VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

DETEKCE VYBRANÝCH ZVUKOVÝCH UDÁLOSTÍ V REÁLNÉM PROSTŘEDÍ
DETECTION OF SELECTED AUDIO EVENTS IN A REAL ENVIRONMENT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Alexander Kowolowski

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Jiří Přinosil, Ph.D.

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Audio inženýrství**
Ústav telekomunikací

Student: Alexander Kowolowski ***ID:*** 175592
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Detekce vybraných zvukových událostí v reálném prostředí

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte možnosti automatické detekce zvukových událostí ve zvukovém signálu s využitím standardních akustických a statistických příznaků. Proveďte teoretický návrh algoritmu pro detekci vybraných zvukových událostí. Proveďte implementaci navrženého algoritmu a ověřte jeho spolehlivost na nahrávkách z reálného prostředí.

DOPORUČENÁ LITERATURA:

[1] MESAROS, Annamaria; HEITTOLA, Toni; VIRTANEN, Tuomas. TUT database for acoustic scene classification and sound event detection. In: 24th European Signal Processing Conference. 2016.

[2] FOGGIA, Pasquale, et al. Reliable detection of audio events in highly noisy environments. Pattern Recognition Letters, 2015, 65: 22-28.

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: Ing. Jiří Přinosil, Ph.D.
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

Abstrakt

Tato práce se zabývá metodami pro rozpoznávání nebezpečných událostí, v tomto případě výstřelů, v reálném prostředí. Nejdříve byla vytvořena testovací a trénovací databáze zvuků ze zadané databáze MIVIA. V této databázi byly soubory obsaženy v šesti verzích odstupu užitečného signálu od šumu, takže následné testování vybraných metod probíhalo pro různě zašuměné soubory a bylo zjištěno, že některé metody jsou například přesnější u čistších nahrávek než jiné, ale už méně přesné u více zašuměných. Pro extrakci typických příznaků ze vstupního zvuku byla vždy použita metoda melovských kepstrálních koeficientů. V práci jsou na vytvořených databázích postupně testovány metody podpůrných vektorů a klasifikace spojením většího počtu slabých klasifikátorů. Tyto metody jsou poté dále optimalizovány, například využitím statistických veličin a po optimalizaci dosahují lepších výsledků, podle předpokladů. V rámci práce byly vytvořeny dva skripty, kde jeden vytváří trénovací databázi a na těchto datech natrénuje klasifikátor a druhý vytváří testovací databázi a vybraný klasifikátor na takto získaných testovacích datech otestuje a vypíše výsledky. Výsledky jsou v práci zpracovány pomocí tabulky záměn a je pro ně vypočteno několik poměrových veličin, jako je přesnost, citlivost, specifičnost a další. Tyto výsledky jsou vždy uvedeny v příslušné kapitole v tabulkách i sloupcových grafech a řádně okomentovány.

Klíčová slova

Strojové učení, klasifikace, melovské kepstrální koeficienty, podpůrné vektory, spojení více slabých klasifikátorů v jeden větší celek.

Abstract

This work deals with methods for the detection of dangerous events, in this case gunshots, in a real environment. First of all, a testing and training database of sounds from the MIVIA database was created. In this database, the files were contained in six versions of signal-to-noise ratio, so the subsequent testing of the selected methods took place for the various shuffled files, and it was found that some methods are more accurate for cleaner recordings than others, but less accurate for more noisy ones. For the typical feature extraction from the input sound, the mel-frequency cepstral coefficients method was always used. In the thesis, the methods of support vector machines and ensemble of a number of weak classifiers are gradually tested on the created databases. These methods are then further optimized, for example by using statistical variables, and after optimization they achieve better results, as expected. In the work, two scripts were created, where one created a training database and on this data trained the classifier and the other created the test database, tested the selected classifier and obtained the results. The results are processed by confusion matrix and several proportional variables such as accuracy, sensitivity, specificity and others are calculated. These results are always listed in the relevant chapter of the thesis in the tables and column charts and are properly commented on.

Keywords

Machine learning, classification, mel-frequency cepstral coefficients, support vector machines, ensemble of a number of weak classifiers.

KOWOLOWSKI, A. *Detekce vybraných zvukových událostí v reálném prostředí.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2017. 49 s. Vedoucí bakalářské práce Ing. Jiří Přinosil, Ph.D..

Prohlášení o původnosti

Prohlašuji, že svou bakalářskou práci na téma Detekce vybraných zvukových událostí v reálném prostředí jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona č.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonu (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledku vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne 14.12.2016 ……

podpis autora

Poděkování

Tímto bych chtěl poděkovat vedoucímu své práce panu Ing. Jiřímu Přinosilovi, Ph.D. za cenné rady, pomoc s prostředím Matlab a trpělivost.

V Brně dne 14.12.2016 ……

podpis autora

Obsah

Seznam obrázků 9
Seznam tabulek 10
Úvod 11
1 Obecný návrh algoritmu 12
2 Segmentace 13
2.1 Hannovo okno 13
2.2 Hammingovo okno 13
3 Extrakce příznaků ze vstupního signálu 14
3.1 Melovské kepstrální koeficienty 15
3.1.1 Rychlá Fourierova transformace 15
3.1.2 Filtrování podle melovské škály 16
3.1.3 Logaritmizace 17
3.1.4 Výpočet kepstrálních koeficientů 17
3.1.5 Rozšíření o deriváty 19
4 Strojové učení 20
4.1 Klasifikátory 20
4.1.1 Podpůrné vektory 20
4.1.2 Perceptron 21
4.1.3 Neuronové sítě 21
4.1.4 Gaussovský směsný model 23
4.1.5 Bag of words 23
5 Vytvoření databáze 24
6 Prvotní realizace 25
6.1 Výsledky 26
7 Testování pro celou databázi 30
8 Optimalizace 32
8.1 Typy slabých klasifikátorů 32
8.2 Typy spojovacích metod 33
8.2.1 AdaBoostM1 33
8.2.2 LPBoost a Totalboost 33
8.2.3 Bagging 34
8.2.4 Subspace 34

8.2.5 RUSBoost 34
8.3 Výsledky 35
9 Využití statistických veličin 38
10 Závěr 42
Literatura 44
Seznam použitých zkratek a veličin 46
Seznam příloh 47

Seznam obrázků

Obr. 1.1: Obecné blokové schéma pro zpracování a klasifikaci zvuku. 12
Obr. 2.1: Srovnání Hannova a Hammingova okna. 13
Obr. 3.1: Standardní blokové schéma systému Melovských kepstrálních koeficientů. 15
Obr. 3.2: Typická banka filtrů obsahující 25 trojúhelníkových pásmových propustí [8]. 16
Obr. 3.3: Melovská škála [9]. 17
Obr. 3.4: Rozdělení signálu na nižší a vyšší kepstrální koeficienty. 18
Obr. 4.1: Rozdělení dat pomocí podpůrných vektorů. 21
Obr. 4.2: Srovnání neuronu v lidském mozku a umělého neuronu [14]. 22
Obr. 6.1: Znázornění melovských kepstrálních koeficientů pro jeden z výstřelů. 25
Obr. 6.2: Tabulka záměn [17]. 26
Obr. 6.3: Prvotní testovaní metody podpůrných vektorů pro lineární funkci. 28
Obr. 6.4: Prvotní testovaní metody podpůrných vektorů pro Gaussovu fukci. 29
Obr. 7.1: Klasifikace metodou podpůrných vektorů pro celou dostupnou databázi. 31
Obr. 8.1: Blokové schéma funkce fitensemble [18]. 32
Obr. 8.2: Porovnání různého počtu slabých klasifikátorů pro metodu AdaboostM1. 35
Obr. 8.3: Srovnání spojovacích metod pro 100 slabých klasifikátorů a SNR 5dB. 36
Obr. 8.4: Metoda TotalBoost pro 100 rozhodovacích stromů při různých honotách SNR. 37
Obr. 9.1: Metoda TotalBoost pro 100 trénovacích stromů při využití statistických veličin. 40
Obr. 9.2: Metoda Bag pro 100 trénovacích stromů při využití statistických veličin. 40
Obr. 9.3: Metoda Bag pro 55 trénovacích stromů při využití statistických veličin. 41

Seznam tabulek

Tab. 6.1: Klasifikace výstřelů pro hodnotu SNR 30 dB. 27
Tab. 6.2: Klasifikace výstřelů pro hodnotu SNR 15 dB. 28
Tab. 6.3: Klasifikace výstřelů pro hodnotu SNR 10 dB. 28
Tab. 6.4: Klasifikace výstřelů pro hodnotu SNR 5 dB. 28
Tab. 6.5: Klasifikace výstřelů pro hodnotu SNR 10 dB za použití Gaussovy funkce. 29
Tab. 6.6: Klasifikace výstřelů pro hodnotu SNR 15 dB za použití Gaussovy funkce. 29
Tab. 6.7: Klasifikace výstřelů pro hodnotu SNR 10 dB za použití Gaussovy funkce. 29
Tab. 6.8: Klasifikace výstřelů pro hodnotu SNR 5 dB za použití Gaussovy funkce. 29
Tab. 7.1: Klasifikace metodou podpůrných vektorů pro celou dostupnou databázi. 31
Tab. 8.1: Porovnání různého počtu slabých klasifikátorů pro metodu AdaboostM1. 35
Tab. 8.2: Srovnání spojovacích metod pro 100 slabých klasifikátorů a SNR 5dB. 36
Tab. 8.3: Metoda TotalBoost pro 100 rozhodovacích stromů při různých honotách SNR. 37
Tab. 9.1: Metoda TotalBoost pro 100 trénovacích stromů při využití statistických veličin. ... 39
Tab. 9.2: Metoda Bag pro 100 trénovacích stromů při využití statistických veličin. 40
Tab. 9.3: Metoda Bag pro 55 trénovacích stromů při využití statistických veličin. 41

Úvod

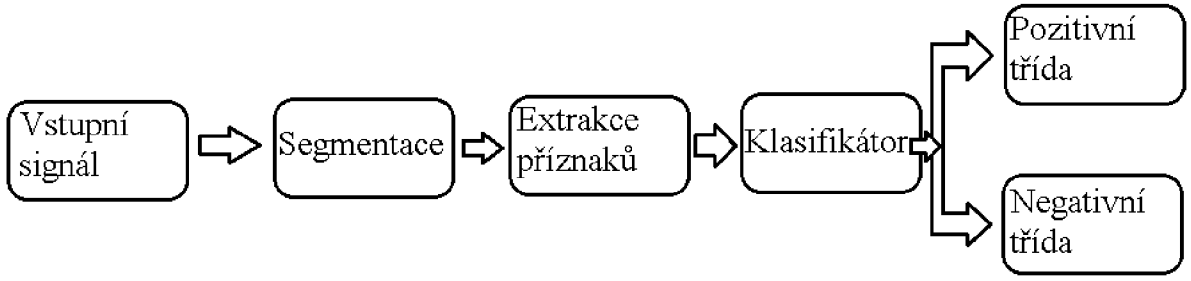
V minulosti byla detekce zvukových událostí používána především na rozpoznávání lidského hlasu a kategorizaci prostředí, ve kterém byla nahrávka pořízena. V dnešní době se analýza zvuku pomalu přesunula spíše do oblasti detekce nebezpečných událostí, jelikož spousta kamer má již v dnešní době zabudovaný mikrofon a existuje spousta nebezpečných zvuků, které se z kamery prakticky nedají poznat (například křik, výstřel mimo záběr atd.). V podstatě se v dnešní době používá jeden ze dvou přístupů. V prvním přístupu je použita jedna z metod extrakce příznaků (většinou MFCC - viz 3.1) a pak je vybrán a naučen klasifikátor. Do této skupiny patří i tato práce. Příklad použití některých metod v dnešní době: Gaussovský směsný model (GMM) byl použit Vacherem a kol. (v roce 2004) a Clavelem a kol. (2005) pro detekci křiku a výstřelů a v roce 2007 Valenzisem a kol. na řešení problému modelování zvuků v pozadí. Rabaoui a kol. (2008) použili metodu podpůrných vektorů (SVM) na zjištění podobnosti mezi zvuky. Foggia a kol. (2015) zase úspěšně aplikovali na detekci nebezpečných zvuků metodu „Pytel slov“ (angl. Bag-of-words), i když se původně jedná o metodu pro klasifikaci textu. Tento přístup může mít problémy s přesností, pokud chceme, aby systém rozpoznal více odlišných zvuků najednou a především, když chceme, aby zpracoval najednou dlouhé i impulzní zvuky. V takovém případě je lepší použít druhý, složitější přístup, kdy použijeme kombinaci více klasifikátorů. Takto to například udělali Rouas a kol. (v roce 2006), kdy zkombinovali GMM a SVM, aby dosáhli lepších výsledků na detekci křiku v hodně zašuměném prostředí, nebo Ntalampiras a kol. (2009), kteří využili dvoufázový GMM klasifikátor, kde se první klasifikátor stará o třídění na bezpečné a nebezpečné zvuky a druhý ty nebezpečné dále roztřídí do jedné ze skupin [1].

Práce obsahuje obecný teoretický rozbor dané problematiky a také její současný stav. Nejdříve nás obecně seznámí o celém řetězci detekce zvukových událostí. Poté popisuje segmentaci, především Hannovo a Hammingovo okno a teorii metody melovských kepstrálních koeficientů. Práce taktéž obsahuje teoretický rozbor všech použitých klasifikačních metod i dalších, dnes běžně pro tuto problematiku používaných a teorii všech v práci použitých statistických veličin. Poslední čtyři kapitoly pojednávají o praktické realizaci práce, všech testovaných metodách a následné optimalizaci. Dosažené výsledky jsou uvedeny v tabulkách a sloupcových grafech vždy u příslušné kapitoly. K práci je rovněž přiloženo CD s vytvořeným programem a dalšími soubory. Seznam souborů obsažených na disku a návod k použití programu jsou uvedeny v příloze.

1 Obecný návrh algoritmu

Obecně je využíváno pro zpracování jakéhokoliv zvukového signálu dvou základních kroků. Nejdůležitější je samozřejmě samotné strojové učení, kde je vybrána jedna z metod a ta následně aplikována na systém, aby byly přiřazeny („naučeny“) výstupy k daným vstupům. Nejdříve je tedy použit trénovací algoritmus, kde například v případě této práce, kdy existují dvě skupiny, do kterých budou data dělena, bude systém naučen, že právě tyto příznaky patří právě do této skupiny. Poté je použit testovací algoritmus, kdy jsou puštěna podobná (ale ne stejná) data do systému a je ověřeno, jak přesně se systém rozhoduje. Obecně by toto samo o sobě stačilo, jenže každý vstupní signál má určitou vzorkovací frekvenci a tento počet vzorků je příliš velký a obsahuje množství zbytečných informací, které systém nepotřebuje znát. Proto je ještě před tímto krokem použita jedna z metod extrakce příznaků, aby byly odstraněny přebytečné informace a systém se stal efektivnějším a rychlejším [1].

Zde bude použita metoda melovských kepstrálních koeficientů, jelikož se jedná o celkem jednoduchou metodu (z hlediska výpočtu systémem) pro zpracování charakteristických příznaků vstupního signálu. Pro její efektivní použití je třeba, aby byl zpracovávaný signál nejdříve segmentován na krátká okna (v našem případě 25 ms).



Obr. 1.1: Obecné blokové schéma pro zpracování a klasifikaci zvuku.

Typu algoritmu strojového učení, který se rozhoduje, do které skupiny roztřídit vstupní data, říkáme klasifikátor. V této práci bude testována především metoda podpůrných vektorů. Výhodou metody je opět jednoduchost. Největší nevýhodou této metody je, že bez vetší dodatečné úpravy se nedá použít jinak, než jako binární klasifikátor, což zde ale nevadí, jelikož se nám jedná jen o to, zda jde o skupinu výstřelů (pozitivní třída) nebo pozadí (negativní třída).

