VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

REDUKCE ŠUMU V AUDIOSIGNÁLU POMOCÍ DEEP LEARNING
AUDIO SIGNAL DENOISING USING DEEP LEARNING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Tomáš Pacal**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Ondřej Mokrý**
ADVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Audio inženýrství**
specializace Zvuková produkce a nahrávání
Ústav telekomunikací

Student: Tomáš Pacal ***ID:*** 229948
Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Redukce šumu ve zvukovém signálu pomocí hlubokého učení

POKYNY PRO VYPRACOVÁNÍ:

Náplní práce je seznámit se s modely znehodnocení zvukového signálu šumem a s vybranými metodami pro jeho redukci (tzv. audio denoising, noise reduction). Jednat se bude zejména o metody založené na hlubokém učení, konkrétně neuronové sítě. Cílem práce bude vhodnou síť implementovat ve zvoleném prostředí, provést její učení a otestovat její schopnost potlačení rušení na simulovaných i reálných nahrávkách řeči a hudby. Výstupem testování bude jak subjektivní hodnocení kvality výsledného signálu, tak posouzení pomocí objektivních metrik (PEAQ, PEMO-Q). Výsledky budou porovnány s jinými používanými metodami pro potlačení rušení, například metodami založenými na řídkosti spektra.

DOPORUČENÁ LITERATURA:

[1] LECUN, Yann, Yoshua BENGIO a Geoffrey HINTON. Deep learning. Nature. 2015. Dostupné z: http://www.nature.com/articles/nature14539

[2] XU, Yong, Jun DU, Li-Rong DAI a Chin-Hui LEE. A Regression Approach to Speech Enhancement Based on Deep Neural Networks. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2015. Dostupné z: https://ieeexplore.ieee.org/document/6932438/

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: Ing. Ondřej Mokrý

doc. Ing. Jiří Schimmel, Ph.D.
předseda rady studijního programu

ABSTRAKT

Tato práce se zabývá odstraňováním šumu v audio signálu za použití hlubokého učení. Jsou zde popsány základní typy neuronových sítí a jejich využití v této oblasti. Možnosti implementace neuronových sítí jsou otestovány v prostředí Matlab a Python. Následně je navržen model neuronové konvoluční sítě, podle kterého jsou realizovány čtyři různé architektury konvolučních sítí, které byly poté trénovány a testovány na různých typech šumů. Na základě těchto testů byla vybrána jedna architektura, která byla společně s metodou redukce šumu využívající vlnkové transformace podrobena srovnávacímu testu na nahrávce řeči a poté na hudební nahrávce. Výsledky jsou vyhodnoceny jak pomocí objektivních metrik kvality zvuku tak pomocí neformálního poslechového testu. Neuronová síť dosáhla lepších výsledků dle všech použitých metrik a v poslechovém testu.

KLÍČOVÁ SLOVA

hluboké učení, konvoluce, neuronová síť, strojové učení, Matlab, Python, PESQ, PEMO-Q, STOI, redukce šumu, zpracování zvukového signálu

ABSTRACT

This thesis deals with noise removal in audio signal using deep learning. The basic types of neural networks and their use in audio signal processing are described. The possibilities of implementing neural networks are tested in Matlab and Python. Subsequently, a convolutional neural network model is proposed, according to which four different convolutional network architectures are implemented and then trained and tested on different types of noise. Based on these tests, one architecture was selected and subjected to a comparative test on a speech recording and then on a music recording, together with a noise reduction method using wavelet transform. The results are evaluated using both objective sound quality metrics and an informal listening test. The neural network achieved better results according to all the metrics used as well as in the listening test.

KEYWORDS

deep learning, convolution, neural network, machine learning, Matlab, Python, PESQ, PEMO-Q, STOI, noise reduction, audio signal processing

PACAL, Tomáš. *Redukce šumu v audiosignálu pomocí deep learning*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023, 67 s. Bakalářská práce. Vedoucí práce: Ing. Ondřej Mokrý

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Tomáš Pacal
VUT ID autora:	229948
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Redukce šumu v audiosignálu pomocí deep learning

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Ondřeji Mokrému, za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Úvod

Se šumem se ve zvukové i vizuální formě setkáváme na denní bázi a jeho potlačení zvyšuje kvalitu i subjektivní lidský dojem, ať už se jedná o akustický šum v podobě hluku okolních aut, davu lidí, zpěvu ptactva nebo šum elektronický vzniklý při zpracovávání signálu. S přibývajícím nežádoucím šumem vyvstává potřeba jeho redukce. V této práci jsou používány různé druhy barevných šumů a příklad šumu z reálného prostředí. Některé metody redukce šumu jsou uvedeny v kapitole 2.1, mezi které se řadí i metody využívající hlubokého učení (angl. *deep learning*), jenž je součástí strojového učení.

První kapitola této bakalářské práce se zabývá umělými neuronovými sítěmi z teoretického hlediska a obsahuje základní popis rozdělení včetně různých druhů algoritmů učení. Je zde zahrnuta podkapitola 1.5 pojednávající o konvoluční neuronové síti, která je používána při odstraňování šumu. Samotnému využití hlubokého učení v audio signálech je věnována celá podkapitola, kde jsou vyjmenovány a stručně popsány různé praktické příklady, jako je převod textu na řeč a obráceně, nebo umělá inteligence se schopností komponovat vážnou hudbu.

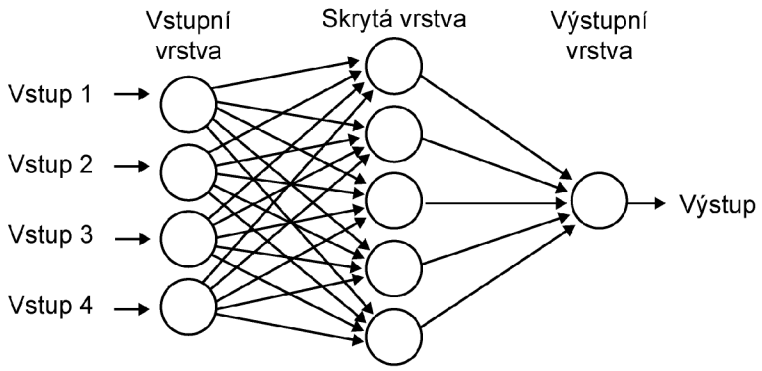
Aby mohl být zvukový signál využit pro tyto aplikace, je nutné ho předzpracovat. Mezi tyto procesy patří Fourierova transformace, která převádí signál z časové oblasti do kmitočtové. Dále je zde popsána segmentace, která patří mezi metody krátkodobé analýzy.

V rámci praktické části je nejprve navržena demonstrační síť pro redukci šumu v obrázcích za účelem srovnání dvou implementačních prostředí, konkrétně tedy Matlab a Python. Pro testování redukce šumu je využit dataset s obrázky, ke kterým je přidán šum a následně filtrován. Průběh i dosažené výsledky jsou vzájemně porovnány pro vhodnou volbu prostředí i parametrů pro následný vývoj konvoluční neuronové sítě pro potlačení šumu ve zvukových signálech. Dále je popsán návrh architektury a její optimalizace. Jsou zde představeny některé datasety vytvořené za účelem trénování sítí, jeden z nich byl vybrán pro tuto práci.

Poslední kapitola zahrnuje výsledky testů navržených konvolučních neuronových sítí na vybraných nahrávkách řeči. Ze čtyř testovaných architektur je poté vybrána jedna, která je porovnána s metodou zabývající se redukcí šumu pomocí vlnkové transformace. Obě metody jsou testovány na vlastní nahrávce řeči a hudby.

1 Neuronové sítě

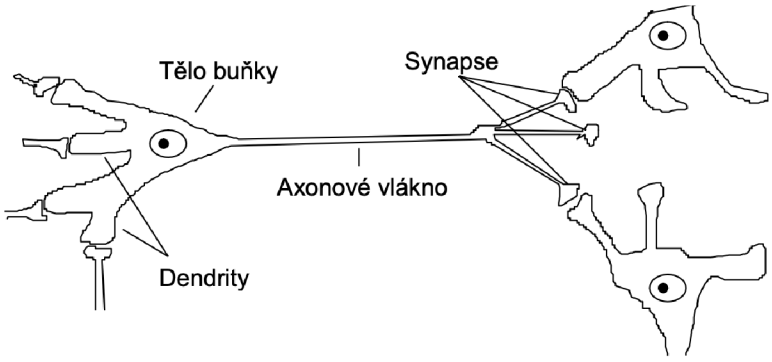
Umělé neuronové sítě (ANN z anglického *Artificial neural networks*) jsou složené z formálních neuronů a ty jsou navzájem propojené takovým způsobem, že výstup jednoho neuronu je vstupem jednoho i více jiných neuronů. Jejich počet a propojení v dané síti určuje její topologii. Tyto neurony rozdělujeme na vstupní, skryté a výstupní, jak je uvedeno na obr. 1.1. Jednotlivé informace v síti jsou šířeny a následně zpracovávány pomocí změn stavů neuronů, nacházejících se ve skrytých vrstvách. Konfigurace neuronové sítě je představena synaptickými váhami spojů mezi neurony a stav celé neuronové sítě je dán stavy všech neuronů. Síť se během učení mění a vyvíjí, ve smyslu adaptace vah a změn stavu jednotlivých neuronů. Architektura (topologie) sítě však zůstává stejná. Celková dynamika sítě je poté určena těmito změnami. Tu si můžeme rozdělit na dynamiku *organizační*, dynamiku *aktivní* a dynamiku *adaptivní*. Tyto dynamiky sítě se řídí počátečním stavem sítě a danou matematickou rovnicí. [1, str. 18]



Obr. 1.1: Neuronová síť, převzato z [2]

1.1 Biologický neuron

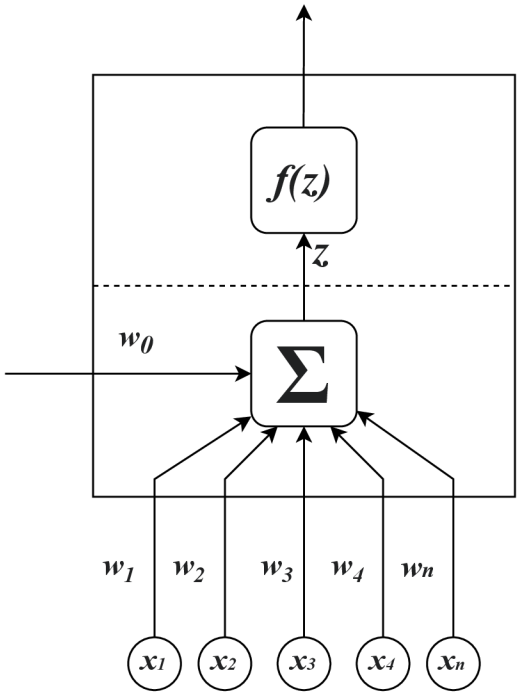
Umělé neuronové sítě jsou tvořeny jednotlivými umělými neurony, které vycházejí z biologického neuronu. Velmi zjednodušeně můžeme biologický neuron popsat na obr. 1.2. Dendrity zde představují část, kudy se signál dostává do neuronu. Tento signál dále putuje do těla buňky, kde je sčítán s ostatními signály. Pomocí nich se signál zeslabí či zesílí a předá dalšímu neuronu. Jedná se tedy o spoje s dalšími neurony. [4, str. 5]



Obr. 1.2: Biologický neuron, převzato z [4, str. 5]

1.2 Umělý neuron

Úplný základ matematického modelu umělé neuronové sítě je něco, co se nazývá formální neuron. Princip fungování je analogický k fungování biologického neuronu s tím rozdílem, že jsou zde zaměněny biologické funkce za funkce matematické. Modely těchto neuronů jsou různé podle jejich složitosti. Nejvíce se rozlišují tvarem své přenosové funkce. Základní rozdělení neuronů podle vstupních a výstupních dat je na neurony binární a spojité. [7, str. 26]



Obr. 1.3: Model formálního neuronu

Formální neuron zpracovává své vstupy podle vztahu

$$y = f(\sum_{i=1}^{n} x_i w_i + w_0). \tag{1.1}$$

Proměnné $x_1, \ldots, x_n$ představují vstupy neuronu. Každý vstup má odpovídající synaptickou váhu $w_1, \ldots, w_n$, která určuje propustnost daného vstupu. Veličina w_0 zde značí práh, na němž závisí z – vnitřní potenciál neuronu ovlivňující jeho aktivační funkci $f(z)$. Ten je zobrazen na obrázku 1.3. [7, str. 26].

