VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE AKUSTICKÉHO PROSTŘEDÍ Z AUDIO NAHRÁVEK

ACOUSTIC SCENE CLASSIFICATION FROM SPEECH

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **FILIP GREPL**
AUTHOR

VEDOUCÍ PRÁCE **ing. PAVEL MATĚJKA, Ph.D.**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/20563/2017/xgrepl05

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Grepl Filip**
Obor: Informační technologie
Téma: **Detekce Akustické Prostředí z Řeči**
Acoustic Scene Classification from Speech
Kategorie: Zpracování řeči a přirozeného jazyka

Pokyny:
Cílem práce je klasifikace audio nahrávky do předem definovaných tříd, které charakterizují prostředí, ve kterém byla nahrávka pořízena - například - kancelář, ulice, park, kavárna ....

1. Prostudujte statistické techniky pro modelování řeči - neuronové sítě.
2. Prostudujte doporučenou literaturu.
3. Seznamte se s metody používanými pro detekci prostředí z audia
4. Seznamte se s daty a základním systémem pro soutěž DCASE 2016
5. Zjistěte úspěšnost základního systému na datech z této soutěže
6. Navrhněte a otestujte alespoň jednu další techniku na zlepšení úspěšnosti.

Literatura:
- http://www.cs.tut.fi/sgn/arg/dcase2016/task-acoustic-scene-classification
- http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification

Pro udělení zápočtu za první semestr je požadováno:
- 1-5

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Matějka Pavel, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato práce se zabývá vytvořením systému, jehož úkolem je z audio signálu rozpoznat, na jakém místě byla vstupní nahrávka pořízena. Klasifikátor je založen na vícevrstvé hustě propojené neuronové síti. Topologie neuronové sítě vychází ze základního systému, poskytovaného k soutěži DCASE. Pro její trénování a evaluaci je využita datová sada rovněž z této soutěže. Experimenty jsou prováděny zejména s reprezentací vlastností jednotlivých audio nahrávek a formátem vstupních dat. Za tímto účelem jsou využity Mel-filter bank, blok Mel-filter bank a MFCC příznaky. Experimenty, provedené v této práci, přinesly oproti základnímu systému soutěže DCASE vyšší přesnost klasifikace o 6.5 %. Celková úspěšnost systému tak dosáhla hodnoty 67.5 %.

Abstract

This thesis deals with creating a system whose task is to recognize what type of location the recording was created at by analyzing the audio signal. The classifier is based on a multilayer, fully connected neural network. The topology of the neural network is based on the baseline system provided for the DCASE competition. A dataset from this competition is also used for training and evaluating the neural network. The experiments are performed in particular with the representation of the properties of the audio records and with the format of the input data of the neural network. For this purpose, Mel-filter bank, block Mel-filter bank and MFCC flags are used. The experiments performed in this thesis brought a classification accuracy increased by 6.5 % compared to the baseline system of DCASE. Overall system success rate reached 67.5 %.

Klíčová slova

vícevrstvá hustě propojená neuronová síť, akustická klasifikace scény, Mel-filter bank, Mel-frequency cepstral coefficients, delta koeficienty, soutěž DCASE, Voting

Keywords

multilayer fully connected neural network, acoustic scene classification, Mel-filter bank, Mel-frequency cepstral coefficients, delta coefficients, competition DCASE, Voting

Citace

GREPL, Filip. *Detekce akustického prostředí z audio nahrávek*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce ing. Pavel Matějka, Ph.D.

Detekce akustického prostředí z audio nahrávek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana ing. Pavla Matějky, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Filip Grepl
10. května 2018

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu bakalářské práce panu ing. Pavlovi Matějkovi, Ph. D. za odborné vedení, pomoc a rady při zpracování této bakalářské práce.

Obsah

1 Úvod **5**

2 DCASE soutěže a přehled literatury **7**
2.1 Soutěž DCASE 7
2.2 Přehled literatury 9

3 Extrakce příznaků z audio signálu **14**
3.1 Mel-filter bank 14
3.2 Blok mel-filter bank 17
3.3 Mel-frequency cepstral coefficients 18
3.4 Delta a double delta koeficienty 19

4 Neuronové sítě **20**
4.1 Neuron - základní prvek neuronové sítě 20
4.2 Architektura vícevrstvé hustě propojené neuronové sítě 21
4.3 Trénování neuronové sítě 22

5 Datová sada pro trénování a evaluaci neuronové sítě **24**

6 Popis systému **25**
6.1 Základní systém poskytnutý k soutěži DCASE 25
6.2 Popis vytvořené neuronové sítě 26

7 Experimenty **29**
7.1 Levý a pravý kanál stereo signálu 29
7.2 Rozšíření trénovací sady 30
7.3 Změna parametru learning rate 30
7.3.1 Pevná hodnota learning rate 30
7.3.2 Skoková změna learning rate v závislosti na počtu iterací trénování . 31
7.3.3 Plynulá změna learning rate v závislosti na počtu iterací trénování . 33
7.4 Normalizace vstupních dat 34
7.5 Vzorkovací frekvence vstupního audio signálu 36
7.6 Využití Blok mel-filter bank koeficientů 37
7.6.1 Velikost kontextu 37
7.6.2 Počet DCT koeficientů 38
7.7 Počet trojúhelníkových filtrů 39
7.8 Využití MFCC, delta a double delta koeficientů 40
7.9 Trénování na více rámcích zároveň 42

7.10 Délka rámce 43
7.11 Fúze systémů 44
7.12 Experimenty s nejlepším systémem 45

8 Závěr **48**

Literatura **50**

A Spuštění programu **53**

B Popis programu **55**

Seznam obrázků

2.1 *Kategorie v soutěži DCASE* . . . 9
2.2 *Dosažené výsledky v soutěži DCASE – 1. úkol* . . . 10

3.1 *Blokové schéma popisující získání MFB koeficientů ze vstupního audio signálu pro jednu nahrávku* . . . 15
3.2 *Závislost stupnice mel na stupnici hertz* . . . 16
3.3 *Rozložení patnácti trojúhelníkových filtrů přes frekvenční spektrum pomocí mel stupnice* . . . 16
3.4 *Blokový diagram popisující získání blok MFB koeficientů s kontextem 5 rámců* 17
3.5 *Blokový diagram popisující získání výstupní matice při použití delta a double delta koeficientů s kontextem 1 rámec* . . . 18

4.1 *Aktivační funkce perceptronu a sigmoid neuronu* . . . 21
4.2 *Topologie vícevrstvé neuronové sítě se 4 skrytými vrstvami* . . . 22

6.1 *Blokové schéma navrženého systému* . . . 27

7.1 *Závislost úspěšnosti systému na fixním velikosti parametru learning rate* . . . 31
7.2 . . . 32
7.3 *Dosažená úspěšnost neuronové sítě pro různé inicializační hodnoty parametru learning rate při použití snižování pomocí Drop-Based Learning Rate Schedule s dropRate = 0.5 a epochDrop = 10* . . . 32
7.4 *Plynulá změna parametru learning rate v závislosti na počtu iterací trénování a na zvoleném parametru decay* . . . 33
7.5 *Dosažená úspěšnost neuronové sítě pro různé počáteční hodnoty learning rate při použití snižování pomocí Time-Based Learning Rate Schedule parmetrem decay = 0.001* . . . 34
7.6 *Dosažená úspěšnost pro různou velikost kontextu rámců při použití blok MFB koeficientů v závislosti na počtu iterací trénování neuronové sítě* . . . 38
7.7 *Dosažená úspěšnost systému v závislosti na počtu trojúhelníkových filtrů při použití blok MFB koeficientů* . . . 40
7.8 *Dosažená úspěšnost systému v závislosti na počtu MFCC koeficientů* . . . 41
7.9 *Dosažená úspěšnost systému v závislosti na počtu rámců v jednom segmentu* 42
7.10 *Dosažená úspěšnost systému v závislosti na délce jednoho rámce* . . . 43

Seznam tabulek

2.1 *Tabulka klasifikačních tříd* 8
2.2 *Srovnání úspěšnosti deseti nejlepších systémů na trénovací a evaluační datové sadě* 10
2.3 *Porovnání úspěšnosti vítězného systému při trénování na základní a rozšířené trénovací datové sadě* 11

5.1 *Odlišnosti datových sad z roku 2016 a 2017* 24

6.1 *Topologie vícevrstvé neuronové sítě* 26

7.1 *Dosažená celková úspěšnost neuronové sítě při trénování a evaluaci na jednotlivých kanálech a průměrné hodnotě obou kanálů* 30
7.2 *Dosažená celková úspěšnost neuronové sítě při trénování na kombinacích levého, pravého a průměrné hodnoty kanálů* 30
7.3 *Dosažená úspěšnost neuronové sítě v závislosti na velikosti parametru batch size a použité normalizaci* 36
7.4 *Dosažená úspěšnost neuronové sítě v závislosti na použité vzorkovací frekvenci* 36
7.5 *Maximální dosažená úspěšnost neuronové sítě v závislosti na velikosti kontextu MFB koeficientů* 37
7.6 *Maximální dosažená úspěšnost neuronové sítě v závislosti na počtu DCT koeficientů* 39
7.7 *Dosažená úspěšnost neuronové sítě v závislosti na velikosti kontextu delta a double delta koeficientů* 42
7.8 *Porovnání třídních úspěšností a celkové úspěšnosti nejlepšího systému s blok MFB a MFCC příznaky s fúzí těchto systémů* 44
7.9 *Dosažená úspěšnost neuronové sítě při trénování a evaluaci na jednotlivých kanálech a průměrné hodnotě obou kanálů v závislosti na použité datové sadě* 45
7.10 *Porovnání třídních úspěšností a celkové úspěšnosti nejlepšího systému v této práci a základních systémů soutěže DCASE na odpovídající evaluační sadě* . 46
7.11 *Porovnání parametrů základního systému soutěže DCASE 2017 a nejlepšího systému v této práci* 47

Kapitola 1

Úvod

Zvuky, které se objevují kolem nás, jsou tvořeny z mnoha různých zdrojů. Tyto zvuky obsahují informace, které člověk z okolního prostředí získává pomocí sluchového vjemu. Na těchto informacích je založeno naše chování a naše myšlenky. Díky nim dokážeme rovněž rozeznat, na jakém místě, případně v jakém prostředí se nacházíme a to dokonce i v případě, že nám nejsou poskytovány vizuální informace. Je tomu tak proto, že naše mysl je dobře vycvičena na zkušenostech – informacích z minulosti. Je nám tedy naprosto jasné, že šplouchání vody pravděpodobně charakterizuje pláž nebo že specifické pípnutí při přečtení čárového kódu charakterizuje obchod s potravinami. Naše smysly nám také umožňují ze zvuku rozluštit velikost prostoru na základě šíření ozvěny. Velké prostory, jako např. stanice metra nebo les rozpoznáme podle toho, že se zde ozvěna šíří velmi dobře, zatímco v kanceláři, která je zaplněná nábytkem, se nešíří ozvěna téměř vůbec. Náš mozek se však toto velké množství informací neučí několik hodin nebo dní, učí se je během celého života. Vytvořit tak automatizovaný systém, který dokáže tyto informace rozpoznat, je velmi obtížné zejména proto, že tento systém nemá k dispozici tolik trénovacích dat, jako náš mozek.

Cílem této práce je tedy vytvořit klasifikátor, který dokáže z audio nahrávky určit, na jakém místě byla tato nahrávka pořízena za předpokladu, že byla pořízena na jednom z předem definovaných míst. Tento klasifikátor může být použitelný pro inteligentní systémy, zejména pro roboty a automobily s autonomním řízením. Může být užitečný rovněž pro policejní složky při hledání pachatele nebo odposlouchávání telefonních hovorů.

Ve své práci se zaměřuji na vytvoření tohoto klasifikátoru pomocí neuronové sítě. Pro správnou funkci neuronové sítě je nezbytné ji před jejím použitím natrénovat. Je tedy nutné získat data pro trénování. Stejně tak je pro zjištění úspěšnosti neuronové sítě klíčové mít data pro evaluaci, tj. data, která se neshodují s daty pro trénování. Protože byla v roce 2016 i v roce 2017 vypsána soutěž na obdobné téma, jakým se zabývám i já ve své práci, a byla v této soutěži k dispozici dostatečně velká datová sada vyhovující mým požadavkům, rozhodl jsem se tyto datové sady z obou let, kdy soutěž probíhala, využít.

Nejprve je uveden krátký přehled o tom, co je to soutěž DCASE 2017 a v jakých kategoriích se soutěží. Dále se zde nachází stručný popis řešení pěti nejlepších týmu, které se této soutěže v minulém roce zúčastnili. Ve 3. kapitole jsou detailně popsány způsoby, jak se získávají vlastnosti, charakterizující jednotlivé audio nahrávky. Je jim věnována zvláštní pozornost, jelikož pochopení této problematiky bylo pro tuto práci nezbytné. V následující kapitole je stručně popsáno, z čeho se skládá neuronová síť, jak vypadá architektura vícevrstvé neuronové sítě a průběh trénování neuronové sítě. Po té jsou popsány parametry obou datových sad, které byly v této práci použity. 6. kapitola je věnována popisu základního

systému soutěže DCASE a systému, vytvořenému v této práci. Dále jsou popsány všechny experimenty, které byly s vytvořeným klasifikátorem provedeny. V každé podkapitole je popsáno, s jakým parametrem nebo zpracováním audio signálu bylo experimentováno. Na závěr této části je nejlepší systém porovnán se základními systémy soutěže DCASE. Poslední kapitola obsahuje zhodnocení celé práce, dosažené výsledky a plán budoucí práce.

Kapitola 2

DCASE soutěže a přehled literatury

V této kapitole je popsáno, co je to soutěž DCASE a na jakou oblast se zaměřuje. Jsou zde uvedeny jednotlivé kategorie, ve kterých se soutěžilo a popsána řešení pěti nejlepších klasifikátorů v první kategorii soutěže, která je předmětem této bakalářské práce.

2.1 Soutěž DCASE

Detection and Classification of Acoustic Scenes and Events, zkráceně DCASE, je soutěž, která je organizována audio výzkumnou skupinou z Tampere University of Technology ve Finsku, univerzitou Carnegie Mellon v USA a univerzitou Invie ve Francii. Nekoná se na jednom konkrétním místě, ale každý účastník své řešení odevzdává elektronicky. Po vyhodnocení všech odevzdaných řešení jsou výsledky vyvěšeny na stránkách výzvy a po té jsou diskutovány na workshopu, který je organizován pořadateli [24].

Ke každému úkolu je k dispozici trénovací a evaluační datová sada. Trénovací sada je k dispozici soutěžícím pro vývoj jejich klasifikátorů, zatímco evaluační sada umožňuje pořadatelům zjistit spolehlivost a úspěšnost systémů, které soutěžní týmy odevzdaly. Jednou z podmínek soutěže je, že pro jejich vývoj nemohou být použita žádná jiná data kromě dodané trénovací sady. Účastníci mají přístup i k evaluační sadě, nemají k ní však soubor s místy, odkud jednotlivé nahrávky pocházejí. Soutěží se ve 4. kategoriích:

- **Akustická klasifikace scény**

  Cílem tohoto úkolu je vytvořit systém, který dokáže vstupní audio nahrávku zařadit do jedné z patnácti definovaných tříd, jak je znázorněno na obr. 2.1a[1]. Seznam jednotlivých tříd je možné nalézt v tab. 2.1. Každá třída charakterizuje prostředí, ve kterém byla nahrávka pořízena [10]. Z této kategorie vychází má bakalářská práce, datová sada k tomuto úkolu je tedy popsána samostatně v kapitole 5.

[1] zdroj: http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification

Tabulka 2.1: *Tabulka klasifikačních tříd*

Místo	Místo
autobus	knihovna
kavárna/restaurace	stanice metra
auto	kancelář
centrum města	obytná oblast
lesní cesta	vlak
obchod s potravinami	tramvaj
domov	park
pláž	

- **Detekce neobvyklé zvukové události**

  V této kategorii je cílem vytvořit systém, který dokáže z audio záznamu detekovat neobvyklé události. Konkrétně se jedná o 3 kategorie, viz obr. 2.1b[2]. Datová sada je tvořena z izolovaných zvukových záznamů pro každou třídu a nahrávek z běžného reálného prostředí, které slouží jako pozadí. Výsledné audio záznamy jsou vytvářeny mixováním těchto 2 různých kategorií. Je tak dosaženo více trénovacích podmínek, než by poskytly reálné nahrávky [2].

- **Detekce zvukové události z reálného života**

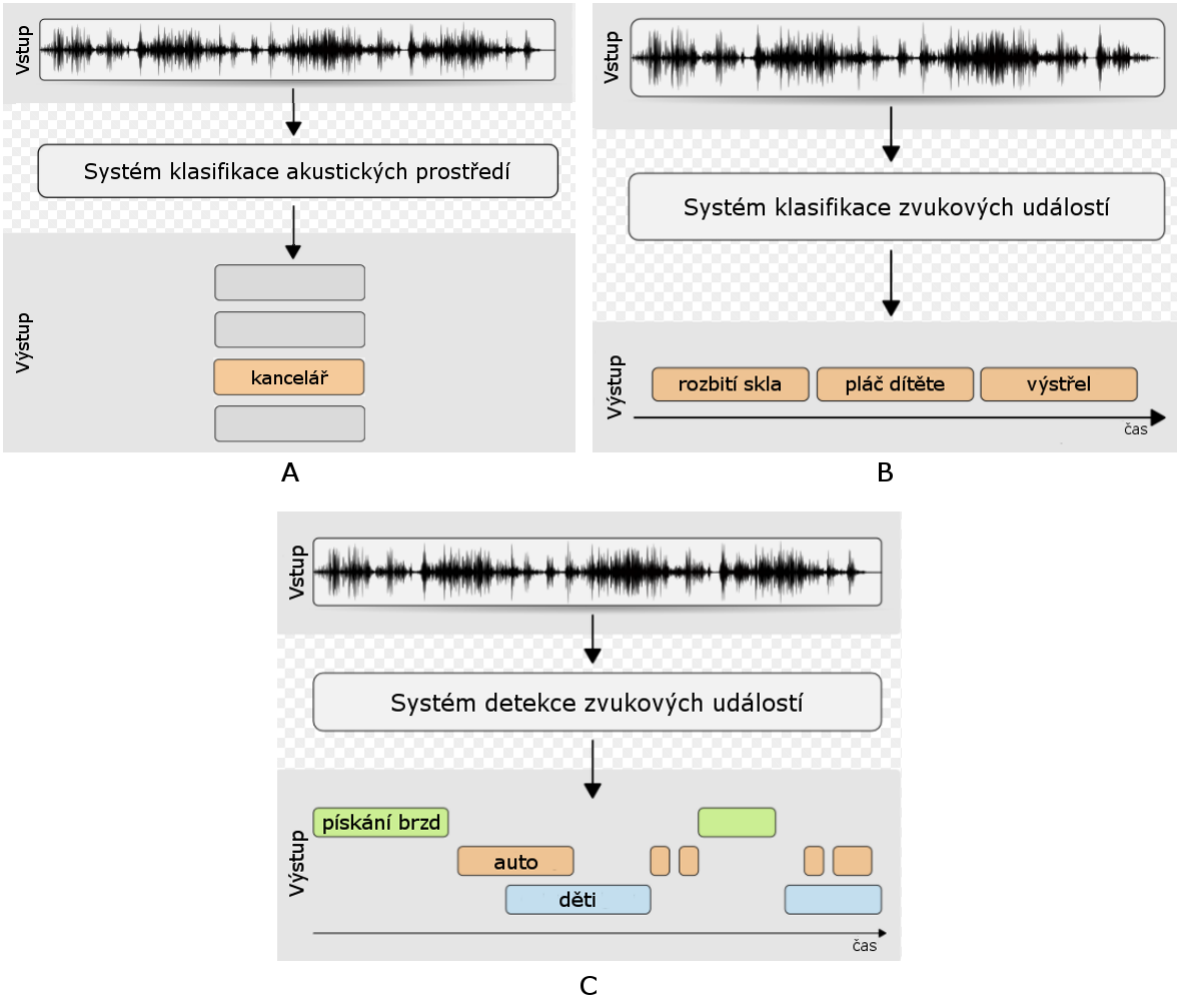
  V běžném životě se izolované zvuky příliš nevyskytují. Mnohem častěji se setkáme se zvuky, které jsou navzájem smíchané a překrývají se. Třetí úkol, jak už název napovídá, se zabývá právě těmito situacemi a klade si za cíl vytvořit systém, který z reálných nahrávek dokáže určit, jaká událost nebo souběh několika událostí v konkrétním čase probíhaly. Vše názorně ukazuje obr. 2.1c[3]. Audio nahrávky v trénovací i evaluační datové sadě byly pořizovány na ulici, protože právě toto prostředí obsahuje mnoho překrývajících se zvuků, což je pro tento úkol nezbytné. Detekovány jsou však jen tyto kategorie: pískání brzd, auto, děti, nákladní auto, řeč lidí a chůze lidí [11].

