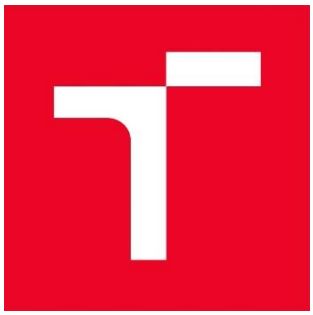


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ENERGETICKÝ ÚSTAV
ENERGY INSTITUTE

VYUŽITÍ STROJOVÉHO UČENÍ PŘI DIAGNOSTICE VODNÍCH STROJŮ
USE OF MACHINE LEARNING FOR WATER MACHINES DIAGNOSIS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. Albert Kindl**
AUTHOR

VEDOUCÍ PRÁCE **doc. Ing. Vladimír Habán, Ph.D.**
SUPERVISOR

BRNO 2025

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Energetický ústav
Student:	**Bc. Albert Kindl**
Studijní program:	Energetické a termofluidní inženýrství
Studijní obor:	Fluidní inženýrství
Vedoucí práce:	**doc. Ing. Vladimír Habán, Ph.D.**
Akademický rok:	2024/25

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Využití strojového učení při diagnostice vodních strojů

Stručná charakteristika problematiky úkolu:

Provoz hydraulického stroje je možno sledovat pomocí signálu, který tento stroj vyzařuje do okolí. Tyto vyzařované signály je možno měřit pomocí snímačů tlaku, zrychlení, zvuku a akustické emise. Úkolem diplomové práce bude z již dříve měřených dat na čerpadle navrhnout systém strojového učení, který dokáže odhalit uměle vytvořenou poruchu na lopatce tohoto stroje.

Cíle diplomové práce:

Navrhnout systém strojové učení pro stanovení poruchy na hydraulické stroji.
Stanovit přesnost této diagnostiky a pokusit se odhadnout její limity a omezení.

Seznam doporučené literatury:

ASSZONYI, Ondřej. Zpracování naměřených signálů z kavitačních experimentů. Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/125083. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Energetický ústav. Vedoucí práce Pavel Rudolf.

ŠEBEK, Denis. Strojové učení při diagnostice vodních strojů. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, Energetický ústav, 2023, 117 s. Diplomová práca. Vedúci práce: doc. Ing. Vladimír Habán, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2024/25

V Brně, dne

L. S.

prof. Ing. Jiří Pospíšil, Ph.D. ředitel ústavu	doc. Ing. Jiří Hlinka, Ph.D. děkan fakulty

ABSTRAKT
Tato diplomová práce se zabývá využitím strojového učení k rozpoznání poruchy na vodním stroji. Cílem práce je navrhnout funkční model neuronové sítě k tomuto účelu a odhadnout jeho omezení. Byla využita data poskytnutá vedoucím práce, která byla naměřena v hydraulické laboratoři VUT. Pocházela z provozu čerpadla v čerpadlovém a turbínovém režimu. Vada byla simulována provrtáním lopatky. Při tvorbě modelů byla použita především knihovna TensorFlow. Problém byl řešen výhradně jako binární klasifikace. Bylo ověřeno několik variant modelů, kdy byl každý kriticky zhodnocen. Byly navrženy varianty pracující s konvoluční sítí zpracovávající surová data a MEL-spektrogramy, dále také plně propojená síť pracující se statistickými charakteristikami částí dat. Nejlepších výsledků obecně dosáhla varianta využívající konvoluční neuronovou síť a MEL-spektrogramy. V práci použitá metodika se snaží alespoň částečně přiblížit schopnost modelu fungovat na datech ze skutečného provozu.

Klíčová slova
vodní stroj, diagnostika, prediktivní údržba, neuronová síť, binární klasifikace, porucha, akustická emise, mikrofon, čerpadlo, turbína

ABSTRACT
This thesis focuses on the application of machine learning techniques to detect a fault on a water machine. The goal of the thesis is to design a functional neural network model for this purpose and to estimate its limitations. The dataset, provided by the thesis supervisor, which was measured in the hydraulic laboratory of the BUT, was used. It came from the operation of the pump in pumping and turbine mode. The defect was simulated by drilling through a blade. The TensorFlow library was mainly used in the development of the models. The problem was solved as a binary classification problem only. Several model variants were evaluated, with each being critically assessed. Variants working with a convolutional network processing raw data and MEL-spectrograms were designed, as well as a fully connected network working with statistical features extracted from segments of the data. In general, the best results were achieved by the variant using a convolutional neural network and MEL-spectrograms. The methodology used in this thesis attempts to at least partially approximate the model's ability to operate in real-world operational data.