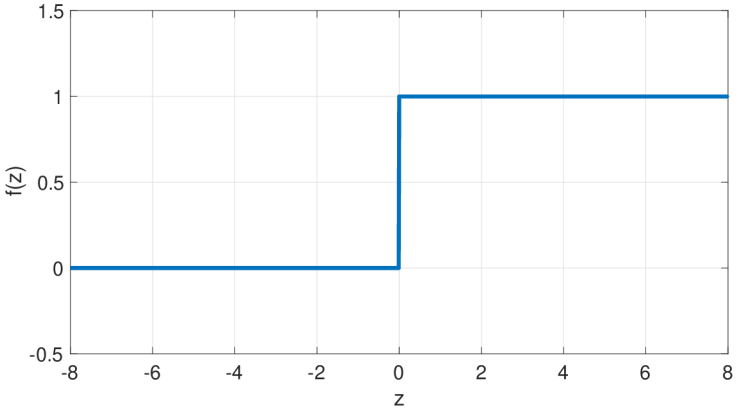
Perceptronem nazýváme základní jednovrstvou neuronovou síť, skládající se jen z jednoho neuronu. Poprvé byl modelován Frankem Rosenblattem v roce 1957. Jeho využití bylo pouze pro klasifikaci lineárně separovatelných kategorií, což vedlo k pozdějšímu nahrazení vícevrstvým perceptronem. [1, str. 24]

1.2.1 Aktivační funkce

Podle aktivační funkce neuronu $f(z)$ je poté určen výstup celého neuronu. Aktivačních funkcí je mnoho a řadí se do několika skupin. Nejdůležitější typy jsou skoková funkce, která je definována rovnicí (1.2) a vykreslena na obr. 1.4, sigmoidální funkce popsaná rovnicí (1.4) a obr. 1.5. Další a často užívanou aktivační funkcí, zvláště u konvolučních neuronových sítí je ReLU (*Rectified Linear Unit*). Jak již název napovídá, jedná se o lineární funkci definovanou v intervalu $\langle 0, \infty)$. Její průběh je znázorněn obrázkem 1.6. [11, str. 4]

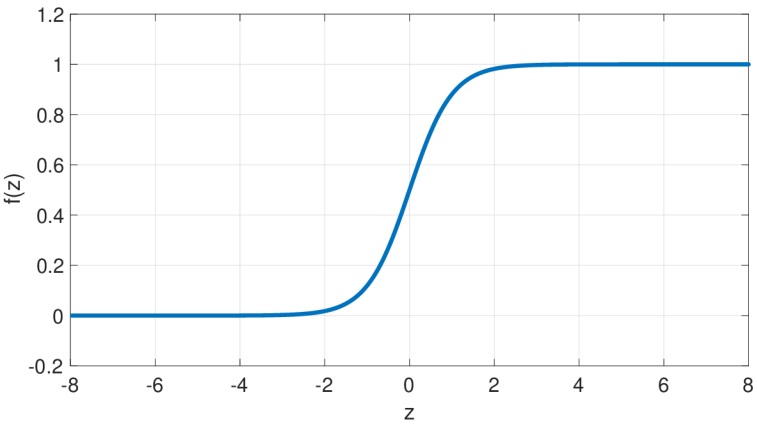
$$f(z) = \begin{cases} 0, & \text{pro } x < 0 \\ 1, & \text{pro } x \geq 0 \end{cases} \tag{1.2}$$

Skoková funkce může nabývat pouze hodnot 0 a 1. Pro záporné hodnoty bude tedy výstupní signál 0.



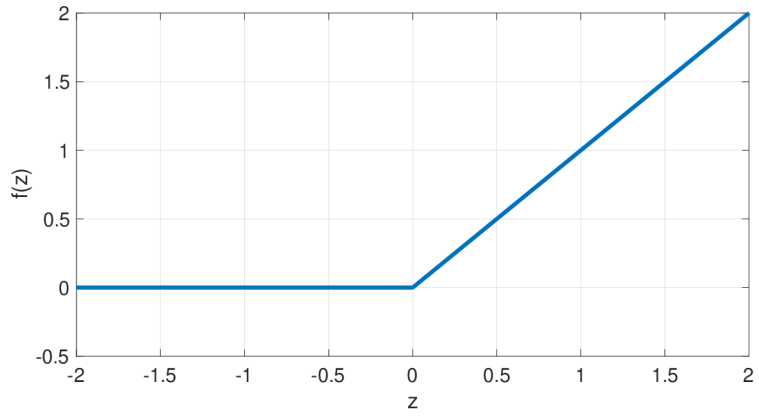
Obr. 1.4: Skoková aktivační funkce

$$f(z) = \frac{\mathrm{e}^z}{1 + \mathrm{e}^z} \tag{1.3}$$



Obr. 1.5: Sigmoidová aktivační funkce

$$f(z) = \max(0, z) \tag{1.4}$$



Obr. 1.6: Aktivační funkce ReLU

1.3 Typy neuronových sítí

1.3.1 Dopředná neuronová síť

Neuronové sítě mají danou architekturu, kterou udává *organizační* dynamika. První z těchto architektur se nazývá *dopředná (acyklická)* síť. U tohoto typu neu-

ronových sítí vede informace jedním směrem – od vstupní vrstvy k výstupní. Podle počtu vrstev se pak dělí *dopředné* neuronové sítě na

- jednovrstvé,
- vícevrstvé.

1.3.2 Rekurentní neuronová síť

Druhým typem architektury je síť *zpětnovazební (rekurentní)*. Ta se liší od klasických dopředných sítí tím, že obsahuje zpětnou vazbu vedenou z neuronové vrstvy zpět na vstup předešlé vrstvy nebo na vstup té samé neuronové vrstvy. Užívají se pro dynamické zpracování informací. [1, str. 19–20]

1.4 Učení neuronové sítě

Dále neuronové sítě můžeme dělit podle jejich způsobu učení, které se dá přirovnat k lidskému učení. To je založené na získávání informací ze svého okolí a na následné adaptaci. Na podobných principech jsou založené učící algoritmy umělých neuronových sítí. Princip spočívá v nastavení prahů a vah každého neuronu sítě na požadované hodnoty, aby síť byla schopna provést požadovanou činnost. [7, str. 33]

Průběh učení sítě je rozdělen na epochy – tréninkové cykly, ve kterém síť použije všechna data k aktualizaci synaptických vah všech neuronů dle rovnice (1.1) přesně jednou. Trénování neuronové sítě obvykle trvá několik epoch, ale na druhou stranu zvýšení jejich počtu nemusí vždy znamenat lepší výsledky sítě. Optimální počet epoch se volí tak dlouho, obvykle dokud nejsou výsledky po sobě jdoucích cyklů stejné. Aby se zmenšila výpočetní náročnost, dělí se tréninková data na dávky (anglicky *batches*) a ty jsou síti poskytnuty postupně. Nastavují se většinou hodnotou parametru *batch_size*. Naučení této jedné dávky se nazývá iterací. Jedna epocha se tedy skládá z několika iterací. [6]

Pro učení neboli také trénování sítě je vybrána sada dat. Často je také zvolena tzv. validační sada, která slouží jako první test sítě na neznámých datech. Pomocí nich se lépe určují hyperparametry samotné sítě jako počty skrytých vrstev, rychlost učení, počty epoch, velikost dávky a mnohé další. Aby bylo učení sítě co nejefektivnější, je možné zvolit i validační frekvenci, která určuje počet iterací mezi užitím validačních dat. [5]

Základní rozdělení je na učení:

- bez učitele,
- s učitelem.

1.4.1 Učení s učitelem

Učení neuronové sítě s učitelem probíhá pomocí učícího algoritmu, který má na rozdíl od neuronové sítě znalosti o vstupních datech a požadovaném výstupu, který chceme u neuronové sítě mít. Námi zvolená síť však na začátku učení zmíněné znalosti nemá.

Neuronové síti poskytneme určitý počet tréninkových dat z okolních prostředí. Stejný soubor dat dáme i učícímu algoritmu, který na základě svých znalostí zareaguje patřičnou odezvou, tou by pak měla zareagovat i neuronová síť. Na základě odezvy učitele, množství chybových a tréninkových dat se upravují jednotlivé parametry sítě. Výstup sítě je upravován v každém kroku, kde se snaží napodobit schopnosti učitele.

Po naučení neuronové sítě na tréninkových datech je pak síť schopna sama reagovat na okolní data i bez přítomnosti učitele, který je pak odebrán z učícího algoritmu. Učení s učitelem je nejčastěji používáno při klasifikaci dat, pomocí neuronové sítě, kdy požadujeme vstupní data roztřídit do určitých kategorií. [7, str. 33–34]

1.4.2 Učení bez učitele

Tento typ učení se liší v tom, že na učící proces a výstup neuronové sítě nedohlíží žádný jiný algoritmus. Algoritmus učení dané sítě nemá přesné informace o vstupech a námi požadovaných výstupech.

Umělá neuronová síť má schopnost měřit a rozpoznávat podobné nebo dokonce totožné hodnoty vstupů. Ty potom třídí do map. Celý princip algoritmu učení spočívá ve výpočtu kvality reprezentace aktuálních hodnot oproti již známým hodnotám.

Příkladem učení bez učitele je učení soutěžní. Neuronová síť má dvě vrstvy, kde první je vstupní vrstva, která obsahuje zdrojové uzly a druhá je ta výstupní s výstupními uzly. Výstupní vrstva má v sobě neurony, které mezi sebou navzájem soutěží. Neuron „vítěz“ má možnost reagovat na vstupní hodnoty z první vrstvy. Metod soutěžení je celá škála. Nejjednodušší neuronová síť se řídí pomocí metody „vítěz bere všechno“, kde je vítěznému neuronu spuštěn jeho výstup. Zbytek neuronů ve vrstvě se uzavře a na výstup sítě nebude mít vliv. [7, str. 34]

1.5 Konvoluční neuronová síť

Umělá neuronová síť obsahující konvoluční vrstvy se nazývá konvoluční neuronová síť (CNN z anglického *convolutional neural network*). Sítě s konvolučními vrstvami se využívají převážně v oblasti zpracování obrazu pro jejich hlavní schopnost detekce jednotlivých prvků snímku, okrajů v různých orientacích nebo vzorů a mnohého dalšího. Tyto prvky utvářejí základ, pro určení více abstraktnějších a složitějších prvků, například obdélníkový tvar stolu, lidská ruka, tvary dopravních značek, části těla zvířete. [8, str. 326]

Velkou problematiku při zpracovávání obrazu představují jeho rozměry. Každý barevný obrázek má obvykle 3 barevné kanály a nejčastěji velikost stovek pixelů. Pro běžnou neuronovou síť je to velmi mnoho, a daná síť by musela obsahovat nespočet parametrů. Konvoluční sítě tuto problematiku řeší čtením vždy malých částí obrázku a provedením vždy stejné operace – konvoluce. Jedná se o součin s vahami sítě, tak jako tomu je u perceptronových sítí v kapitole 1.2. Tato operace je použita na všechna místa na obrázku, která mají totožné váhy. K tomu, aby měl obrázek na výstupu konvoluční vrstvy stejný rozměr jako na jejím vstupu slouží tzv. padding, který přidá rám s nulami ke vstupní matici.

Konvolučních vrstev bývá v CNN hned několik. Všechny jsou aplikovány na stejný obrázek. Lze také tyto vrstvy aplikovat současně a vstupní obrázek je zpracován hned několika konvolucemi naráz. Výsledek má podobné rozměry jako obrázek na vstupu, ale s více kanály. [9]

1.5.1 Operace konvoluce

U každé matematické operace s názvem konvoluce je potřeba určit tzv. konvoluční jádro, neboli matici o libovolných rozměrech, dle využití konvoluce. To samozřejmě platí pro 2D konvoluce, které pracují s obrázky. V případě 1D kovoluce se jedná o vektor. Po definování rozměru jádra a velikosti parametru *padding* určíme velikost kroku. Obecně je konvoluce definována následovně:

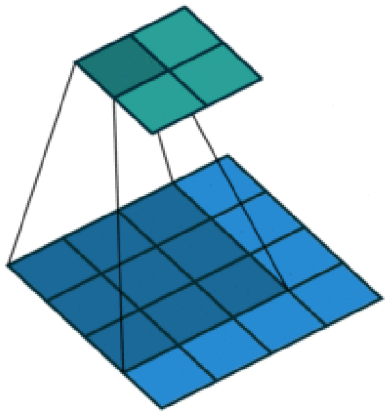
$$\mathbf{S} = \mathbf{X} * \mathbf{W}. \tag{1.5}$$

V kontextu konvolučních sítí chápeme $\mathbf{S}$ jako výstupní matici, $\mathbf{X}$ jako vstupní matici a $\mathbf{W}$ jako konvoluční jádro. Pro prvky jednotlivých matic pak platí:

$$s_{i,j} = \sum_{k=1}^{K} \sum_{l=1}^{L} f_{k,l} x_{i+j,j+l} \tag{1.6}$$

Konvoluce je potom prováděna způsobem, že jádro je přiloženo na vstupní matici, překrývající prvky se vynásobí a jejich součet je zapisován do výstupní matice,

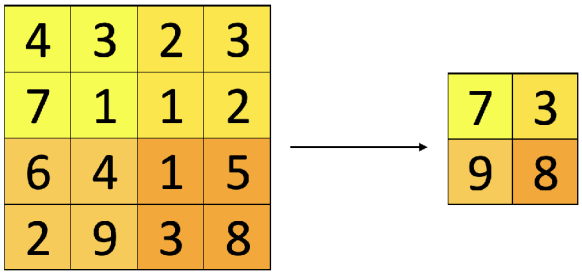
jak ilustruje obrázek 1.7. Následně se jádro posune o předem definovaný krok (u konvolučních vrstev nazývaný *stride*) a postup se znovu opakuje. Na obrázku níže je vstupní matice znázorněna modrou a výstupní matice zelenou barvou. Zde zmíněná konvoluce je popsána pro toto konkrétní využití u konvolučních neuronových sítí. Zde funguje jako filtr, tudíž zmenšuje rozměry. V jiných případech rozměry zůstavají stejné či se dokonce zvětšují. [11, str. 6]



Obr. 1.7: Operace konvoluce, převzato z [10]

1.5.2 Pooling

Jedná se o vrstvu, která se většinou střídá s konvoluční vrstvou. Její princip spočívá ve snižování velikosti obrázků na vstupu. Toho můžeme využít například u klasifikačních úloh, kde máme např. 20 různých tříd a na vstupu obrázky o velkých rozměrech. Chceme získat pouze pravděpodobnosti příslušnosti jednotlivých obrázků do určité třídy, proto můžeme zredukovat dimenzi obrázku.