2 Segmentace

Předpokladem je, že každý zvuk má své typické příznaky, které zůstavají po krátkou dobu (v rozmezí 5 - 100 ms) neměnné. Může být odhadováno, že se signál v těchto krátkých časových úsecích chová stacionárně. Pokud mají být tyto příznaky získány, musí proběhnout segmentace. K tomu se používá váhovací funkce. Váhovací funkce je ve zpracování signálů typ matematické funkce, která je vždy mimo vybraný interval nulová. Pokud je jiná funkce nebo signál vynásoben váhovací funkcí, výsledek je také nulový mimo vybraný interval, tím pádem zůstane jen část, kde se obě funkce překrývají. Dá se říci, že je daný úsek signálu sledován přes „okno“ - odtud název pro různé váhovací funce - váhovací okna. Nejčastěji užívaná okna mají obdélníkový, troúhelníkový tvar, nebo tvar jiné pásmové propusti. Nejčastěji používaná okna jsou okna Hannovo a Hamingovo. Různá okna mají různé charakteristiky, váhují rozdílně a tím pádem i produkují jiné segmenty. Okna se mohou překrývat. Obecně platí, že větší překrývání sousedních oken znamená menší změnu parametrů sousedních vzorků, ale potřebuje více výpočetní síly [2][3].

2.1 Hannovo okno

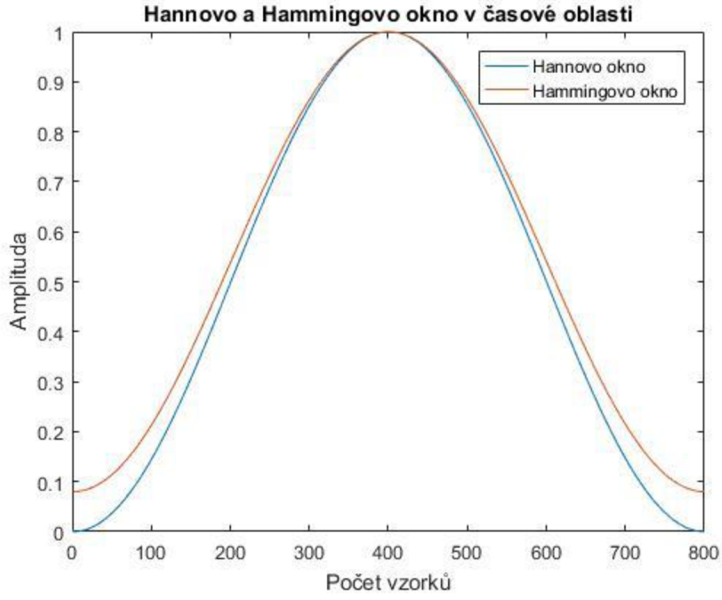
Všechna Hannova i Hammingova okna jsou popsána obecnou rovnicí

$$w(n) = \alpha - \beta\cos\left(\frac{2\pi n}{N-1}\right). \tag{2.1}$$

Jedná se o základní okno této skupiny, kde α a β jsou koeficienty popisující okno - u Hannova okna jsou oba nastavené na 0,5. N je šířka okna a n počet vzorků.

2.2 Hammingovo okno

Jedná se o optimalizovanou verzi Hannova okna, kdy koeficienty α a β jsou nastaveny na $\alpha = 0{,}54$ a $\beta = 1 - \alpha = 0{,}46$, což snižuje jeho výšku oproti oknu Hannovu [4].



Obr. 2.1: Srovnání Hannova a Hammingova okna.

3 Extrakce příznaků ze vstupního signálu

Hlavním účelem extrakce příznaků u automatické detekce zvukových signálů je umožnit samotnou klasifikaci zpracováním typických příznaků z malých oken vstupního signálu. Jak již bylo popsáno výše, je to hlavně proto, že nezpracovaný vstupní signál obsahuje informace, které jsou klasifikátoru k ničemu a mají vysokou dimenzionalitu. Obě tyto vlastnosti vstupního signálu je třeba odstranit, jinak by se klasifikátor vyznačoval vysokou chybovostí. Výsledkem extrakce příznaků je tedy vektor charakteristických příznaků s nižší dimenzionalitou. Nižší dimenzionalita dat totiž snižuje čas potřebný k výpočtům a počet trénovacích vzorků k naučení systému. Takovýto vektor by měl klást důraz na všechny pro klasifikátor důležité informace a snažit se potlačit všechny ostatní. Nežádoucí charakteristiky, jako jsou typické charakteristiky mluvčího, charakteristiky hluku pozadí a charakteristiky nahrávacích zařízení, by měly být potlačeny, jelikož akorát přidávají systému na chybovosti.

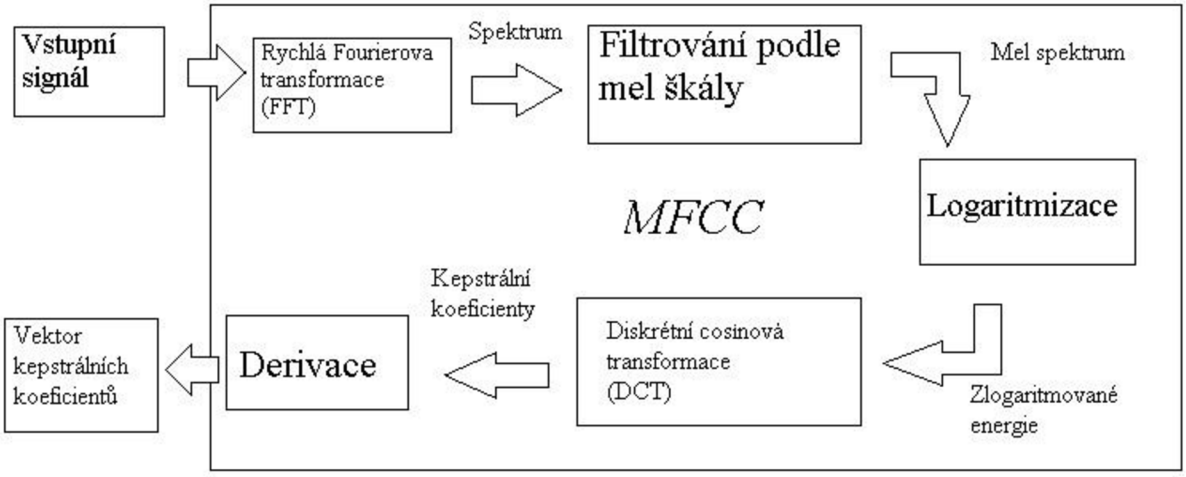
Nejčastěji používané metody na extrakci příznaků jsou v dnešní době melovské kepstrální koeficienty, příznaky založené na vlnkách, nebo například nezáporná maticová faktorizace. Všechny tyto metody jsou navrženy pro zpracování pouze krátkých oken v délce 10 - 30 ms.

Každá z výše zmíněných metod má alespoň nějaký nedostatek. K odstranění těchto nedostatků se někdy používá kombinace více metod. Takovéto systémy dokáží úspěšně zpracovávat i delší časová okna v délce třeba 200 ms. Abychom získali vektory příznaků pokrývající delší časový rámec, n-sekvenční nízkoúrovňové příznakové vektory uvnitř posuvného okna jsou spojeny v jeden a zpracovávány souběžně [5].

Nyní bude podrobně popsána metoda melovských kepstrálních koeficientů.

3.1 Melovské kepstrální koeficienty

Melovské kepstrální koeficienty, neboli anglicky mel-frequency cepstral coefficients (MFCC), je nejčastěji používaná metoda na extrakci dat pro automatické rozpoznávání hlasu. Tato metoda byla poprvé zmíněna Bridlem a Brownem v roce 1974 a později rozvedena Paulem Mermelsteinem v roce 1976. Snaží se napodobit logaritmické vnímání hlasitosti a výšky lidského sluchového aparátu a zároveň potlačit typické charakteristiky, vynecháním základní frekvence a vyšších harmonických složek. K získání kepstrálních koeficientů je zapotřebí několik kroků, jak můžeme vidět na blokovém schématu.



Obr. 3.1: Standardní blokové schéma systému Melovských kepstrálních koeficientů.

Vstupem do systému je jakýkoliv zvukový signál, který byl předtím rozdělen na menší vzorky v délce nejlépe 20 až 40 milisekund [5][6].

3.1.1 Rychlá Fourierova transformace

Prvním krokem ve zpracování signálu je pak výpočet diskrétní Fourierovy transformace. Diskrétní Fourierova transformace (neboli DFT) převádí signál o konečné délce ze své původní oblasti (většinou časové), do znázornění frekvenční oblasti vstupního signálu. Rychlá Fourierova transformace (FFT) pak rychle počítá takové transformace rozkladem DFT matice na určité množství takzvaných řídkých matic, což jsou matice, ve kterých je převážná většina prvků nulových, což snižuje složitost výpočtu DFT.

DFT je definována vzorcem

$$c_1 = \left| \frac{1}{N} \sum_{j=0}^{N-1} f_j \, exp\left[-i2\pi \frac{jk}{N} \right] \right| \qquad k = 0,1, \ldots, \left(\frac{N}{2}\right) - 1, \tag{3.1}$$

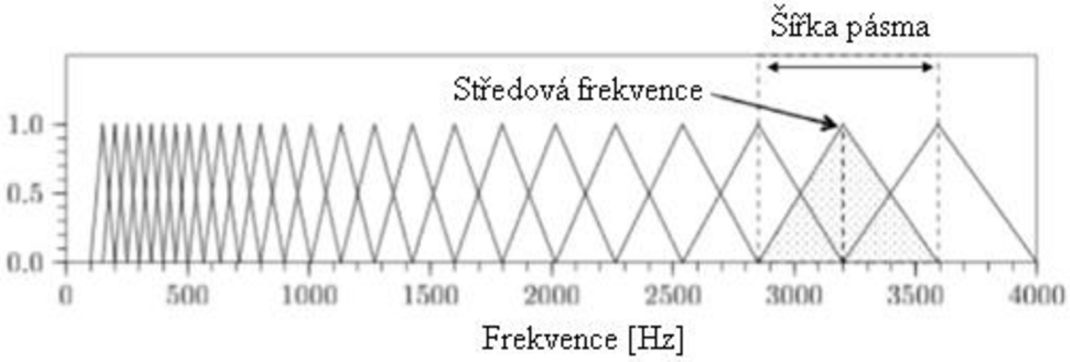
kde f je frekvence vstupního signálu, k je řada vypočtená pomocí N, což je počet samplů uvnitř časového okna. Přímé vyhodnocení těchto sum by zabralo $O(N^2)$ aritmetických operací. FFT naproti tomu poskytuje složitost pouze $O(N*\log N)$ operací a tím zrychluje výpočet. Nyní bude vytvořeno frekvenční spektrum signálu, což není nic jiného, než řada komplexních čísel popsaných výše [7].

3.1.2 Filtrování podle melovské škály

Druhým krokem je výpočet spektra mel-frekvence. Nejdříve bude spektrum filtrováno pomocí N_d rozdílných filtrů typu pásmová propust a vypočteny energie pro každé frekvenční pásmo. Tento proces může být vyjádřen vzorcem

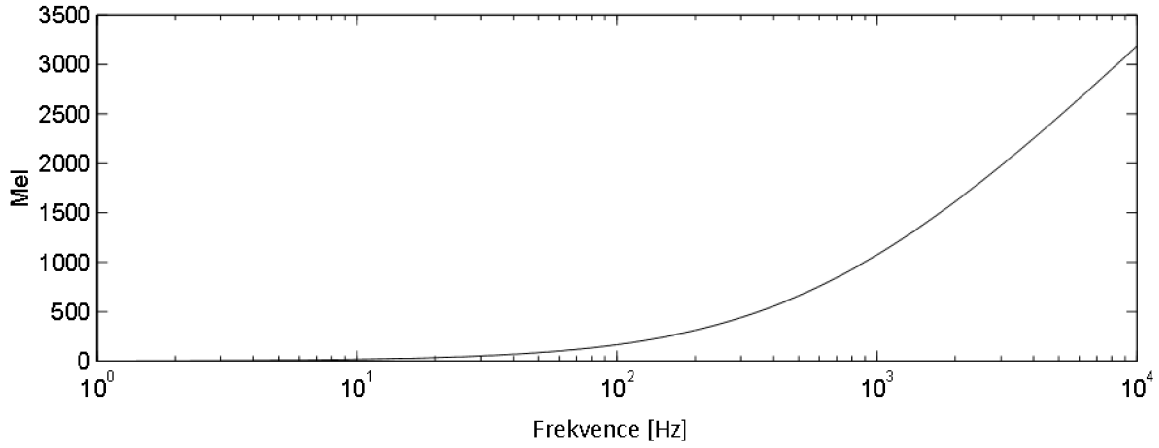
$$c_2 = \sum_{k=0}^{\frac{N}{2}-1} d_{j,k}\, c_1 \qquad j = 0,1,\dots,N_d, \tag{3.2}$$

kde d je amplituda pásmové propusti s indexem j na kmitočtu k. Banka filtrů s pásmovými propustmi ale nemůže napodobit charakteristiku lidského ucha, jelikož lidské ucho může použít jakoukoliv frekvenci jako středovou frekvenci. Pro rozpoznávání signálů je tedy použito N_d ekvidistantních pásmových filtrů na melovské stupnici. Melovská stupnice je nelineární škála, která je uzpůsobena nelineárnímu vnímání výšky lidského sluchu.



Obr. 3.2: Typická banka filtrů obsahující 25 trojúhelníkových pásmových propustí [8].

Bylo prokázáno, že použití méně, nebo naopak mnohem více filtrů negativně ovlivňuje třídící schopnosti systému a také, že překrývající se filtry dosahují lepších výsledků než nepřekrývající se. Jak může být vypozorováno z obrázku, první filtr je velmi úzký a dává údaj o tom, kolik energie se vyskytuje kolem 0 Hz. S rostoucím kmitočtem jsou trojúhelníková okna rozšiřována, jelikož méně záleží na variacích. Podává hrubou informaci o tom, kolik energie se vyskytuje na určitém místě.



Obr. 3.3: Melovská škála [9].

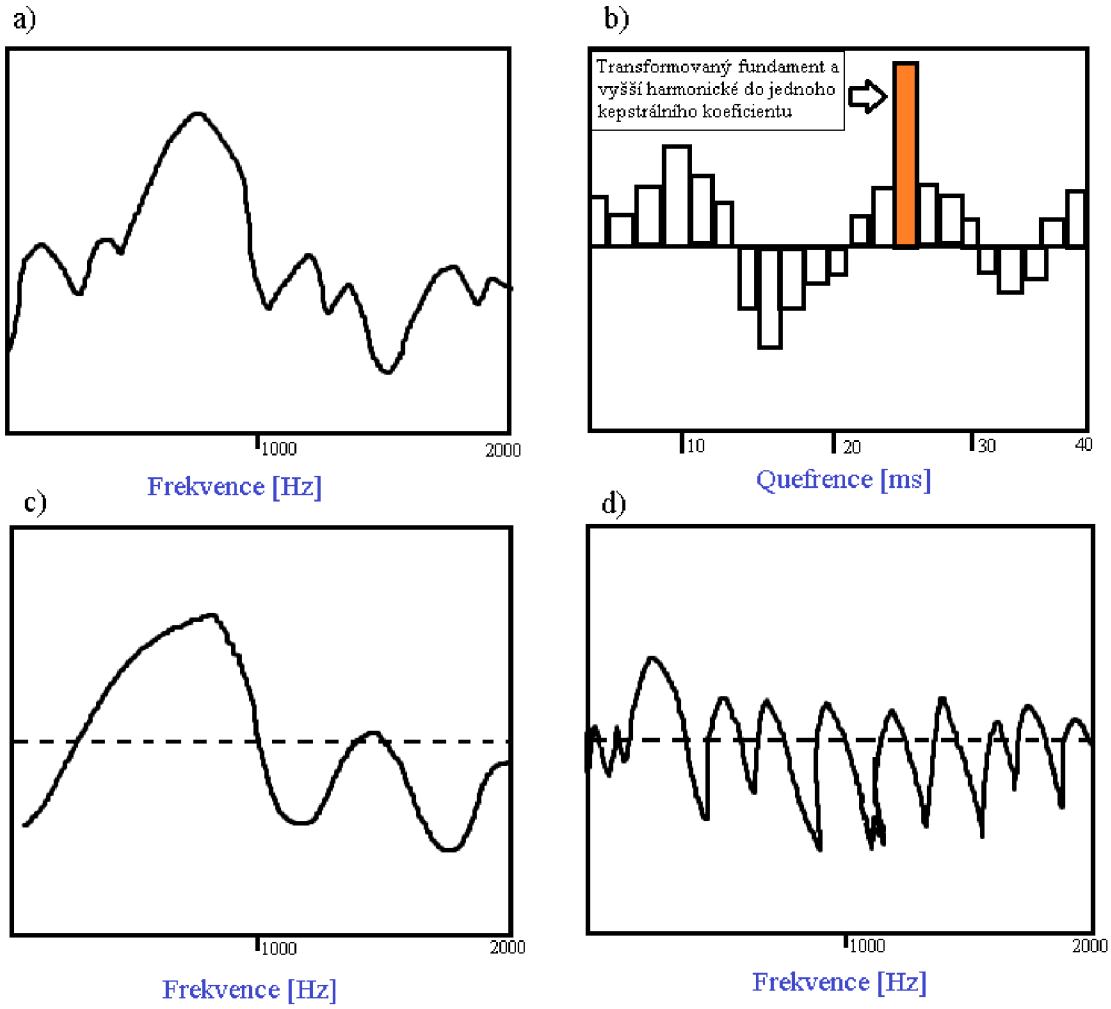
3.1.3 Logaritmizace

Jakmile budou tyto energie získány, tak proběhne jejich logaritmizace. To je opět prováděno kvůli lidskému sluchu, jelikož člověk neslyší hlasitost lineárně. Například aby došlo ke zdvojnásobení vnímané hlasitosti zvuku, je zapotřebí zvýšit vstupní energii alespoň osmkrát [8].

$$c_3 = log(c_2). \tag{3.3}$$

3.1.4 Výpočet kepstrálních koeficientů

Čtvrtým krokem jsou eliminovány nechtěné typické charakteristiky výpočtem kepstrálních koeficientů. Kepstrum tedy může být chápáno jako spektrum spektra. Závislé frekvence první harmonické jsou, za reálných podmínek, transformovány do jednoho kepstrálního koeficientu vyššího řádu. Zpětná transformace nižších kepstrálních koeficientů interpretuje frekvenční odezvu hlasového ústrojí a inverzní transformace kepstrálních koeficientů vyššího řádu ukazuje frekvenční charakteristiku vstupního signálu. Sebráním těchto nižších kepstrálních koeficientů pro další zpracovaní jsou vyšší harmonické mluvčího potlačeny.



