ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

F3

Fakulta elektrotechnická
Katedra kybernetiky

Bakalářská práce

Expertní systém pro určování dominantní projevované emoce z hlasu

Jakub Šmíd
Kybernetika a robotika

20. května 2022
Vedoucí práce: Ing. Jan Hejda, Ph.D.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šmíd** Jméno: **Jakub** Osobní číslo: **483554**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Expertní systém pro určování dominantní projevované emoce z hlasu

Název bakalářské práce anglicky:

Expert System for Determining the Dominant Expressed Emotion from the Voice

Pokyny pro vypracování:

Cílem práce je návrh expertního systému pro identifikaci dominantní projevované emoce z hlasu mluvčího. Navržený systém bude schopen určovat příslušnost hlasem projevované emoce k jedné ze šesti sledovaných kategorií vycházejících z teorie základních emocí [4] a využívaných v současných validovaných datasetech. V rámci práce bude dále simulací otestována přesnost systému při analýze hlasu z VHF/UHF vysílaček.
Student v rámci práce analyzuje a použije dostupné relevantní validované datasety (EmoDB, RAVDESS, CREMA-D) a metody pro hodnocení hlasu. Vybranou metodu následně implementuje a na základě experimentů navrhne její parametry pro klasifikaci. Student otestuje přesnost systému pro jiné slovní fráze než ty, které byly součástí trénovací a validační množiny.
Hlavní úkoly:
1) Analyzujte dostupné validované datasety EmoDB, RAVDESS a CREMA-D.
2) Navrhněte metodu analýzy hlasu včetně její architektury, augmentace dat a případné extrakce příznaků.
3) Na základě experimentů navrhněte parametry vytvořeného systému a srovnejte jeho přesnost s jinými state-of-art řešeními.
4) Otestujte a zhodnoťte přesnost systému pro jiné slovní fráze než ty, které jsou součástí trénovací a validační množiny.
5) Simulací omezením frekvenčního pásma a přidáním umělého šumu otestujte a zhodnoťte přesnost systému při analýze hlasu z VHF/UHF vysílaček.
6) V jazyce Python implementujte aplikaci pro demonstraci navrženého systému.

Seznam doporučené literatury:

[1] Fayek, Haytham M., Margaret Lech, and Lawrence Cavedon. "Evaluating deep learning architectures for Speech Emotion Recognition." Neural Networks 92 (2017): 60-68
[2] Seehapoch, Thapanee, and Sartra Wongthanavasu. "Speech emotion recognition using support vector machines." 2013 5th international conference on Knowledge and smart technology (KST). IEEE, 2013
[3] Swain, Monorama, Aurobinda Routray, and Prithviraj Kabisatpathy. "Databases, features and classifiers for speech emotion recognition: a review." International Journal of Speech Technology 21.1 (2018): 93-120
[4] Ekman, Paul. "An argument for basic emotions." Cognition & emotion 6.3-4 (1992): 169-200

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Jan Hejda, Ph.D. katedra zdravotnických oborů a ochrany obyvatelstva FBMI

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

Datum zadání bakalářské práce: **28.01.2022** Termín odevzdání bakalářské práce: **20.05.2022**

Platnost zadání bakalářské práce: **30.09.2023**

Ing. Jan Hejda, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____
Datum převzetí zadání

_____
Podpis studenta

Poděkování / Prohlášení

Chtěl bych poděkovat vedoucímu mé bakalářské práce Ing. Janu Hejdovi, Ph.D. za vytrvalé konzultování problematiky, za jeho cenné nápady a důležité připomínky, které vedly k výsledkům, o nichž se můžete v této práci dočíst.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 20. května 2022

.....

Abstrakt / Abstract

Obsahem této práce je analýza metod používaných pro rozpoznávání emocí z hlasu, metod pro extrakci příznaků ze hlasových nahrávek, analýza dostupných anotovaných datasetů pro trénování klasifikátoru a implementace systému pro rozpoznávání emocí z hlasu mluvčího. Cílem práce je navržení parametrů klasifikátoru a jeho otestování na produkčních datech.

V práci jsou popsány dvě architektury konvolučních neuronových sítí. Obě architektury jsou nezávislé, liší se už samotným vektorem, který do nich vstupuje. Vstupem do první sítě jsou vybrané vlastnosti (příznaky) nahrávek. Za použití této metody bylo při validaci dosaženo 88.2 % přesnosti, ale výsledky sítě velmi závisely na použitých příznacích, proto byla v práci implementována i druhá síť. Vstupem do druhé navržené architektury jsou celé spektrogramy, zde bylo dosaženo přesnosti 76.5 % při validaci.

Testování modelů neuronových sítí bylo provedeno na simulaci VHF/UHF vysílaček, kdy k nahrávkám se vzorkovací frekvencí 11 kHz byl připočten bílý šum. Obě architektury dosahují na testovacích datech přibližně stejné přesnosti okolo 50 %.

V případě reálného nasazení lze systém použít například pro zajištění bezpečnosti cestujících v letadle tak, že systém bude sledovat emoční rozpoložení pilota. Rozpoznávání emocí z hlasu nachází ale uplatnění i v mnoha jiných oborech.

Klíčová slova: rozpoznávání emocí; zpracování hlasu; klasifikace; rozpoznávání emocí z hlasu; neuronové sítě; konvoluční sítě.

The scope of this thesis is to analyze the methods used for speech emotion recognition, methods for extracting features from voice recordings, analysis of available annotated datasets for training a classifier and implementation of a system for speech emotion recognition. The aim of this work is to design the parameters of the classifier and test it on production data.

In this work, two architectures of convolutional neural networks are described. Both architectures are independent, differing in the vector that enters them. The input to the first network is selected features of the recordings. Using this method, the validation achieved 88.2 % accuracy, but the network results were highly dependent on the used features, so a second network was implemented in this work. The input to the second proposed architecture is whole spectrograms, here 76.5 % accuracy was achieved in validation.

The neural network models were tested on a simulation of VHF/UHF walkie talkies, where white noise was added to the recordings with a sampling rate of 11 kHz. Both architectures achieve approximately the same accuracy of around 50 % on the test data.

If the system is deployed, it can be used, for example, to ensure the safety of passengers on an aircraft by monitoring the emotional state of the pilot. However, voice emotion recognition also finds applications in many other fields.

Keywords: emotion recognition; speech analysis; classification; speech emotion recognition; neural networks; convolutional networks.

Title translation: Expert System for Determining the Dominant Expressed Emotion from the Voice

/ Obsah

1 Úvod **1**

2 Metody řešení rozpoznávání emocí z hlasu **2**

2.1 Metoda podpůrných vektorů . . . 3
2.2 Vícerozměrná lineární regrese . . 3
2.3 Neuronové sítě 4

3 Neuronové sítě **6**

3.1 Model neuronu 6
3.1.1 Aktivační funkce 9
3.1.2 Ztrátová funkce 12
3.1.3 Optimalizátor 13
3.2 Vrstvy sítí 15
3.2.1 Fully connected 15
3.2.2 Konvoluční vrstva 15
3.2.3 MaxPool 16
3.2.4 Dropout 17
3.2.5 BatchNorm 17
3.3 Problémy 17
3.4 Augmentace 18

4 Extrakce příznaků ze zvuku **20**

4.1 Zero Crossing Rate (ZCR) . . . 20
4.2 Root Mean Square (RMS) . . . 21
4.3 Spektrogram a Mel-spektrogram 21
4.4 Chromagram 24
4.5 Mel-frekvenční kepstrální koeficienty (MFCC) 24
4.6 Průměr a standardní odchylka základní frekvence . . . 25

5 Datasety pro rozpoznávání emocí z hlasu **27**

5.1 EmoDB 27
5.2 RAVDESS 27
5.3 CREMA-D 28
5.4 Zpracování nahrávek 28

6 Síť s kombinací několika druhů příznaků **30**

7 Síť trénovaná na spektrogramu **38**

8 Diskuze **43**

9 Závěr **45**

Literatura **46**

Tabulky / Obrázky

2.1 Přehled přesností klasifikace a jednotlivých metod 2
5.1 Překlady emocí 27
6.1 Přesnost v závislosti na délce střihu 31
6.2 Přesnost v závislosti na příznacích 32
8.1 Dosažené výsledky 43

2.1 Úvod do SVM 3
2.2 Algoritmus lineární regrese 4
2.3 Schéma rekurentní sítě 4
2.4 Schéma dopředné sítě 4
3.1 Model neuronové sítě 6
3.2 Nervová buňka - neuron 7
3.3 Matematický model neuronu 7
3.4 Schéma dopředné a zpětné propagace a modifikace vah 8
3.5 Graf lineární funkce 9
3.6 Graf sigmoidy 10
3.7 Graf tanh(x) 11
3.8 Graf ReLU 11
3.9 Graf Leaky ReLU 12
3.10 Graf ztrátové funkce v závislosti na nastavení váhy 13
3.11 Malý learning rate 14
3.12 Velký learning rate 14
3.13 Porovnání SGD a Mini-Batch . 14
3.14 Princip konvoluční vrstvy 16
3.15 Princip MaxPoolingové vrstvy 16
3.16 Underfitting a overfitting 18
4.1 Zero Crossing Rate (ZCR) 20
4.2 Root Mean Square (RMS) 21
4.3 Spektrogram 22
4.4 Melová stupnice 22
4.5 Mel spektrogram 23
4.6 Chromagram 24
4.7 Mel-frekvenční kepstrální koeficienty (MFCC) 25
4.8 Základní frekvence 26
5.1 Rozložení datasetů 28
5.2 Počty a délky nahrávek 29
6.1 Tvorba vstupního vektoru neuronové sítě 30
6.2 Spektrogram hněvu 32
6.3 Spektrogram smutku 33
6.4 Architektura 1 34
6.5 Graf trénování sítě 1 36
6.6 Confusion matrix 1 36
6.7 Graf trénování sítě 1 - testování 37
6.8 Confusion matrix 1 - testování . 37
7.1 Architektura 2 39
7.2 Graf trénování sítě 2 41

7.3 Confusion matrix 2 41
7.4 Graf trénování sítě 2 - testování 42
7.5 Confusion matrix 2 - testování . 42

Kapitola 1
Úvod

S emocemi se setkáváme všichni každý den, společně je prožíváme a vnímáme projevy emocí druhých osob, které potkáváme. Vnímané emoce z našeho okolí se mohou šířit zpět na nás. Emoce mění naše pocity a ovlivňují naše chování, jsou jakousi bezprostřední reakcí na vnější podněty.

Člověk dokáže poměrně snadno odhadnout a popsat, jaké emoce prožívá aktér, kterého jedinec pozoruje nebo poslouchá. Rozpoznávání lidských emocí je však pro počítače poměrně náročný problém. Náročnost je způsobena tím, že žádná měřitelná veličina není přímo závislá na projevované emoci. Emoce tedy nelze měřit žádným senzorem. Nelze ani nastavit žádná jednoduchá pravidla, podle kterých by byl stroj schopný se rozhodovat a rozpoznávat lidské emoce.

Projevovaná emoce se výrazně promítne v srdeční frekvenci jedince, v jeho krevním tlaku nebo v mozkových vlnách. Abychom získali tato data, potřebujeme zdravotnické zařízení a vyškolený personál, což není praktické pro každodenní použití. Proto se s rozvojem umělé inteligence v posledních několika letech rozvinuly i výzkumy, které se zabývají rozpoznáváním emocí z audiovizuálního materiálu, který můžeme pořídit běžnou kamerou nebo mikrofonem. Výkonné počítače umožňují natrénovat rozsáhlé neuronové sítě, které umí obdobné problémy efektivně řešit.

Počítačové rozpoznávání lidských emocí hraje důležitou roli v oblastech, kterými jsou například: doprava (rozpoznávání stavu řidiče nebo pilota), bezpečnost (sledování emocí ve veřejných prostorech), zákaznická zpětná vazba (automatické vyhodnocení), počítačové hry (adaptace hry v závislosti na emocích hráče), zdravotnictví (vyhodnocování stresové zátěže), a mnoha dalších.

Cílem této práce je navrhnout systém, který dokáže co nejlépe zařadit zvukovou nahrávku mluvčího do jedné z šesti sledovaných emocí. K tomu je nutné nejdříve analyzovat dostupné datasety zvukových nahrávek, které mají přiřazenou emoci, která je v nich zachycena. Dále se práce zabývá analýzou a návrhem metody pro analýzu hlasu. Testování systému je provedeno na větě jiné, než která byla použita při tvorbě tohoto systému. Jelikož interkomy nebo vysílačky přenáší zvuk v horší kvalitě, než v jaké jsou schopné ho zaznamenat, bude přesnost systému otestována i při simulaci těchto zhoršených podmínek. Posledním úkolem této práce je vyvinutí aplikace, která rozpoznává emoce v reálném čase ze signálu získaného z mikrofonu počítače.

Kapitola 2
Metody řešení rozpoznávání emocí z hlasu

Emoce, které jsou v této práci použity můžeme označit za základní emoce. Paul Ekman s W. V. Friesenem definovali základní emoce jako univerzální emoce se shodným projevem napříč kulturami. Podle nich mezi základní emoce patří šest emocí: štěstí, smutek, překvapení, strach, vztek a znechucení. Ekman s Friesenem předkládali fotografie obličejů (které zachycovaly ony definované emoce) domorodcům z kmene Fore na Nové Guinei. Domorodci dokázali přiřadit fotografie přesně tak, jak je vnímá západní společnost. Z podstaty experimentu se lze také domnívat, že výraz v obličeji jedince dokáže odhalit, jakou emoci jedinec vyjadřuje. Stejně tak vrozená reakce na vztek (zvýšení tlaku, zrychlení tepu) je přirozená pro všechny lidi. Základní emoce by se tedy měly projevit i v hlase a tyto projevy by měly být navzájem nezávislé, odlišitelné. [1–2]

Přiřazení zvukových nahrávek k jednotlivým emocím se nazývá klasifikace. Klasifikace je obecně problém, který řeší algoritmus nazývaný klasifikátor. Máme-li nějaký objekt, který spadá do jedné či více tříd (kategorií, labelů), pak je úkolem klasifikátoru najít přiřazení onoho objektu ke správné třídě nebo ke správným třídám. V našem případě je tento neznámý objekt zvuková nahrávka hlasu mluvčího a kategorie jsou jednotlivé základní emoce, ke kterým bude nahrávka přiřazena. Naším cílem je přiřadit jednu nahrávku vždy k jedné z emocí, proto problém můžeme konkrétněji označit - jedná se o multi-class klasifikaci (přiřazení k jedné z více tříd).

Protože nelze obecně a jednoduše popsat pravidla, která určují výskyt dané emoce v hlasové nahrávce, nemůžeme ani vytvořit program, který by podle těchto pravidel klasifikoval. Nicméně můžeme klasifikaci řešit metodami strojového učení (jedna z metod umělé inteligence), které dokáže efektivně tato pravidla nalézt při dostatku trénovacích dat, ze kterých se bude stroj učit nám neznámé souvislosti. Strojové učení je takto schopné vytvořit model, který má naučená pravidla pro klasifikaci. Podle tohoto natrénovaného modelu jsme pak schopni klasifikovat nahrávky.

Pro rozpoznávání emocí z hlasu se nejčastěji používá metoda podpůrných vektorů (Support Vector Machines), lineární regrese (Multivariable Linear Regression) nebo neuronové sítě. [3–6] Tabulka 2.1 zobrazuje maximální průměrnou dosaženou přesnost v závislosti na různých klasifikátorech.

Metody	Průměrná přesnost [%]
Metoda podpůrných vektorů	76.48
Vícerozměrná lineární regrese	68.90
Rekurentní neuronová síť	79.20
Konvoluční neuronová síť	83.61

Tabulka 2.1. Přehled přesností klasifikace a jednotlivých metod podle [3].

Je však nutné dodat, že dosažená přesnost klasifikátoru závisí nejenom na vybrané metodě, ale i na výběru datasetu (nebo jejich kombinací), množství tříd, do kterých klasifikujeme nebo na zpracování dat před klasifikací.

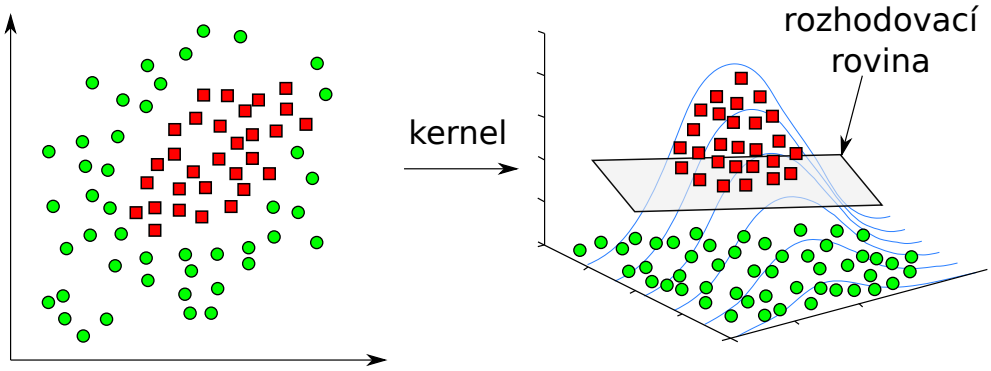
2.1 Metoda podpůrných vektorů

Metoda je také známá pod anglickým názvem Support Vector Machines či pod zkratkou SVM. Metoda se používá pro klasifikaci nebo pro regresní úlohy.

Hlavní myšlenkou algoritmu je transformace původních vstupů (vstupních příznaků) datasetu do vysokodimenzionálního prostoru za použití jádrové funkce (kernel function), a poté dosažení optimální klasifikace v tomto novém prostoru. Díky této transformaci lze klasifikovat i lineárně neseparovatelná vstupní data.

Klasifikace dat probíhá hledáním optimálních n-dimensionálních hyperrovin, které optimálně rozdělují data na požadované kategorie (třídy). K hledání optimálních hyperrovin se využívá tzv. podpůrných vektorů, což jsou body v prostoru, které reprezentují trénovací data, která jsou nejblíže k hyperrovině. Optimální hyperrovina je pak taková hyperrovina, která maximalizuje svou vzdálenost od těchto podpůrných vektorů. [7]

Obrázek 2.1 znázorňuje, jak funguje SVM. Na levé straně máme vstupní data ve 2D prostoru, která chceme automaticky klasifikovat do zelené a červené třídy. Použitím jádrové funkce tato data transformujeme do prostoru vyšší dimenze. Poté nalezneme rovinu, která je zakreslena na obrázku. Toto je optimální rovina, která rozdělí transformovaný prostor vstupních dat, pomocí níž můžeme data klasifikovat. V původním 2D prostoru nelze najít takovou přímku, která by optimálně rozdělila původní prostor. Tudíž je použití jádrové funkce klíčové pro řešitelnost znázorněné úlohy.



Obrázek 2.1. Transformace dat do vyšší dimenze a rozdělení nadrovinou v SVM [8]

Článek [3] uvádí přesnost za použití SVM až 76,5 % na datasetu SAVEE [9]. V článku [5] se uvádí přesnost za použití SVM až 63,3 % na datasetu EmoDB [10] v německém jazyce a až 77,6 % na autorském datasetu ve španělském jazyce. Článek [6] uvádí průměr přesnosti 68,3 % z devíti testovaných datasetů.

2.2 Vícerozměrná lineární regrese

Lineární regrese se v literatuře často vyskytuje pod zkratkami LRC (Linear Regression Classification) nebo MVR/MLR (Multivariate Linear Regression). Klasifikace se provádí podle níže popsaného algoritmu na obrázku 2.2, který byl v článku [11] použitý pro klasifikaci emocí z obrázků, ale lze ho využít i pro klasifikaci emocí z hlasu.

Nejprve potřebujeme vygenerovat matici X_i pro každou z i-tříd. Tato matice obsahuje ve sloupcích vektory trénovacích dat, které jsou lineárně nezávislé. Poté podle bodu 2. z popisu algoritmu vypočítáme vektor parametrů $\hat{\beta}_i$ pro každou ze tříd. Dále spočítáme lineární kombinaci vektorů trénovacích dat s vektorem parametrů, tím získáme predikce

$\hat{y}_i$ pro každou třídu. Nakonec vypočítáme vzdálenosti neznámého obrázku (vektoru y_i) od predikce $\hat{y}_i$. Neznámý obrázek poté klasifikujeme do třídy, jejíž norma je nejmenší.

Algorithm: Linear Regression Classification (LRC)
Inputs: Class models $\mathbf{X}_i \in \mathbb{R}^{q \times p_i}, i = 1, 2, \ldots, N$ and a test image vector $\mathbf{y} \in \mathbb{R}^{q \times 1}$.
Output: Class of $\mathbf{y}$

1. $\hat{\beta}_i \in \mathbb{R}^{p_i \times 1}$ is evaluated against each class model, $\hat{\beta}_i = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{y}, \quad i = 1, 2, \ldots, N$
2. $\hat{\mathbf{y}}_i$ is computed for each $\hat{\beta}_i$, $\hat{\mathbf{y}}_i = \mathbf{X}_i \hat{\beta}_i, \quad i = 1, 2, \ldots, N$
3. Distance calculation between original and predicted response variables $d_i(\mathbf{y}) = \|\mathbf{y} - \hat{\mathbf{y}}_i\|_2, \quad i = 1, 2, \ldots, N$
4. Decision is made in favor of the class with the minimum distance $d_i(\mathbf{y})$

Obrázek 2.2. Algoritmus lineární regrese [11]

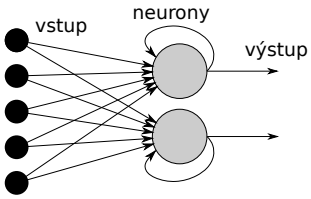
Článek [3] uvádí přesnost metody klasifikace pomocí lineární regrese v průměru až 68,9 % na datasetu SAVEE. V článku [5] se uvádí přesnost za použití lineární regrese v průměru až 75,9 % na datasetu EmoDB a až 82,4 % na autorském datasetu ve španělském jazyce.