Obr. 1.8: příklad funkce MaxPooling, překresleno z [11, str. 8]

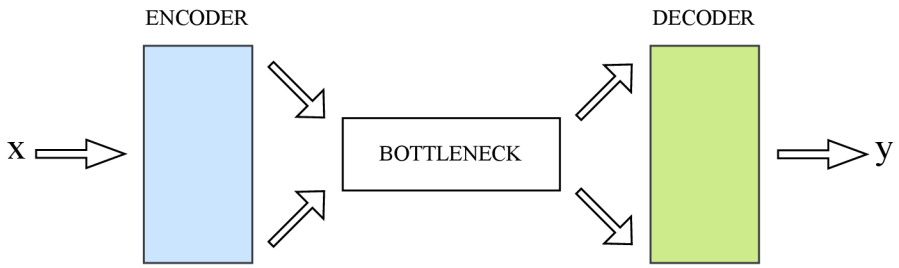
Dva nejpoužívanější způsoby jsou *max pooling* nebo *average pooling*. Máme zde okno o rozměrech například 2×2. Příklad funkce *max pooling* je zobrazen na obr. 1.9.

Na rozdíl od konvolučních vrstev se nepoužívá překrývání. Díky tomu se poté rozlišení sníží na polovinu. Pooling vrstvy snižují náročnost učení dané sítě, jelikož síť nemusí pracovat s obrázky o velkém rozlišení. Podobně tomu je i u audiosignálů. [11, str. 7]

1.5.3 Autoencoder

Pod pojmem *autoencoder* se rozumí umělá neuronová síť, která má za úkol kódovat a komprimovat data a následně je zpět rekonstruovat do co nejvíce přesné podoby původního vstupu. Tento typ sítě snižuje rozměry dat tak, že se snaží ignorovat šum v datech. Skládá se ze tří hlavních částí:

- Encoder,
- Bottleneck,
- Decoder.



Obr. 1.9: Schéma autoencoderu

Encoder slouží k redukci dimenze a kompresi vstupních dat. *Bottleneck* je vrstva, která obsahuje komprimovaná data. Vstupní data v této vrstvě mají nejmenší možné rozměry. V části *Decoder* se síť učí jak co nejlépe rekonstruovat data ze zakódované reprezentace. Další částí, která není přímo součástí sítě, je *Reconstruction Loss*. Ta porovnává rekonstruovaná data s původními vstupními daty a vyhodnocuje chybu. Tato část je součást učícího algoritmu. [12]

U-Net

Jedním z mnoha druhů CNN je konvoluční neuronová síť zvaná U-Net. Byla vyvinuta pro rychlou segmentaci biomedicínských snímků. Za jejím vznikem v roce 2015 stojí Olaf Ronneberger, Philipp Fischer a Thomas Brox. Samotná síť vychází z plně konvoluční sítě, která byla upravena tak, aby uměla pracovat s menším počtem tréninkových dat a dokázala snímky lépe segmentovat. V biomedicínském prostředí tak nejen rozlišuje zda se jedná o nemoc, ale také dokáže najít různé abnormality na snímku. Tato schopnost najít a rozlišit hranice je způsobena tím, že síť klasifikuje každý pixel, takže vstup i výstup mají stejnou velikost. [13]

1.6 Generativní soupeřící sítě

Speciálním typem umělých neuronových sítí jsou takzvané *Generativní soupeřící sítě* (GAN z anglického *Generative adversarial network*). Na rozdíl od základních neuronových sítí, které dokáží pracovat pouze se vstupními daty, má GAN schopnost generovat syntetická data na základě těch vstupních. Zjednodušeně jde o dvě neuronové sítě soupeřící mezi sebou. První síť je generátor, který data vytváří a snaží se, aby byla nerozeznatelná od skutečných. Ta druhá se nazývá diskriminátor a má za úkol co nejlépe rozlišovat mezi skutečnými daty a těmi, které vytvořil generátor. Cílem celého souboru sítí je, aby generátor byl schopen vytvářet data, které diskriminátor nebude schopen rozlišit od skutečných dat. [14, str. 12]

1.7 Využití v audiosignálech

Umělé neuronové sítě lze využít k mnohým úlohám v oblasti zpracování zvukových signálů – ať už k redukci šumu v nahrávkách nebo rekonstrukci poškozených částí. K velkým inovacím v deep learningu dochází v oblastech rozpoznání řeči. Pomocí neuronových sítí lze detekovat, lokalizovat a sledovat zvuk. V rámci klasifikačních úloh lze přiřadit jednotlivé emoce k nahrávce pomocí intonačních křivek. [15]

AIVA

Jedno z nejpokročilejších užití ANN v oblasti kompozice hudby se nazývá AIVA. Tato umělá inteligence na bázi hlubokého učení, vytvořená bratry Pierrem a Vincentem Barreauovými, je schopna dokončovat nedopsané partitury skladeb vážné hudby nebo dokonce psát nové skladby. Projekt vznikl jakožto magisterská práce Pierra a později přešla ve startup projekt. Jedná se o první umělou inteligenci zapsanou ve společnosti autorů jako virtuální skladatel. Mezi dokončená díla od tohoto „skladatele“ patří i dokončení skladby od českého skladatele Antonína Dvořáka nesoucí název *AIVA / Antonín Dvořák: From the Future World op. 71.* [16]

Audio inpainting

Dalším možným využitím hlubokých neuronových sítí je obnova chybějícího zvukového obsahu na základě jeho kontextu, často nazýván jako *inpainting* zvuku. Studií se zabýval například Andrés Marafioti a kolektiv v článku [18]. Zde byla pro obnovu části chybějící hudby v řádech několika desítek milisekund použita konvoluční neuronová síť typu *encoder-decoder*.

Text to speech

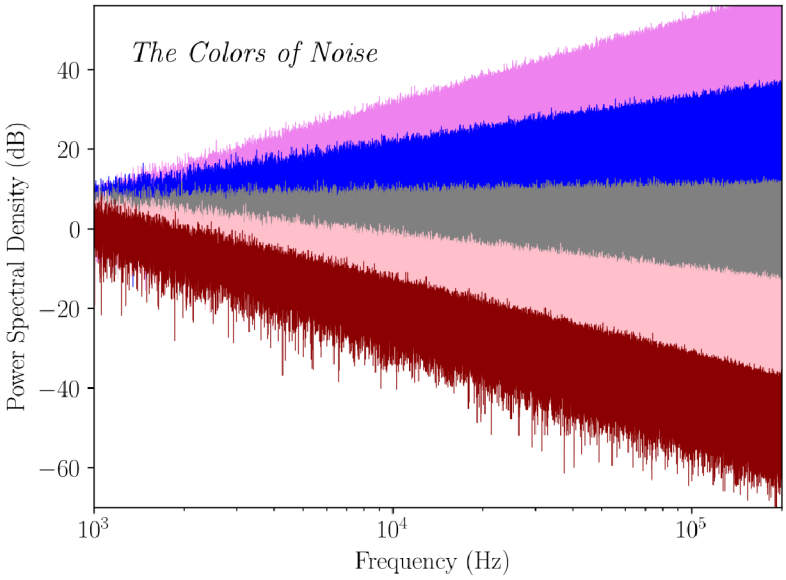
Hluboké učení je hojně využíváno při umělé produkci lidské řeči, konkrétně se využívá pro převod textu na řeč nebo při navigačních systémech. Příkladem těchto umělých sítí je síť *WaveNet*, která generuje nejmodernější výsledky převodu textu na řeč pro dva jazyky – angličtinu a mandarínštinu. [19]

Speech to text

Stejně jako převádět text na mluvené slovo, umí ANN i obrácený postup, tedy převod mluveného slova na text. Některé modely zašly v tomto odvětví ještě dál a převáděné řeči porozumí. Nejznámější příklady jsou asistenti v elektronických zařízeních jako *SIRI* od společnosti Apple nebo *CORTANA* od firmy Microsoft. Oba tito asistenti využívají hlubokého učení.

2 Šum ve zvukových signálech

Ve zvukových signálech je možné nalézt nespočet rušivých složek, které znehodnocují signál a zhoršují kvalitu poslechu. Samotný šum může mít mnoho původů, podle kterého se dělí na různé typy. Za akustický šum lze považovat ruch okolí, např. hluk aut na ulici, hučení větráku počítače, zpěv ptáků či hlasy lidí v pozadí. Příkladem šumů, vznikajících při digitálním nebo analogovém zpracování signálů, mohou být tzv. *barevné šumy*, které bývají mnohdy i žádoucím jevem pro jejich využití v praxi. Dělí se podle hustoty výkonu na bílý, růžový, modrý, černý šum a další.



Obr. 2.1: Barvy šumů, podle hustoty výkonu, převzato z [20]

Bílý šum

Patří k nejznámějším typům barevných šumů. Jedná se o signál, obsahující různé a náhodné frekvence se stejnou intenzitou – jeho výkonová spektrální hustota je konstantní. Je zde pozorovatelná analogie k bílé barvě, která v RGB modelu tvoří součet všech barev, stejně jako bílý šum je součtem všech frekvencí se stejnou intenzitou. Ovšem většina lidí vnímá spíše vyšší frekvence v tomto šumu. V praxi lze najít tento typ šumu např. u bicích nástrojů. Užívá se také jako pomoc lidem s tinnitem či při problémech se spánkem. [21]

Růžový šum

Hustota výkonu tohoto šumu klesá o 3 dB na oktávu. Výkon pásma růžového šumu je tedy nepřímo úměrný frekvenci. Podle vlastností je často řazen mezi hnědý neboli červený a bílý šum, proto název růžový. Bývá přirovnáván k foukání ventilátoru nebo k víření vody. V praxi bývá používán, podobně jako bílý šum, pro soustředění a spánek. Dále nachází uplatnění při konstrukci reproduktorových soustav a při dalších testech v audio inženýrství. [22]

Hnědý šum

Hnědý šum, někdy také označován jako *červený šum*, vzniká podle Brownova pohybu. Jedná se náhodný vzorec pohybu používaný k popisu chování částic v kapalném nebo plynném prostředí. Je podobný růžovému šumu, ale pokles hustoty výkonu je mnohem prudší – až 6 dB na oktávu. Bývá tvořen nízkofrekvenčními basovými tóny a jeho zvuk je přirovnáván k vodopádům. [22]

Modrý šum

Tento šum bývá často nazýván *azurový šum*. Jedná se o opak růžového šumu, protože hustota výkonu roste o 3 dB na oktávu. Lidé jeho zvuk popisují jako syčení. Pro terapeutické účely bývá využíván velmi zřídka, kvůli jeho ostrosti na vyšších frekvencích. Uplatňuje se hlavně ve zvukovém inženýrství při ditheringu, kde slouží k úpravě zvuku a snížení slyšitelnosti jeho zkreslení. [22]

Fialový šum

Jedná se o opak hnědého šumu. Má větší nárůst hustoty výkonu než modrý šum, konkrétně 6 dB na oktávu, takže je velmi ostrý na vysokých frekvencích. Často se používá pro maskování vysokofrekvenčních hluků v pozadí a v některých případech pro léčbu tinnitu. [21]

Šedý šum

Šedý šum představuje psychoakustickou křivku s váhou A, proto zní jako kdyby každá frekvence ve spektru hrála na stejné úrovni, i když tomu tak není. Přesto každému člověku bude znít trochu jinak kvůli individuálnímu sluchovému vnímání. Využívá se také pro léčbu tinnitu a hyperakuzie. [22]

2.1 Redukce šumu

Existuje mnoho metod zabývajících se redukcí nechtěného rušení ve zvukových signálech. Mezi ty nejjednodušší patří filtry typu horní nebo dolní propust, pásmová propust nebo zádrž. Tato metoda je účinná hlavně pro oblasti, kde se nepřekrývají frekvenční spektra šumu a signálu. Její srovnání s metodou vlnkové transformace pro potlačení šumu v signálu bylo popsáno v práci [23]. Typem filtru pro redukci šumů z okolí např. v naslouchátkách je i *Wienerův filtr* [24]. Studie [25] zabývající se *empirickou modální dekompozicí* ukazuje potlačení šumu rozkladem signálu na vnitřní oscilační složky a porovnává je s dalšími metodami. V prvních letech řešení této problematiky byla navržena metoda odčítání spektrálního odhadu šumu od zašuměného signálu [26]. Tento způsob je stále oblíbený pro jeho výpočetní účinnost.

V rámci této práce byla metoda redukce šumu pomocí neuronových sítí porovnána s metodou potlačení šumu využívající vlnkovovu transformaci. Ta rozkládá zašuměný signál na skupiny koeficientů na různých frekvencích. Koeficienty představující šum v signálu jsou následně odečteny a signál je složen zpět bez nich. Pro správný denoising se volí různé typy vlnek, jejich počet a hodnota prahování. [27]

3 Zpracování audiosignálu

3.1 Fourierova transformace

V této práci je použito několik matematických funkcí ke zpracování audiosignálu. Základní funkcí a matematickým nástrojem pro analýzu a následné zpracování signálu je lineární integrální transformace známá jako Fourierova transformace. Jedná se o zobrazení, které každému signálu přiřadí jinou funkci. Z jejich vlastností pak lze zjistit informace o původním signálu.

Fourierova tranformace (také Fouriuerův obraz) převádí signál z časové oblasti do kmitočtové oblasti. Obecně pro signály se spojitým časem je Fourierova tranformace dána vztahem

$$S(\omega) = \int_{-\infty}^{\infty} s(t)\mathrm{e}^{-\mathrm{j}\omega t}\mathrm{d}t, \tag{3.1}$$

kde $S(\omega)$ značí obraz transformace a $s(t)$ její předmět.