- **Velkokapacitní detekce zvukových událostí s nízkou kontrolou pro inteligentní automobily**

  Poslední kategorií v soutěži DCASE je detekce zvukových událostí z automobilového prostředí. Výsledky tohoto úkolu pomohou při vývoji nových způsobů detekce akustických událostí pro autonomní vozidla. Zajímavostí je datová sada, která je tvořena výňatky z videí uložených na internetovém serveru *YouTube*, ve kterých se objevují zvuky z automobilového prostředí ze 2 kategorií, které jsou rozpoznávány. První kategorií jsou varovné zvuky, jako např.: autoalarm, siréna sanitky, záchranné služby nebo křik. Druhou kategorií jsou zvuky vozidel, např.: autobus, motocykl, nákladní automobil, vlak atd. [3].

[2] zdroj: http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-rare-sound-event-detection

[3] zdroj: http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-sound-event-detection-in-real-life-audio



Obrázek 2.1: *Kategorie v soutěži DCASE*

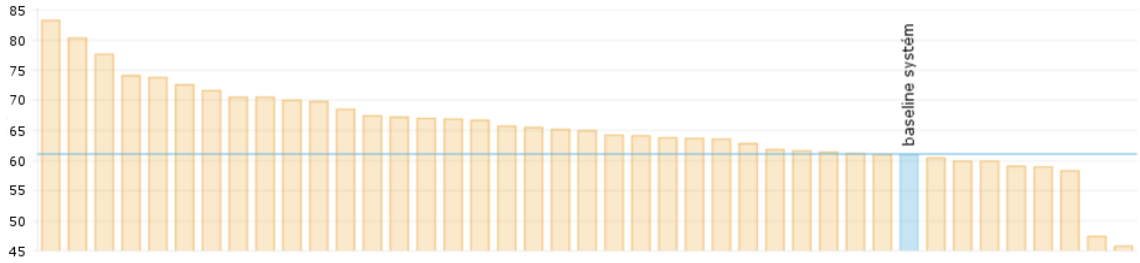
2.2 Přehled literatury

Jak je uvedeno v kapitole 2, tato bakalářská práce vychází ze soutěže DCASE, přesněji z její první kategorie. Na obr. 2.2[4] jsou znázorněny úspěšnosti všech 41 týmů, které se této soutěže zúčastnili. Jelikož mohl každý tým odevzdat až čtyři různá řešení, bylo celkem odevzdáno 97 systémů. Uváděné úspěšnosti v této kapitole jsou vztaženy k trénovací datové sadě, pokud není uvedeno jinak.

Úspěšnost odevzdaných systémů se pohybovala v rozmezí od 46 do 83 % na evaluační datové sadě. V tab. 2.2 je vidět srovnání úspěšností deseti nejlepších týmů. Z výsledků uvedených v této tabulce vyplývá, že dosáhly velmi rozdílné úspěšnosti na evaluační a trénovací datové sadě. Většina systémů, které dosáhly velmi vysoké úspěšnosti na trénovací sadě, tj. přes 90 %, dosáhla na evaluační datové sadě o 10–20 % horších výsledků, protože byly přetrénované a negeneralizovaly se na obecná data. Základní systém, který byl k danému úkolu poskytnut, dosáhl úspěšnosti na stejné evaluační sadě 61 %. Vítězem se stal tým

[4] zdroj: http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification-results

GAN_SKMUN, jejichž klasifikátor dosáhl oproti základnímu systému zlepšení úspěšnosti o 22% na celkových 83.1 %.



Obrázek 2.2: *Dosažené výsledky v soutěži DCASE – 1. úkol*

Tabulka 2.2: *Srovnání úspěšnosti deseti nejlepších systémů na trénovací a evaluační datové sadě*

Pořadí	Označení týmu	Název týmu	Úspěšnost na evaluační sadě	Úspěšnost na trénovací sadě
1	Mun_KU_task1_1	GAN_SKMUN	83.3 %	87.1 %
2	Han_COCAI_task1_3	FlEnsemSel	80.4 %	91.9 %
3	Xing_SCNU_task1_2	DCNN_SVM	77.7 %	89.9 %
4	Hasan_BUET_task1_1	BUETBOSCH1	74.1 %	88.1 %
5	Lehner_JKU_task1_4	JKU_All_ca	73.8 %	91.3 %
6	Park_ISPL_task1_1	ISPL	72.6 %	83.6 %
7	Kukanov_UEF_task1_1	K-CRNN	71.7 %	85.8 %
8	Yu_UOS_task1_4	UOS_res	70.6 %	95.8 %
8	Piczak_WUT_task1_1	amb200	70.6 %	82.3 %
9	Zhao_ADSC_task1_1	MResNet-34	70.0 %	85.6 %
10	Bisot_TPT_task1_1	TPT1	69.8 %	90.1 %

Způsoby řešení jednotlivých týmů:

- *Seongkyu Mun, Sangwook Park, David K. Han, Hanseok Ko* z týmu *GAN_SKMUN* se v roce 2017, jak je zřejmé z tab. 2.2, stali vítězi této soutěže [13]. Protože je trénovací datová sada, poskytnutá k soutěžnímu úkolu, relativně malá, rozhodli se rozšířit tuto sadu o nové vzorky, což bylo největším přínosem pro zvýšení úspěšnosti základního systému. Použili pro to GAN (Generative Adversarial Network) neuronovou síť, která se skládá ze dvou podsítí, generátoru a diskriminátoru. Generátor má za úkol generovat z dostupných dat nová data a diskriminátor ověřovat, jestli se jedná o data generovaná nebo reálná. Použita jsou pouze ta vygenerovaná data, která jsou diskriminátorem označena jako reálná. V tab. 2.3 jsou vidět výsledky jejich experimentů. S původní datovou sadou dosáhli nejlepší úspěšnosti 79.3 % a to s klasifikátorem SVM (Support Vector Machines), stejně jako pro rozšířenou trénovací datovou sadu, kdy však byla dosažená úspěšnost 85.6 %. V obou případech byly pro získání příznaků reprezentujících danou audio nahrávku použity MFCC (Mel-frequency cepstral coefficients). Z této tabulky dále vyplývá, že u všech navržených systémů přinesla rozšířená

datová sada výrazné zvýšení úspěšnosti, v průměru o 6.5 %. Jako výsledný systém použili fúzi všech 4 navržených systémů, tedy FCNN (Fully Connected Neural Network) a SVM klasifikátor s MFCC koeficienty a FCNN a SVM klasifikátor s DFT koeficienty. Výsledný systém tedy na základní trénovací sadě dosáhl úspěšnosti 81.5 %. Při použití rozšířené trénovací sady byla dosažená úspěšnost 87.1 %. Na evaluační datové sadě tento výsledný systém dosáhl úspěšnosti 83.3 % a umístil se tak na 1. místě.

Tabulka 2.3: *Porovnání úspěšnosti vítězného systému při trénování na základní a rozšířené trénovací datové sadě*

Průměrná úspěšnost [%]	Originální datová sada				Rozšířená datová sada			
	DFT-FCNN	MFB-FCNN	DFT-SVM	MFB-SVM	DFT-FCNN	MFB-FCNN	DFT-SVM	MFB-SVM
	75.4	75.1	78.2	79.3	83.2	83.7	81.6	85.6

- *Yoonchang Han, Jeongsoo Park, Kyogu Lee* z týmu *FlEnsemSel*, kteří skončili celkově na 2. místě, se rozhodli pro své řešení použít CNN (Convolutional Neural Network) [6], přesněji fúzi čtyř konvolučních neuronových sítí. Zaměřili se zejména na zvýraznění akustických odlišností jednotlivých nahrávek, použili pro to 3 způsoby:

  - **Binaural representations**

    Tento princip využívá toho, že v reálném světě přichází do každého lidského ucha odlišný zvuk, tedy že i každý kanál stereo nahrávky je mírně odlišný. Převodem stereo do mono signálu tyto odlišnosti zanikají [27]. Proto se rozhodli vstupní signál využít obdobně, jako tým popsaný v předchozím bodě. První neuronová síť byla trénována na obou kanálech s nezměněnou vzorkovací frekvencí, zatímco druhá na průměrné hodnotě obou kanálů a rozdílu levého a pravého kanálu.

  - **Harmonic-percussive source separation**

    Harmonic-percussive source separation, zkráceně HPSS, je přístup, který říká, že každý zvuk patří do jedné ze 2 kategorií – buď harmonických nebo nárazových zvuků. Harmonické zvuky jsou viditelné v časové oblasti, protože trvají delší časový úsek. Příkladem takového zvuku může být například zvuk projíždějícího auta. Protikladem jsou nárazové zvuky, které trvají velmi krátký časový úsek a jsou nejlépe viditelné ve frekvenční oblasti [14]. Příkladem nárazového zvuku pak může být pád sklenice ze stolu na zem. Třetí neuronová síť tak využívá jako vstup původní mono signál zpracovaný HPSS algoritmem pro detekci nárazových a harmonických zvuků.

  - **Background substraction**

    Poslední metoda, která byla pro zvýraznění odlišností jednotlivých nahrávek použita, je *background substraction*, která se snaží o odstranění šumu z audia. V tomto případě bylo pro odstranění šumu použito tzv. filtrování mediánu, které spočívá v tom, že je od původního audio signálu odečtena hodnota, která odpovídá velikosti mediánu určitého počtu vzorků. Poslední, čtvrtá, neuronová síť tedy využívá jako vstup mono audio nahrávky po odstranění šumu.

  Všechny výše popsané způsoby využívají MFCC příznaky. Při použití mono audio signálu dosáhl jejich systém úspěšnosti 84,4 %. V případě aplikace metody *Binaural*

representations se přesnost klasifikace zvýšila o 2.7 % pro levý a pravý kanál audia a o 3.5% pro rozdíl a průměrnou hodnotu obou kanálů. Pro způsob HPSS se úspěšnost zvýšila o 2.5 %. Při odstranění šumu metodou *back substraction* se přesnost klasifikace pohybovala v rozmezí od 80.1 do 84.3 % v závislosti na velikosti použitého filtru. Výsledný klasifikátor tak dosáhl průměrné úspěšnosti na trénovací sadě 91.9%. V soutěži se však umístil na druhém místě, jelikož rozpoznával místa, kde byly nahrávky pořizovány, s přesností 80.4 %.

- *Zheng Weiping1, Yi Jiantao1, Xing Xiaotao1, Liu Xiangtao2, Peng Shaohu* z týmu *DCNN_SVM* se rozhodli, stejně jako předchozí tým, pro svůj systém použít konvoluční neuronovou síť [26]. Pro vytvoření spektogramů jednotlivých nahrávek nepoužili MFCC tak, jako předchozí 2 týmy, ale využili pro to jak běžně používanou STFT (Short Time Fourier Transform), tak CQT (Constant Q Transform).

  CQT je vhodná zejména pro hudební audio signály, jelikož vytváření frekvenčního spektra z audia se velmi podobá vnímání zvuků člověkem. Lidské ucho totiž u nízkých frekvencí rozpozná i velmi malé odlišnosti, zatímco u vysokých frekvencí tomu tak není. V praxi to tedy znamená, že člověk od sebe dokáže rozeznat frekvenci 50 Hz a 60 Hz, ale už od sebe nedokáže rozlišit 15 000 Hz a 15 010 Hz. Z toho vychází CQT transformace. Nízké frekvence proto převádí s vyšším spektrálním rozlišením, zatímco vyšší frekvence s nižším spektrálním a vyšším časovým rozlišením. Ve srovnání s STFT je CQT mnohem časově náročnější. STFT má však pro stejné spektrální rozlišení, jako má CQT na nízkých frekvencích, daleko větší paměťové nároky.

  Výše popsanou CQT transformaci použili na každý kanál audio nahrávky zvlášť. Protože pro každou nahrávku bylo vygenerováno několik spektogramů, měla neuronová síť několik různých výstupů, které bylo nutné vyhodnotit a určit tak konečné rozhodnutí, o jaké místo se jedná. Použili k tomu 2 způsoby:

  – První způsob, který použili, se nazývá *Voting*. Jedná se nejčastěji používaný způsob – jako výsledná třída, do které je nahrávka zařazena, je vybrána ta, která je vyhodnocena pro jednotlivé spektogramy z dané nahrávky nejvícekrát.
  – Jako druhý způsob si vybrali SVM. Vstupem do SVM pak byl jediný vektor pro každou nahrávku, který sestával z výstupů neuronové sítě pro spektogramy této nahrávky, agregovaných způsobem popsaným v předchozím bodě.

  Při vytváření spektogramů pomocí STFT dosáhli nejlepších výsledků s velikostí okna 32 vzorků. Zpracování výstupů pomocí *Votingu* přineslo úspěšnost 84.5 %. Při použití SVM byla dosažená úspěšnost 85.4 %. SVM tedy v tomto případě přineslo zlepšení o 0.9 %. S CQT spektogramy bylo nejvyšší přesnosti klasifikace dosaženo zprůměrováním výstupů pomocí *Votingu*, kdy dosažená úspěšnost činila 80.5 %. Dodatečná SVM klasifikace však přinesla horší výsledek pouze o 0.2 %. Jako výsledný systém použili neuronovou síť natrénovanou jak na spektogramech STFT, tak na spektogramech CQT. Zprůměrováním výstupů dosáhli úspěšnosti 87.4 %, zatímco klasifikací pomocí SVM 89.9 %. Ve výsledném klasifikátoru tedy závěrečné vyhodnocování pomocí SVM zvýšilo úspěšnost o 2.5 %. Nahrávky v evaluační sadě byly úspěšně klasifikovány s přesností 77.7 %. Tým *DCNN_SVM* tak skončil celkově na 3. místě.

- *Rakib Hyder, Shabnam Ghaffarzadegan, Zhe Feng, Taufiq Hasan* z týmu *BUET-BOSCH1* použili ve svém řešení fúzi CNN a GMM (Gaussian Mixture Model), jejímž

výstupem byl tzv. SV (Super Vector) [8]. Protože byl tento vektor pro další zpracování příliš velký, rozhodli se pro snížení dimensionality použít techniku LDA (Linear Discriminant Analysis) [21]. Po té tyto super vektory normalizovali a použili je pro trénovaní PLDA (Probabilistic Linear Discriminant Analysis) systému.

Audio nahrávky v datové sadě nejprve převzorkovali z původních 44.1 kHz na poloviční frekvenci, tedy 22.05 kHz. Po té byla tato data použita pro výpočet standardních MFCC a MFB koeficientů. První CNN neuronová síť, kterou navrhli, byla natrénována na všech datech. Následně tuto neuronovou síť spojili s PLDA systémem tak, že výstupní vektor z CNN sítě byl použit jako vstup do PLDA. Po té jednotlivé spektogramy rozdělili na 3 části – nízké, střední a vysoké frekvenční pásmo a vytvořili 3 nezávislé CNN+PLDA klasifikátory, přičemž byl každý z nich trénován na jednom frekvenčním pásmu.

Základní PLDA systém dosáhl průměrné úspěšnosti 81.08 % a samotná CNN síť 80.85 % pro MFB a 81.00 % pro MFCC koeficienty. Jejich spojením se úspěšnost zvýšila na 83.13 % s MFCC. S MFB koeficienty byla přesnost klasifikace vyšší o pouhých 0.17 %. Nejvyšší úspěšnosti, 84.5 %, dosáhli pro 3 nezávislé CNN+PLDA klasifikátory s MFB koeficienty, které jsou popsané v předchozím odstavci. V případě MFCC koeficientů byla průměrná úspěšnost o 0.28 % nižší. Z výše uvedeného tedy vyplývá, že použití MFB koeficientů vedlo ve všech případech k nepatrně lepším výsledkům, nicméně na výslednou úspěšnost mělo použití MFB a MFCC koeficientů pouze nepatrný vliv. Jako výsledné řešení použili 4 různé fúze výše popsaných systémů. Nejlepší z nich dosáhla úspěšnosti 74.1 % na evaluační datové sadě a skončili tak celkově na 4. místě.

- *Bernhard Lehner, Hamid Eghbal-zadeh, Matthias Dorfer, Filip Korzeniowski, Khaled Koutini, Gerhard Widmer* z týmu *JKU_All_ca* si pro své řešení vybrali také CNN [7]. Rozhodli se, rovněž jako vítězný tým, zvětšit trénovací datovou sadu a to tak, že každou nahrávku posunuli nahoru a dolů o 25, 50 a 100 centů [5]. Tím se jejich datová sada zvětšila 6x. První klasifikátor, který vytvořili, byl založen na I-vektorech a trénován na MFCC koeficientech. Další 2 systémy, se kterými experimentovali, byly založené na CNN sítích. Nejprve zkoušeli vytvořit síť, která se učila na spektogramech pro celé nahrávky, po té CNN, která na vstup dostávala dvousekundové výňatky z audio nahrávek. Experimentováním a změnami parametrů neuronové sítě jako např. learning rate, aktivační funkce, batch size atd. si vybrali 5 CNN modelů s nejlepší průměrnou úspěšností. Výstupy zpracovávali pomocí LLR (Linear Logistic Regression) a to jak pro I-vektory, tak pro výstupy jednotlivých CNN. Každému výstupu z jednotlivých systémů tedy přiřazovali různé váhy tak, aby dosáhli co nejvyšší průměrné úspěšnosti. Nakonec provedli fúzi I-vektorů a pěti nejlepších CNN sítí, kde zkoušeli pro zpracování výstupů použít kromě LLR také techniku zvanou *Voting*, jejíž princip je přiblížen v předchozím odstavci u popisu systému vytvořeného týmem *DCNN_SVM*.

  Při použití LLR pro I-vektory dosáhli průměrné úspěšnosti 84.45 %, pro CNN sítě pak 89.03 %. V případě zpracování výstupů pomocí *Votingu* byla průměrná přesnost fúze obou systémů 87.70 %. Nejvyšší přesnosti klasifikace, 91.29 %, bylo dosaženo pro fúzi obou systémů s následnou klasifikací pomocí LLR. Ta dosáhla i nejvyšší úspěšnosti na evaluační sadě, tedy 73.8 %, a celkově tak tým *JKU_All_ca* skončil na 5. místě.

[5] Cent je bezrozměrná jednotka pro měření velikosti intervalů, používaná v hudební akustice i hudbě. Definice centu vychází z rovnoměrně temperovaného ladění, dělícího oktávu na 12 stejně velkých půltónů. Jeden cent je definován jako 1/100 temperovaného půltónu, tedy 1/1200 oktávy [28].

Kapitola 3

Extrakce příznaků z audio signálu

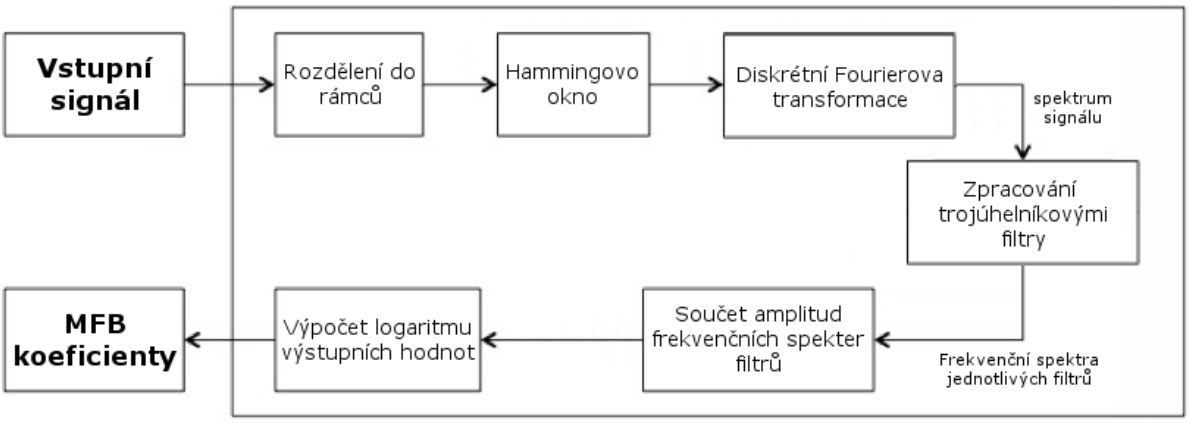
V této kapitole jsou detailně popsány způsoby získávání příznaků, tedy specifických vlastností pro danou nahrávku, z jednotlivých vzorků audio signálu. Prvním z nich jsou velmi často používané MFB koeficienty. Dále pak MFCC koeficienty, které přímo vycházejí z MFB, avšak přinášejí o daném signálu více informací. Aby bylo možné zachytit změny signálu ve větším časovém úseku, tedy mezi více rámci, zabývá se tato práce i tzv. delta koeficienty pro MFCC a blokovým zpracováním MFB.