2.3 Neuronové sítě

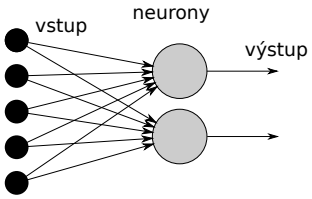
Neuronové sítě jsou prostředkem pro realizaci hlubokého učení, což je podskupina strojového učení. Hluboké učení se zaměřuje na zpracování reprezentace vstupních dat ve vrstvách, kdy každá z vrstev transformuje data na smysluplnější reprezentaci.

Neuronová síť se skládá z vrstev neuronů, kdy každý z neuronů nějakým způsobem zpracovává informace na svém vstupu (skrze natrénované parametry), toto zpracování poté předá na svůj výstup, kde si informaci přeberou neurony z další vrstvy. Aktivace určité skupiny neuronů v síti může zapříčinit aktivaci jiných neuronů v hlubší vrstvě, což může připomínat chování biologických neuronů. Přestože některé původní myšlenky neuronových sítí se inspirovaly v savčím mozku, neexistuje důkaz, že lidský mozek implementuje jakýkoliv podobný algoritmus, jaký využívá strojové učení. Neuronové sítě se nemodelují podle lidského mozku. [12]

Neuronové sítě můžeme rozdělit na několik kategorií podle toho, jak jsou neurony v síti vzájemně propojeny, případně jak samotné neurony ve vrstvách fungují. Pro úlohu rozpoznávání emocí z hlasu se používají dva druhy sítí: rekurentní nebo dopředné (konvoluční), případně kombinace obou dvou.



Obrázek 2.3. Schéma rekurentní sítě



Obrázek 2.4. Schéma dopředné sítě

Rekurentní sítě mají cyklické propojení neuronů v rámci jedné vrstvy, což je znázorněno na obrázku 2.3. Tyto sítě se hodí pro zpracování sekvenčních dat s různou délkou, jako je text nebo zvuk. Long-Short Term Memory (LSTM) síť je speciální druh

rekurentní vrstvy, která dokáže vyhledat a zpracovat závislosti v datech, které se v nich vyskytují dlouhodobě v čase. LSTM sítě stojí za velkým úspěchem v oblasti automatického překladu textu nebo v oblasti převodu řeči na text. [13]

Konvoluční neuronové sítě jsou dopředné sítě (obrázek 2.4). Oproti rekurentním dokáží naopak zpracovat lokální kontext, který se vyskytuje ve vstupních datech jednoho vzorku, jako jsou například vzory v obrázku. Síť pracuje pouze se statickými vstupními daty, nedokáže zpracovat sekvence - stará data, která prošla sítí neovlivní učení sítě z dat, která přišla nově na její vstup. Konvoluční sítě dosáhly velkého úspěchu v oblasti strojového vidění, jako je detekce objektů nebo jejich klasifikace v obrázcích. [13–14]

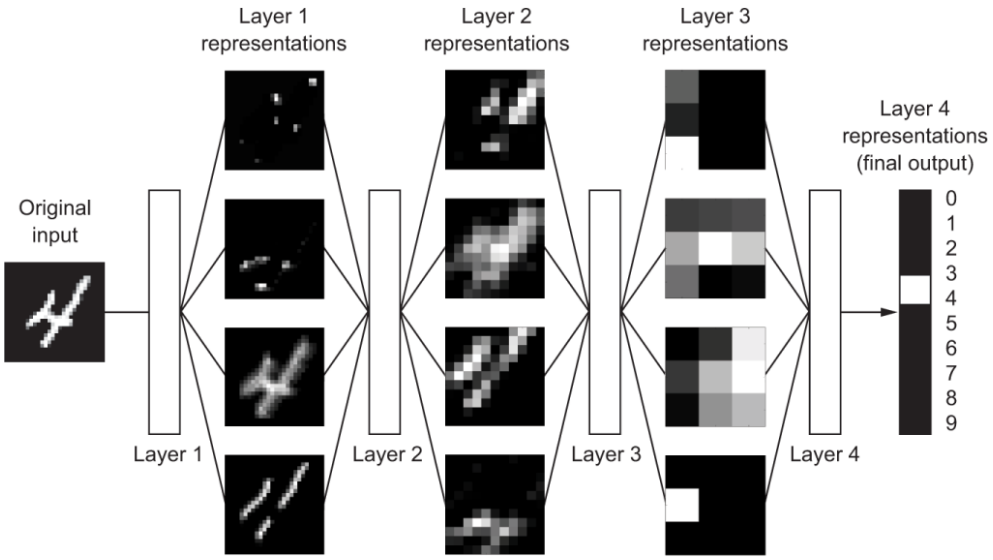
Autoři článku [3] provedli experimenty s datasetem SAVEE i na neuronových sítích. Při testování rekurentní neuronové sítě dosáhli přesnosti 79,2 %. Při testování konvoluční neuronové sítě však dosáhli přesnosti v průměru až 83,6 %. V článku [5] uvádí autoři při použití rekurentní neuronové sítě přesnost až 69,6 % na datasetu EmoDB [10] a až 90 % na autorském datasetu ve španělském jazyce.

Z uvedených článků, které srovnávají jednotlivé metody klasifikace tedy vyplývá, že nejlepších výsledků dosahují právě neuronové sítě. Rozdíl v přesnosti dosažené použitím rekurentní sítě a použitím konvoluční sítě je minimální. Ačkoliv se v poslední době zdá, že konvoluční sítě dosahují mírně lepších výsledků. Proto jsem i já v dalším postupu zvolil metodu konvolučních neuronových sítí, které detailně popíšu v následující kapitole.

Kapitola 3
Neuronové sítě

Neuronová síť je model složený z několika vzájemně kompatibilních vrstev, které jsou propojené za sebou. Tyto vrstvy můžeme skládat za sebe obdobně, jako můžeme spojovat kostky LEGA. Každá z těchto vrstev transformuje data na smysluplnější reprezentaci dat. Čím hlouběji pozorujeme data v síti, tím více se liší od vstupních dat, avšak tím více informují o výsledku, resp. řešení dané úlohy.

Na obrázku 3.1 můžeme vidět model neuronové sítě, která se skládá ze 4 vrstev. Na vstupu modelu je obrázek, na kterém je napsaná číslice 4. První vrstva vstupní obrázek zpracuje. Data, která z vrstvy vystupují jsou jistým způsobem ochuzena o informace, které nejsou důležité pro konečné rozpoznání číslice, která je na vstupním obrázku. Konkrétně na reprezentaci dat, která vystupují z první vrstvy vidíme, že se zdůraznily například hrany číslice 4. Tato data projdou do následujících 3 vrstev. Reprezentace poslední čtvrté vrstvy už je požadovaný výsledek. Poslední vrstva má na výstupu 10 proměnných, jejichž aktivace značí číslici, která byla na vstupu. Na modelu je znázorněna aktivace u 4, což značí, že síť správně rozpoznala číslici na vstupním obrázku.

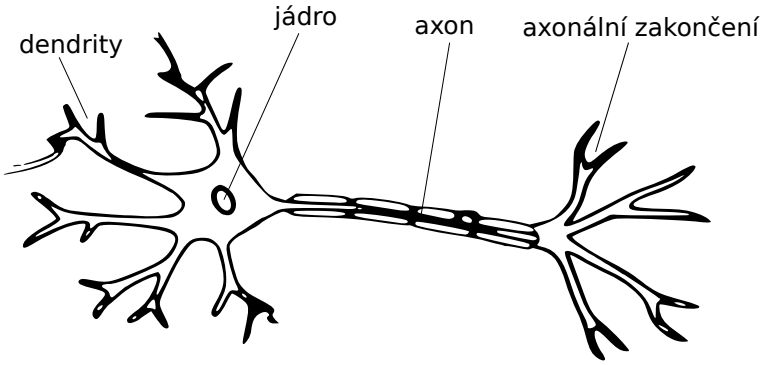


Obrázek 3.1. Model neuronové sítě zobrazující číslovku 4 na vstupu, jednotlivé vrstvy, reprezentace dat v jednotlivých vrstvách a výstup poslední vrstvy (výsledek) [12]

3.1 Model neuronu

Každá vrstva sítě je tvořena neurony, což jsou jednoduché základní buňky neuronových sítí, které zpracovávají dílčí informace. Síla neuronových sítí spočívá právě ve spojení těchto neuronů. Mnohé používané principy v sítích lze ale vysvětlit pouze na modelu neuronu.

Motivací pro vytvoření matematického modelu neuronu byla nervová buňka - biologický neuron. Tyto neurony tvoří nervovou soustavu např. savců. Nákres, jak biologický neuron vypadá, je na obrázku 3.2. Skládá se ze třech hlavních částí - dendritů, těla a axonu.

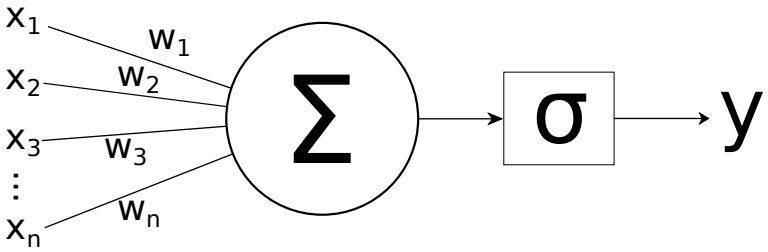


Obrázek 3.2. Nákres buňky neuronu

Pro vytvoření matematického modelu neuronu jsou důležité následující pozorování. Dedrity jsou spojeny tzv. synapsemi s ostatními neurony. Skrze synapse se šíří signál a dendrity je sváděn do těla neuronu. V těle neuronu se nachází jádro, které zpracuje a vyhodnotí příchozí informace. Vyhodnocení odpovídá jisté aktivaci neuronu. Tato aktivace se pak šíří skrze axon do dalších neuronů.

Při pohledu na jeden neuron lze tedy považovat dendrity za vstupy. Jádro za matematickou funkci. Axonem se šíří výsledek matematické funkce k ostatním neuronům, které jsou napojeny na axonální zakončení.

Na obrázku 3.3 se nachází schéma matematického modelu neuronu. Ten se skládá z vektoru vstupů (nebo také vstupních příznaků) $\mathbf{x}$. Každý z těchto vstupů je vynásoben příslušnou váhou $\mathbf{w}$. Váhy přisuzují vstupům různou důležitost. Pokud nějaký ze vstupů není důležitý pro rozhodování neuronu, pak je váha tohoto vstupu oproti ostatním nízká. Vážené vstupy jsou pak sečteny. Výsledek sumy pak vstupuje do aktivační funkce σ. Hodnoty aktivační funkce odpovídají aktivaci daného neuronu, což je i jeho výstup y. Ten pak může být předán do další vrstvy neuronové sítě.



Obrázek 3.3. Matematický model neuronu

Parametrem, kterým neuronovou síť učíme, jsou právě zmíněné váhy. Naším cílem při trénování sítě je najít takovou kombinaci všech vah, která nám dá co nejlepší výsledek - predikci. Abychom mohli měřit kvalitu výsledku, který je na výstupu sítě, tak musíme zavést tzv. ztrátovou funkci (loss function). Vstupem ztrátové funkce je predikce neuronové sítě a požadovaný výsledek, který po síti chceme. Výstupem ztrátové funkce je pak skóre, které reprezentuje, jak moc je predikce vzdálena od správné třídy vstupních dat. Ztrátové skóre si také můžeme představit jako jistou penalizaci neuronové sítě za její (špatnou) predikci. Pokud změníme váhy, uděláme novou predikci a spočítáme ztrátu

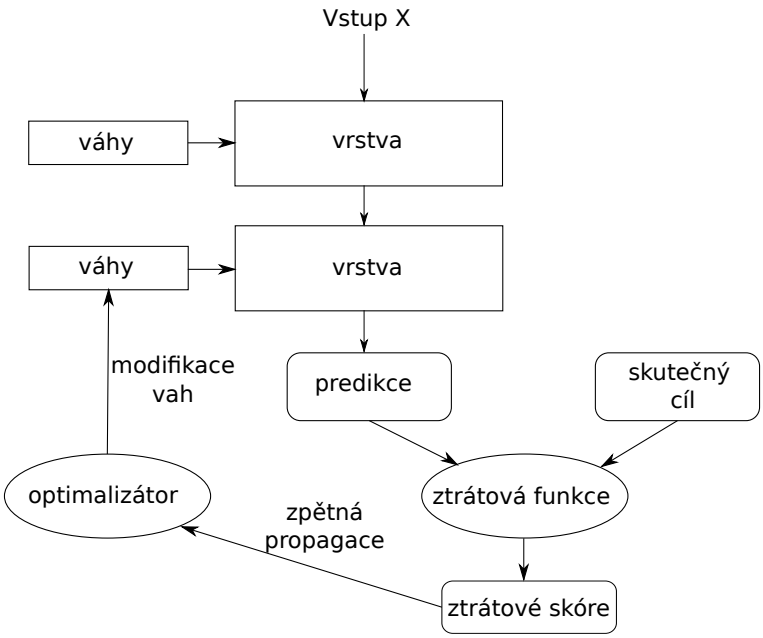
sítě, můžeme rozhodnout o tom, zda změna vah byla prospěšná. Je-li ztráta sítě větší než před změnou vah, tak výsledky sítě jsou horší, a proto je horší i aktuální nastavení vah.

Jakmile máme na vstupu sítě nějaká data, tak je zpracuje první vrstva sítě. Ta předá výsledky další vrstvě, atd. Na konci sítě máme vytvořenou predikci, ze které spočítáme ztrátu sítě. Při tomto procesu prochází vstupní data celou sítí a nazýváme ho dopředná propagace. Při dopředné propagaci si ukládáme výpočetní graf matematických operací nad daty prostupujícími sítí.

Protože ztrátová funkce nabývá pouze nezáporných hodnot, nulová ztráta znamená nejlepší natrénování sítě. Proto lze na trénování sítě pohlížet jako na optimalizační problém, kdy hledáme minimum ztrátové funkce. Abychom byli schopni změnit váhy tak, abychom dosáhli menší ztráty, musíme spočítat gradient ztrátové funkce (směr největšího růstu). Procesu výpočtu gradientu se říká zpětná propagace a spočívá v tom, že jdeme v opačném směru výpočetního grafu - ve směru od ztrátové funkce k neuronům a jejich vahám. Při postupu výpočetním grafem spočítáme derivaci v každém uzlu výpočetního grafu. Tím získáme gradient ztrátové funkce v závislosti na všech vahách v síti.

Jelikož neuronové sítě mohou mít miliony parametrů, tak najít globální minimum ztrátové sítě je prakticky nemožné. Proto se na začátku trénování neuronové sítě inicializují váhy náhodně. Následně se provede dopředná a zpětná propagace. Poté optimalizátor (algoritmus měnící váhy) změní příslušně váhy tak, abychom je posunuli v opačném směru gradientu. Tímto krokem se přiblížíme k nějakému lokálnímu optimu v prostoru vah. Poté provedeme další iteraci učení - načtení dalších dat, dopředná propagace, atd.

Na obrázku 3.4 je schéma výše popsaných procesů. Do neuronové sítě o dvou vrstvách vstupují data, která jimi projdou. Výstupem z poslední vrstvy je nějaká predikce. Tato predikce vstoupí do ztrátové funkce společně se skutečným cílem. Ztrátová funkce poté vrátí ztrátové skóre. Zpětnou propagací se spočítá gradient a optimalizátor nakonec rozhodne, jak změnit váhy.



Obrázek 3.4. Schéma dopředné a zpětné propagace a modifikace vah [12]

Pro trénování s učitelem tedy potřebujeme anotovaná data - v našem případě zvukové nahrávky s anotací, o jakou emoci se jedná. Platí zásada, že se přesnost natrénované sítě neověřuje na trénovacích datech. Síť je schopná si memorovat trénovací data, proto kdyby došlo k testování sítě na trénovacích datech, přesnost sítě by byla znatelně lepší, než přesnost na datech, ze kterých se neučila. Proto celou množinu anotovaných dat rozdělíme na trénovací a validační dataset. Na základě trénovacího datasetu síť upravuje své váhy a pomocí validačního datasetu můžeme otestovat přesnost sítě. Obecně lze data rozdělit v poměru 80:20 ve prospěch trénovacího datasetu.

Protože ale i trénovací dataset obsahuje obrovské množství dat, nelze jej celý nahrát do operační paměti. Proto se síť trénuje po malých dávkách (batch). Každý batch dat pak obsahuje několik desítek až stovek jednotlivých vzorků (nahrávek) v závislosti na velikosti jednotlivého vzorku a velikosti operační paměti, kterou máme k dispozici. Trénovací batche získáme tak, že náhodně zamícháme trénovací dataset a rozdělíme jej na jednotlivé batche. Trénování sítě probíhá po jednotlivých epochách. Jedna epocha odpovídá modifikaci vah pro každou z dostupných dávek. Epocha odpovídá jednomu zpracování všech dostupných trénovacích dat.

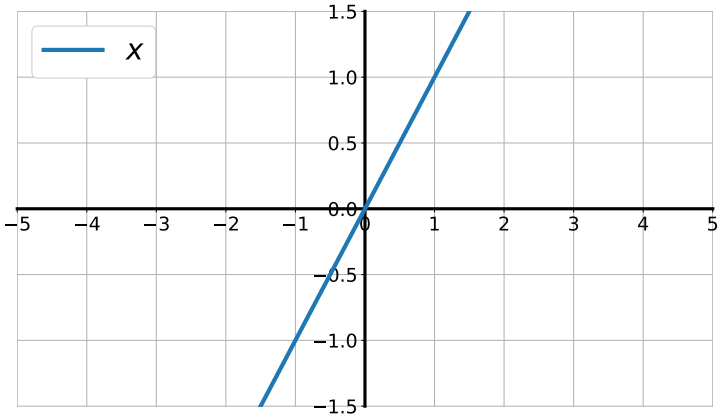
3.1.1 Aktivační funkce

Aktivační funkce jsou v matematickém modelu neuronu důležité, protože vnáší do výpočetního grafu nelinearitu. Kdybychom měli výpočetní graf skládající se pouze z lineárních operací (lineárních kombinací vektoru vah s vektorem vstupů), pak by se síť mohla naučit pouze lineární transformace vstupních dat. Prostor těchto transformací je však příliš malý proti prostoru všech možných transformací vstupních dat. Proto zavádíme aktivační funkci, která je nelineární.

Aktivační funkce mají různé vlastnosti, některé se používají ve skrytých (vnitřních) vrstvách sítě, jiné se používají ve výstupní vrstvě. Nejčastěji používané aktivační funkce jsou popsány v následujícím seznamu:

- **Lineární funkce**

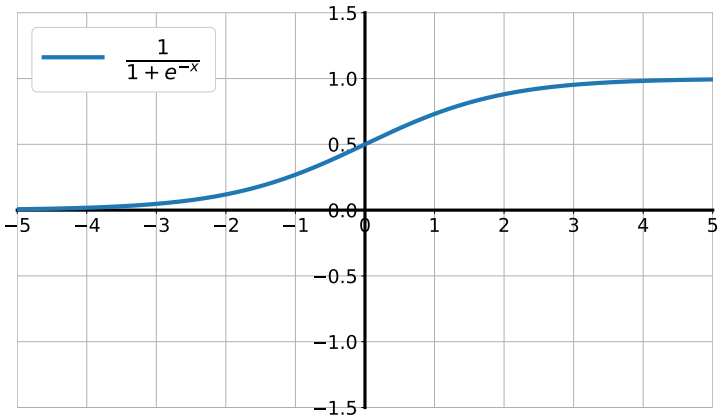
  Lineární aktivační funkce se používá pouze ve výstupní vrstvě. Typicky při řešení regresních úloh, kdy výstupem je nějaké požadované číslo - například cena nemovitosti. Její nevýhoda je, že derivace lineární funkce je konstanta. Vrstva s lineární aktivační funkcí tedy nezmění gradient ztrátové funkce, proto nemůže jakkoliv zlepšit vlastnosti naší neuronové sítě.



Obrázek 3.5. Graf lineární funkce

Sigmoida

Sigmoida se jako aktivační funkce používá běžně u neuronů ve výstupní vrstvě, když řešíme binární klasifikaci (přiřazení k jedné ze dvou tříd). Hodnoty sigmoidy se pohybují mezi 0 a 1, což reprezentuje pravděpodobnost, že vstupní data patří do dané třídy. Při použití sigmoidy může vzniknout problém mizení gradientu. Jestliže se dostaneme do krajní oblasti sigmoidy, pak se okolí hodnot sigmoidy blíží k hodnotám konstatní funkce a její derivace se blíží k nule. Při zpětné propagaci pak tato téměř nulová derivace způsobí, že se i gradient ztrátové funkce bude blížit k nule. Síť se proto přestává učit.



Obrázek 3.6. Graf sigmoidy

Softmax

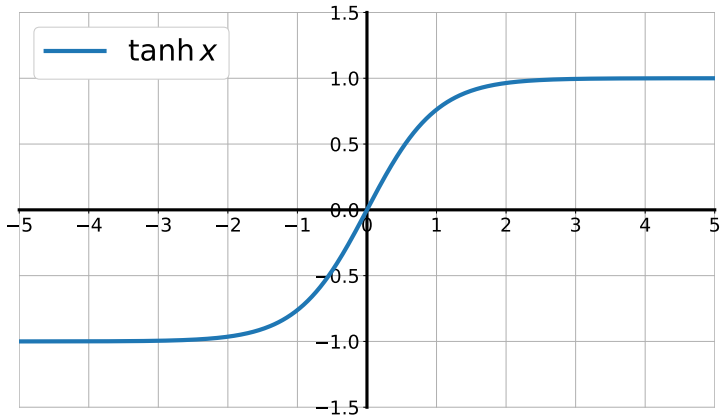
Softmax je zobecněná sigmoida do více dimenzí. Používá se v poslední vrstvě při klasifikaci do jedné z více než dvou tříd. V poslední vrstvě je právě tolik neuronů, kolik máme tříd. Výstup každého neuronu z poslední vrstvy reprezentuje pravděpodobnost, že vstupní data patří právě k dané třídě. Součet aktivací všech neuronů výstupní vrstvy se softmaxem je 1. Je jistota, že vstupní data patří do jedné ze tříd.

