TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Akustický detektor rozbití skla

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 1802T007 – Informační technologie

Autor práce: **Bc. Jaroslav Čmejla**
Vedoucí práce: doc. Ing. Zbyněk Koldovský, Ph.D.

Liberec 2016

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Acoustic Glass Break Detector

Master thesis

Study programme: N2612 – Electrotechnology and informatics
Study branch: 1802T007 – Information technology

Author: **Bc. Jaroslav Čmejla**
Supervisor: doc. Ing. Zbyněk Koldovský, Ph.D.

Liberec 2016

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2015/2016**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jaroslav Čmejla**
Osobní číslo:	**M13000176**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Informační technologie**
Název tématu:	**Akustický detektor rozbití skla**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Analyzujte vzorky záznamů rozbití různých druhů skla. Použijte vzorky, které vám poskytla firma Jablotron, a vzorky, které najdete na internetu. Pokuste se pořídit i vlastní záznamy.
2. Navrhněte a optimalizujte způsob detekce zvukových událostí, které odpovídají rozbití určitého druhu skla. Rozhodněte jaký druh příznakových vektorů použít a jaký použít model pro detekci.
3. Navrženou metodu testujte na reálných datech. Vyhodnoťte úspěšnost metody a její výpočetní náročnost.

Rozsah grafických prací:	**Dle potřeby dokumentace**
Rozsah pracovní zprávy:	**cca 40 až 50 stran**
Forma zpracování diplomové práce:	**tištěná/elektronická**

Seznam odborné literatury:

[1] **B. Porat, "A Course in Digital Signal Processing", John Wiley & Sons, 1997.**

[2] **Řeč a počítač: principy hlasové komunikace, úlohy, metody a aplikace, editoři J. Nouza, Z. Koldovský a R. Vích, ISBN 978-80-7372-548-8, Technická univerzita v Liberci, 2009.**

Vedoucí diplomové práce:	**doc. Ing. Zbyněk Koldovský, Ph.D.** Ústav informačních technologií a elektroniky
Konzultant diplomové práce:	**Ing. Jaromír Šubčík** Jablotron Alarms a.s.
Datum zadání diplomové práce:	**14. září 2015**
Termín odevzdání diplomové práce:	**16. května 2016**

prof. Ing. Václav Kopecký, CSc.
děkan

Technická univerzita v Liberci
L.S.
- 14 -

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 14. září 2015

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 16.5.2016

Podpis:

Abstrakt

Tato práce se zabývá problematikou rozpoznávání akustické události – rozbití skla. Je v ní vysvětleno několik stěžejních pojmů, které jsou nezbytné pro zpracování dané problematiky. Hlavním výstupem je schéma rozpoznávače rozbití skla. V návrhu rozpoznávače je využito MFCC příznaků a neuronových sítí. V práci jsou mimo jiné formulovány jednotlivé kroky budování rozpoznávače, dále možnosti nastavení parametrů jednotlivých částí a jejich vlivů na úspěšnost detekce. V závěru jsou uvedeny ukázky práce detektoru a srovnání výsledků trénování neuronových sítí.

Klíčová slova: detekce akustické události, rozbití skla, MFCC, klasifikace, neuronové sítě

Abstract

This thesis deals with the issue of recognition of acoustic events, namely breaking of glass. It explains several essential terms, which are necessary for understanding this issue. The main output of this thesis is a diagram of a glass breaking detection algorithm. The proposed detection mechanism employs MFCC features, as well as neural networks. The thesis describes each step of building the detection algorithm, possible settings and their impact on detection capabilities. Conclusion provides the demonstration of detector function and the comparison of results given by various neural network trainings.

Key words: recognition of acoustic events, glass breakage, MFCC, classification, neural networks

Poděkování

Děkuji vedoucímu práce doc. Ing. Zbyňku Koldovskému, Ph.D. za neocenitelné rady a pomoc při tvorbě této práce.

Dále bych rád poděkoval panu Ing. Jaromíru Šubčíkovi, šéfkonstruktérovi, z firmy Jablotron Alarms a.s. za věcné připomínky a za trpělivost.

V neposlední řadě bych rád poděkoval své rodině, své přítelkyni a její rodině za podporu a pomoc při tvorbě práce.

Obsah

Seznam obrázků 9
Seznam zkratek 10

Úvod **11**

1 Teoretická část **14**
1.1 Signál 14
1.2 Digitalizace 15
1.3 Segmentace signálu 16
1.4 Analýza signálu 16
1.4.1 Časová posloupnost 16
1.4.2 Frekvenční analýza 16
1.5 Příznaky, příznakové vektory, parametrizace 18
1.6 Klasifikace 20
1.7 Neurony a neuronové sítě 20
1.8 Umělé neuronové sítě 21
1.8.1 Matematický model neuronu 22
1.8.2 Skládání neuronů do neuronových sítí 24
1.8.3 Učení (trénování) neuronových sítí 26

2 Praktická část **31**
2.1 Zdrojová data 31
2.1.1 RIR reprodukce nahrávek 32
2.1.2 Předzpracování 33
2.1.3 Rozdělení do datových množin 33
2.2 Analýza zdrojových nahrávek 34
2.3 Návrh detektoru 35
2.3.1 Klasifikace dynamického průběhu energie 36
2.3.2 Klasifikace spektrálního rozložení 37
2.3.3 Architektura detektoru 37
2.4 Trénování 38
2.4.1 Vytvoření datových množin 38
2.4.2 Vliv nastavení parametrů neuronových sítí na výsledek učení . 40
2.4.3 Vliv nastavení parametrů příznakových vektorů na výsledek učení 42
2.5 Testování detektoru 43

2.6 Zhodnocení výpočetní náročnosti 45

Závěr **47**

Reference **49**

Obsah přiloženého CD **51**

Seznam obrázků

1.1 Signál . . . 14
1.2 Kvantování . . . 15
1.3 Segmentace . . . 16
1.4 Melovská banka filtrů . . . 19
1.5 Parametrizace . . . 19
1.6 Schéma umělého neuronu . . . 22
1.7 Sigmoida . . . 24
1.8 Skoková funkce . . . 24
1.9 Hyperbolický tangens . . . 24
1.10 Gaussova funkce . . . 24
1.11 Dopředná neuronová síť . . . 25

2.1 Spektrogram . . . 34
2.2 Vývoj energie . . . 35
2.3 Detektor – parametrizace . . . 36
2.4 Detektor – klasifikace vývoje energie . . . 37
2.5 Detektor – klasifikace spektrálního rozložení . . . 37
2.6 Trénování – spektrum – bez skryté vrstvy . . . 40
2.7 Trénování – spektrum – (64+64) neurony . . . 41
2.8 Trénování – spektrum – (8+4) neurony . . . 41
2.9 Trénování – energie – 16 neuronů . . . 42
2.10 Upravené rozložení banky filtrů. . . . 43
2.11 Průběhy trénování pro upravené banky filtrů. . . . 43
2.12 Testování detektoru – výstup pravděpodobností . . . 44
2.13 Testování detektoru – demonstrace . . . 44

Seznam zkratek

DCT	Diskrétní Kosínová transformace
DFT	Diskrétní Fourierova transformace
DNN	Deep Neural networks
FT	Fourierova transformace
GMM	Gaussian mixture model
HMM	Hidden Markov model
IFT	Inverzní Fourierova transformace
MFCC	Mel-frequency cepstrum coefficients
NN	Neural Networks
RIR	Room Impulse Response
RPROP	Resilient Propagation

Úvod

Úlohy rozpoznávání zvukových jevů výpočetními systémy jsou velice oblíbené a žádané. Typickou úlohou může být například rozpoznávání řečových signálů. Jiným příkladem může být rozpoznávání akustických anomálií spalovacích motorů nebo také systémy určené pro detekci požadované zvukové události např. zvuk vniknutí cizí osoby do objektu.

Všechny vyjmenované příklady jsou založené na zpracovávání a rozpoznávání akustických signálů, ale každý z nich má odlišný charakter. Pro tuto rozmanitost je nalezení řešení těchto úloh málokdy jednoduché a přímočaré. Jejich úspěšné řešení je předmětem dlouhodobého testování a hledání všemožných nástrojů využívajících nových nebo starých metod. Při řešení se často využívají technologie, které byly použity již v jiných úspěšně řešených úlohách.

Motivace

Motivací pro vznik této práce bylo zadání sestavené firmou **Jablotron Alarms a.s.** Zadáním je vytvoření akustického detektoru rozbití skla.

Firma již dodává funkční akustické rozpoznávače rozbití skel. Stávající řešení pracuje na principu detekce určitých frekvencí obsažených ve zvuku rozbití skla. Slabinou těchto rozpoznávačů může být snížena úspěšnost detekce v případě změny typu či úpravy skla (lepená fólie, tvrzení, …). Takto upravená skla mají při rozbití odlišnou charakteristiku zvuku a tradiční rozpoznávače mohou při detekování selhávat. Zadání bylo vytvořeno, aby se nalezl nový způsob, jak takové rozbití skel rozpoznávat. Návrh by měl být postaven tak, aby bylo možné rozpoznávací jádro případně měnit (přeučit) pro různé typy skel.

Navržený detektor by posléze mohl být implementován do nízkokapacitního výpočetního systému napájeného bateriemi (například čidla v místnosti). Proto je při konstrukci rozpoznávače nutné brát v úvahu jeho výpočetní náročnost. Vzhledem k bateriovému provozu by zařízení mělo být osazeno energeticky úspornou výpočetní jednotkou, která má snížený výpočetní výkon.

Inspirace v rozpoznávání řeči

Při návrhu řešení se pokusíme využít technologií, které se používají při procesech rozpoznávání řeči.

Základní princip řečového rozpoznávání spočívá v rozdělení vstupního signálu na menší části. Z těchto částí jsou vypočteny tzv. **příznakové vektory**. Tyto vek-

tory jsou potom zvolenou reprezentací nějakých vlastností (nejčastěji frekvenčních vlastností) vybrané části. Poté jsou stanovena pravidla, podle kterých se na základě příznaků určí, co daný signál představuje. Tento proces se nazývá **klasifikace**. Může se jednat o určité slovo nebo třeba o mluvčího. Každé slovo (případně mluvčí) tvoří jednu třídu klasifikátoru.

Pravidla pro klasifikaci byla v minulosti určována tzv. statistickými modely. Do této skupiny se řadí například směsi Gaussových komponent (GMM) a skryté Markovovy modely (HMM).

Sada příznakových vektorů má specifické statistické vlastnosti, které jsou pro každého mluvčího jiné. Směs Gaussových komponent se dá popsat jako lineární kombinace Gaussových rozložení. Parametry jednotlivých rozložení lze přizpůsobit tak, aby výsledná směs odpovídala statistickým vlastnostem vybraného mluvčího. Tomuto procesu se říká trénování. Každý mluvčí je poté reprezentován vlastním modelem. Pro vstupní sadu příznakových vektorů, lze pak určit pravděpodobnost toho, že se jedná o vybraného mluvčího [1].

Skryté Markovovy modely se využívají při rozpoznávání slov. Každé slovo je dáno specifickou posloupností menších částí – fonémů. Každý takový foném má specifické statistické vlastnosti. Skrytý Markovův model lze popsat jako množinu po sobě jdoucích stavů, kde každý stav je reprezentován pravděpodobnostním rozložením. Každé slovo se poté dá reprezentovat HMM modelem. A pro vstupní sadu po sobě jdoucích příznaků jsme schopni určit pravděpodobnost toho, že přísluší vybranému slovu [1].

V dnešní době se úspěšnost pravděpodobnostních modelů podařilo překonat použitím neuronových sítí, a proto se je pokusíme využít v našem návrhu. Neuronové sítě použijeme jako klasifikátor, jenž nám umožní určit, o jaký zvuk se jedná. Stejně jako ve zpracování řeči budeme klasifikovat krátké úseky zvuku. Poté zvolíme kritérium, podle kterého oddělíme zvuky rozbití skla od ostatních zvuků (pro nás ruchů).

Při rozpoznávání řečových signálů se jako vstupu do klasifikátoru využívá příznakových vektorů upraveného kepstra tzv. **MFCC**. V této práci se je pokusíme použít a následně je upravit a optimalizovat pro zvýšení úspěšnosti při rozpoznávání rozbití skel [1].

Výstup práce

Výstupem práce by měl být návrh systému, který je schopen rozpoznávat zvuky rozbití skla. Tento návrh se pokusíme implementovat na platformě Windows v programovacím jazyce C#. Platforma .NET byla zvolena, protože k ní mám nejblíže a je pro ni dostupná široká škála již implementovaných knihoven pro zpracování signálů a používání neuronových sítí.

Práce a její členění

Práce je dělená do čtyř kapitol, které lze rozdělit na dvě části:

1. Teoretická část (Úvod + Teoretická část (viz 1))

V této části se budeme zabývat popisem aparátů, struktur a pojmů, které jsou potřebné pro pochopení fungování námi vytvořeného detektoru a jeho následné implementace.

2. Praktická část (Praktická část (viz 2) + Závěr (viz 2.6))

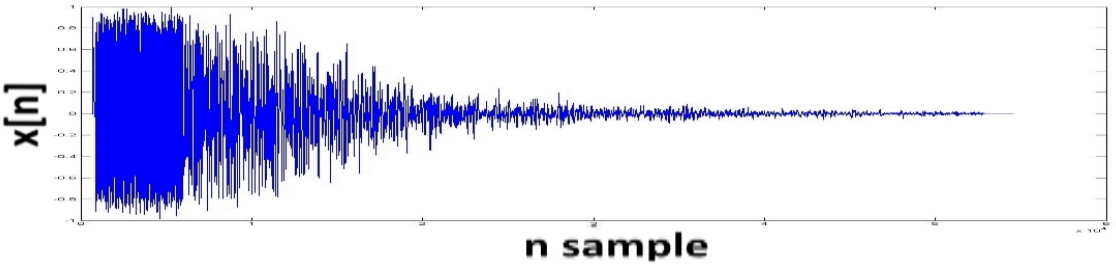
   Tato část popisuje vytvoření samotného detektoru. V kapitole „Praktická část“ jsou popsány postupy získávání dat, jejich předzpracování, optimalizace parametrizační části, trénování neuronových sítí a nakonec zhodnocení úspěšnosti a výpočetní náročnosti. Závěr shrnuje celkovou práci na detektoru a ukazuje pohledy na jeho další možnosti vývoje.

1 Teoretická část

V této kapitole se budeme zabývat definicí pojmů a aparátů použitých při vytváření detektoru. Jedná se jen o minimum, které je potřebné k pochopení fungování námi vytvořené jednotky. Začneme krátkou definicí signálů a operací, které nad nimi budeme provádět. Poté definujeme problém klasifikace a nakonec se seznámíme s významem, strukturou, tvorbou a učením umělých neuronových sítí.

1.1 Signál

Signálem se rozumí vývoj zvolené fyzikální veličiny v čase či v závislosti na jiné veličině. Signály máme např.: optické, elektrické, elektromagnetické, **akustické** apod. Vzhledem k povaze naší úlohy se budeme zabývat pouze signály akustickými. Z hlediska spojitosti signály dělíme na analogové (spojité) a diskrétní [2].



Obrázek 1.1: Ukázka vzorkovaného signálu.

- **Analogové**

  Vývoj veličiny je spojitý v čase. Zvolená veličina může nabývat libovolných hodnot.

- **Diskrétní**

  Vývoj veličiny se nemění spojitě v závislosti na čase. Ve většině případů tento signál vzniká vzorkováním analogového (viz 1.2).

V běžném životě se setkáváme se signály, které lze označit za analogové, příkladem je **zvuk**, který je tvořen mechanickým vlněním, je spojitý v čase a obor hodnot je také spojitý na určitém intervalu od minimální intenzity do maximální. Pro zpracování výpočetními systémy je analogový signál potřeba převést na diskrétní a jeho

obor hodnot převést do omezené skupiny hodnot (digitální). V případě, že se v další části práce setkáme s pojmem signál, tak budeme uvažovat signál akustický, který je digitalizovaný a je tedy tvořen uspořádanou posloupností hodnot [2].