3.1 Mel-filter bank

Z informací uvedených v kapitole 2 vyplývá, že téměř všichni soutěžící použili pro získání vlastností jednotlivých nahrávek MFB, případně MFCC koeficienty. Hlavní přínos a smysl MFB je ten, že se snaží reprezentovat jednotlivé frekvence, zastoupené v každé audio nahrávce tak, jak je reálně vnímá člověk [9].

Každá audio nahrávka je reprezentována určitým počtem vzorků, který je závislý jednak na délce nahrávky a jednak na velikosti vzorkovací frekvence. Postup získání MFB koeficientů pro jednu audio nahrávku znázorňuje obr. 3.1. Je zřejmé, že tento postup se skládá z několika kroků:

1. Rozdělení nahrávek do rámců
2. Získání frekvenčního spektra každého rámce
3. Získání frekvenčního spektra každého filtru
4. Součet jednotlivých amplitud a výpočet logaritmu každého výstupního frekvenčního spektra z jednotlivých filtrů

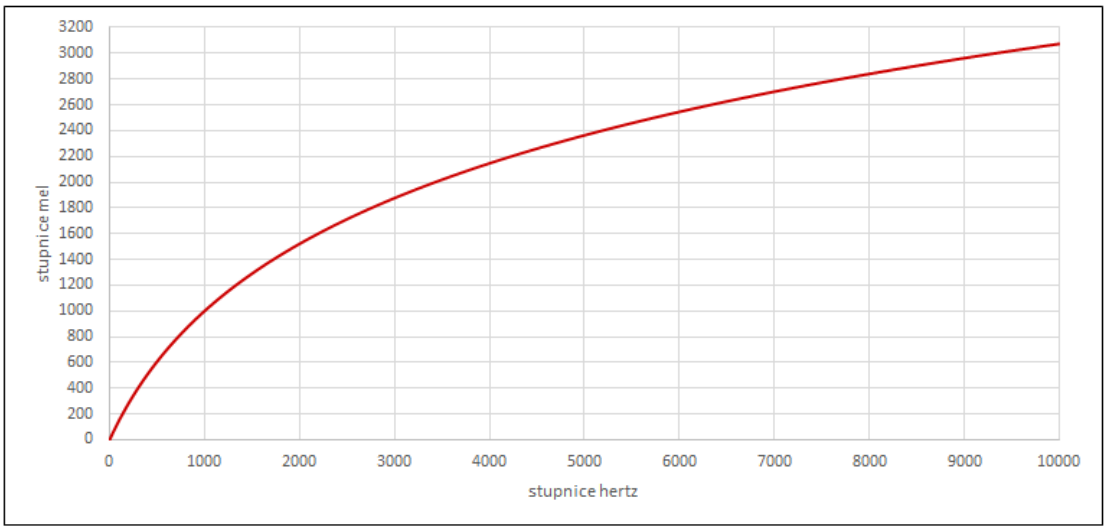


Obrázek 3.1: *Blokové schéma popisující získání MFB koeficientů ze vstupního audio signálu pro jednu nahrávku*

Protože se obecně audio signál v čase výrazně mění, je nutné v prvním kroku rozdělit signál do jednotlivých rámců tak, že se jednotlivé rámce navzájem překrývají běžně o 50–60 % své délky. Při vstupním audio signálu se vzorkovací frekvencí 8000 Hz se tedy používají rámce dlouhé 20–40 ms, tedy 160–320 vzorků [4].

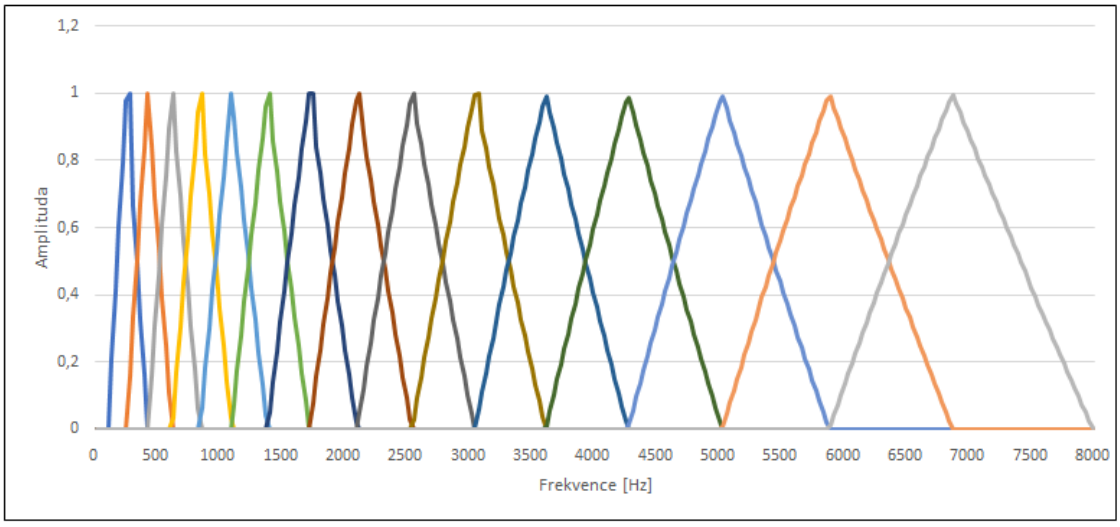
Druhým krokem je získání frekvenčního spektra každého rámce. Nejprve je na jednotlivé vzorky každého rámce aplikována tzv. *window funkce*, která do značné míry omezuje problém spektrálního úniku, protože není možné zajistit, že jeden rámec odpovídá právě celočíselnému násobku jedné periody vstupního signálu. Spektrální únik je situace, kdy frekvenční spektrum, získané pomocí DFT neodpovídá reálnému spektru signálu, ale jeho tzv. rozmazané podobě. To znamená, že jsou v něm obsaženy i frekvence, které skutečný signál neobsahuje [15]. Po té je vypočítána DFT (Discrete Fourier Transform), audio signál každého rámce je tedy převeden z časové oblasti do frekvenční. Tento krok je nutný proto, že v časové oblasti jsou jednotlivé frekvence promíchány a není tak zřejmé, jaké frekvence audio signál obsahuje.

Dalším krokem je zpracování frekvenčního spektra několika trojúhelníkovými filtry, běžně se jedná o 20–40 filtrů. Každý filtr má odezvu velikosti 1 na střední frekvenci a po té lineárně klesá jak směrem k vyšším, tak směrem k nižším frekvencím, až na odezvu 0, která odpovídá střední frekvenci sousedního filtru. Důležitou vlastností těchto filtrů je fakt, že jsou přes frekvenční spektrum rozloženy lineárně, ne však v hertzích, ale ve stupnici mel. Závislost mezi frekvenční stupnicí mel a frekvenční stupnicí hertz je uvedena na obr. 3.2.



Obrázek 3.2: *Závislost stupnice mel na stupnici hertz*

Nejprve je maximální a minimální frekvence ve frekvenčním spektru převedena na stupnici mel. Tento nově vzniklý rozsah v mel hodnotách je po té lineárně rozdělen v závislosti na požadovaném počtu filtrů, přičemž každá hodnota představuje počáteční bod následujícího filtru a zároveň vrchol předchozího filtru. Tyto hodnoty jsou po té opět převedeny na jednotky ve stupnici hertz. Výsledkem je, že na nižší frekvence z frekvenčního spektra je aplikováno více filtrů, zatímco na vyšší frekvence jich je aplikováno méně, což odpovídá tomu, že lidské ucho dokáže lépe rozlišit rozdíly na nízkých, než na vysokých frekvencích. Příklad rozložení patnácti filtrů pomocí stupnice mel znázorňuje obr. 3.3.

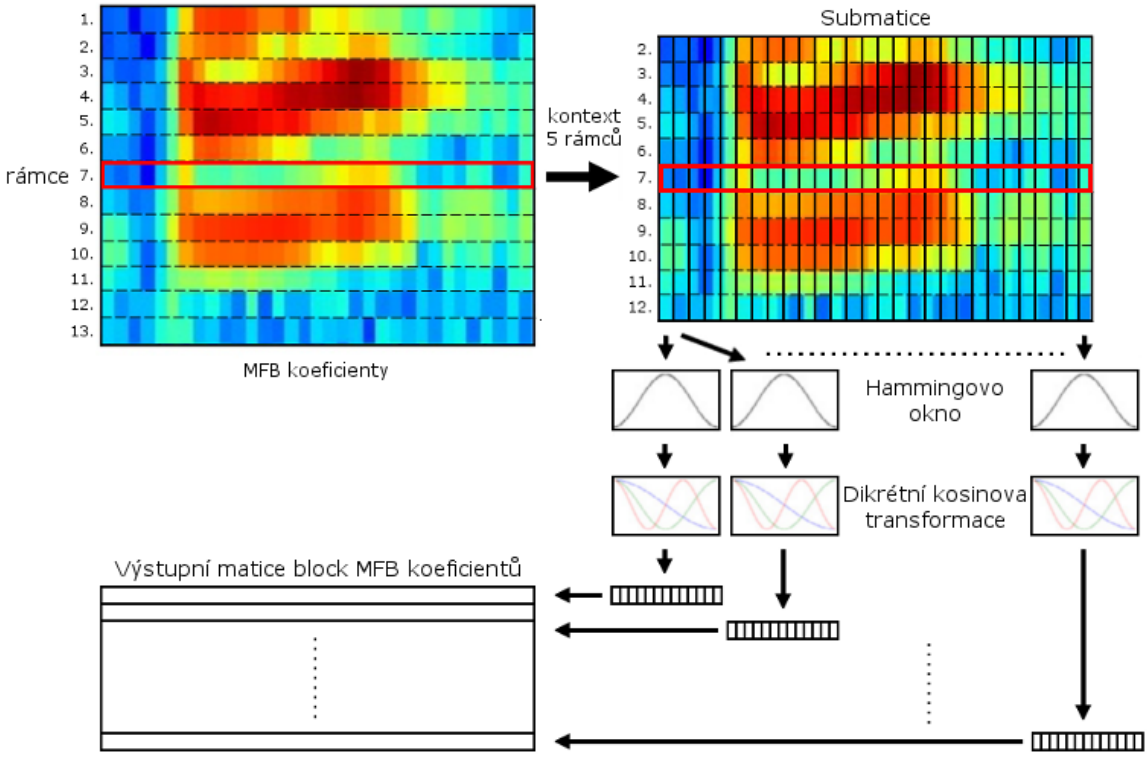


Obrázek 3.3: *Rozložení patnácti trojúhelníkových filtrů přes frekvenční spektrum pomocí mel stupnice*

Výstupy těchto trojúhelníkových filtrů jsou nová spektra, která jsou však pro většinu frekvencí rovna nule. Výjimkou jsou frekvence, které daný filtr propouští. Jejich amplitudy jsou sečteny a po té je vypočítán logaritmus této hodnoty. Tento krok vychází rovněž z vlastností lidského sluchu – odezva na úroveň signálu je logaritmická. Lidské ucho totiž dokáže více rozlišit rozdíly u nízkých amplitud, než u vysokých a navíc jsou tím eliminovány výkonové rozdíly, tedy např. závislost vzdálenosti zdroje zvuku od mikrofonu. Tím je dán výstup jednoho trojúhelníkového filtru. Celkový počet MFB výstupů pro jeden rámec tedy závisí na počtu filtrů.

3.2 Blok mel-filter bank

Tento způsob získávání audio nahrávek, jak již název napovídá, opět vychází z MFB. Pro každý rámec jsou tedy vypočítány MFB koeficienty způsobem, která je popsaný v kapitole 3.1. Jedna nahrávka je po té reprezentována maticí, která se skládá z vektorů MFB koeficientů pro jednotlivé rámce.



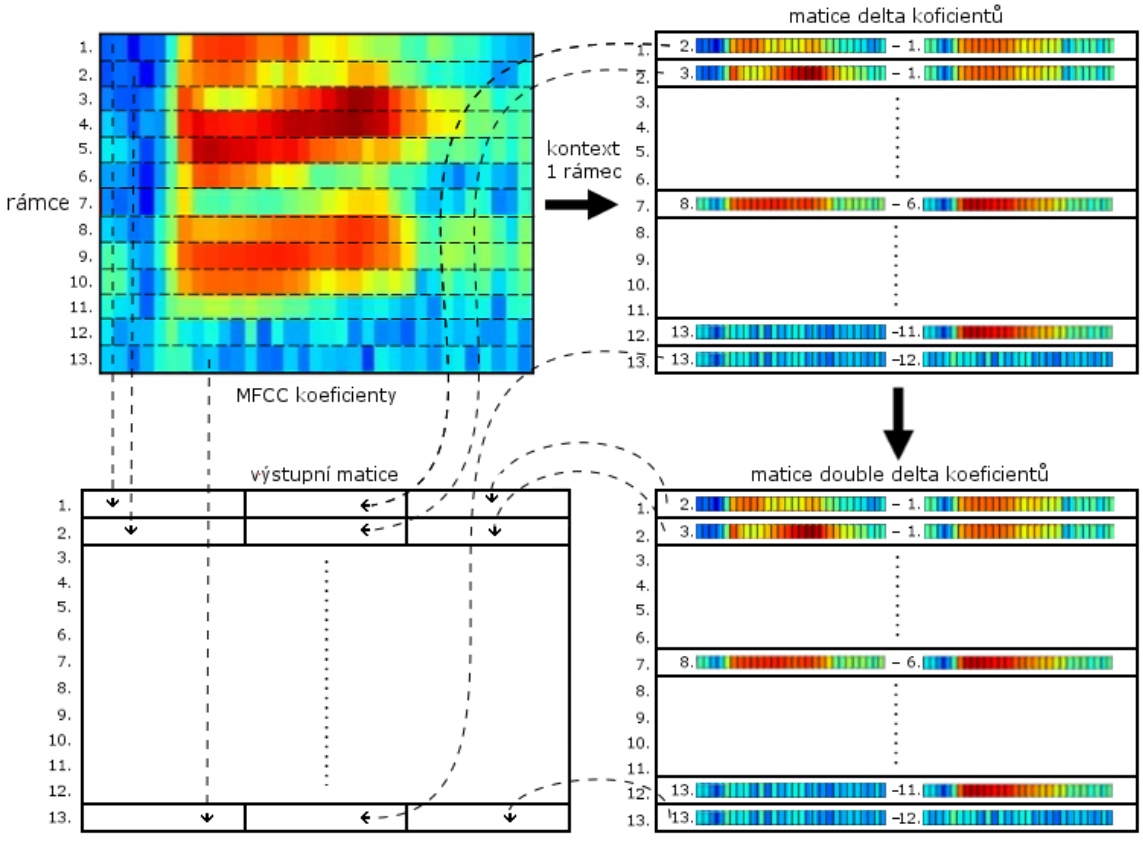
Obrázek 3.4: *Blokový diagram popisující získání blok MFB koeficientů s kontextem 5 rámců*

Následně je v závislosti na nastaveném kontextu vybráno pro daný rámec n okolních vektorů, které tvoří submatici MFB koeficientů. Na odpovídající si výstupy trojúhelníkových filtrů, tedy shodné indexy jednotlivých vektorů, je po té aplikována tzv. *window funkce* a následně je z těchto koeficientů vypočítána diskrétní kosinova transformace [5]. Tento postup je pro kontext s 5 rámci znázorněn na obr. 3.4. Díky tomu je nyní více zřejmé, jak se zastoupení jednotlivých frekvencí vyvíjela v čase. Z výstupů DCT se nejčastěji jako

vstupní hodnoty pro klasifikátor využívá první polovina koeficientů. Tento krok je odůvodněn v předchozí kapitole.

3.3 Mel-frequency cepstral coefficients

Při použití MFCC koeficientů je vypočítána DCT (Discrete Cosine Transform) nebo také IFT (Inverse Fourier Transform) z MFB popsaných v kapitole 3.1. Tento krok se provádí proto, aby bylo frekvenční spektrum převedeno zpět do časové oblasti. MFB spektra zdůrazňují amplitudy jednotlivých frekvencí, které jsou navíc navzájem propojeny, jelikož se jednotlivé filtry vzájemně překrývají. Po aplikaci DCT však jednotlivé příznaky reprezentují množství energie, což přispívá k jejich dekorelaci. Tato skutečnost lépe odráží reálnou charakteristiku audio signálu a nese tak s sebou větší množství informace ve stejném nebo menším počtu koeficientů. Proto se DCT používá i pro jistý druh komprese, např. u formátu JPEG. Po aplikaci DCT se zpravidla pro trénování klasifikátoru využívá pouze první polovina těchto koeficientů, jelikož koeficienty vyšších řádů reprezentují rychlé změny v energiích MFB a jsou tak z hlediska vstupního audio signálu méně podstatné. Druhým, avšak méně důležitým důvodem pro použití jen části DCT výstupů je obecná snaha o to, aby počet koeficientů reprezentující jeden rámec audio signálu byl co nejmenší.



Obrázek 3.5: *Blokový diagram popisující získání výstupní matice při použití delta a double delta koeficientů s kontextem 1 rámec*

3.4 Delta a double delta koeficienty

Protože MFCC koeficienty, popsané v kapitole 3.3, reprezentují množství energie na daných frekvencích pouze pro jeden rámec, chybí ve vlastnostech nahrávky zachycení dynamiky vývoje změn energie v čase, tj. během několika rámců. Proto se využívají tzv. delta, double delta... atd. koeficienty, známé také jako diferenciální a akcelerační koeficienty [9].

Delta koeficienty se vypočítají z matice MFCC pro danou audio nahrávku tak, že se pro každý rámec r s kontextem k rámců vypočítá rozdíl všech odpovídajících si hodnot mezi vektory s indexy $r+k$ a $r-k$. Tím vznikne nová matice delta koeficientů o stejném rozměru, jako původní MFCC matice. Následně se z této delta matice stejným způsobem, jako z matice MFCC, vypočítají double delta koeficienty... atd. Výstup, který se využívá pro trénování klasifikátoru je pak matice, která vznikne spojením původní, delta, double delta... atd. matice. Tento postup je detailně znázorněn na obr. 3.5.

Kapitola 4

Neuronové sítě

V první části této kapitoly je popsán základní typ umělého neuronu. Nejprve je uveden princip jeho činnosti. Po té je vysvětleno, proč je jeho použití velmi omezené a jak je tento nedostatek odstraněn. Ve 2. podkapitole je popsána architektura vícevrstvé neuronové sítě a čím se tento typ sítí vyznačuje. V poslední části je zjednodušeně vysvětleno, jak probíhá proces trénování neuronové a co je to ztrátová funkce,

4.1 Neuron - základní prvek neuronové sítě

Nejjednodušším typem umělého neuronu je *perceptron*. Vstupy tohoto neuronu mohou být výstupy jiných neuronů nebo informace z vnějšího prostředí a mohou nabývat pouze hodnoty 0 nebo 1. Každému vstupu je v závislosti na jeho důležitosti přiřazena určitá váha. Výstupní hodnota neuronu je dána tzv. aktivační přenosovou funkcí, která je definována vztahem 4.1. Ta nabývá hodnoty 1 v případě, že součet vstupních hodnot vynásobených odpovídajícími velikostmi vah je větší, než stanovená prahová hodnota. V opačném případě je rovna 0. Stejně jako vstupní hodnoty může i výstup perceptronu nabývat pouze 2 binárních hodnot. Výstupní hodnota neuronu tedy odpovídá tzv. skokové funkci, která je zobrazena na obr. 4.1a. Je zřejmé, že čím je prahová hodnota nižší, tím se zvyšuje pravděpodobnost, že výstupní hodnota perceptronu bude nabývat hodnoty 1 a naopak [17].

$$f(x) = \begin{cases} 1 & \text{pro } \sum_{i=1}^{N} w_i x_i \geq \text{t} \\ 0 & \text{jinak} \end{cases} \tag{4.1}$$

kde: N – počet vstupů
w_i – váhy jednotlivých vstupů
x_i – jednotlivé vstupy neuronu
t – prahová hodnota

Perceptrony lze použít pouze pro lineárně oddělitelné množiny, tedy pro množiny, pro něž platí, že je lze ve dvourozměrném prostoru oddělit přímkou tak, že všechny prvky jedné množiny jsou na jedné straně přímky a zároveň všechny prvky druhé množiny jsou na druhé straně přímky [29]. Pokud není tato podmínka splněna, nebude učení konvergovat ke správnému řešení. Tato skutečnost značně omezuje využití perceptronů, jelikož se v praxi vyskytuje lineárně oddělitelných množin velmi málo.