Softmax pro i-tou třídu z N vypočítáme podle následujícího vztahu.

$$\frac{e^{x_i}}{\sum_{j=0}^{N} x_j}$$

Tanh

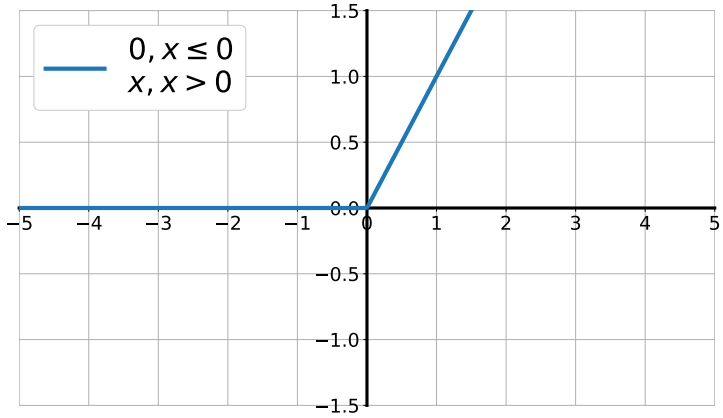
Tangens hyperbolický se používá obdobně jako sigmoida. Výhodou této aktivační funkce je, že vstupní data v okolí 0 zhruba zachová. Naopak více vychýlená vstupní data namapuje do okolí -1 nebo 1.



Obrázek 3.7. Graf funkce tanh(x)

ReLU

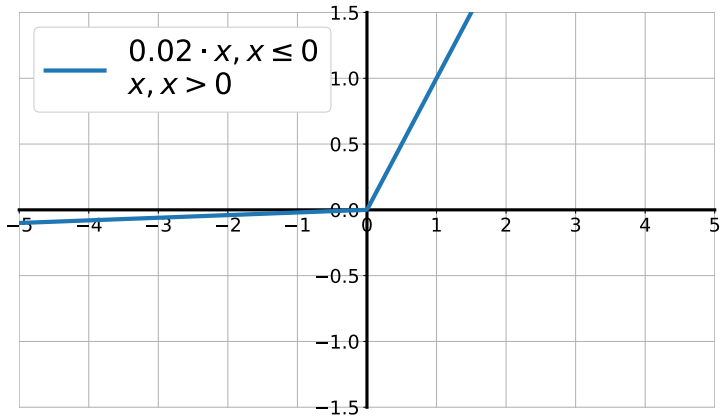
ReLU z anglického Rectified Linear Unit je často používaná aktivační funkce ve skrytých vrstvách neuronové sítě. Výhoda oproti tangentě je výpočetní nenáročnost funkce. Pro kladné vstupní hodnoty je funkce lineární a s trénováním není problém. Ale pokud do aktivační funkce vstoupí nekladná hodnota, pak je automaticky výstup neuronu nulový. Takovýto neuron se v síti chová jako mrtvý a již ho nelze zotavit, jelikož derivace v nekladné části ReLU je nulová.



Obrázek 3.8. Graf ReLU

Leaky ReLU

Leaky ReLU je obdoba ReLU, ale hodnoty v nekladné části jsou vynásobeny malým koeficientem. Díky tomu má funkce podobné vlastnosti jako ReLU, ale navíc je zde vyřešen problém s vymíráním neuronů.



Obrázek 3.9. Graf Leaky ReLU

3.1.2 Ztrátová funkce

Ztrátová funkce se používá při trénování sítě k vyhodnocení, jak moc se predikce sítě liší od skutečné třídy, do které trénovací data patří. V této sekci popíšu několik používaných ztrátových funkcí.

- **Mean Absolute Error (MAE)**

  MAE je střední absolutní chyba (L1 ztráta). Průměrná absolutní odchylka se používá pro regresní úlohy. MAE pro klasifikaci do N tříd je definována vztahem:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

  Kde y_i je anotovaná hodnota vstupu a $\hat{y}_i$ je predikce.

- **Mean Squared Error (MSE)**

  Je také známá pod názvem střední kvadratická chyba (L2 ztráta). Využívá se stejně jako MAE, ale oproti ní je velmi citlivá na outliery, díky čtverci rozdílu požadované třídy od predikce. Zároveň je MSE méně výpočetně náročná, než MAE. MSE je definována vztahem:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

  Kde y_i je anotovaná hodnota vstupu a $\hat{y}_i$ je predikce.

- **Cross-Entropy**

  Entropie je fyzikální veličina popisující chaotičnost nebo uspořádanost systému. Cross-entropy (křížová entropie) popisuje rozdíl mezi dvěma pravděpodobnostními rozděleními. Minimalizace křížové entropie tedy odpovídá vzájemnému přiblížení pravděpodobnostních rozdělení. Křížová entropie je definována vztahem:

$$\mathrm{E(p,\ q)} = -\sum p(x)\log q(x)$$

  Kde p a q jsou pravděpodobnostní rozdělení.

  V neuronových sítích se využívají dva druhy ztrátových funkcí křížové entropie. Prvním z nich je binární křížová entropie, která se uplatňuje při klasifikaci do dvou tříd a je kompatibilní pouze se sigmoidou jako výstupní aktivační funkcí. Druhým

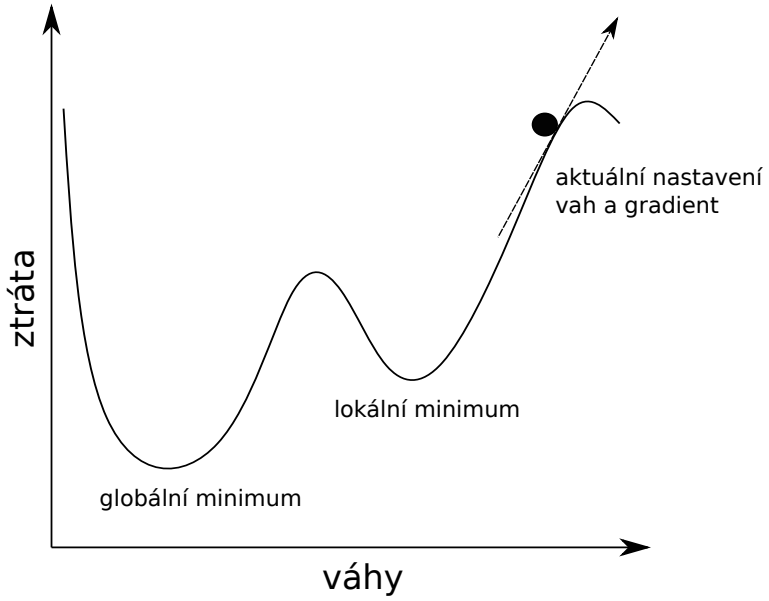
typem je kategorická křížová entropie, která je kompatibilní pouze se softmaxem. Kategorická křížová entropie se využívá při klasifikaci do jedné z více než dvou tříd. Kategorická křížová entropie se vypočítá podle následujícího vztahu:

$$\mathrm{L} = -\sum_{i=1}^{N} y_i \log \hat{y}_i$$

Při klasifikaci do N tříd je y_i buď 0, pokud vstupní data nepatří do i-té třídy. Nebo 1, pokud naopak data do dané třídy spadají, $\hat{y}_i$ je predikce sítě.

3.1.3 Optimalizátor

Jak jsem již zmínil, dimenze prostoru vah je příliš velká na to, abychom našli globální minimum nebo našli optima analyticky. Z tohoto důvodu se optima hledají iterativně aktualizací vah a ověřením prospěšnosti jejich aktualizace. K tomu nám slouží optimalizátor. Pro vytvoření jednoduché představy, jak se pohybujeme v prostoru vah slouží obrázek 3.10, který ukazuje graf ztrátové funkce v závislosti na nastavení jednodimenzionální váhy. Křivku grafu (hodnoty funkce) však v praxi neznáme a nelze ji spočítat. Známe pouze hodnotu funkce pro aktuální nastavení vah a gradient této funkce.



Obrázek 3.10. Graf ztrátové funkce v závislosti na nastavení váhy

Následující optimalizační algoritmy jsou srovnány v článku [15].

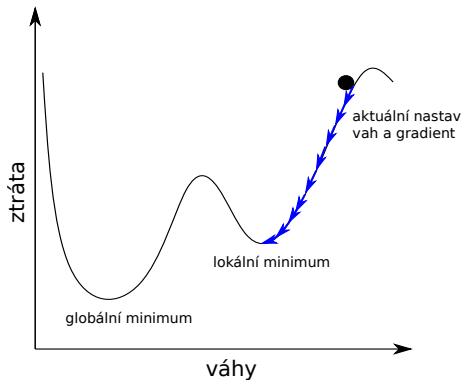
Gradient descent

Nejjednodušší optimalizační algoritmus se nazývá gradient descent. Algoritmus spočívá v posunu vah v opačném směru gradientu. Gradient descent má jediný parametr - learning rate. Learning rate škáluje velikost posunu vah vzhledem k velikosti gradientu. Jestliže aktuální nastavení vah je $\mathbf{w}_n$, poté aktualizujeme váhy:

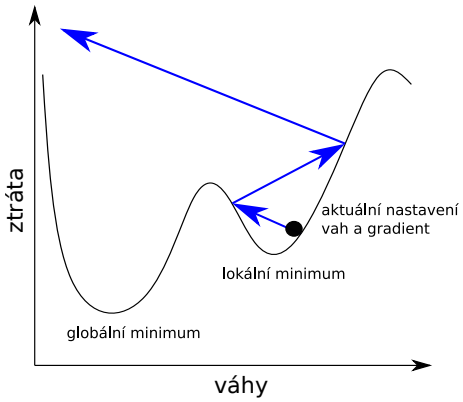
$$\mathbf{w}_{n+1} = \mathbf{w}_n - \alpha \nabla L(\mathbf{w}_n)$$

Kde α je learning rate a L je ztrátová funkce.

Nastavení learning rate je kruciální. Pokud bude learning rate příliš malý, bude konvergence k minimu velmi pomalá (obrázek 3.11). Naopak bude-li příliš velký,



Obrázek 3.11. Malý learning rate

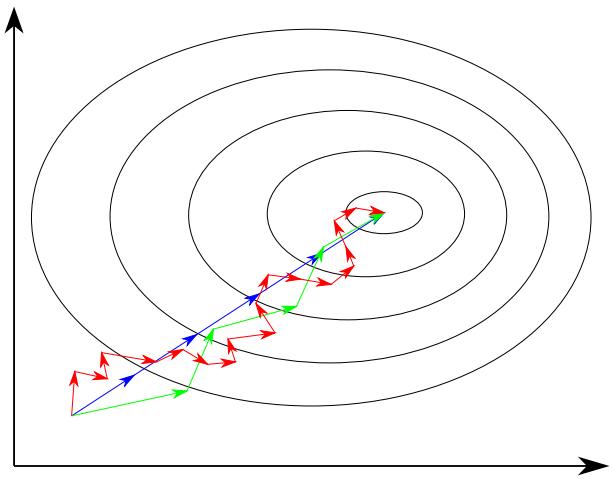


Obrázek 3.12. Velký learning rate

může nastat oscilace nad lokálním minimem nebo úplné přeskočení minima, do kterého jsme se blížili (obrázek 3.12). Kromě toho může nastat situace, že se váhy dostanou do místa, kde je gradient ztrátové funkce téměř nulový - plateau. Gradient descent se s tím neumí vypořádat a na těchto rovinách nastavení vah uvázne. Přesto se algoritmus často používá pro svoji jednoduchost a rychlost výpočtu.

Původní gradient descent počítá s tím, že ztrátovou funkci jsme vypočítali pro všechna trénovací data. Často to není možné kvůli velikosti trénovacích dat a omezenosti operační paměti. Proto vznikly dvě modifikace. První z nich je Stochastic Gradient Descent (SGD), který aktualizuje váhy při každém průchodu jednoho vzorku celou sítí. Díky tomu konvergence k optimu není přímočará ve vztahu k celému datasetu. Druhou modifikací je Mini-Batch Gradient Descent, což je vlastně kompromis mezi původním algoritmem a SGD. Aktualizace vah se provádí po průchodu určitého množství dat - dávky, batche.

Na obrázku 3.13 je znázorněný 2-D prostor vah s vrstevnicemi ztrátové funkce. Váhy máme nastavené v počátečením bodě šipek. Pokud použijeme gradient descent s aktualizací vah po průchodu všech dat, budeme k optimu konvergovat po modré přímce. Pokud použijeme SGD, bude konvergence pomalá a poměrně chaotická - červené šipky znázorňují iterace SGD. Zelené šipky odpovídají iteracím Mini-Batche. Typicky se snažíme zvolit co největší velikost batche, aby konvergence byla co nelepší.



Obrázek 3.13. Porovnání kroků optimalizátorů Gradient Descent (modře), SGD (červeně) a Mini-Batch (zeleně) při hledání optima

- **Momentum**

  Momentum je vylepšením gradient descentu o tzv. setrvačnost. Inspirací pro vznik algoritmu byla fyzikální setrvačnost.

  Na obrázku 3.10 jsou váhy znázorněny kruhem, nikoliv bodem na vodorovné ose. Mohli bychom si představit, že tento kruh je těleso, které se uvede do pohybu směrem k minimu funkce. Kdyby tento kruh měl nějakou hmotnost, tak by se v lokálním minimu díky své hybnosti nezastavil a přejel do globálního minima. Pokud by v našem světě působila i odporová síla, pak by se těleso mohlo ustálit v našem globálním minimu.

  Proto se v algoritmu Momentum vyskytuje druhý volitelný parametr (kromě learning rate), kterým je setrvačnost (momentum). Aktualizace vah je závislá nejenom na aktuálním gradientu ztrátové funkce, ale také na hybnosti, kterou váhy drží z předchozích iterací.

  Momentum dokáže efektivně překonávat plateau a mělká lokální optima. Konvergence je obecně rychlejší a s menšími oscilacemi než u Gradient Descentu.

- **Adaptivní algoritmy**

  Nevýhoda předchozích zmíněných optimalizátorů je, že musíme správně zvolit jejich parametry. Pokud jsou zvoleny špatně, nedojde k požadované konvergenci. I když parametry zvolíme vhodně, jsou po celou dobu optimalizace fixní, což není žádoucí. Pokud se blížíme k optimu, tak bychom intuitivně chtěli aktualizovat váhy s menším learning ratem. Proto byly vyvinuty adaptivní optimalizátory jako jsou AdaGrad, RMSprop nebo Adam (Adaptive Moment Estimation). V závislosti na změnách ztráty při aktualizaci vah samy mění své parametry. Úspěšnost a rychlost konvergence adaptivních optimalizátorů prakticky není závislá na počátečním nastavení jejich parametrů. Adaptivní optimalizátory ve většině případů předčí Gradient Descent nebo Momentum, jsou však výpočetně náročné.

3.2 Vrstvy sítí

3.2.1 Fully connected

Fully connected je vrstva, ve které má každý neuron tolik vstupů, kolik je výstupů předchozí vrstvy, přičemž na každý výstup předchozí vrstvy je napojený právě jeden vstup neuronu. Zjednodušeně lze říci, že každý neuron je spojen se všemi neurony předchozí vrstvy.

Fully connected vrstva se používá jako jedna z posledních vrstev. První důvod je, že oproti konvoluční vrstvě neumí efektivně vyhledávat příznaky ve vstupních datech. Druhý důvod je množství vah. Kdybychom měli vstupní tenzor dat o velikosti 150 a vstupní fully connected vrstva by obsahovala 1000 neuronů, pak jenom tato první vrstva v síti vytvoří 150000 vah. Tedy 150000 učitelných parametrů, což je příliš mnoho.

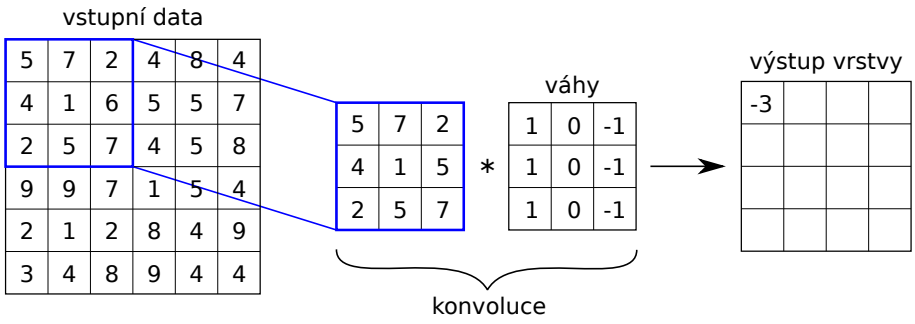
3.2.2 Konvoluční vrstva

Konvoluční vrstva umí efektivně najít vzory, které se ve vstupních datech vyskytují. Tyto vzory jsou vlastně příznaky, které kódují vlastnost, podle které síť klasifikuje. Konvoluce se používá jako vstupní vrstva při klasifikaci obrázků. U obrázků může konvoluční vstupní vrstva detekovat třeba hrany.

Konvoluční vrstva na vstupní data aplikuje diskrétní konvoluci skrze kernel - jádro obsahující váhy. Konvoluční vrstvy mohou být jedno nebo vícedimenzionální. Obsahují určité množství filtrů (jednotlivých kernelů) - pro obrázky by to znamenalo,

že první filtr detekuje například vodorovné hrany a druhý filtr horizontální hrany. Kernel má určitý rozměr ve stejné dimenzi jako vstupní data. Vrstva může obsahovat padding - doplnění okrajů dat určitou hodnotou. Padding se využívá mimo jiné pro zachování rozměrů vstupu a výstupu konvoluční vrstvy. Posledním parametrem je stride, který říká, o kolik se má mřížka kernelu posouvat.

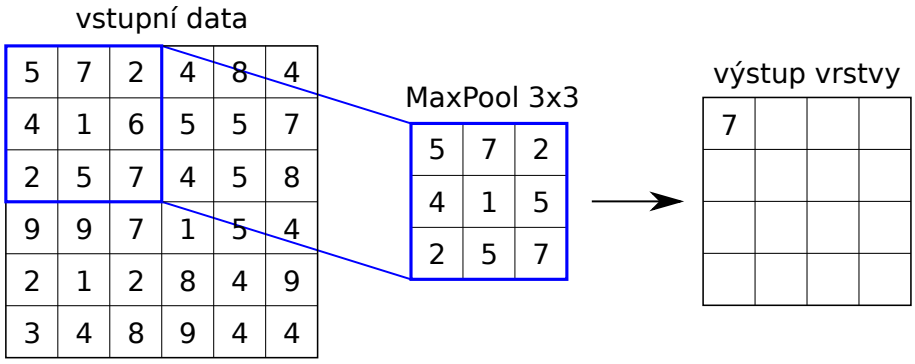
Obrázek 3.14 ukazuje, jak pracuje konvoluční vrstva. Modelová vrstva z obrázku by se dala charakterizovat jako 2-D konvoluční vrstva s jádrem o velikosti 3x3, jedním filtrem, bez paddingu. Konvoluční vrstva nejdřív vybere vstupní data, provede konvoluci s kernelem (vahami) a výsledek zapíše na příslušnou výstupní pozici. Pokud bychom předpokládali stride 1, pak by se modrý čtverec ve vstupních datech posunul o jeden sloupec doprava a postup by se zopakoval. Takto se získá celý výstup konvoluční vrstvy.



Obrázek 3.14. Princip zpracování dat v konvoluční vrstvě

3.2.3 MaxPool

MaxPoolingová vrstva je druhem subsamplingové (sdružovací) vrstvy. Tato vrstva neobsahuje žádné učící se parametry, slouží pouze ke zmenšení dimenze vstupních dat. Vrstva obsahuje jádro, které se pohybuje po vstupních datech podobně jako konvoluční jádro v konvoluční vrstvě. I zde je nutné nastavit alespoň dva parametry: rozměr jádra a stride. Jádro poté ze vstupních příznaků, které překrývá, vybere ten největší prvek (obrázek 3.15).



Obrázek 3.15. Princip zpracování dat v MaxPoolingové vrstvě

Díky MaxPoolingu zmenšíme objem dat v síti a následující vrstvy tedy obsahují menší množství učících se parametrů. Do jisté míry se tím zlepšuje detekce objektů, jelikož se zahodí nedůležitá data (z určité oblasti se vybere pouze maximum). Jinými slovy, jestliže je detekován příznak s vysokou hodnotou např. v konvoluční vrstvě, následující MaxPooling zajistí, že nezáleží, kde přesně v MaxPool jádru byl příznak detekován.

3.2.4 Dropout

Dropout je další vrstvou, která neobsahuje učící se parametry. Vrstva náhodně deaktivuje určitý počet svých vstupů. Množství znehodnocené informace je parametr této vrstvy. Chceme-li zlepšit vyhledávání obecných vztahů ve vstupních datech, nikoliv pouze memorování trénovacích dat, dropout k tomu může pomoci. Vrstva přidá do sítě jistý druh šumu. Díky tomu síť nebude tolik citlivá na mírné změny ve vstupních datech.

3.2.5 BatchNorm

BatchNorm vrstva také neobsahuje učící se parametry, slouží k normalizování střední hodnoty a standardní odchylky dávky dat v konkrétní části sítě. Obecně platí, že normalizovaná vstupní data dosahují v neuronových sítích lepších výsledků, než data nenormalizovaná. Data po průchodu částí sítě však normalizovaná už jistě nejsou. Proto je vhodné použít BatchNorm. Normalizace pomáhá k mizení nebo explodování gradientu, tím pádem pomáhá i ke zpětné propagaci chyby do hloubky sítě.