1.2 Digitalizace

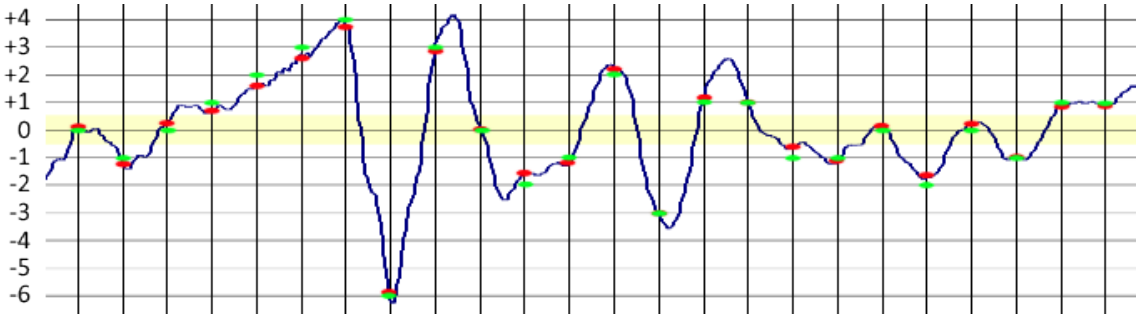
Jedná se o převod analogového signálu na digitální, tedy spojitého signálu na posloupnost hodnot, které mají stanovenou maximální požadovanou přesnost. S tímto procesem souvisejí dvě operace:

- **Vzorkování**

  Jedná se o operaci, při které se vybírají z analogového (spojitého) signálu jednotlivé vzorky v určitých časových bodech (vzorkovací frekvence). Kroky, které určují, jaké vzorky z analogového signálu vybírají, jsou zpravidla ekvidistantní. Tyto vzorky mohou obecně nabývat libovolné hodnoty. Aplikací této operace vzniká tzv. vzorkovaný signál. Při volbě vzorkovací frekvence je nutné se řídit **vzorkovacím teorémem**, podle kterého má zvolená vzorkovací frekvence být minimálně dvakrát vyšší než je maximální frekvence obsažená ve vzorkovaném signále (zabrání se tím „aliasingu“). Aliasing lze jednoduše popsat jako ztrátu informace při převodu spojitého signálu na diskrétní [2].

- **Kvantování**

  Jedná se o operaci, při které přiřadíme hodnotám jednotlivých vzorků hodnoty z našeho určeného omezeného rozsahu. Tento rozsah můžeme nazývat rozlišením či bitovou hloubkou. Aplikací této operace vzniká tzv. kvantovaný signál [2].

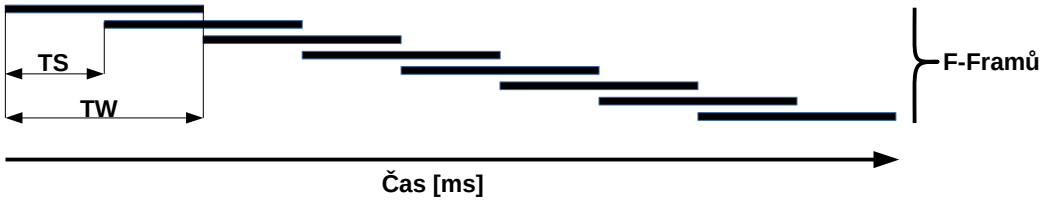


Obrázek 1.2: Ukázka výsledného kvantovaní na již navzorkovaném signálu. modrá spojnice – spojitý signál, červené body – „přesné“ hodnoty vzorkovaného signálu, zelené body – hodnoty po přiřazení [4]

Aplikací výše uvedených operací najednou vznikne digitální signál. Při tvorbě digitálního signálu můžeme zvolit vzorkovací frekvenci a rozlišení v podobě bitové hloubky. Digitální signál je z analogového signálu generován pomocí tzv. A/D převodníku, jenž je součástí zvukové karty. Maximální použitelná bitová hloubka a vzorkovací frekvence je dána schopnostmi A/D převodníku [2].

1.3 Segmentace signálu

Segmentace signálu je rozdělení signálu na menší části, které budeme zpracovávat. Rozdělené části mají zpravidla stejnou délku. Jednotlivé části budeme nazývat **framy**. Při zpracování signálů je typické, že se tyto sousední framy částečně překrývají (viz obrázek 1.3). Toto překrývání napomáhá lepšímu navazování parametrů (příznaků) definovaných níže a dále zabraňuje vynechávání informací v případě, že segmentaci provedeme v části, kde spolu rozdělené části jednoznačně souvisí např. uprostřed hlásky při zpracování řečových signálů [1].



Obrázek 1.3: Ukázka segmentace. Upravený obrázek z [1].

Obrázek č. 1.3 znázorňuje segmentaci signálu do F framů. TW je délka framů určená v milisekundách nebo počtem vzorků. TS znázorňuje o kolik milisekund (případně vzorků) se posuneme v původním signále k výběru vzorků do dalšího framu (typicky $TS = TW/2$) [1].

1.4 Analýza signálu

V této sekci se budeme zabývat pohledy, kterými lze nahlížet na signály. Vzhledem k tomu, že se v případě akustických signálů jedná o digitální reprezentaci mechanického vlnění, tak můžeme zkoumat jeho vlastnosti ve frekvenčním spektru nebo na něj můžeme nahlížet v časové oblasti jako na posloupnost hodnot.

1.4.1 Časová posloupnost

Jedním z pohledů na signály je analýza jejich jednotlivých vzorků. Na základě pohledu na jednotlivé vzorky můžeme vyvozovat závěry o intenzitě daného signálu v jeho jednotlivých částech. Při analýze signálu se hojně využívají statistické metody, které nám poskytují obecnější pohled buď na celkový signál či na jeho části [2].

1.4.2 Frekvenční analýza

Signály se sebou nesou velké množství informací, které nejsou ze samotné posloupnosti vzorků patrné a pro jejich odhalení mohou sloužit transformace, které signál v časové oblasti (posloupnost vzorků) mohou vyjádřit v oblasti jiné např. frekvenční. Každý signál lze vyjádřit složením řady harmonických funkcí (sinus a kosinus).

Základním příkladem transformace používané při zpracování signálů je Fourierova transformace (FT). Ta slouží právě pro převod z vyjádření signálu v závislosti na čase do jeho vyjádření v závislosti na frekvenci, tedy vyjádření jako součtu harmonických funkcí o určitých frekvencích. Pro převod z frekvenční oblasti zpět do časové se používá inverzní Fourierova transformace (IFT). V této práci se s Fourierovou transformací setkáme při vytváření MFCC příznakových vektorů, konkrétně tedy s diskrétní Fourierovou transformací, DFT, značena $\mathcal{F}$. Ta se používá při numerickém výpočtu vyjádření spektrálních vlastností digitálních signálů (vzorec pro výpočet 1.1).

$$S[k] = \mathcal{F}(s[n]) = \sum_{n=0}^{N-1} s[n] e^{-j\frac{2\pi}{N}kn}, \text{ kde } k = 0, \ldots, N-1 \tag{1.1}$$

Výpočet inverzní diskrétní Fourierovy transfomace, IDFT, značené $\mathcal{F}^{-1}$:

$$s[n] = \mathcal{F}^{-1}(S[k]) = \frac{1}{N} \sum_{k=0}^{N-1} S[k] e^{+j\frac{2\pi}{N}kn}, \text{ kde } n = 0, \ldots, N-1 \tag{1.2}$$

Výpočet spektra při použití přesné definice je výpočetně náročný, a proto se v praxi používá optimalizovaná varianta FFT (Fast Fourier Transform) [2, 3, 5].

Při práci se signály se využívá vlastností frekvenčního vyjádření signálu. Jednou z nich je například zjednodušení operace konvoluce dvou signálů v časové oblasti. Ta se ve frekvenční oblasti provádí jednodušší operací, násobením. S tím souvisí pojem Kepstrum signálu. Jedná se o vyjádření upraveného spektra signálu zpět v časové oblasti. Jedná se o transformaci signálu značenou $\mathcal{D}$. Kepstrum signálu je značeno $\hat{s}[n]$. Výpočet kepstra signálu je dáno následujícím vztahem:

$$\hat{s}[n] = \mathcal{D}(s[n]) = \mathcal{F}^{-1}(\ln(S[k]), \text{ kde } n = 0, \ldots, N-1 \tag{1.3}$$

Upravené spektrum ($\hat{S}[k]$) se vyznačuje aplikací logaritmu

$$\hat{S}[k] = \ln(S[k]), \text{ kde } k = 0, \ldots, N-1 \tag{1.4}$$

který zmiňované násobení převede na operaci součtu logaritmů. A po převedení jednotlivých spekter obou signálů zpět do časové oblasti se tedy operace konvoluce

$$s[n] = e[n] * h[n] \tag{1.5}$$

změní na operaci sčítání

$$\hat{s}[n] = \hat{e}[n] + \hat{h}[n] \tag{1.6}$$

(součet signálů v časové oblasti se rovná součtu spekter ve frekvenční oblasti). To je výhodné použít v případě oddělování jednotlivých složek signálu, ze kterých je výsledný signál složen. V rozpoznávání řeči se kepstra využívá, protože se vychází z faktu, že zvukový signál vycházející z hlasového ustrojí se skládá konvolucí ze dvou složek: výstup z hlasivek a impulzní odezvy hlasového ústrojí. V případě, že před aplikací logaritmu je vybrána absolutní hodnota (magnitudové spektrum)

$$c[n] = \mathcal{F}^{-1}(\ln(|S[k]|), \text{ kde } n = 0, \ldots, N-1 \tag{1.7}$$

tak budeme hovořit o reálném kepstru, značeném $c[n]$ [1, 3].

Spektrogram

Spektrum je výsledkem Fourierovy transformace. Jedná se o vyjádření zastoupení jednotlivých harmonických složek v rozkladu. Je typické, že se analyzuje spektrum v jednotlivých částech signálů např. v jednotlivých framech. Výsledkem této krátkodobé analýzy je spektrogram [3].

1.5 Příznaky, příznakové vektory, parametrizace

Parametr neboli příznak je informace, která vypovídá o námi určené vlastnosti vybraného framu. Příznakem může být prakticky libovolný výsledek popisu nějaké vlastnosti, kterou můžeme vyjádřit hodnotou či skupinou hodnot, v našem případě hodnot z oboru reálných čísel. V případě, že se jedná o skupinu hodnot, tak příznak nazveme příznakovým vektorem. Různé parametry můžeme řadit za sebe a vytvořit příznakový vektor nebo již vybraný vektor rozšířit [1].

V rámci této práce se budeme zabývat příznakem energie a MFCC (Mel-Frequency-Cepstral-Coeficients) příznaky. Energie signálu vyjadřuje informaci o jeho intenzitě. Energie signálu se vypočítá podle vztahu č. 1.8.

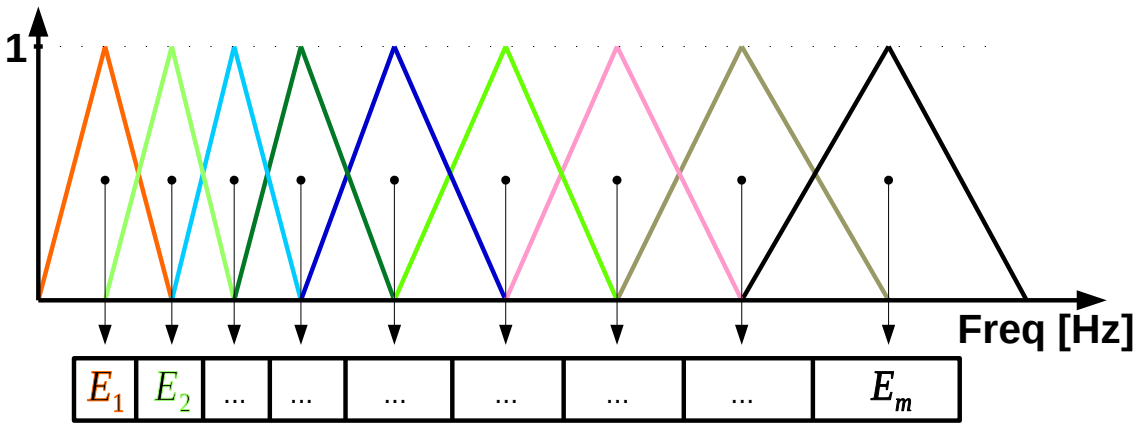
$$\sigma^2 = \frac{\sum_{i=0}^{N-1} x[i]^2}{N} \tag{1.8}$$

MFCC jsou příznaky vyvinuté pro systémy rozpoznávání řeči. Výpočet těchto koeficientů probíhá tak, že se vypočte amplitudové spektrum daného signálu. Toto spektrum je poté filtrováno bankou trojúhelníkových filtrů, což se později využije k výpočtu energie v pásmech jednotlivých filtrů. Uspořádání těchto filtrů je v Melovském měřítku frekvencí. Toto měřítko je charakteristické tím, že je postavené tak, že převod koresponduje s rozlišením lidského ucha při rozpoznávání frekvencí. Vztahy pro převod frekvencí do melovského měřítka a zpět (rovnice č. 1.9 a 1.10)[1, 3].

$$\text{Mel}(f) = 1127ln(1 + \frac{f}{700}) \tag{1.9}$$

$$\text{Mel}^{-1}(m) = 700((e^{\frac{m}{1127}}) - 1) \tag{1.10}$$

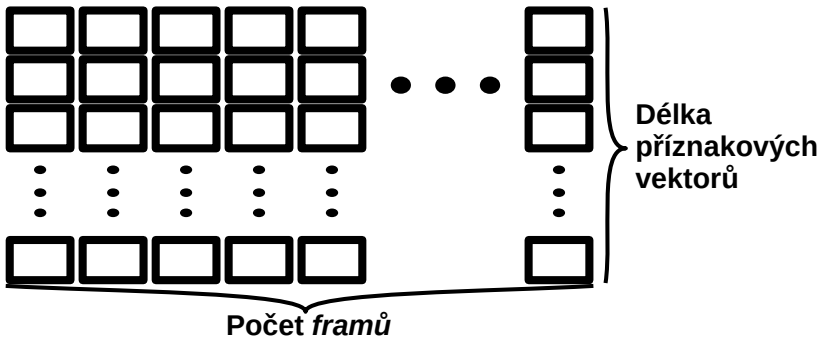
Výsledkem výpočtů energií v jednotlivých pásmech trojúhelníkových filtrů je vektor energií. Převod na vektor kepstrálních koeficientů je proveden aplikací diskrétní kosínové transformace na zlogaritmované hodnoty vektoru energií. Při tvorbě MFCC příznaků se může upravit jejich počet a tvar úpravou parametrů. Jako první uvedeme možnost změny frekvenčního pásma celé banky filtrů. Frekvenci nastavujeme v závislosti na úloze, kterou řešíme (pro zpracování řeči je typické pásmo od 300 Hz do 3700 Hz). Lze měnit počet filtrů v bance, doporučené nastavení je mezi 20 - 40 (zpracování řeči typicky 26). Dále je možné si zvolit, kolik výstupních příznaků vyžadujeme (omezeno zvoleným počtem filtrů), typicky se nevybírají všechny parametry, ale pouze spodní polovina [1, 3].



Obrázek 1.4: Vykreslení banky filtrů. Frekvenční osa je převedena do melovo stupnice a v ní jsou rozděleny filtry (ekvidistantně). Po převedení zpět získáme vyobrazené rozložení filtrů. Upravený obrázek z [1].

Parametrizací obecně rozumíme převedení segmentovaného signálu na matici příznakových vektorů (viz obrázek 1.5), kde sloupce matice představují příznakové vektory extrahované ze segmentovaných framů [1].

V případě, že je zachována souslednost jednotlivých segmentů, tak je možné jednotlivé příznaky rozšířit o tzv. dynamické (někdy delta) příznaky. V případě, že si dva libovolné sousední příznakové vektory označíme $X(i)$ a $X(i+1)$, tak se dynamické příznaky vypočítají jako rozdíl těchto dvou příznakových vektorů a výsledek je přidán jako rozšíření vektoru $X(i+1)$. Při aplikaci tohoto pravidla na všechny příznakové vektory vzniká problém s rozšířením prvního příznakového vektoru (neexistuje prvek pro index $i-1$), a proto tento první frame z matice příznakových vektorů vyloučíme [1].



Obrázek 1.5: Znázornění matice příznaků.