K této operaci existuje i inverzní operace zvaná Zpětná Fourierova tranformace. Ta se stará o převod signálu z kmitočtové oblasti do oblasti časové. [28]

$$s(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} S(\omega)\mathrm{e}^{\mathrm{j}\omega t}\mathrm{d}\omega \tag{3.2}$$

Pokud pracujeme v diskrétním čase, je Diskrétní Fourierova tranformace obecně dána vztahem

$$S(\Omega) = \sum_{\mathrm{k}=-\infty}^{\infty} s(k)\mathrm{e}^{-\mathrm{j}\Omega k}. \tag{3.3}$$

Stejně jako je tomu u signálů se spojitým časem, tak i u signálů s časem diskrétním máme zpětnou Fourierovu tranformaci

$$s(k) = \frac{1}{2\pi}\int_{0}^{2\pi} S(\Omega)\mathrm{e}^{\mathrm{j}\Omega k}\mathrm{d}\Omega. \tag{3.4}$$

3.1.1 Krátkodobá Fourierova tranformace

Pro časově-frekvenční analýzu signálu se používá krátkodobá Fourierova tranformace (STFT z anglického *Short-time Fourier Tranform*). Signál je rozdělen na krátké segmenty a ty jsou vynásobeny okenní funkcí. U signálů se spojitým časem je STFT definována jako

$$S_{\mathrm{STFT}}(\omega, \tau) = \int_{-\infty}^{\infty} s(t)w(t-\tau)\mathrm{e}^{-\mathrm{j}\omega t}\mathrm{d}t, \tag{3.5}$$

kde $s(t)$ značí spojitý signál, $w(t-\tau)$ použité časové okno. Parametr τ určuje posunutí časového okna. [29, str. 51–52]

Pro diskrétní signály vzorec STFT vypadá následovně:

$$S_{\mathrm{STFT}}(\mathrm{e}^{j\omega}, m) = \sum_{n=-\infty}^{\infty} s[n]w[n-m]\mathrm{e}^{-\mathrm{j}\omega n} \tag{3.6}$$

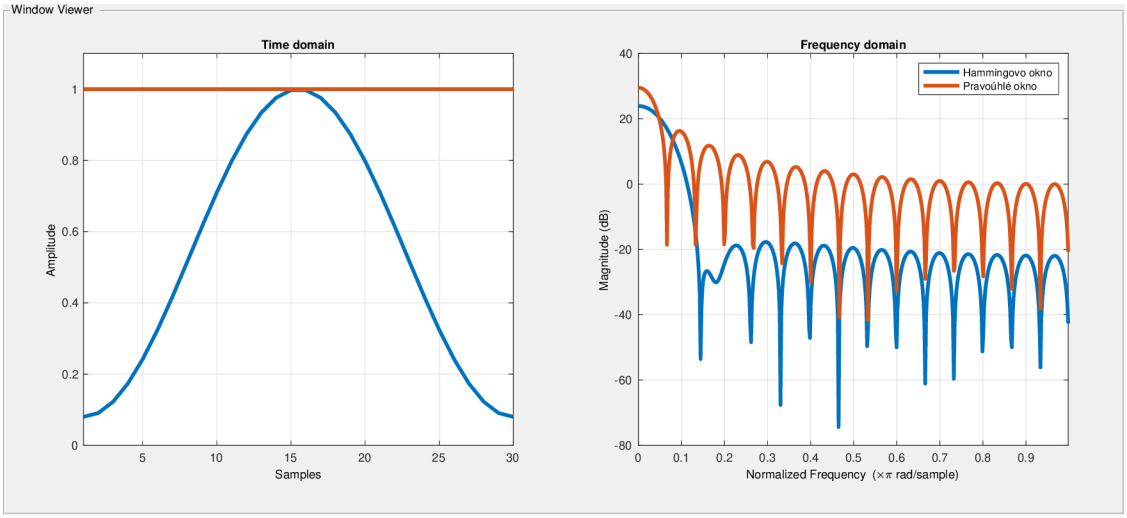
3.2 Segmentace

Audiosignál řeči je zpracováván pomocí metod krátkodobé analýzy. Při zpracovávání se signál rozděluje na jednotlivé úseky pomocí oken. Tyto úseky neboli segmenty mají určitou délku N a v jistých případech se mohou i překrývat. Rozdělení delšího úseku signálu na více kratších segmentů je realizováno časovými okny. V případě audiosignálu je nejpoužívanější Hammingovo a pravoúhlé okno. Obě jsou vykreslena na následujícím obrázku 3.1. V levé části je zobrazen časový průběh a vpravo modulová spektra obou oken v decibelové stupnici. Z nich lze vyčíst, že hlavní lalok Hammingova okna je dvojnásobně širší, než je tomu u pravoúhlého okna. Podle toho můžeme usoudit, že Hammingovo okno má horší kmitočtové rozlišení než pravoúhlé. Laloky u pravoúhlého okna mají menší útlum, takže je zde větší pravděpodobnost výskytu chyb, způsobených prosakováním spektrálních složek z jiných laloků. Hammingovo okno je z tohoto důvodu vhodnější pro použití. [29, str. 50–51] Hammingovo okno je vyjádřeno vzorcem:

$$\begin{aligned} w[n] &= 0{,}54 - 0{,}46\cos\left(n\tfrac{2\pi}{N}\right), && \text{pro } n = 0, 1, \ldots, N-1, \\ w[n] &= 0, && \text{pro ostatní } n. \end{aligned} \tag{3.7}$$

Pravoúhlé okno je obecně definováno jako:

$$\begin{aligned} w[n] &= 1, && \text{pro } n = 0, 1, \ldots, N-1, \\ w[n] &= 0, && \text{pro ostatní } n. \end{aligned} \tag{3.8}$$



Obr. 3.1: Hammingovo a pravoúhlé okno, $N = 30$

4 Volba frameworku

Existuje mnoho architektur neuronových sítí, které lze použít pro potlačení šumu ve zvukovém signálu. Například *encoder-decoder* Wave U-Net [33], konvoluční rekurentní neuronová síť pro vylepšování řeči v reálném čase [34]. Samuel Kyung Won Park ve své práci porovnal několik typů ANN a technik pro aktivní potlačení šumu [35]. V této bakalářské práci byla implementována jednoduchá konvoluční neuronová síť na potlačení šumu ve dvou programovacích prostředích a poté byly porovnány jak výsledky redukce šumu, tak průběhy trénování sítě v daných prostředích.

4.1 Srovnání dvou prostředí

Jelikož při redukci šumu v audio signálu jde obvykle o redukci šumu ve spektrogramu daného signálu, což je vlastně 2D reprezentace signálu, kterou lze brát jako obrázek, byla v rámci bakalářské práce implementována jednoduchá neuronová síť na potlačení šumu právě u obrázků. Test nejprve proběhl v programovacím jazyce Python a následně za velmi podobných podmínek v programu Matlab. Obě neuronové sítě pracovaly s MNIST *(Modified National Institute of Standards and Technology)* datasetem, což je soubor malých, černobílých, ručně psaných obrázků o rozměrech 28×28 pixelů. Používá se obvykle pro testování a trénování různých neuronových sítí a pro tuto práci byl využit kvůli malému rozměru obrázků, které zaručovaly menší výpočetní náročnost při testování v obou prostředích.

Ze subjektivního hlediska se lépe pracovalo v prostředí Matlab, které uživateli umožňuje přehlednější zobrazení proměnných a nabízí chytrou nápovědu při vytváření kódu. Grafické rozhraní a předem vytvořené aplikace zahrnuté v toolboxech jsou uživatelsky přívětivější zejména pro někoho, kdo je v této oblasti méně zkušený. Ve zpracování signálu je Matlab také mnohem intuitivnější, avšak některé postupy jsou přesně definované a nelze je měnit.

Na druhou stranu Python, jakožto dospělejší programovací jazyk, poskytuje větší variabilitu. Samotný ale nemá tak propracované grafické rozhraní a je v mnoha oblastech velmi neintuitivní. Co se týče umělých neuronových sítí, poskytuje rychlejší učení a hlavně úspěšnější výsledky, jak bude podrobněji popsáno v kap. 4.1.5 a 4.1.6.

4.1.1 Python

Jako první byl otestován programovací jazyk Python. Pro jeho realizaci byl využit program Visual Studio Code od společnosti Microsoft. K provedení jednoduchého úkonu bylo vytvořeno virtuální prostředí *mlp* s verzí Pythonu 3.8.13. Důvod tohoto

kroku bylo použití několika knihoven, které nejsou kompatibilní s *ARM64* architekturou, používanou novějšími zařízeními s *Apple Sillicon* procesory od společnosti Apple. Knihovny, které byly použity jsou vypsány níže.

NumPy – Knihovna pro práci s číselnými daty, jedno- a vícerozměrnými maticemi a vektory. Jedná se o volně dostupný software.

Matplotlib – Slouží k zobrazení dat nebo grafů v jazyce Python. Umožňuje také vytvářet animované a interaktivní vizualizace.

Keras – Jedná se o vysoce kvalitní rozhraní sloužící k programování aplikací, které umožňuje uživateli snadno pracovat se všemi druhy umělých neuronových sítí od jejich návrhu přes trénink po jejich implementaci.

TensorFlow – TensorFlow je bezplatná knihovna pro strojové učení od společnosti Google. Je to jedna ze tří knihoven, kterou v současné době nabízí rozhraní *Keras*. Jako konkurenci lze považovat např. knihovny PyTorch od společnosti META nebo scikit-learn. [30, str. 318]

4.1.2 Matlab

Jako druhé prostředí byl využit matematický software Matlab. Používaná verze aplikace byla Matlab R2021b. Obdobně jako Python pracuje s knihovnami funkcí, Matlab používá tzv. toolboxy. Pro tuto úlohu byly nainstalovány a následně využity níže zmíněné toolboxy.

Image Processing toolbox – Knihovna určená k zpracování, vizualizaci a analýzu obrázků. Tento Toolbox umožňuje segmentaci, vylepšování obrazu nebo redukci šumu u obrazu, čehož bylo využito v této bakalářské práci. Knihovna obsahuje řadu aplikací sloužící ke zpracování obrazu.

Deep Learning toolbox – Deep Learning Toolbox je knihovna, která poskytuje uživateli nástroje k vytváření, analýze, tréninku a následné implementaci umělých neuronových sítí. Pomocí několika aplikací lze modelovat a analyzovat neuronové sítě a následně je využívat k daným úlohám.

Deep Network Designer

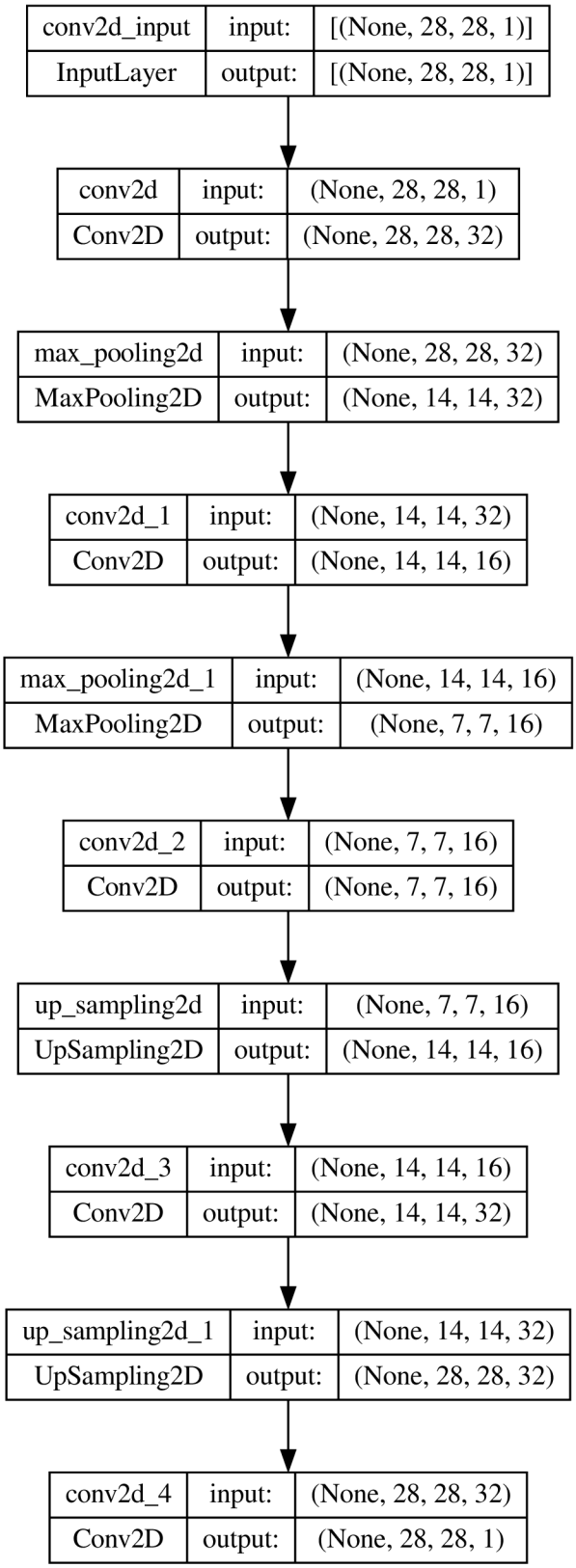
Součástí Deep Learning toolboxu je i aplikace Deep Network Designer, která slouží k názornému modelování a vizualizaci hlubokých neuronových sítí. Aplikace umožňuje načtení dat o síti přímo z pracovního prostředí Matlabu a jejich následnou úpravu pro vlastní použití. Návrh v aplikaci je ukázán v následující kapitole na obr. 5.1.