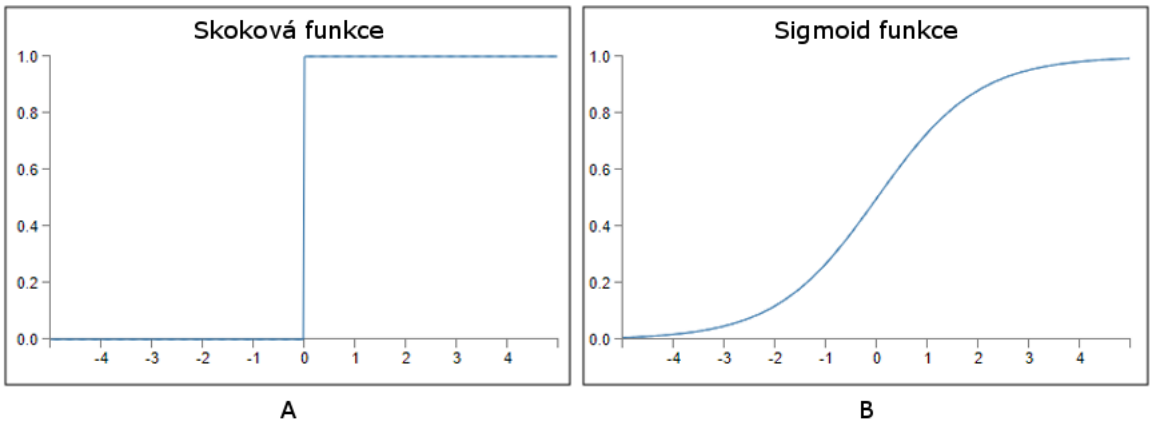
Hlavním nedostatkem perceptronu je, že malá změna jakékoliv váhy neuronu může zcela změnit jeho výstupní hodnotu. Tento problém řeší tzv. *sigmoidní* nebo také *logistický neuron*. Rozdíl mezi perceptronem a sigmoid neuronem je, že u sigmoid neuronu nemusí být výstupní ani vstupní hodnotou neuronu pouze binární hodnota, ale jakékoliv reálné číslo z intervalu $\langle 0{,}1 \rangle$. Této změny je dosaženo změnou aktivační přenosové funkce. Výstupní hodnota neuronu již nemá podobu skokové funkce, ale tzv. *sigmoidní funkce*, která je definována vztahem 4.2 a je zobrazena na obr. 4.1b, přičemž součet vstupních hodnot vynásobených odpovídajícími váhami je dán vztahem 4.3. Kromě aktivační funkce sigmoid existuje celá řada dalších aktivačních funkcí, např. ReLu, tanh, softmax atd [25].

$$f(x) = \frac{1}{1 + e^{z}} \tag{4.2}$$

kde: $f(x)$ – je výstupní hodnota
z – součet vstupních hodnot vynásobených odpovídajícími váhami

$$z = \sum_{i=1}^{N} w_i x_i - b \tag{4.3}$$

kde: z – součet vstupních hodnot vynásobených odpovídajícími váhami
w_i – váhy jednotlivých vstupů
x_i – jednotlivé vstupy neuronu
b – bias = opačná hodnota k prahové hodnotě

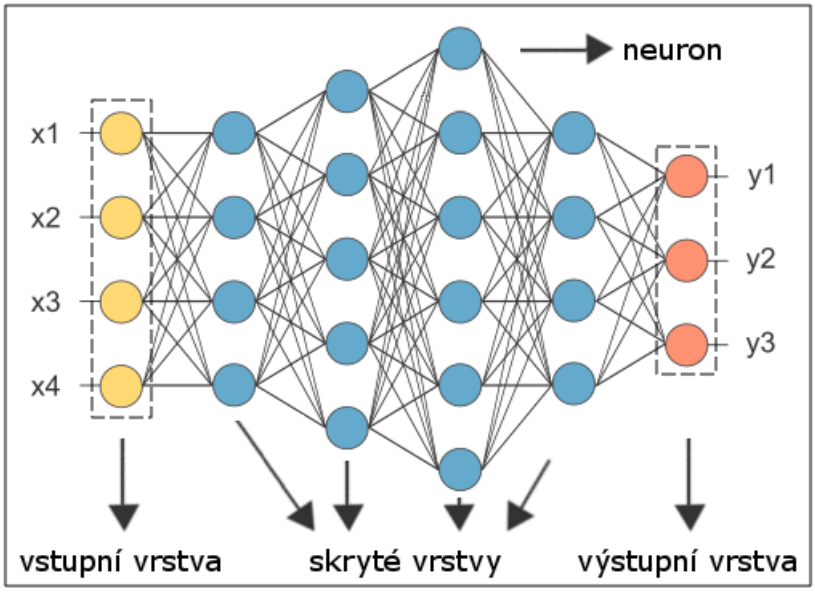


Obrázek 4.1: *Aktivační funkce perceptronu a sigmoid neuronu*

4.2 Architektura vícevrstvé hustě propojené neuronové sítě

Každá vícevrstvá neuronová síť se skládá z minimálně 3 vrstev – jedné vstupní vrstvy, *1* až *n* skrytých vrstev a jedné výstupní vrstvy. Ve vstupní vrstvě je při vytváření neuronové sítě

nutné specifikovat formát vstupních dat. Počet neuronů ve výstupní vrstvě je dán počtem klasifikačních tříd. Velmi často se pro tuto vrstvu používá aktivační funkce *Softmax*, která zajistí, že výstupní hodnoty sítě udávají pravděpodobnost, že vstupní data náleží dané třídě. Příklad topologie vícevrstvé hustě propojené neuronové sítě se 4 skrytými vrstvami je možné vidět na obr. 4.2. Je zřejmé, že každý neuron má vazbu na všechny neurony jak z předchozí, tak z následující vrstvy, což je pro hustě propojenou neuronovou síť charakteristické. Další charakteristickou vlastností pro tento typ neuronových sítí je, že neobsahují žádné smyčky. Výstup neuronu a zároveň i celé sítě tedy vždy závisí na výstupu předchozí vrstvy, případně vstupních datech. Existují však i sítě, ve kterých jsou zpětné smyčky možné. Tyto sítě mají schopnost uchovávat si svůj stav, obsahují tedy jistý druh paměti. Jedná se o tzv. rekurentní neuronové sítě [18].



Obrázek 4.2: *Topologie vícevrstvé neuronové sítě se 4 skrytými vrstvami*

4.3 Trénování neuronové sítě

Pro trénování neuronové sítě jsou zapotřebí 2 druhy dat. První z nich jsou vzorky dat, reprezentující danou klasifikační třídu, např. audio nahrávku, obrázek apod. ve formě n-dimenzionálního pole. Druhým typem dat je 2D pole, kde je první rozměr dán počtem vzorků trénovacích dat, zatímco druhý rozměr je dán počtem klasifikačních tříd. Jedná se o k–dimenzionální vektory, které slouží pro přiřazení určitého vzorku dat ke konkrétní třídě. Příslušnost k dané třídě je dána číslem 1 na konkrétním indexu tohoto vektoru. Hodnoty na ostatních indexech jsou nulové. Pro neuronovou síť na obr. 4.2 by tedy měl tento vektor pro vzorek dat, který přísluší ke 2. třídě, tvar $(0, 1, 0)$.

Pro zjištění,do jaké míry se reálné výstupy neuronové sítě liší od požadovaných výstupů, slouží tzv. MSE (Mean Squared Error), která je někdy také označována jako *ztrátová* nebo *objektivní funkce*. Výpočet MSE je dán rovnicí 4.4. Kromě MSE, která je nejznámější a velmi často používaná, existuje celá řada dalších ztrátových funkcí, jako např. mean absolute error, categorical crossentropy, binary crossentropy atd. [20]. Vysoká hodnota této funkce značí,

že se reálné výstupy neuronové sítě podstatně liší od požadovaných výstupů. Proto je při trénování sítě snaha o to, aby se hodnota ztrátové funkce blížila 0.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2 \tag{4.4}$$

kde: MSE – odchylka reálných výstupů neuronové sítě od požadovaných výstupů

n – počet klasifikačních tříd

Y_i – požadované výstupy neuronové sítě pro daná testovací data

$\hat{Y}_i$ – reálné výstupy neuronové sítě pro daná testovací data

Samotný algoritmus učení neuronové sítě se nazývá *Gradient descent*, který pro výpočet gradientů využívá efektivní metodu *Backpropagation* [16]. Jeho úkolem je najít takové váhy a prahové hodnoty jednotlivých neuronů, aby platilo $MSE \approx 0$. Jelikož se jedná o iterativní algoritmus, je nutné trénovat neuronovou síť na více iteracích trénovací sady. Pro správnou funkci algoritmu je rovněž nezbytné před každou iterací provést náhodné seřazení trénovacích dat. Dalším důležitým parametrem je parametr *batch size*, který udává, po kolika vzorcích trénovacích dat bude vypočtena jedna iterace algoritmu *Gradient descent*.

Kapitola 5

Datová sada pro trénování a evaluaci neuronové sítě

V této kapitole jsou detailně popsány obě datové sady poskytované k soutěži DCASE 2016 a 2017, které jsou v této práci použity jak pro trénování, tak pro evaluaci vytvářeného systému. Jsou zde uvedeny jak shodné vlastnosti, tak odlišnosti obou sad včetně toho, jakým způsobem a v jaké kvalitě byly audio nahrávky pořizovány.

V obou letech, 2016 i 2017, kdy byla soutěž pořádána, byla k prvnímu úkolu k dispozici trénovací datová sada, která byla určená pro trénování soutěžních systémů. Dále byla k tomuto úkolu poskytnuta evaluační sada, na které byla hodnocena přesnost, tedy procentuální úspěšnost jednotlivých systémů. Na každém z patnácti míst byly pořízeny nahrávky dlouhé 3 – 5 minut, které byly rozděleny na několik částí.

V tab. 5.1 jsou pro srovnání uvedeny vlastnosti obou datových sad. Nahrávky v obou datových sadách byly pořizovány se vzorkovací frekvencí 44.1 kHz a s rozlišením 24 bitů. Hlavním rozdílem mezi oběma datovými sadami je, že v datové sadě z roku 2016 jsou nahrávky dlouhé 30 sekund, zatímco v datové sadě z roku 2017 pouze 10 sekund. Další významnou odlišností obou datových sad je počet nahrávek, viz 5.1. Trénovací datová sada je rozdělena čtyřmi různými způsoby na 2 části. První část obsahuje vždy 25 % audio nahrávek, zatímco druhá část zbylých 75 %. Větší z nich je určena pro trénování vyvíjeného systému a menší z nich pro jeho evaluaci. Je to z důvodu tzv. *cross-validace*, tj. aby bylo možné zjišťovat, jaké úspěšnosti klasifikátor dosahuje na různých množinách dat. V tomto případě se jedná o podmnožiny trénovací sady.

Tabulka 5.1: *Odlišnosti datových sad z roku 2016 a 2017*

Parametr	DCASE 2016	DCASE 2017
Vzorkovací frekvence	44.1 kHz	44.1 kHz
Rozlišení	24 bitů	24 bitů
Formát nahrávky	wav	wav
Délka nahrávky	30 s	10 s
Počet trénovacích nahrávek	1170	4680
Počet evaluačních nahrávek	390	1620
Velikost trénovacích dat	8,64 GB	11,5 GB
Velikost evaluačních dat	2,88 GB	3,99 GB

Kapitola 6

Popis systému

V této kapitole je popsán základní systém poskytovaný k prvnímu úkolu soutěže DCASE v roce 2017 a základní systém, se kterým je experimentováno v této práci. Dle zadání měl však systém v této práci vycházet ze soutěže, která proběhla o rok dříve, tedy v roce 2016, kdy byl základní systém implementován pomocí GMM. Po konzultaci s vedoucím bakalářské práce byl však zvolen základní systém z roku 2017, jelikož klasifikace audio nahrávek v datové sadě z tohoto roku je náročnější a obsahuje více dat, viz kapitola 7. Výhodou základního systému 2017 je také to, že je implementován pomocí neuronové sítě, což bylo cílem této práce.

6.1 Základní systém poskytnutý k soutěži DCASE

K prvnímu úkolu soutěže DCASE byl poskytován základní systém využívající neuronovou síť, která se skládá ze vstupní vrstvy, tří *Dense* vrstev, dvou vrstev *Dropout* a jedné výstupní vrstvy, jak ukazuje tab. 6.1. *Dense* vrstva je skrytá vrstva neuronové sítě, která je detailně popsána v kapitole 4.2. *Dropout* je vrstva, která při každém trénování náhodně deaktivuje určitý počet neuronů ve skryté vrstvě. Množství deaktivovaných neuronů závisí na pravděpodobnosti, s jakou je každý neuron deaktivován. Tuto pravděpodobnost je nutné specifikovat při vytváření sítě. Neuronová síť se po deaktivaci některých neuronů chová tak, jako by tyto neurony v dané vrstvě při konkrétní iteraci vůbec neexistovaly. Tento princip se uplatňuje pouze při trénování. Při evaluaci je pravděpodobnost deaktivace neuronu nastavena na 0 [23]. Díky *Dropout* vrstvě se zabraňuje přetrénování neuronové sítě na konkrétní trénovací data. Důsledkem přidání této vrstvy do modelu je také rychlejší trénování modelu, jelikož během jedné iterace je do trénovacího algoritmu zapojen menší počet neuronů než v případě absence této vrstvy.

Z tab. 6.1 je zřejmé, že model obsahuje 2 skryté vrstvy a každá z nich má 50 neuronů. Tyto vrstvy využívají aktivační funkci *ReLu* (Rectified Linear unit). Poslední, výstupní, vrstva obsahuje 15 neuronů, což odpovídá počtu klasifikačních tříd, které jsou uvedeny v tab. 2.1. Využívá aktivační funkci *Softmax*, díky níž jsou výstupy neuronové sítě rovnají pravděpodobnosti, že vstupní audio nahrávky náleží dané třídě. Součet pravděpodobností pro všechny třídy je tedy 1.

Tabulka 6.1: *Topologie vícevrstvé neuronové sítě*

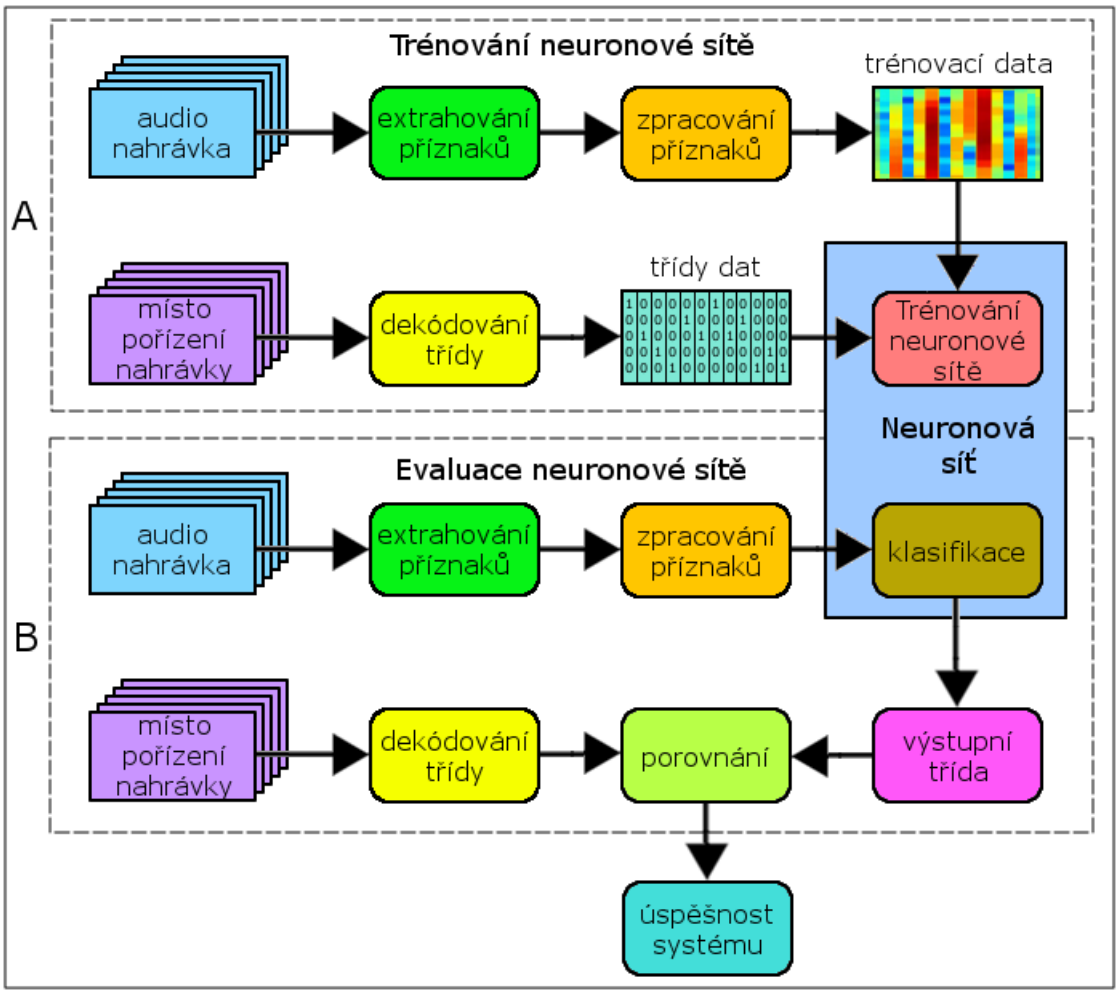
Input layer()
Dense(ReLu, 50)
Dropout(0.2)
Dense(Relu, 50)
Dropout(0.2)
Dense(Softmax,15)

Základní systém využívá jak pro trénování, tak pro evaluaci, průměrnou hodnotu obou kanálů, přičemž původní signál není převzorkován na nižší frekvenci, ale jsou použity všechny vzorky signálu a není použita žádná normalizace. Jako vlastnosti jednotlivých nahrávek využívá MFB koeficienty. Je použito 40 trojúhelníkových filtrů. Délka jednoho rámce je 40 ms s překrýváním 50 %, tedy 20 ms. Neuronová síť je trénována na segmentu MFB koeficientů odpovídajících pěti rámcům. Vstupní vektor má tedy 200 hodnot. Síť je trénována na 200 iteracích s využitím optimalizátoru *Adam*. Parametr *learning rate* je nastaven na 0.001 a velikost *batch size* na 256. Finální klasifikace je implementována pomocí techniky zvané *Voting*, která je popsána v kapitole 2.2. Pro optimalizaci matematických výpočtů využívá jako backend knihovnu *Theano*.

Na evaluační sadě dosáhl tento poskytovaný základní systém úspěšnosti 61 % a v celkovém pořadí se ze 41 týmů umístil na 31. místě.

6.2 Popis vytvořené neuronové sítě

Základní systém v této práci vychází ze základního systému soutěže DCASE 2017. Hlavním rozdílem obou systémů je použití odlišných příznaků reprezentace audio nahrávky. Topologie neuronové sítě zůstala po mnoha provedených experimentech s počtem skrytých vrstev velmi podobná topologii základního systému této soutěže, uvedené v tab. 6.1, jelikož přidávání dalších vrstev nevedlo k lepším výsledkům. Experimentováno bylo rovněž s počtem neuronů ve skrytých vrstvách. Při vyšším množství trénovacích dat dosahovala síť s větším počtem neuronů lepších výsledků. z tohoto důvodu obsahují obě skryté vrstvy 200 neuronů, tedy o 150 neuronů více, než má základní systém ze soutěže DCASE 2017. Při dalším zvyšování počtu neuronů ve skrytých vrstvách se již úspěšnost systému nezvyšovala, avšak trénování neuronové sítě bylo pomalejší. Proto byl použit výše zmíněný počet neuronů v každé skryté vrstvě.



