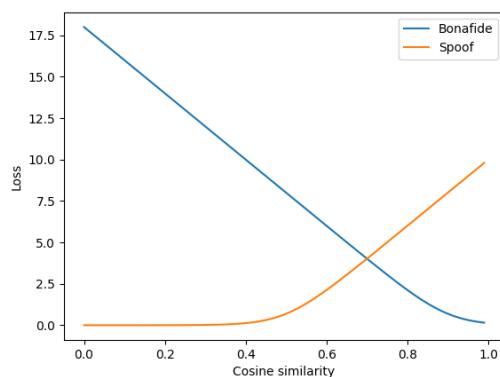
Skóre jsou následně převedena na ztrátu pomocí Softplus funkce, která jen mírně penalizuje nízká skóre, záporná skóre nuluje a penalizuje vyšší kladná skóre - podobné chování jako ReLu funkce. Typická podoba funkce Softplus je znázorněna na obrázku 3.5. Pro zvýšení efektu jsou před vstupem do funkce skóre násobeny parametrem $\alpha > 1$.



Obrázek 3.5: Ilustrace Softplus funkce. Přejato z [52]

Výsledný průběh ztrátové funkce v závislosti na kosinové podobnosti pro přirozené a podvržené promluvy je znázorněn na obrázku 3.6.

Při testování, tedy při normálním běhu, jsou jako výstup využity přímo kosinové podobnosti.



Obrázek 3.6: Průběhy ztrátových funkcí pro přirozené a podvržené promluvy pro $r_{real} = 0.9$, $r_{fake} = 0.5$, $\alpha = 20$

3.3 Dostupné datasety

Existuje několik datasetů sloužících ke trénování systémů pro detekci uměle syntetizované řeči, ať už pro ASV systémy či pro jiná využití. V průběhu let databází přibývá a obsahují stále více nástrojů TTS a VC, což umožňuje detekčním systémům lépe generalizovat. Problémem však zůstává nedostatečná jazyková rozmanitost, jak je patrné z tabulky 3.2. Většina datasetů je v angličtině nebo čínštině a systémy natrénované na těchto jazycích nemusí být optimální pro jiné jazyky.

Název	Jazyk	Generujících systémů	Počet promluv
ASVspoof15 [53]	Angličtina	10	263,151
ASVspoof19 LA [37]	Angličtina	19	121,461
ASVspoof21 LA [54]	Angličtina	13	164,612
ASVspoof21 DF [54]	Angličtina	100+	593,253
FakeAVCeleb [55]	Angličtina	1	11,857
FoR [56]	Angličtina	7	195,541
Voc.v [57]	Angličtina	8	82,048
In-The-Wild [58]	Angličtina	?	31,779
PartialSpoof [59]	Angličtina	19	121,461
WaveFake [60]	Angličtina, Japonština	9	136,085
ADD2022 [61]	Čínština	?	493,123
ADD2023 [62]	Čínština	?	517,068
FMFCC-A [63]	Čínština	13	50,000
HAD [64]	Čínština	2	160,836
CFAD [65]	Čínština	12	347,400
MLAAD [66]	38	82	154,000

Tabulka 3.2: Dostupné datasety. Přejato a přeloženo z [66]

Níže je uveden detailnější popis jednoho z anglických datasetů: ASV Spoof 2019, který byl vybrán pro prvotní experimenty v této práci.

3.3.1 ASV Spoof dataset

První z této série datasetů vznikl v roce 2015 při příležitosti první ASV Spoof Challenge, která si klade za cíl vývoj co nejlepší metody detekce podvržené řeči pro ASV systémy [67]. Nejnovějším kompletním datasetem je ASV Spoof Dataset 2019, který obsahuje data pro trénování, ladění i testování navrženého systému. Posledním datasetem je pak ASV Spoof 2021 dataset, který v LA scénáři primárně slouží pro testování modelů natrénovaných na předchozím datasetu (2019). Tento dataset obsahuje také s Deepfake scénář, kde nahrávky vznikají podobně jako v LA scénáři, ovšem s větším počtem systémů a aktuálnějšími metodami.

ASV Spoof Dataset je složen ze dvou hlavních částí: PA scénáře a LA scénáře. PA se soustředí na útoky přehráním, kdy útočník má k dispozici nahrávku cíle svého útoku a přehraje je jí na vstup (mikrofon) ASV systému. Tato část datasetu využívá různá reprodukční zařízení a simulované RIR. Pro tuto práci není PA scénář důležitý.

LA scénář obsahuje útoky pomocí strojově generované nebo konvertované řeči. Dataset zahrnuje přirozené promluvy i promluvy generované různými nástroji TTS a VC.

LA scénář

Tato část obsahuje celkem 121 461 promluv (vět), které jsou jednotlivě uloženy jako **.flac** soubory do tří složek: trénovací, vývojové a hodnotící (testovací). Uměle vytvořené promluvy jsou utvořeny z přirozených promluv z databáze pomocí 19 systémů. Šest z těchto systémů je považováno za známé a slouží pro trénování a vývoj, 13 ostatních jsou neznámé a slouží pro hodnocení výkonu rozpoznávacího systému a jeho schopnosti generalizace. Základní rozdělení systémů je znázorněno v tabulce 3.3. Přirozené promluvy byly získány od 107 řečníků (46 mužů, 61 žen).

Dataset neobsahuje pouze soubory s promluvami, ale i složky s pomocnými **.txt** soubory, které obsahují informace o jednotlivých souborech a lze je využít pro trénování i následné testování či statistiky.

CM protocols

Tři soubory pro trénování, vývoj a testování, které obsahují základní informace o promluvách. Každý ze souborů obsahuje 5 sloupců. První sloupec je kód řečníka, druhý kód promluvy (název souboru promluvy bez přípony), třetí je zde nevyužit (pouze -), čtvrtý popisuje systém, kterým promluva vznikla ("-" - přirozená řeč, A01-A19 typ útoku), pátý pak obsahuje základní informaci, zda je promluva přirozená (bonafide), či podvržená (spoof).

ASV protocols

Soubory v této složce jsou rozděleny podle několika kritérií a jsou uloženy jako ASVspeek2019.LA.asv.<1>.<2>.<3>.txt, kde

1. obsahuje buď **dev** nebo **eval** v závislosti na tom, zda soubor popisuje vývojové či evaluační promluvy
2. je buď **male (m)** nebo **female (f)** či **gender independent (gi)**. Toto rozděluje řečníky (promluvy) na muže a ženy. V gender independent souborech jsou za sebe seřazeny napřed mužské, následně ženské promluvy.

	Systém	Typ	Popis
Známé	A01	TTS	Neural waveform model
	A02	TTS	Vocoder
	A03	TTS	Vocoder
	A04	TTS	Waveform concatenation
	A05	VC	Vocoder
	A06	VC	Spectral filtering
Neznámé	A07	TTS	Vocoder+GAN
	A08	TTS	Neural waveform
	A09	TTS	Vocoder
	A10	TTS	Neural waveform
	A11	TTS	Griffin lim
	A12	TTS	Neural waveform
	A13	TTS/VC	Waveform concatenation+Waveform filtering
	A14	TTS/VC	Vocoder
	A15C	TTS/VC	Neural waveform
	A16	TTS	Waveform concatenation
	A17	VC	Waveform filtering
	A18	VC	Vocoder
	A19	VC	Spectral filtering

Tabulka 3.3: Rozdělení systémů generujících podvrženou řeč, podrobnější popis systémů: [37]

3. obsahuje **trl** - popis promluv, nebo **trn** - rozdělení promluv dle řečníka. Soubor **trl** navíc obsahuje informaci, zda daná promluva, pokud je přirozená, patří do seznamu registrovaných uživatelů ASV systému.

ASV Scores

Tato složka obsahuje dva soubory: jeden pro vývojovou druhý pro evaluační část. Každý ze souborů obsahuje následující informace o jednotlivých promluvách:

1. přirozené (bonafide) a uměle vytvořené (A01 - A19)
2. cílové - target (předkládaný řečník odpovídá tomu z databáze ASV), podvodný - nontarget (mluvčí mimo ASV databázi) a strojově podvržený - spoof.

3.4 Tvorba českého datasetu

Jelikož se tato práce zabývá detekcí podvržené řeči v češtině, bylo nutné vytvořit dataset, který bude obsahovat dostatečné množství přirozených i podvržených promluv, řečníků i nástrojů pro strojové generování a konverzi řeči. Takovýto dataset v době psaní práce neexistuje a je třeba ho sestavit. Datasety obsahující přirozené promluvy v češtině existují a jsou popsány níže, datasety obsahující strojově generované české promluvy však nikoliv. Jediné větší množství strojově generovaných promluv se nachází ve vícejazyčném MLAAD datasetu, vygenerované pomocí nástroje XTTS, obsahuje však jen jednoho mluvčího, což je nedostačující pro cíl práce.

3.4.1 Použité datasety

Pro generování řeči byly využity anotované datasety, obsahující přepisy i kódy řečníka pro jednotlivé promluvy. Kódy řečníků u jednotlivých promluv jsou velmi důležité ze dvou důvodů: První z nich je podstatný pro nástroje hlasové konverze, kde je vhodné zabránit mapování hlasových vlastností z řečníka na sebe sama. Druhým z nich je schopnost některých nástrojů využít jako hlasovou referenci (target speaker) více souborů obsahujících promluvu od jednoho řečníka a tudíž je nutné znát řečníka u každé z promluvy. Přepisy mohou být vhodné také pro výběr ideálně dlouhých promluv, oproti počtu vzorků v promluvě nejsou ovlivněny hlasovými pauzami. Přepisy lze rovněž využít pro zhodnocení kvality konvertovaných promluv – viz níže. Z datasetů byly vybrány promluvy obsahující 8 - 15 slov.

Byly použity následující datasety:

CTU Test: Dataset, který pochází z pracoviště vedoucího práce a který obsahuje 2 169 foneticky bohatých promluv od 43 řečníků. Každá z promluv byla zaznamenána na dva mikrofony - klosový a headsetový. Pro generování řeči byla využita část zaznamenána na klosový mikrofon.

Voxpopuli CS: Jedná se o českou část českou část Voxpopuli datasetu obsahující 62 hodin promluv od 138 řečníků. Tento dataset byl vytvořen z nahrávek z Evropského parlamentu z let 2009 - 2020 [68], [8].

Common Voice: Byla využita česká část tohoto datasetu. Tento dataset vzniká jako crowdsourcingový projekt tj. zadané promluvy jsou nahrávány dobrovolníky v domácích podmínkách. Pro generování řeči byly využity pouze validované promluvy [69].

3.4.2 Použité nástroje

Nástroje TTS a VC, které je možné použít pro tvorbu datasetu, jsou omezeny několika klíčovými podmínkami. První z nich je možnost přijímat hlasové vektory řečníka (*speaker embedings*), popisující hlasové charakteristiky jednotlivých řečníků, tj. modely natrénované na jednoho řečníka byly zavřeny. Další z podmínek platí pro nástroje TTS a pro nástroje VC, které vyžadují textový přepis. Zde je zásadní limitace jazyk, na kterém byly trénovány, jelikož nástroje netrénované na češtině neznají písmena specifická pro češtinu a současně nedokáží syntetizovat česky znějící promluvy. Nástroje pro hlasovou konverzi bez nutnosti přepisu (tj. bez ASR) tímto limitovány nejsou. Obecným hlavním kritériem pro výběr systému je pak schopnost systému přiměřeně dobře simulovat českou výslovnost, prosodii a fonaci. Pro tvorbu datasetu byly vybrány následující nástroje, které dobře pokrývají různé metody syntézy signálů:

XTTS

Jedná se o vícejazyčný model od společnosti Cocqui AI podporující i češtinu. Tento nástroj umožňuje převod textu na řeč s využitím nejen předtrénovaných řečníků, ale i vlastních, a to za pomocí extrakce vektoru řečníka (*speaker embedings*) z jeho zvukového záznamu. Výhodou tohoto modelu je tzv. přímé vložení (*zero-shot*), při kterém se model nemusí učit na parametry řečníka, ale postačí pouze jeho vektor vyextrahovaný z min. 6 sekundového záznamu. Model umožňuje extrakci i z více cílových (referenčních) souborů. Dále také umožňuje přenos mluvčího jednoho jazyka do jazyka jiného.

Free VC

Free VC je model hlasové konverze založený na frameworku VITS. Obsahuje enkodér založený na WavLM a bottleneck vrstvu pro extrakci kontextových informací nezávislých na řečníkovi. Ty jsou extrahovány zvlášt pomocí encodéru řečníka z cílové (target) promluvy. Samotná konverze probíhá na úrovni spektrogramů bez nutnosti textového přepisu. Systém vyžaduje jeden soubor jako referenci.

kNN VC

KNN VC je algoritmicky jednodušší model pro hlasový převod bez nutnosti textového přepisu. Model extrahuje řečové parametry ze zdrojové (source) a cílové (target) promluvy za pomocí modelů HuBERT a WavLM. Každý z rámců ve zdrojové promluvě je následně zaměněn za nejbližší rámec cílový pomocí metody k-nejbližších

sousedů (kNN - k - *Nearest Neighbours*). Signál je následně zrekonstruován pomocí vocodéru.

Seed VC

Seed-VC je zero-shot systém pro převod řeči a zpěvu, který dokáže převést hlas mluvčího na jiný bez nutnosti předchozího tréninku. Využívá obsahový enkodér (např. Whisper, XLS-R, HuBERT) pro extrakci řečového obsahu, enkodér řečníka k získání identity cílového mluvčího z krátkého vzorku a difuzní generátor s vocoderem (uViT a HiFi-GAN nebo BigVGAN) k vytvoření výstupního zvuku. Díky nízké latenci (300 ms) je vhodný pro reálný čas.

Diff-Hier VC

Tento model je založený na dvou difuzních modelech. Prvním z nich je DiffPitch, který generuje f_0 ve stylu cílového řečníka. Následně DiffVoice pomocí vygenerovaného f_0 převádí řeč do podoby s cílovými charakteristikami. Výstupem je Mel-spektrgram, který je následně převeden na signál. Vylepšení kvality zajišťuje i vstupní filtrační enkodér, který umožňuje lepší rozdelení řeči.

3.4.3 Hodnocení kvality generovaných promluv

Pro hodnocení kvality promluv existuje několik metrik, většinou odvozených od MOS (z anglického *Mean Opinion Score*) jako je např. PESQ (z anglického *Perceptual Evaluation of Speech Quality*). Základní nevýhodou je nutnost referenčního signálu, což v případě generování řeči není možné. Tento problém řeší metody jako DNSMOS a podobné které již referenční signál nevyžadují. Všechny tyto metody slouží dobře pro celkové obecné zhodnocení kvality promluvy, ale nedokážou zhodnotit kvalitu českého jazyka v promluvě obsaženého, což je zásadní, jelikož všechny nástroje výše popsané jsou buď vícejazyčné (XTTS) či trénované na anglických datech (zbylé).

Nástroje, které by dokázaly zhodnotit jazykovou kvalitu generovaných promluv a jejich podobnost s přirozeným mluvčím nejsou pro český jazyk k dispozici a jejich vytvoření je mimo rozsah diplomové práce, byl vymyšlen alternativní postup. Nejdříve byl využit český ASR model Whisper-large-v3-czech-cv13 [70] pro přepis vygenerovaných promluv. Předpokladem je, že promluvy s lingvisticky vyšší kvalitou povedou k přesnějšímu přepisu. Tento přepis i přepis referenční jsou následně standardizovány - čísla převedena na slova, odebrány jiné než alfanumerické znaky a všechna písmena převedena na malá. V dalším kroku je spočtena pro oba přepisy perplexita pomocí českého modelu Cze-GPT 2 [71]. Tyto perplexity jsou normalizovány na počet slov

v přepisu a následně vzájemně porovnány formou poměru. Nižší poměr značí větší výskyt pravopisních chyb v přepisu či špatnou volbu slov ASR systémem - tudíž horší kvalitu promluvy. Vyšší poměr než jedna značí, že ASR systém nesprávně porozuměl řečeným slovům a na výstup prezentoval jiná, která však v kontextu perplexity dávají větší smysl. Z těchto důvodů byl nastaven interval pro výběr promluvy do finálního datasetu na 0.2 - 1.2.

3.4.4 Dataset strojově generované řeči

S využitím dostupných nástrojů a databází byl vygenerován následující český dataset, popsaný v tabulce 3.4. S využitím výše popsané metriky byly vybrány promluvy do finálního datasetu SP Spoof CS.

Pro nástroj XTTS bylo pomocí ChatGPT vygenerováno 50 vět sloužících jako zdrojové (source). Pro všechny nástroje byly z datasetů vybrány pouze promluvy o délce 5 - 15 slov. Příliš krátké sekvence jsou nevhodné, příliš dlouhé by byly ve fázi trénování klasifikátoru oříznuty. U některých nástrojů, které to umožňují, bylo použito více cílových promluv, náhodně zvolených, od stejného řečníka vždy pro každou zdrojovou promluvu - viz tabulka 3.4. Ukázky z datasetu jsou dostupné na <https://smidjir5.github.io/speech-spoof-detection-cz/dataset.html>.

Dataset byl následně doplňován o přirozené promluvy dle potřeb jednotlivých experimentů.