Obr. 3.4: Rozdělení signálu na nižší a vyšší kepstrální koeficienty.

a) Vstupní spektrum signálu po zlogaritmování. *b) Kepstrální koeficienty vstupního signálu.*
c) Spektrum nižších kepstrálních koeficientů. *d) Spektrum vyšších kepstrálních koeficientů.*

K výpočtu kepstrálních koeficientů je používána diskrétní cosinová transformace (DCT). Diskrétní kosinová transformace je jedna z integrálních transformací podobných Fourierově transformaci, jejíž odlišností je hlavně to, že produkuje pouze reálné koeficienty. To zde ale může být použito, jelikož absolutní hodnota spektra je symetrická a je to vždy reálné číslo.

Pro výpočet kepstrálních koeficientů se používá vzorec

$$c_4 = \sum_{j=0}^{N_d} c_3 \cos\left[\frac{k(2j-1)\pi}{2N_d}\right] \qquad k = 0,1,\ldots,N < N_d, \tag{3.4}$$

jedná se o cosinovou transformaci zlogaritmovaných energií z třetího kroku.

3.1.5 Rozšíření o deriváty

Posledním krokem je rozšíření kepstrálních koeficientů na vektor, přidáním první a druhé derivace oněch koeficientů, aby byla zastoupena dynamická povaha střelby.

$$c = [c_4, \Delta c_4, \Delta\Delta c_4]. \tag{3.5}$$

Typický MFCC vektor vypočtený z okna řekněme pěti sty dvanácti vzorků, by se pak skládal z třinácti kepstrálních koeficientů, třinácti prvních derivací a třinácti derivací druhého řádu. V tomto příkladu by tedy došlo ke snížení vzorků z 512 na 39 a to je přesně to, co je zde zapotřebí [8][9].

4 Strojové učení

Strojové učení je podoblastí informatiky, která zkoumá možnosti daného systému rozhodovat se a „učit se“, aniž by k tomu byl speciálně naprogramován. Takovéto systémy pak předvídají a samy se rozhodují, než aby se držely striktně daného kódu.

Strojové učení je možno, podle toho, jaká data jsou systému dostupná, typicky roztřídit do čtyř základních skupin, a to:

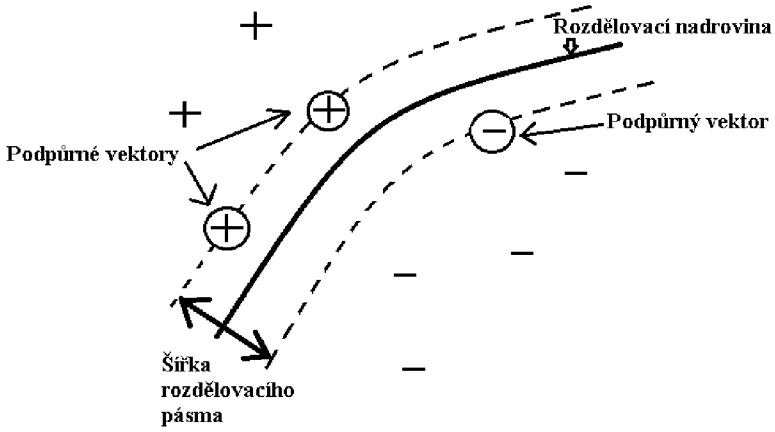
- **Kontrolované učení**: Programu jsou prezentovány vzorové vstupy a jejich požadované výstupy a jeho úkolem je zjistit vzorec, který tyto vstupy a výstupy propojí.
- **Nekontrolované učení**: Algoritmus strojového učení je nucen sám si zjistit strukturu na vstupu a podle toho najít výstup.
- **Částečně kontrolované učení:** Kde trénovací signál není kompletní, určitá část cílových výstupů chybí.
- **Zpětnovazební učení:** Kde je program nucen provádět nějakou činnost a přitom neví, zda ji dělá dobře. Svého cíle se snaží dosáhnout pomocí zpětné vazby dynamického prostředí, ve které danou činnost provádí. Jedná se tedy o takovou metodu typu „pokus - omyl“ [10].

4.1 Klasifikátory

Klasifikátory jsou algoritmy k roztřídění vstupních dat do určitého počtu skupin. Většinou se jedná o poddruh kontrolovaného učení. Používají se při filtrování spamů, získávání dat z databází, rozpoznávání jazyka, obrazů, videa, ve zpracování signálů i na korekci chyb. Tyto algoritmy mohou třídit na základě jednoduchých třídících pravidel, až po složité funkce, které využívají mechanismy strojového učení. Nyní si ukážeme některé z nejběžněji používaných klasifikátorů.

4.1.1 Podpůrné vektory

Podpůrné vektory, neboli anglicky support vector machines (SVM), je poměrně nová metoda (přelom 20. a 21. století) strojového učení, která se používá především pro binární klasifikaci a regresi. Binární klasifikace se používá na rozdělení výsledků přesně do dvou skupin. Tato metoda vychází ze statistické učící teorie vyvinuté Vapnikem and Chervonenkisem. Vykazuje dobrý výkon při předpovědi nových dat (dobrá generalizace), což je důležitá věc pro odvětví umělé inteligence. Podpůrné vektory třídí data nalezením nejlepší nadroviny, která roztřídí body jedné skupiny od bodů druhé skupiny, což je ta nadrovina, která je nejvzdálenější bodům obou skupin. Pro popsání nadroviny používáme její nejbližší body, tzv. podpůrné vektory a odtud název metody. Definujeme také šířku rozdělovacího pásma, anglicky margin.



Obr. 4.1: Rozdělení dat pomocí podpůrných vektorů.

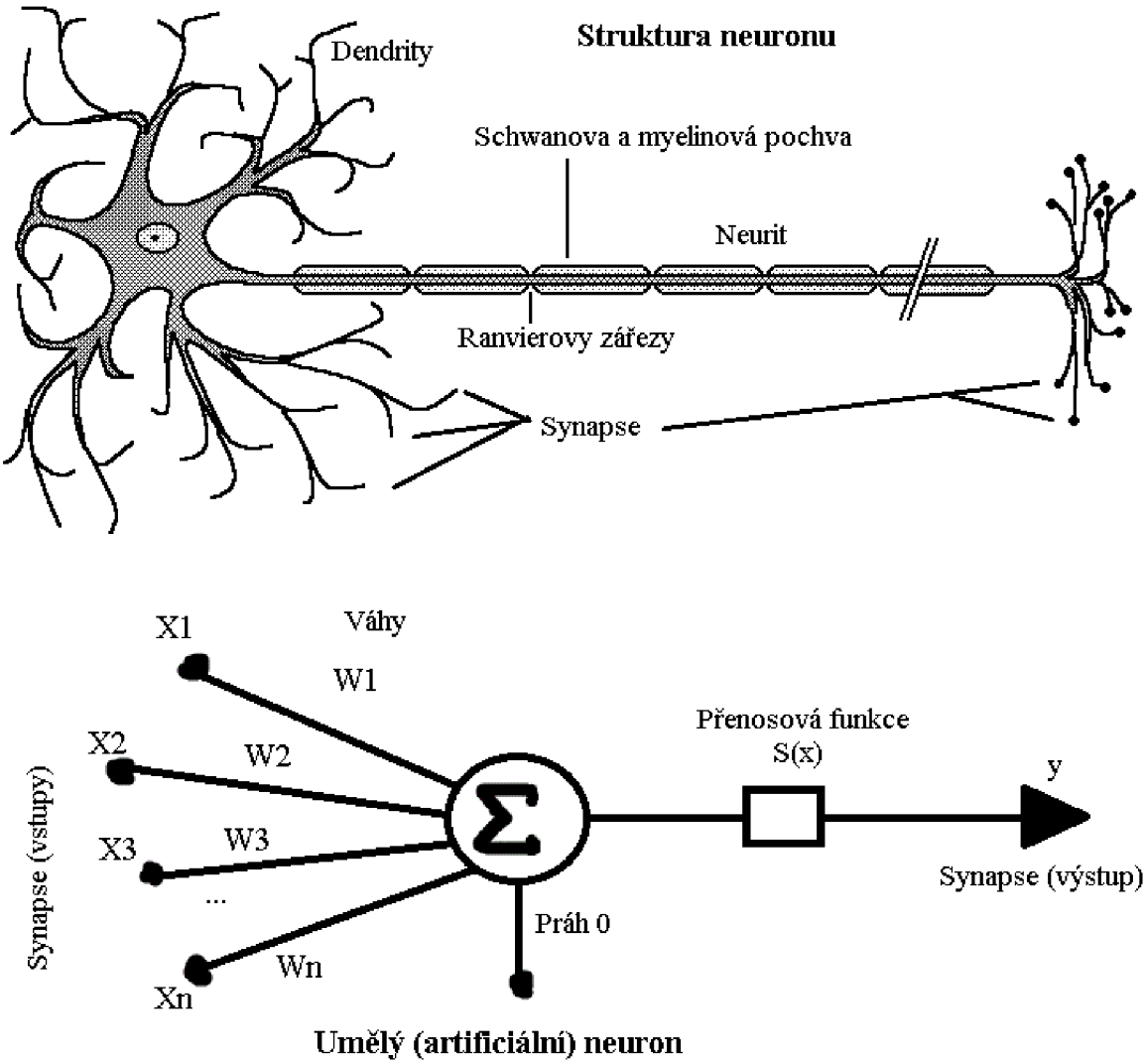
Pro jednodušší klasifikační metody se používá lineární rozdělovací rovina. Pokud nemohou být třídy lineárně rozděleny v originální rovině, metoda SVM tuto rovinu nejdříve nelineárně transformuje za použití různých nelineárnmích mapovacích funkcí na rovinu o vyšší dimenzi [11][12].

4.1.2 Perceptron

Petceptron je nejjednodušším modelem neuronové sítě a jeho základním kamenem. Sestává se pouze z jednoho neuronu. Jedná se o binární klasifikátor pro kontrolované učení. Lze jej použít pouze pro lineární klasifikaci. Tento model se sice již v dnešní době tak nepoužívá, ale přišlo mi zajímavé jej zahrnout, jelikož byl vynalezen již v roce 1957 Frankem Rosenblattem, což z něj dělá jednu z nejstarších metod strojového učení vůbec. Struktura (jednoho neuronu) viz Obr. 4.2 [13].

4.1.3 Neuronové sítě

Existuje spousta úloh umělé inteligence, které jsou pomocí algoritmů velmi složitě řešitelné, zatímco člověk je v řadě případů úspěšně řeší. Metoda neuronových sítí se tedy snaží o takovou „hardwarovou realizaci lidských schopností“. Neuronové sítě jsou postaveny na fungování lidského nervového systému. Nervová soustava se skládá z bohatě rozvětvených nervových buněk - neuronů, které vedou vzruchy. Neuron je vysoce specializovaná buňka, z jejíhož těla vybíhá mnoho větvících se výběžků - dendritů a jeden nápadně dlouhý výběžek - neurit, který je zakončen synapsí (spojem), pomocí kterého se neurit připojuje k dalšímu neuronu. Systém neuronových sítí se toto snaží napodobit vytvořením řady umělých (virtuálních) neuronů.



Obr. 4.2: Srovnání neuronu v lidském mozku a umělého neuronu [14].

Typy neuronových sítí

Základním kritériem pro členění sítí je, zda byly vytvořeny pro praktické aplikace, nebo pro neurologické modelování.

- **Aplikační modely:** Jedná se o modely pro klasifikaci. Tyto neuronové sítě mohou být použity pro kontrolované, částečně kontrolované i nekontrolované učení. U nekontrolovaného učení je používáme hlavně pro tzv. "clustering" - redukce vstupních dat bez ztráty relevantní informace (např. různým výslovnostem slova přiřadit jeho význam).
- **Neurologické modely:** Jsou modely, které se snaží objasnit činnost nervové soustavy. Ale struktura nervové soustavy je příliš složitá, než aby bylo možné vytvořit relativně jednoduchý model, odrážející její chování. Alespoň prozatím [14].

4.1.4 Gaussovský směsný model

Z anglického Gaussian mixture model (GMM), jedná se o sumu Gaussovských funkcí. Tento model je často používán na rozpoznávání hlasu, jelikož dokáže zobrazit široké rozložení vzorků a dobře pracuje s charakteristikami mluvčího. Klasifikátory založené na tomto modelu bývají také často používány na detekci nebezpečných událostí. Často bývá použit právě v kombinaci s jiným modelem, jako například právě s podpůrnými vektory, nebo sám jako dvoufázový Gaussovský klasifikátor. Obecně platí, že složitější klasifikátory, jako tyto kombinace si lépe poradí s šumem pozadí, ale jsou mnohem náročnější na výpočet a navíc člověku zabere více času zpracovat dataset pro takovýto systém. Je proto většinou výhodnější použít jednoduchý systém a dostatečně jej optimalizovat [15][1].

4.1.5 Bag of words

Nejedná se přímo o klasifikátor, ale spíše o kompletní systém, používaný většinou na klasifikaci textu a obrázků. Anglicky „pytel slov“, v této metodě je text (například věta) zastoupen multimnožinou svých slov, bez ohledu na gramatiku a slovosled. Pro uvedení příkladu, pokud je zvolena věta:

Lyžování mám rád. Také mám rád práci v prostředí MATLAB.

Systém vytvoří následující multimnožinu.

```
[ "Lyžování", "mám", "rád", "Také", "práci", "v", "prostředí", "MATLAB" ]
```

Po transformaci textu na bag of words mohou být vypočteny různé hodnoty, aby byl text charakterizován. Nejčastěji je počítána frekvence, tím je myšleno, kolikrát se dané slovo v textu vyskytuje. V tomto případě by byl výsledkem vektor

```
[1, 2, 2, 1, 1, 1, 1, 1, 1, 1],
```

jelikož se všechna slova vyskytují v textu jednou, kromě slov "mám" a "rád", která se v textu vyskytují dvakrát. Tento systém se dá optimalizovat vynecháním hodně častých slov, jako jsou například spojky, které klasifikaci zhoršují. Také se dá použít n-gramový posloupnostní model, který pak počítá se všemi dvojicemi slov, místo s jednotlivými slovy, což dále zvyšuje přesnost [16].

```
[ "Lyžování mám", "mám rád", "Také mám", "rád práci" atd.]
```

Dobrým příkladem použití metody bag of words je například filtrovaní spamů v e-mailu. Nebyla sem však zahrnuta proto, že se jedná o důležitý klasifikátor v jiných oblastech, než je klasifikace zvukových událostí, ale proto, že Ph.D. Pasquale Foggia a kol. v roce 2014 použili tuto metodu na klasifikaci nebezpečných zvuků a dosáhli dobrých výsledků. Hlavní myšlenkou je, že každý zvukový datový tok, může být složen z malých jednotek lidského slyšení, kterým se říká sluchová slova (z anglického aural words), jejichž rozložení v konečném časovém intervalu nám napoví, o jaký typ zvuku se jedná [1].