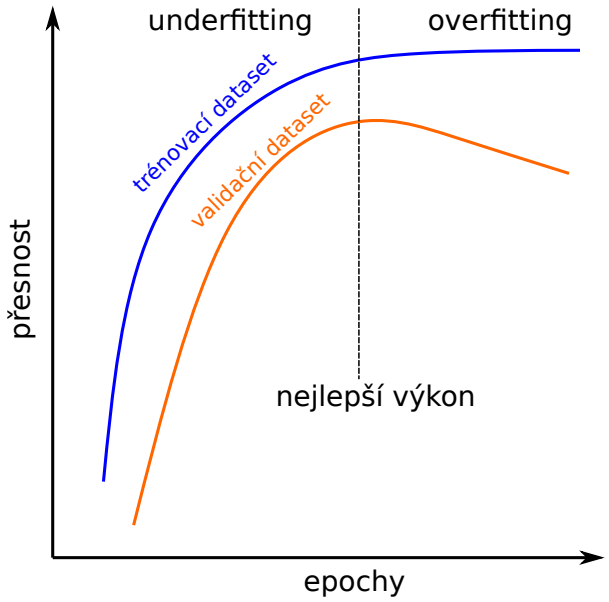
3.3 Problémy

V předchozí části práce jsem zmínil některé problémy, na které jsou neuronové sítě náchylné. Problém mizení gradientu zapříčiní to, že se nebudou učit neurony, které jsou uložené hluboko v síti. Naopak při explodování gradientu může dojít až k přetečení datového typu, ve kterém je uložená aktivace. Výsledek sítě tak nebude interpretovatelný. Vznik nebo důsledky těchto problémů lze výrazně omezit vhodným výběrem aktivačních funkcí nebo BatchNormem.

Kvalitu učení sítě můžeme sledovat několika způsoby. Jedním z nich je graf přesnosti v závislosti na čase - resp. trénovací epoše. Přesností myslíme pravděpodobnost, že prvek bude klasifikován do správné třídy. Do grafu se kreslí dvě křivky - přesnost trénovacích a validačních dat. Podle vzájemné polohy těchto křivek je možné odhadnout, jak probíhá učení. Cílem je dosáhnout co nejvyšší přesnosti na validačních i trénovacích datech.

Jestliže dosahujeme vysoké přesnosti na trénovacích datech, ale nízké na validačních, dochází k přeučení sítě (overfittingu). K jevu přispívá např. malý dataset, rozsáhlá architektura sítě nebo příliš dlouhá doba učení. Abychom zabránili overfittingu, můžeme přidat Dropout, případně MaxPool vrstvy, zmenšit neuronovou síť, zkrátit dobu trénování sítě nebo rozšířit dataset. Dataset je možné rozšířit uměle tzv. augmentací datasetu.

Jestliže dosahujeme nízké přesnosti na validačních i trénovacích datech, došlo k nedoučení sítě (underfitting). Architektura je příliš mělká na vyřešení našeho problému nebo trénovali jsme síť krátkou dobu. Síť není schopná vyhledat souvislosti ve vstupních datech a často nefunguje ani na trénovacím datasetu.



Obrázek 3.16. Graf přesnosti v závislosti na počtu epoch znázorňující oblast underfittingu, overfittingu a dělící čáry, kdy je nejlepší přerušit učení

Další problém, se kterým se můžeme potýkat, je nevyvážený dataset. Představme si, že bychom klasifikovali do dvou tříd A a B. V trénovacím datasetu bychom měli počet vzorků třídy A a B v poměru 1:10. Síť by se naučila klasifikovat primárně do třídy B, nezávisle na vstupních datech, jelikož klasifikace do třídy B je 10x pravděpodobnější, oproti přiřazení vstupu do třídy A. Tento problém můžeme vyřešit několika způsoby. Jedním způsobem je zahrnutí dat z minoritně zastoupené třídy vícekrát v jedné epoše, tak aby se počty vzorků jednotlivých tříd v jedné epoše vyrovnaly. Druhým způsobem, který byl použit v této práci, je poměrové rozvážení hodnot ztrátové funkce pro jednotlivé třídy. V uvedeném příkladu bychom hodnotu ztrátové funkce pro třídu A zvětšili 10x a hodnotu ztrátové funkce pro třídu B bychom nezvětšovali.

Obecně se snažíme dosáhnout co nejlepší generalizace predikcí, tedy aby síť co nejméně memorovala příznaky vyskytující se v trénovacích datech a aby se co nejvíce naučila obecné spojitosti v příznacích, které vedou ke správné predikci. Tato vlastnost nemusí být dobře patrná ani z grafu přesnosti klasifikace. Například při rozpoznávání emocí se síť do jisté míry učí celé věty, nikoliv pouze emoce. Proto mohou být výsledky sítě lepší na větách vyskytujících se v datasetu, než na větách, které v datasetech obsaženy nebyly.

3.4 Augmentace

Augmentace je umělé rozšíření datasetu, za účelem předejití přeučení sítě na trénovacích datech. Pro představu uvedu příklad neuronové sítě, která rozeznává auta z obrázků. Pokud se síť bude učit pouze na datech červených aut, nebude schopná rozeznávat auta jiných barev. Abychom rozšířili dataset, nepotřebujeme vytvářet nové fotografie aut. Stačí náš dataset rozšířit o ty samé obrázky, které mají pouze změněné barvy.

Při augmentaci nesmí dojít ke ztrátě informace, která je klíčová ke klasifikaci nebo ke změně třídy, do které data klasifikujeme. V příkladu s auty se změnou barev jistě nezměnil objekt na obrázku, stejně tak je člověk schopný rozeznat, že na něm je stále auto. Proto lze takovou augmentaci použít.

Augmentaci zvukových nahrávek lze realizovat jejich zašuměním, zpomalením, posunem nahrávky ve frekvenční oblasti nebo posunem nahrávky v časové oblasti. Při posunu v časové oblasti se začátek nahrávky posune a vzniklé místo se zaplní koncem stejné nahrávky (operace roll - podobně jako by nahrávka byla kotouči, který bychom pootočili). [16]

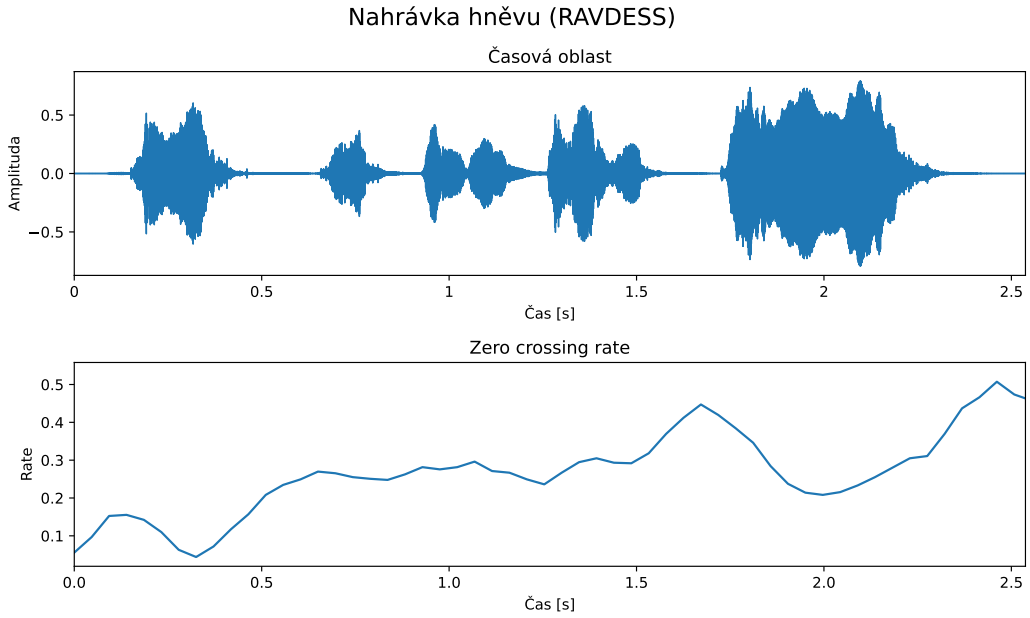
Kapitola 4
Extrakce příznaků ze zvuku

Někdy není žádoucí, aby data vstupovala do sítě v surovém stavu tak, jak je máme k dispozici (jako zvukovou nahrávku - posloupnost hodnot v čase). Extrakce příznaků je proces transformace dat před vstupem do sítě. Smyslem extrahování je usnadnění zpracování dat síti, resp. zrychlení jejího natrénování. Snažíme se o zahození dat, u kterých si myslíme, že nejsou pro klasifikaci důležitá, nebo naopak získáváme reprezentace dat, ze kterých lze přímočaře vytvářet predikce. Extrahování je vždy závislé na metodě strojového učení, jelikož každá metoda, příp. architektura neuronové sítě očekává data v jiném formátu a jiné reprezentaci.

Ze zvukové vlny můžeme získat mnoho příznaků. Hlas v nahrávce je charakterizován parametry, které jsou popsány např. v článku [17]. Příznaky lze získat transformacemi nahrávky v časové nebo frekvenční oblasti. Příznaky, které jsou níže popsány byly použity v článcích [18–19] pro SVM klasifikátor. V článcích [20–21] byly použity tyto příznaky pro klasifikaci na neuronových sítích.

4.1 Zero Crossing Rate (ZCR)

Jedná se o podíl průchodu signálu nulou v rámci určitého časového okna. Okno se posouvá postupně v časové oblasti zvukové vlny a tím vygeneruje vektor těchto příznaků.

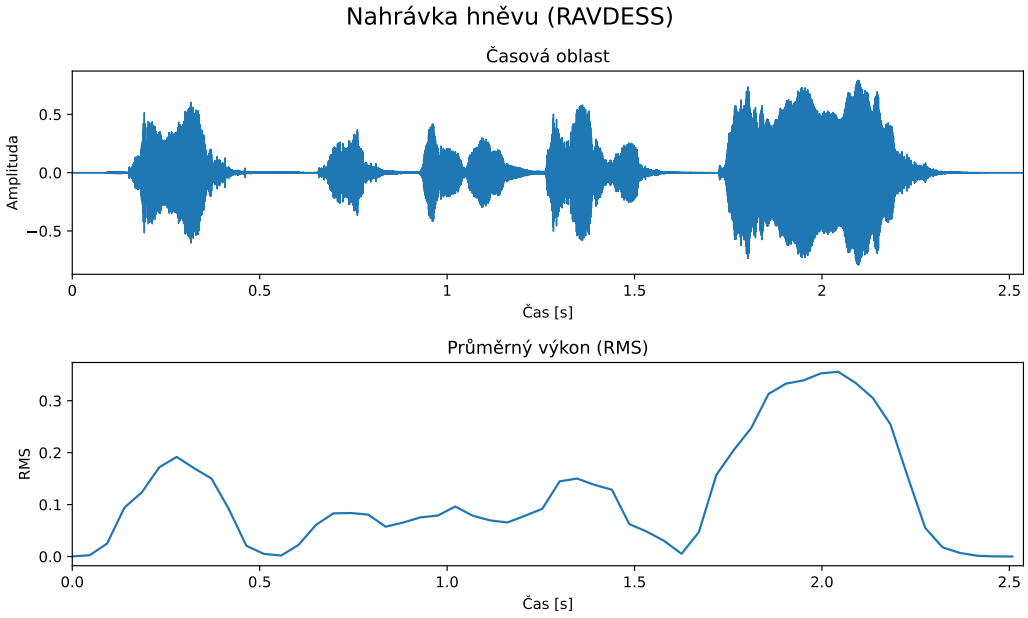


Obrázek 4.1. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její Zero Crossing Rate

ZCR se často používá pro detekci znělých úseků ve zvukovém záznamu. Je-li podíl průchodů nulou vysoký, může to vypovídat o tichu nebo šumu v dané části nahrávky.

4.2 Root Mean Square (RMS)

RMS neboli kvadratický průměr je číslo, které udává průměrnou velikost energie zvukové vlny v daném okně. Hodnota může odpovídat např. hlasitosti zvuku, který je v daném okně zaznamenán.



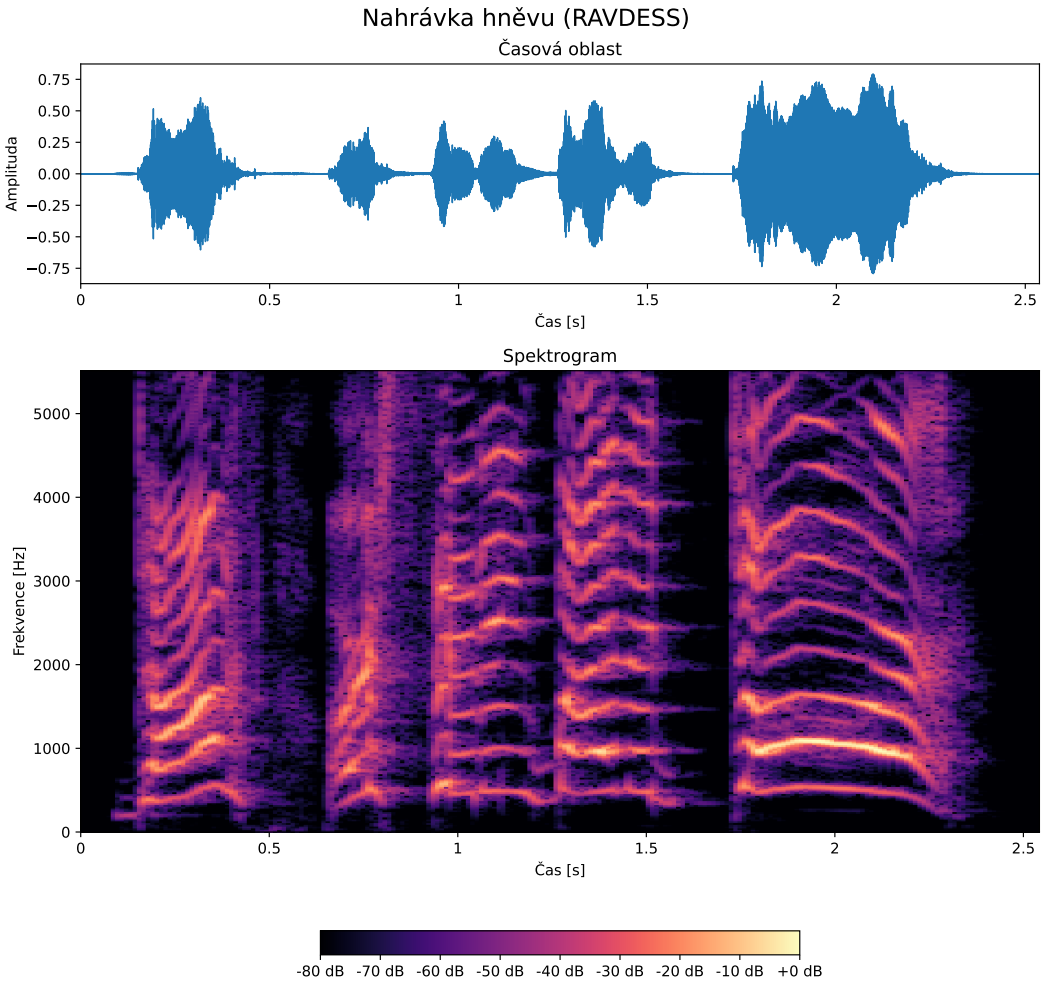
Obrázek 4.2. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její Root Mean Square

4.3 Spektrogram a Mel-spektrogram

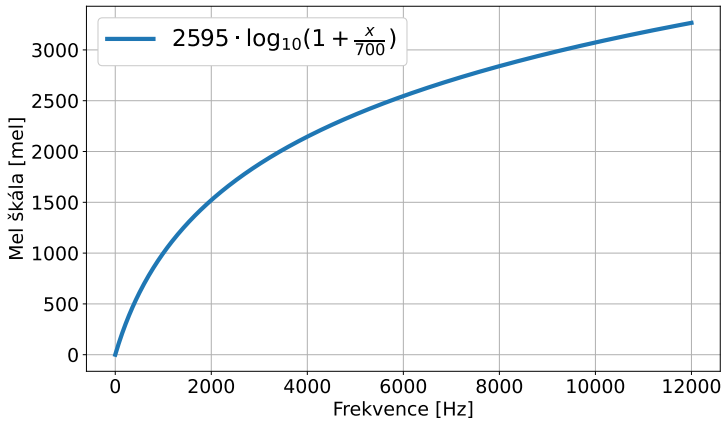
Frekvenční spektrum je vektor reprezentující frekvenční zastoupení v určitém časovém úseku nahrávky. Spektrogram zobrazuje změny frekvenčního spektra v čase (frekvenční spektrum se vypočítá z malého okna, které se posouvá v čase), jde o další z příznaků. Příklad spektrogramu se nachází na obrázku 4.3.

Lidské vnímání obecně není lineární, ale logaritmické. To platí i u vnímání frekvencí. V nízkých frekvencích snadno vnímáme malé diference (např. rozdíl mezi 100 a 150 Hz), pokud stejně velké diference uslyšíme ve vyšších frekvencích (10000 a 10050 Hz), rozdíl téměř nepoznáme. Naše vnímání tedy potlačuje rozdíly tím více, čím vyšší frekvenci slyšíme. Melová stupnice (od slova melody) přeškáluje frekvence do melů, kde diference v melech odpovídá zhruba stejné diferenci i v lidském vnímání. Převodní vztah vyjadřuje obrázek 4.4.

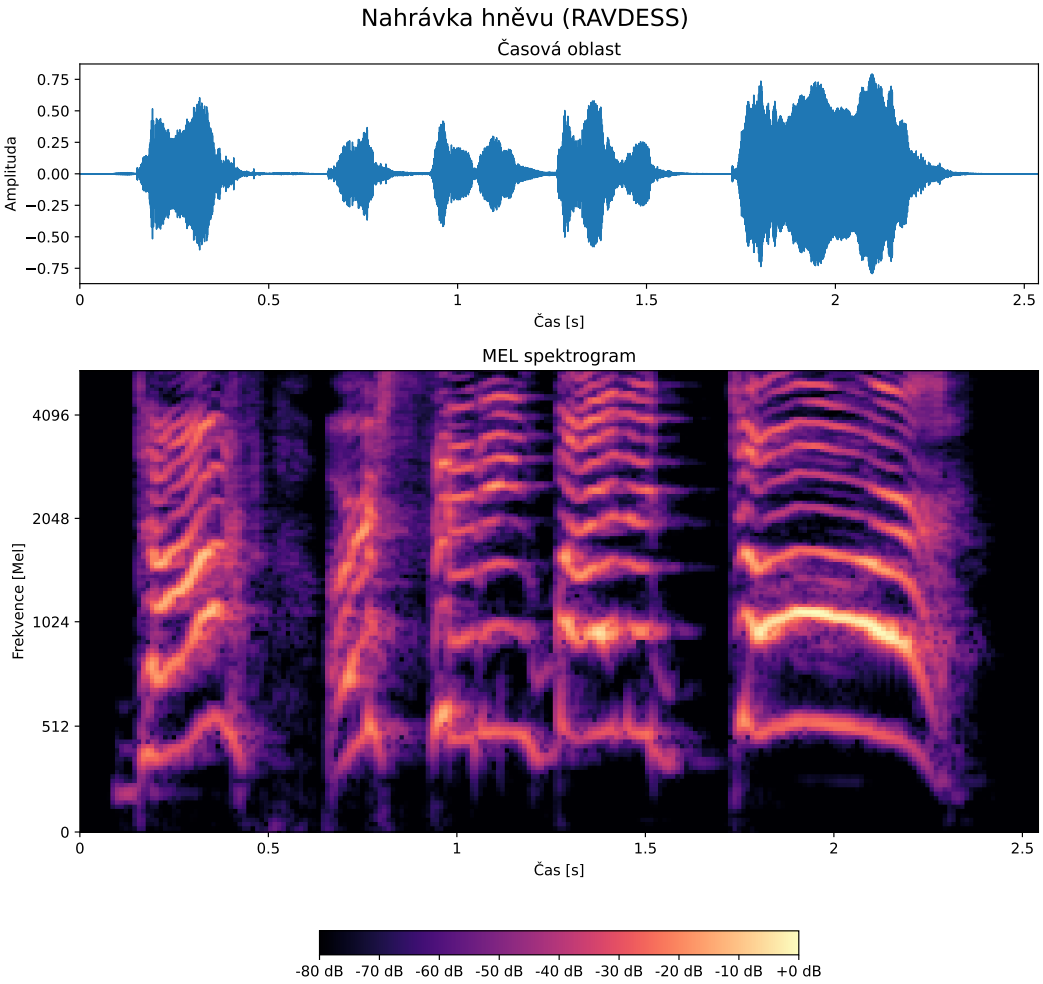
Pokud frekvence ve spektrogramu převedeme do melů, získáme tzv. Mel spektrogram. Tento spektrogram má zhuštěné vyšší frekvence. Příklad Mel spektrogramu se nachází na obrázku 4.5.