Model	Dataset	Referenční promluvy	Generované promluvy	Vybrané promluvy	Délka vybraných promluv [h]
XTTS	Voxpopuli	4	2550	2011	3.10
XTTS	CTU Test	4	2000	1653	2.67
XTTS	CTU Test	10	2000	1395	2.43
Free VC	Voxpopuli	1	2846	2069	2.43
Free VC	CTU Test	1	1801	1256	2.43
kNN VC	CTU Test	5	1801	1134	2.14
kNN VC	CTU Test	10	1801	1209	2.29
kNN VC	Voxpopuli	10	3335	2456	3.59
kNN VC	Common Voice	10	2333	1698	2.69
Seed VC	CTU Test	1	1801	1298	2.50
Seed VC	Common Voice	1	2550	2058	3.27
Diff-HierVC	CTU Test	1	1801	1016	2.12
Diff-HierVC	Voxpopuli	1	2846	1963	2.33
Diff-HierVC	Common Voice	1	3500	2105	3.36

Tabulka 3.4: Vygenerované a vybrané promluvy

3.5 Volba příznaků

Volba vstupních příznaků je zásadní pro detekci podvržené řeči, jestliže není využit systém pracující se samotným signálem [72]. Strojově generovaná, či konvertovaná řeč je mnohdy k nerozeznání od řeči přirozené, a tudíž je nutná volba takových vstupních příznaků NN, které tyto drobné rozdíly (nedokonalost vocoderů, nepřirozená prosodie, chyby ve fázi) dokážou zachytit. Dalším problémem při volbě je nutnost generalizovat na neviděné typy útoků (nástrojů TTS/VC), které mohou zanechávat odlišné formy artefaktů. Dále je třeba brát v potaz variabilitu přirozených promluv způsobenou různými řečníky i nahrávacími podmínkami. Taktéž je třeba zvážit velikost extrahaných příznaků, jelikož příznaky o větších dimenzích vedou k větším modelům NN a delšímu času pro trénování i samotný běh struktury v provozu, což může být v některých aplikacích nežádoucí.

3.5.1 Kepstrum

Vhodnou volbou příznaků pro klasifikaci by mohlo být kepstrum, resp. kepstrogram. Nejjednodušší formou je přímý výpočet kepstra z PSD. V tomto případě je na signál aplikována preemfáze, je vhodně nasegmentován, dále je aplikováno vhodné okno (nejčastěji Hammingovo), zabraňující prosakování ve spektru a zkreslení vzniklému konvoluci segmentu s pravoúhlým oknem a následně je pro segmenty spočtena časofrekvenční transformace (nejčastěji DFT). V dalším kroku je aplikována absolutní hodnota, aby bylo získáno amplitudové spektrum, které již není komplexní a následně je spočteno výkonové spektrum. Na to je aplikován logaritmus a další ortogonální transformace (IDFT, DCT). Výpočet kepstra je tedy následující:

$$c_n = \text{IDFT}(\ln(|\text{DFT}(s[n])|^2)), \quad (3.9)$$

kde c_n je kepstrum a $s[n]$ řečový signál (segment).

Ve zpracování řeči je této nelineární operace využito pro dekorelaci na přenos vokálního traktu a buzení (hlasivek). Řeč lze ve zjednodušené formě chápat jako výsledek konvoluce buzení (hlasivky pro znělé hlásky, šum pro neznělé) a impulzové odezvy hlasového traktu tj.

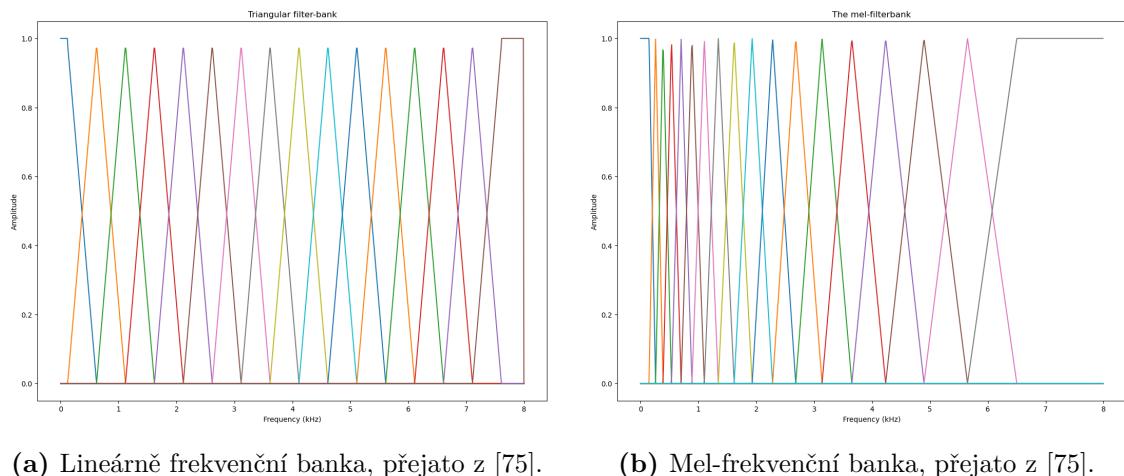
$$x[n] = s[n] * h[n], \quad (3.10)$$

kde $x[n]$ je výsledný řečový signál (segment), $s[n]$ buzení a $h[n]$ impulsová odezva traktu. Koeficienty kepstra s nižším indexem postihují obálku spektra, což odpovídá tvaru vokálního traktu $h[n]$ (formanty atd.), vyšší pak odpovídají buzení $s[n]$ tj. výše hlasu (vibraci hlasivek), či jiné periodicitě.

Spektrum spočtené pomocí DFT je vhodné pro extrakci spektrální obálky či f_0 , ale není efektivní, jelikož obsahuje mnoho koeficientů (stejně jako byl počet koeficientů DFT), které neobsahují mnoho informací, jsou blízké nule. Z tohoto důvodu se využívá bank filtrů (nejčastěji trojúhelníkových) aplikovaných na spektrum, kdy je každý z filtrů vynásoben se spektrem a výsledné hodnoty pro jeden filtr sečteny. Výsledkem jsou koeficienty, jejichž počet odpovídá počtu filtrů. Na tyto koeficienty je následně aplikován logaritmus a ortogonální transformace. Toto má za následek nižší počet koeficientů, které dobře popisují tvar spektra. Nejpoužívanějšími trojúhelníkovými bankami jsou banky lineárně-frekvenční a Mel-frekvenční.

Lineárně frekvenční: Tato banka obsahuje trojúhelníkové filtry s překryvem, které jsou na frekvenční ose rovnoměrně rozmištěny a mají stejnou šířku, což má za následek rovnoměrné zpracování ve všech frekvenčních pásmech i těch méně podstatných pro běžné zpracování řeči, což může být výhodné při detekci podvržené řeči, jelikož se artefakty strojového generování řeči objevují i ve vyšších frekvencích [73], [74]. Obrázek 3.7a ukazuje jeden z možných tvarů banky.

Mel-frekvenční: Tato banka bývá často používána ve zpracování řeči, jelikož mapuje spektrum s ohledem na lidské vnímání. V tomto případě jsou filtry od sebe vzdáleny stejně s ohledem na vnímání lidského sluchu. S ohledem na logaritmické vnímání toto vede k bance filtrů s užšími filtry blíž u sebe na nižších frekvencích a se širšími a více vzdálenými filtry na vyšších frekvencích. Toto je vhodné, jelikož na nižších frekvencích se vyskytuje více informací o řeči (formanty atd.). Pro detekci podvržené řeči toto může být nevhodné díky nižšímu rozlišení na vyšších frekvencích. Obrázek 3.7b ukazuje jeden z možných tvarů banky.



Obrázek 3.7: Ukázka tvarů lineárně frekvenční a mel-frekvenční banky

Jako hlavní příznaky na vstup ResNetu byly zvoleny lineárně-frekvenční kepstrální koeficienty (LFCC - *Linear Frequency Cepstral Coefficients*) ve formě kepstrogramu.

Všechny signály byly převzorkovány na 16 kHz a následně nasegmentovány po 20 ms s 10 ms překryvem. Z těchto segmentů byla následně počítána kepstra o 20 prvcích a jejich Δ a $\Delta\Delta$ koeficienty. Aby bylo možné porovnat chování struktury s výsledky prezentovanými v ASV Spoof Challenge, kepstra byla počítána pomocí Matlabového scriptu poskytnutého autory výzvy ASV Spoof 2021 [76], který byl přepsán do Pythonu a upraven tak, aby špička každého filtru v bance měla zisk 1.

3.6 Zvolená metrika pro testování

Pro hodnocení kvality klasifikace systému byla zvolena metrika převzatá z ASV Spoof Challenge. Jejím cílem je nalezení optimálního prahu tak, aby FPR a FNR resp. FAR (False Acceptance Rate) a FRR (False Rejection Rate) byly stejné. Toto řešení je velmi praktické pro porovnávání jednotlivých struktur a jejich schopnosti přiřadit správná skóre, tj. provést dobrou separaci skóre pro přirozené a podvržené promluvy. Metrika je počítána z výstupních skóre OC Softmax na testovacím datasetu. Pro praktické využití za běhu, kdy data z testovacího datasetu nejsou k dispozici pro statistické zpracování není tato metoda vhodná. Pro testování schopnosti sítě rozdělit přirozené a podvržené promluvy je však optimální, jelikož v ní není zanesená chyba metody, která by sloužila ke klasifikaci a jejíž nastavení by se odvíjelo z výsledků na trénovacím či vývojovém datasetu, tj. tato metrika hodnotí pouze schopnost sítě dobré od sebe oddělit přirozené a podvržené promluvy.

V prvním kroku jsou spočtená skóre ($\cos(\theta)$) seřazena dle velikosti a slouží jako potencionální prahy τ_i . Následně je spočten kumulativní součet přes odpovídající seřazené invertované ($0 \rightarrow 1, 1 \rightarrow 0$) labyly. Pro index i a při vzestupném seřazení to vyjadřuje, kolik přirozených promluv se vyskytuje pod tímto indexem. FRR je následně spočten jako:

$$FRR[i] = \frac{\sum_{j=0}^i l_{sorted}[j]}{N_{target}}, \quad (3.11)$$

kde l_{sorted} je pole seřazených labelů a N_{target} je počet přirozených (cílových) promluv.

Zároveň se počítá i FAR:

$$FAR[i] = \frac{N_{nontarget} - (i - \sum_{j=0}^i l_{sorted}[j])}{N_{nontarget}}, \quad (3.12)$$

kde $N_{nontarget}$ odpovídá počtu strojově generovaných promluv. V posledním kroku je hledán takový index i , pro který se $FRR[i]$ a $FAR[i]$ liší nejméně. EER je následně spočten jako:

$$EER = \frac{FRR[i] + FAR[i]}{2} \quad (3.13)$$

a klasifikační práh je tudíž τ_i .

Tento klasifikační práh je následně využit i pro klasifikaci v jednotlivých kategoriích promluv, tj. přirozených promluv a jednotlivých typů útoků, pro něž je spočten ER (Error Rate), tj. FPR nebo FNR.

Takto počítaný práh a metrika byly využity až pro experimenty se složitější strukturou. V prvních experimentech s jednoduší strukturou byl FPR, FNR a EER počítán standardním způsobem.

4 Implementace

Tato kapitola se zabývá implementací celého projektu, tj. prvotními experimenty s jednodušším ResNetem, vytvořením českého datasetu i experimenty s ním a složitější strukturou. Pro celou práci byl využit programovací jazyk Python, jelikož obsahuje značné množství knihoven pro práci se soubory, zpracování signálů i trénování neuronových sítí. Níže následuje výčet hlavních použitých knihoven:

1. **Soundfile**: Knihovna sloužící pro práci se zvukovými soubory. V práci využita pro načítání souborů s promluvami.
2. **Pandas**: Nástroj pro datovou analýzu a další práce s daty. Vhodný především pro data, která lze reprezentovat 2D tabulkou (SQL databáze, *.csv soubory a jiné). V práci využit pro uskladnění informací o jednotlivých souborech (promluvách), tj. jméno souboru, typ promluvy (přirozená nebo uměle generovaná), přepis promluvy atd.. Pandas je oproti reprezentaci dat pomocí dvourozměrného seznamu přehlednější a snazší na správu (přidávání, odebírání, vyhledávání) a je taktéž rychlejší.
3. **Numpy**: Nástroj pro práci s číselnými vektory, maticemi i více dimenzionálními objekty. Oproti reprezentaci v listu je rychlejší.
4. **SciPy**: Knihovna pro numerické výpočty, která výrazně usnadňuje práci s daty v aplikacích, kde je potřeba matematické modelování, statistika nebo signálová analýza.
5. **Spafe**: Knihovna pro práci se signály. Umožňuje extrakci různých příznaků: LFCC, MFCC, GTCC a jiné, dále pak preprocessing signálů či filtraci a banky filtrů.¹
6. **Pickle**: Nástroj pro ukládání dat jako objektů. Uloží např. celé numpy array jako jeden objekt do jednoho souboru, bez nutnosti ukládat postupně jednotlivé prvky z objektu. V práci využit pro ukládání souborů s vyextrahovanými příznaky.
7. **Threading**: Knihovna pro vícevláknové výpočty.
8. **torch**: Základní balíček umožňující práci s vícedimenzionálními tensory a matematické operace s nimi.

¹Verze knihovny ke stažení se může lišit od verze v dokumentaci a její argumenty a jejich pořadí je třeba ověřit přímo z kódu..

9. **torch.utils.data Dataset**: Je jedním z primitivních typů, které umožňují organizovanou a jednoduchou práci s vlastními i veřejně dostupnými organizovanými datasety. Výstupem jsou obvykle samplify a labely. Pro práci s vlastním datasetem je nutné napsat třídu popisující a ovládající dataset, která bude z `torch.utils.data.Dataset` dědit.
10. **torch.utils.data Dataloader**: Třída umožňující jednoduché načítání a práci s daty. Jestliže jsou data uložena pomocí `torch.utils.data Dataset`, je možné je obalit pomocí `Dataloader`, který je iterovatelný ve smyčce a umožňuje i rozdelení dat na dávky (batch) i zamíchání pořadí jednotlivých samplů a labelů (shuffle).
11. **torch.nn**: Modul obsahující třídy a funkcionality pro vytvoření a trénování neuronových sítí. Skládá se ze čtyř hlavních částí: Základní bloky (`nn.Linear`, `nn.Conv2d`, `nn.Conv1d`, `nn.RNN`, `nn.LSTM`, `nn.BatchNorm2d...`), aktivační funkce (`nn.ReLU`, `nn.Linear`, `nn.Tanh ...`), ztrátové funkce (`nn.CrossEntropyLoss`, `nn.MSELoss ...`), struktury modelu (Sekvenční struktura pomocí kontejneru `nn.Sequential` či vlastní struktura pomocí dědičnosti z `nn.Module`).

Kromě výše uvedených knihoven, se v práci vyskytují i balíčky založené na knihovně `transformers` od Hugging Face, využívané v jednotlivých nástrojích TTS/VC.

4.1 Implementace prvních experimentů

Tato sekce obsahuje ukázky základní implementace s využitím jednoduchého ResNetu a ASV Spoof 2019 datasetu. Jedná se o zkrácené a jednodušší struktury kódu.

4.1.1 Dataset

Třída navržená pro práci s ASV Spoof 2019 datasetem pro LA i PA scénář. Vlastní třída `Dataset` dědí z `torch.utils.data.Dataset`. V konstruktoru dochází k nastavení cest ke vstupním příznakům (příznaky vyextrahované z `.flac` souborů a objektově uložené jako NumPy array ve stejné adresářové hierarchii). Dále se nastavují cesty pro ovládací soubory (CM protocols), délka příznaků, scénář (LA,PA), využití prvního příznaku (u kepstra c_0) a způsob doplnění příznakového tensoru (2D matice), jestliže by byla kratší než je nastavená délka příznaků.

Dále jsou definovány tagy pro jednotlivé scénáře. Pro LA scénář jsou tagy brány jako systém, který vygeneroval promluvu či přirozená řeč. Tyto textové řetězce

("-", "A01"..."A19") je vhodné převést do číselné reprezentace. Dále jsou do číselné reprezentace převedeny i labely (přirozené - bonafide, podvržené - spoof).

```

if self.access_type == 'LA':
    self.tag = { "-": 0, "A01": 1, "A02": 2, "A03": 3,
    "A04": 4, "A05": 5, "A06": 6, "A07": 7, "A08": 8,
    "A09": 9, "A10": 10, "A11": 11, "A12": 12, "A13": 13,
    "A14": 14, "A15": 15, "A16": 16, "A17": 17, "A18": 18,
    "A19": 19}
else:
    self.tag = { "-": 0, "AA": 1, "AB": 2, "AC": 3, "BA": 4,
    "BB": 5, "BC": 6, "CA": 7, "CB": 8, "CC": 9}
self.label = { "spoof": 1, "bonafide": 0}

```

Následně dochází k načtení protokolového (ovládacího) souboru pomocí `pandas`. Ten je reprezentován jako `pandas.DataFrame` o čtyřech sloupcích.

Třída následně obsahuje dvě metody, které každá třída dědí z `torch.utils.data.Dataset` musí obsahovat a to `__len__(self)`, která vrací počet prvků v datasetu, a `__getitem__(self, idx)`, která vrací i-tý sample a label. Z pandas dataframu se vybere i-tý řádek, z něj se vezme sloupec se jménem, to se spojí s příponou, vytvoří se cesta k souboru a soubor je pomocí `pickle.load` načten. Label je opět vzat z dataframu.

4.1.2 Model

Model neuronové sítě je rovněž reprezentován třídou dědící z `nn.Module`. V konstruktoru dochází k inicializaci jednotlivých komponent sítě, v povinné metodě `forward` pak k jejich propojení.

```

class Basic_Block(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv_layer =
            nn.Conv2d(in_channels=32,out_channels=32,
            kernel_size=(3,3),padding=1,stride=1)
        self.conv_layer2 =
            nn.Conv2d(in_channels=32,out_channels=32,
            kernel_size=(3,3),padding=1,stride=1)
        self.bn_layer = nn.BatchNorm2d(32)
        self.relu = nn.LeakyReLU()

```

```
def forward(self, x):
    out = self.conv_layer(x)
    out = self.bn_layer(out)
    out = self.relu(out)
    out = self.conv_layer2(out)
    out = out + self.conv_layer2(x)
    out = self.bn_layer(out)
    out = self.relu(out)
    return out
```

Objekty vytvořené z takto definovaných tříd pak lze použít jako komponenty do dalších struktur.