1.6 Klasifikace

Řekněme, že máme nějakou množinu prvků o určitých vlastnostech. Klasifikací nazveme úlohu, při které je potřeba prvky oné množiny roztřídit do skupin, tříd (z anglického „class“), na základě jejich vlastností. Uvedeme si příklad, který bude reprezentovat náš detektor. Naši množinu vstupních dat můžeme rozdělit na dvě skupiny: zvuk obsahující tříštění skla a zvuk bez tříštění skla. Předpokládáme, že zvukový signál máme rozdělen do jednotlivých framů (viz. segmentace) a z každého framu jsme si vytvořili příznakové vektory. Při analýze jednotlivých framů, již převedených do matice příznaků, budeme určovat, zda zkoumaný frame patří do té či oné skupiny. Výsledkem této operace získáme posloupnost, která říká, který frame patří do které skupiny [5, 6].

S pojmem klasifikace úzce souvisí pojem separabilita vstupního datového prostoru. Jedná se o to, zda a jak můžeme vstupní datový prostor rozdělit na dvě či více částí.

Rozlišujeme následující typy separabilních prostorů:

- Lineárně separabilní

  Data v daném prostoru můžeme rozdělit přímkou (prostor R^2), rovinou (prostor R^3). Obecně prostory R^n: (n) rozměrnou nadrovinou [5, 6].

- Nelineárně separabilní

  Data v daném prostoru můžeme rozdělit křivkou (prostor R^2), plochou (prostor R^3). Obecně pro prostory R^n: (n) rozměrnou nadplochou [5, 6].

- Neseparabilní

  O datovém prostoru můžeme například říci, že je lineárně neseparabilní v případě, že jeho prvky nelze rozdělit na jednotlivé skupiny lineární funkcí (přímkou, rovinou, nadrovinou) [5, 6].

Na základě typu separability vstupního datového prostoru budeme později vybírat aktivační funkce neuronových sítí.

1.7 Neurony a neuronové sítě

Vývoj nových technologií, principů a metod je často založen na pozorování přírodních jevů nebo na pozorovaném fungování biologických struktur. Jedním z takových případů je i pozorování fungování mozku.

Mozek má spousty specifických funkcí. Výběrem z nich může být schopnost předvídat a rozeznávat různé vzory. Příkladem takového vzoru můžeme uvést viditelný obraz, snímaný zrakovým ústrojím. Tento obraz je ve zrakovém ústrojí převeden na biologický signál a ten dále pokračuje ke zpracování do mozku. Mozek poté vyhodnotí, o jaký druh obrazu (např. předmětu) se jedná.

Mozek je tvořen nervovou tkání a ta se skládá z jednotlivých neuronů. Neuron je základní jednotkou nervové tkáně a jedná se o buňku, která je specializovaná na zpracování, uchovávání a přenosy informací. Na základě opakování nejrůznějších situací a opakovaném zpracovávání vzorů jsou neurony schopny si pamatovat a uchovávat informaci. Nervová buňka se skládá ze 4 základních částí:

- **Soma**

  Tělo neuronu, kde se nachází buněčné jádro [6].

- **Dendrity**

  Dendrity jsou vstupní jednotky zajišťující vedení „vzruchů“ směrem k tělu neuronu. Typicky jich je větší počet přibližně 10^4 [6].

- **Axon**

  Jedná se o jednu výstupní jednotku z těla neuronu. Každý neuron má jen jeden axon. Jeho délka je od několika mikronů do několika desítek centimetrů [6].

- **Synapse**

  Je nervové zakončení na konci axonu. Jedná se o jakési rozhraní na konci axonu, které slouží k napojení na dendrit jiného neuronu. Průchodnost tohoto rozhraní není konstantní, ale přizpůsobuje se v závislosti na potřebách a důležitosti daného nervového spojení. Přizpůsobení propustnosti se provádí při procesu učení a neuronová síť je potom na základě těchto upravených propustností schopna rozeznávat zapamatované vzory [6].

Mozek je tvořen širokou skupinou neuronů, které jsou vzájemně propojené a na základě vzruchů nějak reagují. Při procesu učení jsou u skupiny neuronů upravovány synapse a tato síť je poté schopna rozpoznávat známé skupiny vzruchů a na základě toho potom reagovat [6].

1.8 Umělé neuronové sítě

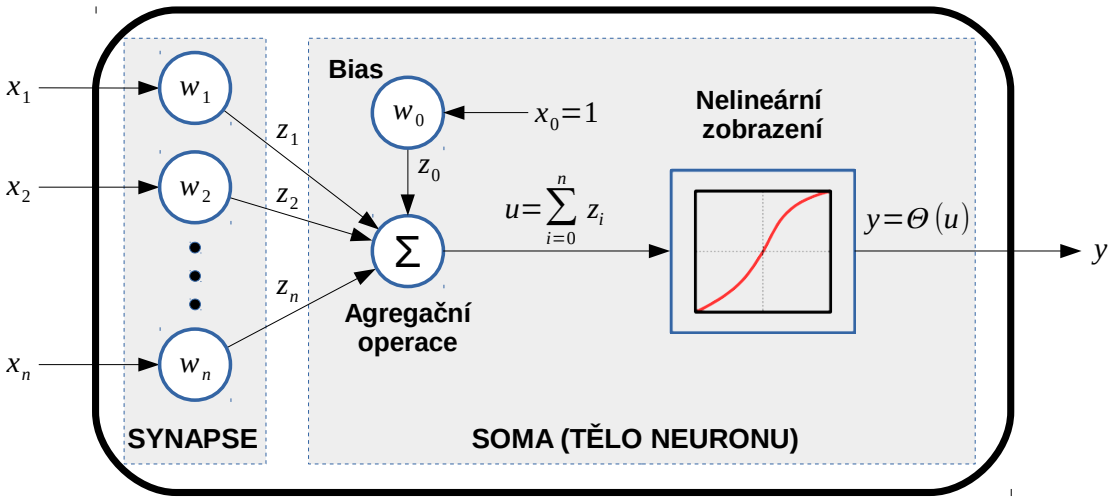
Jedná se o matematický model inspirovaný biologickými neuronovými sítěmi. Tento model poté může sloužit k různým účelům např. klasifikace, regrese, … Matematický model je o mnoho jednodušší než biologická jednotka, a proto nelze očekávat, že se bude jednat o tak mocný nástroj jako v případě např. lidského mozku.

I přes to, že technologie a modely neuronových sítí jsou již delší dobu známé, tak se jejich masivnější uplatnění nalezlo teprve až s příchodem optimalizovaných učících algoritmů a výkonných grafických karet, které jsou vhodné pro jejich učení. Mnohem více laboratoří a jedinců se s příchodem výpočetních standardů CUDA nebo OPENCL, které slouží k masivní paralelizaci, mohlo začít s trénováním rozsáhlejších vícevrstvých neuronových sítí. Toto trénování se poté z hlediska časové náročnosti velice zkrátilo. Již je možné trénovat rozsáhlejší neuronové sítě na běžném PC.

Pro sítě s vysokým počtem vrstev a vysokým počtem neuronů existuje označení **DNN** (Deep Neural Networks). Tyto sítě se používají ke zpracovávání velkého počtu dat a stojí za zvýšením úspěšnosti klasifikačních problémů posledních let.

1.8.1 Matematický model neuronu

Matematický model neuronu a neuronových sítí má základ v biologických systémech (nervové soustavy). Existuje více modelů neuronů, ale nejrozšířenější jsou deriváty od MCCulloch-Pittův neuronu, **formálního neuronu**. Ten, v základní podobě připouští vstupy, které nabývají pouze binárních hodnot a to se pro účely naší práce příliš nehodí, a proto využijeme obecnější verzi, která na vstupech připouští i reálná čísla [5, 6].



Obrázek 1.6: Schéma matematického modelu neuronu.[6]

Umělý neuron budeme považovat za matematický procesor, který z n-rozměrného vektoru vstupního signálu (analogie s dendrity) na základě předem definovaných (synaptických a somatických) operací vytvoří jedinou hodnotu, která je potom výstupem daného neuronu (axon) [6].

Označení (v literatuře se může mírně lišit) [5, 6, 7, 10]:

- $\boldsymbol{x}$ – je vektor vstupních hodnot neuronu. x_i je označení i-tého vstupu. $i = 1, \ldots, n$, kde n je počet vstupů.
- $\boldsymbol{w}$ – je vektor synaptických vah neuronu. w_i je označení synaptické váhy náležící i-tému vstupu.
- $\Theta(u)$ – je přenosová funkce neuronu (aktivační funkce)
- B – je parametr prahování aktivační funkce (Bias). Někdy označován jako práh (threshold).
- $y = f(\boldsymbol{x})$ – je výstupní hodnota z neuronu. Funkce $f(\boldsymbol{x})$ je transformační funkce neuronu. Přiřazuje výstupní hodnotu k předloženému vstupnímu vektoru $\boldsymbol{x}$. Její rozpis je $f(\boldsymbol{x}) = \Theta(\sum_{i=1}^{n} x_i w_i)$

Synaptické operace

Jsou to operace, které vstupní vektor upravují na základě paměti daného neuronu – synaptických vah. V obecném případě můžeme používat různé operace, ale v našem základním modelu budeme uvažovat operaci násobení mezi i-tým prvkem vstupního vektoru a i-tým prvkem z vektoru synaptických vah (lineární kombinace). Výsledný vektor označíme $\boldsymbol{z}$ [6, 5].

$$z_i = x_i.w_i, i \ = 1, \ldots, n \tag{1.11}$$

Somatické operace

Jedná se o souhrn operací, mezi které řadíme operaci zajišťující agregaci vstupního vektoru $\boldsymbol{z}$ a operaci nelineárního zobrazení spolu s prahováním.

Agregační funkci je možné zvolit téměř libovolně dle požadavků, jen je nutné dodržet pravidlo, aby funkce agregovala (výstupem je jedna hodnota) vstupní vektor do skalární hodnoty. Tuto skalární hodnotu si označíme u. V námi definovaném základním modelu budeme za agregační funkci používat operaci sumace. V našem případě je výsledkem součet všech složek vektoru $\boldsymbol{z}$ [6, 5].

$$u = \sum_{i=1}^{n}(z_i) \tag{1.12}$$

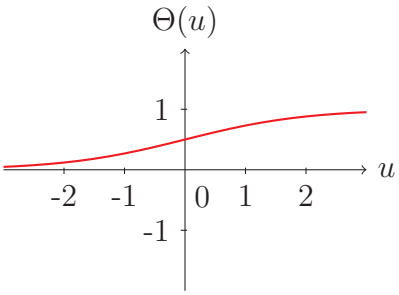
Prahování je operací, která definuje, zda je či není neuron aktivní.

V řadě modelů neuronu je bias definován jako další parametr aktivační funkce. Tento parametr pak ovlivňuje aktivační funkci (posouvá její střed) – $\Theta(u + B)$. Tato operace slouží k tomu, aby se výstup agregační funkce posunul do oblasti, kde aktivační funkce není saturovaná. Tento parametr patří do upravovaných složek při učení sítě [5].

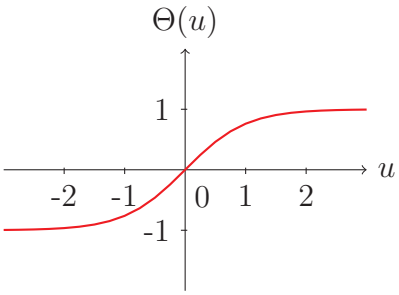
Zápis prahu se dá zjednodušit tak, že rozšíříme vstupní vektor $\boldsymbol{x}$ o nultou složku. Ta bude reprezentovat virtuální vstup, který bude mít pevně definovanou hodnotu $x_0 = 1$. Tento virtuální vstup rozšíříme také o synaptickou váhu (w_0), která bude definovaná na hodnotu námi požadovaného prahu ($w_0 = B$) (viz 1.6). S virtuálním vstupem pracujeme stejně jako s ostatními vstupy, je tedy zahrnut do agregační funkce. Výsledkem této úpravy je zjednodušení matematického zápisu, což se v případě pozdější implementace může hodit při využití maticových operací či při odvození algoritmů pro trénování sítí [6, 5].

Aktivační funkce je operace realizující většinou nelineární zobrazení. V základním modelu neuronu je definovaná skoková funkce (viz 1.8). Bohužel, v případě jejího použití nelze využít optimalizovaného učícího **algoritmu zpětného šíření chyby**. Pro použití tohoto algoritmu je potřeba, aby aktivační funkce byla hladká a tedy diferencovatelná na celém definičním oboru, proto se tato skoková funkce nahrazuje její hladkou aproximací, **sigmoidou**, $\Theta(u) = 1/(1 + e^{(-u/T)})$ (viz 1.7, parametr T určuje strmost funkce) [5, 6].

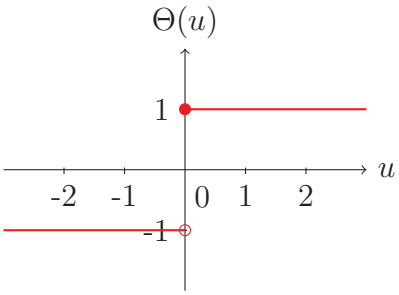
Sigmoida není jediná funkce, kterou lze využít. Podobný průběh, i když s odlišným oborem hodnot, má například **hyperbolický tangents**, $\Theta(u) = tanh(u/T)$



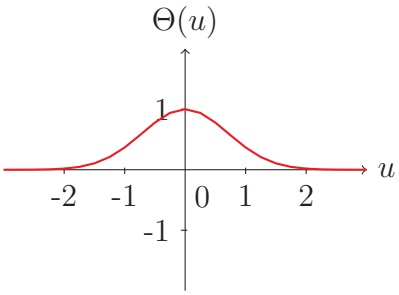
Obrázek 1.7: Sigmoida



Obrázek 1.9: Hyperbolický tangens



Obrázek 1.8: Skoková funkce



Obrázek 1.10: Gaussova funkce

(viz 1.9).

Se zvolením vhodné aktivační funkce souvisí pojem separability vstupního datového prostoru. V případě, že vstupní datový prostor je lineárně separabilní, tak lze použít skokovou aktivační funkci. V případě, že vstupní datový prostor je nelineárně separabilní, tak je vhodné zvolit některou nelineární aktivační funkci. Obrázky 1.7 až 1.10 znázorňují vykreslené nejvyužívanější aktivační funkce [5, 6].

Na volbě aktivační funkce je velice závislá rychlost konvergence učících algoritmů a také výpočetní náročnost v případě průchodu informace neuronovou sítí [5].

1.8.2 Skládání neuronů do neuronových sítí

Pro praktické použití neuronů k řešení složitějších (např. klasifikačních) problémů nám nestačí využití pouze jednotlivých neuronů. Stejně jako v případě biologických neuronových sítí lze umělé neurony skládat (zapojovat) do složitějších struktur. Tyto struktury (dále neuronové sítě) se dají poté učit jako celek pomocí specializovaných algoritmů.

Neuronové sítě se skládají ze třech základních částí: vstupní vrstvy, **vnitřních skrytých vrstev** a výstupní vrstvy. Vstupní vrstva definuje vstup do neuronové sítě (počet vstupujících parametrů), které jsou poté pouze předány do dalších vrstev. „Black box“, který je reprezentován skrytými vrstvami zajišťuje samotnou logiku sítě a výstupní vrstva definuje výstup, který je určen definicí naší úlohy.