Obrázek 4.3. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její spektrogram



Obrázek 4.4. Převodní graf z frekvence v Hz na Melovou stupnici

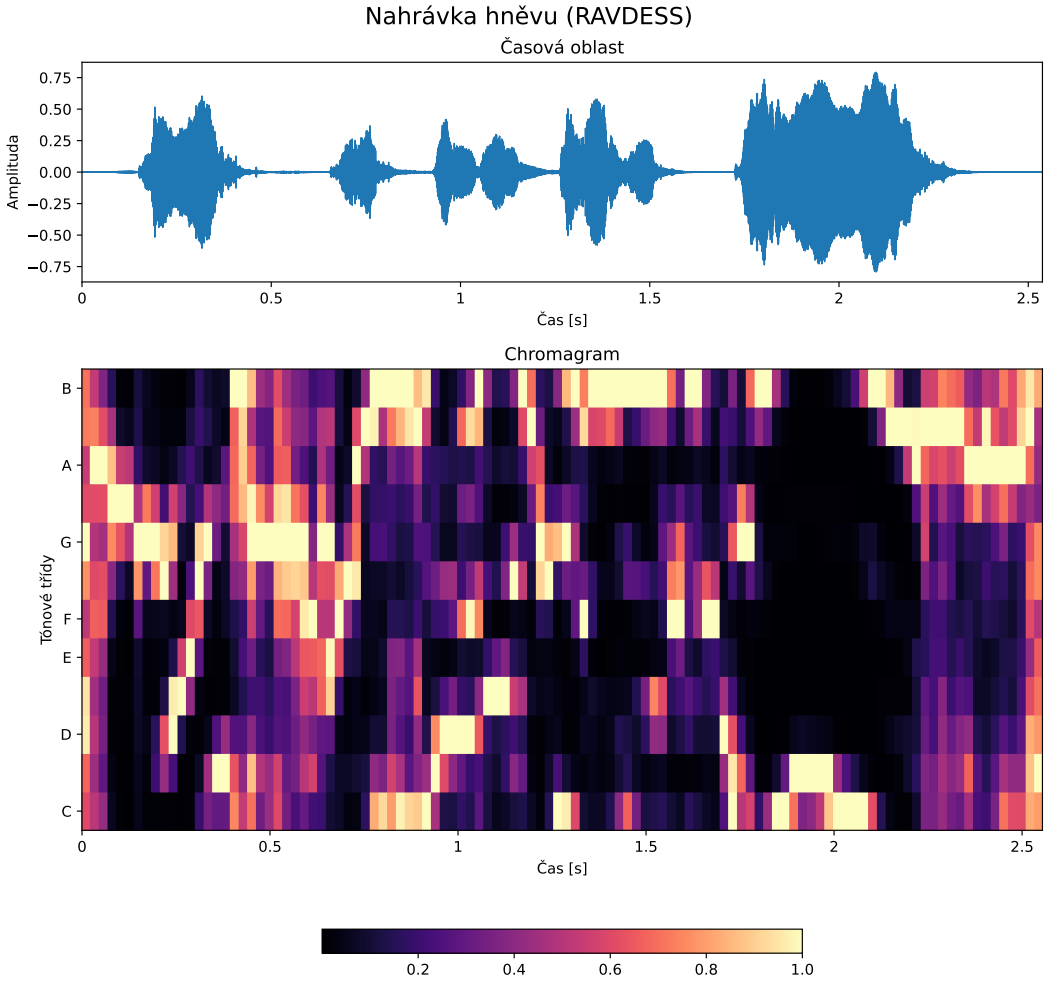


Obrázek 4.5. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její Mel spektrogram

4.4 Chromagram

Chroma je vektor, který vyjadřuje množství energie, které je zastoupeno v jednotlivých tónových třídách (půltónech). Tónových tříd je 12, každá třída je stejně velká. Tónové třídy dohromady tvoří oktávu. Jestliže zdvojnásobíme frekvenci půltónu, vytvoříme stejný půltón ve vyšší oktávě, tím pádem násobky frekvence (vyšší harmonické frekvence) daného půltónu přísluší stejné tónové třídě. Dva různé půltóny jsou lidmi vnímány barevně podobně, pokud se od sebe liší o oktávu.

Chromagram je posloupnost chroma vektorů v čase (obdoba spektrogramu). Chromagram se nachází na obrázku 4.6.



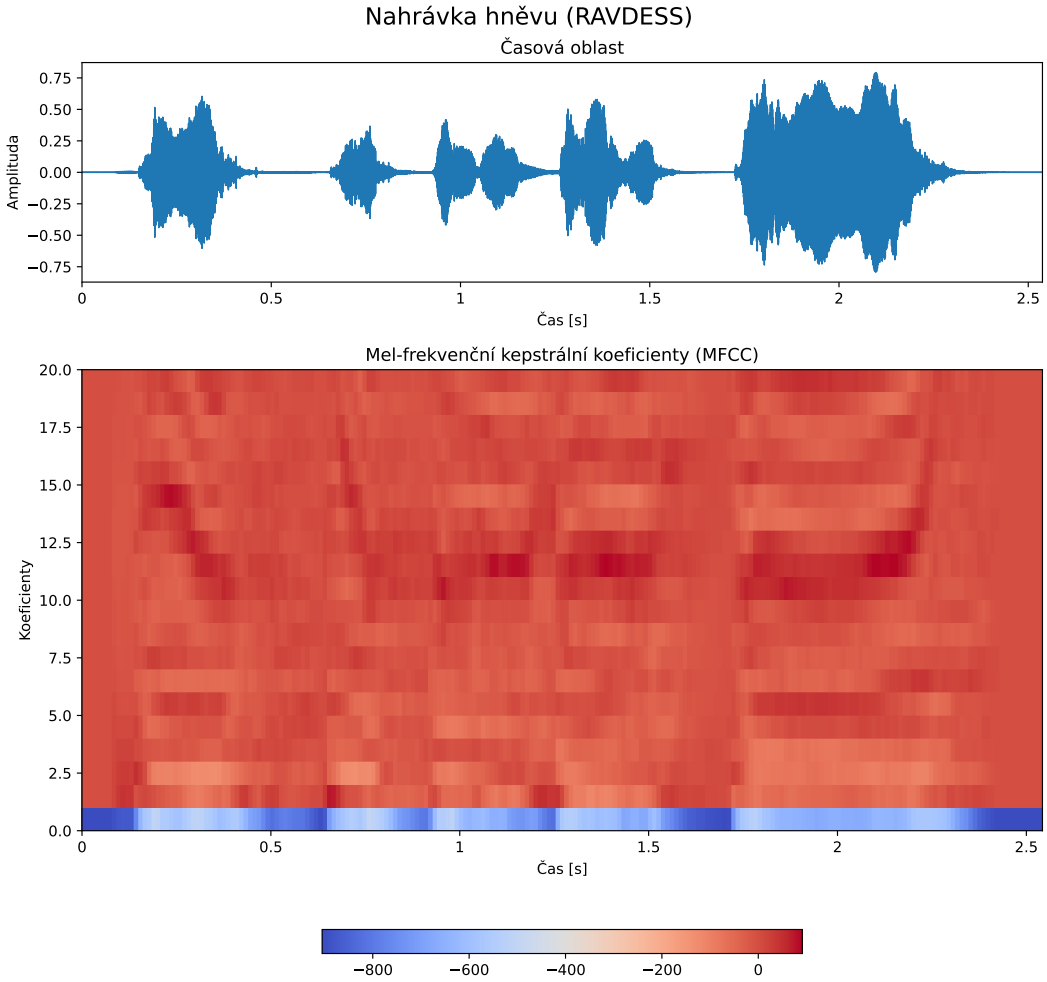
Obrázek 4.6. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její Chromagram

4.5 Mel-frekvenční kepstrální koeficienty (MFCC)

Kepstrum je spektrum spektra. Kepstrum reprezentuje periodicitu ve spektru signálu.

Mel-frekvenční kepstrální koeficienty [23] jsou koeficienty, které vzniknou součtem amplitud kepstra v melech, které je překryto jistým počtem trojúhelníkových filtrů.

Nejprve se ze signálu v časové oblasti vypočítá spektrum pomocí krátkodobé Fourierovy transformace, které se umocní, abychom získali výkonové spektrum. Následně se spektrum převede do Melovy stupnice. Takovéto spektrum se pak překrývá filtry. Vyfiltrované hodnoty se sečtou, zlogaritmují a diskrétní kosinovou transformací se získají koeficienty MFCC.



Obrázek 4.7. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její MFCC

4.6 Průměr a standardní odchylka základní frekvence

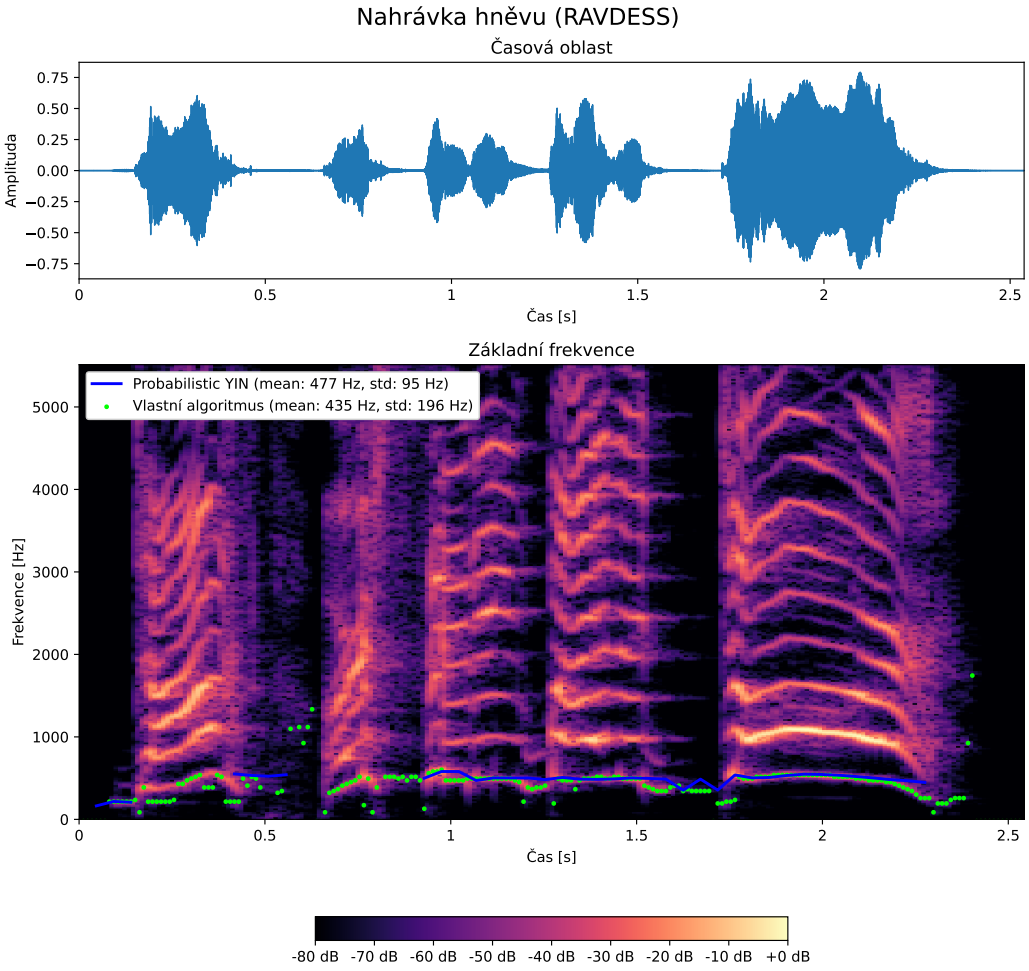
Pokud bychom chtěli hodnotově vyjádřit zvlněnost spektrogramu, musíme z něj extrahovat základní frekvenci hlasu. Základní frekvencí myslíme nejnižší frekvenci, kde se vyskytuje lokální maximum energie zaznamenaného hlasu. Vyšší harmonické frekvence hlasu kódují jeho barvu. Pokud bychom postupně snižovali maximální frekvenci přenosového pásma (odřezávali bychom vyšší harmonické frekvence), snižovali bychom zabarvení řeči pro daného konkrétního mluvčího. Stále bychom ale byli schopni přesně rozeznat, co mluvčí říká. Toho se využívá například u telefonů, aby

se ušetřila šířka přenosového pásma, proto zvuk z telefonního hovoru zní odlišně od zvuku nahraného na stejný mikrofon.

Jestliže získáme posloupnost základní frekvence v čase, vypočítáme z ní průměr a standardní odchylku. Průměr se standardní odchylkou základní frekvence jsou hodnoty, které přibližně popisují zvlnění spektrogramu.

Obrázek 4.8 zobrazuje spektrogram s vyznačenou základní frekvencí. Modrá čára ve spektrogramu vyjadřuje základní frekvenci získanou metodou Probabilistic YIN [24]. Zelené tečky vyznačují vlastní algoritmus, který hledá základní frekvence v rozsahu 65 a 2093 Hz. Algoritmus postupuje po sloupcích zespoda spektrogramu. Vždy hledá v uvedeném rozsahu frekvenci, která obsahuje energii o 9 dB větší než je na nejnižší frekvenci spektra. Díky tomu algoritmus narazí na energeticky významnější oblast a teprve v ní začne hledat lokální maximum. Lokální maximum poté odpovídá základní frekvenci.

Průměr základní frekvence spočítané metodou Probabilistic YIN je 477 Hz, metodou mnou navrženého algoritmu je průměr 435 Hz. Standardní odchylka spočítaná metodou Probabilistic YIN je 95 Hz, standardní odchylka spočítaná mnou navrženým algoritmem je 196 Hz.



Obrázek 4.8. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její základní frekvence

Kapitola 5
Datasety pro rozpoznávání emocí z hlasu

V práci používám tři níže zmíněné datasety, které jsem použil k trénování, validaci a testování sítě. Tyto datasety se používají i v řadě současných publikací zabývajících se rozpoznáváním emocí z hlasu.

Každý dataset obsahuje určité množství zvukových nahrávek s anotací, o jakou emoci se jedná. Datasety se vzájemně liší například: svým rozsahem, kvalitou pořízené nahrávky, jazykem mluvčího, větami nahrávek, počtem mluvčích, druhy anotovaných emocí, atd.

Protože v této práci pracuji s přeloženými názvy tříd emocí, v tabulce 5.1 uvádím originální názvy tříd a překlady používané v této práci.

EmoDB	RAVDESS	CREMA-D	Překlad
Ärger (Wut)	Angry	Anger	Hněv
Ekel	Disgust	Disgust	Znechucení
Angst	Fearful	Fear	Strach
Freude	Happy	Happy/Joy	Štěstí
Trauer	Sad	Sad	Smutek
Neutral	Neutral	Neutral	Neutrální

Tabulka 5.1. Originální názvy tříd a jejich překlad použitý v této práci.

5.1 EmoDB

EmoDB neboli Berlin Database of Emotional Speech je dataset, který byl nahrán v letech 1997 a 1999 na Technické univerzitě v Berlíně. Dataset byl nahraný v anechoické komoře pěti ženami a pěti muži, kteří postupně simulovali emoce při vyslovování pěti krátkých a pěti delších vět. Věty jsou v německém jazyce. V datasetu je zastoupených 7 emocí: hněv, nuda, úzkost, štěstí, smutek, znechucení a neutrální emoce.

Dataset obsahuje celkem 535 nahrávek ve studiové kvalitě (záznam vzorkovací frekvencí 48 kHz a poté podvzorkování na 16 kHz). Každá nahrávka je identifikovatelná číslem herce (vč. jejich věku a pohlaví). Dále textem, který zní na nahrávce. A nakonec je označená emocí, která je na nahrávce zachycená. [10]

5.2 RAVDESS

RAVDESS je zkratkou pro Ryerson Audio-Visual Database of Emotional Speech and Song. Dataset obsahuje audiovizuální nahrávky dvanácti žen a dvanácti mužů se severoamerickým akcentem. Protože se ve své práci zaobírám pouze rozpoznáváním emocí z hlasu, dále budu analyzovat a pracovat pouze se hlasovým materiálem.

Hlasových nahrávek v RAVDESS je 1440. Zvuk je zaznamenaný ve studiové kvalitě 16bit, 48kHz. Každý z 24 herců má 60 nahrávek: 2 věty, 2 opakování, 8 emocí a 2

intenzity (kromě neutrální emoce). Dataset obsahuje neutrální emoci, klid, štěstí, smutek, hněv, strach, znechucení a překvapení. Každá nahrávka je identifikovatelná číslem herce (a jeho pohlavím), emocí, emoční intenzitou (normální/silná) a textem věty.[22]

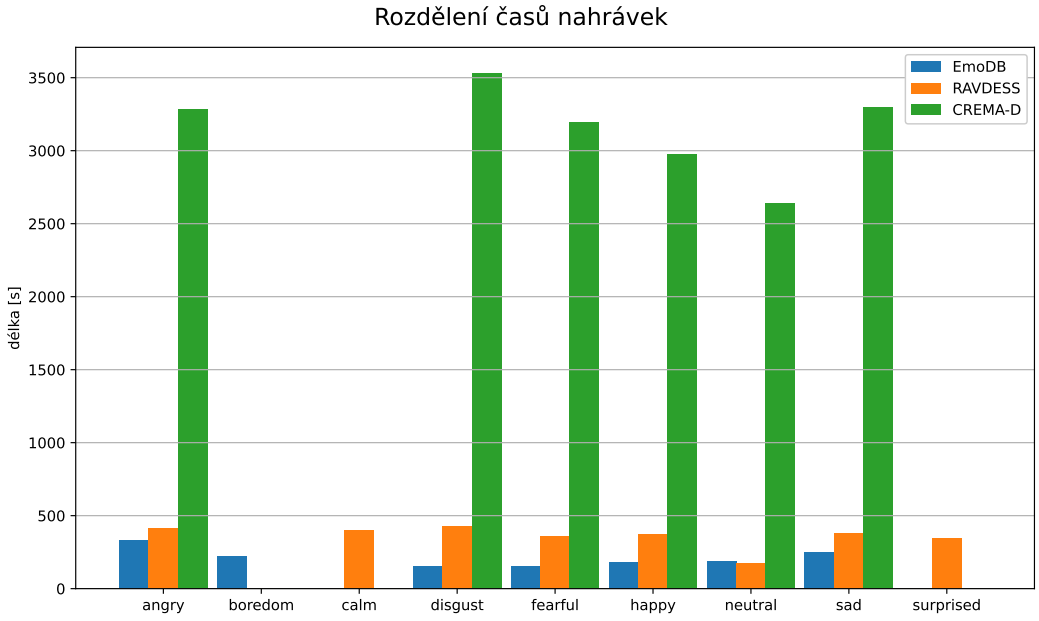
5.3 CREMA-D

CREMA-D je zkratkou pro Crowd-sourced Emotional Multimodal Actors Dataset, dataset je velmi rozsáhlý. Obsahuje 7442 nahrávek od 91 herců. Dataset je v britské angličtině a je různorodý, díky početnému zastoupení různých etnik, ras a širokému rozložení věku herců.

Herci simulují emoce na dvanácti větách. Emoce jsou: hněv, znechucení, strach, štěstí, smutek a neutrální emoce. Emoce jsou také rozděleny podle intenzity: nízká, střední, vysoká a nespecifikovaná.[25]

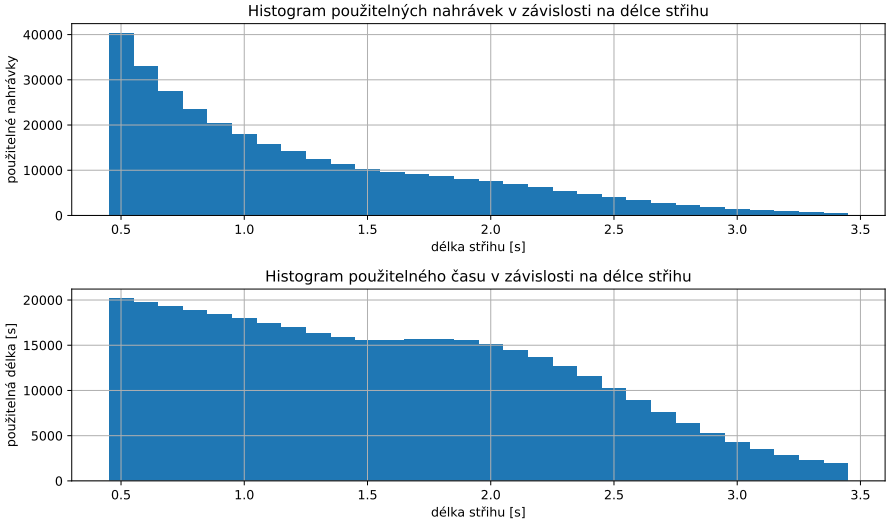
5.4 Zpracování nahrávek

Abychom mohli použít co největší množství nahrávek, vzájemně zkombinujeme zmíněné datasety. Protože dataset CREMA-D obsahuje nejméně emočních tříd, omezíme se v ostatních datasetech pouze na tyto třídy. Klasifikace bude provedena pouze do šesti emocí: hněv, znechucení, strach, štěstí, smutek a neutrální emoci. Taktéž sjednotíme vzorkovací frekvenci nahrávek. Ze zadání práce budeme síť testovat na hlasu z vysílaček, které mají nízké přenosové pásmo. Proto byly trénovací nahrávky podvzorkovány na frekvenci 11 025 Hz. Zároveň byly všechny nahrávky převedeny do nekomprimovaného formátu wav. Některé datasety mají na začátku a na konci neznělé úseky o délce několika stovek milisekund. Tyto úseky byly odstraněny. Ke zmíněným operacím byl použit program ffmpeg.



Obrázek 5.1. Rozložení počtu nahrávek v datasetech EmoDB, RAVDESS a CREMA-D pro jednotlivé třídy emocí s nepřeloženým názvem

Taktéž je žádoucí mít normalizovanou délku nahrávek. Normalizaci realizujeme nastříháním nahrávek na kratší úseky, přičemž tyto kratší úseky zachovají stejnou třídu (emoce se zkrácením nahrávky nezmění). Pokud bychom nastříhali nahrávky na příliš krátké úseky, nebude v nich patrná emoce ani pro člověka. Pokud zvolíme příliš dlouhou délku střihu, budeme uměle zmenšovat dataset (zahodíme celé nahrávky, které jsou krátké). Z horního histogramu 5.2 je patrné, že při krátké délce střihu vznikne příliš mnoho krátkých nahrávek. V dolním histogramu použitelných délek nahrávek vidíme, že existuje lokální maximum při střihu 1.8 s. To je dostatečná délka nahrávky pro slyšitelnost projevované emoce, proto jsem zvolil tuto délku. Střih je realizovaný rozdělením nahrávky na úseky o 1.8 s, poslední úsek nahrávky je zahozen, jelikož nemá požadovanou délku.