Key words
water machine, diagnostics, predictive maintenance, neural network, binary classification, failure, acoustic emission, microphone, pump, turbine

BIBLIOGRAFICKÁ CITACE

KINDL, Albert. *Využití strojového učení při diagnostice vodních strojů*. Online, diplomová práce. Vladimír HABÁN (vedoucí práce). Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2025. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/169443. [cit. 2025-04-24].

PROHLÁŠENÍ

Prohlašuji, že jsem diplomovou práci na téma „Využití strojového učení při diagnostice vodních strojů“ vypracoval samostatně pod odborným vedením doc. Ing. Vladimíra Habána, Ph.D. Také prohlašuji, že všechny zdroje obrázkových i textových informací, ze kterých jsem čerpal, jsou řádně uvedené a citované v seznamu použitých zdrojů.

22. 05. 2025	Bc. Albert Kindl
Datum	*Jméno a příjmení*

PODĚKOVÁNÍ

Děkuji doc. Ing. Vladimíru Habánovi, Ph.D. za odborné vedení, cenné rady a připomínky. Dále děkuji za jeho ochotu a možnosti práci konzultovat kdykoliv bylo potřeba a za lidský a přátelský přístup nejen v rámci vedení práce, ale během celého studia. Dále děkuji rodině a přátelům za jejich podporu v průběhu studia. Jmenovitě pak děkuji dobrému kamarádovi Ing. Michalu „Lil Mikey“ Reinovi za jeho podporu při tvorbě této práce.

OBSAH

ÚVOD

Využití neuronových sítí v nynější době zažívá slibný vzestup. Ukazuje se, že v málokterém odvětví průmyslu se pro strojové učení nenajde uplatnění. V energetice tomu není jinak, a obzvláště pak v té vodní. Vodní energetika nabízí široké spektrum problémů, které lze řešit efektivně za pomocí těchto moderních algoritmů. Mezi nimi je třeba i optimalizace provozu. Tím se zabývá například projekt HYDROGRID Insight, který nabízí komplexní systém pracující se strojovým učením. Tento sytém umožňuje například prediktivní modelování přítoků nebo cen elektrické energie, a to v krátkodobém, střednědobém i dlouhodobém horizontu. Dále umožňuje automatizaci plánování výroby nebo třeba optimalizaci provozu kaskád s vícero nádržemi [1].

Další využití strojového učení se nachází v prediktivní údržbě vodních strojů. Vodní stroje, jakými jsou vodní turbíny nebo čerpadla, při provozu emitují mnoho měřitelných signálů. Od hluku přes vibrace až po tlakové pulzace. Díky modelům neuronových sítí je možné tyto signály efektivně zpracovávat a v reálném čase vyhodnocovat nově vzniklé. To umožňuje včasné odhalení poruch či potřebu údržby. V průmyslové praxi se prediktivní údržbě pomocí neuronových sítí věnuje například společnost Voith se svým projektem OnCare.Acoustic, což je systém pracující s akustickými signály získanými mikrofony umístěnými na různých místech elektrárny, kdy dokáže rozeznat, zda aktuální chování stroje odpovídá chodu s výskytem anomálie, či nikoliv. V takovém případě dokáže systém upozornit technika elektrárny. Systém se s dalšími měřeními a získanými daty dále zdokonaluje. Díky systému jsou sníženy nároky na lidské zaměstnance elektrárny. Tento systém byl spuštěn jako pilotní v roce 2018 na islandské vodní elektrárně Búðarháls, která je provozována islandskou energetickou společností Landsvirkjun [2]. Jedná se o moderní vodní elektrárnu spuštěnou v roce 2014 se dvěma Kaplanovými turbínami o celkovém výkonu 95 MW pracujícími při spádu 40 m [3].

Právě detekcí anomálie, respektive poruchy se zabývá i tato diplomová práce. Konkrétně se zabývá zpracováním již naměřených záznamů signálů ze snímačů akustické emise, mikrofonu, snímače zrychlení a dvou snímačů tlaku. Cílem práce je na těchto datech sestrojit funkční systém strojového učení a pokusit se určit jeho omezení. Práce se pokusí odpovědět na otázky, jaký přístup a druh neuronových sítí je nejvhodnější pro řešení tohoto problému. Vzhledem ke struktuře obdržených dat je problém řešený jako binární klasifikace. V práci bude nejprve zpracován krátký úvod k neuronovým sítím jako takovým, bude následovat popis problému a samotné řešení neuronových sítí.