Obrázek 6.1: *Blokové schéma navrženého systému*

Celý systém se dělí na 2 hlavní části – trénování neuronové sítě a evaluace neuronové sítě. Průběh trénování neuronové sítě je znázorněn na obr. 6.1A. Nejprve jsou načtena data a místo pořízení audio nahrávky ze zvolené trénovací sady, po té jsou ze vzorků pro danou audio nahrávku získány příznaky, charakterizující danou audio nahrávku, a následně jsou zpracovány a rozděleny na určitý počet segmentů. Způsob zpracování příznaků a počet segmentů, na které je audio nahrávka rozdělena, závisí na použitém způsobu reprezentace vlastností jednotlivých nahrávek. Může se jednat o MFB, MFCC nebo delta a double delta koeficienty. Z místa pořízení audio nahrávky je dekódována třída, do které audio nahrávka patří a po té je převedena do formátu požadovaného neuronovou sítí. Tento formát je detailně popsán v kapitole 4.3. Tímto způsobem jsou zpracovány všechny audio nahrávky. Trénování není možné provádět po jednotlivých nahrávkách z důvodu nutnosti náhodného seřazení dat před každou trénovací iterací. Výsledkem jsou 2 matice, přičemž první z nich reprezentuje příznaky všech audio nahrávek a druhá z nich třídy, do kterých audio nahrávky patří. Po té je s pomocí těchto matic spuštěno trénování neuronové sítě.

Evaluace neuronové sítě, jejíž průběh je znázorněn na obr. 6.1B, probíhá na evaluační sadě v závislosti na zvolené trénovací sadě. Nejprve jsou získány příznaky ze vzorků jedné nahrávky stejným způsobem, jako při trénování neuronové sítě, který je popsaný v před-

chozím odstavci. Hlavní rozdíl spočívá v tom, že predikce místa, odkud audio nahrávka pochází, probíhá po jednotlivých nahrávkách, nikoli pro všechny nahrávky naráz, tak jak je to při trénování neuronové sítě. Pro každý segment audio nahrávky je získán vektor výstupních hodnot z neuronové sítě. Tyto vektory výstupních hodnot jsou po té zprůměrovány technikou zvanou *Voting*, která je popsána v kapitole 2.2. Výsledný vektor je porovnán s očekávanými výsledky. Tento postup je proveden pro všechny evaluační audio nahrávky. Po té je na základě výsledků vypočítána úspěšnost klasifikace jednotlivých tříd a celková úspěšnost systému.

Správně, podle pravidel soutěže, by se měla úspěšnost systému během jeho vývoje ověřovat na části trénovací sady s využitím tzv. *cross-validace* tak, jak to prováděly jednotlivé týmy, které se soutěže zúčastnily. Až po té by měla být úspěšnost finálního klasifikátoru stanovena pomocí evaluační sady. Protože však byla evaluační sada při vytváření této bakalářské práce již k dispozici, je využita pro stanovení úspěšnosti systému při každém experimentu.

Kapitola 7

Experimenty

V této kapitole jsou popsány experimenty, které byly prováděny se systémem, který je popsaný v kapitole 6.2. V zadání této bakalářské práce bylo zadáno, že má být použita datová sada z roku 2016, po prvních experimentech a konzultaci s vedoucím bakalářské práce však byla zvolena datová sada z roku 2017, jelikož jsou audio záznamy v této datové sadě o dvě třetiny kratší a obsahuje celkově více dat. Klasifikace nahrávek je tedy náročnější a výsledky více věrohodné. Systém, který dosáhl nejvyšší úspěšnosti na datové sadě z roku 2017 je trénován a testován na obou těchto sadách, výsledky jsou uvedeny v kapitole 7.12.

7.1 Levý a pravý kanál stereo signálu

Audio nahrávky v obou datových sadách byly pořízeny ve stereo kvalitě s tím, že levý a pravý kanál jsou velmi podobné, ovšem obsahují určité odlišnosti. Z tohoto důvodu bylo prvním experimentem zjištění úspěšnosti neuronové sítě při trénování na levém kanálu, pravém kanálu a průměrné hodnotě obou kanálů. Z tab. 7.1 vyplývá, že při trénování a následné evaluaci neuronové sítě na levém kanálu audio nahrávky je dosažená úspěšnost o 1.54–2.53 % nižší v závislosti na velikosti parametru *batch size*, než při použití pravého kanálu stereo signálu. Přesnost klasifikace se při trénování pouze na pravého kanálu a průměrné hodnotě obou kanálů pro jednotlivé třídy lišila. Z výsledků je však zřejmé, že celková úspěšnost systému je zcela shodná a to pro oba parametry *batch size*.

Při tomto experimentu byla neuronová síť trénována na MFB koeficientech s parametrem *learning rate* $1 * 10^{-4}$ a optimalizátorem Adam [19]. Nahrávky byly z původní vzorkovací frekvence 44.1 kHz převzorkovány na 8 kHz a každá z nich byla rozdělena na rámce po 200 vzorcích s překrýváním 60 %, tedy 120 vzorků. Na každý rámec bylo aplikováno 24 filtrů v rozmezí 64 Hz – 3800 Hz, tedy hodnoty o 200 Hz menší, než je polovina vzorkovací frekvence. Velikost segmentu vstupních dat je jeden rámec. Pokud není řečeno jinak, jsou tyto parametry použity i ve zbývajících experimentech.

Tabulka 7.1: *Dosažená celková úspěšnost neuronové sítě při trénování a evaluaci na jednotlivých kanálech a průměrné hodnotě obou kanálů*

Batch size	Úspěšnost [%]		
	L kanál	P kanál	$\frac{L+P}{2}$
100	56.05	58.15	58.64
200	56.30	58.52	58.52
300	56.23	**58.77**	**58.77**

7.2 Rozšíření trénovací sady

Dalším experimentem bylo zkombinování obou kanálů a jejich průměrné hodnoty, ve snaze zvýšit množství trénovacích dat. Díky tomu se trénovací sada zvětšila 2–3x v závislosti na použité kombinaci. V experimentech, prováděných v kapitole 7.1, dosáhl systém při trénování pouze na pravém kanálu a pouze na průměrné hodnotě obou kanálů shodné a zároveň nejvyšší úspěšnosti ze všech experimentů. Spojení těchto 2 datových sad vedlo ze všech možných kombinací rovněž k nejvyšší úspěšnosti, jak ukazuje tab. 7.2, kdy byla výsledná úspěšnost 58.45 %. Tato hodnota je však o 0.31 % horší, než při trénování a evaluaci neuronové sítě pouze na pravém kanálu nebo pouze na průměrné hodnotě obou kanálů. Rozšíření trénovací sady pomocí různých kombinací obou kanálů tedy nevedlo k lepším výsledkům, proto je v dalších experimentech využita pro trénování a evaluaci neuronové sítě pouze průměrná hodnota obou kanálů, není-li řečeno jinak.

Tabulka 7.2: *Dosažená celková úspěšnost neuronové sítě při trénování na kombinacích levého, pravého a průměrné hodnoty kanálů*

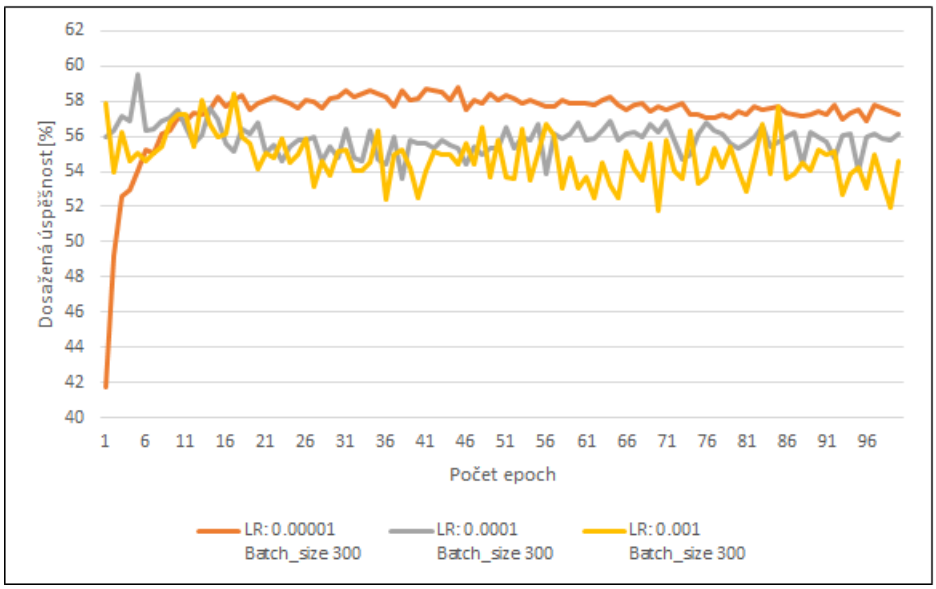
Batch size	Úspěšnost [%]			
	$L+P$	$L+\frac{L+P}{2}$	$P+\frac{L+P}{2}$	$L+P+\frac{L+P}{2}$
100	58.02	57.41	**58.46**	57.84
200	58.33	57.35	58.27	57.53
300	58.27	57.65	58.27	57.65

7.3 Změna parametru learning rate

7.3.1 Pevná hodnota learning rate

V této kapitole bylo experimentováno s parametrem *learning rate*, jehož velikost zůstala pro všechny trénovací iterace stejná. Graf 7.1 ukazuje závislost úspěšnosti systému na různých hodnotách tohoto parametru pro *batch size* velikosti 300. Je zřejmé, že po 100 iteracích dosahoval nejvyšší úspěšnosti, přesně 57.28 %, systém s nejmenší hodnotou parametru *learning rate*, tedy s hodnotou $1 * 10^{-5}$ a výsledná úspěšnost byla v rámci jednotlivých iterací více ustálená oproti ostatním hodnotám tohoto parametru. Z grafu dále vyplývá, že se úspěšnost zvyšovala do 46 iterace, kdy dosáhla hodnoty 58.77 % a po té začala postupně klesat. Pro *learning rate* vyšší, než je uveden v grafu 7.1, se síť natrénovala vždy na trénovací data, která byla poslední v pořadí. Pro jakýkoliv vstup pak vždy vracela stejnou třídu. Vzhledem

k počtu tříd tak byla dosažená úspěšnost 6.67 %. Obdobně se síť chovala i pro *learning rate* menší než $1 * 10^{-5}$, kdy se ani po 100 iteracích nic nenaučila.



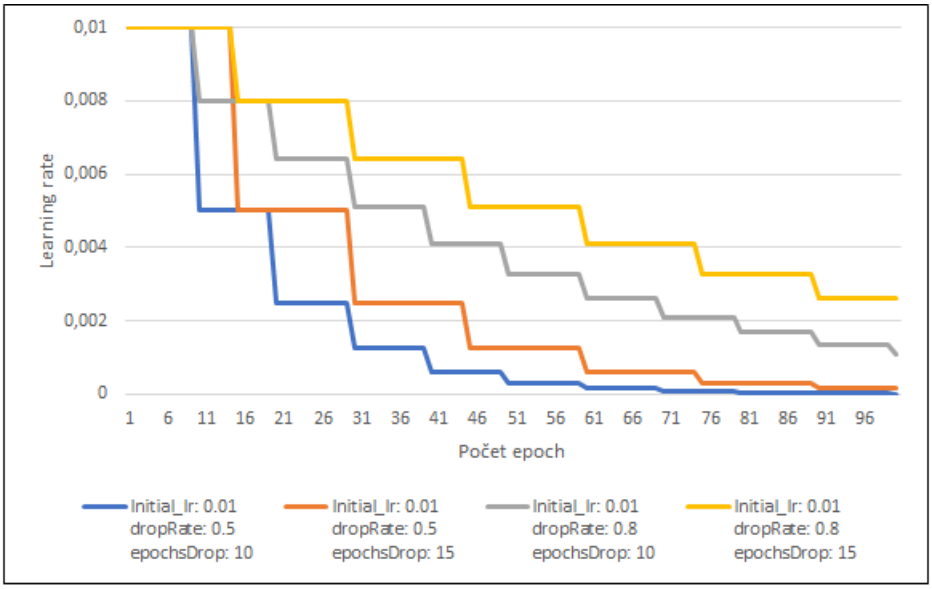
Obrázek 7.1: *Závislost úspěšnosti systému na fixním velikosti parametru learning rate*

7.3.2 Skoková změna learning rate v závislosti na počtu iterací trénování

Druhou možností je dynamické snižování parametru *learning rate* v závislosti na aktuální trénovací iteraci. Prvním způsobem pro změnu v průběhu trénování neuronové sítě je tzv. *Drop-Based Learning Rate Schedule* [1]. Parametr *learning rate* není měněn pro každou trénovací iteraci, ale skokově po předem stanoveném počtu iterací, což vyplývá z následujícího vztahu 7.1 pro výpočet aktuální hodnoty *learning rate*.

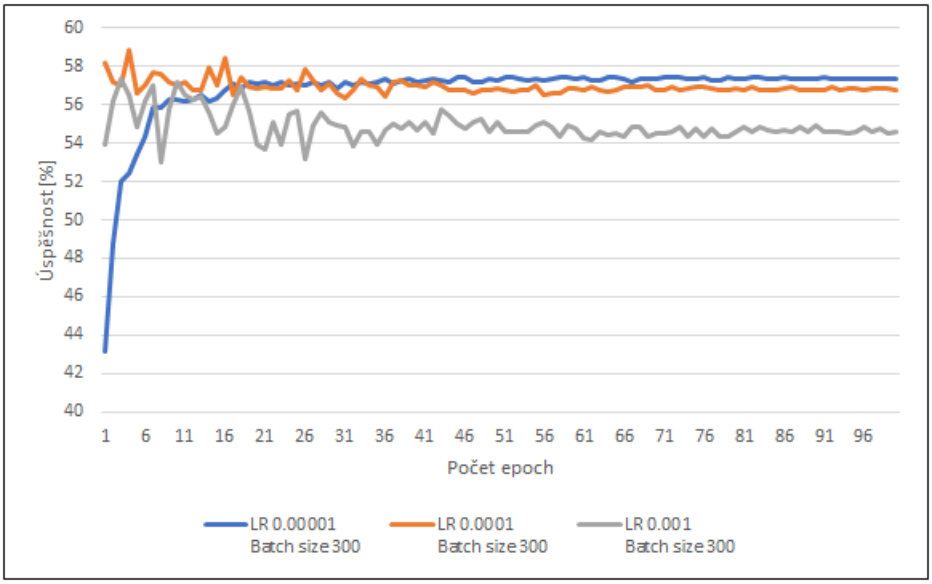
$$LearningRate = InitialLearningRate * dropRate^{floor\left(\frac{epoch}{epochDrop}\right)} \tag{7.1}$$

kde: $InitialLearningRate$ = Počáteční hodnota learning rate,
$LearningRate$ – je aktuální hodnota parametru learning rate,
$dropRate$ – parametr, který určuje velikost změny learning rate,
$epochDrop$ – parametr, který určuje, po kolika iteracích se bude learning rate měnit,
$epoch$ – aktuální iterace trénování neuronové sítě



Obrázek 7.2:
Závislost změny parametru learning rate na parametrech dropRate, epochDrop a počtu iterací trénování

V grafu 7.2 je vidět, že po 100 iteracích byla nejvyšší úspěšnost systému, 57,47 %, dosažena pro nejnižší počáteční *learning rate*, stejně jako při fixním nastavení rychlosti učení neuronové sítě. Při srovnání výsledků s grafem 7.4 je zřejmé, že oproti fixnímu parametru *learning rate* se úspěšnost o 0.19 % zlepšila a ustálila se na stabilní hodnotě.



Obrázek 7.3: *Dosažená úspěšnost neuronové sítě pro různé inicializační hodnoty parametru learning rate při použití snižování pomocí Drop-Based Learning Rate Schedule s dropRate = 0.5 a epochDrop = 10*

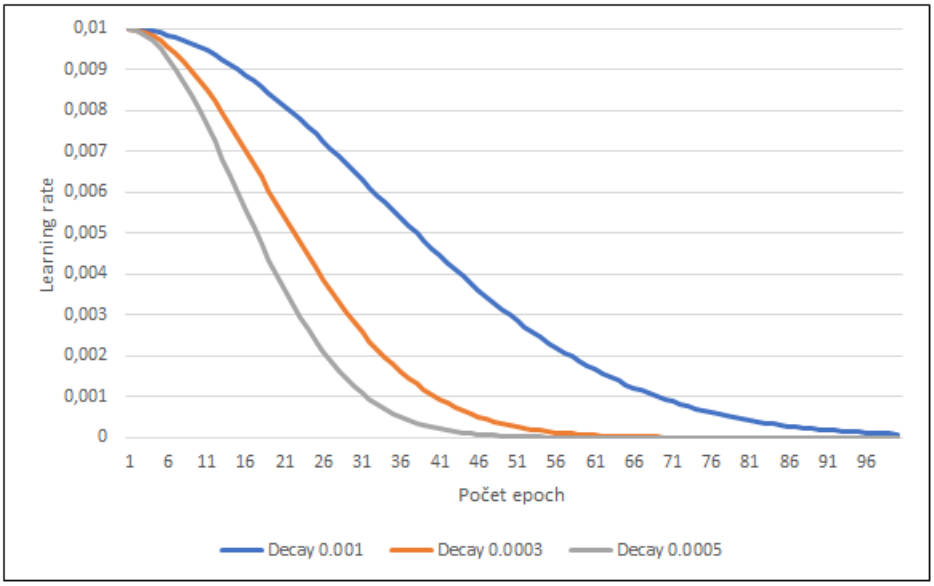
7.3.3 Plynulá změna learning rate v závislosti na počtu iterací trénování

Druhým způsobem pro dynamickou změnu parametru *learning rate* je tzv. *Time-Based Learning Rate Schedule* [1]. Narozdíl od předchozího způsobu není změna skoková po určitém počtu trénovacích iterací, ale plynulá. Při každé iteraci má tedy parametr *learning rate* jinou hodnotu, která je vypočtena dle vztahu 7.2.

$$LearningRate = LearningRate * \frac{1}{(1 + decay * epoch)} \tag{7.2}$$

kde: $LearningRate$ – je aktuální hodnota parametru learning rate, který musí být inicializován na počáteční hodnotu,
$decay$ – parametr, který určuje rychlost snižování learning rate,
$epoch$ – aktuální iterace trénování neuronové sítě

Závislost rychlosti snižování *learning rate* na různých hodnotách parametru *decay* je patrná z grafu 7.4.

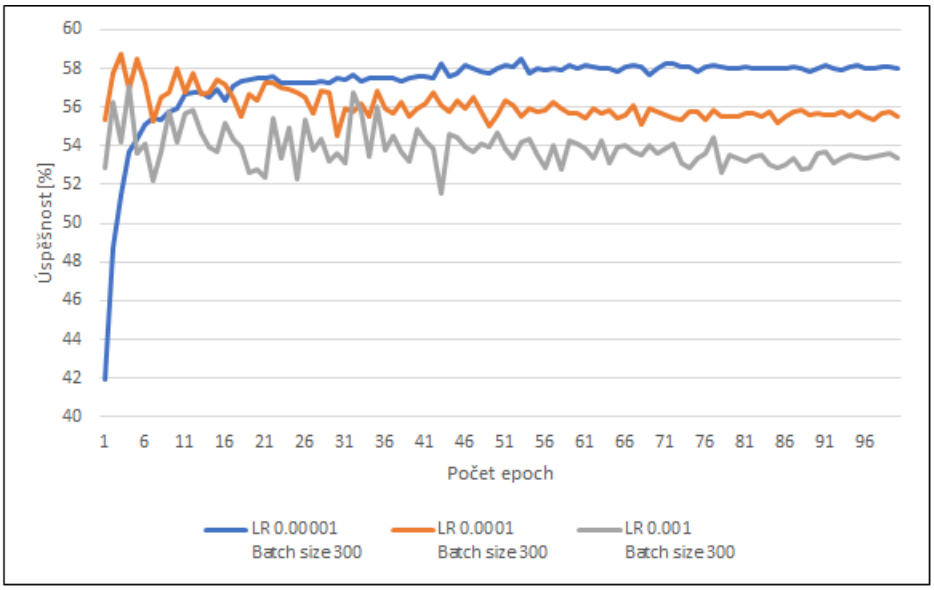


Obrázek 7.4: *Plynulá změna parametru learning rate v závislosti na počtu iterací trénování a na zvoleném parametru decay*

Z grafu 7.5 vyplývá, že po 100 iteracích učení dosahuje systém při použití *Time-Based Learning Rate Schedule* pro snižování *learning rate* nejvyšší úspěšnosti, 58.08 %, pro nejnižší počáteční hodnotu tohoto parametru, stejně jako v předchozích dvou experimentech. Oproti fixní rychlosti učení se úspěšnost zvýšila o 0.8 %. Při srovnání s předchozí metodou skokového snižování byla výsledná přesnost rovněž o 0.61 % vyšší.