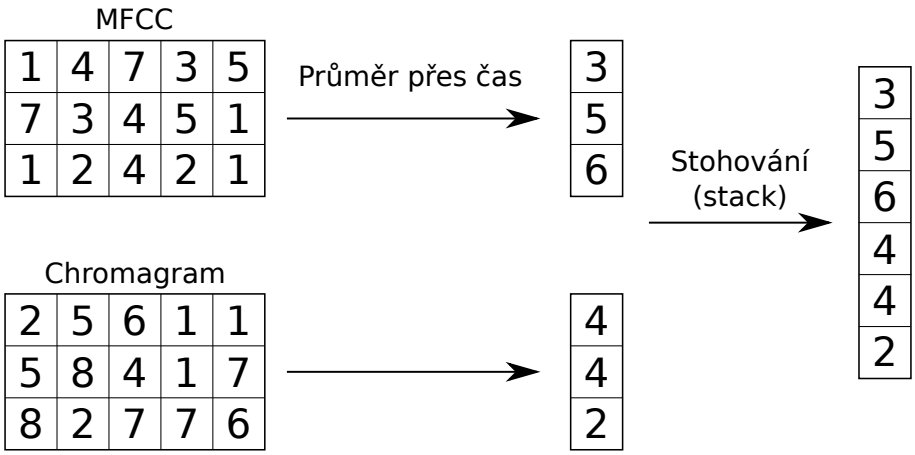
Obrázek 5.2. Histogram počtu nahrávek v závislosti na délce střihu (nahoře) a histogram délek nahrávek v závislosti na délce střihu (dole)

Kapitola 6
Síť s kombinací několika druhů příznaků

V této kapitole nejprve zmíním postup při vývoji sítě a nakonec popíšu takové sestavení, se kterým jsem dosáhl nejlepších výsledků.

Neuronová síť byla realizována v programovacím jazyku Python, který je populárním jazykem zejména pro svoji jednoduchost, platformní nezávislost, flexibilitu, velké spektrum knihoven a pro širokou komunitu. Pro načtení nahrávek z wav souboru a následnou extrakci jejich příznaků jsem použil knihovnu LibROSA [26]. Pro vývoj umělé inteligence jsem poté využil knihovnu Keras [27], která poskytuje jednoduché programovací rozhraní (API) pro knihovnu TensorFlow [28], která zajišťuje tenzorové operace.

Po přípravě dat podle sekce 5.4 jsem začal s návrhem neuronové sítě, vycházel jsem z článků [29, 3], které extrahují MFCC. Koeficienty vstupují do neuronové sítě, jejíž architektura se skládá z několika bloků 1D konvolucí s MaxPool vrstvou. Bloky jsou doplněny o Dropout nebo BatchNorm. Nakonec jsou data zploštěna a přejdou do Fully Connected vrstvy. Ta je buď poslední vrstvou nebo jsou to dvě vrstvy za sebou. Obdobný přístup je i v článku [20]. V článku jsou však extrahovány kromě koeficientů MFCC i Zero Crossing Rate, Root Mean Square, Chromagram a Mel spektrogram. Pro každý z těchto příznaků je vypočítaný průměr přes čas. Následně jsou tyto průměry seskupeny za sebou do vektoru, který vstupuje do neuronové sítě. Z obrázku 6.1 můžeme získat představu o generování vstupního vektoru a jeho rozměrech pro každou z nahrávek.



Obrázek 6.1. Znázornění získání příznaků, průměr přes časovou osu a vytvoření vstupního vektoru do neuronové sítě

Veškeré skripty obsahující kód, který pochází z této části bakalářské práce jsem uložil do složky `./src1/`. Nejprve jsem vytvořil modul `datasets.py` načítající z připravené složky nahrávky jednotlivých datasetů, modul zároveň dekóduje anotovanou třídu z názvu souboru nahrávky. Poté jsem vytvořil modul `augmentation.py` pro augmentaci, obsahující funkce pro jednotlivé augmentační metody - vstupem je

posloupnost čísel reprezentující zvukovou vlnu, výstupem je opět zvuková vlna, na kterou byl aplikovaný požadovaný efekt. Třetí vytvořený modul je generátor příznaků `features_generator.py`, který načte každý soubor z datasetu, zvukovou vlnu rozstříhá na požadovanou délku 1.8 s, pro každý úsek vytvoří augmentace a originální i augmentovaná data převede na požadovaný vektor příznaků. Vektory příznaků jsou poté uloženy do souborů ve složce `./features/`.

Modul zajišťující trénování sítě `train.py` načte soubory s příznaky, které rozdělí v poměru 80:20 na trénovací a validační množinu. Data z těchto souborů jsou načítána pouze v požadovaných dávkách. Jakmile je vytvořena trénovací množina, je možné určit početní zastoupení vzorků v jednotlivých třídách. Díky tomu vytvoříme váhy pro vyvážení tříd datasetu. Poté modul sestaví požadovanou architekturu sítě a začne síť trénovat. V případě dobrého výsledku je možné natrénovaný model sítě uložit pro pozdější produkční použití.

Před laděním samotných parametrů jsem ověřil výběr délky střihu. Střih nahrávky na délku 1.8 s je dostatečně dlouhý, aby člověk emoci z nahrávky rozeznal a nachází se zde lokální maximum délky celého použitého datasetu. Experimentálně jsem zjistil přesnost sítě při různých délkách střihu okolo 1.8 s.

Délka střihu [s]	Přesnost sítě [%]
0.6	36.17
0.8	38.76
1.0	39.73
1.2	43.47
1.4	43.44
1.6	45.47
1.8	45.08
2.0	43.55

Tabulka 6.1. Přesnost neuronové sítě v závislosti na délce střihu nahrávek

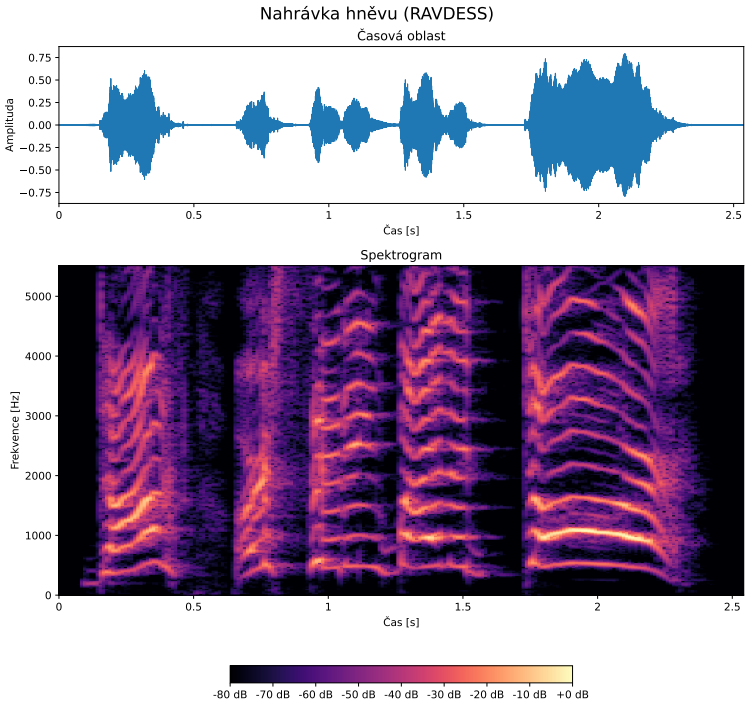
Z tabulky 6.1 vidíme, že při délce střihu mezi 1.6-1.8 s dosahujeme nejvyšší přesnosti na validačním datasetu. Přestože 1.8 s má mírně horší přesnost, než délka 1.6 s, setrval jsem právě u délky střihu 1.8 s. Přesnost v tabulce může být mírně ovlivněna počáteční náhodnou inicializací vah sítě, ale především při délce střihu 1.8 s dosáhneme maxima použitelných délek - neboli ztratíme méně dat z původních nenastříhaných datasetů.

Následně jsem experimentoval s výběrem parametrů a druhy příznaků, které vstupují do neuronové sítě. Původně jsem začal podle článku [20] s průměry RMS, Zero Crossing Rate, Chroma, MFCC a Mel spektrogramem. Ukázalo se, že důležitým parametrem u příznaků vyžadující výpočet spektrogramu je velikost okna, ze kterého se počítá Fourierova transformace. Přesnost sítě na tomto parametru velmi závisí. Čím menší okno bude, tím jemnější spektrogram v čase budeme mít. Experimentoval jsem i s různými variacemi použitých příznaků. Dosažené přesnosti po odstranění jednotlivých příznaků je možné nalézt v tabulce 6.2. Přestože samotné odstranění příznaků nevedlo ke zlepšení přesnosti klasifikace na validační množině, vliv byl patrný na zlepšení přesnosti na trénovací množině. Z tohoto pohledu odstranění MFCC zlepší kvalitu sítě nejvíce. Kombinace několika odstraněných příznaků už vedlo pouze k horším výsledkům.

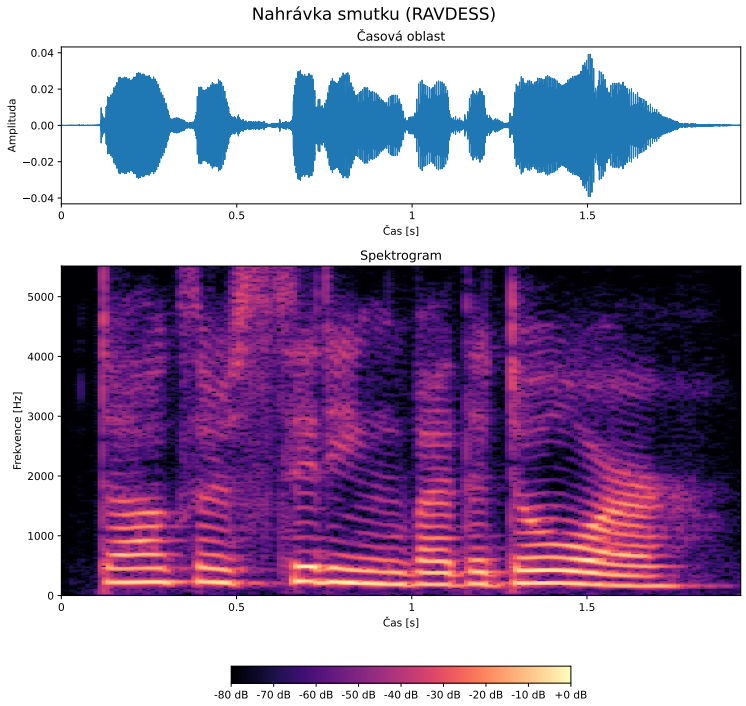
Odstraněno	Trén. mn. [%]	Valid. mn. [%]
-	75.0	43.0
RMS	71.7	42.0
ZCR	72.7	43.3
MFCC	82.3	41.7
Mel sp.	66.0	40.7
Chroma	77.3	41.7
Chroma+MFCC	79.7	41.7

Tabulka 6.2. Přesnost na trénovací i validační množině při trénování s příznaky RMS, ZCR, Chroma, MFCC a Mel spektrogram. A poté přesnosti při odstranění konkrétního příznaku z uvedené množiny.

Při pohledu na spektrogramy jednotlivých emocí stejné věty je zřejmé, že se liší svým zvlněním. Například nahrávka hněvu (obrázek 6.2) má mnohem vlnitější frekvence nesoucí hercův hlas oproti nahrávce smutku (obrázek 6.3). Proto jsem do vektoru příznaků přidal průměr a standardní odchylku základní frekvence. Jejich získání zmiňuji v sekci 4.6. Při použití mého algoritmu došlo ke zhoršení přesnosti na trénovací množině ze 75 % na 63 %, ke změně přesnosti na validační množině nedošlo. Při použití algoritmu Probabilistic YIN nedošlo k žádnému výraznému zlepšení ani zhoršení. Průměr ani standardní odchylku jsem proto do vektoru příznaků nepřidával.



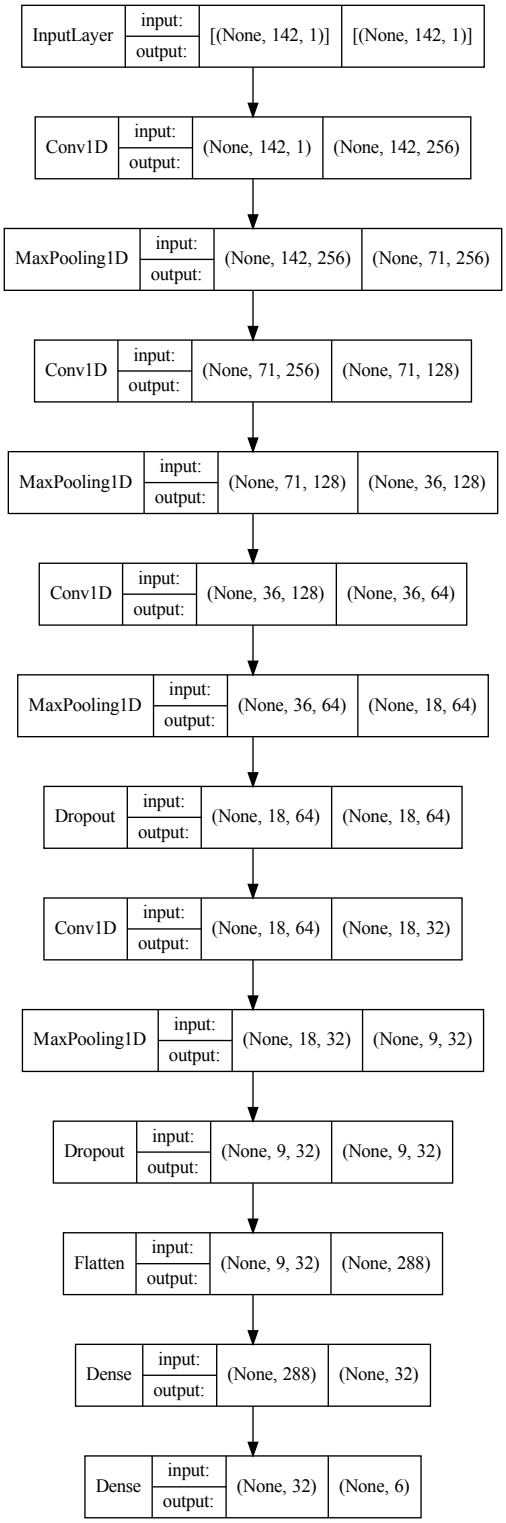
Obrázek 6.2. Graf nahrávky hněvu z datasetu RAVDESS [22] v časové oblasti a její spektrogram



Obrázek 6.3. Graf nahrávky smutku z datasetu RAVDESS [22] v časové oblasti a její spektrogram

Po nastavení parametrů příznaků jsem přistoupil k nastavování augmentace. Původně jsem experimentoval se všemi čtyřmi metodami z kapitoly 3.4. Když jsem si poslechl augmentované nahrávky, zjistil jsem, že při posunu nahrávky ve frekvenční oblasti vznikne v nahrávce echo, které nahrávku značně změní. Podobné echo se vyskytovalo i u augmentace se změnou rychlostí nahrávky. Knihovna LibROSA totiž realizuje tyto dvě operace na spektrogramu nahrávky. Nejdříve tedy musí Fourierovými transformacemi získat spektrogram, poté provede požadovaný augmentační efekt a nakonec převede spektrogram zpět do časové oblasti. Problém je, že se při tomto procesu nemusí zachovat fáze. Fázovým posuvem poté vznikne zmíněná ozvěna. Po přidání augmentace (všech 4 metod) a nastavení nejlepších možných parametrů jsem dosáhl přesnosti 60 % na trénovací množině a 58.5 % na validační množině. Když jsem následně nechal augmentace pouze šumem a posunutím nahrávky v časové oblasti, zlepšila se přesnost trénovací množiny na 71 % a přesnost validační množiny na 67 %.

Následně jsem upravoval pouze architekturu a parametry samotné sítě. Velikost vstupního vektoru neuronové sítě je 142 (skládá se z průměrů Zero Crossing Rate, Chroma, RMS a Mel spektrogramu). Nejlepších výsledků jsem dosáhl použitím 4 bloků 1D konvoluce (kernel o velikosti 5 a krok 1) a MaxPooling (kernel o velikost 5 a krok 2). Poté jsou data zploštěna a následují dvě Fully Connected vrstvy. Síť obsahuje 226 246 učících se parametrů. Architektura je zobrazena na diagramu 6.4. Každý blok v obrázku představuje jednu z vrstev, v bloku je uvedený druh vrstvy, vstupní a výstupní rozměr tenzoru. Rozměry tenzoru jsou ve formátu (None, a, b) - číslo a udává velikost vektoru příznaků, číslo b udává počet filtrů. Například na vstupu máme vektor velikosti 142 a je pouze jeden (jeden filtr). Výstupem první konvoluční vrstvy je vektor stejné délky, ale ve více variantách, jelikož na vstup bylo aplikováno 256 filtrů.



Obrázek 6.4. Architektura dosahující nejlepších výsledků při trénování sítě na několika příznacích

V celé síti jsem aktivační funkci ReLU, kromě výstupní vrstvy, kde je použitý Softmax. Ztrátová funkce je kategorická křížová entropie a optimalizátor RMSProp. Trénování sítě proběhlo na 250 epochách s velikostí batche 64 vzorků.

Při tomto nastavení jsem dosáhl přesnosti 88.23 % na validačním datasetu. Průběh tohoto trénování sítě je vyobrazen na obrázku 6.5.

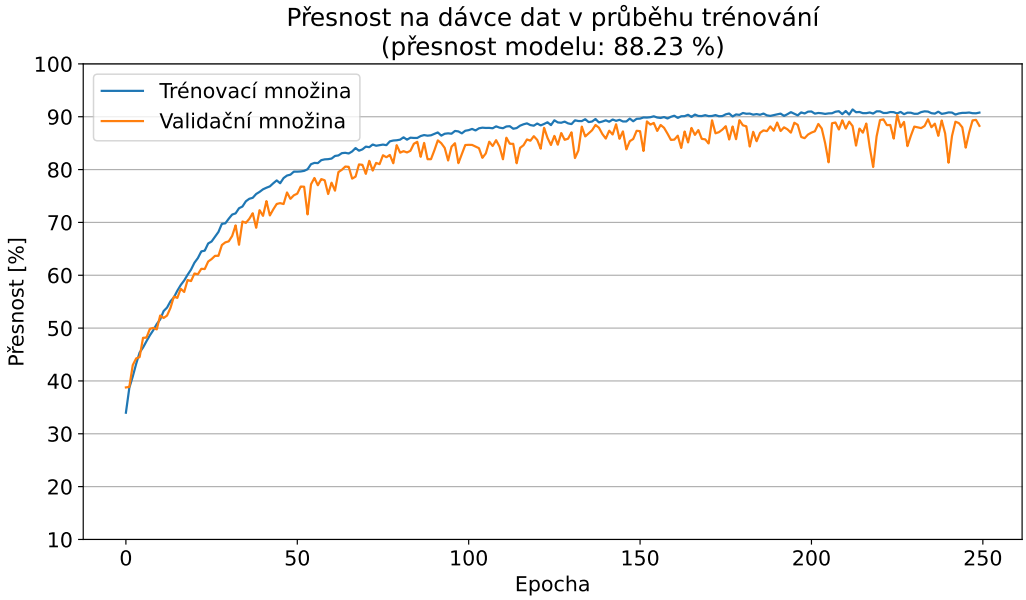
K vyjádření přesnosti a vlastností natrénované sítě se používá i tzv. confusion matrix. Confusion matrix je matice obsahující v řádcích anotované emoce, ve sloupcích obsahuje predikce emocí. V práci používám normalizovanou matici v řádcích. Číslo v řádku r a sloupci s vyjadřuje v kolika procentech síť klasifikovala nahrávku k emoci ve sloupci s, přestože nahrávka byla anotovaná emocí v řádku r. Confusion matrix natrénované sítě se nachází na obrázku 6.6.

Takto natrénovaná síť (architektura a její váhy) byla uložena do složky `model`. Model je možné opět načíst a použít pro vytváření predikcí. Natrénovanou neuronovou síť bude potřeba načítat v demonstrační aplikaci, kterou implementuji později.

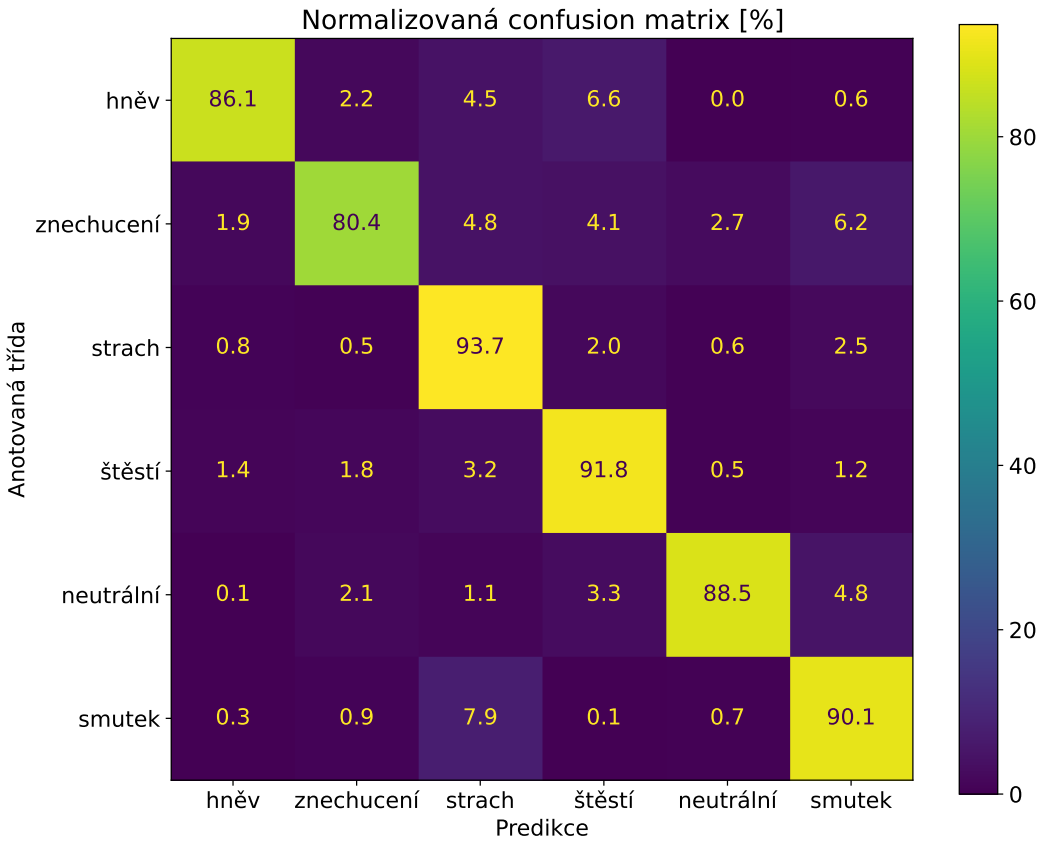
Testování sítě jsem provedl následujícím způsobem. Nejdříve jsem vygeneroval soubory s příznaky pro sjednocené datasety, ze kterých byla odstraněna věta „I wonder what this is about“, která je obsažena v datasetu CREMA-D - cíleně jsem odebral větu z největšího datasetu. Takto byla vytvořená nová trénovací množina. Následně jsem vygeneroval příznaky naopak pouze s větou „I wonder what this is about“, tato data budou sloužit jako nová validační množina. Poté byla uvedená architektura znovu natrénována a validována na tomto novém složení datasetů. Učení tedy probíhalo na datasetu, ze kterého byla odstraněna jedna věta. Jelikož jsem použil shodné parametry, tak jsem po prvním běhu zjistil, že trénování na 250 epochách je příliš mnoho a dochází k přeučení. Proto jsem učení zkrátil pouze na 10 epoch, protože v 10. epoše se nacházelo maximum přenosti sítě. Při druhém pokusu jsem dosáhl přesnosti klasifikace 48.83 % na oné chybějící větě. Graf přesnosti v průběhu učení se nachází na obrázku 6.7 a confusion matrix je na obrázku 6.8. Z matice je patrné, že model má problém při klasifikaci do třídy znechucení a do třídy strachu. Na druhou stranu často klasifikuje emoci jako neutrální, smutek nebo štěstí.