4.1.3 Trénování

Trénovací skript je navržen tak, aby jej bylo možné spouštět nejen z IDE, ale i z příkazové řádky. Všechna potřebná nastavení lze udělat pomocí přepínačů. K tomuto a jednoduché práci s argumenty byl využit modul `argparse`, konkrétně `argparse.ArgumentParser()`, viz ukázka níže. Výstupem je pak objekt k jehož proměnným (např. `lr,bs,epochs`) se lze dostat pomocí tečkové notace. Tento objekt je používán jako nosná struktura i pro další funkce v souboru, které z něj čtou, nebo do něj přidávají.

```
parser = argparse.ArgumentParser()
parser.add_argument("--lr", type=float, default=5e-5)
parser.add_argument("--bs", type=int, default=32)
parser.add_argument("--epochs", type=int, default=25)
args = parser.parse_args()
```

Dále probíhá inicializace. V této fázi se ověřuje zda je možné na daném zařízení využít trénování na GPU (CUDA). Jestliže ano, je proměnná `args.device` nastavena na "CUDA". Při práci s neuronovou sítí i s jednotlivými tenzory či datasety je nutné u nich nastavit parametr `device`, aby správně fungovaly na CPU i GPU.

```
args.cuda = torch.cuda.is_available()
print("Cuda device available: ", args.cuda)
args.device = torch.device("cuda" if args.cuda else "cpu")
```

Poté jsou nastaveny cesty k protokolovým souborům, základní cesty k databázi a je vytvořena složka pro uložení modelu a následných pomocných, informativních a popisných souborů.

Následuje inicializace Datasetů (trénovacího a vývojového), Dataloaderů a modelu neuronové sítě.

```

training_set = ASV_Spoof_2019_dataset( ... )
validation_set = ASV_Spoof_2019_dataset( ... )

train_data_loader = DataLoader(training_set , args.bs ,
shuffle=True)
validation_data_loader = DataLoader(validation_set , args.bs ,
shuffle=True)

model = Residual_Net( ... ) . to(args.device)

```

Dále je nastaven optimalizátor a ztrátová funkce. Poté již začíná trénovací smyčka. Hlavní smyčka prochází přes epochy, první vnořený cyklus pak přes jednotlivé dávky (batch) trénovacího Dataloaderu. Zde je ovšem nutné upravit dimenze příznakové matice z Dataloaderu před vstupem do NN. Uvažujme uložený a následně načtený, zkrácený či prodloužený kepsrogram o velikosti (19, 200) a velikost dávky 32. Výstupem z dataloaderu je pak tensor o velikosti (32, 19, 200). Konvoluční vrstvy však požadují vstup ve formátu: (Velikost dávky (batch size), počet kanálů, výška, šířka). Toto je způsobeno tím, že konvoluční vrstvy pracují s více konvolučními jádry a výsledek konvoluce s každým jádrem má svůj kanál. Je třeba tedy přidat jednu prázdnou dimenzi mezi 32 a 19, aby formát byl (32, 1, 19, 200) (metoda `unsqueeze`) tj. jeden kanál na vstupu.

```
lfcc = lfcc . unsqueeze(1) . float() . to(args.device)
```

Jednotlivé predikce modelu pro každý z batchů jsou ukládány do proměnných a jsou počítány i statistiky (ACC, loss a další viz Kapitola 5.1). Vše je následně uloženo. Toto se děje i pro validační dataset, ovšem model je v této fázi nastaven do evaluačního režimu a nedochází ke změnám v gradientu.

4.2 Implementace pro hlavní Experimenty

Tato sekce se zabývá popisem ovládacích struktur celého projektu, použitých nástrojů a postupů, a také implementací jednotlivých součástí — od tvorby datasetu po samotné trénování klasifikační struktury.

4.2.1 Ovládací struktura

Tvorba datasetu i trénování jsou ovládány za pomocí *.tsv souborů (souborů u nichž jsou sloupce odděleny tabulátory). Základní datasety přirozených promluv obsahují tento typ souborů také, jednotlivé názvy sloupců i jejich počet se u nich liší, vždy však obsahují následující položky: jméno souboru, kód řečníka a přepis promluvy.

Přípravná fáze

Při přípravě databáze byly tyto soubory sjednoceny: sloupce přejmenovány, u přepisů promluv ponechány pouze čisté přepisy promluv malým písmem (odstraněny především různé značky značící nádechy, ticho atd., a interpunkce). Jelikož každý z datasetů obsahuje různé podsložky, ve kterých jsou výsledné soubory, byl doplněn i sloupec s názvy těchto podsložek ve kmenové složce.

Nástroje TTS/VC

V dalším kroku bylo nutné připravit *.tsv soubory pro jednotlivé TTS/VC nástroje. Toto bylo způsobeno tím, že některé z nástrojů pracují pouze s jednou cílovou promluvou (target) jiné s více. V případě více promluv byly tyto promluvy umístěny do jednoho sloupce, oddělené čárkami. Dále byl vygenerován sloupec výstupních jmen souborů. Toto bylo uděláno pro každý z nástrojů a každý z originálních datasetů. Tyto soubory posloužily i pro zápis vypočtených perplexit. Výsledné soubory mají následující strukturu.

1. **DIR_SRC**: Podsložka v originálním datasetu, kde se zdrojová promluva nachází
2. **SRC**: Název zdrojového souboru bez přípony
3. **SPK_SRC**: Kód zdrojového mluvčího
4. **DIR_TARGET**: Podsložka v originálním datasetu, kde se cílová promluva nachází
5. **TARGET**: Název cílového souboru bez přípony
6. **SPK_TARGET**: Kód cílového mluvčího
7. **RES**: výsledný název souboru bez přípony
8. **LBO**: Přepis zdrojové promluvy

9. **LBO_RES**: Přepis vygenerované promluvy pomocí Whisper ASR
10. **PERP_SRC**: Relativní perplexita spočtená z **LBO**
11. **PERP_RES**: Relativní perplexita spočtená z **LBO_RES**
12. **PERP_DIF**: Poměr relativních perplexit

Pro FVC bylo nutné ještě vytvořit ***.txt** souboru v jiném formátu, díky jeho specifickým požadavkům

SP Spoof CS Dataset

Pomocí výše zmíněného intervalu poměrů perplexit byly vybrány promluvy do finálního datasetu podvržených promluv. Aby byl dataset přehledný, byly vybrané promluvy z jednotlivých kombinací datasetů a nástrojů ponechány ve svých příslušných složkách. Z dílčích ovládacích souborů byly vyextrahovány řádky obsahující vybrané promluvy a ty byly sloučeny a upraveny do souboru s následující strukturou sloupců.

1. **DIR**: Podskočka v SP SPoof CS datasetu, kde se promluva nachází
2. **SRC**: Název promluvy bez přípony
3. **TOOL**: Nástroj pomocí kterého byla promluva vytvořena

Výsledný struktura *.tsv****

Výsledný ovládací soubor, který může obsahovat různé kombinace přirozených či podvržených promluv, má následující strukturu.

1. **DIR**: Podskočka v SP SPoof CS, či originálním datasetu, kde se promluva nachází
2. **SRC**: Název souboru s promluvou
3. **DS_SRC**: Název datasetu, ze kterého promluva pochází
4. **TYPE**: Přípona souboru
5. **TOOL**: Nástroj jímž byla promluva vytvořena, pro přirozené promluvy **"-**
6. **BFS**: Označení, zda se jedná o přirozenou, či podvrženou promluvu.

Takto formovaná struktura, může sloužit i pro jiné než výše popsané datasety.

4.2.2 Úprava nástrojů TTS/VC

Pro účely práce byly vybrány především nástroje s implementací v Pythonu, aby je bylo možné upravit pro výše popsané struktury datasetů a ovládacích souborů. Toto se neukázalo jako problém, jelikož většina existujících nástrojů je založena na Pythonu; výjimkou je pouze FVC. Nástroje však bylo nutné upravit, jelikož každý z nich očekává jiný formát vstupu. Z tohoto důvodu došlo k editaci kódu.

4.2.3 Implementace výpočtu kepster

Pro úsporu času při trénování klasifikační struktury, byla část kódu pro výpočet příznaků oddělena. Pro načítání je využit výsledný `*.tsv` soubor, který je načten pomocí `pandas.DataFrame`. Tabulka je následně rozdělena podle chtěného počtu vláken ke zpracování. Pro zrychlení zpracování, je využito vícevláknové programování pomocí `threading.Thread`. Každé z vláken nezávisle zpracovává svou část tabulky - od vytvoření finální cesty k souboru, přes jeho načtení, výpočet příznaků i jejich uložení pomocí `pickle`, jako celé `numpy.array`.

Parametry výpočtu kepster jsou řízeny konfiguračním `*.csv` souborem. Pro vytvoření banky filtrů i výpočty spekter aj. byla využita knihovna `scipy`.

Jak je zmíněno výše ovládací `*.tsv` soubor obsahuje sloupec s podsložkami umístění souborů v jednotlivých datasetech, cesty k domovským adresářům musely být zadány v kódu. Takové řešení zajišťuje, že při změně umístění hlavního adresáře datasetu, nebude nutné znova generovat `*.tsv` soubory.

4.2.4 Implementace klasifikační struktury

Tato podsekce popisuje základní implementaci celé klasifikační struktury, načítání dat a fázi trénování, ověřování i testování.

Načítání dat

Aby bylo načítání a reprezentace dat co nejfektivnější, byla použita třída dědící z `torch.utils.data.Dataset`. Taková struktura umožňuje jednoduchou a efektivní reprezentaci a načítání dat pomocí třídy, která dědí z právě ze třídy `Dataset`. V konstruktoru je načten ovládací `*.tsv` soubor, labeły a typy jsou útoků převedeny na číselnou reprezentaci a nastaveny další parametry. V override funkci `__getitem__` je pak sestavena cesta k souboru, soubor načten a jeho délka je upravena. V případě, že

délka přesuhaje zadaný požadovaný rozměr, dochází k ořezu kepstrogramů. V případě, že je délka kratší, než požadovaná, dochází k doplnění chybějících zadních částí ze začátku kepstrogramu. Ten ve formě `np.array` je převeden na `torch.tensor` a vrácen.

Při trénování i inferenci je nutné postupně načítat i-té prvky z datasetu (obvykle po dávkách) při každé iteraci. K tomuto účelu byl využit `torch.utils.data DataLoader`, který umožňuje pokročilé iterace přes `Dataset` po dávkách či promíchání pořadí prvků v datasetu. Jednou z jeho klíčových vlastností je možnost asynchronního načítání dat pomocí více CPU workerů (argument `num_workers`), jelikož v případě, že je pro práci s NN využita grafická karta, načítání souborů, které je zajišťované pomocí CPU, velmi zpomaluje proces. Při využití defaultního nastavení jsou tato vlákna po každé epoše zahozena a inicializována znova, což vyžaduje extra čas, z tohoto důvodu byly nastaveny pomocí argumentu `persistent_workers=True` tak, aby přetrvaly po celou dobu trénování. Dále byla využita možnost zapojení tzv. pinned paměti (někdy také označované jako page-locked memory), která umožňuje rychlejší a asynchronní přenos dat do GPU.

Trénování

V průběhu trénování je trénován jak ResNet, tak i centrální vektor u OC Softmax. Oboje trénování se řídí standardní Pytorch strukturou a postupy. Jako ztrátová funkce trénování ResNetu je použita OC Softmax či její modifikace. Další parametry trénování v tabulce 4.1.

Parametr	Hodnota
Délka kepstrogramu	750
Počet výstupních příznaků	512
LR	3×10^{-5}
Pokles LR	polovina za 10 epoch
Optimalizace	Adam
Adam β_1	0.9
Adam β_2	0.999
Adam ϵ	1×10^{-8}
Seed	598
CUDA	Ano
Optimalizace OC Softmax	SGD
Inicializace	Kaiming uniform

Tabulka 4.1: Základní trénovací parametry ResNet

Po každé trénovací epoše je proveden test na vývojovém/validační datasetu. Jestliže je výsledek horší než v předchozí epoše, trénování se ukončuje a jsou uloženy modely ResNet i OC Softmax s parametry z předchozí epochy tzv. Early Stopping.

Velikého zrychlení (2x-3x) bylo dosaženo pomocí automatického nastavení přesnosti (Automatic mixed precision) s využitím `torch.amp autocast`, `GradScaler`. Toto nastavení automaticky přepíná mezi `float32`, který je výchozím formátem, a `float16` v částech kódu kde je to výhodné. Tato metoda využívá struktury u novějších grafických karet, tzv. tensorových jader (*tensor core*) uvnitř CUDA jader, která jsou specializována na maticové výpočty především pro `float16`. Operace jako maticové násobení, operace prvek po prvku či počítání s dávkami, tj. operace, u kterých je to bezpečné bez ztráty přesnosti jsou prováděny s nižší přesností a operace u kterých je velmi důležitá přesnost jako Softmax, Cross-Entropy, log, exp atd. jsou ponechány ve `float32`. Numericky stabilní operace, kde nehrozí podtečení tedy využívají `float16`, ty kritické pak `float32`. `GradScaler` zajistuje ponechání gradientu ve správném rozsahu pro různé operace v síti.

Trénování probíhalo na PC autora práce s následujícími parametry: procesor Intel Core i5-9400F, GPU NVIDIA GeForce RTX 2060 (6 GB) a RAM 16 GB. Výpočty byly také prováděny na Metacentru [77].

4.3 Implementace pro Metacentrum

Metacentrum [77] je gridová distribuovaná výpočetní infrastruktura umožňující provádět velmi náročné výpočty, které by kvůli nedostatečné kapacitě, výkonu či strojovému času nebylo možné spočítat na běžných počítačích nebo počítačových strukturách vědeckých týmů. Umožňuje zadávat interaktivní i dávkové úlohy. Vše je ovládáno z příkazové řádky a pomocí bash scriptů přes SSH spojení. Metacentrum nabízí také rozmněrné úložiště pro skripty i datasety. Nahrání a manipulace se soubory jsou možné pomocí FTP. Výpočty jsou podporované jak na CPU, tak i na GPU.

Metacentrum podporuje výpočty v Pythonu a veliké množství knihoven je obsaženo v jednotlivých modulech [78]. Pro standardní sériové CPU výpočty postačí do .sh skriptu, který ovládá a spouští Python skript, přidat příslušný modul pomocí příkazu `module add xxx`. Jestliže požadovaná knihovna není v žádném z modulů, je možné jí ve scriptu nainstalovat pomocí příkazu `pip install`.

Starší verze frameworku Pytorch jsou také dostupné jako moduly. Jejich použití však není doporučováno. Doporučené je využití tzv. NVIDIA kontejnerů, což je přenosné software prostředí optimalizované pro běh na GPU. Kontejnery v sobě obsahují vše potřebné pro běh Pytorch, včetně dalších knihoven. Tyto kontejnery se pro dávkové soubory standardně spouští pomocí:

```
singularity run -nv /cvmfs/singularity.metacentrum.cz/...
.../NGC/PyTorch:24.10-py3.SIF my_python_script.py,
```

kde cvmfs/singularity.metacentrum.cz//NGC/PyTorch:24.10-py3.SIF je název a umístění kontejneru.

Jestliže je v kódu třeba využít knihovnu, která není obsažena v kontejneru, je nutné kód upravit. Jelikož se jedná o svébytné prostředí, využití `pip install` mimo kontejner nebude fungovat. Existují dvě možnosti řešení. První a složitější je zkopírování kontejneru, jeho otevření (vytvoření sandboxu), doplnění dalších knihoven a jeho opětovné zavření. Toto je na Metacentru poměrně komplikovaný proces s mnoha problémy. Řešení v podobě vytvoření, či replikace kontejneru staženého přímo od NVIDIA, také nelze začínajícím uživatelům doporučit. Druhá možnost je využití `singularity exec` místo `singularity run`. Exec na rozdíl on `run`, které spustí pouze defaultní akci kontejneru (v tomto případě python script), umí spustit příkazy uvnitř kontejneru, tj. je možné využít pip uvnitř kontejneru a pomocí něj nainstalovat příslušné knihovny. Zde je nutné brát v potaz, že pip se nevyskytuje ve všech kontejnerech a na všech strojích na kterých je výpočet prováděn. Proto je nutné pip nainstalovat (např. pomocí souboru get-pip.py). Dále je nutné zohlednit, že snaha instalovat pip tam, kde již instalován je, povede k pádu programu. Řešení

může vypadat následovně:

```
singularity exec -nv /cvmfs/singularity.metacentrum.cz/NGC/PyTorch:  
24.10-py3.SIF bash -c "if ! command -v pip &> /dev/null; then  
python get-pip.py; fi && pip install torchsummary && pip install argparse  
&& python my_script.py"
```

5 Experimentální část

5.1 Prvotní experimenty

První experimenty měly za cíl vybudovat infrastrukturu celého projektu, tj. kód pro extrakci a uložení příznaků, třídy pro práci s datasety, kód pro trénování, evaluaci a testování, kód pro výpočet statistik atd. Všechny kódy byly koncipovány od začátku tak, aby nebylo jejich spuštění možné pouze z IDE s nutností přepisovat proměnné, ale aby je bylo možné spustit z příkazové řádky či pomocí bash skriptu, a to nejen na PC, ale i na gridových výpočetních službách jako je Metacentrum, kam byl kód postupně přesouván. Dalším cílem bylo seznámení se postupy implementace, ovládání a ladění pokročilejších struktur NN, než jsou DNN pouze s dopřednými vrstvami.