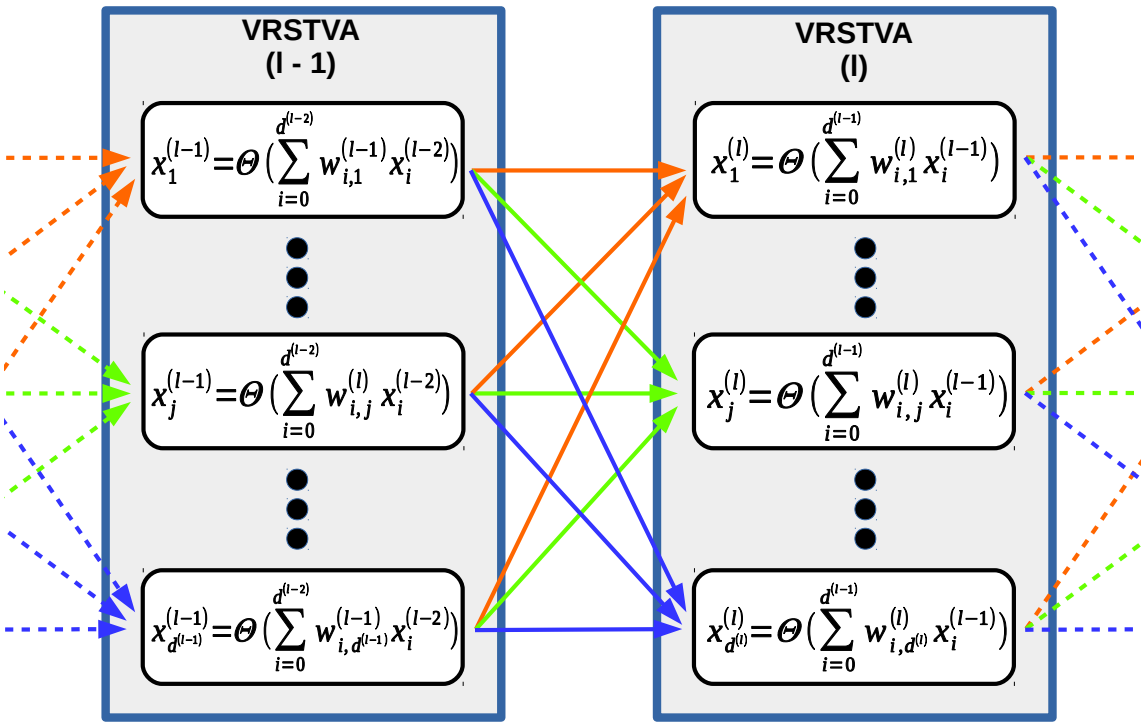
Každá vrstva může obsahovat různý počet neuronů. Pro účely identifikace jednotlivých vah v neuronové síti si zavedeme následující značení: $w_{i,j}^{(l)}$, kde spodní index i značí identifikaci vstupu určeného neuronu, index j určuje, o jaký neuron v rámci jedné vrstvy jde, a horní index (l) určuje, o jakou jde vrstvu v rámci sítě [7, 5].

Výstupní vektor ze sítě značíme $\boldsymbol{y}$. Tento vektor je složený z výstupních hodnot neuronů v poslední vrstvě neuronové sítě. Jedná se o výsledek transformační funkce neuronové sítě $\boldsymbol{y} = f(\boldsymbol{x})$, kde $\boldsymbol{x}$ je vektor vstupních hodnot pro celou síť. Počet neuronů v l-té vrstvě budeme značit $d^{(l)}$ [7, 5].

Dopředné sítě (Feed-Forward)

Neuronové sítě můžou mít různou topologii (možnosti propojení neuronů), která charakterizuje schopnosti dané sítě a také určuje, jaké algoritmy lze použít pro jejich učení. V rámci této práce budeme využívat topologii zvanou **dopředné sítě** (Feed-forward).

Propojení neuronů této topologie je v rámci dvou sousedních vrstev úplné, tvoří úplný bipartitní graf. Tento typ sítě pozůstává minimálně ze 3 vrstev: vstupní vrstvy, minimálně jedné skryté vrstvy a jedné vrstvy výstupní. V této topologii není povoleno zapojení zpětných vazeb, kdy je výstup neuronu připojen na vstupy neuronů, které jsou obsaženy v nějaké nižší vrstvě. Toto propojení je k dispozici u topologií rekurentních neuronových sítí. Dalším omezením je nemožnost použití tzv. laterálních propojení (výstup neuronu přiveden na vstup neuronu ve stejné vrstvě) [5, 6].



Obrázek 1.11: Ukázka propojení vedlejších vrstev $(l-1)$ a (l).

Signál (vstupní vektor) vstupující do sítě je šířen směrem od nejnižších vrstev k nejvyšším. Signály, které jsou šířeny na vstupy mezi vrstvami, jsou značeny $x_i^{(l)}$. Vzhledem k tomu, že sousední vrstvy jsou úplně propojené, tak lze výstup j-tého

neuronu v l-té vrstvě popsat následujícím vztahem:

$$x_j^{(l)} = \Theta(u_j^{(l)}) = \Theta(\sum_{i=0}^{d^{(l-1)}} w_{i,j}^{(l)} x_i^{(l-1)}) \tag{1.13}$$

Z toho plynou následující rozsahy indexů – $l : \{1 \leq l \leq L\}; i : \{0 \leq i \leq d^{(l-1)}\}; j : \{1 \leq j \leq d^{(l)}\}$ [7].

1.8.3 Učení (trénování) neuronových sítí

Učením neuronových sítí (biologických i umělých) rozumíme úpravu parametru průchodnosti synapsí (v umělé síti tuto průchodnost reprezentuje váha w u každého vstupu do zvoleného neuronu). Úprava parametrů je založena na tzv. **účelové funkci**, která nám v podstatě říká, jestli jsme váhy sítě upravili správně/špatně a jakého zlepšení/zhoršení jsme docílili. Učení umělých neuronových sítí se dělí na dvě skupiny: trénování s učitelem a trénování bez učitele. V obou případech je pro učení potřeba mít množinu trénovacích dat (vzorů) [5, 6].

Trénovací množina vzorů je v případě trénování s učitelem postavená tak, že ke každému trénovacímu vzoru máme dodanou informaci o námi požadovaném výstupu ze sítě. V případě klasifikace jde o označení skupiny, do které daný vzorek patří, a v případě regrese přímo požadovaná hodnota či vektor hodnot [5, 6].

Trénování bez učitele funguje na principu samo-organizování (seskupování) neuronů na základě typických vlastností vzorů z trénovací množiny. Zpravidla se používá topologie sítě, která má dvě vrstvy: vstupní, výstupní. Vstupní vrstva je definovaná úlohou. Výstupní vrstva je tvořena neurony, které jsou většinou poskládány do mřížky. Každý z neuronu ve výstupní vrstvě je plně propojen se vstupní vrstvou. To zajišťuje stejnou dimenzi vstupního vektoru s dimenzí vektoru vah pro každý neuron ve výstupní vrstvě. Umístění neuronu v prostoru je poté reprezentováno nastavením jeho vah. Učení probíhá tak, že se porovnává vzdálenost (typicky Euklidovská) mezi trénovacím vzorem a všemi neurony. U neuronu, který je danému trénovacímu vzoru nejblíže (vítězi), se upraví váhy tak, aby se předloženému vzoru ještě více přiblížil. Tato operace se provede pro všechny trénovací vzory. Po ukončeném trénování bude, v ideálním případě, možné v síti nalézt shluky neuronů, které budou reprezentovat jednotlivé vlastnosti trénovací množiny [5, 6].

Existuje spousta algoritmů určených k učení umělých neuronových sítí. O všech ale můžeme říci, že to jsou v podstatě optimalizační procesy, které se snaží chytře upravit váhy tak, aby aktuální výstup z neuronové sítě odpovídal tomu požadovanému. Pro optimalizaci je potřeba znát účelovou funkci, kterou budeme optimalizovat. Účelová funkce je v případě učení s učitelem vyjádřením **chybové funkce**, která nám říká, jaké chyby se síť dopouští oproti požadovaným výstupům. V případě samo-organizace (učení bez učitele) je funkcí, která vyjadřuje vzdálenosti mezi neurony a prvky trénovací množiny v prostoru. V obou případech zjednodušeně síti předkládáme trénovací vzory s požadavkem, aby se z nich síť naučila nějakou informaci. V rámci této práce se dále budeme zabývat pouze trénováním s učitelem [5, 6].

Úprava parametrů sítě se může provádět po předložení jednotlivých trénovacích vzorů nebo po předložení skupiny trénovacích vzorů (může to být i celá trénovací množina) tzv. **batch-training (dávkové učení)**. V druhém případě je chybová funkce vyjádřením chyby všech trénovacích vzorů z předložené skupiny. Každá iterace učení, při níž se změní nastavení vah, se nazývá **epocha**. Algoritmus úprav parametrů při trénovaní se dá popsat následujícím výčtem operací [5].

1. Vložení trénovacího vzorku (skupiny vzorů) do sítě
2. Vyčíslení účelové funkce
3. Úprava parametrů sítě
4. Opakování kroků 1–3 až do dosažení určené chybovosti nebo do dosažení časového limitu učení

V rámci této práce se budeme zabývat skupinou algoritmů založených na gradientních metodách. Hlavním představitelem této skupiny je algoritmus zvaný **algoritmus zpětného šíření chyby** (Back-Propagation Algorithm), který je vzorem mnoha modifikací, které se jej snaží zefektivnit [5, 8, 9].

Algoritmus zpětného šíření chyby

Tento algoritmus byl odvozen pro potřeby trénování dopředných sítí. Budeme předpokládat, že pro učení máme k dispozici množinu trénovacích vzorů. Ta je definována následovně:

$$\{\boldsymbol{x(k)}; \boldsymbol{t(k)}; 1 \leq k \leq K\ \} \tag{1.14}$$

kde $\boldsymbol{x(k)}$ je vstupní vektor do neuronové sítě, $\boldsymbol{t(k)}$ je požadovaná hodnota nebo vektor (target), K je celkový počet trénovacích vzorů [5].

Výstup ze sítě pro vstupní vektor $\boldsymbol{x(k)}$ nazveme $\boldsymbol{y(k)}$ a všechny váhy v neuronové síti definujeme jako množinu $\mathcal{W} = w_{i,j}^{(l)}$. Jedná se o gradientní metodu, která se snaží najít minimální hodnotu chybové funkce pro různé nastavení $\mathcal{W}$. V základní verzi algoritmu je za chybovou funkci použito vyjádření součtu obsahů čtverců sestavených mezi výstupem $\boldsymbol{y(k)}$ a požadovaným $\boldsymbol{t(k)}$ výstupem. K minimalizaci jsou upravovány pouze volné parametry sítě — váhy [7, 5].

Požadovaný vektor, $\boldsymbol{t(k)}$, je dán námi definovaným problémem. Zde bych uvedl, že v případě klasifikace je požadovaná hodnota výstupu definována jako vektor, který má stejnou dimenzi, jako je počet skupin, do kterých vzory chceme klasifikovat. Jednotlivé skupiny jsou vektorem znázorněny tak, že vektor má nulové složky a pouze na pozici, která je vybraná pro danou skupinu, má hodnotu jedna [6].

$$E(\mathcal{W}) = \sum_{k=1}^{K} E(\boldsymbol{y(k)}, \boldsymbol{t(k)}) = \sum_{k=1}^{K} E(f(\boldsymbol{x(k)}), \boldsymbol{t(k)}) \tag{1.15}$$

$$E(\mathcal{W}) = \sum_{k=1}^{K} [\boldsymbol{t(k)} - \boldsymbol{y(k)}]^2 \tag{1.16}$$

Rovnice 1.15 definuje chybovou funkci obecně. Chyba může být vyjádřena libovolným pravidlem nějaké chybové funkce, musí ale vyjadřovat nějakou informaci o porovnání požadovaného výstupu, $\boldsymbol{t(k)}$, s výstupem ze sítě pro vstupní vektor $\boldsymbol{x(k)}$. Rovnice 1.16 ukazuje konkrétní podobu chybové funkce a to je **metoda nejmenších čtverců**. V obou případech se jedná o vyjádření funkce která charakterizuje chybu přes všechny trénovací vzory [5, 7].

Vzhledem k tomu, že nám jde o minimalizaci, tak použijeme obecnou formuli gradientní metody (rovnice 1.17) — **metody největšího spádu**:

$$\mathcal{W}(t+1) = \mathcal{W}(t) - \eta \nabla E(\mathcal{W}(t)) \tag{1.17}$$

kde t je označení konkrétní iterace. Rozsah iterací je ve většině případu omezen a t může nabývat hodnot: $t = 1, \ldots, T$, kde T značí maximální počet iterací. Nastavení vah v iteračním kroku t je značeno $\mathcal{W}(t)$. Jedná se o iterační metodu, kdy se postupnými kroky snažíme dosáhnout optimálního řešení [7].

Při použití této metody nemusí být iterátor t omezen a algoritmus skončí po dosažení námi definované maximální chybové hodnoty. Gradient vyjádřený jako vektor parciálních derivací je ve vztahu 1.18 [5, 6, 7].

$$\nabla E(\mathcal{W}) = \frac{\partial E(\mathcal{W})}{\partial w_{i,j}^{(l)}} \tag{1.18}$$

Gradient je vektor parciálních derivací a udává směr největšího růstu chybové funkce v závislosti na nastavení vah sítě. Princip metody je tedy „jít“ v opačném směru gradientu o krok (délka kroku je nastavena parametrem η) blíže k hledanému minimu. Poté nastavit nově nalezené parametry volných parametrů optimalizace (vah). Tento proces se opakuje až do dosažení nějaké stanovené maximální chyby nebo do dosažení maximálního počtu iterací. Zde je nutné podotknout, že v případě nalezení minima se nemusí jednat o minimum globální, ale optimalizační proces může tzv. „uvíznout“ v nějakém jiném lokálním minimu. Absolutně optimálního řešení nemusíme nikdy dosáhnout. S tím souvisí počáteční nastavení vah, které jsou v případě algoritmu zpětného šíření chyby nastaveny na malá náhodná čísla a optimalizační proces se doporučuje spustit několikanásobně, abychom vyloučili taková uvíznutí [5, 7, 10].

Silným faktorem úspěšné optimalizace je vhodné nastavení parametru určujícího délku kroku ve směru gradientu. V případě, že je krok nastaven na příliš vysokou hodnotu, tak se může stát, že vždy požadované minimum „přeskočíme“ a naopak při příliš malém kroku bude metoda konvergovat k optimu velice pomalu a nemusí nám stačit rozsah iterací. Dalším důležitým faktorem je volba aktivační funkce [5, 7, 10].

Vyjádření gradientu je možné provést analyticky (případně numericky) jednotlivě pro každé l, j, i. To ovšem není postup, který by byl výhodný pro případy, kdy jsme si zvolili síť s větším počtem neuronů a větším počtem vrstev. Pro výpočet jednotlivých složek gradientu je k dispozici optimalizovaný výpočet, který je založen na využití rekurze. V prvním kroku určíme hodnotu gradientu pro váhy neuronů, které se nacházejí v poslední vrstvě, tedy vrstvě (L). Vzhledem k tomu, že hodnota

u je hodnota závislá na nastavení vah, tak můžeme provést rozpis gradientu podle rovnice 1.19:

$$\frac{\partial E(\mathcal{W})}{\partial w_{i,j}^{(l)}} = \frac{\partial E(\mathcal{W})}{\partial u_j^{(l)}} \cdot \frac{\partial u_j^{(l)}}{\partial w_{i,j}^{(l)}} \tag{1.19}$$

$$\frac{\partial u_j^{(l)}}{\partial w_{i,j}^{(l)}} = x_i^{(l-1)} \tag{1.20}$$

$$\frac{\partial E(\mathcal{W})}{\partial u_j^{(l)}} = \delta_j^{(l)} \tag{1.21}$$

Rozklad byl proveden na základě pravidla o derivování složené funkce (řetízkové pravidlo). Jednotlivé členy součinu lze vyjádřit rovnicemi 1.20 a 1.21 [5, 7].

Chybu, kterou jsou zatíženy váhy $w_{i,j}^{(l)}$, tedy váhy j-tého neuronu v (l)-té vrstvě označíme $\delta_j^{(l)}$. Pro výstupní vrstvu ji můžeme rovnou vyčíslit:

$$\delta_j^{(L)}(k) = \frac{\partial E(\mathcal{W})}{\partial u_j^{(L)}(k)}, \ kde \ E(\mathcal{W}) = \sum_{(k=1)}^{K} [\boldsymbol{t(k)} - \Theta(u_j^{(L)})(k)]^2 \tag{1.22}$$

tedy po zderivování

$$\delta_j^{(L)}(k) = -2 \sum_{k=1}^{K} [\boldsymbol{t(k)} - \boldsymbol{y(k)}] \Theta'(u_j^{(L)}(k)) \tag{1.23}$$

v případě, že zvolíme jako aktivační funkci sigmoidu, tak

$$\delta_j^{(L)}(k) = -2 \sum_{k=1}^{K} [\boldsymbol{t(k)} - \boldsymbol{y(k)}] \boldsymbol{y(k)} [1 - \boldsymbol{y(k)}] \tag{1.24}$$

Výpočet chyb pro všechny neurony se provádí vrstvu po vrstvě směrem od výstupní vrstvy (tu lze vyčíslit). Při výpočtu chyby v nižší vrstvě, $\delta_j^{(l-1)}$, použijeme již vypočtenou chybu, $\delta_j^{(l)}$, jedná se tedy o rekurentní algoritmus. Vztah pro výpočet:

$$\begin{aligned} \delta_i^{(l-1)}(k) = \frac{\partial E(\mathcal{W})}{\partial u_i^{(l-1)}(k)} &= \sum_{j=1}^{d^{(l)}} \frac{\partial E(\mathcal{W})}{\partial u_j^{(l)}(k)} \cdot \frac{\partial u_j^{(l)}(k)}{\partial x_i^{(l-1)}(k)} \cdot \frac{\partial x_i^{(l-1)}(k)}{\partial u_j^{(l-1)}(k)} \\ &= \sum_{j=1}^{d^{(l)}} \delta_j^{(l)}(k) \cdot w_{i,j}^{(l)} \cdot \Theta'(u_i^{(l-1)}(k)) \end{aligned} \tag{1.25}$$

Po vyčíslení všech δ již lze podle vztahu 1.19 dopočítat hodnoty všech složek gradientu.