Model naučený při testování sítě byl použitý i pro testování sítě na zašuměném vstupu. Byla tedy ověřena přesnost klasifikace tohoto modelu na nahrávkách věty „I wonder what this is about“, ke kterým byl přičten bílý šum. Intenzita bílého šumu byla konstantní a zhruba poloviční, oproti maximální intenzitě šumu používaného pro augmentaci. Intenzita šumu v augmentaci však byla vynásobena náhodným číslem z rovnoměrného rozdělení (0, 1) - tedy byla pro každou nahrávku unikátní. Velikost šumu použitého v testování jsem poslechem ověřil a usoudil, že zašumění je adekvátní šumu z běžných vysílaček. Na zašuměné větě dosáhl model mírně lepší přesnosti 49.41 %, než na větě nezašuměné. To však není překvapivé, jelikož se síť učila na augmentovaných datech, kde intenzita šumu nebyla konstatní a amplituda šumu byla až dvakrát větší.

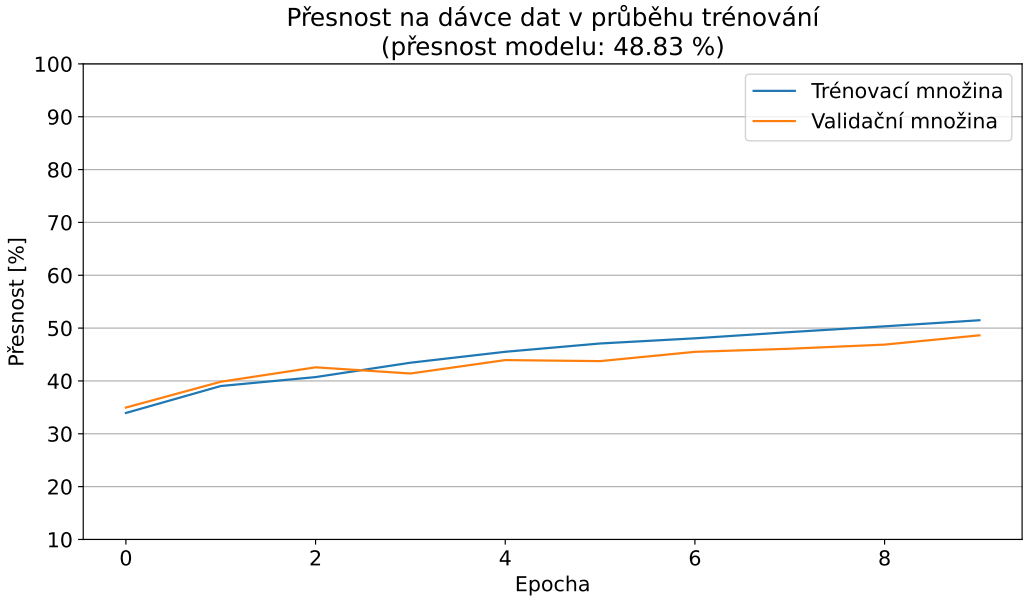
Natrénovaný model, který vznikl při testování sítě, jsem uložil do složky `model_test`, pro jeho možné budoucí použití. Předchozí uložený model dosahuje vynikajících výsledků, ale pouze na větách, které jsou v datasetech obsaženy. Model uložený nyní je ale optimalizovaný pro klasifikaci vět, které nejsou součástí trénovacího datasetu (jelikož bylo trénování ukončeno v momentě, kdy dosáhl nejlepší přesnosti při klasifikaci věty, která byla z trénovací množiny vyloučena).



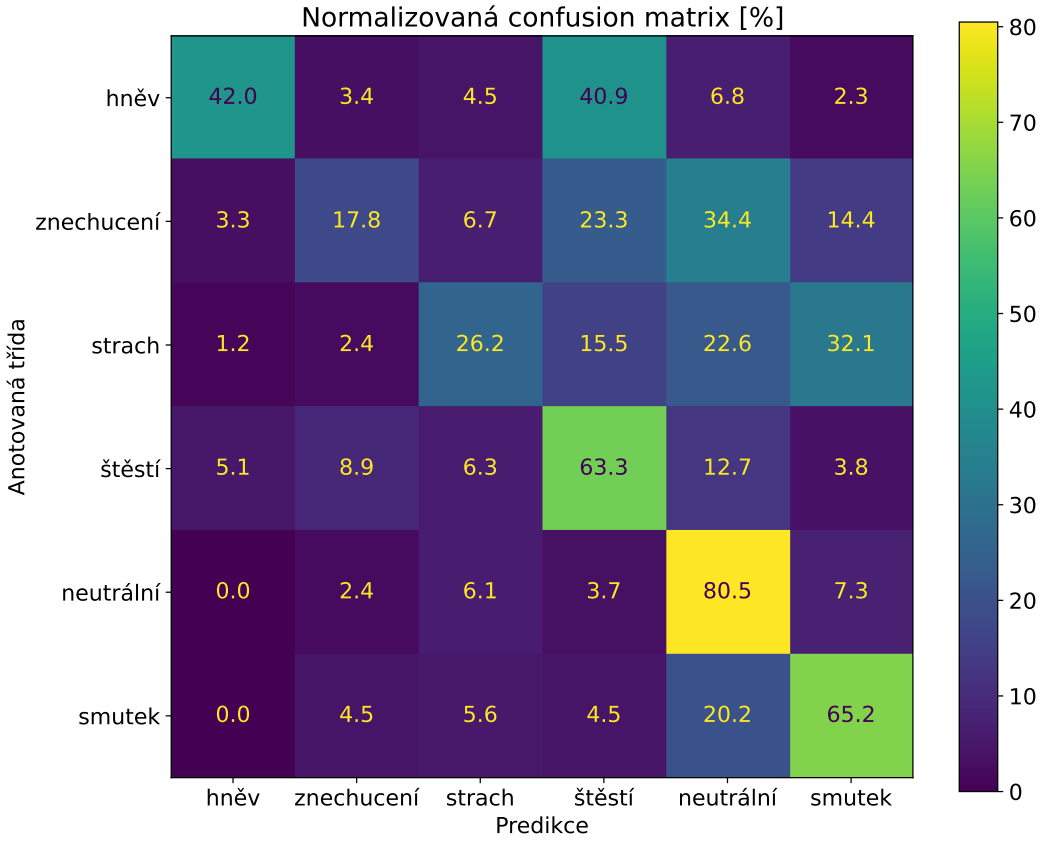
Obrázek 6.5. Graf přesnosti sítě na trénovací a validační množině v průběhu učení sítě na 250 epochách



Obrázek 6.6. Confusion matrix natrénované sítě na několika druzích příznaků



Obrázek 6.7. Graf přesnosti sítě na trénovací (s odstraněnou větou) a validační (odstraněná věta) množině v průběhu učení sítě na 10 epochách



Obrázek 6.8. Confusion matrix při testování sítě

Kapitola 7
Síť trénovaná na spektrogramu

Přesnost natrénované neuronové sítě v předchozí kapitole byla velmi závislá na výběru extrahovaných příznaků a je možné, že sama extrakce příznaků mohla vést ke ztrátě důležitých dat. Proto jsem se rozhodl, že vyzkouším jiný přístup řešení problému rozpoznávání emocí. Pro další postup jsem se inspiroval článkem [30], který přímo zmiňuje citlivost výběru příznaků při přípravě dat před vstupem do neuronové sítě. Aby se tato citlivost eliminovala, autoři článku nechali neuronovou síť naučit se příznaky přímo z celého spektrogramu, který byl vhodně obarvený. Upravili předtrénovanou síť AlexNet, která byla navržena pro klasifikaci obrázků do 1000 tříd (mnoho druhů zvířat nebo všedních předmětů). Autoři doučili AlexNet rozpoznávání emocí (resp. obrázků spektrogramů) do 7 tříd (dataset EmoDB). Nejlepší výsledky byly dosaženy použitím Mel spektrogramů. Takto dosáhli přesnosti až 80.5 %. Autoři také uvádí experiment, při kterém snížili vzorkovací frekvenci použitých nahrávek z 16 kHz na 8 kHz. Což zhoršilo přesnost klasifikace pouze o 3,3 %.

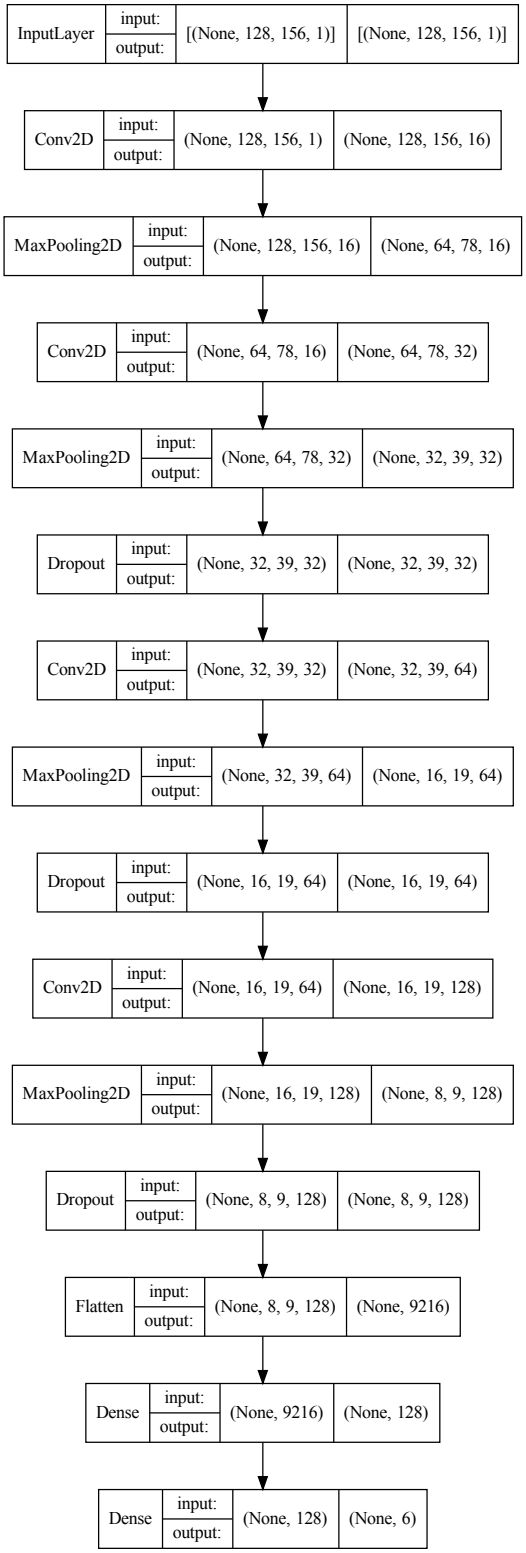
Článek [31] používá neuronovou síť složenou z 2 bloků 2D konvoluce - MaxPooling a následně 2 Fully connected vrstev. Nahrávku nejdříve autoři převedou na logaritmický výkonový spektrogram, který následně normalizují metodou PCA whitening. Vstupem do sítě je normalizovaný spektrogram s rozměry 60x40. Autoři experimentovali s datasetem IEMOCAP (5 tříd) [32] a popsaným způsobem dosáhli přesnosti 40 %. Když autoři sestavili vektor příznaků, podobně jako jsem sestavil vektor v předchozí kapitole, dosáhli pouze 37.6 % přesnosti.

Autoři v článku [33] vygenerovali spektrogramy o rozměrech 128x128 a použili architekturu složenou pouze ze 7 konvolučních vrstev a 2 Fully connected vrstev. Takto dosáhli přesnosti až 79 % na datasetu RAVDESS. Z nahrávek byl odstraněn šum a neznělé úseky.

V článku [34] je použita pro klasifikaci emocí z hlasu architektura s bloky 2D konvoluce, MaxPool, BatchNorm a nakonec 3 Fully connected vrstvy. Vstupem je spektrogram o rozměrech 64x64. Na datasetu EmoDB bylo dosaženo přesnosti 93 %.

Rozhodl jsem se tedy, že použiji pro vstup do neuronové sítě Mel spektrogramy, jelikož z uvedené literatury dosahují nejlepších výsledků. Obdobně jako v předchozí kapitole jsem vygeneroval spektrogramy pro originální a augmentované nahrávky rozstříhané na délku 1.8 s. Ke generování spektrogramů jsem použil délku okna 512 vzorků, z nichž se počítala Fourierova transformace. Tato hodnota je doporučená v dokumentaci knihovny pro aplikace zpracovávající lidský hlas. Výsledný Mel spektrogram (vstup do neuronové sítě) má tedy rozměry 128x156. Spektrogram jsem neobarvoval, lze tedy na něj pohlížet jako na obrázek s jedním kanálem.

Po vygenerování příznaků jsem začal experimentovat s architekturou neuronové sítě. Architektury ve zmíněné literatuře jsou velmi podobné architekturám obecně známým, které se používají pro rozpoznávání jednoduchých obrázků. Proto jsem začal architekturou s bloky 2D konvoluce, MaxPooling, Dropout a 4 Fully connected vrstvy.



Obrázek 7.1. Architektura dosahující nejlepších výsledků při trénování sítě na spektrogramu

Postupně jsem zkoušel používat různý počet bloků s konvolucemi, různé počty filtrů, různé hodnoty dropoutu, apod. Nejlepších výsledků jsem dosáhl se čtyřmi bloky: konvoluce s jádrem 3x3, MaxPool s jádrem 2x2, Dropout 10 %. Po blocích následuje zploštění a 2 Fully connected vrstvy. Architektura je zobrazená na obrázku 7.1.

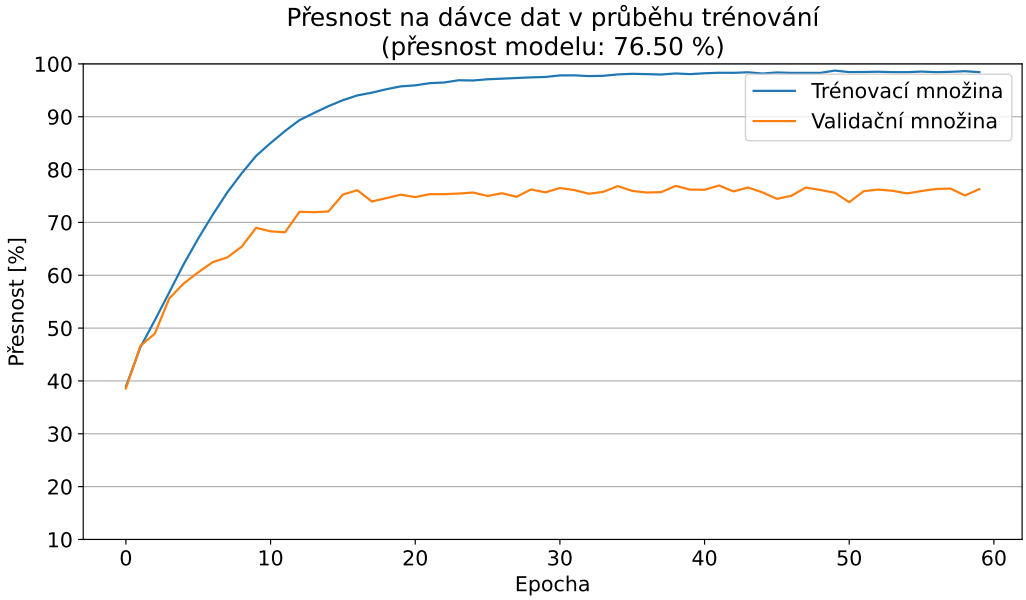
Aktivační funkci jsem použil vždy ReLU, kromě výstupní vrstvy, kde je Softmax. Což je totožné jako v předchozí kapitole. Ztrátová funkce je opět kategorická křížová entropie a optimalizátor je RMSProp. Liší se pouze vstupní vektor a architektura samotné sítě. Síť obsahuje 1 277 702 učících se parametrů. Trénování sítě proběhlo na 60 epochách s velikostí batche 64 vzorků.

Při tomto nastavení jsem zjistil, že nejlepších výsledků dosáhne síť při použití amplitudových Mel spektrogramů, které nejsou převedené do dB. Dosáhl jsem přesnosti 76.50 % na validačním datasetu (tedy při rozdělení mixu RAVDESS, CREMA-D a EmoDB v poměru 80:20 na trénovací a validační množinu). Natrénovaný model byl uložen do složky `./src2/model` pro použití v demonstrační aplikaci. Průběh trénování sítě a normalizovaná confusion matrix se nachází na obrázcích 7.2 a 7.3.

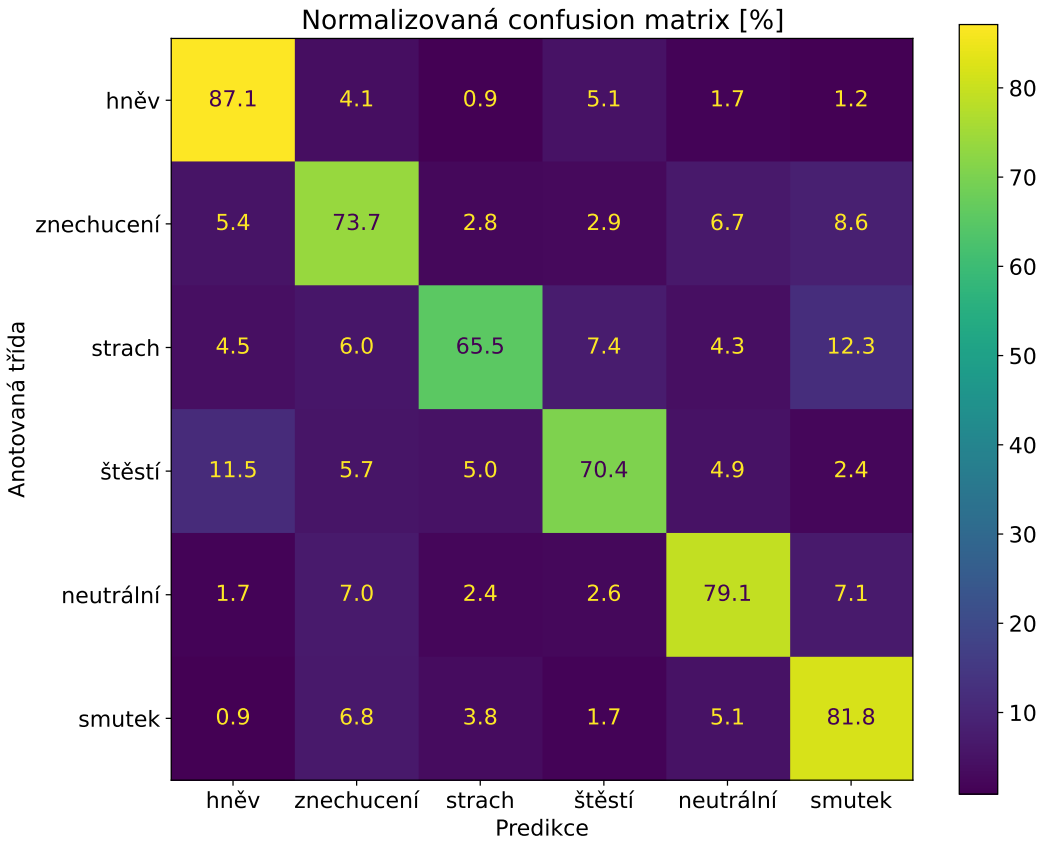
Testování sítě proběhlo stejným způsobem jaký byl popsán v předchozí kapitole - odstraněním věty „I wonder what this is about“, novým natrénováním sítě a následnou validací na této větě. Trénování sítě proběhlo opět pouze na 10 epochách, jelikož větší počet epoch způsobil přetrénování sítě. Takto jsem dosáhl přesnosti 52.54 %. Natrénovaný model byl uložen do složky `./src2/model_test` pro pozdější produkční použití. Graf přesnosti při testování sítě a confusion matrix testování na nezašuměné větě se nachází na obrázcích 7.4 a 7.5. Z matice je patrné, že síť opět upřednostňuje některé predikce, tentokrát do třídy strachu a smutku. Stejně jako předchozí architektura má i tato síť problém s klasifikací hněvu a znechucení oproti ostatním třídám.

Přesnost jsem opět otestoval i na zašuměné větě a dosáhl jsem 51.17 %. Ani zde nebyl výrazný rozdíl v přesnosti na zašuměném a nezašuměném vstupu.