5 Vytvoření databáze

K realizaci celé praktické části bylo použito prostředí Matlab, přesněji verze R2016a. Pro praktickou část této práce bylo nejdříve nutné vytvořit databázi (respektive dvě databáze), pro natrénování a následné otestování systému. K tomu byl použit veřejný dataset MIVIA Database. Ten se skládá z trénovacích a testovacích nahrávek. Všechny nahrávky jsou ve formátu wave s délkou lehce přes tři minuty, se vzorkovacím kmitočtem 32000 Hz a skládají se z pozadí, které bylo pro každou nahrávku jiné (projíždějící auta, kancelářské prostředí, potlesk) a náhodně se vyskytujících nebezpečných zvuků, kdy se jednalo vždy o výstřel, výkřik, nebo rozbití skla. Všechny nahrávky jsou v databázi navíc v šesti různých verzích odstupu užitečného signálu od šumu (anglicky SNR = signal-to-noise ratio), jmenovitě odstup 30, 25, 20, 15, 10 a 5 dB. Jelikož by bylo časově nesplnitelné najít a vystřihnout z nahrávek všechny výstřely, byly k nahrávkám přiloženy i XML soubory, kde byla v každém XML souboru uveden typ (označen jako *class*) nebezpečného souboru s počáteční a koncovou sekundou.

K vystřižení těchto událostí bylo nejdříve potřeba nahrát XML soubory do prostředí Matlab a pak najít všechny typy událostí "*gunshot*" (výstřel) a použít informaci "*startsecond*" (počáteční sekunda) a "*endsecond*" (koncová sekunda) o začátku a konci události k jejich vystřižení. Prostředí Matlab nepodporuje nahrávání XML souborů, proto byl použit XML toolbox pro Matlab, přesněji funkce `xml_load`, která rozpozná XML soubor a pomocí `xml_parse` jej převede na datovou strukturu nebo proměnnou prostředí Matlab. Protože je každý zvukový signál po nahrání do prostředí Matlab automaticky převeden do číslicové podoby jako vektor vzorků, bylo potřeba čas začátku události převést na první vzorek oné události a zvolit nějakou univerzální délku souboru. Je totiž nutné, aby byly všechny trénovací nahrávky stejně dlouhé. Byla použita délka 1 sekunda. Sice spousta výstřelů měla třeba jen 300 ms, byly ale i takové, které měly 800 - 900 ms. Jelikož byla známa vzorkovací frekvencr nahrávek, stačilo provést následující

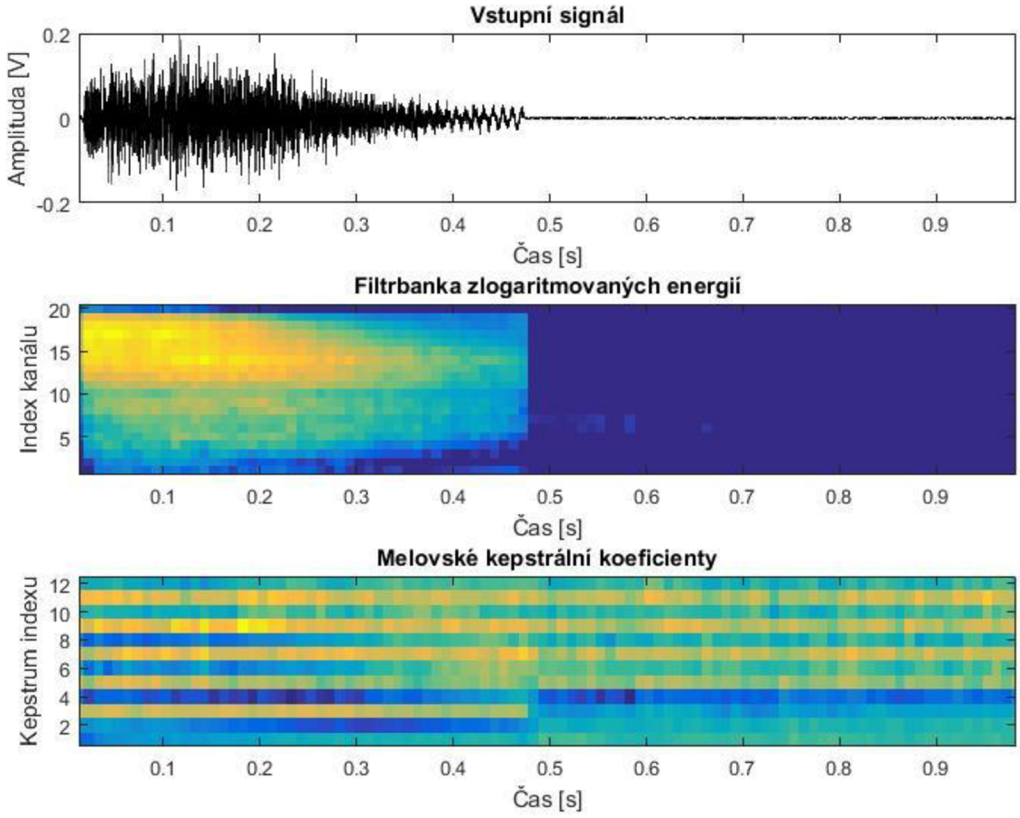
$$x = y(f_s * t_s : f_s * t_e + f_s), \tag{5.1}$$

kde x je onen nebezpečný zvuk, y je celá tříminutová nahrávka, t_s je počáteční sekunda, t_e koncová sekunda a f_s je vzorkovací frekvence, z anglického "sampling frequency" (dvojtečka v tomto kontextu znamená „až“).

6 Prvotní realizace

Nejdříve byla takto vzniklá šablona počátečních a koncových vzorků výstřelů aplikována na všechny nahrávky s různými hodnotami SNR. Výsledkem pak bylo 166 trénovacích zvuků různých výstřelů, dvojnásobek zvuků pozadí a jiných událostí, 300 testovacích zvuků, opět v poměru výstřelů ku pozadí 1:2, a to vše v šesti verzích SNR, takže 996 trénovacích výstřelů, 1992 trénovacích pozadí a 1800 testovacích vzorků.

Dále byly vypočteny melovské kepstrální koeficienty. K tomu, aby je systém mohl ze vstupního signálu (matice vstupních hodnot) spočítat, potřebuje znát několik dalších hodnot. Jmenovitě jsou to vzorkovací frekvence (32 kHz), u FFT je nastavována šířka a překrývání oken (25 ms a 10 ms) a typ použitého okna (Hammingovo). Dále počet trojúhelníkových filtrů (25), frekvenční rozsah (20 Hz - 4 kHz) a nakonec počet kepstrálních koeficientů, který je požadován (bylo ponecháno přednastavených 12).



Obr. 6.1: Znázornění melovských kepstrálních koeficientů pro jeden z výstřelů.

Výsledkem je matice kepstrálních koeficientů. Aby byla získána matice příznaků pro naučení systému, bylo třeba udělat z každé matice kepstrálních koeficientů vektor a následně vytvořit jednu velkou matici, kde každý řádek znázorňoval jeden zpracovaný sekundový signál a každý sloupec pak jeden jeho příznak. Celkový počet příznaků (a tím i sloupců) byl 1274.

Nakonec bylo nutné získat naučený model klasifikátoru. Do funkce podpůrných vektorů v prostředí Matlab (*fitcsvm)* je nutno zadat matici příznaků z předchozího kroku a vektor, ve kterém je zaznačeno, které řádky matice jsou výstřely a které pozadí. Byl tedy vytvořen vektor délky počtu trénovacích signálů, ve kterém byla vždy na místech s pořadovým číslem odpovídajícím místu výskytu výstřelu zaznačena logická 1 a na místech odpovídajících pozadí zaznačena 0, z čehož se systém naučil příznaky typické pro pozitivní a negativní třídu a výstupem z *fitcsvm* byl natrénovaný model.

```
klasifikator = fitcsvm(training,a,'KernelFunction','linear','KernelScale','auto')
```

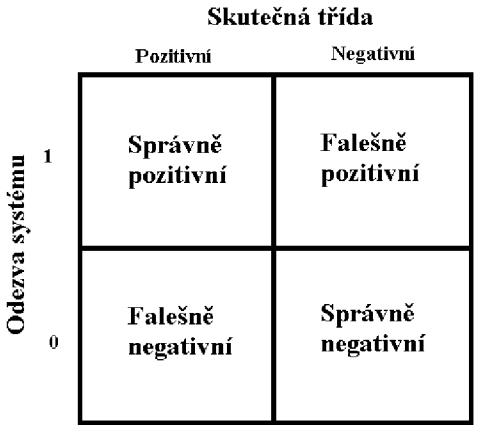
Tento model byl testován pomocí funkce *predict*, kde je vstupem natrénovaný model a příznaková matice, vytvořená stejným způsobem jako ta pro natrénování systému, ale z nahrávek jiných výstřelů a pozadí. Zde je výsledkem vektor, v němž je zapsáno, které vzorky rozpoznal systém jako výstřel a které jako pozadí a ten byl poté porovnán s reálnými údaji. Toto bylo několikrát zopakováno, pro různé hodnoty SNR i pro různé úpravy funkce *fitcsvm*.

```
vysledek = predict(klasifikator,testing).'
```

Výsledky po porovnání s reálnými údaji jsou uvedeny níže.

6.1 Výsledky

Pro zpracování výsledků binárních klasifikátorů je většinou využívána tzv. tabulka záměn (někdy také nazývána tabulka možností). Ta je složena ze čtyř základních hodnot, a to pozitivních, negativních, falešně pozitivních a falešně negativních klasifikací. Pozitivní (dále jen SP) a negativní klasifikace (SN) znamenají správné rozpoznání pozitivní (výstřel) nebo negativní (pozadí) třídy. Falešně pozitivní (FP) klasifikace nastává, pokud systém rozpoznal jeden ze vzorků pozadí jako výstřel a naopak je tomu u falešně negativní (FN) klasifikace, kdy systém výstřel vůbec nepozná.



Obr. 6.2: Tabulka záměn [17].

Dále mohou být využity hodnoty z tabulky záměn pro výpočet mnoha poměrů mezi nimi. Zkratka CP vyjadřuje celkový počet pozitivních vzorků a zkratka CN celkový počet vzorků pozadí. Ve výsledcích budou uvedeny tyto hodnoty:

- **Přesnost** - udává poměr všech správných observací k celkovému počtu. Je jedním z nejdůležitějších faktorů klasifikátorů.

$$Přesnost = \frac{SP + SN}{CP + CN} * 100\ [\%]. \tag{6.1}$$

- **Citlivost** (také nazývána poměr správně pozitivních) - udává míru správně rozpoznaných pozitivních observací

$$Citlivost = \frac{SP}{CP} * 100\ [\%]. \tag{6.2}$$

- **Specifičnost** (také nazývána poměr správně negativních) - udává míru správně rozpoznaných negativních observací

$$Specifičnost = \frac{SN}{CN} * 100\ [\%]. \tag{6.3}$$

Přesnost, citlivost i specifičnost zde bude uváděna v procentech.

- **Falešně pozitivní míra** - ukazuje poměr falešně pozitivních výsledků a všech negativních.

$$FPM = \frac{FP}{CN} = 1 - Specifičnost. \tag{6.4}$$

- **Falešně negativní míra -** ukazuje poměr falešně negativních výsledků a všech pozitivních [17].

$$FNM = \frac{FN}{CP} = 1 - Citlivost. \tag{6.5}$$

Nyní bude předvedena přesnost klasifikátoru pro různé hodnoty SNR. Vstupní trénovací i testovací vzorky mají vždy stejnou hodnotu SNR.

Tab. 6.1: Klasifikace výstřelů pro hodnotu SNR 30 dB.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	100	190	10	0	96,67	1	95	0,05	0

Výsledky se pro odstup signál-šum 25 dB a 20 dB nijak nelišily od 30 dB. Zhoršení přišlo až s testováním dat s SNR 15 dB a nižším. Výsledky jsou následující:

Tab. 6.2: Klasifikace výstřelů pro hodnotu SNR 15 dB.

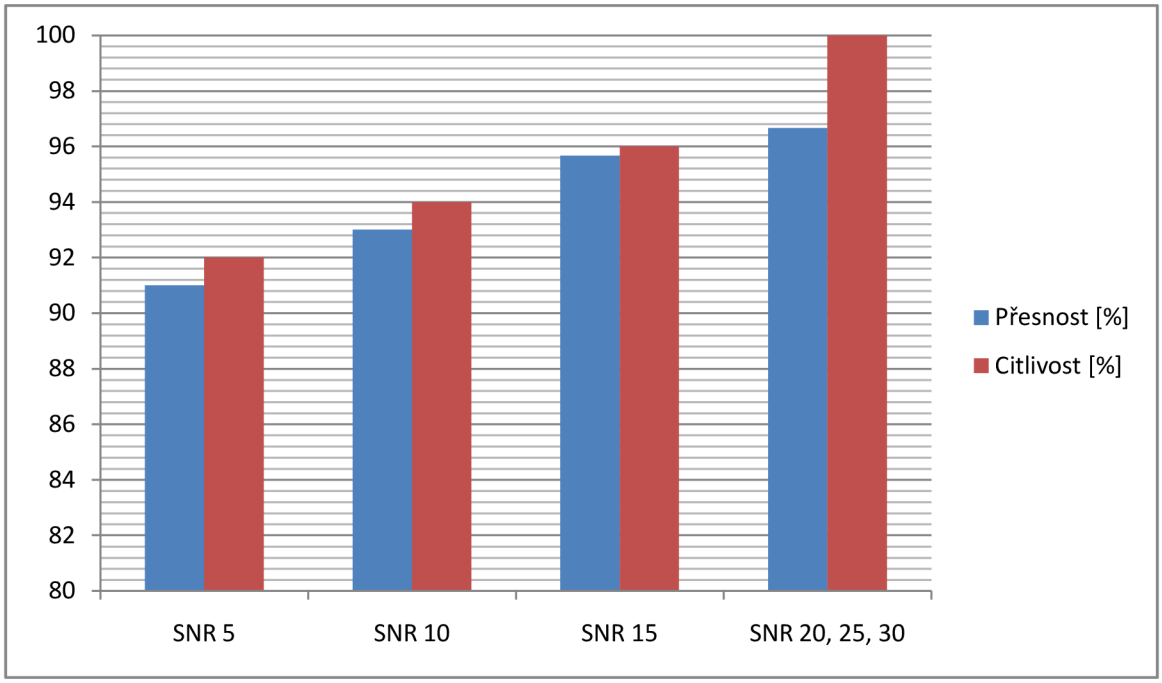
CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	96	191	9	4	95,67	96	95,5	0,045	0,04

Tab. 6.3: Klasifikace výstřelů pro hodnotu SNR 10 dB.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	94	185	15	6	93	94	92,5	0,075	0,06

Tab. 6.4: Klasifikace výstřelů pro hodnotu SNR 5 dB.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	92	181	19	8	91	92	90,5	0,095	0,08



Obr. 6.3: Prvotní testovaní metody podpůrných vektorů pro lineární funkci.