Z tohoto experimentu tedy vyplývá, že dynamické snižování parametru *learning rate* v průběhu trénování neuronové sítě přispívá k nepatrně lepším výsledkům. Z grafu 7.1 je zřejmé, že při nastavení parametru *learning rate* na fixní hodnotu dochází po určitém počtu iterací k přetrénovávání neuronové sítě. Z výsledků obou použitých způsobů pro snižování

tohoto parametru v průběhu učení vyplývá, že tento problém odstraňují, jelikož se zvyšujícím se počtem iterací se přesnost klasifikace ustaluje na určité hodnotě. Experimentováno bylo se dvěma metodami snižování *learning rate* – *Drop-Based Learning Rate Schedule* a *Time-Based Learning Rate Schedule*, přičemž vyšší konečné přesnosti po 100 trénovacích iteracích bylo dosaženo z druhým z nich a to o 0.8 % oproti fixní hodnotě.



Obrázek 7.5: *Dosažená úspěšnost neuronové sítě pro různé počáteční hodnoty learning rate při použití snižování pomocí Time-Based Learning Rate Schedule parmetrem decay = 0.001*

7.4 Normalizace vstupních dat

V doposud prováděných experimentech byla použita nenormalizovaná data. V tomto experimentu bylo tedy cílem vyzkoušet, do jaké míry ovlivní výsledky normalizace dat a zda přinese lepší výsledky. Pro normalizaci dat byla použita tzv. *z-score* normalizace [22], tedy normalizace na základě odchylky od průměrné hodnoty.

Při použití normalizace dat po jednotlivých nahrávkách byl vypočítán vektor průměrných hodnot podle rovnice 7.3 a vektor směrodatných odchylek podle rovnice 7.4 z matice MFB koeficientů pro každou nahrávku zvlášť a to jak pro trénovací, tak pro evaluační data. Na základě takto vypočtených vektorů středních hodnot a směrodatných odchylek byla provedena normalizace matice MFB koeficientů, kterou vyjadřuje rovnice 7.5.

Obdobným způsobem byla provedena i globální normalizace, avšak vektory středních hodnot a směrodatných odchylek nebyly vypočítány jen z matic MFB koeficientů pro každou nahrávku zvlášť, ale ze všech matic MFB koeficientů pro trénovací data. Tyto vektory po té byly využity pro normalizaci dat.

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{7.3}$$

kde: N – je počet vektorů v datech,
x_i – odpovídající MFB koeficient v každém vektoru

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{7.4}$$

kde: N – je počet vektorů v datech,
x_i – odpovídající MFB koeficient v každém vektoru
μ – střední hodnota pro daný MFB koeficient,

$$z = \frac{x - \mu}{\sigma} \tag{7.5}$$

kde: z – jsou normalizovaná výstupní data,
x – nenormalizovaná vstupní data,
μ – vektor středních hodnot
σ – vektor směrodatných odchylek

V tab. 7.3 je uvedena výsledná úspěšnost systému pro oba způsoby normalizace a s různými velikostmi *batch size* při trénování neuronové sítě. Z výsledků vyplývá, že normalizace vstupních dat po jednotlivých nahrávkách přinesla výrazné zhoršení úspěšnosti klasifikátoru. Při použití globální normalizace se dosažená úspěšnost pohybovala mezi 55.74 – 56.17 %, což je při srovnání úspěšnosti se základním systémem, viz tab. 7.1, o 2–3 % horší výsledek. Z výsledků je patrné, že hustě propojená neuronová síť se lépe učí na datech, jejichž hodnoty jsou rozptýlené do většího intervalu, než na normalizovaných datech.

Kromě výše uvedeného způsobu lze pro normalizaci vstupních dat pro neuronovou síť využít i jiné způsoby normalizace [12], např.:

- min-max normalizace – lineární transformace
- dekadická normalizace – posuv desetinné čárky hodnot tak, aby spadaly do daného intervalu
- soft-max normalizace – nelineární transformace logistickou funkcí

Vzhledem k tomu, že použití *z-score* normalizace vedlo k horším výsledkům, než při trénování sítě na nenormalizovaných datech, nezabývá se tato práce experimenty s dalšími možnými způsoby normalizace vstupních dat.

Tabulka 7.3: *Dosažená úspěšnost neuronové sítě v závislosti na velikosti parametru batch size a použité normalizaci*

Batch size	Normalizace	Úspěšnost [%]
100	Globální	56.17
	Jednotlivé nahrávky	28.95
200	Globální	55.74
	Jednotlivé nahrávky	29.01
300	Globální	55.93
	Jednotlivé nahrávky	28.95

7.5 Vzorkovací frekvence vstupního audio signálu

Před samotným zpracováním vstupního audio signálu, který by ve formátu *wav*, jej bylo nutné převést na formát *raw*. Další experiment se tedy týkal toho, s jakou vzorkovací frekvencí je původní audio signál převeden. Experimenty byly prováděny se vzorkovací frekvencí 8 kHz, 16 kHz a 44.1 kHz.

V tab. 7.4 jsou uvedeny dosažené výsledky, pro úplnost a možnost porovnání je zde uvedena i úspěšnost z předchozího experimentu pro vzorkovací frekvenci 8 kHz popsaného v kapitole 7.3.1. Z výsledků tohoto experimentu vyplývá, že zdvojnásobení vzorkovací frekvence z 8 kHz na 16 kHz přineslo jen drobné zlepšení pro *batch size* 100, zatímco pro zbývající dvě velikosti tohoto parametru byly výsledky průměrně o 0.55 % horší. Při ponechání původní vzorkovací frekvence, tedy 44.1 kHz, se však úspěšnost zvýšila pro všechny velikosti *batch size* o 5.5 – 5.98 %. Nejlepšího výsledku tedy dosáhl systém pro velikost *batch size* 300 a vzorkovací frekvenci 44.1 kHz, kdy byla dosažená úspěšnost 64.75 %. Při použití vyšší vzorkovací frekvence je k dispozici více vzorků audio signálu, tedy i lepší rozlišení v časové oblasti. Výstupy trojúhelníkových filtrů tak díky většímu množství dat pro jednotlivé rámce lépe charakterizují klasifikační třídy a systém tak dosahuje vyšší přesnosti klasifikace.

Tabulka 7.4: *Dosažená úspěšnost neuronové sítě v závislosti na použité vzorkovací frekvenci*

Batch size	Vzorkovací frekvence [kHz]	Úspěšnost [%]
100	8	58.64
	16	58.77
	44.1	64.14
200	8	58.52
	16	58.09
	44.1	64.38
200	8	58.77
	16	58.09
	44.1	**64.75**

7.6 Využití Blok mel-filter bank koeficientů

Pro experimenty v této kapitole byla použita data se vzorkovací frekvencí 44.1 kHz, jelikož v předchozím experimentu dosáhla výrazně lepších výsledků. Parametr *learning rate* byl nastaven na fixní hodnotu $1 * 10^{-5}$, protože v experimentu, popsaném v kapitole 7.3.1, bylo s touto hodnotu dosaženo nejlepších výsledků. Pro reprezentaci audio nahrávek byly využity blok MFB koeficienty, jejichž výpočet je popsán v kapitole 3.2.

7.6.1 Velikost kontextu

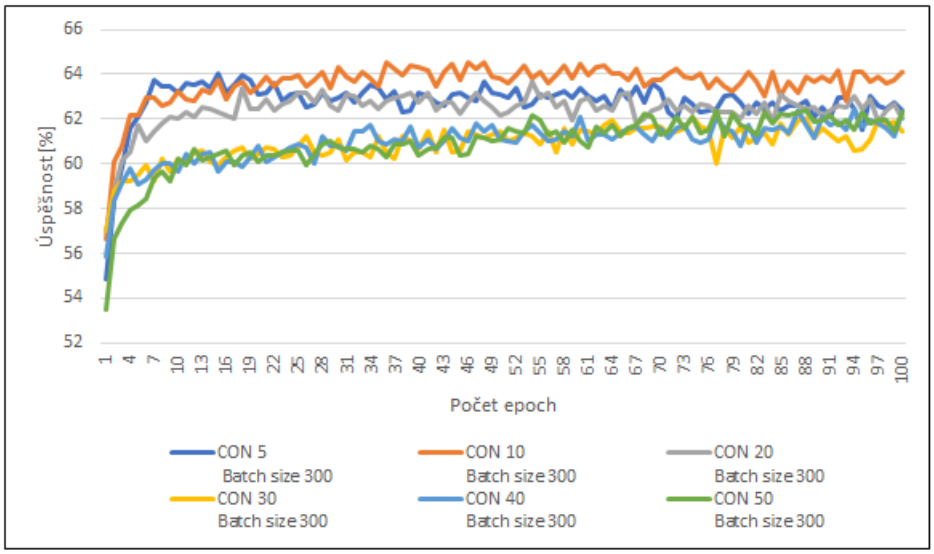
První experimentem, prováděném s blok MFB koeficienty byla změna kontextu pro jeden rámec, tedy počet okolních rámců, který je zahrnut do výpočtu DCT. Kontext rámců byl braný vždy symetricky na obě strany. Výpočet blok MFB koeficientů s kontextem 5 rámců tedy znamená, že pro jeden rámec je využito 5 předchozích rámců, aktuálně zpracovávaný rámec a 5 následujících rámců. Celkem je to tedy 11 rámců. Vše znázorňuje obr. 3.4 v kapitole 3.2. Ve všech experimentech v této části bylo pro každý řádek MFB submatice vypočítáno 6 DCT koeficientů.

Experimenty probíhaly s kontextem 5, 10, 20, 30, 40 a 50 rámců. Dosažené úspěšnosti v závislosti na kontextu rámců a velikosti parametru *batch size* jsou uvedeny v tab. 7.5. Je zřejmé, že pro všechny jeho velikosti byla přesnost klasifikace nejvyšší pro kontext 10 rámců. V tabulce jsou uvedeny pouze maximální hodnoty, z grafu 7.6 je však patrné, jak se pohybovala úspěšnost neuronové sítě v závislosti na zvoleném kontextu pro *batch size* velikosti 300. Pro ostatní velikosti tohoto parametru vypadal graf obdobně. Z grafu vyplývá, že nejvyšší úspěšnosti pro kontext 10 rámců nebylo dosaženo pouze při jedné konkrétní iteraci, ale neuronová síť dosahovala s tímto nastavením, při zanedbání počátečních 20 iterací, nejvyšší průměrné úspěšnosti.

Z výsledků vyplývá, že při použití blok MFB koeficientů celkovou úspěšnost systému výrazně ovlivňuje zvolený kontext, jelikož se výsledná úspěšnost lišila v závislosti na použitém kontextu pro jednotlivé velikosti *batch size* o 2.78–3.76 %. Jak je vidět v tab. 7.5, nejvyšší dosažená úspěšnost činí 64.88 %, což je o 0.13 % více, než při použití MFB koeficientů.

Tabulka 7.5: *Maximální dosažená úspěšnost neuronové sítě v závislosti na velikosti kontextu MFB koeficientů*

Kontext [počet rámců]	Batch size	Úspěšnost [%]	Kontext [počet rámců]	Batch size	Úspěšnost [%]
5	100	64.07	30	100	61.91
	200	63.89		200	62.53
	300	64.01		300	62.10
10	100	64.69	40	100	62.04
	200	64.63		200	61.30
	300	**64.88**		300	62.47
20	100	63.52	50	100	60.93
	200	62.72		200	62.22
	300	63.70		300	62.41



Obrázek 7.6: *Dosažená úspěšnost pro různou velikost kontextu rámců při použití blok MFB koeficientů v závislosti na počtu iterací trénování neuronové sítě*

7.6.2 Počet DCT koeficientů

V této podkapitole byly prováděny experimenty s počtem DCT koeficientů vypočítaných pro shodné trojúhelníkové filtry v sousedních rámcích. Protože bylo v předchozím experimentu dosaženo nejvyšší úspěšnosti pro systém s kontextem 10 rámců, bylo v této části experimentováno právě s tímto systémem.

Počet DCT koeficientů byl měněn v rozsahu od 4 do 18 koeficientů s krokem 2. V tabulce 7.6 jsou uvedeny výsledky dosažených experimentů. Výsledná úspěšnost se v závislosti na počtu DCT koeficientů lišila pro velikost *batch size* 200 o 1.91 %, pro velikost *batch size* 300 pak o 2.83 %. Nejvyšší úspěšnosti dosáhl systém pro 12 DCT koeficientů. Při dalším zvyšování DCT koeficientů začala úspěšnost opět klesat.

Jelikož byla druhá nejlepší úspěšnost dosažena pro obě velikosti parametru *batch size* při 6 DCT koeficientech, tedy při stejném počtu koeficientů, které byly použity v experimentu s velikostí kontextu v kapitole 7.6.1, nebylo zvýšení úspěšnosti tak výrazné, jako kdyby byl předchozí experiment prováděn s např. 10 DCT koeficienty. Oproti předchozímu systému se tedy přesnost systému zvýšila pouze o 0.1 % na 64,98 %.

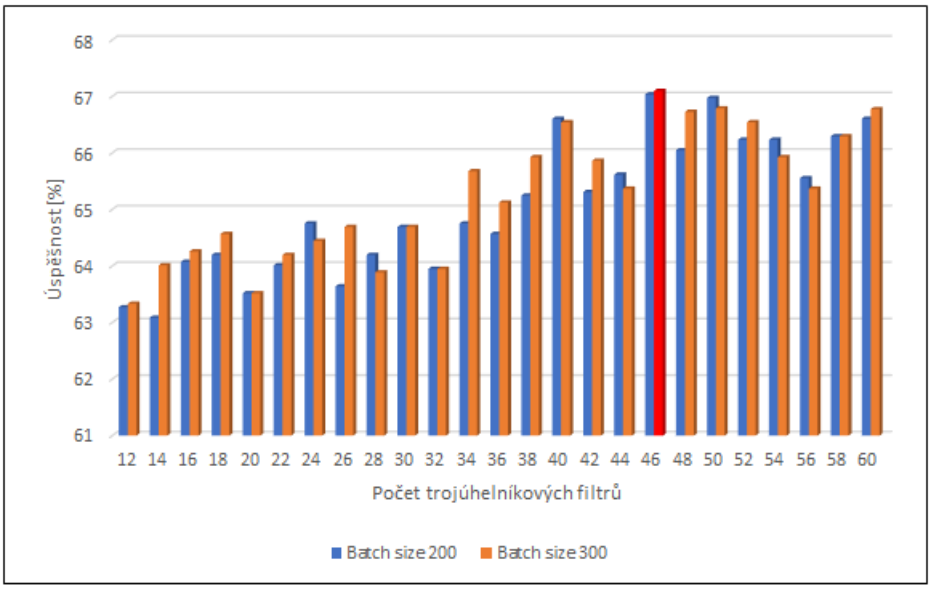
Tabulka 7.6: *Maximální dosažená úspěšnost neuronové sítě v závislosti na počtu DCT koeficientů*

Počet DCT koeficientů	Batch size	Úspěšnost [%]	Počet DCT koeficientů	Batch size	Úspěšnost [%]
4	200	63.58	12	200	64.75
	300	63.33		300	**64.98**
6	200	64.63	14	200	63.27
	300	64.88		300	62.72
8	200	63.40	16	200	64.26
	300	63.64		300	64.51
10	200	62.84	18	200	63.46
	300	62.10		300	63.33

7.7 Počet trojúhelníkových filtrů

V tomto experimentu byla zjišťována úspěšnost výsledného systému při použití blok MFB příznaků v závislosti na počtu MFB koeficientů, neboli trojúhelníkových filtrů. Experimentováno bylo s 12 až 60 filtry s krokem 2. Výsledky experimentu jsou uvedeny v grafu 7.7. Z grafu vyplývá, že se zvyšujícím se počtem trojúhelníkových filtrů vzrůstá i úspěšnost systému s oběma experimentovanými velikostmi *batch size*.

Nejvyšší úspěšnosti dosahuje systém pro 46 filtrů pro obě velikosti parametru *batch size*. Se zvyšujícím se počtem filtrů úspěšnost opět klesá. Je zřejmé, že počet trojúhelníkových filtrů velmi významným způsobem ovlivňuje výslednou úspěšnost systému, jelikož nejnižší dosažená úspěšnost dosáhla hodnoty 63.1 %, činí rozdíl mezi nejlepším a nejhorším výsledkem 4 %. Protože bylo v předchozích experimentech použito pouze 24 filtrů, zvýšila se úspěšnost systému o 2,12 %. Z původních 64.98 % se tedy úspěšnost zvýšila na hodnotu 67.1 %.



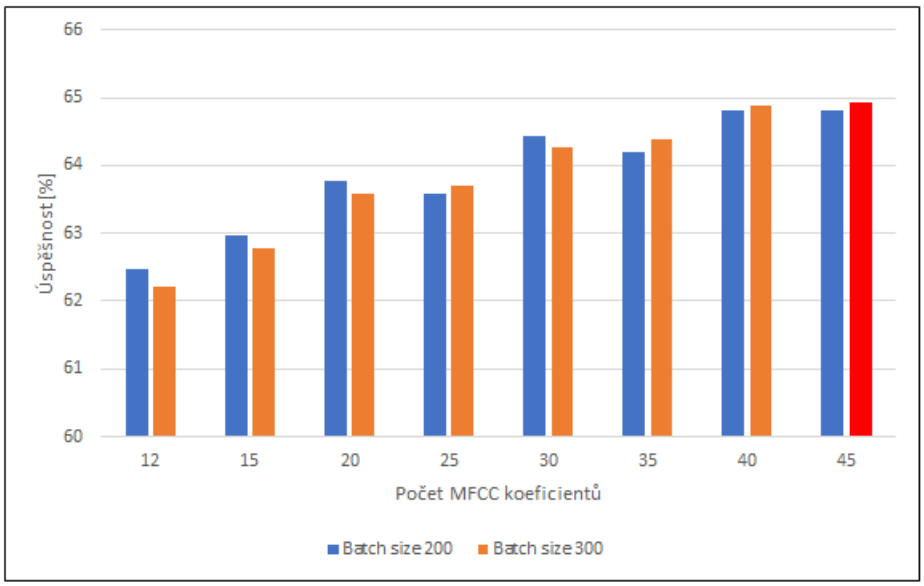
Obrázek 7.7: *Dosažená úspěšnost systému v závislosti na počtu trojúhelníkových filtrů při použití blok MFB koeficientů*

7.8 Využití MFCC, delta a double delta koeficientů

V této kapitole byl proveden experiment s využitím MFCC, delta a double delta koeficientů. Byly v něm využity nahrávky s původní vzorkovací frekvencí 44.1 kHz a 46 trojúhelníkových filtrů, jelikož v experimentu, popsaném v kapitole 7.7, dosáhl systém pro tento počet filtrů nejlepších výsledků. Získání MFCC koeficientů z audio signálu je popsáno v kapitole 3.3, následný výpočet delta a double delta koeficientů je popsaný v kapitole 3.4.

V prvním části tohoto experimentu byla zjišťována úspěšnost systému pro 6 a 12 MFCC koeficientů, jelikož se jedná o výpočet DCT stejně jako v případě blok MFB příznaků, při jejichž použití bylo dosaženo nejvyšší přesnosti právě s těmito počty příznaků. S 6 MFCC koeficienty na rámec byla nejvyšší dosažená úspěšnost 51.60 %, zatímco pro 12 MFCC koeficientů 62.47 %. Rozdíl mezi těmito experimenty byl velmi výrazný, jelikož činil 10.87 %. Vzhledem k tak velkému rozdílu byly provedeny další experimenty pro 15 – 45 MFCC koeficientů s krokem 5. Výsledky je možné vidět v grafu 7.8. Z výsledků je patrné, že úspěšnost se se zvyšujícím se počtem vzrůstala až do 40 MFCC koeficientů na rámec. Další zvýšení počtu DCT příznaků již pro *batch size* velikosti 200 nevedlo ke zvýšení přesnosti klasifikace. Pro *batch size* velikosti 300 se úspěšnost zvýšila jen minimálně a to o 0.06 %. Jednalo se zároveň o nejvyšší dosaženou úspěšnost v tomto experimentu, tedy 64.94 %. Je zřejmé, že počet DCT koeficientů při použití MFCC ovlivňuje výslednou úspěšnost značným způsobem, protože se v závislosti na jejich počtu pohybovala v rozmezí od 51.60 do 64.94 %. Narozdíl od blok MFB příznaků, kde bylo nejvyšší úspěšnosti dosaženo pro 12 DCT koeficientů, se úspěšnost při zvyšování počtu MFCC koeficientů téměř lineárně zvyšovala. Nejvyšší hodnoty dosáhla při 45 DCT koeficientech, tedy když byl jejich počet téměř shodný s počtem trojúhelníkových filtrů. Výše popsané výsledky byly dosaženy s využitím nultého koeficientu. Při jeho zanedbání se úspěšnost se zvyšujícím se počtem MFCC příznaků rovněž zvyšovala, avšak

přesnost klasifikace byla ve všech experimentech průměrně o 1.4 % nižší. Proto jsou v této kapitole uvedeny pouze výsledky s využitým nultým koeficientem.