Výpočet končí upravením vah mezi vstupní vrstvou a první skrytou vrstvou [5, 7].

Modely neuronových sítí jsou známé již od počátku 20. let, ovšem až s příchodem tohoto algoritmu se jim dostálo širšího využití [6].

Analýza průběhu učení

Při učení neuronových sítí je potřeba sledovat průběh daného učení. Toho dosáhneme sledováním vývoje chybové funkce v průběhu trénovacích epoch. Je nutné sledovat její trend. V případě, že je učení úspěšné, tak je možné pozorovat sestupný trend chybové funkce. V opačném případě lze pozorovat monotonní či oscilující průběh.

S dosažením úspěchu při trénování neuronových sítí se pojí hlavní faktor této problematiky a tím je vhodné nastavení parametrů (velikost trénovacího souboru, rozvržení sítě, krok učení, …). Monotonní průběh chybové funkce může ukazovat na příliš velký soubor trénovacích dat nebo na nevhodné nastavení kroku učení – příliš malá hodnota parametru η. Oscilující průběh může být příčinou malé trénovací množiny nebo nastavení velkého kroku učení [5, 6].

Dalším problémem je odhadnutí doby, kdy je vhodné ukončit probíhající učení. K tomu slouží vytvoření další datové množiny, kterou nazveme **testovací množinou**. Do této množiny přiřadíme vzory, které nebudou síti předkládány v průběhu učení tj. v případě jejich předložení nebudou na jejích základě upravovány parametry sítě. V případě předložení této množiny neuronové síti bude pouze vyčíslena chybová hodnota na výstupu sítě, přes všechny testovací vzory. Trénování je většinou ukončeno po dosažení požadované chybovosti sítě pro testovací množinu.

Při trénování je sledován zároveň průběh chybové funkce trénovací množiny a chybové funkce testovací množiny. V případě, že je učení úspěšné, budou průběhy obou funkcí podobné. Naopak v případě, že se trendy budou rozcházet, zejména tedy v případě, kdy je chybovost určení trénovací množiny extrémně nižší než chybovost určení testovací množiny, tak je síť pravděpodobně přeučena tzv. **overfitting** [5, 6, 11].

Před vysvětlením pojmu přeučení a jeho příčin je potřeba zavést pojem **generalizace** sítě. Jedná se o schopnost sítě naučit se z trénovacích dat vlastnosti, na základě kterých zůstane zachována schopnost správně klasifikovat vzory, které nebyly součástí trénovací množiny. Obecně je u neuronových sítí požadavkem dobrá generalizace. Ve většině případů jsou neuronové sítě využívány ke zpracovávání vzorů, které jsou pouze „podobné“ vzorům určeným k trénování (nejsou úplně stejné) [11].

Přeučení sítě je jev, který můžeme popsat jako opak generalizace. Jedná se tedy o stav, kdy je síť naučena na všechny vlastnosti trénovací množiny (má pro její rozpoznávání vysokou úspěšnost), ale ztrácí schopnost rozpoznávat ostatní vzory. Přeučení sítě může mít mnoho příčin. Nejčastější z nich bývá nedostatečné množství trénovacích dat. Řešením může být jejich rozšíření nebo upravení modelu sítě [5, 11].

Opakem přeučené sítě je síť, jejíž výsledky nijak nekorespondují s požadovanými hodnotami výstupů. V tomto případě se chyba může skrývat již v trénovací množině, kdy jsou data natolik odlišná, že v nich nelze nalézt žádné závislosti. Jinou příčinou může být příliš jednoduchý model sítě.

Vliv na rychlosti učení (ve smyslu konvergence chybové funkce) může mít nastavení parametru η či námi zvolená aktivační funkce. Určení přesného modelu, jako je topologie sítě, zvolení aktivační funkce, nastavení rychlosti učení, je předmětem pokusů a dlouhodobých zkušeností a neexistuje pro něj přesné přímočaré řešení.

2 Praktická část

V této kapitole se budeme zabývat samotným postupem při vytvoření návrhu detektoru rozbití skla. První část bude věnována získávání zdrojových dat. Zdrojová data budou tvořena zvukovými nahrávkami. Nalezené nahrávky budou posléze analyzovány a předzpracovány. Výsledky analýzy budou stěžejní při samotném navrhování detektoru. Po vytvoření návrhu se pokusíme některé jeho části optimalizovat, aby bylo dosaženo lepších schopností klasifikace. Výsledky optimalizační části budou zdokumentovány a v závěru této kapitoly provedeme jejich srovnání.

2.1 Zdrojová data

Prvním problémem při tvorbě detektoru bylo získání zdrojových dat. Vzhledem k tomu, že se jedná o detektor rozbití skla, tak by pro zajištění dostatečného množství nahrávek bylo zapotřebí získání velkého množství skel. Tato skla by bylo potřeba rozbít a produkované zvuky zaznamenat. To by bylo časově i materiálově velice náročné, a proto námi pořízené zvukové záznamy sloužily především k ověření autentičnosti dat získaných z internetových databází.

Zdrojová data pro zpracování tématu byla získána hlavně z internetových zdrojů. Nalezené záznamy byly v surové podobě a bylo zapotřebí je předzpracovat. Počet nahrávek byl uměle navýšen použitím RIR (Room Impulse Response) generátoru (viz sekce 2.1.1). Během zpracování práce bylo několik záznamů pořízeno vlastním nahrávacím zařízením.

Při získávání dat již bylo předpokládáno, že pro klasifikaci bude využita technologie neuronových sítí. Pro úspěšnou klasifikaci s využitím neuronových sítí je zapotřebí získat dostatečné množství dat ze všech skupin, do kterých chceme předkládané vzory třídit. Náš detektor je v tomto ohledu jednoduchý, klasifikátor data třídí do dvou skupin. Do první skupiny patří zvuky tříštění skel. Druhá skupina bude zastoupena všemi ostatními zvuky. Zvuky patřící do druhé skupiny jsou pro nás ruchy.

Výsledný detektor by měl být instalován například jako ochrana proti nechtěnému vniknutí do objektu. Na základě zvoleného objektu by měly být vybrány zvuky, které zastupují pozici ruchů. Příkladem takového objektu je rodinný dům, pro který může skupina ruchů vypadat následovně: mluva, zvuky tekoucí vody, zvuky projíždějících aut okolo objektu apod. Zvuky rozbití skla by měly být složeny hlavně ze zvuků produkovaných rozbitím skleněných tabulí. Zvuk vydaný při rozbití skla je ovlivněn mnoha faktory. Hlavním z nich je typ skla a jeho povrchové úpravy

(např. klasické sklo, temperované sklo s folií atd.). Jiným faktorem je velikost skleněné tabule. Vzhledem k rozmanitosti skupin jsme museli získat dostatečné množství nahrávek.

Zvuky rozbití skel se podařilo získat hlavně z internetové databáze zvukových nahrávek — **Freesound** [12]). V této databázi byl nalezen „balík“, ve kterém jsou shromážděny nahrávky od uživatele *GEVAROY*[13]. Uživatel pořídil 48 nahrávek rozbití skel[1] a tyto nahrávky vložil do uvedeného balíku. Přidaný popis archívu ukázal, že balík obsahuje: nahrávky obsahující zvuky rozbití různě velkých kusů skel, zvuky dopadajících předmětů na skleněné desky a také zvuky způsobené škrábáním na sklo různými předměty. Pořizování nahrávek bylo provedeno na staveništi. Autentičnost nahrávek byla ověřena porovnáním s vlastními nahranými zvuky.

Většina nahrávek ruchů[2] byla získána z otevřeného projektu CHiME Speech Separation and Recognition Challenge (výsledky prezentovány v [14], WEB: [15]). Jedná se o kampaň, ve které se skupiny snaží překonávat současné limity v úlohách zpracovávání řečových signálů (separace, rozpoznávání řeči apod.). Po registraci získají uživatelé přístup k nahrávkám, potřebným ke zpracování úkolu. Dostupné soubory mimo jiné obsahují nahrávky, které byly pořízeny v různě zarušených prostředích. Z jejich databáze jsme vybrali následující rušná prostředí: autobus, rušná pozemní komunikace, kavárna a pěší zóna (vlaková stanice). Ke každému zvukovému záznamu je k dispozici více zvukových stop. Nahrávání stop probíhalo současně z několika mikrofonů najednou. Zvuky jsou nahrávané při použití 16kHz vzorkovací frekvence a 32 bitové hloubky. Kromě záznamů získaných z projektu CHiME jsme vytvořili zvukové nahrávky bílého šumu o různých parametrech.

Při zajišťování vlastních nahrávek byla vždy použita stejná externí zvuková karta. To by mělo případně zabránit chybám vzniklým různými parametry A/D převodníků různých zvukových karet. Ve všech případech byl použit levný stojanový mikrofon.

2.1.1 RIR reprodukce nahrávek

Vzhledem k tomu, že byl nalezen menší počet nahrávek zvuků tříštění, tak jejich množství bylo uměle navýšeno použitím RIR generátoru [16].

RIR generátor (generátor impulzní odezvy místnosti) slouží k vytvoření impulsní odezvy námi definované místnosti. Konvolucí vygenerované impulzní odezvy se zvoleným zvukem dosáhneme vytvoření nového signálu. Tento signál simuluje zvuk, který by byl vytvořen skutečným zdrojem zvuku v podobné místnosti, jako jsme zvolili v parametrech RIR generátoru.

Mezi parametry generátoru patří hlavně rozměry zvolené místnosti. Dále určení pozice zdroje a pozice mikrofonu.

Pro rozmnožení zvukových nahrávek bylo použito nastavení 5 místností. Námi zvolená nastavení jsou uvedena v tabulce č. 2.1.

[1] Archív zvuků rozbití skel je poskytován jako *public domain* – volné dílo.

[2] Nahrávky získané z kampaně CHiME jsou poskytována pod licencí *Creative Commons Attribution-NonCommercial-ShareAlike license, version 2.0.*

MÍSTNOST	1	2	3	4	5
ROZMĚRY [metry]	[5 3 2]	[5 3 2]	[5 3 2]	[5 3 2]	[5 3 2]
ZDROJ [metry]	[2 0 2]	[0 3 1]	[2 0 1]	[2 0 1]	[2 0 1]
MIKROFON [metry]	[2 2 2]	[2 2 2]	[2 2 2]	[5 3 2]	[5 3 2]
β [sekundy]	0.4	0.4	0.4	0.4	0.25

Tabulka 2.1: Výpis nastavení parametrů RIR pro jednotlivé virtuální místnosti

Zatím nezmíněným parametrem RIR generátoru je také zvolení odrazových vlastností místnosti – β. Tento parametr se nastavuje v sekundách a udává, za jak dlouho (v sekundách) se sníží intenzita odraženého zvukového signálu v místnosti o 60 dB. Při generování impulzní odezvy byla použita referenční hodnota uvedená v manuálu k RIR generátoru – 0.4 a hodnota 0.25, která by měla odpovídat místnosti s nižší odrazivostí.

2.1.2 Předzpracování

Získané zvuky byly pořizovány s různým nastavením. Zvuky rozbití skla byly zaznamenávány s 48kHz vzorkovací frekvencí při použití 32 bitové hloubky. Oproti tomu zvuky ruchů získané z projektu CHiME byly pořizovány s 16kHz vzorkovací frekvencí a též při použití 32 bitové hloubky. Nahrávací zařízení, kterým byly pořizovány vlastní nahrávky, bylo nastavené na 48 kHz vzorkovací frekvence při 32 bitové hloubce. Všechny nahrávky s vyšší vzorkovací frekvencí byly převzorkovány na 16 kHz (bitová hloubka nezměněna).

Zvukové nahrávky rozbití skla obsahovaly různě dlouhé zvuky ruchů (před a po rozbití). Tyto ruchy byly v rámci předzpracování z nahrávek odstraněny, tedy byly vybrány pouze části, které obsahují zvuk rozbití skla. Dlouhé záznamy ruchů byly rozstříhány na nahrávky o délce 3 vteřin. Tato délka byla zvolena, protože koresponduje s průměrnou délkou záznamů tříštění skla.

Nahrávky které obsahovaly zvuky škrábání na sklo či poklepávání na sklo byly z použitých nahrávek vyloučeny.

2.1.3 Rozdělení do datových množin

Po shromáždění dostatečného množství zvuků obou typů nahrávek (ruchů, rozbití skla) byly nahrávky rozděleny do dvou datových množin: trénovací a testovací.

Nahrávky byly do těchto množin rozděleny náhodně. Bylo pouze zajištěno, aby v obou datových množinách bylo stejné zastoupení zvuků obsahujících rozbití skla a ruchů. Žádné zvuky z trénovací množiny nebyly obsaženy v testovací množině a naopak.

Nahrávky vyprodukované RIR generátorem byly do celkové množiny zvuků zahrnuty také.

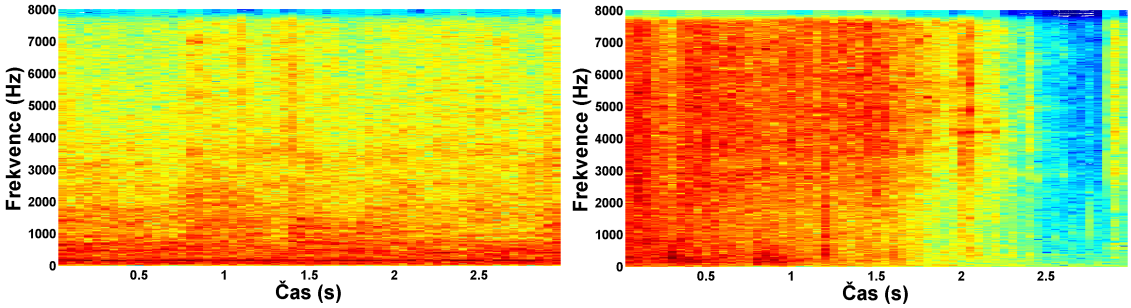
V rámci datové množiny byly zvuky rozděleny do složek podle toho, o jaký zvuk se jedná. V případě, že se jednalo o zvuky rozbití skel — složka 1, v opačné případě

— složka 0. Do trénovací množiny bylo zahrnuto celkem 352 nahrávek (8m a 35s – rozbití skla, 8m a 45s – ruchy) a do testovací množiny bylo zahrnuto 60 nahrávek (1m a 29s – rozbití skla, 1m a 30s – ruchy).

2.2 Analýza zdrojových nahrávek

Před návrhem samotného detektoru byla provedena analýza získaných zvukových záznamů. Analýza byla směřována ke spektrální oblasti signálů a dále byl porovnán průběh vývoje energie v čase (resp. v jednotlivých framech). Zjištěné charakteristiky byly porovnány mezi nahrávkami ruchů a nahrávkami rozbití skla. Z nahrávek ruchů byl vybrán zvuk pořízený v kavárně.

Porovnání bude znázorněno vždy dvojicí grafů, přičemž levý graf znázorňuje charakteristiku pro nahrávku ruchu a pravý graf pro rozbití skla. Byly zvoleny stejně dlouhé nahrávky, jejichž délka byla tři vteřiny.