Jedná se o téma poměrně nové, rozhodně aktuální a relativně málo probádané, proto může být tato diplomová práce oboru přínosná ať už v odhalení cest průchozích či slepých.

1 Úvod do neuronových sítí

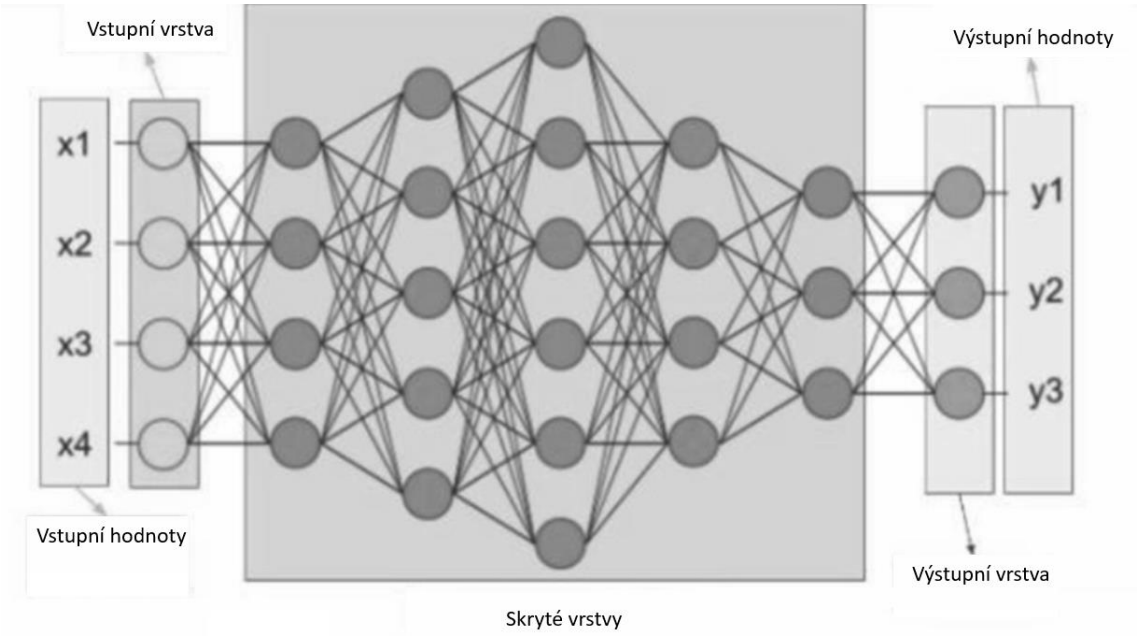
Umělá inteligence je obecný pojem, který zahrnuje veškeré systémy, které svým fungováním napodobují fungování lidské mysli. Podmnožinou umělé inteligence je potom strojové učení. Strojové učení zahrnuje algoritmy, které na základě vstupních dat dokážou předpovídat určité výstupy. Konkrétním příkladem takového algoritmu jsou neuronové sítě. Umělé neuronové sítě jsou inspirovány lidským mozkem, kde jsou jednotlivé neurony vzájemně propojeny. Pokud má neuronová síť více než tři vrstvy (viz další podkapitola), potom se hovoří o deep learningu neboli hloubkovém učení. [4]

1.1 Struktura neuronových sítí

Konkrétní struktura neuronové sítě závisí na jejím konkrétním typu. Mezi nejzákladnější typy patří síť plně propojená, která bude využita k vysvětlení hlavních principů struktur. Jednotlivé typy modelů neuronových sítí jsou popsány dále v práci v kapitole 2.