Přednastaveným vstupem do funkce SVM klasifikátoru - *fitcsvm* v prostředí Matlab je hodnota *linear*, která roztřídí data pomocí nadroviny, jak bylo ukázáno v kapitole 4.1.1. Funkce *fitcvm* může být dále specifikována pro použití jiných třídících funkcí, jako je Gaussova funkce. Toto je tedy forma experimentu, kdy byla použita stejná data jako

v předchozí kapitole a otestována na Gaussově funkci. Sice se snížila citlivost klasifikátoru, ale počet falešně pozitivních pozorování se snížil natolik, že se celková přesnost systému zvýšila. Ukázka kódu pro Gaussovu funkci:

```
klasifikator = fitcsvm(training,a,'KernelFunction','Gaussian','KernelScale','auto')
```

Tab. 6.5: Klasifikace výstřelů pro hodnotu SNR 10 dB za použití Gaussovy funkce.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	98	197	3	2	98,33	98	98,5	0,015	0,02

Pro hodnoty SNR 30 dB, 25 dB a 20 dB se výsledky opět nijak nelišily.

Tab. 6.6: Klasifikace výstřelů pro hodnotu SNR 15 dB za použití Gaussovy funkce.

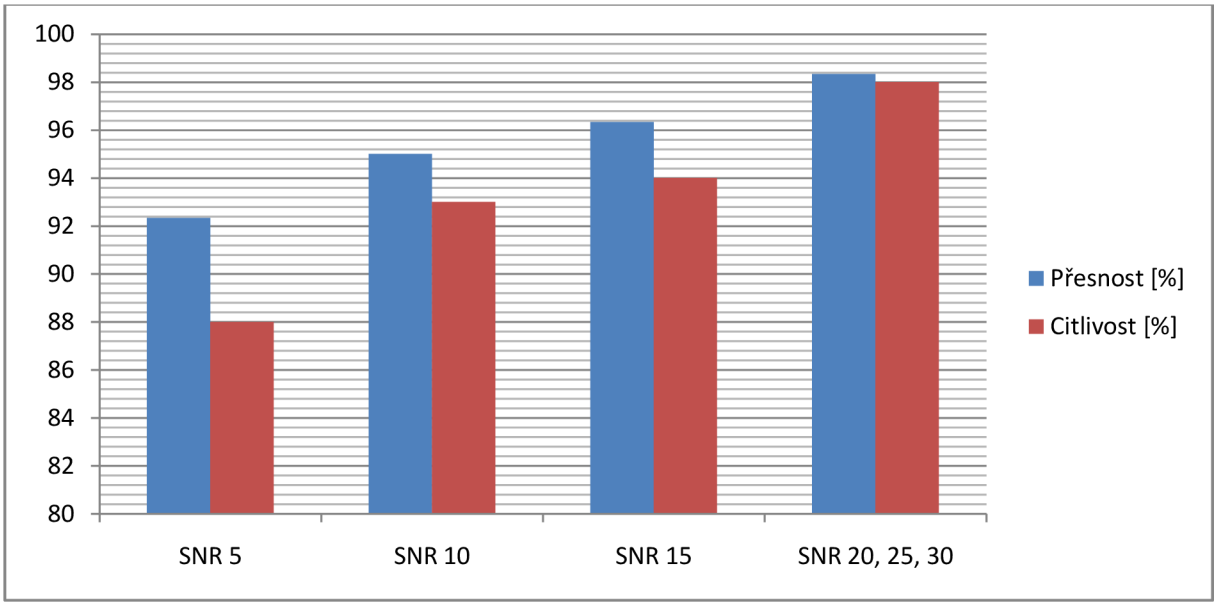
CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	94	195	5	6	96,33	94	97,5	0,025	0,06

Tab. 6.7: Klasifikace výstřelů pro hodnotu SNR 10 dB za použití Gaussovy funkce.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	93	192	8	7	95	93	96	0,04	0,07

Tab. 6.8: Klasifikace výstřelů pro hodnotu SNR 5 dB za použití Gaussovy funkce.

CP	CN	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
100	200	88	189	11	12	92,33	88	94,5	0,055	0,12



Obr. 6.4: Prvotní testovaní metody podpůrných vektorů pro Gaussovu fukci.

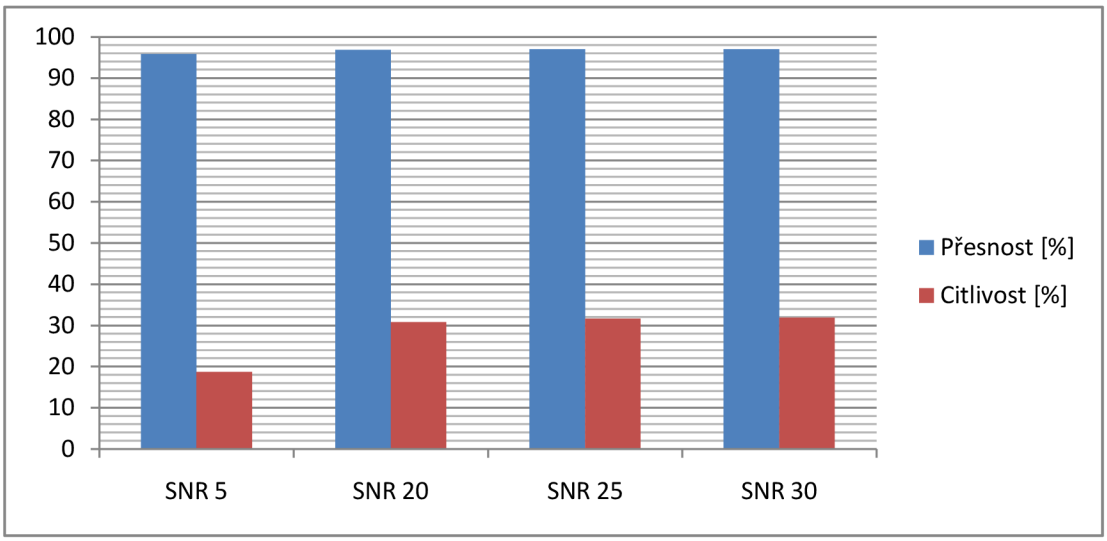
7 Testování pro celou databázi

Dále byl naučený SVM klasifikátor vyzkoušen na celé dostupné databázi. Program avšak nebyl automatizovaný, takže byl nepoužitelný pro taková kvanta dat. Byl proto vytvořen nový skript pro matlab `bakalarskaprace.m`. Tento skript nejdříve vybere z testovací databáze jen soubory s určitou koncovkou, jelikož název každého wave souboru z databáze byl zakončen číslem 1-6 podle jeho hodnoty SNR. Dále je načte a rozkouskuje na sekundové intervaly s půlsekundovým překryvem funkcí `enframe`. Pro tyto intervaly jsou získány melovské kepstrální koeficienty a opět pomocí funkce `predict` předpovězen výsledek. Poté je za pomocí přiložených XML souborů získána pravdivá poloha všech výstřelů, poté porovnána s předpovědí a je vypočten počet správně a falešně pozitivních a správně a falešně negativních detekcí. Nakonec je vypočtena přesnost, citlivost, specifičnost, falešně pozitivní a negativní míry a vypsány výsledky.

Po otestovaní klasifikátoru z předchozího kroku bylo na první pohled zřejmé, že je tento klasifikátor naučen na příliš nízkém počtu trénovacích zvuků, a to hlavně u pozadí. Byl tedy vytvořen druhý program `pretrenovaniklasifikatoru.m`, který podobným způsobem jako výše uvedený zpracovává celou databázi trénovacích dat a klasifikátor byl přeučen na celé databázi trénovacích zvuků. Tento skript opět načte jen soubory s vybranou hodnotou SNR. Poté postupně projede celou matici Train, což je matice všech přiložených XML souborů převedených do Matlabovské struktury. V ní jsou obsaženy všechny události roztříděné podle typu na čtyři skupiny. Pokud se jedná o výstřel (skupina 3), je z matice Train získána informace o začáteční a koncové sekundě a vynásobena vzorkovací frekvencí, aby byla zjištěna poloha výstřelu v souboru v pořadí vzorků. Jelikož natrénování klasifikátoru potřebuje všechny vstupní soubory stejně dlouhé, tak pokud je výstřel kratší, než zvolená jedna sekunda, je sebrán kousek pozadí po skončení výstřelu, aby celková délka byla právě 32000 vzorků. Pro tyto sekundové úseky jsou poté zjištěny melovské kepstrální koeficienty a ty dosazeny po řádcích do trénovací matice výstřelů. V dalším cyklu jsou výstřely získány stejným způsobem, až na to, že zde jsou v jednom cyklu všechny uloženy a naráz vystřihnuty ze vstupního souboru (čistě jen vzorky výstřelů). Poté je celá vstupní nahrávka bez výstřelů rozkouskována na sekundové intervaly a pro tyto jsou opět získány melovské kepstrální koeficienty a dosazeny po řádcích do trénovací matice pozadí. Nakonec jsou tyto dvě matice spojeny a je vytvořen logický trénovací vektor, kde je nejdříve počet logických jedniček délky počtu řádků v trénovací matici výstřelů a poté počet logických nul délky řádků v matici pozadí. Tímto byly získány oba dva vstupy pro natrénování klasifikátoru a může být použita funkce *fitcsvm*. Zpracováním všech trénovacích zvuků bylo získáno celkem 700 výstřelů a 10976 zvuků pozadí.

Tab. 7.1: Klasifikace metodou podpůrných vektorů pro celou dostupnou databázi.

SNR	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
30	143	9659	3	297	97,03	31,92	99,97	0,00031	0,68080
25	142	9659	3	301	96,99	31,67	99,97	0,00031	0,68304
20	138	9655	7	310	96,87	30,80	99,93	0,00072	0,69196
5	91	9638	24	396	95,86	18,69	99,75	0,00248	0,81314

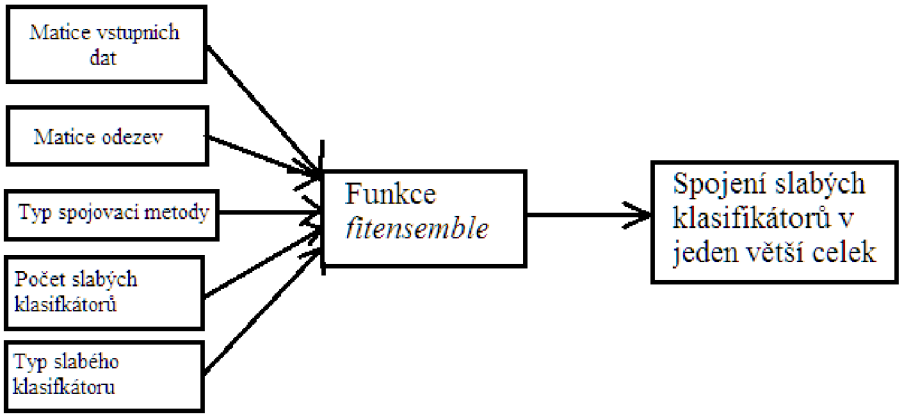


Obr. 7.1: Klasifikace metodou podpůrných vektorů pro celou dostupnou databázi.

Zde můžeme vidět, že se tato metoda nehodí pro vysoce zašuměné prostředí. Citlivost je velmi nízká, což by zase tak nevadilo, pokud by byl například i počet falešně pozitivních observací nízký až nulový, což ale není.

8 Optimalizace

Byla tedy použita metoda, která spojuje velké množství slabých klasifikátorů v jeden větší celek. K tomu slouží v prostředí matlab funkce *fitensemble.* Ta má pět vstupů. Zůstává matice trénovacích dat a vektor s logickými jedničkami a nulami jako u funkce *fitcsvm.* Dále je ve funkci volena spojovací metoda, počet slabých klasifikátorů a jejich typ. Na výběr je ze tří typů slabých klasifikátorů a několika metod pro jejich spojení. Upozorňuji, že tato metoda je mnohem náročnější, co se výpočetní síly týče, než metoda podpůrných vektorů a jedno měření proto trvalo i v řádech několika hodin (v závislosti na použité spojovací metodě a počtu použitých slabých klasifikátorů).



Obr. 8.1: Blokové schéma funkce fitensemble [18].

8.1 Typy slabých klasifikátorů

- **`'Discriminant'`** - diskriminantová analýza. Jedná se o klasifikační metodu. Předpokládá, že různé třídy generují data založená na různých Gaussových distribucích. Pro natrénování klasifikátoru funkce odhaduje parametry Gaussovy distribuce pro každou třídu. Tato metoda je doporučována hlavně pro spojovací metodu `'Subspace'`.
- **`'KNN'`** - počet nejbližších sousedů. Pokud máme sadu X s počtem n bodů a distanční funkci, KNN najde k nejbližších bodů k těmto bodům. KNN vyhledávací technika a algoritmy na KNN založené jsou široce používány v mnoha odvětvích. Zde je tato metoda bohužel použitelná pouze pro spojovací metodu `'Subspace'`.
- **`'Tree'`** - rozhodovací strom. Rozhodovací stromy předpovídají odezvu na vstupní data. Strom postupuje po uzlech od prvního, tzv. kořenového uzlu, až po tzv. uzel listu (poslední uzel, obsahuje předpokládanou odezvu). Každý krok v predikci zahrnuje kontrolu hodnoty jednoho prediktora (proměnné). Pokud predikce sedí, výsledkem je logická jednička a pokračuje na kontrolu dalšího prediktora tak dlouho, dokud nedojde k uzlu listu. Pokud kdekoliv během cesty predikce nesedí, výsledkem je logická nula a predikce se ukončí. Tato metoda se dá použít naopak na všechny spojovací metody kromě metody `'Subspace'`. Byla proto využita ke srovnání spojovacích metod a jelikož se ukázala jako nejpřesnější, tak i ve výsledných klasifikátorech.

8.2 Typy spojovacích metod

8.2.1 AdaBoostM1

AdaboostM1 je oblíbená metoda pro binární klasifikaci. Trénuje klasifikátory postupně. Pro každý slabý klasifikátor s indexem *t* AdaboostM1 vypočítá klasifikační chybu jako

$$\varepsilon_t = \sum_{n=1}^{N} d_n^{(t)} \mathrm{I}(y_n \neq h_t\,(x_n)), \tag{8.1}$$

kde x_n je vektor předpovídacích hodnot pro observaci n, y_n je pravdivá třída, h_t je předpověd klasifikátoru s indexem *t*,I je indikátorová funkce a $d_n^{(t)}$ váha observace n na kroku pořadí *t*.

AdaboostM1 poté zvýší váhu pro observace, které byly špatně klasifikovány klasifikátorem s indexem *t* a sníží váhy pro správně rozpoznané observace. Následující klasifikátor $(t+1)$ je tedy natrénován na datech s upravenou váhou $d_n^{(t+1)}$.

Po natrénování AdaboostM1 vypočítá předpověď pro nová data jako

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t\,(x), \tag{8.2}$$

kde α_t jsou váhy slabých hypotéz v naší spojovací funkci. Vypočteme jako

$$\alpha_t = \frac{1}{2}\log\frac{1-\varepsilon_t}{\varepsilon_t}. \tag{8.3}$$

Existuje také verze AdaboostM2, která se používá pro klasifikaci tří a více tříd.

8.2.2 LPBoost a Totalboost

Oba algoritmy LPBoost (zkratka pro linear programming boost) a TotalBoost provadí klasifikaci pomocí maximalizace šířky rozdělovacího pásma (anglicky margin, viz metoda podpůrných vektorů) mezi oběmi skupinami trénovacích dat. K tomu je zapotřebí optimalizačních algoritmů. LPBoost používá optimalizační algoritmus lineární programování a TotalBoost kvadratické programování. TotalBoost potřebuje sice více výpočetní síly, ale dokumentace pro Matlab jej upřednostňuje, jelikož skoro vždy vychází lépe. Upozorňuji, že pro obě metody je třeba mít Optimalizační toolbox™ pro Matlab.

Rozdělovací pásmo je vyžadováno co největší, protože tzv. generální chyba (chyba nových dat) je pravděpodobnost získání záporné šířky pásma. Toto se popisuje pomocí nerovnice pravděpodobnosti pro získání záporné šířky pásma, jak ji popsali R. Schapire a Y. Singer v roce 1999

$$P_{test}\,(m \le 0) \le P_{train}\,(m \le \theta) + O\left(\frac{1}{\sqrt{N}}\sqrt{\frac{\log^2\left(\frac{N}{V}\right)}{\theta^2} + \log\left(\frac{1}{\delta}\right)}\right), \qquad (8.4)$$

kde m je šířka rozdělovacího pásma, θ jakékoliv kladné číslo, V je Vapnik–Chervonenkisova dimenze, N velikost trénovací sady a δ jakékoliv malé kladné číslo. Tato nerovnost tedy říká, že k získání nízké generální chyby musíme v trénovacím setu minimalizovat počet observací pod šířkou pásma θ.