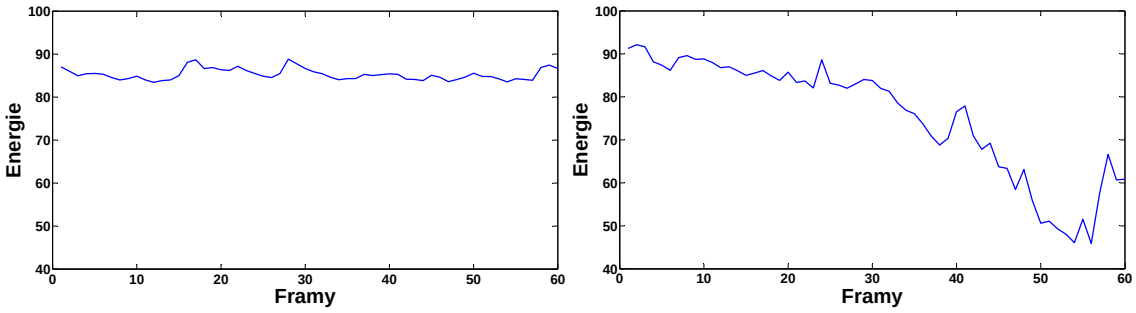


(a) Ruch (b) Sklo

Obrázek 2.1: Spektrogram

Dvojice obrázků č. 2.1 znázorňuje oba signály ve spektrální oblasti. Ze spektrogramů je patrné, že zvuk tříštění skla (2.1b) obsahuje vyšší frekvence oproti zvuku zastupujícího množinu ruchů (2.1a). Dále je z grafů patrné, že v případě rozbití skla se zastoupení jednotlivých frekvencí výrazně mění s časem. Postupem času ubývá vysokých frekvencí. Oproti tomu zvuk ruchu obsahuje frekvence, které signál provází po celé jeho délce, aniž by se výrazně měnily.

Dvojice grafů č. 2.2 vykresluje vývoj energie v jednotlivých framech. Při pohledu na grafy je vidět, že v případě zvuku ruchu (2.2a) je energie víceméně konstantní. Anomálie nacházející se kolem 18. a 30. framu byly pravděpodobně způsobeny zvukem cinknutí talířů. Energie zvuku tříštění skla (2.2b) má oproti tomu silně klesající průběh. Zvuk začíná proražením skla, to generuje zvuk o vysoké energii, která se poté postupně vytrácí. V poslední části grafu je patrný slabý vzestup energie, který je způsoben spadnutím většího kusu skla do střepů ležících pod rozbíjenou tabulí.



(a) Ruch (b) Sklo

Obrázek 2.2: Vývoj energie

2.3 Návrh detektoru

Návrh detektoru byl inspirován řešením úlohy pro rozpoznání mluvčího (blíže popsané v [1]). Při řešení této úlohy se snažíme ze vstupního signálu poznat, o jakého mluvčího se jedná. Trénovací signál je nejdříve převeden na matici příznakových vektorů, ze které je posléze učen pravděpodobnostní model GMM.

GMM se dá popsat jako lineární kombinace Gaussových rozložení. Parametry daných rozložení se snaží co nejlépe reprezentovat rozložení vstupních dat. Pro nalezení vhodných parametrů Gaussových rozložení se používá EM algoritmus. Naučený model je poté schopen určit pravděpodobnost toho, že námi vybraný vstupní vektor patří do zjišťované skupiny. V této práci bylo použito podobného postupu s tím, že klasifikátor GMM byl nahrazen neuronovou sítí.

Při analýze signálu (1.4) byly zjištěny specifické vlastnosti příslušící signálům obsahujícím zvuk tříštění skla. Jedná se o specifické rozložení frekvencí a také o vývoj energie v čase. V návrhu našeho rozpoznávače bylo využito obou vlastností a detektor byl rozdělen na dvě části.

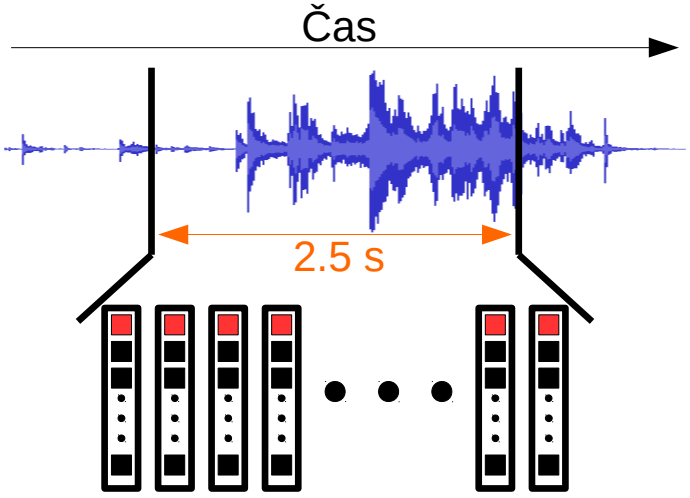
První část rozpoznávače je navržena tak, aby rozpoznávala určité rozložení frekvencí[3] v jednotlivých framech. Klasifikátor, který zajišťuje samotnou detekci, je tvořen neuronovou sítí.

Druhou částí detektoru je klasifikátor rozpoznávající určitý vývoj energie, ten je též tvořen neuronovou sítí.

Vzorová úloha rozpoznání mluvčího byla v našem případě mírně modifikována. Při rozpoznávání řečníka není potřeba do detektoru zanášet informaci o vývoji jednotlivých příznaků v čase. Postačí pouhá klasifikace izolovaných framů a na základě výsledků se určí kritérium, podle kterého určíme, o jakého řečníka se jedná. Při rozpoznávání rozbití skla jsme do detektoru zanesli složku, která sleduje vývoj energie vstupního signálu. Z tohoto pohledu se námi vytvořený systém podobá systémům pro rozpoznání řeči. V případě rozpoznávání řečových signálů je potřeba sledovat vývoj příznaků v čase.

Klasifikace vývoje energie i klasifikace spektrálního rozložení se provádí v rámci

[3]Zde je nutno podotknout, že v případě využití MFCC příznaků jde vlastně o rozložení kvefrencí (pracujeme s kepstrem).



Obrázek 2.3: Znázornění parametrizace v reálném čase. Signál je vyobrazen modrou barvou. Výstupem této operace jsou příznakové vektory. První složka vektoru je energie (znázorněna červeně).

určeného časového úseku. Délka tohoto úseku byla nastavena na 2,5 vteřiny. Jedná se o zkrácenou průměrnou dobu, po kterou trvá zvuk rozbití skla. Dobu, po kterou dochází k rozbití a následnému rozsypání skla, tedy bez dodatečného padání větších zbylých kusů. Zvuk v této době je segmentován na framy a z každého z nich je vypočten příznakový vektor MFCC (viz obr. 2.3).

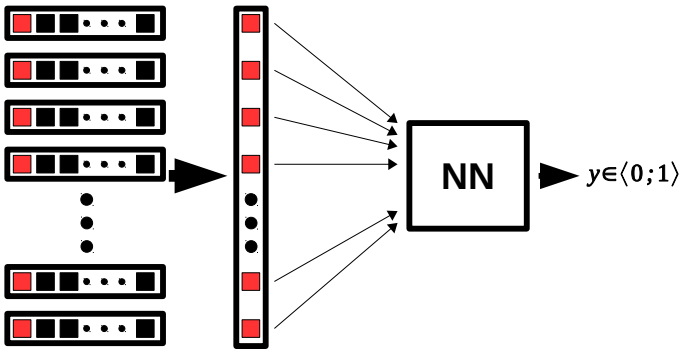
2.3.1 Klasifikace dynamického průběhu energie

Klasifikátor průběhu energie slouží k určení, zda rozpoznávaný signál má podobný průběh energie jako signál obsahující zvuk rozbití skla, či nikoli.

Předpokládáme, že všechny framy z analyzovaných 2,5 vteřin jsou převedeny na MFCC příznakové vektory. První složka z MFCC příznakového vektoru je vyjádření energie signálu. Jsou vybrány energie všech framů. Z těchto energií je postupně vytvořen nový vektor. Složky nového vektoru jsou nakonec ještě normalizovány. Normalizace je provedena vydělením všech složek vektoru jeho maximální hodnotou. Normalizovaný vektor energií slouží jako vstup do neuronové sítě (viz. 2.4).

Samotná klasifikace je prováděna dopřednou neuronovou sítí, která má ve výstupní vrstvě dva neurony. Součet výstupu obou neuronů je normalizován na 1 (k tomu slouží aktivační funkce SoftMax). Výstup prvního neuronu určuje, s jakou pravděpodobností vstupní vektor (průběh energií v 2,5 vteřinách) odpovídá průběhu energií zvuku tříštění skla. Druhý výstup je obdobný s tím rozdílem, že se jedná o doplňkovou pravděpodobnost, s jakou vstup odpovídá vstupu ruchů.

Výstupem z této části detektoru je zvolen výstup z prvního neuronu ve výstupní vrstvě.

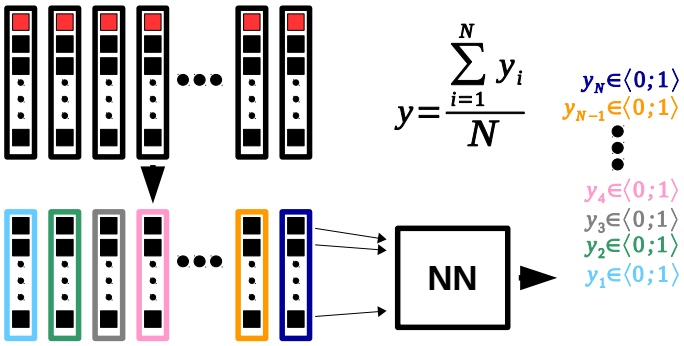


Obrázek 2.4: Operace klasifikace vývoje energie. NN znázorňuje neuronovou síť.

2.3.2 Klasifikace spektrálního rozložení

Ve stejném časovém úseku, jako v případě vyhodnocování průběhu energie (2,5 vteřiny), probíhá klasifikace na základě vlastností spektra (resp. kepstra). Výsledné určení, zda se v daném okně nachází signál tvořený zvukem rozbití skla, probíhá ve 2 krocích.

Nejdříve jsou do klasifikátoru postupně vkládány jednotlivě příznakové vektory pro jednotlivé framy. Tyto vektory jsou zbaveny první složky (energie). Je nežádoucí, aby tato část klasifikace byla závislá na intenzitě vstupního signálu. Pro každý frame je zjištěna pravděpodobnost, zda odpovídá framům rozbití skla. Pro klasifikaci jednotlivých framů je použita neuronová síť (viz 2.5). Mimo vstupní vrstvy se jedná o obdobnou síť jako v případě klasifikátoru průběhu energie.



Obrázek 2.5: Znázornění části pro rozpoznávání spektra. Celkový výstup (y) je průměr, ze všech složek y_i.

Výstupem z této klasifikační části je aritmetický průměr ze všech spočítaných pravděpodobností všech framů v detektorovém okně.

2.3.3 Architektura detektoru

Celkový detektor vyhodnocuje signál v rámci 2,5 vteřinového okna. V rámci tohoto okna jsou najednou použity obě navržené části detektoru. Z každé části je definován výstup. Obě výstupní hodnoty jsou reprezentovány číslem a nabývají hodnot od 0 do

1. Pro každou část je zvlášť nastavena námi požadovaná citlivost, práh. V případě, že obě výstupní hodnoty překročí tento definovaný práh, tak je analyzovaný signál vyhodnocen jako zvuk rozbití skla.

Vzhledem k tomu, že je pevně stanovená délka vstupního vektoru neuronové sítě, tak nelze změnit délku detektorového okna. Pro změnu by bylo zapotřebí změnit parametry sítě určené pro rozpoznání průběhu energie a síť přetrénovat.

2.4 Trénování

Klasifikátory detektoru bylo potřeba před použitím natrénovat. Pro trénování bylo zapotřebí vytvořit datové množiny: trénovací a testovací. Předzpracované zvuky již byly rozděleny do složek — trénovací a testovací (viz 2.1.3). Dále bylo potřeba každý zvuk segmentovat a parametrizovat. Vzhledem k tomu, že bylo využito metody učení s učitelem, tak každý frame byl označen informací, zda patří ke zvukům rozbití skla či ruchů.

Všechny operace prováděné s neuronovými sítěmi byly zajištěny externí knihovnou pro C# – Encog[4] [17]. V rámci této knihovny je k dispozici několik učících algoritmů. Při trénování sítí byl použit tzv. RPROP algoritmus (Resilient Propagation). Tento algoritmus je modifikací klasického algoritmu zpětného šíření chyby. RPROP se vyznačuje rychlejší konvergencí [8]. Mimo vyšší rychlosti konvergence je také výhodou, že algoritmus nevyžaduje nastavení parametru rychlosti učení (η). Algoritmus pracuje s mechanismy, které jej upravují během učení automaticky (více o algoritmu zde: [8, 9]).

2.4.1 Vytvoření datových množin

Návrh detektoru obsahuje dvě neuronové sítě, které bylo potřeba natrénovat. Každá ze sítí má jiný charakter, a proto bylo potřeba vytvořit dva typy datových množin. V obou případech se využilo stejných parametrů segmentace i parametrizace.

Velikost jednoho framu byla zvolena na 800 vzorků (při 16 kHz se jedná o dobu 50 ms). V rámci 2,5 vteřinového okna bylo tedy segmentací získáno 50 framů.

Pro účely parametrizace byla vytvořena třída pro C#, která byla schopna generovat MFCC příznakové vektory z vloženého signálu. Třída byla postavena tak, aby měla široké možnosti nastavení, včetně změny rozložení filtrů (to je nativně zvoleno na Melovské stupnici). Implementace třídy vychází z manuálu, který je k dispozici zde: [18, 3]. Výstupní vektory byly porovnány s výstupy jiné implementace, která byla naprogramována ve vývojovém prostředí programu Matlab. Základní parametry byly zvoleny podle následujícího výčtu:

- Frekvenční rozsah: 300–8000 Hz
- Počet filtrů: 20
- Výsledných koeficientů: 12

[4]Framework je poskytován pod licencí – *Apache License 2.0*

- Melovské rozdělení filtrů

Datové množiny dynamického průběhu energie

Pro trénování sítě rozpoznávající průběh energie byla vytvořena trénovací a testovací datová sada.

V první řadě byly načteny zvukové nahrávky z obou skupin, ruchů i rozbití skla. Nahrávky byly poté oříznuty na velikost okna detektoru (2,5 vteřiny) a následně byly segmentovány podle výše uvedených parametrů. Z každého framu byla spočítána energie. Ze všech energií byl vytvořen nový vektor. Tento vektor byl opatřen označením skupiny. V případě, že se jednalo o rozbití skla (složka 1), tak byl k danému vektoru přiřazen vektor — $(0, 1)$ (viz poznámka o požadované hodnotě při řešení klasifikačního problému v: 1.8.3 – Algoritmus zpětného šíření chyby). V opačném případě byl k vektoru přiřazen vektor — $(1, 0)$.

Každý vektor energií spolu s vektorem označení tvoří jeden trénovací/testovací vzor v trénovací/testovací množině.

Datové množiny spektrálního rozložení

Pro účely vytváření datových sad pro trénovaní klasifikátoru spektrálního rozložení byl vytvořen nahrávací program.

Tento program po libovolně dlouhou dobu nahrával zvuk z připojeného mikrofonu a v náhodných časových intervalech přehrával zvukové soubory. Tyto soubory byly vybírány náhodně z obou skupin (sklo/ruch). Program má paměť již přehraných souborů. Pokud již nějaký soubor byl přehrán, tak do vyčistění paměti není znova přehráván. Čištění paměti je prováděno po přehrání všech souborů v příslušné složce (trénovací/testovací).

Výstupem z programu je datový soubor, který obsahuje matici příznakových vektorů spolu s jejich označením (target) pro celou nahrávku. Vedlejším výstupem je samotný zvukový soubor s celkovou nahrávkou a soubor s označením, kde se v nahrávce nacházejí zvuky tříštění skel. Tento výstup může sloužit v případě, kdy je potřeba změnit nastavení parametrů segmentace/MFCC a není potřeba nebo není možné znovu nahrávat zvukový soubor.