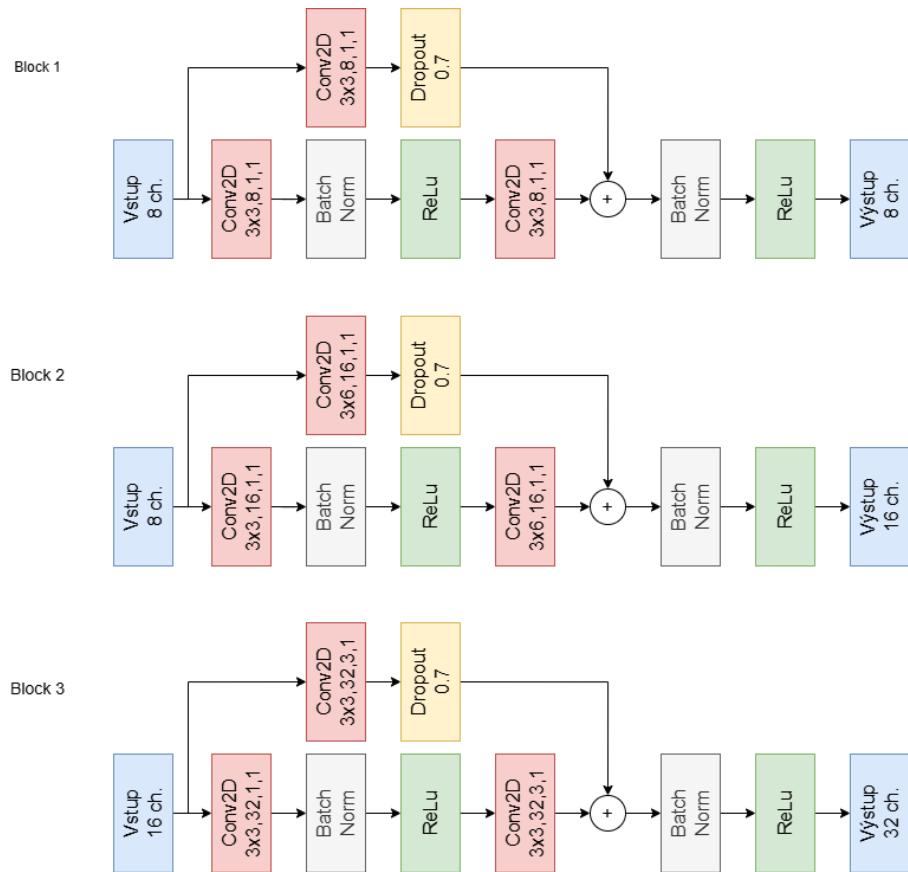
Pro první experimenty byl využit ASV Spoof dataset. Za příznaky byly zvoleny LFCC, konkrétně 19 kepstrálních koeficientů (c_1-c_{19}), bez Δ a $\Delta\Delta$ koeficientů z frekvenčního rozsahu 0 - 4kHz. Jako struktura sítě byl zvolen dvou-blokový ResNet s dropouth nastavenými na 30 % a s 800 neurony v plně propojené vrstvě. Struktura sítě je popsána k sekci 3.2.1. LR byl nastaven na 1e-4 s ad-hoc schodovým klesáním. Prvky sítě byly inicializovány náhodně.

Jako metrika úspěšnosti učení byla zprvu vybrána ACC. Z pohledu této metriky síť již po několika epochách vykazovala velmi dobré výsledky (ACC kolem 90 %) pro všechny tři datasety. Následnou analýzou výstupů byla zjištěna příčina, která spočívala v tendenci sítě označit všechny vstupy za podvržené. Jak již bylo zmíněno podvržených promluv je v datasetu přibližně 9x více, tudíž 90% přesnost predikce (ACC) byla zapříčiněna tímto. Aby bylo možné toto sledovat a následně korigovat, byly zavedeny další metriky, a to: FPR (False Positive Rate) a FNR (False Negative Rate). Následně se ještě počítá EER (Equal Error Rate) a F1 skóre. Jestliže je FPR vysoké, znamená to, že systém klasifikuje příliš mnoho přirozených promluv jako podvržené. Optimální poměr mezi FPR a FNR záleží na konkrétním využití, a může být předmětem diskuze. V této části práce bylo cílem obě dvě hodnoty zachovat vyrovnané.

Jedním z kroků mírnících nevyváženos datasetu je přidání vah ke ztrátové funkci Cross Entropy. Tyto váhy určují, jak moc má být brána v potaz každá z klasifikačních kategorií. Optimální hodnota se ukázala být 3-4:1 ve prospěch přirozených promluv. Takto optimalizovaná síť dosáhla EER = 30%.

Další pokusy měly za cíl postupné zlepšování výsledků sítě. K LFCC (bez c_0) byly přidány Δ a $\Delta\Delta$ koeficienty. Struktura sítě byla zvětšena na 4 bloky a bylo experimentováno i se 64 kanály u některých konvolučních vrstev. Toto vedlo k delšímu výpočetnímu času a ke zlepšení výsledků na trénovací množině. Na vývojové a testovací množině se opět výrazně zvýšila falešná pozitivita (FP). Síť byla tudíž velmi přetrénovaná na daná data a nedokázala dobře generalizovat (overfitting).

Jako řešení se ukázalo zmenšení počtu kanálů, jejich postupný růst a využití asymetrických konvolučních jader (delší v časové dimenzi). Dalším podstatným zjištěním bylo, že síť velmi využívá při své práci přemostění, jehož účelem je pouze zamezit ztrátě velikosti gradientu při zpětné propagaci, nikoliv sloužit pro dopředný průchod. Z tohoto důvodu byl k přemostění přidán dropout ve výši 70%. Toto mělo zásadní vliv. Dále došlo k využití větších posunů (stride) některých konvolučních jader. Jako optimalizační algoritmus byl využit Adam s L2 regularizací (weight decay) = 1e-9. Struktura bloků této sítě je znázorněna na obrázku 5.1. Další nastavení jsou popsána v tabulce 5.1 a výsledky na evaluačním datasetu v tabulce 5.2.



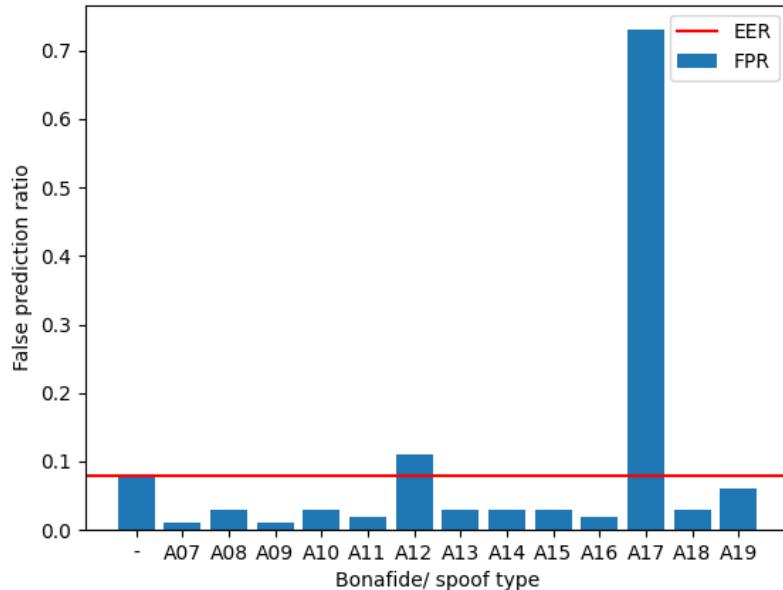
Obrázek 5.1: Znázornění tří bloků Resnetu

Výsledky ukazují 8% chybovost a vyváženou klasifikaci pro obě dvě skupiny. Z toho je patrná poměrně dobrá schopnost generalizace. Výsledky pro jednotlivé podskupiny jsou zobrazeny na obrázku 5.2. Z průměru se velmi odchyluje systém 17.

Parametr	Nastavení
Příznaky	LFCC
Dimenze příznaků	(3*18) x 200
Délka okna segmentace	
Preemfáze	Ano
$\Delta \Delta\Delta$	Ano
LR	5e-5
Pokles LR	Schodovitý
Optimalizační alg.	Adam
Velikost dávky	32
Epochy	25
Váhy pro Ztr. funkci (bonafide:spoof)	3:1
Neuronů v lin. vrstvě	800

Tabulka 5.1: Základní parametry sítě

TP	TN	FP	FN	EER	F1
91.04	92.00	8.000	8.060	8.030	92.01

Tabulka 5.2: Výsledky upravené sítě**Obrázek 5.2:** Výsledky pro přirozené promluvy a jednotlivé systémy TTS a VC

5.2 Hlavní experimenty

Oproti prvotním experimentům, které měly za cíl seznámení se se systémy detekce podvržené řeči, datasety i prací s frameworkm pytorch a další, tato část již popisuje experimenty s klasifikátorem pro český jazyk.

Cílem těchto experimentů je ověření schopností vybrané struktury klasifikovat v českém jazyce, ověřit její fungování pro známé a neznámé typy útoků i pro neznámé podmínky. Rozpoznávání podvržené řeči je poměrně širokým okruhem a pro každou konkrétní aplikaci je třeba počítat s různými typy podmínek, proto je co největší schopnost generalizace klíčová.

Základní SP Spoof CS dataset byl doplněn o 23 000 promluv z originálních datasetů, tj. z databází CTU Test, Voxpopuli a Common Voice. V průběhu experimentů byly vytvářeny nové datasety, přidáním dalších přirozených promluv či přesunutím jednotlivých nástrojů TTS/VC do testovací množiny. Jestliže není uvedeno jinak, experimenty využívají stejný dataset jako předchozí.

5.2.1 Experiment s ASV Spoof Datasetem

Při prvním experimentu byla využita výše popsaná struktura natrénovaná na ASV Spoof 2019 datasetu. Cílem bylo ověření schopnosti modelu generalizovat na jiné jazyky, typy nástrojů TTS/VC i podmínky než ve kterých byl trénovací dataset nahrán. Autorem vytvořený dataset **SP Spoof CS** byl využit pro testování. Základní výsledky popisuje tabulka 5.3.

Práh	-0.941
EER [%]	41.23

Tabulka 5.3: Celkové výsledky Experimentu 1

Z hodnoty prahu je patrné, že výstupní vektory z ResNetu u SP Spoof CS datasetu jsou velmi odlišné od těch z ASV Spoof Datasetu, jelikož skóre by se mělo pohybovat v kladných hodnotách, kde jsou nastaveny i hodnoty pro clustering. Tomuto odpovídá i EER, které je příliš vysoké. Z těchto výsledků je patrné, že systém natrénovaný na ASV Spoof datasetu není možné využít pro češtinu.

5.2.2 Experiment SP SPoof CS + OC Softmax

V tomto experimentu již byl zcela využit vlastní dataset, který byl náhodně rozdělen na trénovací, vývojovou a testovací množinu v poměru 60%:10%:30 %.

Jak je patrné z tabulek 5.4 a 5.5, struktura se dokáže velmi rychle přizpůsobit v průběhu trénování a klasifikovat velmi dobře pro známé typy útoků i známé podmínky prostředí.

Práh	0.406
EER [%]	0.892

Tabulka 5.4: Celkové výsledky Experimentu 2

Kategorie/nástroj	ER
Bonafide (FPR)	0.521
XTTS	0.234
FVC	0.938
kNN VC	0.642
Seed VC	0.915
Diff-Hier VC	0.023

Tabulka 5.5: Error rate (False Positive/False Negative) Experimentu 2 pro jednotlivé kategorie

5.2.3 Experiment 3 - Neviděný útok

Cílem tohoto experimentu bylo ověření schopnosti struktury generalizovat i na neviděné typy útoků, tj. na neviděné nástroje TTS/VC. Z tohoto důvodu byly promluvy vygenerované nástrojem Seed VC přesunuty do testovací množiny a zbylý dataset byl rozdělen v poměru 60% : 10% : 30%.

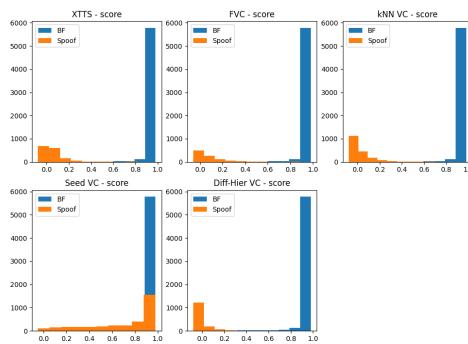
V tomto případě byla již struktura trénovaná od začátku po 5 epoch s počáteční LR 3×10^{-5} . Výsledky uvedené v tabulkách 5.6 a 5.7 ukazují, že struktura je již po velmi krátké době přetrénovaná na viděné typy útoků (overfitting) a nedokáže dobře generalizovat, jak je patrné z obrázku 5.3. Toto je pravděpodobně způsobeno tvarem ztrátových funkcí pro přirozené a podvržené promluvy, jak je patrné z obrázku 3.6. Tato ztrátová funkce je velmi diskriminativní a shlukuje promluvy z dané kategorie velmi blízko zadaných kosinových podobností. Toto může být nevhodné pro generalizaci, pro kterou by bylo vhodnější širší pásmo výstupních hodnot $\cos(\theta)$.

Práh	0.934
EER [%]	11.63

Tabulka 5.6: Celkové výsledky Experimentu 3

Kategorie/nástroj	ER
Bonafide (FPR)	5.803
XTTS	0.000
FVC	0.000
kNN VC	0.000
Seed VC	16.120
Diff-Hier VC	0.000

Tabulka 5.7: Error rate (False Positive/False Negative) Experimentu 3 pro jednotlivé kategorie



Obrázek 5.3: Skóre testovacího datasetu pro 1 neznámý útok - Seed VC

5.2.4 Experiment 4 - OC Softmax + Margin

Z důvodů uvedených v předchozím experimentu byla část OC Softmax funkce počítající ztrátu při trénování upravena. Jak již bylo psáno, cílem originální funkce je rozdělení výstupních skóre co nejbližše zadáným hodnotám $r_{fake} = 0.5$ pro podvržené a $r_{real} = 0.9$ pro přirozené pomocí kosinové podobnosti a softplus funkce. Tato metoda však tlačí skóre na jednu či druhou stranu příliš silně, a výsledkem je distribuce skóre, která je velmi úzká a má "exponenciální" charakter s velmi silným růstem v krajiných bodech. Jak je patrné z předchozího experimentu, takováto distribuce je nevhodná pro neznámé typy útoků a pravděpodobně bude nevhodná i pro neznámé podmínky u přirozených promluv.

Vhodným řešením tohoto problému se ukázala úprava distribucí skóre pro přirozené a podvržené promluvy. Pro tuto úpravu byl využit vektor kosinových podobností spočtený v kroku `scores = x @ w.transpose(0,1)`. Pro podobnosti z intervalu 0.2 - 1.2 (resp. 1) byla použita penalizace ve formě softplus funkce s příkrostí ovlivněnou parametrem `alpha`. Tato penalizace je následně přičtena ke ztrátě spočítané pomocí standardní OC Softmax. Kód upravené OC Softmax vypadá následovně:

```

class OCSoftmax_custom(nn.Module):
    def __init__(self, feat_dim=2, r_real=0.9, r_fake=0.5, alpha=15.0):
        super(OCSoftmax_custom, self).__init__()
        self.feat_dim = feat_dim
        self.r_real = r_real
        self.r_fake = r_fake
        self.alpha = alpha
        self.center = nn.Parameter(torch.randn(1, self.feat_dim))
        nn.init.kaiming_uniform_(self.center, 0.25)
        self.softplus = nn.Softplus()

    def forward(self, x, labels):

        w = F.normalize(self.center, p=2, dim=1)
        x = F.normalize(x, p=2, dim=1)

        scores = x @ w.transpose(0,1)
        output_scores = scores.clone()

        scores[labels == 0] = self.r_real - scores[labels == 0]
        scores[labels == 1] = scores[labels == 1] - self.r_fake

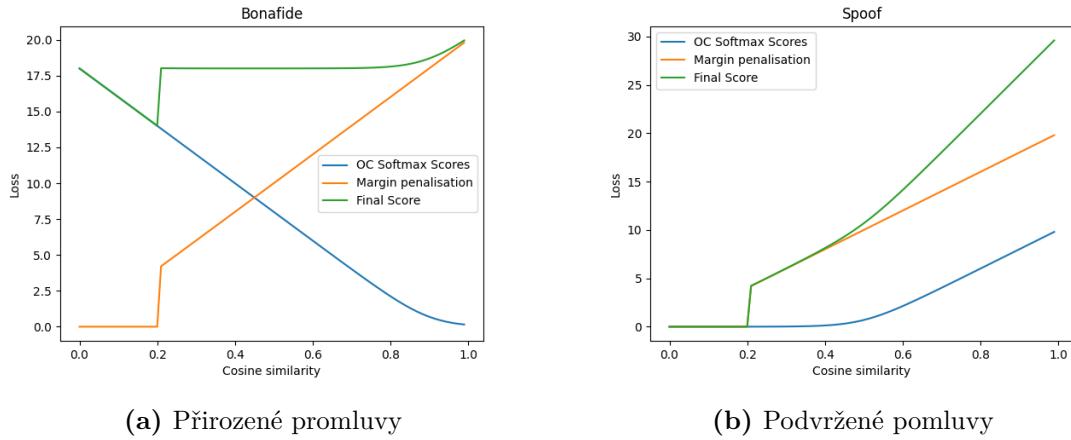
        loss = self.softplus(self.alpha * scores).mean()

        margin = 0.3 # Defines the uncertainty zone
        loss_conf = torch.where(
            (output_scores > self.r_fake - margin) & (output_scores <
            ↳ self.r_real + margin),
            self.softplus(self.alpha * output_scores)
            torch.tensor(0.0, device=x.device)
        ).mean()
        loss += loss_conf

    return loss, output_scores.squeeze(1)

```

Tato úprava funguje jako protisíla proti OC Softmax především v oblasti přirozených promluv. OC Softmax tlačí přirozené promluvy ke $\cos(\theta) = [0.9, 1]$, oproti tomu margin je nechává rozprostřít a mírně je posouvá k hodnotám kolem 0.2. Průběhy této ztrátové funkce s porovnáním s tradiční OC Softmax jsou znázorněny na obrázku 5.4.