8.2.3 Bagging

Nebo také bag, je zkratkou pro "bootstrap aggregation" (anglicky svévolné seskupení). Tato metoda vytvoří spoustu kopií datasetu, na kterém je testována a naučí slabé klasifikátory na těchto replikách. Používá k tomu resamplované verze dat. Každou repliku původního datasetu získá náhodným zvolením N z N observací s náhradou, kde N je velikost datasetu. K získání finálního klasifikátoru jsou zprůměrovány odezvy slabých klasifikátorů ze všech kopií původního datasetu.

Jedná se o nejuniverzálnější z výše uvedených metod, jelikož je jediná, kterou lze použít pro binární klasifikaci, pro klasifikaci více tříd i regresi. LPBoost, TotalBoost i Bag se ale nedoporučují pro velké množství observací, jelikož se pak snižují jejich rozpoznávací vlastnosti.

8.2.4 Subspace

Subspace je metoda používáná na vylepšení přesnosti diskriminantové analýzy a metody *k* nejbližších sousedů. Subspace má tu výhodu, že využívá méně operační paměti, než jiné spojovací metody a dokáže se vypořádat s chybějícími hodnotami (v Matlabu tzv. `NaNs`).

Základní algoritmus typu Subspace používá tři parametry: *m* je počet dimenzí, které se mají nasamplovat v každém slabém klasifikátoru, *d* je počet dimenzí ve vstupních datech (počet sloupců v maticí vstupních dat) a *n* je počet slabých klasifikátorů. Subspace nejdříve vybere náhodný set *m* prediktorů z počtu *d* možných hodnot. Pak natrénuje jeden slabý klasifikátor za použití pouze *m* vybraných prediktorů. Následně opakuje předchozí dva kroky stále dokola, dokud není naučen zadaný počet *n* slabých klasifikátorů. Nakonec předpovídá pomocí průměru skóre všech naučených slabých klasifikátorů a klasifikuje kategorii s největším průměrným skóre.

8.2.5 RUSBoost

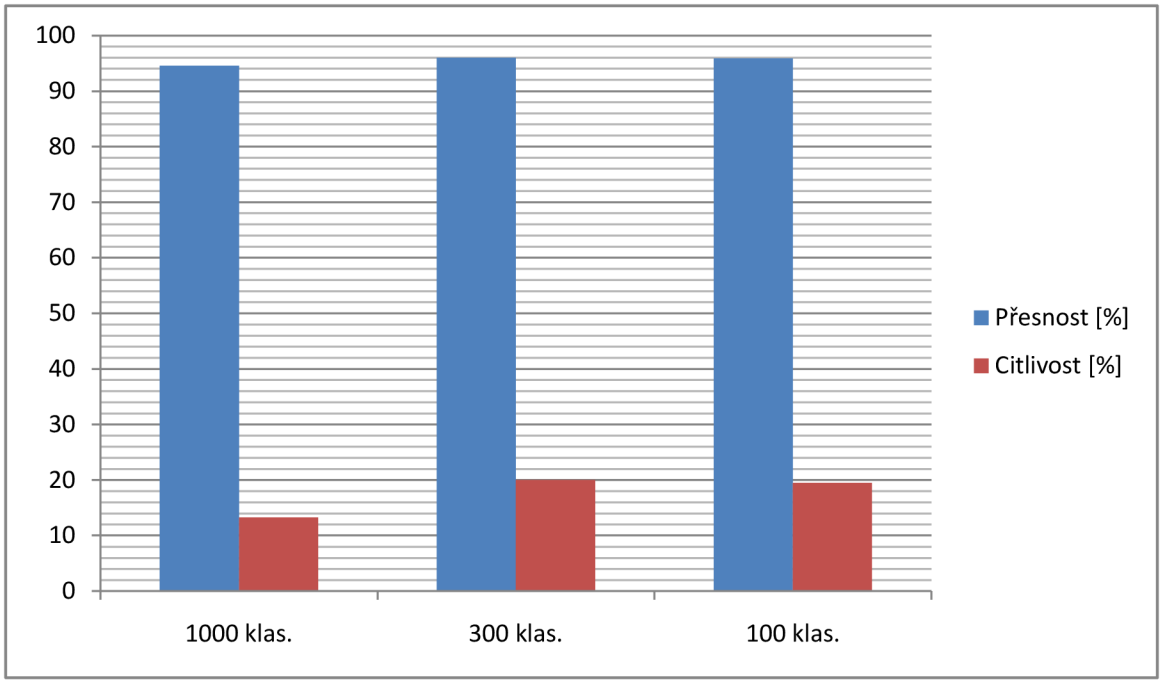
Z anglického random under sampling - náhodné podvzorkování. Tato metoda by měla být zvláště účinná pro klasifikaci nevyrovnaných dat, což znamená, že u binárního klasifikátoru je v trénovací matici počet observací jedné skupiny mnohem nižší než počet observací druhé skupiny (případ této práce, výstřelů bylo mnohonásobně méně než pozadí). RUSBoost zjistí počet pozorování oné mnohem menší skupiny jako konstantu *N*. Každá další větší třída (RUSBoost se dá použít i pro klasifikaci více než dvou tříd) je podvzorkována sebráním pouze *N* náhodných observací [18].

8.3 Výsledky

Při testování byla testována funkce `fitensemble` právě pro soubory s nejnižším poměrem signál-šum, aby došlo ke zjištění, která z metod se bude hodit právě pro tuto problematiku, jelikož právě toto je cílem této práce. Dokumentace k Matlabu doporučuje nejdříve vyzkoušet spojovací metodu AdaBoostM1 a až poté srovnávat ostatní metody. Byla tedy vyzkoušena pro různé počty slabých klasifikátorů.

Tab. 8.1: Porovnání různého počtu slabých klasifikátorů pro metodu AdaboostM1.

Počet klas.	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
1000	67	9550	112	439	94,58	13,24	98,84	0,01159	0,86759
300	97	9647	15	388	96,03	20	99,85	0,00155	0,8
100	95	9636	26	392	95,88	19,51	99,73	0,00269	0,80493



Obr. 8.2: Porovnání různého počtu slabých klasifikátorů pro metodu AdaboostM1.

Test pro tisíc slabých klasifikátorů byl proveden pro diskriminantovou analýzu, jelikož se ale tento typ slabého klasifikátoru doporučuje pouze pro metodu subspace, ve zbylých dvou měřeních proběhla klasifikace za pomocí rozhodovacích stromů. Vidíme, že pro 1000 slabých klasifikátorů vyšly výsledky velmi špatně. Proč tomu tak je? Zde ani tak nejde o to, že byla použita diskriminantová analýza, ale jedná se o jev známý jako syndrom přeučení. Každé spojení klasifikátorů má hranici, od které se s každým přidaným slabým klasifikátorem přesnost již nezlepšuje, naopak se začne při větším počtu zbytečných jedinců snižovat. Tento počet se různí podle použité spojovací funkce. Metody LPBoost a TotalBoost jsou sebeterminující, což znamená, že když je dosaženo slabého klasifikátoru, který už nezlepšuje výkon klasifikace, samy se ukončí a další slabé klasifikátory již netrénují, nezávisle na zadaném počtu, takže nemusí být předpovídán počet členů. Toto bylo zkontrolováno pro

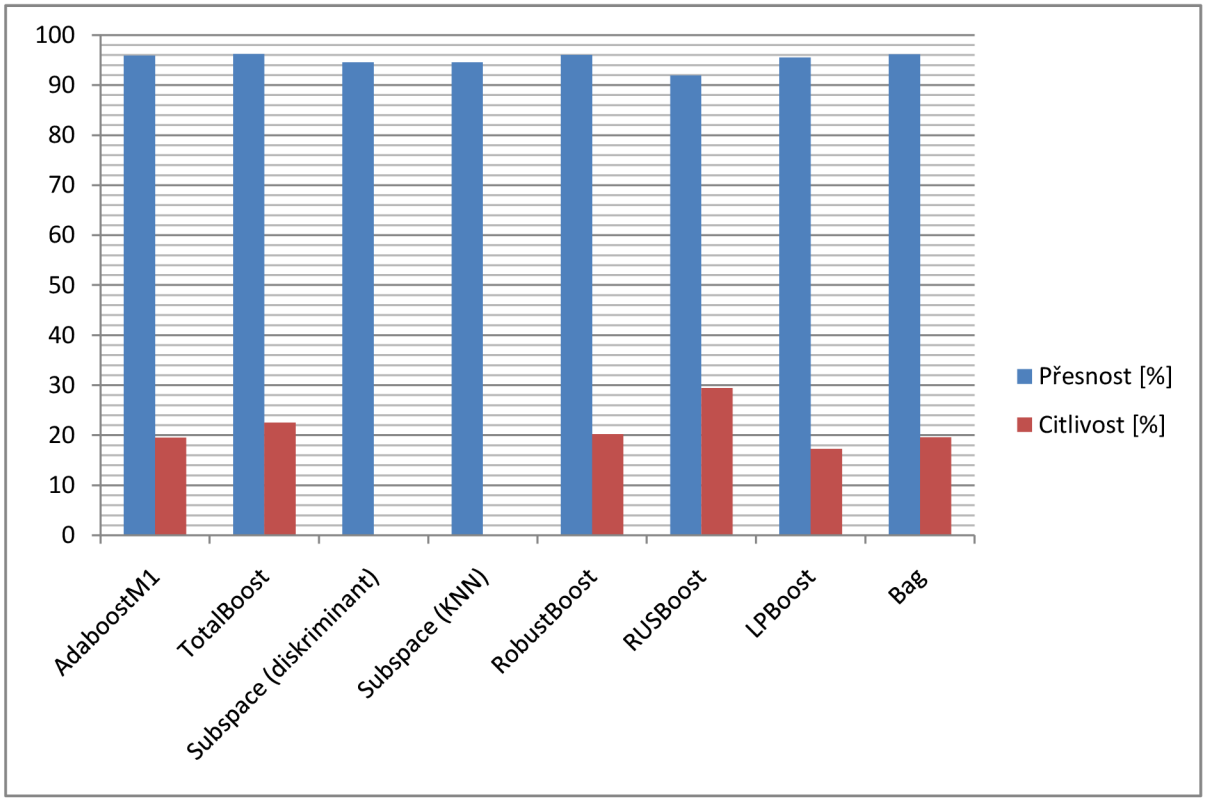
metodu TotalBoost a bylo zjištěno, že se sama ukončí po naučení 81. stromu. Všechna testování tedy proběhla pro stovku slabých klasifikátorů. Ukázka kódu:

```
klasifikator = fitensemble(training,c,'TotalBoost',100,'Tree').
```

Výsledky jsou následující:

Tab. 8.2: Srovnání spojovacích metod pro 100 slabých klasifikátorů a SNR 5dB.

Spojovací metoda	Typ slabého klasifikátoru	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifič nost [%]	FPM	FNM
AdaboostM1	Tree	95	9636	26	392	95,88	19,51	99,73	0,00269	0,80493
TotalBoost	Tree	107	9647	15	369	96,21	22,48	99,85	0,00155	0,77521
Subspace	Discriminant	0	9662	0	557	94,55	0	100	0	1
Subspace	KNN	0	9662	0	557	94,55	0	100	0	1
RobustBoost	Tree	98	9645	17	387	96,02	20,21	99,85	0,00176	0,79794
RUSBoost	Tree	132	9161	501	317	91,91	29,40	94,82	0,05185	0,70601
LPBoost	Tree	86	9618	44	413	95,50	17,23	99,55	0,00455	0,82766
Bag	Tree	95	9659	3	390	96,13	19,59	99,97	0,00031	0,80412



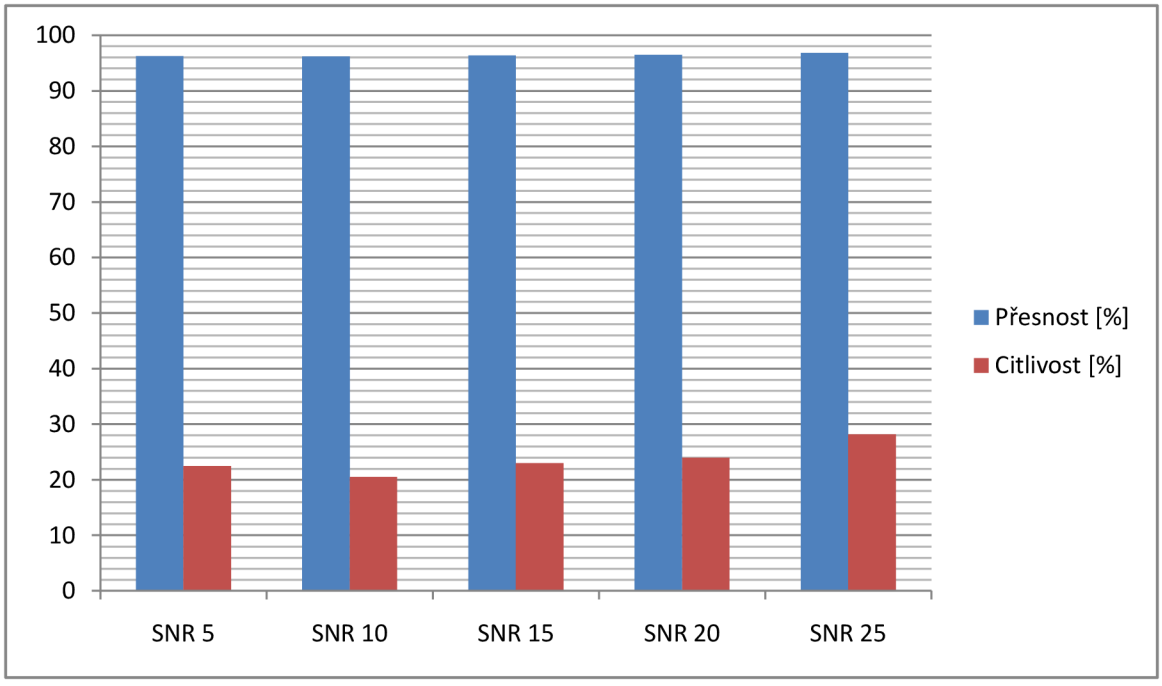
Obr. 8.3: Srovnání spojovacích metod pro 100 slabých klasifikátorů a SNR 5dB.

Zde si jako první všimneme, že metoda Subspace se nezávisle na použitém typu slabého klasifikátoru nehodí pro zašuměné prostředí. S nulovou citlivostí se moc pracovat nedá. Metody AdaboostM1 a LPBoost mají přes svou nízkou citlivost příliš velký poměr falešně pozitivních observací (AdaboostM1 by ale potřeboval větší počet slabých klasifikátorů).

Zajímavý výsledek vidíme u metody RUSBoost, jelikož z teoretického hlediska by měla vycházet nejlépe. Podle dokumentace k Matlabu RUSBoost dosahuje skvělých výsledků, pokud je jedna skupina trénovacích dat (naše výstřely) početně mnohem menší než druhá. RUSBoost sice dosáhl nejvyšší citlivosti, ale za cenu obrovské chyby. Nejlépe zde tedy vychází metoda TotalBoost. Má nejvyšší přesnost, relativně nízkou chybu a druhou nejvyšší citlivost. Dobře vyšla i metoda Bag, jelikož u takto zašuměných nahrávek jsou pouze tři falešně pozitivní observace něco, co je třeba brát v potaz. RobustBoost není daleko za nimi. Metoda TotalBoost byla tedy otestována i pro ostatní hodnoty SNR. Výsledky jsou obsaženy v následující tabulce:

Tab. 8.3: Metoda TotalBoost pro 100 rozhodovacích stromů při různých honotách SNR.

SNR	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
5	107	9647	15	369	96,21	22,48	99,85	0,00155	0,77521
10	99	9660	2	384	96,20	20,50	99,98	0,00021	0,79503
15	109	9655	7	365	96,33	23	99,93	0,00072	0,77004
20	112	9658	4	355	96,46	23,98	99,96	0,00041	0,76017
25	128	9662	0	326	96,78	28,20	100	0	0,71806



Obr. 8.4: Metoda TotalBoost pro 100 rozhodovacích stromů při různých honotách SNR.