1.1.1 Vrstvy neuronové sítě

Neuronové sítě se skládají z minimálně tří vrstev neuronů. Každá neuronová síť má vstupní vrstvu a vrstvu výstupní. Vrstva nebo vrstvy mezi nimi se pak nazývají skryté vrstvy. Vstupní vrstva slouží k přijímání vstupních dat. Počet neuronů ve vstupní vrstvě se odvíjí od počtu vstupních atributů – pokud bude například neuronová síť vyhodnocovat tlak, průtok a teplotu tekutiny, budou ve vstupní vrstvě tři neurony. Výstupní vrstva slouží k vyhodnocení a zpracování výsledků, které poskytují skryté vrstvy tak, aby s nimi mohl dále uživatel nebo systém pracovat. Výstupní vrstva v rámci interpretace výstupních dat obvykle obsahuje také aktivační funkci, a to v závislosti na typu úlohy, respektive požadovaného výstupu. Může být používána například aktivační funkce *softmax* pro úlohy vícetřídní klasifikace, kdy má výstup podobu pravděpodobnostního vektoru, funkce *sigmoid* pro binární klasifikaci a podobně. Skryté vrstvy přijímají jako vstupy data z neuronů předchozí vrstvy. Za pomoci optimalizace vah spojů a posuny (anglicky *bias*) mezi neurony dochází k určení relevance vstupů a díky aplikaci aktivačních funkcí se do řešené úlohy přidává nelinearita, díky které je možné, aby došlo k rozpoznání složitých vzorů a ke zpřesnění výsledků. V praxi je z hlediska efektivní práce s pamětí výhodné, aby počet skrytých vrstev odpovídal mocnině 2. [5]

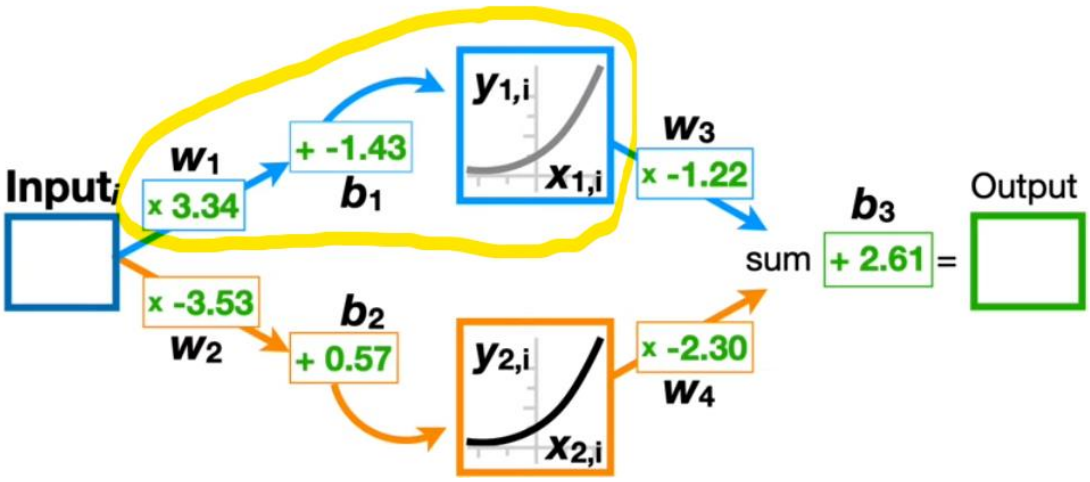


Obrázek 1-1: Plně propojená neuronová síť, upraveno podle [6]

Na Obrázku 1-1 je znázorněna plně propojená neuronová vrstva – každý neuron je propojen se všemi neurony z předchozí a následující vrstvy. Tato konkrétní neuronová vrstva má 5 skrytých vrstev.
Volba počtu skrytých vrstev je často iterační proces a určování musí probíhat s ohledem na konkrétní charakter řešeného problému. Obecně lze tvrdit, že časová náročnost učení neuronové sítě roste s rostoucím počtem skrytých vrstev. Zároveň se však s vyšším počtem skrytých vrstev také obvykle zvyšuje přesnost dosažených výsledků. Volba počtu skrytých vrstev musí probíhat také s ohledem na přeučení a nedoučení (anglicky *overfitting* a *underfitting*), což je podrobněji rozebráno níže v této práci. [6]

1.1.2 Neuron

Jako neuron se obvykle označuje vše, co v dané vrstvě provádí operace s daty ze vstupu či obecně předchozí vrstvy. Jako součást neuronu tedy může být považována aktivační funkce, posun a váha spoje.