4.1.3 Načtení a příprava dat

Pro načtení obrázků byly vytvořeny dva datasety. První `iTest` obsahuje obrázky pro test sítě, `iTrain` pro její trénink. Oba pak byly převedeny do 4D polí s názvy `imgTrain` a `imgTest`. Následně byl zvolen `noiseFactor` a k původním obrázkům byl přidán náhodný šum. Umělé neuronové síti byla poskytnuta data k tréninku i validaci.

4.1.4 Architektura neuronové sítě

ANN typu *encoder-decoder* byla vytvořena v obou programovacích jazycích, kde v případě Matlabu bylo využito aplikace Deep Network Designer. Na rozdíl od Pythonu zde byla pro každou konvoluční 2D vrstvu vytvořena zvlášť další vrstva označující její aktivační funkci. Proto se počet vrstev sítí liší, nicméně na funkci to vliv nemá. Síť je v *encoder* části tvořena konvolučními vrstvami a pooling vrstvami, jež jsou popsány v 1.5.2. Drobný rozdíl se týká vrstev, kde je původní obrázek rekonstruován. I přesto, že je jejich funkce stejná, jsou v Python reprezentaci tyto vrstvy označeny jako *UpSampling2D* a v Deep Netowrk Designeru jako *resize2Dlayer*. Dalším rozdílem je poslední neboli výstupní vrstva. V Pythonu je možno jako poslední vrstvu určit konvoluční 2D vrstvu se sigmoidovou aktivační funkcí, v Matlabu je ovšem potřeba za tuto aktivační vrstvu zařadit ještě výstupní vrstvu *regressionOutput*. Architektura sítě byla inspirována prací [31].



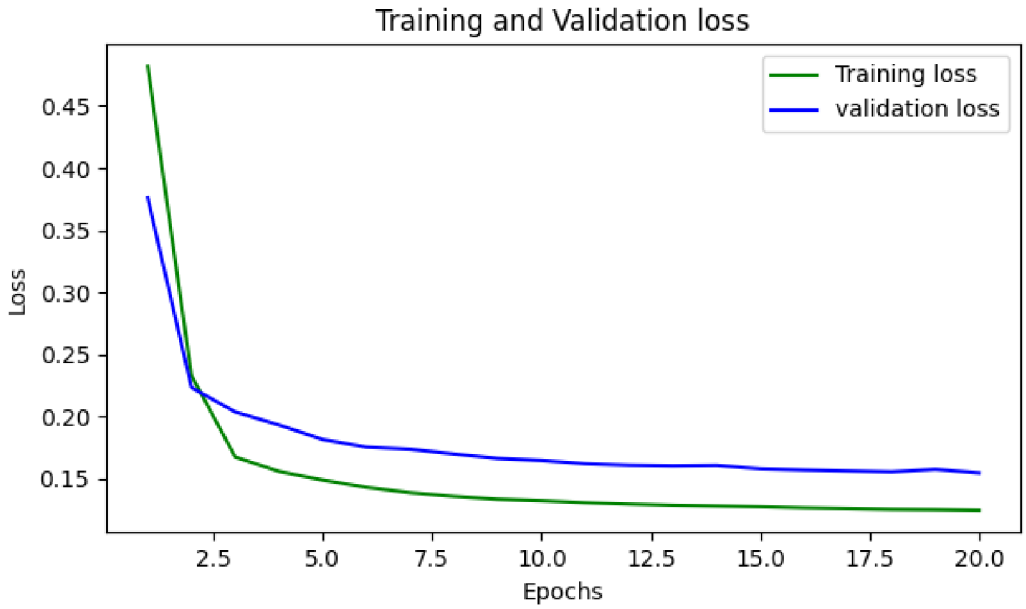
Obr. 4.1: Model sítě implementované v obou prostředích

4.1.5 Učení sítě

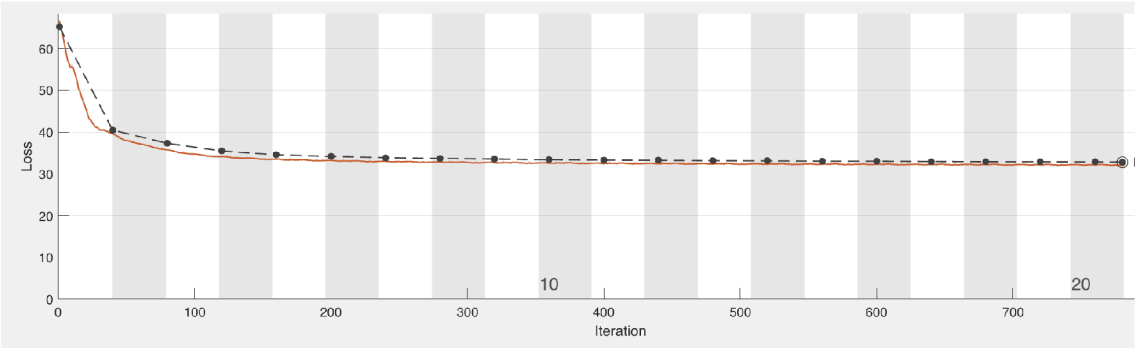
Pro obě ANN byly nastaveny podobné tréninkové podmínky. Po řadě pokusů o jejich vyladění byl pro model navržený v Pythonu nastaven počet epoch na 20, *batch_size* na 256 a byla zvolena data určené k validaci a samotná validační frekvence. Dle těchto podmínek se pak trénink sítě nastavil obdobně i v Matlabu. V modelu trénovaném v jazyce Python byla po testování různých ztrátových funkcí (včetně *mean squared error*) nakonec vybrána ztrátová funkce *binary_crossentropy* pro její nejlepší výsledky. Je definována rovnicí 4.1 (převzato z [14]) a hodnotí model jehož výstupem je hodnota z intervalu $\langle 0, 1 \rangle$. Požadovanou hodnotou je y, $\hat{y}$ je zase N jiných hodnot předpovězených modelem. V druhém programu tato funkce nemohla být vybrána, jelikož výstupní regresivní vrstva má pevně definovanou ztrátovou funkci jako MSE.

$$L(y, \hat{y}) = -\frac{1}{N} \sum_{i=0}^{N} (y \cdot \log(\hat{y}_i) + (1 - y) \cdot \log(1 - \hat{y}_i)) \tag{4.1}$$

Průběh samotného trénování se hned na první pohled neshodoval. Zatímco při učení v Pythonu docházelo k poklesu chyby sítě až k poslední epoše, tak v Matlabu již v polovině učícího cyklu byla chyba téměř totožná v každé další epoše, takže mohl být trénink ukončen. Chyby obou sítí se velmi lišily, což bylo zřejmě způsobeno rozdílnou chybovou funkcí.



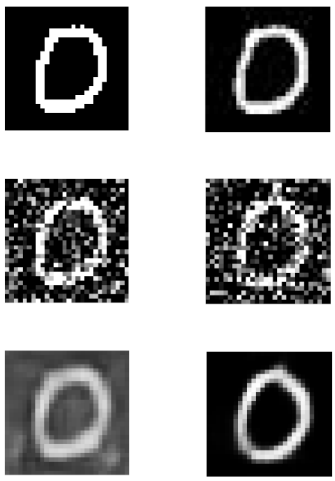
Obr. 4.2: Průběh učení sítě v Pythonu



Obr. 4.3: Průběh učení sítě v Matlabu

4.1.6 Výsledky

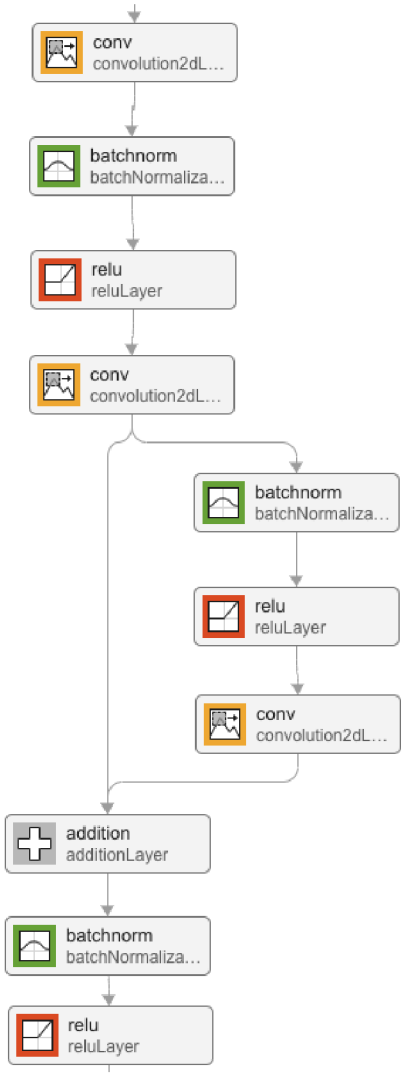
Obrázek vložený do obou sítí byl formátován stejným způsobem a byl k němu přičten totožný šum. Výsledné potlačení rušení, které bylo na výstupu je zobrazeno na obrázku. Levý sloupec znázorňuje výsledky z Matlabu, pravý z Pythonu. V horní řadě jsou původní obrázky přesně tak, jak byly přečteny ze souboru, prostřední řada reprezentuje přidaný šum. Není zde viditelný žádný velký rozdíl, přestože se jedná o náhodně přidaný šum (avšak se stejným faktorem). Spodní řada poté zobrazuje následné odšumění. Jak je patrné, ANN v Pythonu si s tímto problémem poradila o něco lépe, obrázek působí pouze *rozmazaně* a má blízko své původní podobě. Oproti tomu výsledek z Matlabu je v této fázi zcela neuspokojivý. V rámci této bakalářské práce byl však dále využit Matlab pro jeho lepší intuitivnost.



Obr. 4.4: Výsledky potlačení šumu u obrázku

5 Návrh sítě pro potlačení šumu

Na základě rešerše o ANN v oblasti redukce šumu v audio signálu byla shledána konvoluční neuronová síť se skokovým propojením za vhodnou pro tuto práci. Původní návrh zapojení byl použit ve článku [32], kde aplikovaná síť dosahovala lepších výsledků než klasická *encoder-decoder* síť. Skryté vrstvy obsahují pouze tři typy vrstev, a sice neustále se za sebou opakující konvoluční vrstvy, *batch normalization* vrstvy, které slouží k normalizaci dat v jedné dávce tréninkových dat při učení sítě a samotné aktivační vrstvy konvolučních vrstev.



Obr. 5.1: Uspořádání vrstev v umělé neuronové síti vytvořené pomocí Deep Network Designer

Konvoluční vrstvy jsou připojeny k následujícím vrstvám a zároveň skokově připojeny až k vrstvám nacházejícím se v encoder části. Toto propojení se přidává ke každé druhé konvoluční vrstvě a má za úkol zrychlit a zefektivnit trénování sítě. V rámci této práce byly navrženy 4 různé typy neuronových sítí, jejichž schémata jsou obsažena v příloze. Všechna obsahují skoková propojení, ale liší se počtem těchto propojení a počtem konvolučních vrstev, jak lze vidět v následující tabulce. Sítě číslo 1, 2 a 4 mají podobnou topologii. Síť číslo 3 má navíc paralelní rozdělení vrstev.

Tab. 5.1: Druhy testovaných sítí

	Počet vrstev	Počet skokových propojení
SÍŤ 1	16	2
SÍŤ 2	24	5
SÍŤ 3	23	3
SÍŤ 4	19	4

5.1 Dataset nahrávek

K učení neuronových sítí pro potlačení rušení ve zvukovém signálu je třeba vybrat sadu nahrávek, na kterých bude síť trénována a následně testována. Ty se dělí podle účelu sítě, např. *Audio MNIST* vytvořený pro sítě, které mají za úkol klasifikovat audio signály, *Acted Emotional Speech Dynamic Database* pro rozpoznávání emocí nebo *Public Domain Sounds* používající se v studiích zkoumajících detekce objektů podle zvuku. Pro účely bakalářské práce bude využíván již připravený a volně dostupný dataset *Common Voice Corpus 11.0*, který obsahuje nahrávky 578 různých hlasů v češtině.[1]

5.2 Matlab

Práce byla implementována v prostředí Matlab, pro lepší manipulaci a návrh konvolučních neuronových sítí. Kromě základních toolboxů uvedených již v části 4.1.2, bylo využito několika dalších toolboxů.

[1]Použitý dataset: <`https://commonvoice.mozilla.org/cs/datasets`>

Audio toolbox

Toolbox (verze 3.1) umožňujicí načtení a zpracování zvukového signálu. Obsahuje algoritmy a funkce pro ekvalizaci, časový posun a měření zvukového signálů. Jeho součástí je též několik předtrénovaných neuronových sítí. V této práci byl využit pro načtení a následnou práci s datasetem. Zároveň obsahuje funkci pro generaci růžového šumu.

Signal processing toolbox

Knihovna funkcí a aplikací ve verzi 8.7 umožňující zpracování signálů a jejich úpravu. Pomocí tohoto toolboxu lze analyzovat a navrhovat různé typy filtrů. Součástí je také řada aplikací usnadňijící analýzu či práci se signály. Pro zpracování signálů byl také využit DSP system toolbox (verze 9.13).