Můžeme vidět, že už při odsupu signál-šum 25 dB bylo dosaženo nuly falešně pozitivních výsledků. Jediné, co kazí přesnost, je stále nízká citlivost. Byly proto využity statistické veličiny pro zlepšení přesnosti.

9 Využití statistických veličin

Jedním z nedostatků programu bylo, že klasifikátor se jednotlivé koeficienty MFCC učil tak, jak šly za sebou ve vektoru. Výsledkem pak bylo, že program výstřel vůbec nepoznal, pokud byly hodnoty v jiném pořadí. Pro zvýšení citlivosti byl proto program upraven. Hodnoty kepstrálních koeficientů byly ještě před naučením sebrány a byla provedena řada statistických operací. Tato výsledná čísla byla zpět dosazena do trénovací matice a byl na nich naučen klasifikátor. To samé pak bylo provedeno před detekcí. Výsledkem je velké zvýšení citlivosti klasifikátoru (bohužel se zvedl počet jak správně, tak i falešně pozitivních výsledků, i tak ale došlo ke zlepšení). V každém vektoru je použito celkem 15 statistických veličin, jedná se o zde uvedené (percentil je použit čtyřikrát pro 10, 30, 50 a 70%):

- **Aritmetický průměr** - Jedná se o součet prvků ve vektoru dělený jejich počtem

$$\mu = \frac{1}{N}\sum_{i=1}^{N} A_i, \tag{9.1}$$

  kde A_i je náš vektor a N počet prvků v něm.

- **Medián** - Medián je hodnota, která se ve vektoru seřazeném podle velikosti vyskytuje přesně uprostřed. Pokud má vektor sudý počet prvků, medián je průměrem obou hodnot nacházejících se uprostřed.
- **Rozptyl** - Rozptyl může být definován jako aritmetický průměr čtverců odchylek jednotlivých hodnot

$$S = \frac{1}{N-1}\sum_{i=1}^{N} |A_i - \mu|^2, \tag{9.2}$$

  kde A_i je náš vektor, N počet prvků v něm a μ aritmetický průměr.

- **Směrodatná odchylka** - Udává, jak jsou hodnoty rozloženy kolem průměru. Vypočítá se jako odmocnina z rozptylu.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N} |A_i - \mu|^2}. \tag{9.3}$$

- **Minimální a maximální hodnota** - Jak už název napovídá, jedná se o nejmenší a největší hodnotu ve vektoru.
- **Variační rozpětí** - Jedná se o rozpětí vektoru, tedy o rozdíl mezi největší a nejmenší hodnotou.
- **Absolutní odchylka střední hodnoty** - Absolutní odchylka je „vzdálenost“ hodnoty od průměru. Jedná se o průměr absolutních hodnot odchylek od aritmetického průměru.

$$\Delta i = \frac{1}{N}\sum_{i=1}^{N} |A_i - \mu|. \tag{9.4}$$

- **Percentil** - Udává číslo, pod které patří určené procento hodnot ve vektoru.
- **Mezikvartální rozsah** - Udává rozdíl mezi 25. a 75. percentilem.
- **Koeficient šikmosti** - Udává asymetrii hodnot kolem aritmetického průměru vektoru. Pokud se jedná o zápornou hodnotu, jsou data více rozprostřena nalevo od průměru. Pokud o kladnou hodnotu, jsou data více rozprostřena na pravé straně. Pokud je koeficient šikmosti nula, jedná se o perfektní symetrii. Vypočteme jako

$$s = \frac{E(x - \mu)^3}{\sigma^3}, \tag{9.5}$$

kde μ je aritmetický průměr, σ je směrodatná odchylka z x a $E_{(t)}$ reprezentuje předpokládanou hodnotu kvantity t.

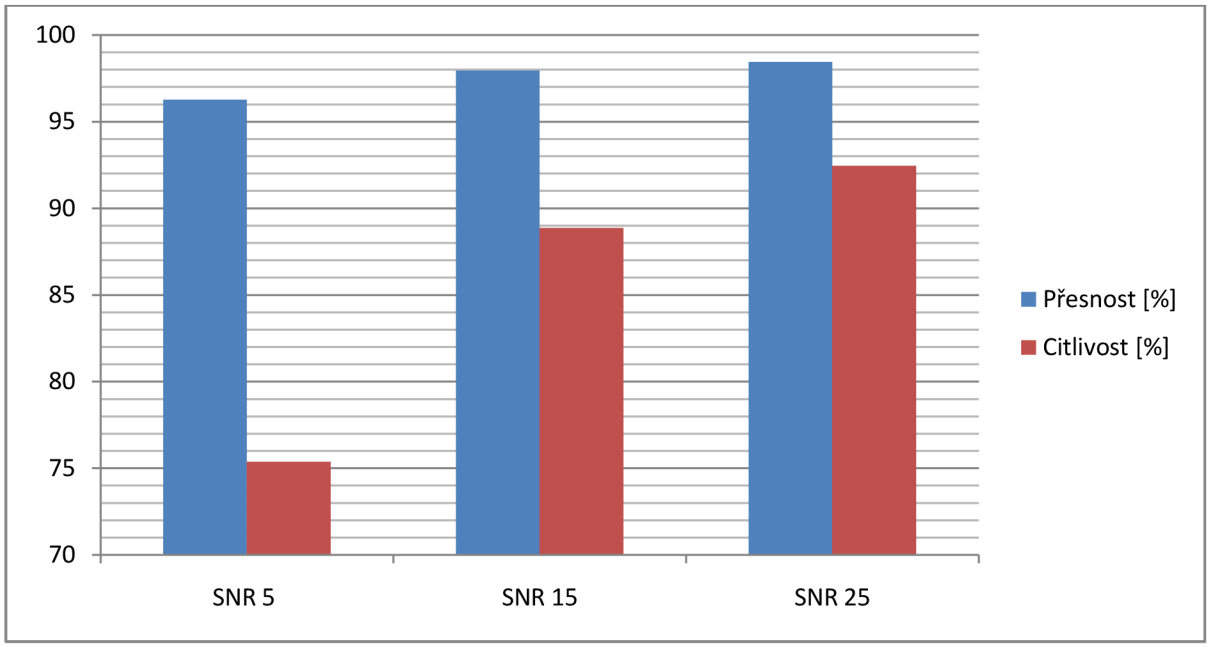
- **Koeficient špičatosti -** Podobné jako koeficient šikmosti, udává, jak moc jsou prvky rozprostřeny směrem ke stranám. Koeficient špičatosti normální distribuce je 3. Pokud jsou prvky více koncentrovány na krajích, je koeficient špičatosti více než 3 a pokud jsou rozprostřeny více uprostřed, je koeficient menší [19].

$$s = \frac{E(x - \mu)^4}{\sigma^4}. \tag{9.6}$$

Zde bylo opět testováno zlepšení na metodě TotalBoost, na několika vybraných hodnotách SNR, jelikož v předchozí kapitole vycházela nejlépe.

Tab. 9.1: Metoda TotalBoost pro 100 trénovacích stromů při využití statistických veličin.

SNR	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
5	315	9389	273	103	96,27	75,36	97,17	0,02826	0,24641
15	359	9500	162	45	97,94	88,86	98,32	0,01677	0,11139
25	367	9535	127	30	98,44	92,44	98,69	0,01314	0,07557

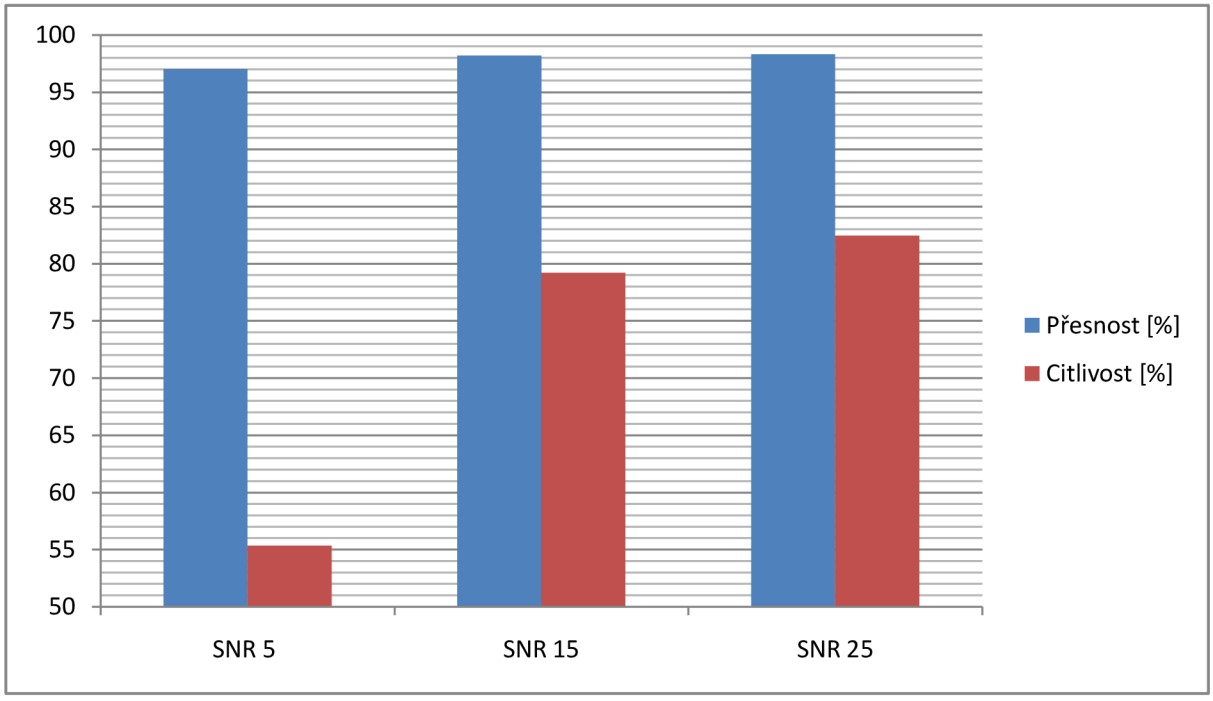


Obr. 9.1: Metoda TotalBoost pro 100 trénovacích stromů při využití statistických veličin.

Tyto výsledky jsou zatím nejpřesnější. Ještě byla otestována metoda Bag, jelikož v předchozí kapitole vycházela s nejmenší chybou.

Tab. 9.2: Metoda Bag pro 100 trénovacích stromů při využití statistických veličin.

SNR	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
5	238	9553	109	192	97,02	55,35	98,87	0,01128	0,44651
15	320	9564	98	84	98,19	79,21	98,99	0,01014	0,20792
25	329	9562	100	70	98,31	82,46	98,97	0,01035	0,17544

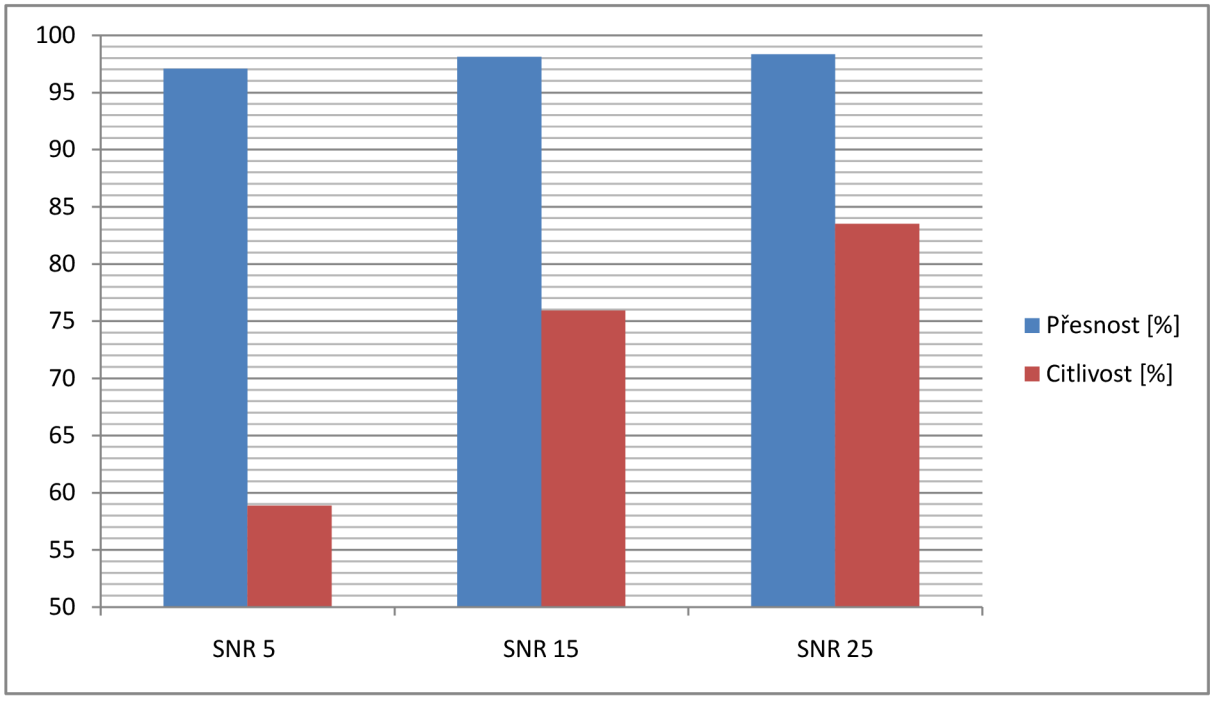


Obr. 9.2: Metoda Bag pro 100 trénovacích stromů při využití statistických veličin.

Tato metoda na nejvíce zašuměných souborech vychází opět s nižší citlivostí, než metoda TotalBoost, ale opět je chyba o tolik menší, že je přesnější. U čistších souborů ale již tak dobře nevychází. Tato metoda ale není sebeterminující, tak byl proveden test pomocí funkce `loss` za účelem zjištění, jestli metoda bag potřebuje 100 trénovacích stromů. Po vykreslení funkce do grafu bylo zjištěno, že chyba je po 55. slabém klasifikátoru již nulová. Otestoval jsem proto metodu ještě jednou právě pro tento počet trénovacích stromů.

Tab. 9.3: Metoda Bag pro 55 trénovacích stromů při využití statistických veličin.

SNR	SP	SN	FP	FN	Přesnost [%]	Citlivost [%]	Specifičnost [%]	FPM	FNM
5	252	9544	118	176	97,09	58,88	98,78	0,01221	0,41121
15	306	9568	94	97	98,10	75,93	99,03	0,00973	0,24069
25	334	9562	100	66	98,35	83,50	98,97	0,01035	0,165



Obr. 9.3: Metoda Bag pro 55 trénovacích stromů při využití statistických veličin.

Zde můžeme vidět, že pro hodnoty 5 dB a 25 dB SNR vyšly klasifikátory přesnější a citlivější s menší počtem falešně negativních výsledků. Klasifikátor pro 15 dB SNR vyšel ovšem hůře.