Obrázek 7.8: *Dosažená úspěšnost systému v závislosti na počtu MFCC koeficientů*

V další části tohoto experimentu bylo zjišťováno, jak ovlivní výslednou úspěšnost systému přidání delta, případně double delta koeficientů. Pro tento experiment byl vybrán nejlepší systém, který je popsaný v předchozím odstavci. Bylo tedy použito 45 MFCC koeficientů s využitím 0. koeficientu. Experimentováno bylo s kontextem 1 – 5 rámců jak pro delta, tak pro double delta koeficienty. Kontext byl nastaven pro obě varianty vždy shodný.

Výsledky jsou uvedeny v tab.7.7. Je zřejmé, že při přidání delta koeficientů k MFCC koeficientům výslednou úspěšnost příliš neovlivnil zvolený kontext rámců, protože rozdíl mezi nejvyšší a nejnižší dosaženou úspěšností v závislosti na použitém kontextu činil 0.61 % a v mnoha případech byla dosažená úspěšnost naprosto shodná. Nejvyšší úspěšnost, 65.49 %, dosáhl systém s kontextem 1 rámec a *batch size* velikost 200. Přidáním delta koeficientů se zmíněným kontextem se tedy úspěšnost systému zvýšila o 0.55 %.

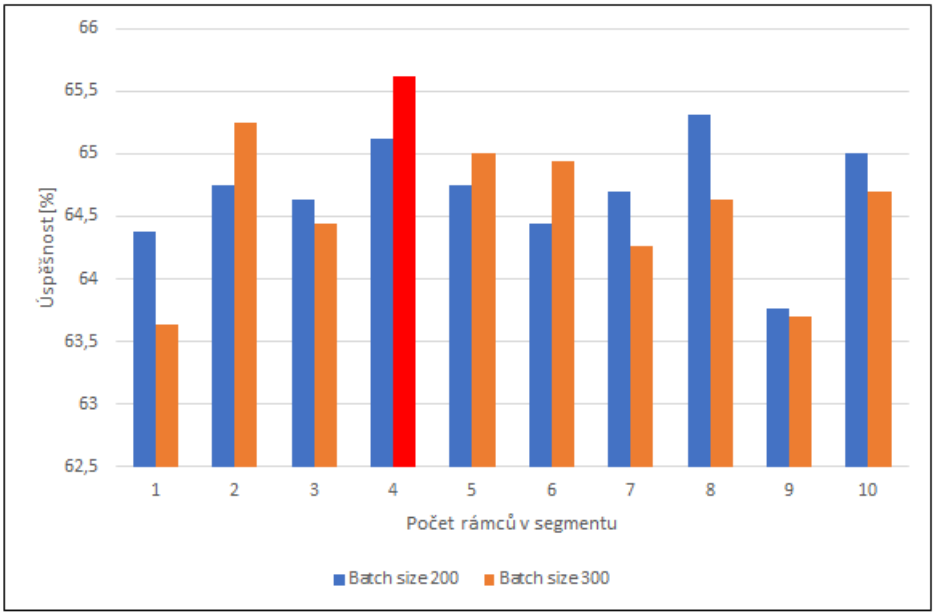
Při přidání double delta koeficientů se výsledná úspěšnost ve všech experimentech lišila o pouhých 0.43 % a v mnoha případech byla rovněž zcela shodná. Zvolený kontext tedy ovlivnil výslednou úspěšnost ještě méně než v případě použití pouze delta koeficientů. Nejlepšího výsledku bylo dosaženo pro kontext (4,4) a *batch size* velikosti 200, kdy byla celková úspěšnost systému 65.31 %, což je však o 0.18 % horší výsledek, než při absenci double delta koeficientů. Jejich přidání k delta koeficientům tedy výslednou úspěšnost mírně zhoršilo.

Tabulka 7.7: *Dosažená úspěšnost neuronové sítě v závislosti na velikosti kontextu delta a double delta koeficientů*

Kontext [rámců]	Batch size	Úspěšnost [%]	Kontext [rámců]	Batch size	Úspěšnost [%]
(1)	200	**65.49**	(1,1)	200	64.94
	300	65.25		300	65.25
(2)	200	65.12	(2,2)	200	64.94
	300	65.12		300	65.00
(3)	200	65.12	(3,3)	200	64.88
	300	65.25		300	65.12
(4)	200	65.25	(4,4)	200	65.31
	300	65.25		300	65.25
(5)	200	65.06	(5,5)	200	65.25
	300	65.43		300	65.25

7.9 Trénování na více rámcích zároveň

Všechny dosavadní experimenty byly prováděny s velikostí segmentu, tedy vstupního vektoru do neuronové sítě, jeden rámec. Základní systém k soutěži DCASE 2017 však využívá pro trénování neuronové sítě segmenty, které se skládají z pěti rámců. Proto byl i v této práci proveden experiment s počtem rámců v jednom segmentu. Byly pro něj využity MFB koeficienty s 46 trojúhelníkovými filtry, které byly naskládány za sebe tak, jak za sebou následují v audio nahrávce. Experiment byl prováděn s 1 až 10 rámci v jednom segmentu.

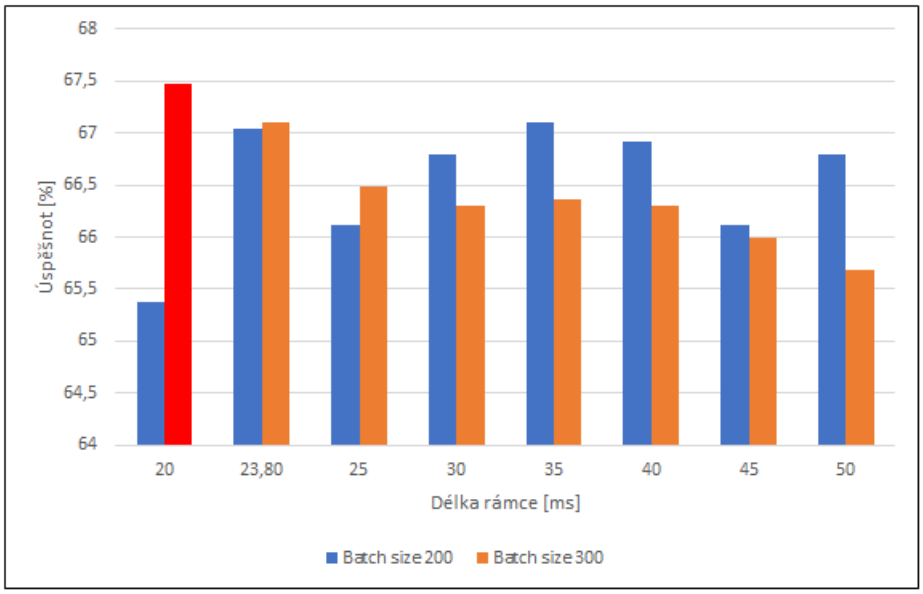


Obrázek 7.9: *Dosažená úspěšnost systému v závislosti na počtu rámců v jednom segmentu*

V grafu 7.9 je znázorněna výsledná úspěšnost systému s oběma experimentovanými velikostmi parametru *batch size*. Je zřejmé, že pro *batch size* velikosti 300 dosáhl systém nejvyšší úspěšnosti při 4 rámcích v jednom segmentu, zatímco pro *batch size* 200 dosáhl systém nejvyšší úspěšnosti pro 8 rámců na segment. Nejvyšší dosažená úspěšnost byla 65.62 %. Z grafu vyplývá, že počet rámců v segmentu ovlivňuje výslednou úspěšnost značným způsobem, jelikož rozdíl mezi nejvyšší a nejnižší úspěšností je 1.54 % pro *batch size* velikosti 200 a pro *batch size* velikosti 300 činí rozdíl dokonce 1.98 %. Nejvyšší dosažená úspěšnost při tomto experimentu však byla o 1.48 % nižší, než v případě použití blok MFB koeficientů.

7.10 Délka rámce

Všechny dosud prováděné experimenty se vzorkovací frekvencí audio nahrávek velikosti 44.1 kHz, byly prováděny s délkou rámce 1050 vzorků, tedy 23.8 ms. Tato délka byla zvolena pro to, aby se pro zpracování využily všechny vzorky audio signálu. V dalším experimentu byla tedy zjišťována úspěšnost systému v závislosti na délce jednoho rámce v rozmezí 20-50 ms s krokem 5 ms.



Obrázek 7.10: *Dosažená úspěšnost systému v závislosti na délce jednoho rámce*

Výsledky experimentu jsou znázorněny v grafu 7.10. Je zřejmé, že závislost úspěšnosti systému na délce jednoho rámce se značně liší pro jednotlivé velikosti parametru *batch size*. Při trénování neuronové sítě s parametrem *batch size* velikosti 300 dosahuje neuronová síť nejvyšší úspěšnosti pro velikost rámce 20 ms, zatímco při použití parametru *batch size* s hodnotou 200 dosahuje systém nejvyšší úspěšnosti pro délku rámce o velikosti 35 ms. Z grafu dále vyplývá, že v prvním případě úspěšnost se zvyšující se délkou rámce postupně klesá, zatímco ve druhém případě se postupně zvyšuje až do již zmíněné hodnoty 35 ms, kde dosahuje maximální hodnoty a po té se snižuje.

Rozdíl mezi systémem s nejvyšší a nejnižší úspěšností činí 2.01 %, délka rámce má tedy na výslednou úspěšnost určitý vliv. Nejvyšší dosažená úspěšnost má hodnotu 67.47 %.

Oproti experimentu, popsanému v kapitole 7.7, kde nejlepší systém dosáhl úspěšnosti 67.1 % se výsledná úspěšnost zvýšila o 0.37 %.

7.11 Fúze systémů

V tomto experimentu byla provedena fúze dvou klasifikátorů. Prvním z nich byl nejlepší systém z experimentu, popsaného v kapitole 7.8. Jedná se o systém, který pro reprezentaci každé třídy využívá 45 MFCC příznaků z každého rámce, ke kterým jsou přidány akcelerační koeficienty s kontextem 1 rámec. Druhý systém, který dosáhl nejvyšší celkové přesnosti klasifikace v experimentu, popsaném v kapitole 7.10, využívá jako vstupní hodnoty do neuronové sítě vektor 552 blok MFB příznaků. V obou případech je použito 46 trojúhelníkových filtrů aplikovaných na audio signál s původní vzorkovací frekvencí 44.1 kHz, který byl rozdělen na rámce o velikost 20 ms. Výstupy obou neuronových sítí byly zprůměrovány technikou *Voting*, která je popsána v kapitole 2.2.

Tabulka 7.8: *Porovnání třídních úspěšností a celkové úspěšnosti nejlepšího systému s blok MFB a MFCC příznaky s fúzí těchto systémů*

Třída	Úspěšnost [%]		
	Systém s MFCC příznaky	Systém s blok MFB příznaky	Fúze obou systémů
autobus	49.1	43.5	44.4
knihovna	40.7	62.0	57.4
kavárna/restaurace	53.7	59.3	59.3
stanice metra	94.4	100.0	100.0
auto	93.5	71.3	75.9
kancelář	73.1	63.0	65.7
centrum města	83.3	92.6	91.7
obytná oblast	76.9	85.2	84.3
lesní cesta	86.1	85.2	85.2
vlak	60.2	66.7	64.6
obchod s potravinami	54.6	61.1	61.1
tramvaj	50.0	70.4	63.6
domov	76.9	78.7	82.4
park	56.5	38.9	40.7
pláž	33.3	34.3	31.5
Celková úspěšnost:	**65.5**	**67.5**	**67.2**

Výsledná třídní úspěšnost každého systému a fúze systémů je uvedena v tab. 7.8. Oproti prvnímu systému s MFCC příznaky přinesla fúze systému zlepšení u 8 klasifikačních tříd. Nejvíce se úspěšnost zvýšila u třídy knihovna a to o 16.7 %. Naopak největší pokles nastal u třídy auto. Celková přesnost fúze systémů klasifikace dosáhla o 1.7 % vyšší hodnoty. V případě srovnání výsledků se 2. systémem byly třídní úspěšnosti velmi podobné, v některých

případech zcela shodné. Celková úspěšnost fúze byla oproti druhému systému o 0.3 % nižší. Z výsledků je zřejmé, že ve všech třech případech bylo největším problémem správně klasifikovat nahrávky pořízené na pláži a v autobuse. Naopak nejlépe klasifikátory rozlišovaly prostředí metra, 2 z nich dokonce s přesností 100 %.

7.12 Experimenty s nejlepším systémem

V tomto experimentu bylo zjišťováno, jaké úspěšnosti dosáhne systém při různém využití obou kanálů stereo signálu pro obě datové sady popsané v kapitole 5. Za tímto účelem byl vybrán nejlepší systém ze všech provedených experimentů popsaných v předchozích kapitolách. Nejvyšší úspěšnosti dosáhl systém pří experimentu popsaném v kapitole 7.10. Tento systém využíval pro reprezentaci vlastností audio nahrávky blok MFB koeficienty získané z průměrné hodnoty obou kanálů se vzorkovací frekvencí 44.1 kHz.

V první části experimentu bylo zjišťováno, jaké úspěšnosti dosáhne tento systém na datové sadě z roku 2016. Výsledky jsou uvedeny v tab. 7.9. Je zřejmé, že zvolený kanál, případně jejich kombinace, vůbec neovlivnily výslednou úspěšnost systému, jelikož bylo ve všech případech dosaženo naprosto shodné úspěšnosti. Výjimku tvoří trénování neuronové sítě pouze na pravém kanálu stereo signálu při nastavení parametru *batch size* na hodnotu 200, kdy byla výsledná úspěšnost oproti ostatním o 0.51 % nižší. V ostatních případech byla dosažená úspěšnost 82.82 %.

Z předchozího odstavce vyplývá, že výběr kanálu audio signálu pro trénování a následnou evaluaci neuronové sítě vůbec neovlivnil celkovou úspěšnost systému. Aby se zjistilo, zda tato skutečnost platí pro obecně jakákoliv data, zaznamenávaná ve stereo kvalitě, byl stejný experiment proveden i na datové sadě z roku 2017. Výsledky jsou rovněž uvedeny v tab. 7.9. Je zřejmé, že úspěšnost se v závislosti na použitém kanálu lišila o více než 4.26 %. Pohybovala se totiž v rozmezí od 63.21 do 63.70 % při trénování neuronové sítě pouze na pravém nebo pouze na levém kanálu, zatímco při použití průměrné hodnoty obou kanálů se pohybovala, v závislosti na parametru *batch size*, od 65.37 do 67.47 %. Na této datové sadě tak výběr kanálu výrazně ovlivnil výslednou úspěšnost systému, narozdíl od předchozího případu.

Tabulka 7.9: *Dosažená úspěšnost neuronové sítě při trénování a evaluaci na jednotlivých kanálech a průměrné hodnotě obou kanálů v závislosti na použité datové sadě*

Datová sada [rok]	Batch size	Úspěšnost [%]		
		levý kanál	pravý kanál	průměrná hodnota obou kanálů
(2017)	200	63.21	63.33	65.37
	300	63.70	63.64	67.47
(2016)	200	82.82	82.31	82.82
	300	82.82	82.82	82.82

V tab. 7.10 je uvedeno porovnání třídních a celkových úspěšností základních systémů soutěže DCASE a nejlepšího systému v této práci na evaluační sadě odpovídající roku, ke kterému základní systém náleží. Na datové sadě z roku 2016 činila dosažená přesnost klasifikace nejlepšího systému 82.8 %. Oproti základnímu GMM systému se tak celková

úspěšnost zvýšila o 5.9 %, jelikož ten dosáhl úspěšnosti pouze 76.9%[1]. Základní systém z roku 2017, založený na neuronové síti, dosáhl úspěšnosti 61 %[2]. Na datové sadě z roku 2017 tak činilo zlepšení 6.5 %, kdy se celková úspěšnost zvýšila na 67.5 %.

Tabulka 7.10: *Porovnání třídních úspěšností a celkové úspěšnosti nejlepšího systému v této práci a základních systémů soutěže DCASE na odpovídající evaluační sadě*

Třída	Úspěšnost [%]			
	Datová sada 2016		Datová sada 2017	
	Základní systém 2016	Nejlepší systém	Základní systém 2017	Nejlepší systém
autobus	88.5	96.2	38.9	43.5
knihovna	26.9	53.8	30.6	62.0
kavárna/restaurace	69.2	61.5	43.5	59.3
stanice metra	100.0	76.9	93.5	100.0
auto	96.2	100.0	64.8	71.3
kancelář	96.2	100.0	73.1	63.0
centrum města	80.8	84.6	79.6	92.6
obytná oblast	88.5	65.4	77.8	85.2
lesní cesta	65.4	100.0	85.2	85.2
vlak	30.8	46.2	72.2	66.7
obchod s potravinami	88.5	84.6	49.1	61.1
tramvaj	92.3	100.0	57.4	70.4
domov	92.3	92.3	76.9	78.7
park	53.8	92.3	32.4	38.9
pláž	84.6	88.5	40.7	34.3
Celková úspěšnost:	**76.9**	**82.8**	**61.0**	**67.5**

V tab. 7.11 jsou uvedeny parametry základního systému soutěže DCASE z roku 2017 a nejlepšího systému v této práci. Z tabulky vyplývá, že oba systémy využívají ze stereo audio signálu pro každou nahrávku průměrnou hodnotu z levého a pravého kanálu, přičemž není použita žádná normalizace jednotlivých vzorků. Základní systém využívá MFB koeficienty s délkou rámce 40 ms, zatímco vylepšený systém využívá poloviční velikost rámců, tedy 20 ms, jelikož tak bylo při použití blok MFB koeficientů dosaženo nejlepších výsledků, jak je popsáno v kapitole 7.10. U vylepšeného systému byl rovněž použit větší počet MFB filtrů. Dalším rozdílem je délka vektoru vlastností audio nahrávek. Základní systém používá vektor, který reprezentuje MFB koeficienty, z pěti po sobě jdoucích rámců. Vylepšený systém využívá vektor o délce 552 hodnot, reprezentujících jeden rámec, avšak s kontextem 10 předchozích a 10 následujících rámců, jelikož využívá blok MFB koeficienty, jejichž získání z MFB koeficientů je popsáno v kapitole 3.2. Oba systémy využívají optimalizátor

[1] Úspěšnost základního systému mi byla poskytnuta Matúšem Dobrotkou, který se ve své práci zabýval GMM systémem pro klasifikaci audio nahrávek

[2] zdroj: https://www.cs.tut.fi/sgn/arg/dcase2017/documents/dcase-2017-challenge-paper.pdf

Adam, ovšem vylepšený systém se 100x menším parametrem *learning rate*. Je také trénován na poloviční počtu iterací. Parametr *batch size* je naopak nastaven na vyšší hodnotu.

Tabulka 7.11: *Porovnání parametrů základního systému soutěže DCASE 2017 a nejlepšího systému v této práci*

Parametr	Baseline systém	Nejlepší systém
Audio signál		
Zpracování kanálů	průměrná hodnota obou kanálů	průměrná hodnota obou kanálů
Normalizace	žádná	žádná
Reprezentace vlastností audio nahrávek		
Typ	MFB koeficienty	blok MFB koeficienty
Délka rámce	40 ms	20 ms
Překrývání	20 ms	12 ms
Počet filtrů	40	46
Vektor vlastností audio nahrávek		
Kontext	5 za sebou jdoucí rámců	10 rámců doleva/doprava
Délka	200	552
Nastavení neuronové sítě		
Optimalizátor	Adam	Adam
Learning rate	$1 * 10^{-3}$	$1 * 10^{-5}$
Počet iterací	200	100
Batch size	256	300

Kapitola 8

Závěr

V této práci byly prováděny experimenty s vícevrstvou hustě propojenou neuronovou sítí. Z výsledků, popsaných v předchozí kapitole, vyplývá, že při použití kratších audio nahrávek ve stereo kvalitě, dosahuje neuronová síť nejvyšší přesnosti klasifikace při trénování na průměrné hodnotě obou kanálů. Čím je však jejich délka vyšší, tím je závislost celkové úspěšnosti systému na použitém kanálu nižší. Kombinace obou kanálů a jejich průměrné hodnoty se ukázaly jako nevhodný způsob pro rozšíření trénovací sady, jelikož vedly k mírně horším výsledkům. Výsledky rovněž ukázaly, že nejvyšší přesnosti klasifikace dosahuje tento typ neuronových sítí bez normalizace vstupních dat.