Parallel computing toolbox

Pomocí této knihovny lze řešit výpočetně náročné operace v matlabu. Umí využít více jádrových procesorů počítače, případně GPU. V této bakalářské práci byla využita verze 7.5 pro manipulaci s datasety a trénování neuronových sítí, které byly výpočetně velmi náročné.

Communications Toolbox

Knihovna ve verzi 7.6 obsahuje funkce a aplikace pro navrhování a analýzu komunikačních systémů. Byl využit pro vytváření sad pro neuronové sítě.

Wavelet Toolbox

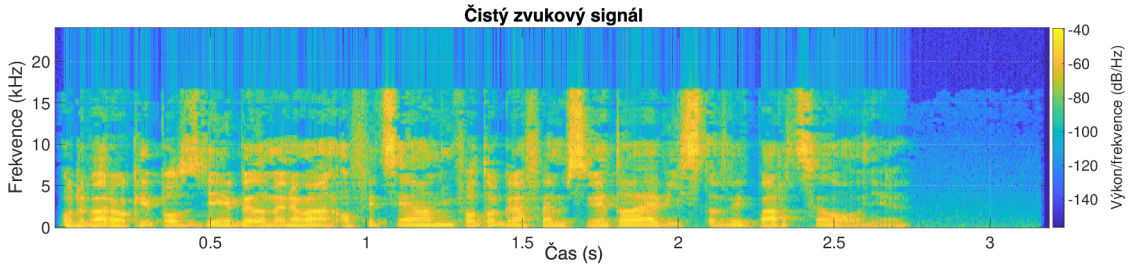
Toolbox ve verzi 6.0 obsahuje několik funkcí a aplikací používající vlnkovou transformaci. V této práci je z této knihovny využívána aplikace *Wavelet signal denoiser*, která umožňuje redukci šumu za použití vlkové transfomrace.

5.3 Trénování sítě

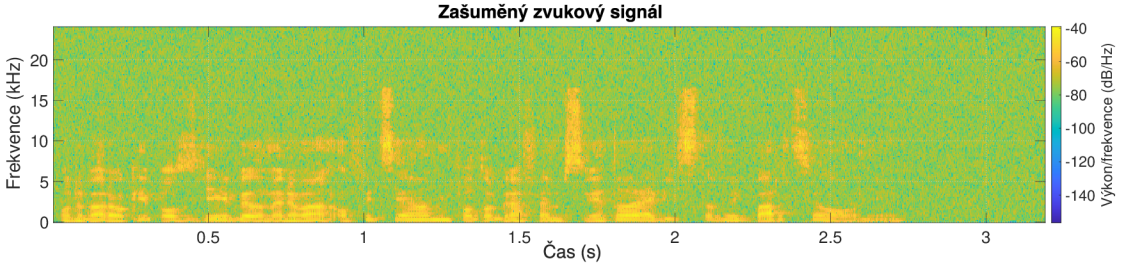
Pro využití nahrávek jako tréningových dat sítě byl v souboru `TRAIN.m` načten zvolený dataset. Za účelem snížení výpočetní náročnosti byl počet nahrávek omezen na 100. Poté byl vytvořen šum a přičten k původnímu zvukovému signálu na dané úrovni SNR, konkrétně tedy 10. V rámci celé práce byly přičítány 3 uměle vytvořené barevné šumy a jeden skutečný šum vodopádu.[2] Čistý a zašuměný zvukový signál byl

[2]Vzorek šumu vodopádu je k dostání zde: `https://pixabay.com/sound-effects/waterfall-nature-sound-121190/`

poté pomocí krátkodobé Fourierovy transformace převeden z časové do frekvenční oblasti a dále bylo zpracováváno jeho modulové spektrum. Velikost spektrálního vektoru byla zmenšena na polovinu vynecháním záporných hodnot frekvencí, což při zpracování reálného signálu vzhledem k symetrii Fourierovy transformace nemá vliv. Data byla rozdělena na cíle a vstupy (prediktory), v kódu na `TARG` a `PRED`, kdy vstupy tvoří 8 vektorů zašuměného zvukového signálu. Síti byly tedy předkládány zašuměné signály (prediktory), na kterých se síť snaží naučit potlačení šumu a přiblížit se původnímu čistému signálu. Tato práce čerpala z [32] a [36], kde byl tento princip použit.



Obr. 5.2: Příklad spektrogramu vzorku čistého audia z datasetu



Obr. 5.3: Příklad spektrogramu vzorku zašuměného audia z datasetu

Pro přidaní šumu, provedení krátkodobé Fourierovy transformace a roztřídění dat z celého datasetu byla použita funkce `STFTfunction.m`. Takto rozdělená data byla následně použita pro trénink sítě. Nastavení tréninku sítě v Matlabu umožňuje nastavit velikost dávky, počet epoch, rychlost učení, náhodný výběr, validační frekvenci, validační data a poté parametry LearnRateDropFactor, LearnRateDropPeriod, které umožňují adaptivní úpravu rychlosti učení sítě během tréninku.

Vždy byly všechny 4 sítě natrénovány na stejné typy šumů a se stejnými tréninkovými podmínkami.

5.4 Testování sítě

Následně v souboru `TEST.m` byly natrénované sítě testovány pomocí 10 testovacích nahrávek, které byly vyjmuty z datasetu, na kterém byly sítě původně trénovány. Ke každé nahrávce byl přičten daný typ šumu, na který byla síť natrénována a poté byla převedena do frekvenční oblasti pomocí krátkodobé Fourierovy transformace. Stejným principem jako trénovací data byla testovací nahrávka rozdělena na 8 segmentů a poté předložena síti. Nakonec je pomocí zpětné Fourierovy tranformace zvukový signál převeden zpět do časové oblasti. Signál byl hodnocen pomocí různých metrik kvality zvukových signálů. Ze všech výsledků každého z 10 signálů byl poté vypočten průměr hodnot z každé metriky.

6 Vyhodnocení

6.1 Použité objektivní metriky

Zvukový signál s potlačeným šumem byl poté porovnáván za pomocí objektivních metrik pro kvalitu zvukového signálu. Bylo implementováno pět různých metrik a to SNR, SpeechQual, AudioQual, STOI a PESQ. Všechny jsou popsány níže. Pro usnadnění testování pomocí těchto metrik byla vytvořena funkce `Metrix.m`.

SNR

SNR (z anglického *Signal to Noise Ratio*) určuje poměr výkonu čistého zvukového signálu k výkonu šumu. Je to základní nejjednodušší metrika pro zjištění kvality audia. V prostředí Matlab je funkce `snr`, která tento poměr vypočítá.

SpeechQual

Tato metrika je použití Hansenovy metody pro odhadování kvality řeči. Jedná se o předpověď vnímané kvality řečového signálu k původnímu signálu za pomoci modelu sluchového vnímaní PEMO. Na frekvenční pásma je aplikována sada lineárních vah. Nakonec je vypočítán lineární křížový korelační koeficient reprezentací testovaných signálů. Hodnota se pohybuje v rozmezí od 0 do 1, kde 1 je hodnota původního čistého signálu.

AudioQual

Jedná se o implementaci různých metrik objektivního percepčního hodnocení kvality zvuku PEMO-Q. Výstupem je lineární křížový korelační koeficient dvojice vnitřních reprezentací neboli okamžitá míra kvality *PSM*, celková míra kvality *PSMt* a stupeň objektivního rozdílu *ODG*. *PSM* může dosahovat hodnot 0 až 1 s tím, že 1 je hodnota původního signálu. Obě tyto metody SpeechQual i AudioQual byly převzaty od [37].

STOI

Metrika STOI (z anglického *Short-Time Objective Intelligibility*) je míra srozumitelnosti čistého signálu, která je korelována s mírou srozumitelnosti degradovaných zvukových signálů. Výstupní hodnoty jsou od velmi degradovaný signál 0 po zcela nedegradovaný signál -1.[1]

[1] Testovací metrika byla převzata z <`https://ceestaal.nl/code/`>

PESQ

PESQ (z anglického *Perceptual Evaluation of Speech Quality*) je metoda porovnávající dva signály mezi sebou. Výstupem je hodnota nabývající od 0 do 5, kde 0 značí špatnou kvalitu signálu a 5 kvalitu výbornou.[2]

6.2 Srovnání neuronových sítí

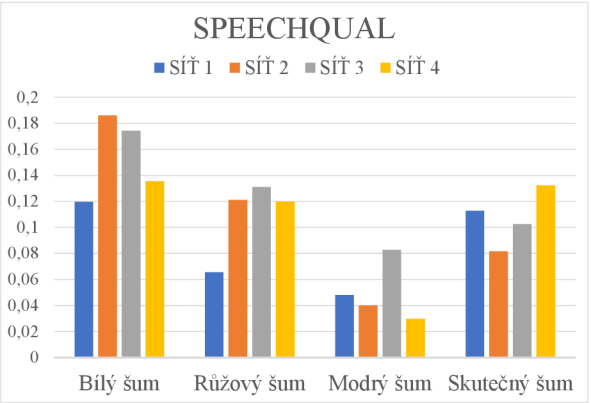
V následujících grafech na obrázku 6.1 lze vidět porovnání sítí naučených na různé typy šumů z hlediska již zmíněných metrik kvality zvukového signálu. V grafech je znázorněn rozdíl hodnot zvukového signálu s potlačeným šumem a původního zašuměného zvukového signálu. Tabulka 6.1 ukazuje průměrné hodnoty použitých testovacích metrik na deseti signálech s přidaným šumem.

Tab. 6.1: Hodnoty pro signál s přidaným šumem

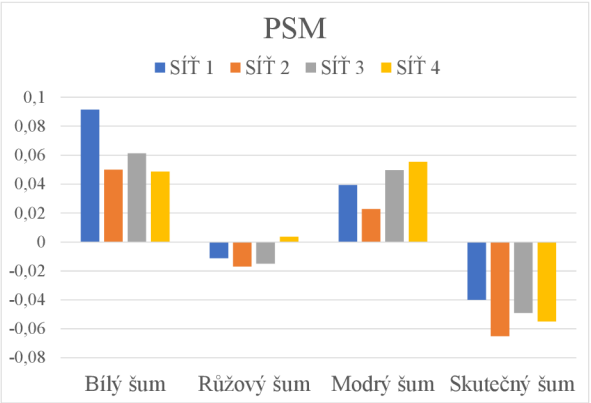
Metriky:	Bílý šum	Růžový šum	Modrý šum	Skutečný šum
SNR	10	10	10	10
SpeechQual	0,4756	0,4932	0,6147	0,4589
PSM	0,55	0,6027	0,5741	0,6405
PSMt	0,1969	0,2423	0,175	0,291
ODG	−3,8426	-3,8246	-3,851	-3,8041
STOI	0,9042	0,8688	0,9619	0,8106
PESQ	1,0623	1,1032	1,0573	1,1998

Dle porovnání SNR si trénované sítě nejlépe poradily s modrým šumem, konkrétně síť číslo 3. Naopak nejhůře si sítě uměly poradit se šumem reálného původu. Podobný závěr vyšel také podle metriky SpeechQual, s tím rozdílem, že zde dopadla nejlépe síť číslo 2. Nejhorší výsledky měly zvukové signály s přidaným modrým šumem. Podle hodnocení PEMO-Q si natrénované sítě poradily nejlépe s bílým a modrým šumem. Naopak u růžového a skutečného šumu došlo k zhoršení kvality zvukového signálu oproti zašuměnému signálu. Metrika STOI ukazuje, že došlo u všech typů šumů ke zhoršení kvality zvukového signálu. U metody PESQ vidíme zhoršení pouze při přičtení skutečného šumu.

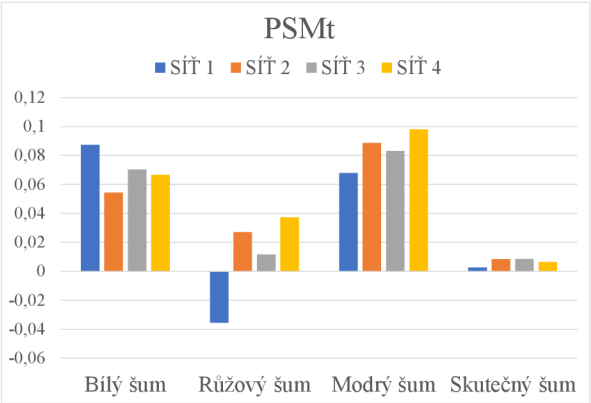
[2]Testovací metrika byla převzata z <`https://github.com/ludlows/pesq-mex`>