Obrázek 1-2: Jednoduchá neuronová síť s jednou skrytou vrstvou a žlutě označeným neuronem, upraveno podle [5]

1.1.3 Aktivační funkce

Důležitost aktivační funkce spočívá ve vnášení nelinearity do řešení. Díky průběhům aktivačních funkcí se příslušné neurony aktivují pouze v případě, že jsou jejich vstupy pro sledovaný problém relevantní. Jak je vidět na obrázku 1-2, bez aktivačních funkcí by data ze vstupu až po výstup byla pouze násobena váhami, byly by k nim přičítány posuny a byla by sčítana mezi sebou. Docházelo by tedy pouze k lineární kombinaci vstupů, což by znamenalo řešení problému lineární regresí, což je řešení, které není schopné rozpoznat složitější vzorce. Existuje mnoho různých aktivačních funkcí, přičemž každá je vhodná k jinému použití [5] [7]. Kromě vnášení nelinearity do řešení za co nejmenšího nárůstu časové náročnosti by dobře navržená aktivační funkce měla také udržovat šíření gradientu (anglicky *gradient flow*), tedy předcházet jevům exploze gradientu a zmizení gradientu (anglicky *gradient exploding* a *gradient vanishing*). Aktivační funkce by také měla zachovávat distribuci dat [8].

Volba aktivační funkce probíhá na základě zkušeností programátora a často se jedná o upravovaný aspekt, který je upravován tak, aby se dospělo k co nejlepšímu výsledku. Zároveň platí, že jeden model může v určitých částech využívat různé aktivační funkce, musí být však zajištěna dopřednost modelu a musí být podchyceno cyklení uvnitř těchto částí. Toho je využíváno v případě, že zpracovávaná data jsou rozdílného typu, například neuronová síť zpracovává dataset skládající se z dvojic obrázek + číselná hodnota atp. Využití různých aktivačních funkcí v jednotlivých vrstvách neuronové sítě je pak běžná praxe [5].

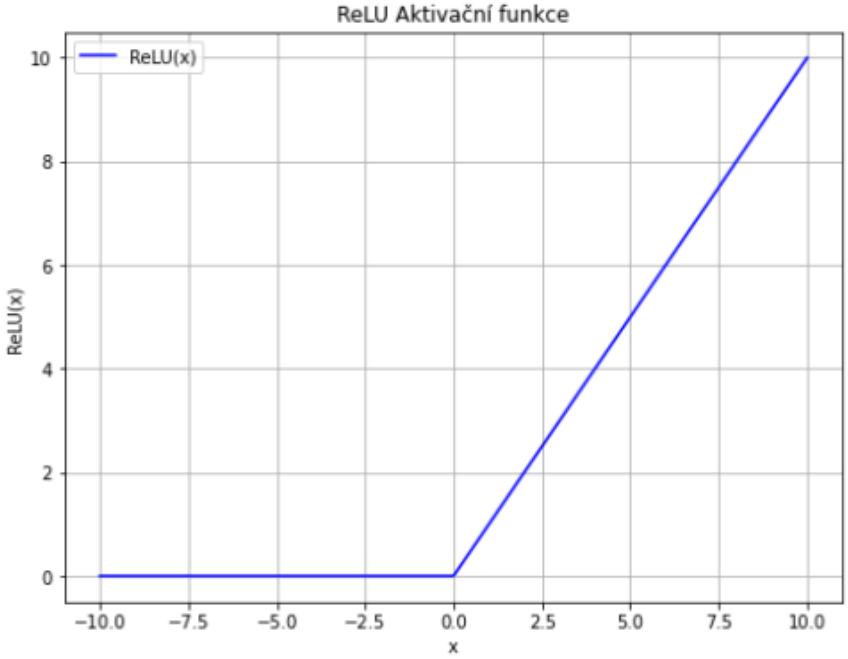
Dále v práci je zpracován přehled nejfrekventovaněji používaných aktivačních funkcí:

- ReLU

Funkce *ReLu* je definována následujícím vztahem:

$$ReLU(x) = \max(0, x) \tag{1-1}$$

Jak je zřejmé z předpisu, funkce nabývá nuly v případě, že je argument nekladné číslo. V opačném případě je průběh funkce lineární. Obor hodnot funkce *ReLU* je tedy <0;∞).