Obrázek 5.4: Porovnání průběhů OC Softmax a OC Softmax + margin v závislosti na $\cos(\theta)$ pro $r_{real} = 0.9$, $r_{fake} = 0.5$ a $\alpha = 20$

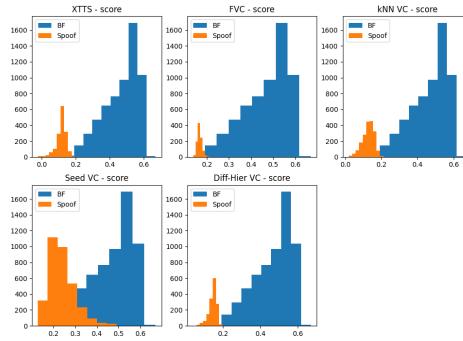
Tato kontraintuitivní úprava (minimální hodnota ztrátové funkce pro přirozené promluvy leží v oblasti podvržených promluv) zabraňuje tomu, aby si byla struktura ve svých rozhodnutích příliš jistá. I dobře klasifikované vzorky jsou mírně penalizovány, což vede k eliminaci exponenciální/špičkové charakteristiky skóre a k jeho širší distribuci. Toto snížení jistoty v rozhodování je vhodné pro neviděné typy útoků, jak je patrné z histogramů na obrázku 5.5 i výsledkových tabulek 5.8 a 5.9, kde došlo k poklesu FNR u Seed VC a také k poklesu FPR u přirozených promluv. Zároveň nedošlo k nárůstu FNR u dalších typů útoků.

	Exp. 3	Exp. 4 - Margin
Práh	0.934	0.289
EER [%]	11.63	6.452

Tabulka 5.8: Celkové porovnání EER pro výsledky Experimentu 3 a 4

Kategorie/nástroj	Exp. 3	Exp. 4 - Margin
Přirozené (FPR)	5.808	3.218
TTTS	0.000	0.000
FVC	0.000	0.000
kNN VC	0.000	0.000
Seed VC	16.120	8.939
Diff-Hier VC	0.000	0.000

Tabulka 5.9: Porovnání ER v jednotlivých kategoriích pro výsledky Experimentu 3 a 4



Obrázek 5.5: Skóre testovacího datasetu pro 1 neznámý útok - Seed VC

5.2.5 Experiment 5 - dva neviděné datasety

Cílem tohoto experimentu bylo ověřit, zda předchozí výsledky s neviděným datasetem nebyly pouze náhodné. V tomto případě byly promluvy vytvořené pomocí nástrojů XTTS a Free VC přesunuty pouze do testovací množiny a zbylá data byla rozdělena v poměru 60% : 10% : 30%. Z výsledků zobrazených v tabulkách 5.10 a 5.11 je patrné, že struktura je schopná generalizovat i na dva neznámé typy útoků. Obrázek 5.6 ukazuje dobré rozložení skóre pro obě kategorie promluv s malým překryvem.

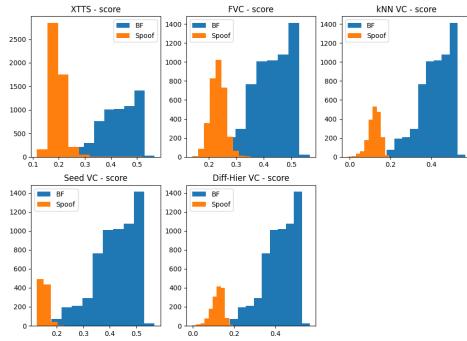
Experiment dále ukázal, že v průběhu trénování může dojít k postupnému převážení vlivu původní OC Softmax, rozdělení u přirozených promluv začne být exponenciální a skóre pro neviděné datasety se rozšíří. Z tohoto důvodu je nutné trénování včas ukončit.

	Exp. 4	Exp. 5
Práh	0.289	0.260
EER [%]	6.452	4.707

Tabulka 5.10: Celkové porovnání EER pro výsledky Experimentu 4 a 5

Kategorie/nástroj	Exp. 4	Exp. 5
Přirozené (FPR)	3.218	2.352
XTTS	0.000	1.216
FVC	0.000	7.15
kNN VC	0.000	0.000
Seed VC	8.939	0.205
Diff-Hier VC	0.000	0.000

Tabulka 5.11: Porovnání ER v jednotlivých kategoriích pro výsledky Experimentu 4 a 5



Obrázek 5.6: Skóre testovacího datasetu pro dva neznámé útoky

5.2.6 Experiment 6 - Neznámé podmínky

Cílem tohoto experimentu bylo ověření schopnosti struktury správně klasifikovat přirozené promluvy i za jiných podmínek než jaké jsou zastoupeny v originálním datasetu. Tyto nové podmínky zahrnují přítomnost aditivních či konvolučních zkreslení.

Konvoluční zkreslení bývají nejčastěji způsobena nahrávacími podmínkami, tj. frekvenční charakteristikou mikrofonu či impulsovou odezvou místnosti (RIR - *Room Impulse Response*). Pro vytvoření promluv s takovýmto zkreslením byl využit CTU Test dataset, jehož promluvy byly zkonvolvovány s náhodně vybranými RIR z 15 předvybraných, které zachycují standardní nahrávací podmínky, tj. běžné místnosti, ve kterých se obvykle nahrává zvuk (kancelář, obývací pokoj, učebna atd.). Impulsové odezvy pochází z datasetu MIT Acoustical Reverberation Scene Statistics Survey [79].

Aditivní šumy zahrnují jakýkoliv s promluvou nekorelovaný signál, jenž je superponován na původní řeč. U běžných nahrávek se jedná o ruchy v pozadí, jako například hučení ventilátoru, tikání hodin, ruch dopravy a další. V případě obsahu publikovaného na sociálních sítích se nejčastěji jedná o hudbu, která byla softwarově přidána do pozadí. Oproti některým jiným typům aditivních šumů, je hudba velmi nestacionárním signálem, hudební nástroje mají svá vlastní buzení a rezonance, což se projeví v kepstrogramech, což může vést k chybné klasifikaci.

Pro otestování hudebního rušení byl využit CTU Test Dataset, jehož promluvy byly smíchány s náhodně vybranými hudebními úsekky vážné hudby pro SNR 10 dB a 20 dB.

Tabulka 5.12 ukazuje, že neznámé nahrávací prostředí nepředstavuje pro strukturu zásadní problém. Je však patrné, že hudební zkreslení velmi zvyšuje chybost při klasifikaci. Pro lepší generalizaci na tyto případy bude třeba rozšířit dataset, či umístit před strukturu filtrační systém.

	Exp. 5	Neznámé RIR	Hudba 10 dB	Hudba 20 dB
FPR [%]	2.352	7.565	58.80	36.41

Tabulka 5.12: Porovnání FPR pro testovací dataset z Exp 5, neznámých RIR a hudebních zkreslení pro SNR 10 dB a 20 dB

5.2.7 Experiment 7 - Neviděné datasety přirozených promluv

Tento experiment měl za cíl ověřit, zda bude struktura schopna dobře klasifikovat přirozené promluvy pocházející z úplně neznámého datasetu, tj. struktura neviděla ani jednu z těchto promluv v průběhu trénování. K tomuto účelu byly využity dva datasety dostupné na pracovišti: CT-Dataset a T-Dataset. CT-Dataset obsahuje 1055 neanotovaných promluv a T-dataset 2245. Experiment probíhal na natrénovaném modelu z podsekce 5.2.5, tj. z Experimentu 5.

Jak je patrné z tabulky 5.13 a z obrázku 5.7a, skóre CT-Datasetu se vyskytuje především v oblasti přirozených promluv, což vede k relativně nízké chybovosti. Toto ukazuje dobrou schopnost modelu generalizovat na neznámé přirozené promluvy.

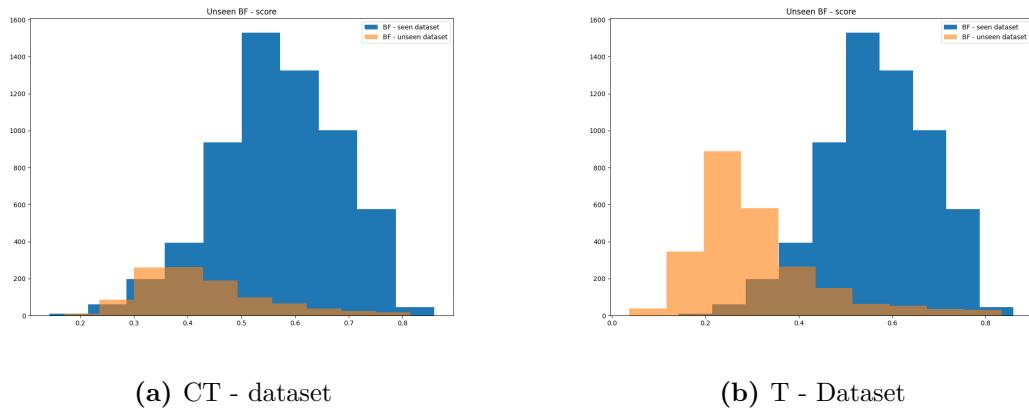
Výsledky s využitím T-Datasetu jsou velmi odlišné. Skóre se pohybuje v prostoru podvržených promluv, viz obrázek 5.7b, a to má za následek $EER = 31.82\%$. To vše ukazuje neschopnost modelu generalizovat na promluvy z tohoto datasetu. Toto zjištění bylo velmi překvapivé, jelikož trénovací dataset obsahuje promluvy od mnoha mluvčích i v různých nahrávacích podmínkách - především promluvy z Common Voice datasetu. Z tohoto důvodu bylo nutné najít příčinu tohoto nepředvídávaného chování struktury. Bylo provedeno několik testů, kdy byla struktura přetrénována s různým nastavením hyperparametrů u ResNetu i OC Softmax, ale bez úspěchu.

	Exp. 5	CT-Dataset	T-Dataset
FPR [%]	2.352	3.948	31.82

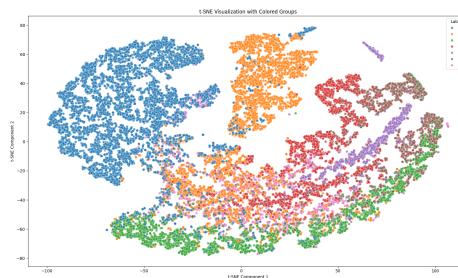
Tabulka 5.13: Porovnání EER u testovacího datasetu (Exp. 5), CT a T datasetu.

Z toho je patrné, že chyba není v klasifikační funkci, ale v příznakových vektorech, které jsou výstupem ResNetu. Pro ověření byly příznakové vektory ze standardního testovacího datasetu i z T-Datasetu zobrazeny pomocí t-SNE do 2D matice, viz obrázek 5.8. Jak je viditelné, výstupní příznakové vektory se nacházejí především v prostoru podvržených promluv.

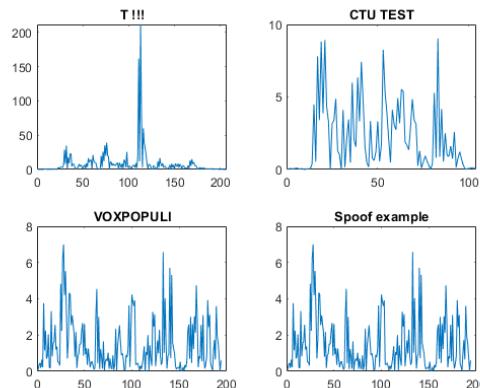
Příčina tohoto chování byla vysvětlena na základě informace poskytnuté autorem datasetu. Tento dataset byl nahráván za pomocí mikrofonu z call-centra, který má v sobě zabudované adaptivní potlačení šumu (ANC - *Adaptive Noise Control*). Jak je vidět na obrázku 5.9,



Obrázek 5.7: Porovnání skóre přirozených promluv testovacího datasetu s CT a T datasety



Obrázek 5.8: t-SNE vizualizace výstupních příznakových vektorů. 0 - přirozené promluvy ze známých podmínek, 1-5 podvržená řeč, 21 - T-Dataset



Obrázek 5.9: Porovnání průběhů stejnosměrné složky pro reprezentativní příklady z jednotlivých datasetů

ANC se projevuje velikými špičkami oproti standardním mikrofonům ve stejnosměrné složce a může způsobovat nespojitosti v signálu. Tyto nahrávací podmínky nejsou však běžné pro standardní typy podvodů a nemusel by na ně být brán zřetel, avšak některé z následujících experimentů zkoumají mimo jiné i vliv jednotlivých úprav na klasifikaci tohoto datasetu.

5.2.8 Experiment 8 - Odstranění c_0

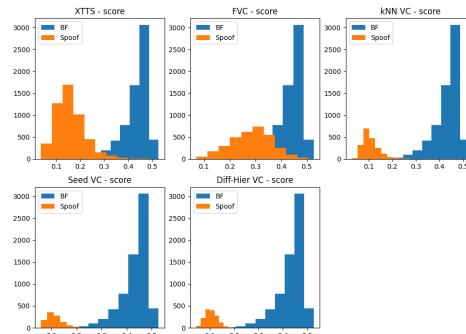
Jelikož se hlavní projevy mikrofonů použitých pro T-dataset projevují nestandardně vysokými špičkami ve stejnosměrné složce signálu, která ovlivňuje energii signálu reprezentovanou nultým kepstrálním koeficientem c_0 , byl v tomto experimentu tento koeficient odebrán společně s jeho Δ a $\Delta\Delta$ koeficienty. Toto řešení je implementačně nenáročné a lze jej zařadit do pipeline, která načítá a předzpracovává předpočítaná kepstra pro neuronovou síť, konkrétně do Dataset. Jak je patrné z tabulek 5.14, 5.15 a histogramů na obrázku 5.10, tato úprava nepřinesla očekávané zlepšení. Došlo k poklesu FNR u XTTS, avšak k jeho navýšení u FVC. Zároveň s tím však nedošlo ke zlepšení u T-Datsetu. Promluvy se hojně vyskytují v prostoru podvržených promluv a $FPR = 49\%$.

	Exp. 5	Exp. 8
Práh	0.260	0.340
EER [%]	4.707	6.792

Tabulka 5.14: Celkové porovnání EER pro výsledky Experimentu 5 a 8

Kategorie/nástroj	Exp. 5	Exp. 8
Přirozené (FPR)	2.3528	3.388
XTTS	1.2160	0.543
FVC	7.15	12.31
kNN VC	0.000	0.000
Seed VC	0.205	0.000
Diff-Hier VC	0.000	0.000

Tabulka 5.15: Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a 8



Obrázek 5.10: Skóre testovacího datasetu pro odstranění c_0

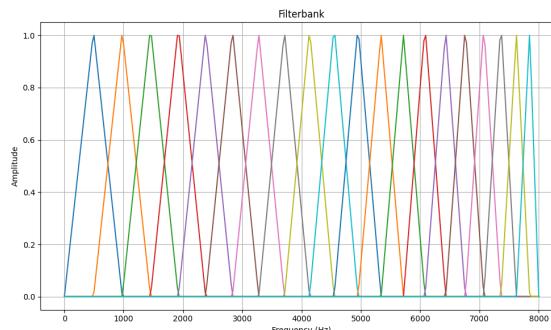
5.2.9 Experiment 9 - Různé banky filtrů

Ve všech předchozích experimentech byly použity LFCC pokrývající celý frekvenční rozsah od 0 do $f_s/2$, přičemž všechny filtry měly stejnou šířku. Jak je patrné z výsledků, struktura využívající těchto příznaků dosahuje dobrých výsledků i přiměřeně dobré generalizace pro neznámé přirozené promluvy i pro neznámé typy útoků. Následující experimenty měly za cíl ověřit, zda volba odlišně tvarované trojúhelníkové banky filtrů nepovede k lepší klasifikaci či generalizaci.

Užší filtry ve vyšších frekvencích

Jak již bylo zmíněno, některé práce předpokládají vyšší výskyt artefaktů způsobených nástroji TTS či VC ve vyšších frekvencích. Z tohoto důvodu byla vytvořena banka, která obsahuje filtry, které se s rostoucí frekvencí zužují tj. vyšší pásma obsahují více filtrů, což vede k vyššímu rozlišení v této oblasti. Použitá banka je znázorněna na obrázku 5.11. Nevýhodou tohoto přístupu může být, že zúžené filtry a nižší energie ve vyšších frekvencích povedou k nižším hodnotám u těchto bank a budou mít nižší vliv v kepstrogramu.

Jak je patrné z tabulek 5.16, 5.17 a obrázku 5.12 tato změna banky filtrů nepřinesla zlepšení v klasifikaci.