10 Závěr

Tato práce nás nejdříve seznámila se současnou problematikou detekce nebezpečných událostí a ukázala obecné schéma, jak takové detekce dosáhnout. Následně bylo řečeno něco o segmentaci signálů a jak z nich získat typické příznaky důležité pro naučení klasifikátoru. Poté byla podrobně popsána teorie melovských kepstrálních koeficientů a jak je získat. V další kapitole bylo uvedeno něco z teorie o strojovém učení a klasifikátorech a následně rozebráno několik druhů klasifikátorů používaných pro danou problematiku. V praktické části bakalářské práce byl nejdříve uveden postup při zpracování databáze MIVIA. Poté probíhalo testování metody podpůrných vektorů. Tato metoda byla otestována pro lineární i kvadratickou gaussovu funkci. Gaussova funkce byla méně citlivá, ale byl zde mnohem menší poměr falešně pozitivních observací, kvůli čemuž byla přesnější. Metoda celkově dosahovala dobrých výsledků především v čistším prostředí, u hodně zašuměných nahrávek již byla dosti chybová. U nejnižší hodnoty SNR měla i přes svou malou citlivost velký počet jak falešně pozitivních, tak falešně negativních výsledků. V další kapitole byly testovány spojovací funkce pro spojení většího počtu slabých klasifikátorů. S výjimkou metody Subspace se jednalo vždy o testovací stromy. Toto testování již probíhalo pouze pro nahrávky s odstupem signál-šum 5 dB, jelikož to, jak se klasifikátor chová právě v zašuměném prostředí, bylo v hlavním zájmu práce. Nejdříve proběhlo srovnání různého počtu slabých klasifikátorů pro metodu AdaBoostM1. Nejpřesnější výsledek byl pro 300 slabých klasifikátorů, kde přesnost nepatrně přesahovala 96%. Dále došlo k porovnání vybraných spojovacích metod pro stovku slabých klasifikátorů. Porovnány byly metody AdaBoostM1, TotalBoost, RobustBoost, RUSBoost, LPBoost, a Bag pro rozhodovací stromy společně s metodou Subspace pro diskriminantovou analýzu a *k* nejbližších sousedů. Metoda AdaBoostM1 nedosahovala takové přesnosti, jako při třech stech stromů a potřebovala by naučit větší počet slabých klasifikátorů. Metoda TotalBoost vyšla s nejlepším výsledkem. Měla přesnost 96,21% a druhou největší citlivost a tím pádem i druhý nejvyšší počet správně pozitivních (107) a falešně negativních výsledků (369). Přesto si zachovala nízkou chybu u falešně pozitivních observací (15). Dobře také vyšla metoda Bag, která sice neměla takovou citlivost a přesnost jako metoda TotalBoost, ale klasifikovala 95 správně pozitivních výsledků a jen 3 falešně pozitivní, což je u takto zašuměných souborů dobrý výsledek. S největší citlivostí a počtem správně pozitivních observací vyšla metoda RUSBoost, která se ale vyznačovala obrovskou chybou v podobě falešně pozitivních výsledků, klasifikovala jich 501, což vyústilo v nejnižší přesnost ze všech klasifikátorů (pouze 91,91%). Zde bych podotkl, že nezávisle na použité metodě k chybné detekci docházelo většinou u vzorků rozbití skla, které se vyznačovalo rapidním nárustem amplitudy, což je podobné i pro výstřel. Špatně vyšla také metoda Subspace, která měla bez ohledu na druh použitého slabého klasifikátoru nulovou citlivost - klasifikovala všechny testované vzorky jako pozadí. Metoda LPBoost také nebyla moc přesná a při pouhých 86 správně pozitivních výsledcích měla druhou nejvyšší chybu. Nakonec metoda RobustBoost vyšla dost podobně jako metoda TotalBoost, akorát byly všechny hodnoty trochu horší. Jako nejlepší byla metoda TotalBoost dále testována i pro ostatní hodnoty SNR. Podle předpokladu se výsledky s rostoucí hodnotou zlepšovaly. U odstupu signál-šum 25 dB už nebyl ani jeden falešně pozitivní výsledek, ale všechny tyto klasifikátory měly stále velmi nízkou citlivost (u 25 dB byla citlivost nejvyšší a stále pouze

28%). Hodnoty pro klasifikátor 10 dB SNR vyšly s nižší citlivostí, než zbylé. Zde bych podotkl, že žádné dva klasifikátory, i když naučené na stejných datech, nejsou vždy úplně stejné, vždy se trochu liší (v našem případě v řádech setin, max jedné desetiny procenta). Kvůli nízké citlivosti byly v práci využity statistické veličiny, kde se klasifikátor neučil přímo melovské kepstrální koeficienty, jak tomu bylo dosud, ale jejich průměr, rozptyl, směrodatnou odchylku a další. Zlepšení bylo testováno na metodě TotalBoost a Bag, jelikož tyto v předchozím testování vycházely nejlépe, opět pro 5 dB SNR a pár vybraných dalších hodnot. Citlivost se zvýšila rapidně. Pro TotalBoost pro 5 dB SNR se zvýšila na 75,36%. Počet správně pozitivních se zvedl na 315, ale počet falešně pozitivních na 273, což je příliš. Pro 15 dB SNR už to bylo jen 162 falešně pozitivních při 359 správně pozitivních a pro 25 dB 127 falešně pozitivních při 367 správně pozitivních a pouze 30 falešně negativních výsledcích. Zde se citlivost vyšplhala již na 92,44% a celková přesnost klasifikátoru byla 98,44%. Pro méně citlivou metodu Bag, byla citlivost u nahrávek s 5 dB SNR 55,35%. Počet správně pozitivních byl 238, ale počet falešně pozitivních pouze 109, takže celková přesnost klasifikátoru byla větší (97,02%), než u metody TotalBoost (96,27%). Počet falešně pozitivních se ale u této metody se vzrůstající hodnotou SNR téměř nelepšil, takže pro čistší soubory již metoda Bag nedosahuje tak dobrých výsledků jako metoda TotalBoost. Funkce TotalBoost je ale sebeterminující a Bag není, takže byl ještě zjištěn přesný počet slabých klasifikátorů, které metoda Bag pro tato data potřebuje a proveden ještě jeden test pro 55 trénovacích stromů. Výsledek pro 5 a 25 dB SNR se ještě zlepšil, ale výsledek pro 15 dB SNR se lehce zhoršil a také byl počet falešně pozitivních výsledků opět téměř nezávislý na hodnotě SNR. Rychlost trénování i testovaní se ale velmi zvýšila a přesnost pro 5 dB SNR se zvedla na 97,09%. Závěrem práce tedy je, že pro tuto problematiku se metoda podpůrných vektorů, především u velmi zašuměných nahrávek příliš nehodí. Také bylo zjištěno, že pokud chceme rapidně zvýšit citlivost klasifikátoru, je dobré učit jej statistické veličiny daných příznaků, místo příznaků samotných. Pokud je tedy zapotřebí dobrého výkonu ve velmi zašuměném prostředí, doporučil bych metodu Bagging pro spojení více klasifikačních stromů a v tišším prostředí bych využil metodu TotalBoost, jelikož tyto v daných oblastech vykazují nejlepší výsledky.

Literatura

[1] FOGGIA P., PETKOV N., SAGGESE A., STRISCIUGLIO N., VENTO M., *Reliable Detection of Audio Events in Highly Noisy Environments,* Pattern Recognition Letters, 9. 7. 2015, Dostupné na internetu: <*http://dx.doi.org/10.1016/j.patrec.2015.06.026*> ISSN 0167-8655

[2] OPPENHEIM A., SCHAFER R., BUCK J.. *Discrete-Time Signal Processing*. Upper Saddle River, NJ: Prentice Hall, 1999, ISBN:0-13-754920-2

[3] HEUSER J., *Preprocessing,* Speech recognition wiki, Technische Universität München, 28.12.2014, Dostupné na internetu: < *http://recognize-speech.com/preprocessing/*>

[4] SMITH J., *Spectral Audio Signal Processing*, 2011, Kapitola Hammingovo okno, Dostupné na internetu: < *https://ccrma.stanford.edu/~jos/sasp/Hamming_Window.html* > ISBN 978-0-9745607-3-1.

[5] LUTTER M., *Feature extraction,* Speech recognition wiki, Technische Universität München, 4.1.2015, Dostupné na internetu: < *http://recognize-speech.com/feature-extraction/*>

[6] MERMELSTEIN P., "*Distance Measures for Speech Recognition – Psychological and Instrumental"*, Pattern Recognition and Artificial Intelligence*,* 1976, Dostupné na internetu: < *http://web.haskins.yale.edu/sr/SR047/SR047_07.pdf/*>

[7] VAN LOAN C.*, Computational Frameworks for the Fast Fourier Transform,* SIAM, 1992, ISBN: 978-0-89871-285-8

[8] LYONS J., *MFCC Tutorial, Practical Cryptography*, Dostupné na internetu: <*http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/*>

[9] LOGAN B., *Mel Frequency Cepstral Coefficients for Music Modelling*, Dostupné na internetu: <*https://pdfs.semanticscholar.org/afe2/38f9ac0678e840ff1521f49c6fe749856109.pdf*>

[10] RUSSEL S., NORVIG P., *Artificial Intelligence: A Modern Approach (2nd edition)*, Prentice Hall, 2003, *ISBN 978-0137903955*

[11] STEINWART I., CHRISTIMANN A., *Support Vector Machines***,** Springer New York, 2008, ISBN: 978038777242

[12] KECMAN, V. *Learning and soft computing: support vector machines, neural networks, and fuzzy logic models*. Cambridge, Mass.: MIT Press, c2001. ISBN 0-262-11255-8.

[13] MAHDAL J., *Srovnání klasifikátorů,* 9.12.2006, Dostupné na internetu: <*http://www.fit.vutbr.cz/study/courses/SRE/public/2007-08/projekty/mahdal_sre.pdf*>

[14] JAKLIN P., *Neuronové sítě,* Dostupné na internetu: <*http://cgg.mff.cuni.cz/~pepca/prg022/mucha/*>

[15] REYNOLDS D., *Gaussian Mixture Models,* MIT Lincoln Laboratory, Dostupné na internetu: <*http://www.ee.iisc.ac.in/people/faculty/prasantg/downloads/GMM_Tutorial_Reynolds.pdf*>

[16] SIVIC J., *Efficient visual search of videos cast as text retrieval (PDF),* květen 2009, *Dostupné na internetu:* < *http://www.di.ens.fr/~josef/publications/sivic09a.pdf*>

[17] FAWCETT T., *An Introduction to ROC Analysis (PDF), 2006,* Pattern Recognition Letters. *Dostupné na internetu:* <*http://people.inf.elte.hu/kiss/11dwhdm/roc.pdf*>

[18] MathWorks™, *Statistics and Machine Learning Toolbox™ User's Guide (R2017a)*, 2017, Dostupné na internetu: <*https://www.mathworks.com/help/pdf_doc/stats/stats.pdf*>

[19] MathWorks™, *Matlab User's Guide (R2017a)*, část Funkce - Deskriptivní statistika, 2017, Dostupné na internetu: <*https://www.mathworks.com/help/matlab/functionlist.html#bs5838u-4*>

Seznam použitých zkratek a veličin

CN - celkový počet negativních výsledků

CP - celkový počet pozitivních výsledků

dB - decibel - jednotka zesílení

DCT - discrete cosine transform - diskrétní kosinová transformace

DFT - discrete Fourier transform - diskrétní Fourierova transformace

FFT - fast Fourier transform - rychlá Fourierova transformace

FN - počet falešně negativních výsledků

FNM - falešně negativní míra

FP - počet falešně pozitivních výsledků

FPM - falešně pozitivní míra

GMM - gaussian mixture model - gaussovský směsný model

Hz - hertz - jednotka frekvence

KNN - k nearest neighbours - počet k nejbližších sousedů

MFCC - mel frequency cepstral coefficients - melovské kepstrální koeficienty

ms - milisekunda - tisícina sekundy

SN - počet správně rozpoznaných negativních výsledků

SNR - signal-to-noise ratio - hodnota odstupu užitečného signálu od šumu

SP - počet správně rozpoznaných pozitivních výsledků

SVM - support vector machines - metoda podpůrných vektorů

XML - extensible markup language - značkovací jazyk pro výměnu dat mezi aplikacemi

Seznam příloh

A Obsah CD 48
A.1 Soubory pro prostředí matlab ke spuštění bakalářské práce 48
A.2 Matice 48
A.3 Nahrávky 48
A.4 Bakalářská práce 48
B Návod ke spuštění 49

A Obsah CD

A.1 Soubory pro prostředí matlab ke spuštění bakalářské práce

`preuceniklasifikatoru.m` - Skript pro prostředí Matlab na naučení vybraného klasifikátoru z trénovací databáze podle zadaných hodnot.

`bakalarskaprace.m` - Skript na otestování naučeného klasifikátoru a výpočet výsledků.

`loss.m` - Skript pro grafické znázornění závislosti klasifikační chyby na počtu slabých klasifikátorů.

`mfcc.m` - Program pro výpočet melovských kepstrálních koeficientů.

`trifbank.m` - Skript s bankou trojůhelníkových filtrů, bez něj skript `mfcc.m` nefunguje.

`vec2frames.m` - Rozděluje vektor hodnot na překrývající se okna, další funkce nutná pro MFCC.

`enframe.m` - Rozdělí vstupní signál na okna o vybrané délce a překryvu a vytvoří matici, kde jsou tato okna uložena po řádcích.

A.2 Matice

`bakalarskaprace.mat` - Obsahuje celé jedno měření (bez klasifikátoru), včetně všech vstupních hodnot pro funkce a matic train.mat a test.mat, což jsou trénovací a testovací XML soubory již převedené do jedně velké Matlabem rozpoznatelné struktury, tím pádem nebylo nutné přikládat skripty `xml_load` a `xml_parse` a XML soubory.

`TotalBoost 5 dB, 15 dB a 25 dB SNR` - Naučené klasifikátory pro metodu TotalBoost při využití statistických veličin pro tři hodnoty SNR a 100 trénovacích stromů.

`Bag 5 dB, 15 dB a 25 dB SNR` - Opět tři naučené klasifikátory pro metodu Bag při využití statistických veličin pro tři hodnoty SNR a 100 trénovacích stromů.

A.3 Nahrávky

Na disku dále najdeme složky training a testing, ve kterých pod složkou sounds najdeme čtyři zvukové soubory ve formátu .wav z trénovací a testovací databáze ve třech verzích SNR.

A.4 Bakalářská práce

Obsahem CD je i plná elektronická verze bakalářské práce ve formátu PDF.

B Návod ke spuštění

Pro spuštění programu je třeba mít nainstalováno na svém zařízení prostředí Matlab. Pokud máte hodně zastaralou verzi programu a program vypíše chybu ve funkci audioread, musíte ji přepsat na funkci wavread, jelikož tato je již v novějších verzích nahrazena funkcí audioread.

1. Spusťte program Matlab.exe.
2. Vlevo nahoře nad oknem "Current folder" můžete vidět tlačítko "Browse for folder". Klikněte na něj a vyberte složku (CD) s bakalářskou prací.
3. Dvojklikem na jednotlivé soubory načtěte všechny přiložené soubory pro Matlab s koncovkou `.m`.
4. Zde záleží, jak chcete pokračovat. Pokud si přejete naučit vlastní klasifikátor a otestovat funkci `preuceniklasifikatoru.m`, pokračujte na další krok. Pokud chcete otestovat jeden z přiložených klasifikátorů, tak ho dvojklikem načtěte ze seznamu, načtěte také `bakalarskaprace.mat` a přeskočte na krok 7.
5. Vyberte v editoru ze seznamu načtených skriptů funkci `preuceniklasifikatoru.m`. Zde máte na řádku č. 6 ve funkci `strfind` předvolenou jedničku pro soubory s odstupem signál-šum 5 dB. Můžete ji změnit na číslo `3`, pokud chcete trénovat pro soubory s 15 dB SNR nebo `5`, pokud chcete trénovat pro 25 dB SNR. Na konci programu (řádek 124) vidíme trénovací funkci `fitensemble`. Zde je přednastavena metoda TotalBoost, ale i metoda, i počet trénovacích stromů je možno měnit. Zde upozorňuji, že počet trénovacích a testovacích dat je jen pro vyzkoušení a je velmi nízký, což se projeví na výsledcích. Vyberte také menší počet slabých klasifikátorů, než je testováno v této bakalářské práci. Ještě bych podotkl, že u metody Bag je třeba navolit, že se jedná o klasifikaci a ne regresi tím, že za `'Tree'` (ještě uprostřed závorky) zkopírujete `,'Type','Classification'`. Pokud máte navoleno, stisknutím klávesy F5, nebo tlačítka run (zelený trojúhelník nahoře uprostřed) spustíte měření.
6. (Volitelné). Pokud chcete vidět klasifikační chybu jednotlivých klasifikátorů, vyberte ze seznamu načtených skriptů funkci `loss.m` a stiskem tlačítka F5 spusťte měření. Vykreslí se Vám graf závislosti klasifikační chyby na počtu klasifikátorů.
7. Ke klasifikaci vyberte ze seznamu načtených skriptů `bakalarskaprace.m`. Zde na pátém řádku můžete vidět pod funkcí `strfind` navolenou `1`. Pokuď budete testovat klasifikátor naučený pro jiné hodnoty SNR než 5, změňte tuto hodnotu (č. `3` pro 15 dB SNR a č. `5` pro 25 dB SNR). Nyní stiskem klávesy F5, nebo tlačítka run (zelený trojúhelník nahoře uprostřed) spusťte měření. Po chvíli se Vám výsledky vypíší do okna "Command Window".