Program byl nastaven na to, aby z každých 800 vzorků nahraného signálu vytvořil frame a z každého framu poté extrahoval MFCC příznakový vektor. Každý frame byl označen skupinou zvuků, do které patří. V případě, že při nahrávání byl přehráván zvuk rozbití skla (složka 1), tak se frame označil vektorem — $(0, 1)$. Ve všech ostatních případech (složka 0), ruch nebo ticho, byl frame označen vektorem — $(1, 0)$. Jeden vzor datové množiny byl v tomto případě tvořen MFCC příznakovým vektorem (zbaveného první složky) spolu s vektorem označení skupiny.

Pro účely trénování byly vytvořeny dvě nahrávky/datové sady. Jedna obsahovala vybrané zvuky pro trénování a druhá pro testování. Výsledná trénovací nahrávka byla dlouhá 57m a 17s. Testovací nahrávka byla dlouhá 14m a 52s.

2.4.2 Vliv nastavení parametrů neuronových sítí na výsledek učení

Tato část práce je věnována ukázkám průběhu trénování pro různé parametry neuronových sítí. V rámci tohoto dokumentu je zde uvedeno omezené množství výsledků. Byly vybrány výsledky, které vedly k nejoptimálnějšímu nalezenému řešení.

Vyobrazení jednotlivých záznamů trénování bude mít následující podobu. Pro každé nastavení neuronové sítě budou zobrazeny dva grafy. Tyto grafy budou umístěny vedle sebe a ukazují různé pohledy na stejná data. Graf vlevo vždy znázorňuje učební proces na celém intervalu chybovosti, $\langle 0; 1 \rangle$. Graf umístěný v pravé části zobrazuje výřez z grafu umístěném vlevo. Zaměřuje se pouze na trénovací část, která již není zatížená počátečním náhodným neoptimalizovaným nastavením vah.

V grafech je zanesena informace o vývoji chybovosti sítě v průběhu učení. Chybovost je procentuální vyjádření chybně klasifikovaných vzorů.

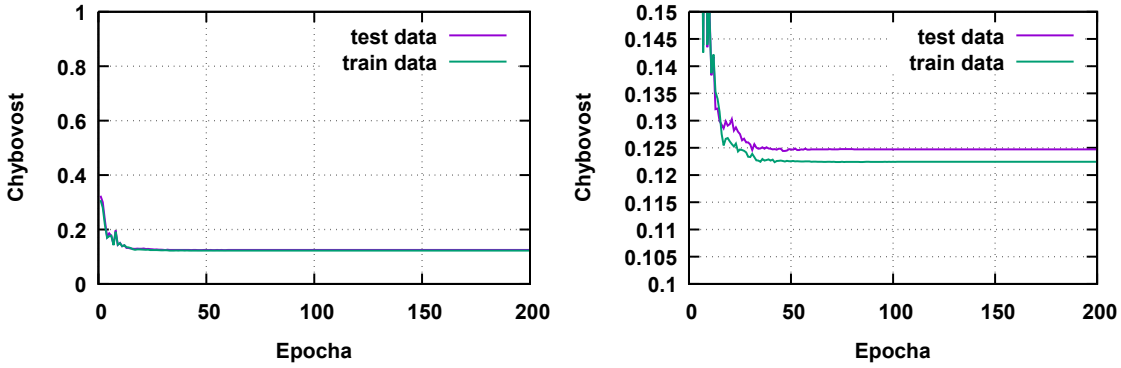
Nastavení datových množin bylo zvoleno podle základního nastavení uvedeného v sekci 2.4.1.

Průběhy trénování sítě pro klasifikaci spektrálního rozložení

Vzhledem k tomu, že pro trénování sítě využité ke klasifikaci spektrálního rozložení máme větší datové množiny, tak výsledky trénování uvedeme jako první.

Parametry neuronové sítě příslušící ke grafům jsou uvedeny v popisku pod dvojicí obrázků.

Uvedené neuronové sítě mají stejný počet neuronů ve vstupní vrstvě – 11. Výstupní vrstva byla ve všech případech tvořena dvěma neurony, reprezentující skupiny: ruchy, rozbití skla. Aktivační funkcí byl zvolen hyperbolický tangens.

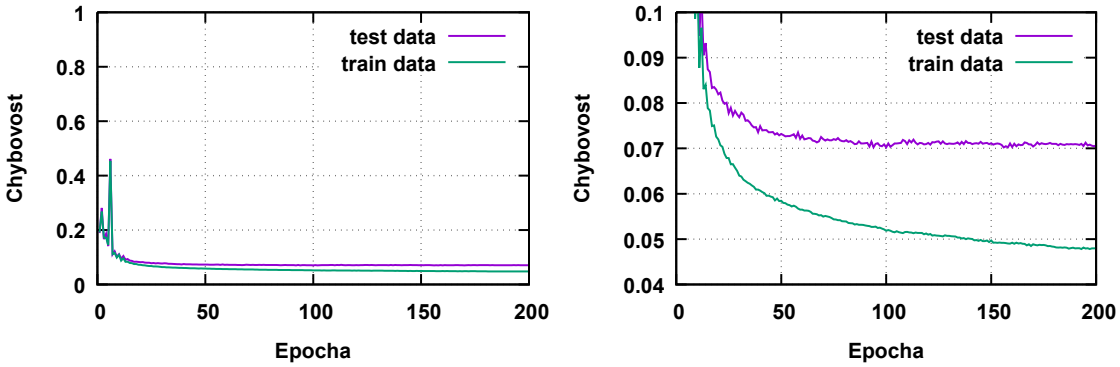


(a) Celý rozsah chybovosti (b) Výřez – rozsah chybovosti: 0–15%

Obrázek 2.6: Bez vnitřní skryté vrstvy. Při učení jsou upravovány pouze váhy, které jsou umístěné na spojích mezi vstupní a výstupní vrstvou. I přes uvedenou a v celku dobře vypadající chybovost na testovacích datech — 12,5 % je síť nepoužitelná ve výsledném rozpoznávači. Síť po cca 50 epochách dosáhla stavu maximální naučenosti.

Měření bylo ve všech případech provedeno opakovaně. Vždy byl vybrán nejlepší výsledek.

První dvojice grafu (obr. 2.6) byla zvolena pro ukázku učení bez jakéhokoli zásahu do parametrů sítě. Následující grafy ukazují výsledky měření pro sítě po přidání skrytých vrstev.

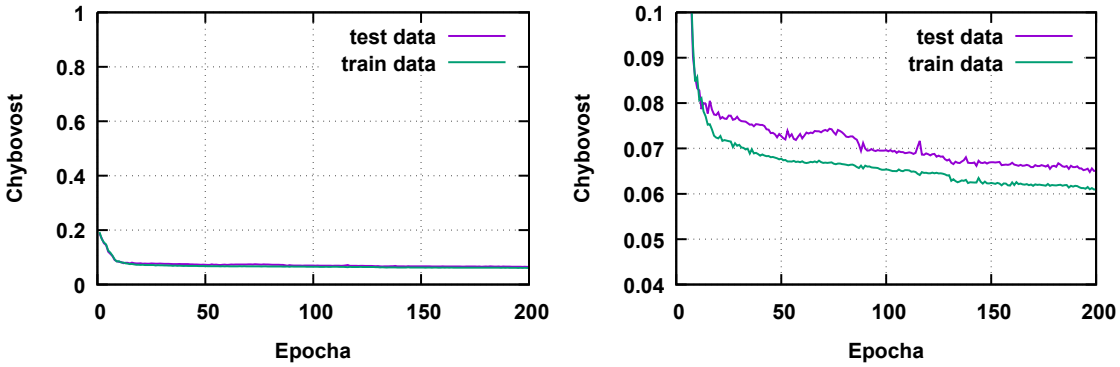


(a) Celý rozsah chybovosti (b) Výřez – rozsah chybovosti: 4–10 %

Obrázek 2.7: Dvě skryté vrstvy: 64 a 64 neuronů. Chybovost sítě byla snížena. Objevuje se problém s přeučením sítě. Síť se každou další epochou učí (klesá chybovost trénovací množiny), avšak hladina chybovosti testovací množiny se již nesnižuje.

Grafy zobrazeny v obr. 2.7 ukazují na počínající přeučení sítě. Tato síť již je použitelná ve výsledném rozpoznávači. Jedním ze způsobů odstranění přeučení je rozšíření trénovací množiny. Již nemáme další vzorky pro možnost rozšíření, a proto se pokusíme síť přinutit generalizace snížením počtu neuronů ve skrytých vrstvách.

Parametry sítě zobrazené ve dvojici obrázků č. 2.8 budou použity pro následnou optimalizaci příznakových vektorů.

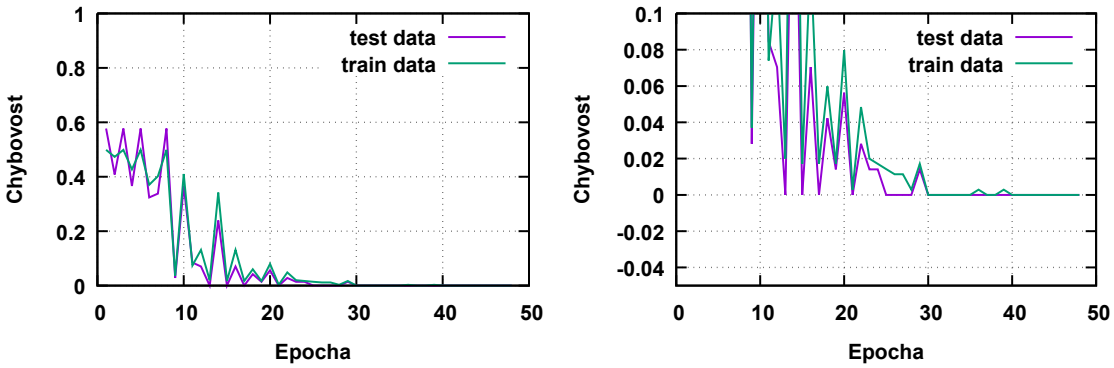


(a) Celý rozsah chybovosti (b) Výřez – rozsah chybovosti: 4–10 %

Obrázek 2.8: Dvě skryté vrstvy: 8 a 4 neurony. Chybovost sítě byla opět snížena. Síť vykazuje dobré vlastnosti učení. Chybovost sítě má klesající trend pro obě množiny.

Průběh trénovaní sítě pro klasifikaci dynamiky energie

Vzhledem k jednoduchosti úlohy a také vzhledem k tomu, že datové množiny jsou při této úloze značně omezené, tak zde uvedeme pouze výsledek učení sítě pro jedno nastavení (viz obrázek 2.9). Uvádění výsledku pro vícero nastavení sítí zde nemá žádný přínos. Vzhledem k malým datovým množinám a jednoduchosti úlohy není potřeba uvažovat sítě s vysokým počtem vrstev ani neuronů.



(a) Celý rozsah chybovosti (b) Výřez – rozsah chybovosti: 0–10 %

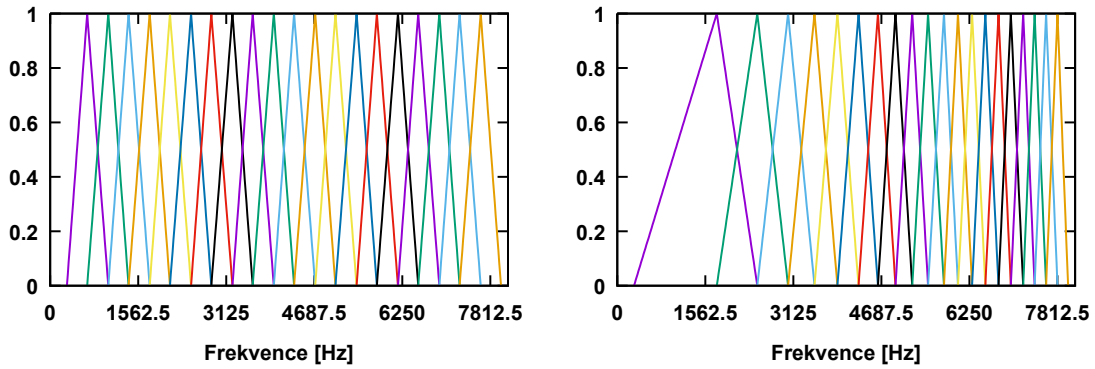
Obrázek 2.9: Jedna skrytá vrstva: 16 neuronů. Učení sítě je rychlé. Učení bylo ukončeno před dosažením maximálního počtu epoch. Síť dosáhla minimální požadované chybovosti.

2.4.3 Vliv nastavení parametrů příznakových vektorů na výsledek učení

Doposud byly v práci využívány MFCC příznakové vektory, které byly vyvinuty a optimalizovány pro zpracovávání řečových signálů. Jak již bylo uvedeno v teoretické části, tak MFCC příznaky jsou založeny na logaritmickém vnímání frekvencí lidského ucha. Klíčovým prvkem MFCC je melovské rozložení filtrů (téměř logaritmické). To zajišťuje vyšší rozlišení pro nízké frekvence a nižší rozlišení pro vysoké frekvence. Funkčnost tohoto řešení byla ověřena mnohým testováním a úspěšnou aplikací při zpracování řeči.

V této práci byl proveden pokus o optimalizaci metody tvorby kepstrálních příznaků za účelem zvýšení úspěšnosti rozpoznávání zvuků rozbití skla. Ze spektrogramu, který je vyobrazený v sekci analýzy nahrávek (viz sekce 2.2, obrázek 2.1) je možné pozorovat, že zvuk rozbití skla obsahuje vyšší frekvence než například ruch. Z tohoto důvodu byl odvozen předpoklad, že by vyšší rozlišení (více filtrů) pro vyšší frekvence mohlo snížit chybovost klasifikace. Bylo proto upraveno rozložení banky filtrů.

V prvním pokusu bylo rozložení frekvencí upravené tak, aby bylo lineární (středové frekvence filtrů jsou stejně daleko od sebe). Takto upravenou banku filtrů je možné vidět na obrázku č. 2.10a. V druhém pokusu byla banka filtrů upravena tak,



(a) Lineární rozdělení filtrů (b) Kvadratické rozdělení filtrů

Obrázek 2.10: Upravené rozložení banky filtrů.

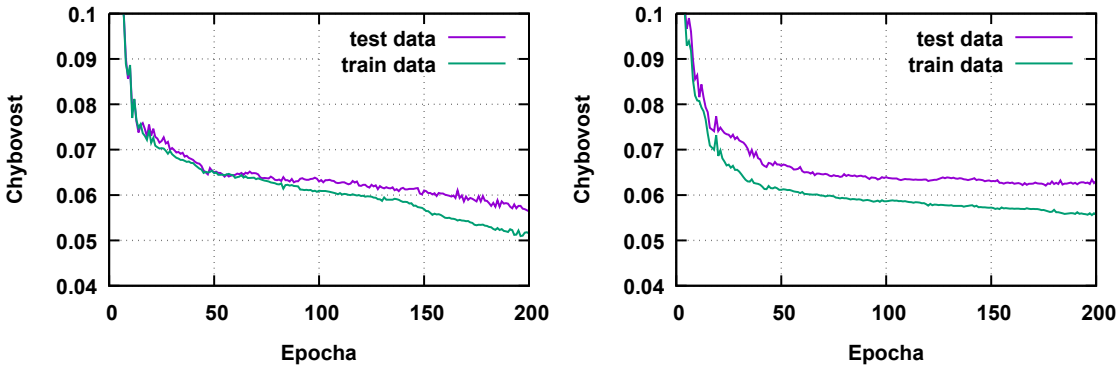
aby bylo rozlišení filtrů vyšší ve vyšších frekvencích. Upravená banka filtrů měla kvadratické rozložení (obr. č 2.10b).

Výsledky pokusu jsou zaznamenané v jedné dvojici grafů (obr č. 2.11). Z obou výsledků je vybrán úsek určité chybovosti.

Pro tento pokus byly použity stejné zvukové nahrávky trénovacích i testovacích zvuků, pro které byly pouze přepočítány příznakové vektory.

Po použití lineárního rozdělení banky filtrů síť vykazuje dobré vlastnosti učení. Chybovost sítě byla snížena pod hranici 6 % (obr. 2.11a).

Kvadratické rozložení filtrů již zlepšení nepřineslo. Síť vykazuje horší vlastnosti učení. Chybovost sítě byla snížena, ale zůstává nad hranicí 6 % (obr. 2.11b).