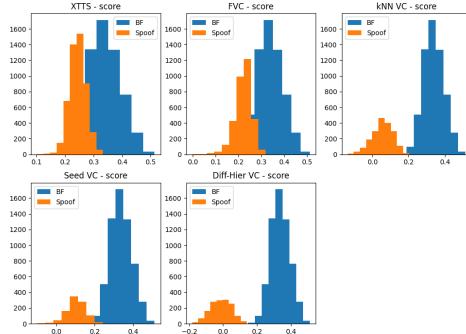


Obrázek 5.11: Banka filtrů se zužující se šírkou

	Exp. 5	Zužující banka
Práh	0.260	0.268
EER [%]	4.707	9.754

Tabulka 5.16: Celkové porovnání EER s výsledky Experimentu 5

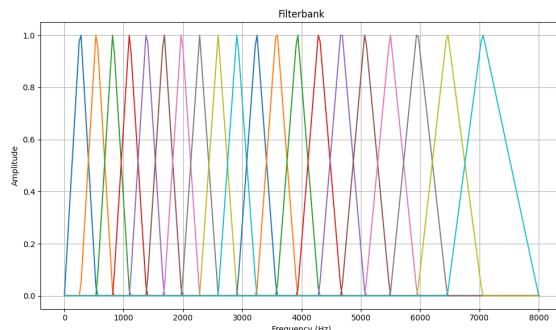
Kategorie/nástroj	Exp. 5	Zužující banka
Přirozené (FPR)	2.3528	4.845
XTTS	1.2160	9.973
FVC	7.15	3.669
kNN VC	0.000	0.000
Seed VC	0.205	0.051
Diff-Hier VC	0.000	0.000

Tabulka 5.17: Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5**Obrázek 5.12:** Skóre testovacího datasetu pro zužující se banku filtrů

Širší filtry filtry ve vyšších frekvencích

V tomto experimentu byla využita banka, u níž se s rostoucí frekvencí filtry rozšiřují. Toto vede k nižšímu rozlišení ve vyšších částech spektra, ale i ke zvýšení energie v těchto pásmech a tudíž k většímu vlivu v kepstrech. Banka je zobrazena na obrázku 5.13.

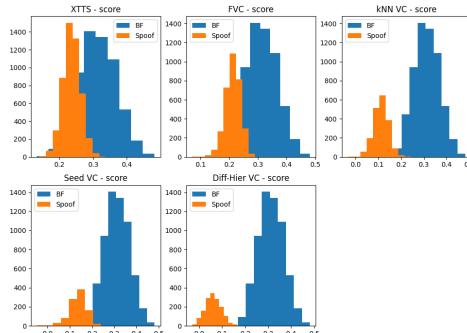
Jak je patrné z tabulek 5.18, 5.19 a obrázku 5.14, tato změna nevedla ke zlepšení klasifikace.

**Obrázek 5.13:** Banka filtrů s rostoucí šírkou

	Exp. 5	Rozšiřující banka
Práh	0.260	0.250
EER [%]	4.707	13.428

Tabulka 5.18: Celkové porovnání EER s výsledky Experimentu 5

Kategorie/nástroj	Exp. 5	Rozšiřující banka
Přirozené (FPR)	2.3528	6.70
XTTS	1.2160	14.69
FVC	7.15	3.459
kNN VC	0.000	0.000
Seed VC	0.205	0.051
Diff-Hier VC	0.000	0.000

Tabulka 5.19: Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5**Obrázek 5.14:** Skóre testovacího datasetu pro rozšiřující se banku filtrů

LFCC s frekvenčním rozsahem 300 - 3400 Hz

Všechny předchozí experimenty pracovaly s bankami filtrů přes celý frekvenční rozsah 0 - $f_s/2$ tj. 0 - 8 000 Hz. V tomto experimentu bylo frekvenční pásmo omezeno na 300 - 3400 Hz, což je standardní pásmo využívané při zpracování řeči, jelikož se v tomto intervalu nacházejí nejdůležitější řečové charakteristiky (formanty apod.). Počet kepstrálních koeficientů zůstal stejný, jelikož struktura zvoleného ResNetu neumožňuje výraznější snížení jejich počtu.

Jak je zřejmé z tabulek 5.20 a 5.21, došlo k výraznému zhoršení klasifikace. Všechny promluvy byly klasifikovány do oblasti $\cos(\theta)$ pro podvržené promluvy, jak je patrné z obrázku 5.15. Existují dvě možné příčiny tohoto výsledku: první spočívá v tom, že se v daném frekvenčním rozsahu nevyskytují artefakty zanesené nástroji TTS/VC; druhá pak v tom, že 20 filtrů je na daný rozsah příliš mnoho a výsledné hodnoty jsou příliš malé. Jelikož není možné snížit počet kepstrálních koeficientů pod 17, byla pro další experiment

kepstra normalizována pomocí min-max normalizace přes celý dataset, aby se zabránilo výskytu příliš nízkých hodnot v kepstrogramu.

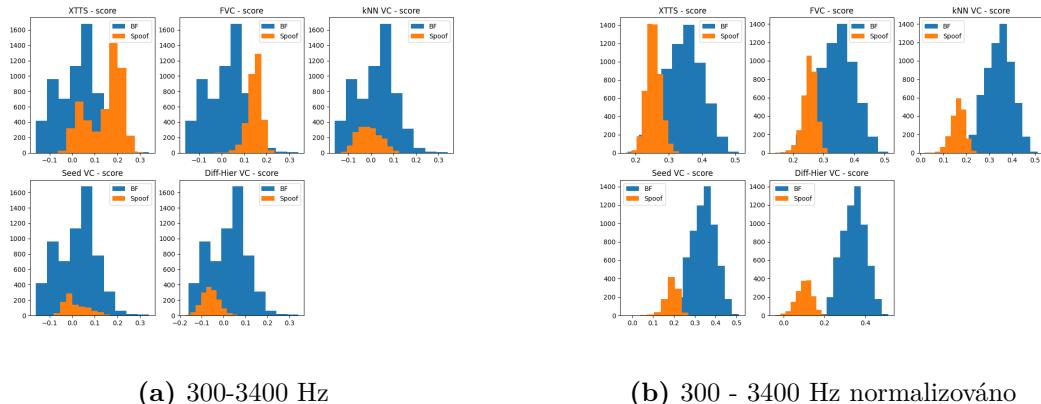
Jak ukazují histogramy i tabulky, normalizace v tomto případě vedla ke zlepšení, což ukazuje, že v tomto frekvenčním intervalu se nacházejí artefakty zanechané nástroji TTS/VC. Před normalizací byly hodnoty kepstrogramů příliš malé pro efektivní zpracování.

	Exp. 5	300 - 3400 Hz	300 - 3400 Hz norm
Práh	0.260	0.054	0.273
EER [%]	4.707	60.67	12.38

Tabulka 5.20: Celkové porovnání EER s výsledky Experimentu 5

Kategorie/nástroj	Exp. 5	300 - 3400 Hz	300-3400 Hz norm
Přirozené (FPR)	2.3528	62.30	6.193
XTTS	1.2160	15.19	9.201
FVC	7.15	43.56	9.805
kNN VC	0.000	23.37	0.000
Seed VC	0.205	3.74	0.102
Diff-Hier VC	0.000	18.064	0.000

Tabulka 5.21: Porovnání ER v jednotlivých kategoriích s výsledky Experimentu 5



(a) 300-3400 Hz

(b) 300 - 3400 Hz normalizováno

Obrázek 5.15: Porovnání skóre přirozených promluv testovacího datasetu pro 300 - 3400 Hz

5.2.10 Experiment 10 - Normalizace

Cílem tohoto experimentu bylo ověření vlivu normalizace a standardizace jednotlivých kepstrálních koeficientů kepstrogramu v čase. Tyto regularizační techniky jsou ve zpracování signálů i ve strojovém učení hojně využívány. Mohou přispět ke stabilnějšímu trénování i lepší generalizaci, protože hodnoty jsou přeskálovány do menšího rozsahu. To je zvláště vhodné pro gradientní algoritmy, které mohou být citlivé na rozsah vstupních dat – velké hodnoty mohou dominovat a malé být ignorovány. Vhodné přeskálování tak může vést k rychlejší konvergenci a lepší generalizaci [80], [81], což může být výhodné zejména v případě T-datasetu.

Mean-std normalizace (Z - Skóre standardizace)

Při této standardizaci dochází k odečtení střední hodnoty a vydelení směrodatnou odchylkou, viz rovnice 5.1

$$\mathbf{x}_{norm} = \frac{\mathbf{x} - \mu_x}{\sigma_x}, \quad (5.1)$$

kde \mathbf{x} je originální signál (příznakový vektor), μ_x střední hodnota a σ_x směrodatná odchylka.

V prvním kroku byla využita normalizace přes trénovací dataset. Z kepstrogramů trénovací množiny byla spočtena střední hodnota a směrodatná odchylka pro všech 60 koeficientů. Tyto hodnoty byly následně využity při trénování i testování, což odpovídá reálné situaci, kdy jsou testovací data předem neznámá.

Výsledky zobrazené v tabulkách 5.22 a 5.23 ukazují nezlepšení celkové klasifikace, celkové EER je vyšší. U známých kategorií došlo k mírnému nárůstu ER. U XTTS došlo k nárůstu o 3.627 %, zatímco u FVC došlo k poklesu o 4.789 %. Změna tvaru histogramu skóre je patrná z obrázku 5.16a.

Jelikož výsledky s normalizací přes dataset nebyly uspokojivé, byla vyzkoušena normalizace přes promluvu, tj. pro každé kepstrum a každý koeficient byla spočtena μ a σ . Jak je vidět z výsledků níže, došlo ke zhorskání ve všech kategoriích.

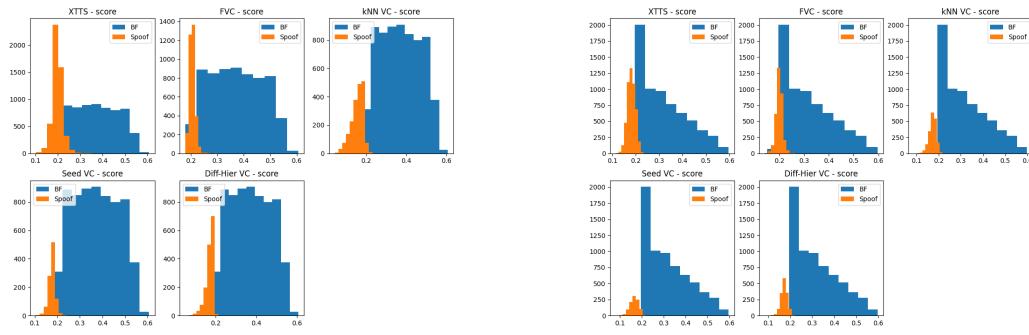
	Exp. 5	Norm. dataset	Norm. promluva
Práh	0.260	0.2238	0.209
EER [%]	4.707	5.092	7.016

Tabulka 5.22: Celkové porovnání EER pro výsledky Experimentu 5 a normovaných kepstrogramů

Dále byly ověřeny účinky normalizace příznaků při klasifikaci na T-datasetu. Jak je patrné z obrázku 5.17, skóre se stále nachází v prostoru podvržených promluv a EER činí **33.82 %** pro normalizaci přes dataset a **22.49%** pro normalizaci přes promluvu.

Kategorie/nástroj	Exp. 5	Norm. dataset	Norm. promluva
Přirozené	2.352	2.546	3.507
XTTS	1.216	4.843	2.085
FVC	7.150	2.361	10.09
kNN VC	0.000	0.08	0.207
Seed VC	0.205	0.247	0.296
Diff-Hier VC	0.000	0.033	0.261

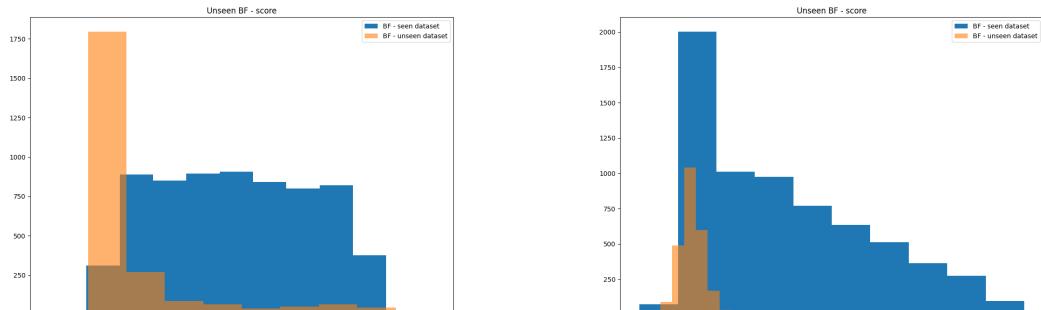
Tabulka 5.23: Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a normovaných kepstrogramů přes dataset



(a) Standardizace přes dataset

(b) Standardizace přes promluvu

Obrázek 5.16: Skóre testovacího datasetu při standardizaci



(a) Standardizace přes dataset

(b) Standardizace přes promluvu

Obrázek 5.17: Skóre T - datasetu při standardizaci

Z experimentů je patrné, že tento typ normalizace nepředstavuje pro zvolenou strukturu celkové vylepšení. Při normalizaci přes dataset je však zajímavý pokles ER u XTTS a nárůst u FVC. „Statisticky nelze určit, zda se jedná o potenciální výhodu; k potvrzení bylo třeba dataset doplnit o další neviděné typy útoků.“

Min-Max normalizace

Při této normalizaci dochází k odečtení minimální hodnoty a vydělení rozsahem hodnot, viz

$$\mathbf{x}_{norm} = \frac{\mathbf{x} - x_{min}}{x_{max} - x_{min}}, \quad (5.2)$$

kde \mathbf{x} je originální signál (vektor kepstrálního koeficientu), x_{min} minimum a x_{max} maximum. Tato metoda převede hodnoty do intervalu $[0,1]$. Potenciální výhoda tohoto přístupu spočívá v zachování tvaru distribuce prvků.

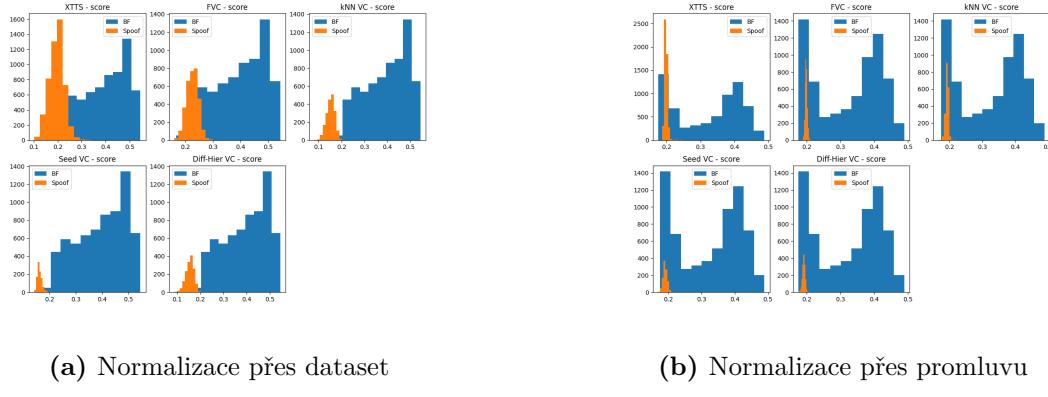
V tomto kroku byly využity normalizace přes promluvu a následně i normalizace přes dataset. Výsledky v tabulkách 5.24 a 5.25 ukazují, že lepších výsledků dosahuje normalizace přes dataset, byť stále nedosahuje výsledků Experimentu 5. Z histogramů na obrázcích 5.18 je patrné, že normalizace přes promluvu vede ke špatné klasifikaci především přirozených promluv a dochází k výraznému překryvu mezi přirozenými a podvrženými promluvami.

	Exp. 5	Norm. dataset	Norm. promluva
Práh	0.260	0.243	0.202
EER [%]	4.707	4.565	9.891

Tabulka 5.24: Celkové porovnání EER pro výsledky Experimentu 5 a normovaných kepstrogramů

Kategorie/nástroj	Exp. 5	Norm. dataset	Norm. promluva
Přirozené (FPR)	2.352	3.783	4.937
XTTS	1.216	2.560	7.956
FVC	7.150	10.752	5.925
kNN VC	0.000	0.000	0.594
Seed VC	0.205	0.000	2.019
Diff-Hier VC	0.000	0.000	0.261

Tabulka 5.25: Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 5 a normovaných kepstrogramů



Obrázek 5.18: Porovnání skóre přirozených promluv testovacího datasetu pro min-max normalizaci

5.2.11 Experiment 12 - Důležité koeficienty

Cílem tohoto experimentu bylo zjistit, které z koeficientů mají nejvyšší přínos pro klasifikaci, tj. kterým je věnována největší pozornost. K tomuto byl využit model z Experimentu 5 a tzv. mapy významnosti (Saliency Maps).

Tyto mapy slouží k vizualizaci jednotlivých vrstev sítě či významných oblastí ve vstupních datech neuronové sítě, což umožňuje lepší pochopení jejího fungování. Pro jejich výpočet existuje mnoho algoritmů. Zde byl zvolen gradientní algoritmus, který lze zapsat jako:

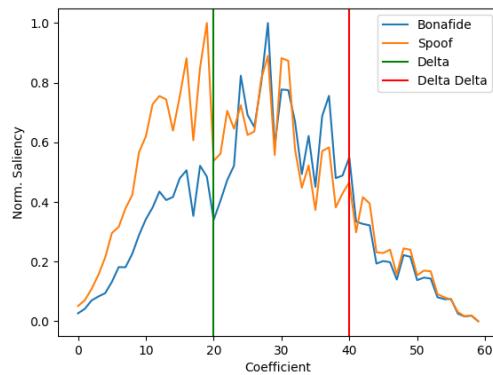
$$S_{c,t} = \frac{\partial f(x)}{\partial x_{c,t}}, \quad (5.3)$$

kde S je mapa, x vstupní kepstrogram, c kepstrální koeficient, $f(x)$ klasifikační skóre a t čas. Mapa má velikost původního kepstrogramu a pro každý jeho prvek udává, jak moc jeho změna ovlivní výstupní skóre.