Experimentováním s parametrem learning rate se zjistilo, že nejlepších výsledků dosahuje neuronová síť při jeho optimální hodnotě. Velmi malá hodnota způsobí, že se neuronová síť nezačne vůbec učit, jelikož se ztrátová funkce během učení nesnižuje. Vysoká hodnota naopak zapříčiní, že váhy neuronů jsou upravovány s velkým krokem a neuronová síť se tak naučí pouze poslední trénovací data. Dalším důležitým poznatkem je, že snižování rychlosti učení v průběhu trénování vede k mírně lepším a stabilnějším výsledkům než při použití jeho fixní hodnoty.

Při reprezentaci vlastností audio nahrávek pomocí Mel-filter bank dosahovala neuronová síť nejvyšší přesnosti klasifikace při ponechání původní vzorkovací frekvence audio signálu. Převzorkování dat na nižší frekvenci vedlo ke snížení úspěšnosti systému. Naopak lepších výsledků bylo dosaženo při větším počtu rámců v jednom segmentu. Z provedených experimentů je dále patrné, že při blokovém zpracování Mel-filter bank příznaků celkovou úspěšnost systému výrazně ovlivňoval použitý kontext rámců i počet filtrů, které byly aplikovány na jeden rámec.

Při využití MFCC příznaků bylo dosaženo mírně horších výsledků než v případě použití blok MFB koeficientů. Přesnost klasifikace velmi výrazně ovlivňoval jejich počet. Úspěšnost systému se zvyšujícím se počtem MFCC příznaků, získaných z jednoho rámce audio signálu, téměř lineárně zvyšovala, narozdíl od blok MFB příznaků, kde zvyšování počtu DCT výstupů k lepším výsledkům nevedlo. Nejvyšší úspěšnosti dosáhl systém v situaci, kdy se počet MFCC příznaků rovnal počtu trojúhelníkových filtrů. Experimenty dále ukázaly, že přidání akceleračních koeficientů výslednou přesnost klasifikace příliš neovlivňuje.

Ke zlepšení úspěšnosti základního systému, který měl při prvním experimentu nejvyšší přesnost klasifikace 58.77 % při trénování na průměrné hodnotě obou kanálů, mi nejvíce pomohlo využití všech vzorků audio signálu, kdy se výsledná úspěšnost zvýšila o 6 %. Další zlepšení bylo ovlivněno reprezentací vlastností audio nahrávek pomocí blok MFB příznaků, zejména počtem trojúhelníkových filtrů. které zvýšily celkovou úspěšnost systému o 2.1 %. Drobného zlepšení na konečných 67.5 % bylo dosaženo změnou velikosti jednoho rámce.

Tyto kroky přinesly oproti základnímu systému soutěže DCASE vyšší přesnost klasifikace audio nahrávek o 6.5 %. Ze 41 týmu, které se soutěže zúčastnily by se tak tento klasifikátor umístil na 12. místě. Pro další zlepšení přesnosti predikce místa pořízení audio nahrávky by bylo vhodné vyzkoušet jiné typy neuronových sítí, zejména konvoluční a rekurentní neuronové sítě a využít při jejich návrhu poznatky získané v této práci.

Literatura

[1] Brownlee, J.: Using Learning Rate Schedules for Deep Learning Models in Python with Keras. online, Červen 2016.
URL https://machinelearningmastery.com/using-learning-rate-schedules-deep-learning-models-python-keras/

[2] Diment, A.; Mesaros, A.; Heittola, T.: Detection of rare sound events - task 2. online.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-rare-sound-event-detection

[3] Elizalde, B.; Raj, B.; Vincent, E.: Large-scale weakly supervised sound event detection for smart cars - task 4. online.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-large-scale-sound-event-detection

[4] Fayek, H.: Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between. online, Duben 2016.
URL http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[5] Fér, R.; Matějka, P.; Grézl, F.; aj.: Multilingually Trained Bottleneck Features in Spoken Language Recognition. *Computer Speech and Language*, ročník 2017, č. 46, 2017: s. 252–267, ISSN 0885-2308.
URL http://www.fit.vutbr.cz/research/view_pub.php?id=11518

[6] Han, Y.; Park, J.; Lee, K.: Convolutional neural networks with binaural representations and backgroud substraction for acoustic scene classification. Technická zpráva.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Han_207.pdf

[7] Han, Y.; Park, J.; Lee2, K.: Classifying short acoustic scenes with I-vectors and CNNs: Challenges and optimisations for the 2017 DCASE ASC task. Technická zpráva.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Lehner_142.pdf

[8] Hyde, R.; Ghaffarzadegan, S.; Feng, Z.; aj.: Buet bosch consortium (B2C) acoustic scene classification systems for DCASE 2017 challenge. Technická zpráva.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Hasan_167.pdf

[9] Lyons, J.: Mel-frequency cepstral coefficients (MFCCs) tutorial. online.
URL http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

[10] Mesaros, A.; Heittola, T.: Acoustic scene classification - task description. online.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification

[11] Mesaros, A.; Heittola, T.: Sound event detection in real life audio - task 3. online.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-sound-event-detection-in-real-life-audio

[12] Meško, D.: Normalizace dat pro neuronovou síť GAME. 2008.

[13] Mun, S.; Park, S.; Han, D. K.; aj.: Generative adversarial network based acoustic scene training set augmentation and selection using SVM hyper-plane. Technická zpráva.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Mun_213.pdf

[14] Müller, P. D. M.: Harmonic-Percussive Source Separation. online.
URL https://www.audiolabs-erlangen.de/content/05-fau/professor/00-mueller/02-teaching/2017w_mpa/LabCourse_HPSS.pdf

[15] National-Instruments-community: Understanding FFTs and Windowing. online.
URL http://download.ni.com/evaluation/pxi/Understanding%20FFTs%20and%20Windowing.pdf

[16] Nielsen, M.: How the backpropagation algorithm works. online.
URL http://neuralnetworksanddeeplearning.com/chap2.html

[17] Nielsen, M.: Using neural nets to recognize handwritten digits. online.
URL http://neuralnetworksanddeeplearning.com/chap1.html

[18] Oinkina; Hakyll: Understanding LSTM Networks. online, Srpen 2015.
URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[19] Opensource: Adam. online.
URL https://keras.io/optimizers/#adam

[20] Opensource: Usage of loss functions. online.
URL https://keras.io/losses/#usage-of-loss-functions

[21] Raschka, S.: Linear Discriminant Analysis. online.
URL http://sebastianraschka.com/Articles/2014_python_lda.html

[22] Raschka, S.: About Feature Scaling and Normalization. online, Červenec 2014.
URL http://sebastianraschka.com/Articles/2014_about_feature_scaling.html#standardizing-and-normalizing---how-it-can-be-done-using-scikit-learn

[23] Srivastava, N.; Hinton, G.; Krizhevsky, A.; aj.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. online, Červen 2014.
URL https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf

[24] Virtane, T.: Challenge rules. online.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/rules

[25] Walia, A. S.: Activation functions and it's types – Which is better? online.
URL https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f

[26] Weiping, Z.; Jiantao, Y.; Xiaotao, X.; aj.: Acoustic scene classification using deep convolutional neural network and multiple spectograms fusion. Technická zpráva.
URL http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Xing_158.pdf

[27] Wikipedia: Binaural fusion. online.
URL https://en.wikipedia.org/wiki/Binaural_fusion

[28] Wikipedia: Cent (music). online.
URL https://cs.wikipedia.org/wiki/Cent_(hudba)

[29] Wikipedia: Linear separability. online.
URL https://en.wikipedia.org/wiki/Linear_separability

Příloha A

Spuštění programu

Program lze spustit s několika povinnými a několika volitelnými parametry. Pokud není jakýkoliv parametr zadán, použije se jeho výchozí hodnota, která je specifikovaná v modulu *Constants.py*:

- `-e --epochs` – Počet iterací trénování neuronové sítě.
- `-b --batch_size` – Velikost parametru batch size.
- `-td --train_data` – Relativní nebo absolutní cesta k textovému souboru, kde jsou uloženy relativní nebo absolutní cesty k jednotlivým nahrávkám odpovídající zvolené trénovací sadě a místa, odkud byly nahrávky pořízeny.
- `-ed --eval_data` – Relativní nebo absolutní cesta k textovému souboru, kde jsou uloženy relativní nebo absolutní cesty k jednotlivým nahrávkám odpovídající zvolené evaluační sadě a místa, odkud byly nahrávky pořízeny.
- `-out_path --output_path` – Absolutní nebo relativní cesta k adresáři, kam bude uložena natrénovaná neuronová síť (její topologie a váhy neuronů) a dosažená úspěšnost neuronové sítě.
- `-out_file --output_file` - Název souboru, kam bude uložen výstup programu.
- `-s_mod_file --save_model_file` - Název souboru, kam bude uložena topologie neuronové sítě.
- `-s_w_file --save_weights_file` - Název souboru, kam budou uloženy váhy jednotlivých neuronů.
- `-l --load` - Absolutní nebo relativní cesta k adresáři, kde je uložena natrénovaná neuronová síť.
- `-l_mod_file --l_model_file` – Název souboru, ze kterého bude načtena topologie neuronové sítě. Pokud není parametr `-l` nebo `--load` zadán, nemá tento parametr žádný význam.
- `-l_w_file --l_weights_file` - Název souboru, ze kterého budou načteny váhy jednotlivých neuronů. Pokud není parametr `-l` nebo `--load` zadán, nemá tento parametr žádný význam.

- `-lr --learning_rate` - Hodnota parametru *learning rate*, který bude použit při trénování neuronové sítě.
- `-chnk --frames_in_chunk` - Počet rámců v jednom segmentu, výchozí hodnota je 1.
- `-cntxt --block_context` - Počet rámců pro výpočet blok MFB koeficientů.
- `-dctb --dct_basis` - Tento parametr má 2 funkce v závislosti na hodnotě parametru `-fea` nebo `--features`. V případě, že jeho hodnota je *MFCC* jedná se o počet těchto příznaků získaných z jednoho rámce. Pokud je jeho hodnota 'BLOK_MFB' jedná se o počet DCT koeficientů získaných z výstupů pro daný filtr v kontextu rámců specifikovaných předchozím parametrem.
- `-mfb --mfb_filters` - Počet MFB filtrů, která budou použity pro výpočet MFB koeficientů.
- `-dlts --deltas` - Tento parametr určuje kontext pro výpočet delta, double delta atd. koeficientů, které budou přidány na konec vektoru MFCC příznaků odpovídajících jednomu rámci. Pokud je hodnota tohoto parametru `""`(prázdný řetězec), nebudou vypočítány žádné delta koeficienty.
- `-window --window_size` - Počet vzorků audio nahrávky odpovídající velikosti jednoho rámce.
- `-noverlap --window_noverlap` - Počet vzorků audio nahrávky odpovídající velikosti překrývání rámců. Pokud není tento parametr zadán, je hodnota překrývání rámců rovna 60 % z hodnoty předchozího parametru.
- `-fea --features` - tento parametr může nabývat hodnot *MFB*, *BLOCK_MFB* a *MFCC*. Určuje, jak budou reprezentovány vlastnosti jednotlivých nahrávek.

Příloha B

Popis programu

Pro implementaci systému byl v této práci využitý framework *Keras* pro skriptovací jazyk Python verze 2.7, jelikož je pro implementaci neuronových sítí velmi často využívaný a také proto, že systém, vytvořený v této práci, navazuje na základní systém soutěže DCASE, který tento framework rovněž využívá. Program je rozdělen do několika modulů:

- `Constants.py` – modul, který obsahuje defaultní hodnoty všech parametrů, které jsou využity v programu.
- `Load_data.py` – třída, která zabezpečuje načtení evaluačních a trénovacích dat
- `Main.py` – hlavní modul, ze kterého je celý program spouštěn. Nejprve jsou zpracovány argumenty, zadané na příkazové řádce. Po té jsou vytvořeny potřebné objekty a následně je spuštěno trénování a evaluace neuronové sítě se zadanými parametry.
- `NN_model.py` – třída, která slouží pro veškerou práci s neuronovou sítí. Zabezpečuje její vytvoření/načtení, trénování, evaluaci a uložení.
- `Processing_record.py` – třída, která slouží pro výpočet MFB, blok MFB nebo MFCC koeficientů. K MFCC koeficientům je rovněž možné přidat delta koeficienty. Všechny metody v této třídě mi byly poskytnuty vedoucím mé práce. Jejich autory jsou *Anna Silnova*, *Pavel Matějka*, *Oldřich Plchot*, *František Grézl*

Program je psán objektově, celkem obsahuje 3 druhy tříd:

- *NN_model* - Třída, která slouží pro práci s modelem neuronové sítě. Operace, které umožňuje z modelem provádět, jsou následující:
  - `create_NN_model(number_of_features)` – Vytvoří model neuronové sítě který je v této metodě specifikován.
  - `compile_model(learning_rate)` - Zkompiluje model neuronové sítě se zadaným parametrem *learning rate*
  - `fit_model(epochs=2, batch_size=200, verbose=1, calculate_RN=False, calculate_G` - Natrénuje neuronovou síť na zadaných trénovacích datech pomocí metody `fit` z knihovny Keras.
  - `fit_generator(epochs=2, verbose=1, calculate_GN=False, calculate_RN=False)` - Natrénuje neuronovou síť na datech, které vrátí generátor z třídy `Load_data` pomocí metody `fit_generator` z knihovny Keras.

  - `load_NN_model(path_to_model)` - Načte model neuronové sítě z adresáře, zadaného jediným parametrem této metody. Pro korektní načtení neuronové sítě jsou zapotřebí 2 soubory. První soubor, ve formátu json, specifikuje topologii neuronové sítě, zatímco druhý soubor obsahuje váhy jednotlivých neuronů. Názvy obou načítaných souborů jsou specifikovány při spuštění programu.
  - `save_model(path_to_save_model)` - Uloží aktuální model neuronové sítě do adresáře specifikovaného parametrem `path_to_save_model`. Existující soubory budou přepsány. Názvy obou ukládaných souborů jsou rovněž specifikované při spuštění programu. Obsah obou souborů je popsán v předchozí bodě.
  - `evaluate_model(path_to_output_file, calculate_GN=False, calculate_RN=False)` - Provede evaluaci neuronové sítě na evaluační datové sadě zadané při spuštění programu. Výstupní úspěšnost pro jednotlivé třídy a celková úspěšnost systému je vypsána do terminálu a uložena do souboru.

- *Load_data* - Tato třída, jak již název napovídá, zajišťuje načtení a zpracování dat. Obsahuje metody pro načtení vstupního textového souboru, jeho zpracování a načtení jednotlivých nahrávek. Pro získání MFCC, MFB, delta a double delta koeficientů z každé nahrávky využívá objekt `Process_record`. Dále zajišťuje globální/lokální normalizaci dat a obsahuje rovněž 2 generátory trénovacích a evaluačních dat.
  - `get_record_features(record)` - Vrátí vlastnosti vstupní audio nahrávky (MFB, blok MFB, MFCC, delta koeficienty).
  - `calc_mean_and_dev_vec(features)` - Vrátí vektor středních hodnot a vektor směrodatných odchylek pro odpovídající si koeficienty.
  - `fea_normalization(features)` - Vrátí normalizovaná vstupní data.
  - `categorical_record_labels(place_name)` - Vrátí 2D pole reprezentující místo odkud pochází audio nahrávka, přičemž počet prvků odpovídá počtu částí, do kterých je rozdělena matice koeficientů reprezentující danou nahrávku.
  - `parse_and_random_meta_data(fname)` - Načte data ze vstupního souboru a vrátí pole cest k jednotlivým audio nahrávkám a pole míst, odkud jednotlivé nahrávky pocházejí.
  - `loading_data(calculate_GN=False, calculate_RN=False)` - metoda, která načte vstupní data a v závislosti na pravdivostní hodnotě obou parametrů provede normalizaci nad celou evaluační sadou nebo nad daty z jedné audio nahrávky.
  - `get_train_data_and_labels(calculate_RN=False, calculate_GN=False)` - Vrátí načtená, případně normalizovaná, trénovací data a k nim odpovídající pole míst, odkud jednotlivá data pocházejí.
  - `get_number_of_eval_records()` - Vrátí počet nahrávek v evaluační sadě.
  - `get_eval_labels()` - Vrátí pole reprezentující místa, odkud pocházejí audio nahrávky z evaluační sady.
  - `get_chunks_per_record()` - Vrátí počet částí, na které je rozdělena matice koeficientů reprezentující jednu audio nahrávku pro trénování neuronové sítě.
  - `get_number_of_features()` - Vrátí počet koeficientů, které jsou vypočítány z jednoho rámce audio nahrávky.

  - `generator_eval(calculate_RN=False, calculate_GN=False)` - Generátor, který generuje data pro evaluaci neuronové sítě. Při každém volání vrací v závislosti na nastavení matici MFB, blok MFB, MFCC s případnými delta a double delta koeficienty pro každou evaluační nahrávku.
  - `generator_train(calculate_RN=False, calculate_GN=False)` - Generátor, který po každé periodě náhodně seřadí vstupní data. Při každém volání vrací pole o n hodnotách v závislosti na velikosti parametru *batch size*.

- *Processing_record* - Třída, která je využívána pro načtení vstupního souboru, který obsahuje cesty k audio nahrávkám a místa odkud tyto nahrávky pocházejí. Dále obsahuje metody pro výpočet MFB, blok MFB, MFCC koeficientů z každé nahrávky.
  - `mel(x)` - Metoda, která slouží pro převedení všech hodnot v poli `x` z hertzové stupnice do stupnice mel.
  - `mel_inv(x)` - Metoda, která slouží pro převod všech hodnot v poli `x`.
  - `mel_fbank_mx(winlen_nfft, fs, NUMCHANS=20, LOFREQ=0.0, HIFREQ=None)` - Metoda, která slouží pro výpočet Mel-filter bank v závislosti na počtu vzorků v jednom rámci a na počtu trojúhelníkových filtrů.
  - `signal2fbank(signal)` - Metoda, která ze vstupního signálu vypočítá velikosti výstupů pro MFB filtry, tedy MFB koeficienty.
  - `load_txt_file(fname)` - Vrátí pole, kde jedna položka představuje jeden řádek vstupního souboru.
  - `read_signal(file_name)` - Načte audio signál pro jednu nahrávku.
  - `framing(window, shift=1)` - Rozdělí načtenou audio nahrávku do jednotlivých rámců.
  - `fbank_htk(x,window,noverlap,fbank_mx)` - Vrátí matici MFB koeficientů pro audio nahrávku. Volání této metody musí předcházet volání metody `mel_fbank_mx`, jejíž výstup musí být této metodě předán jako čtvrtý parametr.
  - `mfcc_htk(x, window, noverlap, fbank_mx, nfft=None, _0="last", _E=None, NUMCEPS=12, USEPOWER=False, RAWENERGY=True, PREEMCOEF=0.97, CEPLIFTER=22.0, ZMEANSOURCE=False, ENORMALISE=True, ESCALE=0.1, SILFLOOR=50.0, USEHAMMING=True)` - Metoda vrátí pro vstupní nahrávku matici MFCC koeficientů. Stejně jako v předchozím případě musí i této metodě předcházet volání metody `mel_vbank_mx`.
  - `add_deriv(fea, winlens=(2,2)` - K matici vstupních koeficientů přidá delta, double delta atd. koeficienty.
  - `preprocess_nn_input(X, left_ctx=5, right_ctx=5, dct_basis=6)` - Pro vstupní matici MFB koeficientů vrátí matici blok MFB koeficientů, viz kapitola 3.2.
  - `preprocess_framing(a,window,shift=1)` - Rozdělí matici MFB koeficientů do jednotlivých rámců v závislosti na zadaném kontextu.
  - `dct_basis(nbasis,length)` - Vypočítá DCT a výsledek vrátí jako matici o velikosti dané parametry `nbasis` a `length`.