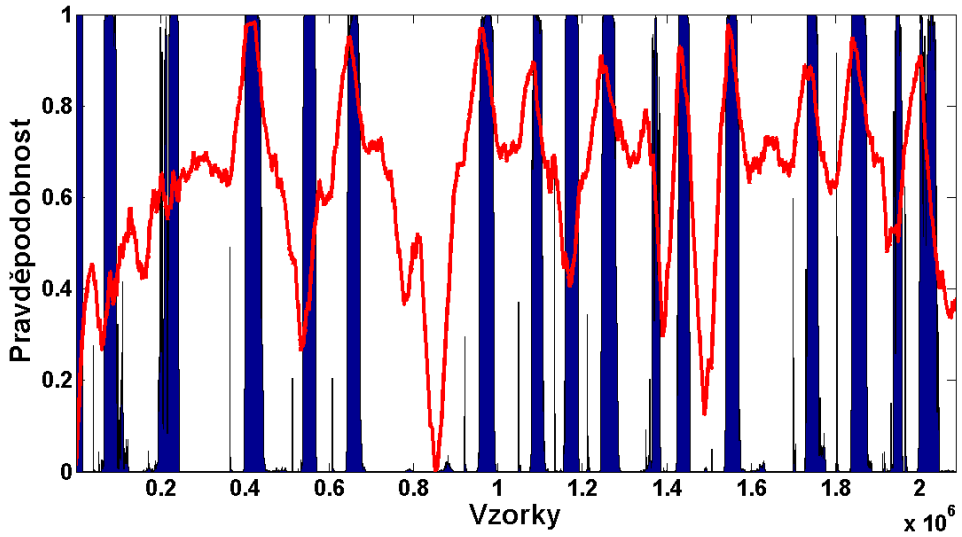


(a) Lineární rozdělení filtrů (b) Kvadratické rozdělení filtrů

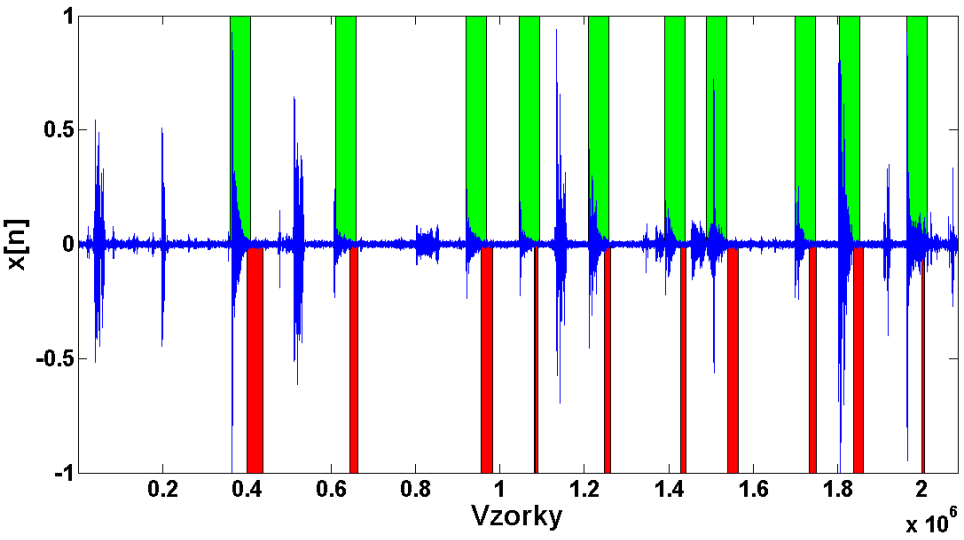
Obrázek 2.11: Průběhy trénování pro upravené banky filtrů.

2.5 Testování detektoru

Tato část se zabývá testováním vytvořeného detektoru. Při testování byl detektor spuštěn v režimu rozpoznávání. Detektor snímal běžné zvuky z místnosti a v náhodných intervalech byly ručně spouštěny nahrávky zvuků rozbití skla.



Obrázek 2.12: Ukázka výstupu z jednotlivých částí detektoru. Modré sloupce znázorňují výstup z části pro rozpoznávání vývoje energie. Červená spojnice představuje výstup z části pro rozpoznávání spektrálních vlastností.



Obrázek 2.13: Ukázka rozpoznávání testovacího zvukového signálu. Signál je znázorněn modrou spojnicí. Zelené sloupce v horní polovině obrázku značí, kde se v signále nacházejí zvuky rozbití skla. Červené sloupce v dolní polovině obrázku značí, kde detektor rozpoznal zvuk rozbití skla.

Nastavení prahů pravděpodobností jednotlivých částí:

- Rozpoznávání vývoje energie: 0.8
- Rozpoznávání vlastností spektra: 0.85

Během detekování probíhal záznam výstupů jednotlivých částí detektoru. Tento záznam je zobrazen na obrázku č. 2.12.

Obrázek č. 2.13 ukazuje: záznam vstupního signálu, označení míst, kde se nacházejí zvuky rozbití skla a označení míst, kde reagoval detektor.

Během nahrávání bylo spuštěno deset testovacích nahrávek rozbití skla. Ve všech deseti případech detektor reagoval podle očekávání.

Je možné pozorovat, že detektor reaguje s menším zpožděním. To je způsobeno částí určenou pro detekci vývoje energie. Zvukový signál rozbití skla musí celý procházet detektorovým oknem, aby tato část detektoru zareagovala. Část detektoru odpovědná za detekci na základě spektra reaguje hned po vstupu prvního framu rozbití skla do detektoru.

2.6 Zhodnocení výpočetní náročnosti

Na výpočetní náročnost můžeme nahlížet ze dvou pohledů: náročnost trénování a náročnost samotného detektoru v režimu rozpoznávání. Obě tyto náročnosti jsou velice závislé na zvolené architektuře sítě.

Trénování je oproti rozpoznávání (lze provádět v reálném čase) náročnější operací. Vzhledem k těžko vyčíslitelnému celkovému počtu operací, zde pro představu uvedeme orientační časovou náročnost trénování. Doba trénování je uvedená pouze pro trénování sítě pro rozpoznávání spektrálního rozložení. Čas potřebný k trénování sítě pro klasifikaci vývoje energie byl řádově v sekundách (malá trénovací množina, cca 40 - 50 epoch).

Použitá sestava:

- CPU – AMD 10-6800K (4 × 4,1 GHz)
- RAM – 8 GB (2132 MHz)

Skryté vrstvy (počet neuronů)	(128+128)	(64+64)	(8+4)
čas[s]/epocha	10	4	1

Tabulka 2.2: Pro měření časové náročnosti byly použity množiny uvedené v: 2.4.1 – Datové množiny spektrálního rozložení.

Mnohem důležitější je pohled na výpočetní náročnost detektoru v režimu rozpoznávání. Tento detektor by měl být napájen bateriemi, a proto má silně omezený výpočetní výkon.

Náročnost rozpoznávání je dána strukturou klasifikátoru. Vzhledem tomu, že pro klasifikaci používáme neuronové sítě, tak je počet operací přesně stanoven architekturou dané sítě.

V detektoru jsou použity dvě nezávislé neuronové sítě. Výčet prováděných operací je pro každou zvlášť rozdělen.

Počet operací rapidně roste při použití sítě s vyšším počtem neuronů.

Klasifikace spektrálního rozložení

Pro ilustraci budeme uvažovat síť, která má 11 vstupních neuronů, 2 skryté vrstvy (8 a 4 neurony) a 2 neurony ve výstupní vrstvě. Počet operací při průchodu vzruchového signálu neuronovou sítí je dán definicí neuronu. Výpočet celkového počtu operací je rozepsán v následujícím výčtu:

- Operací nelineárního zobrazení: $11 + 8 + 4 + 2 = 25$
- Operací násobení: $11 + 8 * 12 + 4 * 9 + 2 * 5 = 153$
- Operací součtů: shodný s počtem násobení (každý výsledek součinu vstupu s váhou je zahrnut do agregační funkce)

Klasifikace vývoje energie

Pro ilustraci budeme uvažovat síť, která má 50 vstupních neuronů, 1 skrytou vrstvu (16 neuronů) a 2 neurony ve výstupní vrstvě. Následující výčet ukazuje rozpis jednotlivých operací:

- Operací nelineárního zobrazení: $50 + 16 + 2 = 68$
- Operací násobení: $50 + 16 * 51 + 2 * 17 = 900$
- Operací součtů: shodný s počtem násobení

Závěr

Předložená práce se zabývá problematikou rozpoznání akustické události. V první kapitole práce byly vysvětleny základní pojmy, metody a principy související s danou problematikou. Pozornost byla věnována analýze akustických signálů, extrakce příznakových vektorů a především technologii umělých neuronových sítí, která je stěžejním pilířem druhé části práce — vytvoření funkčního detektoru rozbití skla.

Nedílnou součástí práce je část popisující zpracování zdrojových dat. Data byla získána z internetových databází zvuků a jejich autentičnost byla ověřena porovnáním s námi nahranými nahrávkami. Zdrojové zvuky byly podrobeny předzpracování a hlubší analýze. Během analýzy byly nalezeny znaky typické pro zvuky rozbití skla a tyto znaky byly využity při navrhování architektury detektoru.

Hlavním výstupem práce je samotný návrh detektoru. Výsledná architektura detektoru se skládá ze dvou částí. První část provádí klasifikaci podle vývoje energie. Tato část zajišťuje, aby vstupní zvuk, klasifikovaný jako zvuk rozbití skla, nabýval specifického dynamického vývoje energie. Druhá část provádí klasifikaci na základě spektrálních vlastností signálu. Při tvorbě této části byla nalezena inspirace v úloze rozpoznání řečníka. V obou částech byly využity jednou spočítané MFCC příznakové vektory. Výsledný návrh má vysoké možnosti variability v podobě nastavení parametrů příznakových vektorů a neuronových sítí.

Zvolené parametry mají silný vliv na úspěšnost detekce a část práce je věnována porovnávání výsledků naměřených pro různá nastavení. Parametry MFCC příznaků byly zvoleny na základě doporučených nastavení pro zpracování řečových signálů. Optimalizace neuronových sítí probíhala systematicky volbou parametrů podle analýzy úspěšnosti trénování a následné detekce. Práce také obsahuje popis optimalizace kepstrálních příznakových vektorů.

Pro vývoj detektoru bylo zapotřebí naprogramovat skripty pro přípravu datových množin a trénovaní neuronových sítí. Součástí výstupu práce je naprogramovaný samotný detektor. Jedná se o GUI (Windows Forms) aplikaci, která umožňuje analyzovat zvuk z mikrofonu připojeného k PC. Vstupní zvuk je klasifikován a případná detekce rozbití skla je v programu názorně zobrazena. Veškeré operace s neuronovými sítěmi byly zajištěny externí knihovnou určenou pro strojové učení – Encog [17].

Některé dílčí části, hlavně tedy ukázka funkčního detektoru, byly prezentovány ve firmě Jablotron Alarms a.s.

Na závěr bych rád uvedl, že tento zvolený způsob detekce nemusí být optimální. Během práce na detektoru byla objevena řada nápadů, které se již nestihly realizovat. Jedním z nich je využití neuronových sítí s učením bez učitele. Neuronová síť se při

tomto učení snaží najít závislosti ve vstupních vzorech a podle toho je třídí do shluků. Tyto shluky poté tvoří klasifikační skupiny a je podle nich definován výstup. Mohlo by se ukázat, že se zvuk rozbití skla dá rozdělit na více po sobě jdoucích částí, které by bylo možné detekovat postupně. Na tomto základě by mohl být vytvořen konečný automat, který by po postupné detekci všech částí skončil v koncovém stavu a byl by spuštěn alarm. Na tomto principu pracují systémy zpracovávání řeči (HMM).

Reference

[1] NOUZA, Jan, Zbyněk KOLDOVSKÝ a Robert VÍCH (eds.). *Řeč a počítač: principy hlasové komunikace, úlohy, metody a aplikace : sborník článků*. Liberec: Technická univerzita v Liberci, 2009. ISBN 978-80-7372-548-8..

[2] PORAT, Boaz. 1997. *A course in digital signal processing*. New York: John Wiley & Sons, xxvi, 602 s. ISBN 0471149616.

[3] HUANG, Xeudong, Alex ACERO a Hsiao-Wuen HON. *Spoken language processing: a guide to theory, algorithm and system development*. Upper Saddle River: Prentice Hall, 2001. ISBN 0-13-022616-5.

[4] Přispěvatelé Wikipedie, A/D převodník [online], In: *Wikipedie: Otevřená encyklopedie*, c2014, Datum poslední revize 21. 11. 2014, 17:14 UTC, [citováno 20. 04. 2016]. Dostupné také z: `http://cs.wikipedia.org/w/index.php?title=A/D_p%C5%99evodn%C3%ADk&oldid=12021260`

[5] TUČKOVÁ, Jana. *Vybrané aplikace umělých neuronových sítí při zpracování signálů*. Vyd. 1. Praha: České vysoké učení technické v Praze, 2009. ISBN 978-80-01-04229-8.

[6] NOVÁK, Mirko. *Umělé neuronové sítě: teorie a aplikace*. Vyd. 1. Praha: C.H. Beck, 1998. Učebnice informatiky. ISBN 80-717-9132-6.

[7] ABU-MOSTAFA, Yaser. Lecture 10 (Neural Networks) [lecture]. Pasadena, CA, USA: Hameetman Auditorium at Caltech, May 3, 2012. In: *Youtube* [online]. Zveřejněno 06. 05. 2012 [cit. 2016-04-25]. Kanál uživatele caltech. Záznam dostupný z: `https://youtu.be/Ih5Mr93E-2c`

[8] RIEDMILLER, M. a H. BRAUN. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: *IEEE International Conference on Neural Networks* [online]. IEEE, 1993, s. 586-591 [cit. 2016-05-09]. DOI: 10.1109/ICNN.1993.298623. ISBN 0-7803-0999-5. Dostupné z: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=298623`

[9] HEATON, Jeff. *Introduction to neural networks with Encog3 in C#*. St Louis (MO, USA): Heaton Research, 2011. ISBN 9781604390261. Dostupné také z: `https://s3.amazonaws.com/heatonresearch-books/free/Encog3CS-User.pdf`

[10] WINSTON, Patrick. Lecture 12A: Neural Nets [video, open courseware]. In: *Massachusetts Institute of Technology: MIT OpenCourseWare* [online]. 2010 [cit. 2016-04-25]. Dostupné z: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/lecture-12a-neural-nets/. License: Creative Commons BY-NC-SA.

[11] HAYKIN, Simon S. *Neural networks and learning machines*. 3rd ed. New York: Prentice Hall, c2009. ISBN 0131471392.

[12] FONT, Frederic, Gerard ROMA a Xavier SERRA. Freesound technical demo. In: *Proceedings of the 21st ACM international conference on Multimedia - MM '13*. New York, New York, USA: ACM Press, 2013, , 411-412. DOI: 10.1145/2502081.2502245. ISBN 9781450324045. Dostupné také z: http://dl.acm.org/citation.cfm?doid=2502081.2502245

[13] GEVAROY. Breaking Glass. In: *Freesound* [online]. 2013 [cit. 2016-05-04]. Dostupné z: https://freesound.org/people/gevaroy/packs/10884/

[14] VINCENT, Emmanuel, WATANABE, Shinji, Jon BARKER a Ricard MARXER. An analysis of environment, microphone and data simulation mismatches in robust speech recognition. Submitted to: *Computer Speech and Language*, 2016.

[15] *CHiME Challenge: The 4th CHiME Speech Separation and Recognition Challenge* [online]. 2016 [cit. 2016-05-05]. Dostupné z: http://spandh.dcs.shef.ac.uk/chime_challenge/index.html

[16] HABETS, E.A.P. *Room impulse response generator*. Technische Universiteit Eindhoven, 2010. Dostupné také z: https://github.com/ehabets/RIR-Generator/blob/master/rir_generator.pdf

[17] HEATON, Jeff. Encog: Library of Interchangeable Machine Learning Models for Java and C#. In: *Journal of Machine Learning Research*. Volume 16. 2015. 1243-1247.

[18] Mel Frequency Cepstral Coefficient (MFCC) tutorial. In: *Practical cryptography* [online]. 2009 [cit. 2016-05-04]. Dostupné z: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

Obsah přiloženého CD

- Diplomová práce v elektronické podobě (PDF)
- Videonahrávky pořízené při získávání zvukových nahrávek
- Program - Akustický detektor rozbití skla
- Manuál k programu
- Projekt programu (vývojové prostředí Microsoft Visual Studio 2015)