Z obrázku 5.19 je patrné, že pro klasifikaci jsou významné především vyšší kepstrální a Δ koeficienty. Toto ukazuje na to, že použité nástroje TTS/VC umí dobře napodobit děletrvající charakteristiky, tj. spektrální obálku (tvar vokálního traktu), ale nedokážou věrně napodobit změny spektrální obálky v čase (nižší Δ koeficienty) ani vlastnosti buzení řečového ústrojí a jejich dynamiku (vyšší kepstrální a Δ koeficienty). Z tvaru obou křivek je patrné, že pro klasifikaci obou skupin využívá struktura téměř stejných koeficientů, což je velmi vhodné. Dalším z nutných závěrů je to, že systém automatické kontroly zesílení a potlačení šumu u T - Datasetu zasahuje i do vyšších kepstrálních koeficientů.

5.2.12 Experiment 13 - Paralelní větev

Tento experiment měl dva cíle. Prvním bylo ověření, zda výměna XTTS za DiffHier VC v testovacím datasetu, bude mít nějaký dopad na klasifikaci, tj. zda schopnost generalizace na neviděné typy útoků platí i pro jiné kombinace trénovacích a testovacích dat.



Obrázek 5.19: Mapa významnosti kepstrálních koeficientů pro přirozené a podvržené promluvy - normalizováno

Druhým cílem bylo ověření vlivu paralelní větve v síti, která předává informace ze vstupních příznaků přímo před poslední plně propojenou vrstvu, kde se spojují s výstupem ze Self Attention Pooling vrstvy. Nejprve byly využity dvě základní statistiky — směrodatná odchylka a rozdíl mezi maximem a minimem hodnot kepstrálních koeficientů v čase, které popisují jejich dynamiku a rozsah. Postupně byl doplněn také rozdíl mezi maximem a střední hodnotou a nakonec počet průchodů nulou u Δ a $\Delta\Delta$ koeficientů (ZCR, z anglického *Zero Crossing Rate*).

Tyto ručně navržené charakteristiky zachycují různé aspekty časové variability řečového signálu, od rozsahu a rozložení hodnot až po četnost změn jejich znaménka. Předpokládá se přitom, že přirozená řeč vykazuje větší míru dynamiky, variability a nepravidelnosti než řeč syntetická, která bývá často hladší, symetričtější a spektrálně méně rozmanitá. Začleněním těchto charakteristik do modelu tedy může dojít k doplnění lokálních reprezentací o globálnější signálové vlastnosti a tím i ke zlepšení schopnosti rozlišovat mezi autentickou a podvrženou řečí.

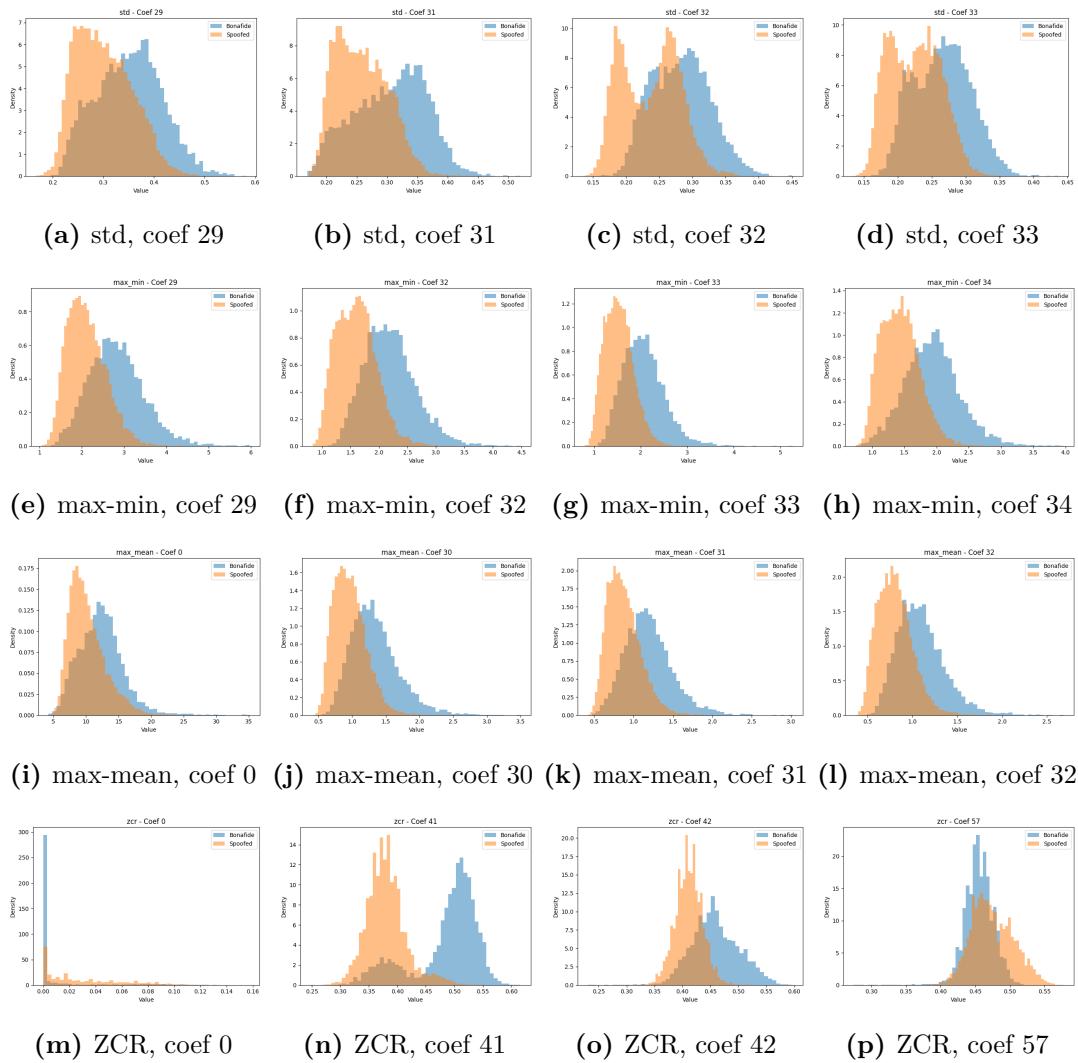
	Bez větve	σ a (max-min)	$+(max-\mu)$	+ZCR
Práh	0.260	0.1803	0.218	0.209
EER [%]	2.6476	2.262	2.220	1.81

Tabulka 5.26: Celkové porovnání EER pro výsledky Experimentu 13 a paralelní větve

Jak je patrné z tabulek 5.26 a 5.27, přidání paralelní větve se statistickými informacemi přineslo zlepšení v klasifikaci, celkově o 0.8 %, což ukazuje ve prospěch hypotézy. Potvrzení hypotézy je viditelné i z obrázku 5.20, kde jsou pro každý z parametrů zobrazeny 4 histogramy pro 4 kepstrální koeficienty s největším distribučním rozdílem pro přirozené a podvržené promluvy z testovacího datasetu. Největší distribuční rozdíl je patrný pro ZCR a koeficient 41 tj. pro 1. $\Delta\Delta$ koeficient. Zajímavé je, že některé z nástrojů TTS/VC mají vliv na energii signálu, projevují se vyšší mírou ZCR u logaritmické energie.

Kategorie/nástroj	Bez větve	σ_a (m-m)	$+(m-\mu)$	+ZCR
Přirozené (FPR)	1.322	1.123	1.098	0.899
XTTS	0.000	0.033	0.000	0.066
FVC	4.015	3.669	1.970	0.797
kNN VC	0.000	0.154	0.000	0.154
Seed VC	0.000	0.050	0.299	0.000
Diff-Hier VC	0.752	0.443	1.475	1.711

Tabulka 5.27: Porovnání ER (FPR/FNR) v jednotlivých kategoriích pro výsledky Experimentu 13 a paralelní větve



Obrázek 5.20: Distribuce parametrů pro 4 nejdiskriminativní koeficienty pro každý parametr (std, max-min, max-mean, ZCR).

5.2.13 Další provedené experimenty

Tato podsekce stručně popisuje další provedené experimenty, které nedosáhly lepších výsledků, než uvedené, ale mohly by posloužit jako inspirace k důkladnějšímu rozpracování pro navazující práce.

Pozornostní mechanika založená na transformerech

V tomto případě byla místo pozornostního mechanismu popsáno v podsekci 3.2.2 použita struktura založená na `torch.nn.TransformerEncoderLayer` se čtyřmi vrstvami a hlavami s využitím pevného i relativního pozičního kódování. Výsledky byly podobné současné využívanému mechanismu.

Více centrálních vektorů

Zde bylo cílem lépe popsat prostor přirozených promluv a roztahnout jejich kosinové podobnosti přes více hodnot. Bylo vytvořeno 10 centrálních vektorů, skóre bylo počítáno od nejbližšího k příznakovému vektoru. Aby nedošlo ke kolapsu vektorů do jednoho, byla při trénování zavedena penalizace, tlačící vektory do ortogonální polohy. Aby bylo zaručeno rozprostření příznakových vektorů po centrálních, byla z kosinových podobností mezi příznakovým vektorem a centrálními vektory počítána entropie, která byla následně škálována a odečtena od ztrátové funkce. V prvním případě byly vektory podvržených promluv tlačeny za r_{fake} od centrálního vektoru s $\min \cos(\theta)$. Dále bylo vyzkoušeno počítat kosinovou podobnost podvržených promluv od vektoru, který vznikl jako vektor středních hodnot centrálních vektorů. V posledním případě byl využit mechanismus obdobný jako u přirozených promluv, tj. více center pro podvržené promluvy. Metoda využívající maximální kosinové podobnosti a metoda založená na více centrech pro přirozené i podvržené promluvy dosáhly výsledků blízkých OC Softmax + Margin.

Grad Aug

GradAug je metoda augmentace dat, která zlepšuje generalizaci modelu tím, že využívá gradient ztrátové funkce vzhledem ke vstupu k vytvoření upravených trénovacích příkladů. Tyto příklady vznikají přidáním malé perturbace ve směru, který maximalizuje chybu modelu, čímž se vytvářejí náročnější vzorky blízké rozhodovací hranici. Tím se model učí být robustnější vůči drobným, ale významným změnám ve vstupních datech, což snižuje riziko přeúčení a zvyšuje odolnost vůči neznámým či odlišným vstupům. GradAug se osvědčuje zejména v úlohách s omezeným množstvím dat nebo při detekci jemných rozdílů. Při trénování bylo této metody využito každou desátou dávku. Metoda nepřispěla ke zlepšení.

FFT2

V tomto případě byla na vstupní kepstrogramy aplikována 2D Fourierova transformace, která dokáže zachytit periodicitu, nepravidelnosti ve struktuře kepster, přes časovou i kvefrenční rovinu. Spektrum bylo centrováno. Struktura dosáhla s těmito příznaky mírně horší výsledky: $EER = 3.22\%$.

Heterogennější dataset

V tomto experimentu byla struktura natrénována na bohatším datasetu. Základní databáze byla doplněna promluvami z CT-Datasetu i promluvami z CTU Testu zkonzolvovanými s RIR. Prvním z cílů bylo ověření vlivu heterogennějšího datasetu přirozených promluv na klasifikaci jednotlivých útoků. Zde byly výsledky srovnatelné s Experimentem 5. Druhým cílem pak bylo ověření klasifikace přirozených promluv z kompletně neznámého datasetu, tj. s jinými řečníky i nahrávacími podmínkami. Pro ověření již nezbýval při tvorbě této práce čas na nalezení a standardizaci formátu nového datasetu. Konzolvovat s RIR promluvy z Voxpopuli nebo Common Voice datasetu, nedává smysl, jelikož promluvy z nich vlastní slyšitelné konvoluční zkreslení způsobené nahrávacím prostředím mají.

Klasifikace pomocí GMM

V tomto experimentu byla nahrazena OC Softmax GMM klasifikátorem založeným na dvou modelech: jedním pro přirozené promluvy, druhým pro podvržené. Vstupem do GMM byly výstupní příznakové vektory z ResNet. Výsledná klasifikace byla obdobná s OC Softmax, pro viděné útoky dobrá, pro neviděné struktura nedokázala generalizovat.

6 Závěr

Tato práce se zabývala detekcí podvržené řeči pro český jazyk s využitím struktury ResNet a jednotřídrové klasifikace. V první fázi byly vybrány vhodné nástroje TTS a VC, které byly upraveny společně s databázemi přirozené řeči k tvorbě českého foneticky bohatého datasetu podvržené řeči, obsahující dobré množství řečníků. Následně byla navržena metrika pro výběr kvalitně vygenerovaných promluv s ohledem na český jazyk.

Dataset byl využit v experimentální části pro ověření fungování struktury ResNet + OC Softmax. Výsledky ukázaly, že zvolená struktura dobře klasifikuje pro známé typy útoků, ale velmi špatně se adaptuje na neznámé. Příčinou se ukázala klasifikační funkce OC Softmax, která tlačí výstupní příznakové vektory silně k zadaným hodnotám kosinových podobností a neumožňuje jejich širší distribuci, která je klíčová pro generalizaci. Dobrým řešením se ukázalo přidání protisíly do OC Softmax, která nutí vzorky přirozené řeči se rozprostřít a mírně je tlačí směrem k prostoru kosinových podobností pro podvržené promluvy.

Následně byly testovány různé konfigurace vstupních příznaků a různé nahrávací podmínky. Byly využity různé tvary bank filtrů generujících kepstrogramy, zkoumaly se různé formy normalizace příznaků v čase i odstranění potenciálně nepotřebných koeficientů. Výsledky ukázaly, že 20 LFCC bez normalizace či úprav poskytují nejvyšší schopnost správné klasifikace, tj. rozdelení kosinových podobností pro přirozené a podvržené promluvy.

Schopnost sítě generalizovat byla ověřena pomocí dvou neviděných datasetů přirozených promluv a datasetu s neznámými RIR. Síť prokázala jistou schopnost generalizace pro většinu standardních podmínek, výjimkou byl pouze T-dataset, vytvořený s mikrofony obsahující ANC, které zanechávají značné artefakty v řečovém signálu.

V poslední fázi experimentální části byla do sítě zavedena paralelní větev, která přenáší statistiky napočtené ze vstupních kepstrogramů před poslední plně propojenou vrstvu. Jak se ukázalo, tento přístup vedl ke zlepšení klasifikace, což dokazuje částečnou schopnost diskriminace mezi přirozenými a podvrženými promluvami na základě těchto statistik. Tato schopnost byla ukázána i za pomocí distribucí jednotlivých statistik pro přirozené i podvržené promluvy.

Celkově struktura dosáhla pro český jazyk dobrých výsledků, ale další práce v této oblasti bude nutná. Především bude nutné rozšířit dataset o další nástroje TTS/VC i o další nahrávací podmínky či řečníky, aby bylo možné ověřit schopnost použité struktury generalizovat. Toto by bylo již přes rozsah této diplomové práce. Tato práce byla na pracovišti prvního druhu, a proto je poměrně obecná co se týče podmínek detekce. Budoucí práce budou muset brát v potaz účel a způsob použití detekce podvržené řeči i potenciální artefakty s tím spojené např. zkreslení telefonním kanálem. Od toho se bude odvíjet i volba použité struktury.

Bibliografie

- [1] J. Flanagan, *Speech Analysis, Synthesis and Perception*. Springer-Verlag Berlin Heidelberg GmbH, 1972.
- [2] K. e. Tokuda, „Speech Synthesis Based on Hidden Markov Models“, *Proceedings of the IEEE*, 2013.
- [3] P. Pollák, „BE2M31ZRE - Speech Synthesis“, University Lecture at FEE CTU Prague, 2024.
- [4] J. Ao, R. Wang, L. Zhou et al., *SpeechT5: Unified-Modal Encoder-Decoder Pre-Training for Spoken Language Processing*, 2022. arXiv: 2110.07205 [eess.AS]. URL: <https://arxiv.org/abs/2110.07205>.
- [5] R. Prenger, R. Valle a B. Catanzaro, *WaveGlow: A Flow-based Generative Network for Speech Synthesis*, 2018. arXiv: 1811.00002 [cs.SD]. URL: <https://arxiv.org/abs/1811.00002>.
- [6] „SpeechT5-base-cs-tts“. (), URL: <https://huggingface.co/fav-kky/SpeechT5-base-cs-tts> (cit. 11.01.2025).
- [7] „Speecht5 finetuned voxpopuli cs“. (), URL: <https://huggingface.co/kfahn/speecht5%5C%5Ffinetuned%5C%5Fvoxpopuli%5C%5Fcs/tree/main> (cit. 11.01.2025).
- [8] C. Wang, M. Rivière, A. Lee et al., *VoxPopuli: A Large-Scale Multilingual Speech Corpus for Representation Learning, Semi-Supervised Learning and Interpretation*, 2021. arXiv: 2101.00390 [cs.CL]. URL: <https://arxiv.org/abs/2101.00390>.
- [9] „Zero-Shot vs. Few-Shot Multi-Speaker TTS Using Pre-trained Czech SpeechT5 Model“. (), URL: <https://jalehecka.github.io/TSD2024/> (cit. 11.01.2025).
- [10] J. Lehečka, Z. Hanzlíček, J. Matoušek a D. Tihelka, „Zero-Shot vs. Few-Shot Multi-speaker TTS Using Pre-trained Czech SpeechT5 Model“, in *Text, Speech, and Dialogue*. Springer Nature Switzerland, 2024, s. 46–57, ISBN: 9783031705663. DOI: 10.1007/978-3-031-70566-3_5. URL: <http://dx.doi.org/10.1007/978-3-031-70566-3%5C%5F5>.
- [11] „Tacotron: An end-to-end speech synthesis system by Google“. (), URL: <https://google.github.io/tacotron/index.html> (cit. 11.01.2025).
- [12] Y. Ren, Y. Ruan, X. Tan et al., *FastSpeech: Fast, Robust and Controllable Text to Speech*, 2019. arXiv: 1905.09263 [cs.CL]. URL: <https://arxiv.org/abs/1905.09263>.
- [13] „How to use text-to-speech with low vision“. (), URL: <https://www.perkins.org/resource/ways-to-read-webpages-without-a-traditional-screen-reader/> (cit. 16.11.2024).
- [14] „The 2024 Guide To Chatbots In Banking“. (), URL: <https://springsapps.com/knowledge/the-2024-guide-to-chatbots-in-banking> (cit. 16.11.2024).
- [15] S. O. Arik, J. Chen, K. Peng, W. Ping a Y. Zhou, *Neural Voice Cloning with a Few Samples*, 2018. arXiv: 1802.06006 [cs.CL]. URL: <https://arxiv.org/abs/1802.06006>.