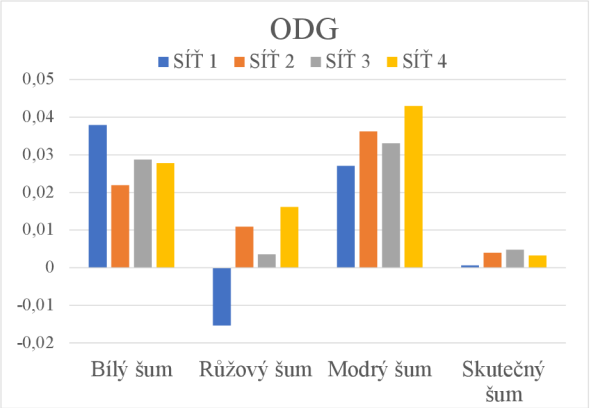
(a) Srovnaní podle SpeechQual



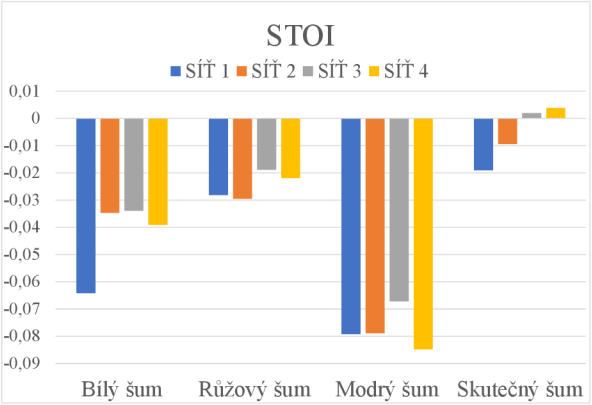
(b) Srovnaní podle AudioQual – PSM



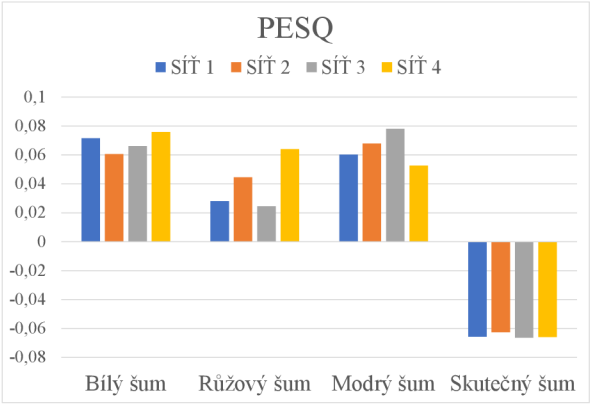
(c) Srovnaní podle AudioQual – PSMt



(d) Srovnaní podle AudioQual – ODG

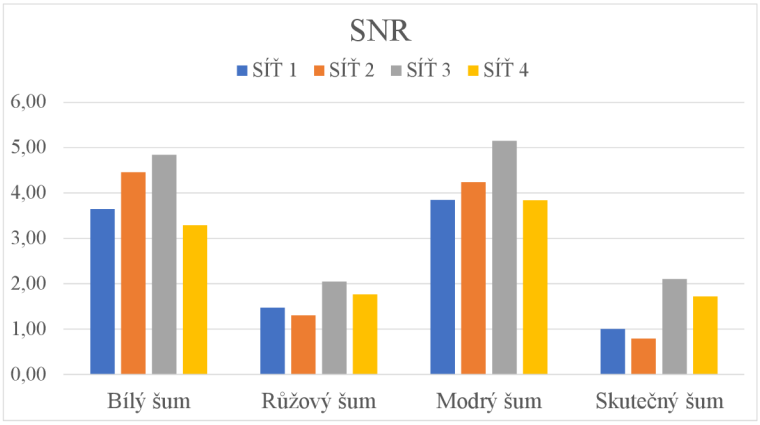


(e) Srovnaní podle STOI



(f) Srovnaní podle PESQ

Obr. 6.1: Grafy srovnání podle různých metrik



Obr. 6.2: Srovnání podle SNR

6.3 Srovnání metod

Jedním z cílů této práce bylo porovnání potlačení šumu pomocí neuronových sítí s jinou referenční metodou. Z důvodu jednoduchosti implementace byla vybrána metoda redukce šumu pomocí vlnkové transformace. Použití této metody v prostředí Matlab bylo realizováno pomocí aplikace *Wavelet signal denoiser*, která je součástí knihovny funkcí *Wavelet toolbox*. Tato aplikace umožňuje různá nastavení parametrů vlnkové transformace, konkrétně tedy typ a počet waveletů, metodu redukce šumu, úroveň vlnkové dekompozice a práh. Obě metody byly testovány na zvukové nahrávce vytvořené přímo pro tuto práci. Jedná se o krátkou větu *„Toto je testovací věta pro moji bakalářskou práci.“*, ke které byl přičten bílý šum na SNR o hodnotě 10 dB.

Redukce pomocí CNN

Redukce pomocí neuronové sítě byla prováděna pomocí sítě číslo 3, která byla vybrána pro její nejlepší výsledky v potlačení bílého šumu z hlediska SNR. Síť byla nejprve trénována na datasetu za různých trénovacích podmínek než bylo dosaženo nejlepšího výsledku při testu na jednom vzorku z trénovacího datasetu. Velikost dávky dat předkládané síti při jejím tréninku byla nastavena na 64. Nejlepšího výsledku bylo dosaženo při nastavení, které je vidět v levé části tabulky 6.2. Poté jí byl předložen vlastní nahraný zvukový signál s přidaným šumem.

Tab. 6.2: Nastavení tréninkových parametrů sítě a parametrů wavelet aplikace

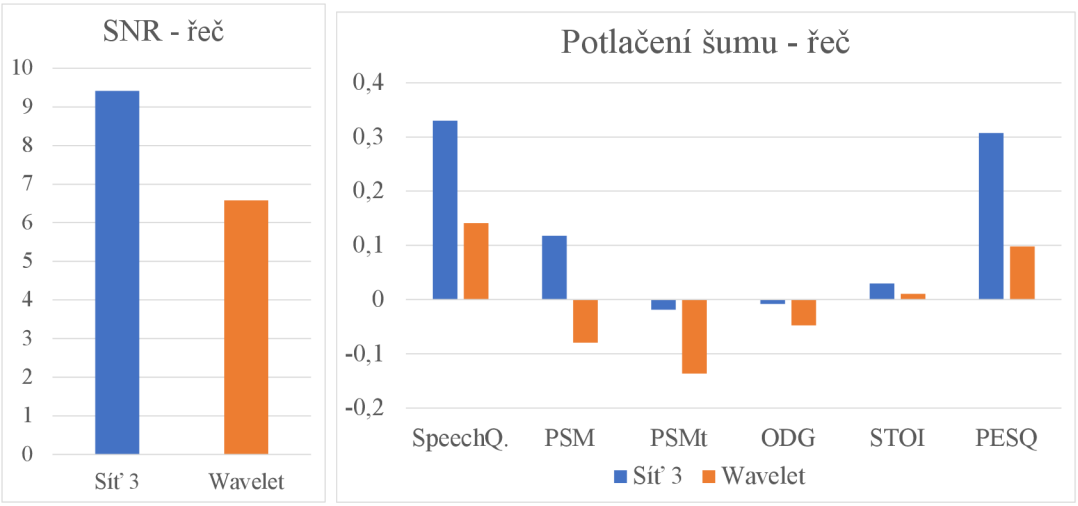
Neuronová síť		Wavelet aplikace	
Epochy	**1**	Wavelet	**db**
IniLearnRate	**3.00E-05**	Počet waveletu	**10**
Shuffle	**every epoch**	Metoda	**SURE**
LearnRDrop	**0.8**	Úroveň	**14**
LearnRDrPer	**1**	Práh	**Soft**

Redukce pomocí WT

Než byla testovací nahrávka vložena do aplikace *Wavelet signal denoiser*, byly vyzkoušeny kombinace všech parametrů na stejném vzorku z testovacího datasetu, na kterém byla testována neuronová síť, při hledání vhodných tréninkových parametrů. Nakonec byla nalezena nejlepší kombinace parametrů, která je uvedena v pravé části tabulky 6.2.

6.3.1 Výsledky

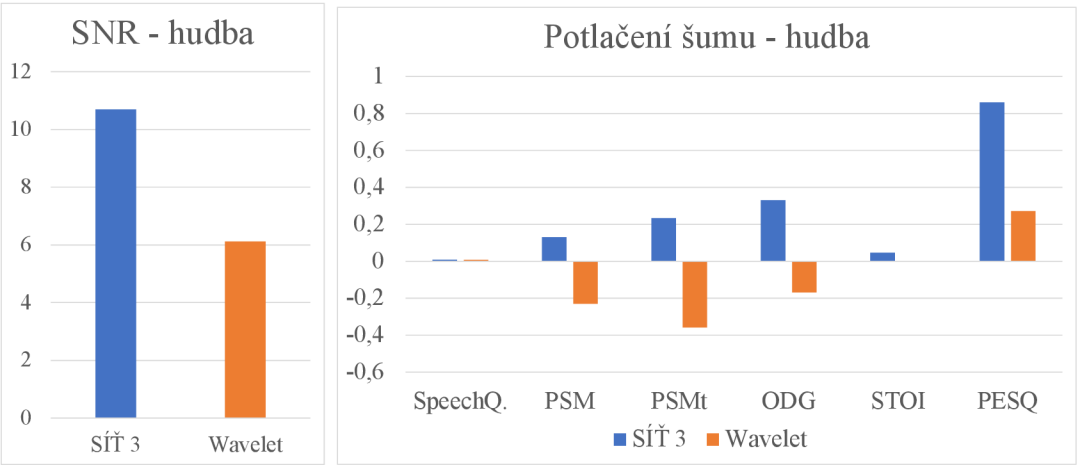
Výsledné dva zvukové signály s potlačeným šumem byly porovnány s původním signálem s přidaným šumem pomocí již dříve zmíněných metrik. Na grafech můžeme vidět opět rozdíly denoisovaného signálu a zašuměného signálu. Lze pozorovat, že neuronová síť dosáhla lepších výsledků než metoda užívající vlnkové tranformace. Podle metriky SNR došlo k zlepšení o 9,4 dB pomocí neuronové sítě, metoda vlnkové transformace dosáhla zlepšení pouze o 6,5 dB. V subjektivním poslechovém testu dosáhla lepšího výsledku také neuronová síť.



Obr. 6.3: Porovnání výsledků podle metriky na nahrávce řeči

6.4 Nahrávka hudby

Dalším cílem této práce bylo vyzkoušet umělou neuronovou síť na potlačení šumu v hudební nahrávce. Pro tento pokus byla zvolena část vlastní písničky s názvem „Dneska to bohužel nejde“. Nahrávka obsahuje bicí soupravu, elektrické kytary, baskytaru a zpěv. K této nahrávce byl přičten bílý šum a poté byla otestována opět pomocí neuronové sítě a vlnkové metody. Výsledný signál s potlačeným šumem byl otestován jak pomocí již zmíněných objektivních metrik, tak pomocí neformálního subjektivního poslechového testu. Výsledky metrik jsou znázorněny v grafu 6.4. I v tomto porovnání dosáhla neuronová síť mnohem lepších výsledků než druhá metoda. Stejného výsledku bylo dosaženo i dle neformálního poslechového testu.



Obr. 6.4: Porovnání výsledků podle metrik na nahrávce hudby

Závěr

Cílem této práce bylo čtenáře seznámit s odvětvím strojového učení, do kterého spadají umělé neuronové sítě, a jejich využitím v oblastech potlačení a redukce šumu včetně jejich praktické aplikace na zvukových signálech.

V první části byly tyto sítě, jejich druhy a uplatnění ve zvukových signálech teoreticky popsány. Byla zde věnována celá podkapitola konvolučním neuronovým sítím, které jsou posléze využity v praktické části. Popsány byly i některé funkce např. *pooling* používaný v těchto sítích. Čtenář byl seznámen se základním rozdělením šumu a s některými způsoby pro jeho redukci. Na vybrané metody pro zpracování zvukových signálů se zde zaměřuje celá kapitola 3, kde je popsány některé druhy Fourierovy transformace a metoda segmentace.

V praktické části byly porovnány základní implementace neuronových sítí na potlačení šumu u jednoduchých černobílých obrázků ve dvou vývojových prostředích. První síť byla implementována v programovacím jazyce Python, druhá v matematickém prostředí Matlab. Následovalo shrnutí a vyhodnocení rozdílů a podobností v orientaci a intuitivnosti v jednotlivých programovacích jazycích 4.1. Srovnány byly jak podobnosti a rozdíly ve tvorbě a učení umělých neuronových sítí, tak jejich výsledky na testovaném odšumovaném obrázku. Úspěšnost testovaných prostředí se lišila. Mnohem lepších výsledků dosáhl Python, na druhou stranu v Matlabu byla práce intuitivnější. Proto byla výsledná síť navrhována a testována právě v tomto prostředí.

Následně byl představen model umělé konvoluční neuronové sítě, podle kterého byly navrženy čtyři různé architektury neuronových sítí. Poté byly natrénovány na stejném datasetu a za stejných trénovacích podmínek. To bylo provedeno pro čtyři různé druhy šumů, celý tento proces popisuje kapitola 5. Porovnání a hodnocení všech architektur za pomocí různých metrik hodnocení kvality zvukového signálu je zobrazeno a popsáno v kapitole 6.2. Na základě těchto testů byla poté vybrána síť číslo 3, která dosáhla nejlepších výsledků z dle metriky SNR. Další testování probíhalo z časových důvodů pouze s přidaným bílým šumem. Tato síť byla poté porovnána s metodou využívající vlnkové transformace na redukci šumu v audio signálu. Obě metody byly testovány na vlastní nahrávce řeči, která byla nahrána za účelem této práce a na krátkém úseku hudební nahrávky. Neuronová síť si s redukcí šumu dle použitých metrik poradila lépe než druhá metoda. Dle SNR dosáhla mnohem lepších výsledků než u původních testovaných nahrávek z datasetu, což je nejspíše také způsobeno větší kvalitou vlastní nahrávky. U testovaných nahrávek se hodnoty SNR zlepšili nejvíce o 5 dB, zato u vlastní nahrávky dosáhla zlepšení o 9,4 dB a u hudební nahrávky až o 10,69 dB.

Závěrem této práce lze říci, že využití neuronové sítě v oblasti redukce šumu má

velký potenciál, jak v oblasti řeči tak v oblasti hudby. V rámci této práce bylo provedeno několik základních testů na redukci šumu, které ukázaly pozitivní výsledky neuronových sítí. Je třeba poznamenat, že výsledky testování slouží spíše jako orientační vzhledem k omezenému časovému rámci, který byl k dispozici. Z hlediska rozsahu této práce se nebylo možné věnovat dalšímu testování, což by umožnilo širší prozkoumání a lepší optimalizaci neuronových sítí, například trénováním na větším objemu dat.