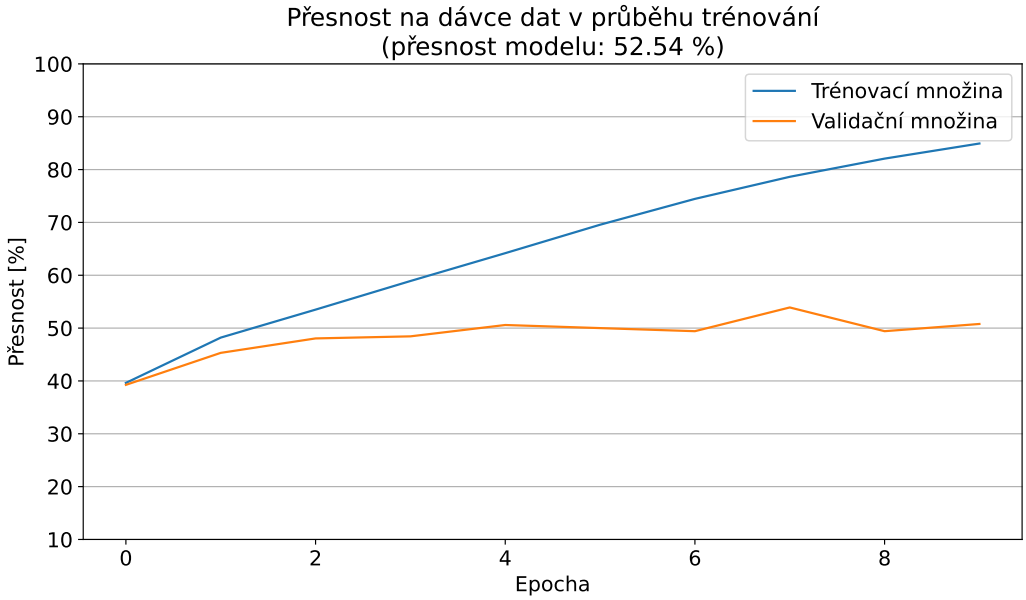
Po dokončení experimentů s neuronovými sítěmi jsem vytvořil demonstrační aplikaci. Po spuštění skriptu `demo.py` lze vybrat jeden ze 4 natrénovaných modelů. Skript následně požadovaný model načte a poté ve smyčce nahrává zvukové úseky o délce 1.8 s z mikrofonu, ze kterých vygeneruje vstupní vektor příznaků. Prostřednictví naučeného modelu skript vytvoří predikci, kterou vypíše na standardní výstup.



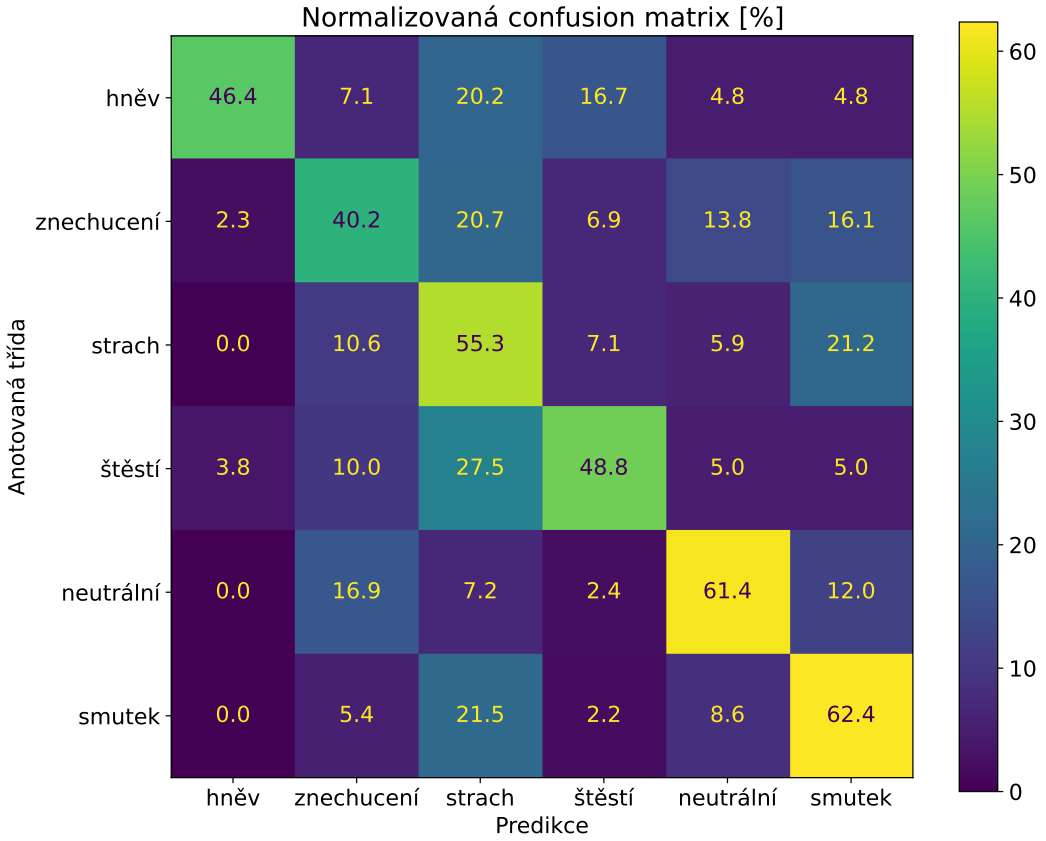
Obrázek 7.2. Graf přesnosti sítě na trénovací a validační množině v průběhu učení sítě na 60 epochách



Obrázek 7.3. Confusion matrix natrénované sítě na spektrogramu



Obrázek 7.4. Graf přesnosti sítě na trénovací (s odstraněnou větou) a validační (odstraněná věta) množině v průběhu učení sítě na 10 epochách



Obrázek 7.5. Confusion matrix při testování sítě

Kapitola 8
Diskuze

V této kapitole nejdříve shrnu mnou dosažené výsledky při klasifikování projevované emoce z hlasu a poté uvedu srovnání s výsledky ostatních, současných řešení klasifikace emocí z hlasu.

Tabulka 8.1 obsahuje dosažené přesnosti mnou navržených neuronových sítí v závislosti na použité metodě. Klasifikaci jsem řešil dvěma nezávislými metodami, které jsou zmíněné v předchozích kapitolách. Trénování a validace sítě byla provedena po smíšení datasetů RAVDESS, EmoDB a CREMA-D a po rozdělení v poměru 80:20 na trénovací a validační množinu. Následně jsem provedl nové natrénování stejné architektury opět na smíšených datasetech, ze kterých však byla odstraněna jedna věta. Na odstraněné větě bylo provedeno testování sítě. Tato věta byla následně zkreslena bílým šumem a opět bylo provedeno testování sítě se zašuměným vstupem.

Metoda	Dataset	Přesnost sítě [%]
Výběr příznaků	validace	88.23
Výběr příznaků	testování bez šumu	48.83
Výběr příznaků	testování se šumem	49.41
Spektrogram	validace	76.50
Spektrogram	testování bez šumu	52.54
Spektrogram	testování se šumem	51.17

Tabulka 8.1. Dosažené výsledky v závislosti na dvou použitých metodách získání vstupního vektoru konvoluční neuronové sítě.

Článek [4] z roku 2021 zmiňuje přesnosti dosažené různými metodami pro jednotlivé datasety používané pro rozpoznávání emocí z hlasu. Pro dataset CREMA-D uvádí maximální dosaženou přesnost 65.77 %, pro RAVDESS 88.7 % a pro EmoDB až 96.97 %.

Autoři publikace [35] z roku 2021 uvádí výsledky pro všechny tři mnou používané datasety. Pro CREMA-D dosáhli přesnosti 83.56 %, pro RAVDESS přesnosti 93.94 % a pro EmoDB 97.06 %. Je patrné, že od komplexnosti datasetu se odvíjí i dosažené přesnosti. Obdobně publikace [36] z roku 2022 uvádí přesnost 96 % na datasetu RAVDESS, ale pouze 84 % na datasetu CREMA-D.

Již zmíněná publikace [33] z roku 2019 dosáhla přesnosti 79 % na datasetu RAVDESS prostřednictvím podobné architektury, jakou používám při rozpoznávání emocí ze spektrogramu. Autoři provedli ale i testování sítě tak, že natrénovali síť na datasetu IEMOCAP a provedli testování na datasetu RAVDESS. Takto se výsledky sítě zhoršily na přesnost pouze 56.5 %.

Jelikož jsem nenašel publikaci, která zmiňuje smíšení datasetů použitých v této práci, nelze přesně srovnávat dosažené přesnosti s ostatními řešeními. Při smíšení datasetů RAVDESS, CREMA-D a EmoDB můžeme předpokládat, že výsledky neuronové sítě budou horší, než kdyby byla síť trénovaná a testování na každém datasetu

zvlášť. Protože dataset CREMA-D je nejrozsáhlejší z celého použitého mixu, můžeme alespoň srovnat výsledky této práce s výsledky v publikacích uváděných pro dataset CREMA-D. Mé výsledky při validaci na 20 % datasetu u metody, kdy jsem vybral několik extrahovaných příznaků, jsou mírně lepší, než výsledky dosažené v uvedených publikacích. Naopak u metody při použití spektrogramu jsou výsledky validace horší oproti výsledkům v publikacích.

Srovnáme-li výsledky testování neuronové sítě na neznámé větě, tak autoři publikace [33] dosáhli pouze o 4 % lepších výsledků, než jakých bylo dosaženo v této práci. Autoři však k testování zvolili mírně odlišný přístup, než jaký byl použit v předchozích kapitolách.

Pokud srovnáme výsledky testování mezi dvěma použitými metodami v této práci, tak lehce lepších výsledků dosahuje neuronová síť, jestliže je na jejím vstupu celý spektrogram. U obou metod však došlo k razantnímu snížení přesnosti při testování sítě na neznámé větě, než jaké přesnosti bylo dosaženo při validaci sítě, když nebyla z trénovací množiny vyloučena žádná věta.

Za účelem objektivnějšího testování sítě jsem oslovil profesorku Veroniku Makarova, která je autorkou neveřejného datasetu RUSLANA [37], který byl vytvořený v ruském jazyce. Ruský jazyk patří ke skupině slovanských jazyků, stejně jako český jazyk, proto by bylo zajímavé otestovat síť na tomto vytvořeném datasetu. Bohužel se mi tento dataset nepodařilo získat.

Kapitola 9
Závěr

Hlavním cílem této bakalářské práce byla realizace systému, který rozpozná jednu z šesti předdefinovaných základních emocí projevovaných v hlasové nahrávce mluvčího (herce). Pro dosažení tohoto cíle bylo nutné analyzovat používané metody, analyzovat dostupné anotované datasety a navrhnout a implementovat konkrétní klasifikátor. Po návrhu parametrů klasifikátoru bylo dalším cílem klasifikátor otestovat a srovnat s výsledky v současných publikacích. Posledním cílem byla implementace aplikace pro demonstraci systému.

V práci byly analyzovány 3 metody klasifikace: SVM klasifikátor, vícerozměrná lineární regrese a neuronové sítě, které dosahují nejlepších výsledků. Poté byly popsány metody augmentace zvukových nahrávek (umělé rozšíření datasetu) - například zašumění nahrávky. Následně byly popsány vlastnosti (příznaky), které lze extrahovat ze zvuku. Těmi jsou například spektrogram nebo akustický výkon (RMS). Také byly analyzovány 3 dostupné datasety: EmoDB, RAVDESS a CREMA-D.

Vzhledem k různým řešením architektur neuronových sítí pro rozpoznávání emocí, byly v práci popsány a implementovány dvě konkrétní architektury konvoluční neuronové sítě. Nejprve byla navržena a odladěna architektura, jejímž vstupním vektorem je několik vybraných druhů extrahovaných příznaků. Toto řešení bylo velmi citlivé na výběr příznaků, proto je v práci popsána i druhá architektura neuronové sítě, na jejímž vstupu je pouze spektrogram dané nahrávky.

Architektury byly natrénovány na 80 % smíšených dat výše uvedených datasetů. Všechny nahrávky byly podvzorkovány na frekvenci 11 kHz. Validace sítí byla provedena na zbylých 20 % smíšených dat. Výsledky validace první zmíněné architektury jsou srovnatelné s výsledky uváděné v soudobé literatuře. Druhá architektura dosahuje výsledků relativně horších. Modely natrénovaných neuronových sítí byly uloženy a byl vytvořen demonstrační skript, který požadovaný model načte, ve smyčce nahrává zvuk z mikrofonu počítače a na pozadí v reálném čase klasifikuje zaznamenaný zvuk.

Obě architektury byly otestovány na větě, která nebyla součástí trénovací množiny. Dále byly otestovány na zašuměné větě (simulace VHF/UHF vysílaček). Výsledky architektur v obou těchto testech byly podobné, ale byly razantně horší, než výsledky získané při validaci. Přesto literatura uvádí při obdobném testování sítě zhruba stejné poklesy přesnosti.

Stanovené cíle práce tedy byly dosaženy. Přesnost systému na větách, které byly v trénovací množině, je i přes poměrně jednoduchou architekturu sítě vynikající. Literatura nejčastěji provádí validování klasifikátoru na stejných datasetech, proto je procento přesnosti poměrně vysoké. V budoucím výzkumu by ale bylo žádoucí se zaměřit na testování klasifikátorů a na jejich přesnosti dosahované na produkčních datech. Bylo by vhodné ověřit, co zapříčiňuje zhoršení přesnosti při testování sítě. Zda prudce klesá přesnost pouze na neznámé větě nebo má vliv na přesnost i herec, který nebyl součástí trénovacího datasetu. Nebo jaký vliv má jazyk mluvčího. Poté by bylo možné klasifikátor upravit tak, aby zhoršení přesnosti nebylo tak značné.

Literatura

[1] Paul Ekman a Wallace V. Friesen. Constants across cultures in the face and emotion. *Journal of Personality and Social Psychology*. 1971, 17 124-129. DOI 10.1037/H0030377.

[2] Paul Ekman. An Argument for Basic Emotions. *Cognition and Emotion*. 1992, 6 169-200. DOI 10.1080/02699939208411068.

[3] Alif Bin, Abdul Qayyum, Asiful Arefeen a Celia Shahnaz. Convolutional Neural Network (CNN) Based Speech-Emotion Recognition. DOI 10.1109/SPICSCON48833.2019.9065172.

[4] Babak Joze Abbaschian, Daniel Sierra-Sosa a Adel Elmaghraby. Deep learning techniques for speech emotion recognition, from databases to models. *Sensors (Switzerland)*. 2021, 21 1-27. DOI 10.3390/S21041249.

[5] Leila Kerkeni, Youssef Serrestou, Mohamed Mbarki, Kosai Raoof a Mohamed Ali Mahjoub. Speech Emotion Recognition: Methods and Cases Study. 2018, DOI 10.5220/0006611601750182.

[6] Björn Schuller, Bogdan Vlasenko, Florian Eyben, Gerhard Rigoll a Andreas Wendemuth. Acoustic emotion recognition: A benchmark comparison of performances. *Proceedings of the 2009 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2009*. 2009, 552-557. DOI 10.1109/ASRU.2009.5372886.

[7] M L Dhore a India Pallavi Yesaware. Speech Emotion Recognition Using Support Vector Machines Speech Emotion Recognition Using Support Vector Machine Yashpalsing Chavhan Student VIT, Pune India. *Article in International Journal of Computer Applications*. 2010, 1 975-8887. DOI 10.1007/978-3-642-21402-8_35.

[8] *Support Vector Machine (SVM). The support vector machine is one of… | by Rishabh Jain | Medium.* `https://medium.com/@rishabhjain9440/support-vector-machine-svm-f23cece26419`.

[9] Sanaul Haq a Philip J.B. Jackson. Multimodal emotion recognition. *Machine Audition: Principles, Algorithms and Systems*. 2010, 398-423. DOI 10.4018/978-1-61520-919-4.CH017.

[10] F. Burkhardt, A. Paeschke, M. Rolfes, W. Sendlmeier a B. Weiss. A database of German emotional speech. *9th European Conference on Speech Communication and Technology*. 2005, 1517-1520. DOI 10.21437/INTERSPEECH.2005-446.

[11] Imran Naseem, Roberto Togneri a Mohammed Bennamoun. Linear regression for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2010, 32 2106-2112. DOI 10.1109/TPAMI.2010.128.

[12] Francois Chollet. *Deep Learning with Python*. 1st vyd.. USA: Manning Publications Co., 2017. ISBN 1617294438.

[13] Gil Keren a Björn Schuller. Convolutional RNN: an Enhanced Model for Extracting Features from Sequential Data.

[14] Srinivas Parthasarathy a Ivan Tashev. CONVOLUTIONAL NEURAL NETWORK TECHNIQUES FOR SPEECH EMOTION RECOGNITION.

[15] Sebastian Ruder. An overview of gradient descent optimization algorithms. 2016, DOI 10.48550/arxiv.1609.04747.

[16] Shengyun Wei, Shun Zou, Feifan Liao a Weimin Lang. A Comparison on Data Augmentation Methods Based on Deep Learning for Audio Classification. *Journal of Physics: Conference Series*. 2020, 1453 DOI 10.1088/1742-6596/1453/1/012085.

[17] K Tarunika, R B Pradeeba a P Aruna. Applying Machine Learning Techniques for Speech Emotion Recognition. 2018, DOI 10.1109/ICCCNT.2018.8494104.

[18] Thapanee Seehapoch a Sartra Wongthanavasu. Speech emotion recognition using support vector machines. *Proceedings of the 2013 5th International Conference on Knowledge and Smart Technology, KST 2013*. 2013, 86-91. DOI 10.1109/KST.2013.6512793.

[19] Anusha Koduru, Hima Bindu Valiveti a Anil Kumar Budati. Feature extraction algorithms to improve the speech emotion recognition rate. *International Journal of Speech Technology*. 2020, 23 45-55. DOI 10.1007/S10772-020-09672-4.

[20] Shivam Burnwal. *Speech Emotion Recognition | Kaggle*. `https://www.kaggle.com/code/shivamburnwal/speech-emotion-recognition/notebook`.

[21] Aisultan Shoiynbek, Kanat Kozhakhmet, Nazerke Sultanova a Rakhima Zhumaliyeva. The Robust Spectral Audio Features for Speech Emotion Recognition. *Appl. Math. Inf. Sci.* 2019, 13 867-870. DOI 10.18576/amis/130521.

[22] Steven R. Livingstone a Frank A. Russo. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PLOS ONE*. 2018, 13 e0196391. DOI 10.1371/JOURNAL.PONE.0196391.

[23] Steven B. Davis a Paul Mermelstein. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 1980, 28 357-366. DOI 10.1109/TASSP.1980.1163420.

[24] Matthias Mauch a Simon Dixon. PYIN: A fundamental frequency estimator using probabilistic threshold distributions. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. 2014, 659-663. DOI 10.1109/ICASSP.2014.6853678.

[25] Houwei Cao, David G. Cooper, Michael K. Keutmann, Ruben C. Gur, Ani Nenkova a Ragini Verma. CREMA-D: Crowd-sourced Emotional Multimodal Actors Dataset. *IEEE transactions on affective computing*. 2014, 5 377. DOI 10.1109/TAFFC.2014.2336244.

[26] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg a Oriol Nieto. *librosa: Audio and music signal analysis in python.* In: *Proceedings of the 14th python in science conference*. 2015.

[27] Francois Chollet a others. *Keras*. 2015. `https://github.com/fchollet/keras`.

[28] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, Xiaoqiang Zheng a Google Research. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 2016, DOI 10.48550/arxiv.1603.04467.

[29] Harini Murugan. Speech Emotion Recognition Using CNN. *International Journal of Psychosocial Rehabilitation*. 2020, 24 DOI 10.37200/IJPR/V24I8/PR280260.

[30] Margaret Lech, Melissa Stolar, Christopher Best a Robert Bolia. Real-Time Speech Emotion Recognition Using a Pre-trained Image Classification Network: Effects of Bandwidth Reduction and Companding. *Frontiers in Computer Science*. 2020, 2 14. DOI 10.3389/fcomp.2020.00014.

[31] W. Q. Zheng, J. S. Yu a Y. X. Zou. An experimental study of speech emotion recognition based on deep convolutional neural networks. *2015 International Conference on Affective Computing and Intelligent Interaction, ACII 2015*. 2015, 827-831. DOI 10.1109/ACII.2015.7344669.

[32] Carlos Busso, Murtaza Bulut, Chi Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N. Chang, Sungbok Lee a Shrikanth S. Narayanan. IEMOCAP: Interactive emotional dyadic motion capture database. *Language Resources and Evaluation*. 2008, 42 335-359. DOI 10.1007/S10579-008-9076-6.

[33] Mustaqeem a Soonil Kwon. A CNN-Assisted Enhanced Audio Signal Processing for Speech Emotion Recognition. *Sensors 2020, Vol. 20, Page 183*. 2019, 20 183. DOI 10.3390/S20010183.

[34] Tursunov Anvarjon, Mustaqeem a Soonil Kwon. Deep-Net: A Lightweight CNN-Based Speech Emotion Recognition System Using Deep Frequency Features. *Sensors (Basel, Switzerland)*. 2020, 20 1-16. DOI 10.3390/S20185212.

[35] Y. Bhanusree, T. Vishnu Vardhan Reddy a S. Karthik Rao. Capsule Networks based Acoustic Emotion Recognition using Mel Cepstral Features. *Proceedings of the 2021 IEEE International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems, ICSES 2021*. 2021, DOI 10.1109/ICSES52305.2021.9633952.

[36] M. Gokilavani, Harshith Katakam, SK Abdul Basheer a PVVS Srinivas. Ravdness, Crema-D, Tess Based Algorithm for Emotion Recognition Using Speech. 2022, 1625-1631. DOI 10.1109/ICSSIT53264.2022.9716313.

[37] Veronika Makarova a Valery A. Petrushin. *RUSLANA: a database of Russian emotional utterances.* In: *Proc. 7th International Conference on Spoken Language Processing (ICSLP 2002)*. 2002. 2041–2044.