- [16] H.-T. Luong a J. Yamagishi, „NAUTILUS: A Versatile Voice Cloning System“, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, roč. 28, s. 2967–2981, 2020. DOI: 10.1109/TASLP.2020.3034994.
- [17] „Unusual CEO Fraud via Deepfake Audio Steals USD 243,000 From UK Company“. (), URL: <https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/unusual-ceo-fraud-via-deepfake-audio-steals-us-243-000-from-u-k-company> (cit. 17. 11. 2024).
- [18] „AI voice cloning is used in a huge heist being investigated by Dubai investigators, amidst warnings about cybercriminal use of the new technology.“ (), URL: <https://www.forbes.com/sites/thomasbrewster/2021/10/14/huge-bank-fraud-uses-deep-fake-voice-tech-to-steal-millions/?sh=5f1a42037559/> (cit. 17. 11. 2024).
- [19] „CEO of world’s biggest ad firm targeted by deepfake scam“. (), URL: <https://www.theguardian.com/technology/article/2024/may/10/ceo-wpp-deepfake-scam> (cit. 17. 11. 2024).
- [20] „Údajná nahrávka telefonátu predsedu PS a novinárky Denníka N vykazuje podľa expertov početné známky manipulácie“. (), URL: <https://fakty.afp.com/doc.afp.com.33WY9LF> (cit. 16. 11. 2024).
- [21] „A fake recording of a candidate saying he’d rigged the election went viral. Experts say it’s only the beginning“. (), URL: <https://edition.cnn.com/2024/02/01/politics/election-deepfake-threats-invs/index.html> (cit. 16. 11. 2024).
- [22] „Deepfake audio of Sir Keir Starmer released on first day of Labour conference“. (), URL: <https://news.sky.com/story/labour-faces-political-attack-after-deepfake-audio-is-posted-of-sir-keir-starmer-12980181> (cit. 16. 11. 2024).
- [23] „S falešnými videi se roztrhl pytel, zneužívají i značku CNN Prima NEWS. Jak se jim bránit?“ (), URL: <https://cnn.iprima.cz/deepfake-cnn-prima-news-falesne-video-podvod-jak-je-poznat-439783> (cit. 12. 01. 2025).
- [24] „ANALÝZA: Rozdám 50 milionů korun, slibuje ve falešném videu Procházka. Jak poznat podvod?“ (), URL: <https://cnn.iprima.cz/analyza-jiri-prochazka-50-milionu-korun-podvod-umela-inteligence-437902> (cit. 12. 01. 2025).
- [25] „MrBeast calls TikTok ad showing an AI version of him a ’scam’“. (), URL: <https://www.nbcnews.com/tech/mrbeast-ai-tiktok-ad-deepfake-rcna118596> (cit. 12. 01. 2025).
- [26] „Automatic speaker verification systems“. (), URL: <https://antispoofing.org/automatic-speaker-verification-systems/#speaker-identification-vs-speaker-verification> (cit. 17. 11. 2024).
- [27] Z. K. Abdul a A. K. Al-Talabani, „Mel Frequency Cepstral Coefficient and its Applications: A Review“, *IEEE Access*, roč. 10, s. 122 136–122 158, 2022. DOI: 10.1109/ACCESS.2022.3223444.
- [28] P. Sovka a P. Pollák, *Vybrané metody číslicového zpracování signálů*. Vydavatelství ČVUT, 2003, ISBN: 80-01-02821-6.

- [29] B. Blankertz. „The Constant Q Transform“. (), URL: <https://doc.ml.tu-berlin.de/bbci/material/publications/Bla%5C%5FconstQ.pdf> (cit. 17.11.2024).
- [30] X. Valero a F. Alías-Pujol, „Gammatone Cepstral Coefficients: Biologically Inspired Features for Non-Speech Audio Classification“, *Multimedia, IEEE Transactions on*, roč. 14, s. 1684–1689, pros. 2012. DOI: [10.1109/TMM.2012.2199972](https://doi.org/10.1109/TMM.2012.2199972).
- [31] S. S. Jagtap a D. Bhalke, „Speaker verification using Gaussian Mixture Model“, in *2015 International Conference on Pervasive Computing (ICPC)*, 2015, s. 1–5. DOI: [10.1109/PERVASIVE.2015.7087080](https://doi.org/10.1109/PERVASIVE.2015.7087080).
- [32] B. Blankertz. „The Constant Q Transform“. (), URL: <https://doc.ml.tu-berlin.de/bbci/material/publications/Bla%5C%5FconstQ.pdf> (cit. 17.11.2024).
- [33] W. M. Campbell, J. P. Campbell, T. P. Gleason, D. A. Reynolds a W. Shen, „Speaker Verification Using Support Vector Machines and High-Level Features“, *IEEE Transactions on Audio, Speech, and Language Processing*, roč. 15, č. 7, s. 2085–2094, 2007. DOI: [10.1109/TASL.2007.902874](https://doi.org/10.1109/TASL.2007.902874).
- [34] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey a S. Khudanpur, „X-Vectors: Robust DNN Embeddings for Speaker Recognition“, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018. DOI: [10.1109/ICASSP.2018.8461375](https://doi.org/10.1109/ICASSP.2018.8461375).
- [35] S. Ioffe, „Probabilistic Linear Discriminant Analysis“, in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof a A. Pinz, ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, s. 531–542, ISBN: 978-3-540-33839-0.
- [36] P. Gupta, H. A. Patil a R. C. Guido. „Vulnerability issues in Automatic Speaker Verification (ASV) systems“. (2014).
- [37] X. Wang, J. Yamagishi, M. Todisco et al., *ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech*, 2020. arXiv: [1911.01601](https://arxiv.org/abs/1911.01601).
- [38] A. S. consortium. „LFCC-GMM baseline model“. (), URL: <https://www.asvspoof.org/asvspoof2015/I3A.pdf> (cit. 08.01.2025).
- [39] J. Villalba, A. Miguel, A. Ortega a E. Lleida. „Spoofing Detection with DNN and One-class SVM for the ASVspoof 2015 Challenge“. (), URL: <https://www.asvspoof.org/asvspoof2015/I3A.pdf> (cit. 08.01.2025).
- [40] A. Baevski, H. Zhou, A. Mohamed a M. Auli, *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*, 2020. arXiv: [2006.11477 \[cs.CL\]](https://arxiv.org/abs/2006.11477). URL: <https://arxiv.org/abs/2006.11477>.
- [41] K. He, X. Zhang, S. Ren a J. Sun, *Deep Residual Learning for Image Recognition*, 2015. arXiv: [1512.03385 \[cs.CV\]](https://arxiv.org/abs/1512.03385). URL: <https://arxiv.org/abs/1512.03385>.
- [42] Y. Zhang, F. Jiang a Z. Duan, „One-Class Learning Towards Synthetic Voice Spoofing Detection“, *IEEE Signal Processing Letters*, roč. 28, s. 937–941, 2021, ISSN: 1558-2361. DOI: [10.1109/LSP.2021.3076358](https://doi.org/10.1109/LSP.2021.3076358). URL: <http://dx.doi.org/10.1109/LSP.2021.3076358>.

- [43] M. Alzantot, Z. Wang a M. Srivastava, *Deep Residual Neural Networks for Audio Spoofing Detection*, čvn. 2019. DOI: 10.48550/arXiv.1907.00501.
- [44] S. Scardapane, L. Stoffl, F. Röhrbein a A. Uncini, „On the use of deep recurrent neural networks for detecting audio spoofing attacks“, in *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, s. 3483–3490. DOI: 10.1109/IJCNN.2017.7966294.
- [45] L. Huang a J. Zhao, „On the Use of LSTM-RNN for Detecting Audio Spoofing Attacks“, in *2022 International Conference on 3D Immersion, Interaction and Multi-sensory Experiences (ICDIIME)*, 2022, s. 147–150. DOI: 10.1109/ICDIIME56946.2022.00040.
- [46] Z. Su, „End-to-End Spoofing Speech Detection based on CNN-LSTM“, in *2022 4th International Conference on Frontiers Technology of Information and Computer (ICFTIC)*, 2022, s. 755–758. DOI: 10.1109/ICFTIC57696.2022.10075096.
- [47] Z. Teng, Q. Fu, J. White, M. E. Powell a D. C. Schmidt, „ARawNet: A Light-weight Solution for Leveraging Raw Waveforms in Spoof Speech Detection“, in *2022 26th International Conference on Pattern Recognition (ICPR)*, 2022, s. 692–698. DOI: 10.1109/ICPR56361.2022.9956138.
- [48] K. He, X. Zhang, S. Ren a J. Sun, *Identity Mappings in Deep Residual Networks*, 2016. arXiv: 1603.05027 [cs.CV]. URL: <https://arxiv.org/abs/1603.05027>.
- [49] „e2e antispoofting“. (), URL: https://github.com/joaomonteirof/e2e_antispoofting (cit. 17. 05. 2025).
- [50] K. Muhammad Hasam a S. Woo, „OC-FakeDect: Classifying Deepfakes Using One-class Variational Autoencoder“, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, čvn. 2020, s. 2794–2803. DOI: 10.1109/CVPRW50498.2020.00336.
- [51] F. Wang, J. Cheng, W. Liu a H. Liu, „Additive Margin Softmax for Face Verification“, *IEEE Signal Processing Letters*, roč. 25, č. 7, s. 926–930, čvc. 2018, ISSN: 1558-2361. DOI: 10.1109/LSP.2018.2822810. URL: <http://dx.doi.org/10.1109/LSP.2018.2822810>.
- [52] „Softplus - Pytorch 2.7 Documentation“. (), URL: <https://docs.pytorch.org/docs/stable/generated/torch.nn.Softplus.html> (cit. 07. 05. 2025).
- [53] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi a M. Sahidullah, „ASVspoof 2015: the First Automatic Speaker Verification Spoofing and Countermeasures Challenge“, zář. 2015. DOI: 10.21437/Interspeech.2015-462.
- [54] X. Liu, X. Wang, M. Sahidullah et al., „ASVspoof 2021: Towards Spoofed and Deepfake Speech Detection in the Wild“, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, roč. 31, s. 2507–2522, 2023, ISSN: 2329-9304. DOI: 10.1109/taslp.2023.3285283. URL: <http://dx.doi.org/10.1109/TASLP.2023.3285283>.
- [55] H. Khalid, S. Tariq, M. Kim a S. S. Woo, „FakeAVCeleb: A Novel Audio-Video Multimodal Deepfake Dataset“, in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL: <https://openreview.net/forum?id=TAXFsg6Za01>.

- [56] R. Reimao a V. Tzerpos, „FoR: A Dataset for Synthetic Speech Detection“, říj. 2019, s. 1–10. DOI: 10.1109/SPED.2019.8906599.
- [57] X. Wang a J. Yamagishi, *Spoofed training data for speech spoofing countermeasure can be efficiently created using neural vocoders*, 2023. arXiv: 2210.10570 [eess.AS]. URL: <https://arxiv.org/abs/2210.10570>.
- [58] N. M. Müller, P. Czempin, F. Dieckmann, A. Froghyar a K. Böttinger, „Does audio deepfake detection generalize?“, *Interspeech*, 2022.
- [59] L. Zhang, X. Wang, E. Cooper, N. Evans a J. Yamagishi, „The PartialSpoof Database and Countermeasures for the Detection of Short Fake Speech Segments Embedded in an Utterance“, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, roč. 31, s. 813–825, 2023, ISSN: 2329-9304. DOI: 10.1109/taslp.2022.3233236. URL: <http://dx.doi.org/10.1109/TASLP.2022.3233236>.
- [60] J. Frank a L. Schönherr, *WaveFake: A Data Set to Facilitate Audio Deepfake Detection*, 2021. arXiv: 2111.02813 [cs.LG]. URL: <https://arxiv.org/abs/2111.02813>.
- [61] J. Yi, R. Fu, J. Tao et al., *ADD 2022: the First Audio Deep Synthesis Detection Challenge*, 2024. arXiv: 2202.08433 [cs.SD]. URL: <https://arxiv.org/abs/2202.08433>.
- [62] J. Yi a C. Y. Zhang, *ADD 2023 Challenge Track 1.2 Training/Development Dataset*, Zenodo, čvc. 2024. DOI: 10.5281/zenodo.12151404. URL: <https://doi.org/10.5281/zenodo.12151404>.
- [63] Z. Zhang, Y. Gu, X. Yi a X. Zhao, „FMFCC-A: A Challenging Mandarin Dataset for Synthetic Speech Detection“, in *Digital Forensics and Watermarking: 20th International Workshop, IWDW 2021, Beijing, China, November 20-22, 2021, Revised Selected Papers*, Beijing, China: Springer-Verlag, 2021, s. 117–131, ISBN: 978-3-030-95397-3. DOI: 10.1007/978-3-030-95398-0_9. URL: <https://doi.org/10.1007/978-3-030-95398-0%5C%5F9>.
- [64] J. Yi, Y. Bai, J. Tao et al., *Half-Truth: A Partially Fake Audio Detection Dataset*, 2023. arXiv: 2104.03617 [cs.SD]. URL: <https://arxiv.org/abs/2104.03617>.
- [65] H. Ma, J. Yi, C. Wang et al., *CFAD: A Chinese Dataset for Fake Audio Detection*, 2023. arXiv: 2207.12308 [cs.SD]. URL: <https://arxiv.org/abs/2207.12308>.
- [66] N. M. Müller, P. Kawa, W. H. Choong et al., „MLAAD: The Multi-Language Audio Anti-Spoofing Dataset“, *International Joint Conference on Neural Networks (IJCNN)*, 2024.
- [67] „ASV Spoof Challenge“. (), URL: <https://www.asvspoof.org> (cit. 25.12.2024).
- [68] „Multimedia Centre - European parliament“. (), URL: <https://multimedia.europarl.europa.eu> (cit. 01.04.2025).
- [69] R. Ardila, M. Branson, K. Davis et al., *Common Voice: A Massively-Multilingual Speech Corpus*, 2020. arXiv: 1912.06670 [cs.CL]. URL: <https://arxiv.org/abs/1912.06670>.

- [70] „whisper-large-v3-czech-cv13“. (), URL: <https://huggingface.co/mikr/whisper-large-v3-czech-cv13> (cit. 29.04.2025).
- [71] „CzeGPT-2“. (), URL: <https://huggingface.co/MU-NLPC/CzeGPT-2> (cit. 15.05.2025).
- [72] W. Ge, J. Patino, M. Todisco a N. Evans, *Raw Differentiable Architecture Search for Speech Deepfake and Spoofing Detection*, 2021. arXiv: 2107.12212 [eess.AS]. URL: <https://arxiv.org/abs/2107.12212>.
- [73] J. Yang a R. K. Das, „Long-term high frequency features for synthetic speech detection“, *Digital Signal Processing*, roč. 97, s. 102622, 2020, ISSN: 1051-2004. DOI: <https://doi.org/10.1016/j.dsp.2019.102622>. URL: <https://www.sciencedirect.com/science/article/pii/S1051200419301769>.
- [74] C. Sun, S. Jia, S. Hou a S. Lyu, „AI-Synthesized Voice Detection Using Neural Vocoder Artifacts“, čvn. 2023, s. 904–912. DOI: [10.1109/CVPRW59228.2023.00097](https://doi.org/10.1109/CVPRW59228.2023.00097).
- [75] „3.8. The cepstrum, mel-cepstrum and mel-frequency cepstral coefficients (MFCCs)“. (), URL: <https://speechprocessingbook.aalto.fi/Representations/Melcepstrum.html> (cit. 10.04.2025).
- [76] „ASVspoof 2021 Baseline CM and Evaluation Package“. (), URL: <https://github.com/asvspoof-challenge/2021/tree/main/LA> (cit. 09.04.2025).
- [77] „Metacentrum NGI“. (), URL: <https://www.metacentrum.cz/> (cit. 30.12.2024).
- [78] „Python - modules“. (), URL: https://wiki.metacentrum.cz/wiki/Python_-_modules (cit. 05.01.2025).
- [79] „MIT Acoustical Reverberation Scene Statistics Survey“. (), URL: https://mcdermottlab.mit.edu/Reverb/IR_Survey.html (cit. 29.04.2025).
- [80] „Everything You Must Know About Data Normalization in Machine Learning“. (), URL: <https://www.markovml.com/blog/normalization-in-machine-learning> (cit. 14.04.2025).
- [81] S. Vinay, „STANDARDIZATION IN MACHINE LEARNING“, břez. 2021. URL: https://www.researchgate.net/publication/349869617_STANDARDIZATION_IN_MACHINE_LEARNING.