Literatura

[1] VOLNÁ, Eva. *Neuronové sítě I* [online]. Ostrava: 2002, 85 s. 2. vydání. [cit. 5. 11. 2022]. Dostupné z URL: <https://web.osu.cz/~Volna/Neuronove_site_skripta.pdf>.

[2] BLÁHA, Milan. *Matematická biologie učebnice: Koncept umělé neuronové sítě.* [online]. [cit. 5. 11. 2022]. Dostupné z URL: <https://rb.gy/uagup4>.

[3] MIKULA, Vladimír. *Umělé neuronové sítě a fuzzy systémy* [online]. Kunovice: EPI, s.r.o, 2003, 60 s. [cit. 5. 11. 2022]. Dostupné z URL: <https://docplayer.cz/29509634-Umele-neuronove-site-a-fuzzy-systemy.html>.

[4] VONDRÁK, Ivo. *Neuronové sítě* [online]. Ostrava: VŠB - Technická univerzita Ostrava, 1994. 56 s. [cit. 6. 11. 2022]. Dostupné z URL: <http://vondrak.cs.vsb.cz/download/Neuronove_site.pdf>.

[5] CARTY, David. *Training, Validation and Testing Data Explained - Applause. Release Faster, With Confidence - Applause* [online]. Copyright © 2022 Applause App Quality, Inc. [cit. 09. 12. 2022]. Dostupné z: <https://www.applause.com/blog/training-data-validation-data-vs-test-data>

[6] Autor neznámý. *The Difference Between Epoch and Iteration in Neural Networks* [online]. by Baeldung, November 15, 2022. [cit. 9. 12. 2022] Dostupné z: <https://www.baeldung.com/cs/neural-networks-epoch-vs-iteration#3-batch>

[7] SOVKA, Michal. *Umělé neuronové sítě pro učení robotů.* [online]. Praha, 2009, 52 s. [cit. 8. 11. 2022] Dostupné z: <https://vskp.vse.cz/13834_umele_neuronove_site_pro_uceni_robotu.> Bakalářská práce. Vysoká škola ekonomická v Praze.

[8] Více autorů. *Deep Learning* [online]. The MIT Press, Cambridge, Massachusetts, 2016. Dostupné z: <http://www.deeplearningbook.org>.

[9] PILÁT, Martin. *Neuronové sítě - konvoluční sítě a zpracování obrazu* [online]. [cit. 16. 11. 2022] Dostupné z: <https://martinpilat.com/cs/prirodou-inspirovane-algoritmy/neuronove-site-konvolucni-site-zpracovani-obrazu

[10] DUMOULIN Vincent, VISIN Francesco. *A guide to convolution arithmetic for deep learning* [online]. ArXiv e-prints, March, 2016. [cit. 17. 11. 2022] Dostupné z: <https://github.com/vdumoulin/conv_arithmetic

[11] ZACHA, Jiří. *Konvoluční neuronové sítě pro klasifikaci objektů z LiDARových dat* [online]. Praha, 2019, 39 s. [cit. 19. 11. 2022] Dostupné z: <https://dspace.cvut.cz/bitstream/handle/10467/82351/F3-BP-2019-Zacha-Jiri-Konvolucni_neuronove_site_pro_klasifikaci_objektu_z_LiDARovych_dat.pdf> Bakalářská práce. České vysoké učení technické v Praze.

[12] BADR, Will. *Auto-Encoder: What Is It? And What Is It Used For? (Part 1)* [online]. Dostupné z: <https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726>

[13] Autor neznámý. *What is U-Net?* [online]. Educative: Interactive Courses for Software Developers. Copyright ©2022 Educative, Inc. All rights reserved [cit. 21. 11. 2022]. Dostupné z: <https://www.educative.io/answers/what-is-u-net>

[14] VENKRBEC, Tomáš. *Generování obličejů s pomocí podmíněných generativních neuronových sítí* [online]. Brno, 2020 [cit. 25. 11. 2022]. Dostupné z: <http://hdl.handle.net/11012/191569> Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav počítačové grafiky a multimédií.

[15] GAVLASOVÁ, Radka. *Vztah emocí a intonačních křivek* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 68 s. Bakalářská práce.

[16] MACHURON, Charles-Louis. *AIVA: The Artificial Intelligence Composing Classical Music* [online]. Silicon Luxembourg magazine, 2016. [cit. 9. 12. 2022] Dostupné z: <https://www.siliconluxembourg.lu/aiva-the-artificial-intelligence-composing-classical-music/>

[17] AIVA (Pierre Barreau). *File:Opus 3 for Piano Solo.pdf* [online]. [cit. 9. 12. 2022] Dostupné z: <https://upload.wikimedia.org/wikipedia/commons/7/78/Opus_3_for_Piano_Solo.pdf>

[18] MARAFIOTI, Andrés, HOLIGHAUS, Nicki, MAJDAK, Piotr, PERRAUDIN, Nathanaël. *Audio Inpainting of Music by Means of Neural Networks* [online]. Project Merlin, 2022. [cit. 9. 12. 2022] Dostupné z:<https://arxiv.org/pdf/1810.12138.pdf>

[19] Aäron van den OORD, DIELEMAN, Sander. *WaveNet: A generative model for raw audio* [online]. DeepMind, 2016. [cit. 10. 12. 2022] Dostupné z: <https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio>

[20] Colors of noise - Wikipedia. [online]. Dostupné z: <https://en.wikipedia.org/wiki/Colors_of_noise>

[21] BAZOYAN, Anna. *White Noise and Other 5 Noise Colors* [online]. Krisp Technologies, Inc. All rights reserved, 2022. [cit. 18. 03. 2023] Dostupné z: <https://krisp.ai/blog/did-you-know-that-noise-has-a-color/>

[22] CONNAGHAN, TYLER. *The Difference Between Types Of Noise* [online]. Online Audio Mastering by Grammy Winning Engineers, eMastered, 2021. [cit. 21. 03. 2023] Dostupné z: <https://emastered.com/blog/different-types-of-noise>

[23] EISENSTEINOVÁ, Gabriela, SEDLÁČEK, Miloš. *Využití matlabu k potlačování aditivního šumu pomocí filtrace a pomocí vlnkové transformace* [online]. České vysoké učení technické v Praze, Fakulta elektrotechnická, katedra měření, 2001. [cit. 11. 12. 2022] Dostupné z: <http://dsp.vscht.cz/konference_matlab/matlab01/blaska1.pdf>

[24] KUMAR, Madam Aravind, CHARI, Kamsali Manjunatha. *Noise Reduction Using Modified Wiener Filter in Digital Hearing Aid for Speech Signal Enhancement* Journal of Intelligent Systems, vol. 29, no. 1, 2020, str. 1360-1378. [cit. 11. 12. 2022] Dostupné z: <https://doi.org/10.1515/jisys-2017-0509>

[25] BOUDRAA, Abdel-O, CEXUS, Jean-Christophe. *Denoising via empirical mode decomposition* [online]. Second International Symposium on Communications, Control and Signal Processing, (2006). [cit. 11. 12. 2022] Dostupné z: <https://www.researchgate.net/publication/47760355_Denoising_via_empirical_mode_decomposition>

[26] BOLL, Steven. *Suppression of acoustic noise in speech using spectral subtraction* [online]. in IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 27, no. 2, str. 113-120, April, 1979. doi: 10.1109/TASSP.1979.1163209. Dostupné z: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1163209>

[27] DAUTOV, Çiğdem Polat and ÖZERDEM, Mehmet Siraç. *Wavelet transform and signal denoising using Wavelet method* [online]. 26th Signal Processing

and Communications Applications Conference (SIU), 2018. [cit. 25. 4. 2023]. Dostupné z <https://ieeexplore.ieee.org/document/8404418>

[28] SMÉKAL, Zdeněk. *Analýza signálů a soustav*: Brno: Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací, 2012. ISBN 978-80-214-4453-9.

[29] SMÉKAL, Zdeněk. *Zpracování řeči*: Brno: Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací, 2013. ISBN 978-80-214-4896-4.

[30] GERON, Aurélien. *Hands-on machine learning with scikit-learn, keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems (2nd ed.).* [online] Sebastopol: O'Reilly Media, 2019. ISBN: 978-14-920-3264-9. Dostupné z: <https://t.ly/Es35>

[31] *Deep-Learning-Projects/Deep CNN Autoencoder - Denoising Image.ipynb at main · aswintechguy/Deep-Learning-Projects · GitHub. GitHub: Let's build from here · GitHub* [online]. Copyright © 2022 GitHub, Inc. [cit. 05. 12. 2022]. Dostupné z: <https://rb.gy/wjsjzs>

[32] PARK Se Rim, LEE Jinwon. *A Fully Convolutional Neural Network for Speech Enhancement* [online] Park2016AFC, ArXiv, 2016, abs/1609.07132. Dostupné z: <https://www.semanticscholar.org/paper/A-Fully-Convolutional-Neural-Network-for-Speech-Park-Lee/9ed8e2f6c338f4e0d1ab0d8e6ab8b836ea66ae95?p2df>

[33] MICHELASHVILI, Michael, WOLF, Lior. *Audio Denoising with Deep Network Priors* ArXiv abs/1904.07612 (2019) [cit. 10. 12. 2022] Dostupné z: <https://arxiv.org/pdf/1904.07612.pdf>

[34] TAN, Ke, WANG, DeLiang. *A Convolutional Recurrent Neural Network for Real-Time Speech Enhancement* [online]. 10.21437/Interspeech.2018-1405. [cit. 10. 12. 2022] Dostupné z: <https://www.researchgate.net/publication/325542192_A_Convolutional_Recurrent_Neural_Network_for_Real-Time_Speech_Enhancement>

[35] PARK, Samuel Kyung Won. *Comparison of Neural Networks and Least Mean Squared Algorithms for Active Noise Canceling* (2018). All Theses. 2920. [cit. 10. 12. 2022] Dostupné z: <https://tigerprints.clemson.edu/all_theses/2920>

[36] *Denoise Speech Using Deep Learning Networks* [online]. MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink Copyright 1994-2023. [cit. 04. 04. 2023]. Dostupné z: <https://www.mathworks.com/help/deeplearning/ug/denoise-speech-using-deep-learning-networks.html>

[37] *Dictionary Learning for Sparse Audio Inpainting* [online]. Österreichische Akademie der Wissenschaften. Copyright © Copyright OEAW. [cit. 19. 04. 2023]. Dostupné z: <https://www.oeaw.ac.at/en/isf/forschung/fachbereiche-teams/mathematik/dictionary-learning-for-sparse-audio-inpainting>

Seznam symbolů a zkratek

ANN	Umělé neuronové sítě
CNN	Konvoluční neuronové sítě
GAN	Generativní soupeřící sítě
AIVA	Artificial Intelligence Virtual Artist
STFT	Krátkodobá Fourierova transformace
MSE	Mean squared error
PEAQ	Perceptual Evaluation of Audio Quality
SNR	Signal to noise ratio
PEMO-Q	Perception Model-Based Quality
STOI	Short-Time Objective Intelligibility
PESQ	Perceptual Evaluation of Speech Quality
WT	Wavelet transformace

Seznam příloh

A Obsah elektronické přílohy

Příloha je složená z 8 souborů, z nichž ty s názvem začínajícím `Final...` obsahují implementaci demonstrační neuronové sítě ve dvou prostředích. Poté příloha obsahuje soubory `TRAIN.m` a `TEST.m`, které představují kódy pro trénink a testování sítě na redukci šumu v audio signálu. Pomocí souboru `TEST_Signal.m` lze redukci vyzkoušet na vlastním zvukovém souboru. Nakonec jsou zde obsaženy dva soubory: pomocná funkce `STFTfunction.m` a funkce metrik `Metrix.m`. Z důvodu velikosti datasetu s obrázky, datasetu nahrávek řeči a natrénované neuronové sítě `v03_FINAL.mat` je tato část přílohy dostupná online. Odkaz je obsažen v souboru `Priloha_Drive.txt`. Na odkazu se nacházejí dva adresáře, `CNNs` a `MATvsPT`. Soubory pro test sítě obsahuje první složka, tedy `CNNs`. Nacházejí se zde i architektury testovaných neuronových sítí označené názvy `CASCADE.mlx` až `CASCADE04.mlx`, jejich schémata a funkce jednotlivých metrik. Pro správnou funkčnost je vhodné nahrát všechny soubory do jednoho adresáře s hlavním testovacím kódem `TEST_Signal.m`. Ve druhé složce je poté dataset, který byl využit při porovnávání dvou prostředí. Je třeba upozornit, že kódy byly optimalizovány a testovány výhradně pro operační systém MacOS a nelze s jistotou garantovat jejich správnou funkčnost na jiných operačních systémech.

- `Příloha` ..... kořenový adresář přiloženého archivu
  - `Final_Matlab_DENOISE_Mnist.m` ..... srovnávací kód v Matlabu
  - `Final_Python_DENOISE_Mnist.py` ..... srovnávací kód v Pythonu
  - `TRAIN.m` ..... kód pro trénink neuronové sítě
  - `TEST.m` ..... kód pro test neuronové sítě
  - `TEST_Signal.m` ..... kód pro test neuronové sítě na vlasní nahrávce
  - `STFTfunction.m` ..... pomocná funkce
  - `Metrix.m` ..... funkce metrik
  - `Priloha_Drive.txt` ..... textový dokument s odkazem na datasety