Obrázek 1-3: Průběh aktivační funkce ReLU(x) vykreslené pomocí knihovny matplotlib

Funkce *ReLU* je vhodná, pokud má být výstup z neuronu číselná hodnota. Výhodou *ReLU* a důvodem jejího frekventovaného používání je linearita v případě nezáporného argumentu. To má za následek jednak výpočetní nenáročnost, jednak urychlení konvergence ztrátové funkce ke svému minimu, a tedy urychlení učení neuronové sítě. Díky linearitě v kladných argumentech není *ReLU* náchylná na mizení gradientu jako v případě jiných aktivačních funkcí. [8] [7]

Naopak nevýhodou této aktivační funkce je takzvaný *ReLU Problem*, který spočívá ve ztrátě schopnosti neuronu učit se pro případ záporných argumentů funkce. Tato náchylnost je způsobena nulovým gradientem funkce v případě záporných argumentů. Tomuto jevu lze předcházet lehkými úpravami této aktivační funkce. Může být například nadefinován předpis, který připouští malou směrnici polopřímky v části se záporným argumentem, taková aktivační funkce se nazývá *LReLU*. V případě aktivační funkce *PReLU* je směrnice této polopřímky dána parametrem, který je také určován učením sítě. Existuje mnoho dalších úprav funkce *ReLU* s cílem zabránit *ReLU problému*. [8]

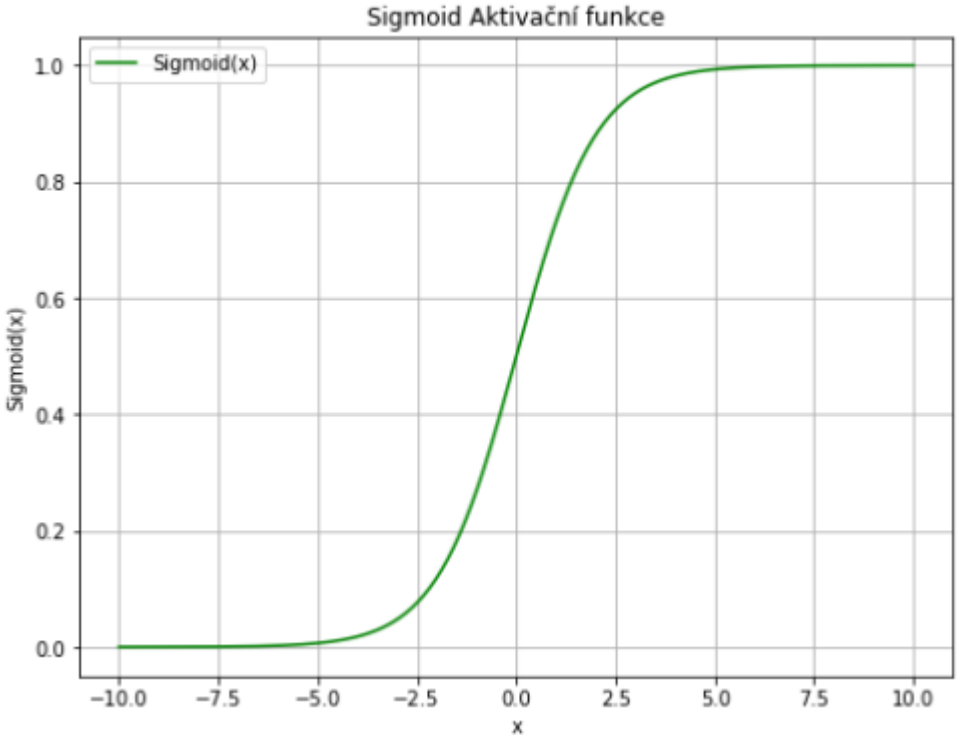
- Sigmoid

Aktivační funkce *sigmoid* je vhodná k použití, pokud je neuron určený k binární klasifikaci, protože výstup je možné interpretovat jako pravděpodobnost. Na rozdíl od předchozí aktivační

funkce je funkce *sigmoid* funkcí hladkou, což je vhodné při využívání metody zpětné propagace (anglicky *backpropagation*). Funkce je definována následujícím předpisem:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{1-2}$$

Obor hodnot je (0;1). Funkce obsahuje exponent a samotný výpočet tedy může být náročnější než například u funkce *ReLU*. [8]



Obrázek 1-4: Průběh aktivační funkce sigmoid(x) vykreslené pomocí knihovny matplotlib

Na obrázku 1-4 je znázorněn graf funkce *sigmoid*. Při pohledu na tento graf je zjevné, že při argumentu, který bude dál od 0, dochází k takzvané saturaci funkce a může docházet k mizení gradientu. Tomuto problému se v praxi předchází vhodným znormalizováním vstupů. [9]

- GeLU

GeLU aktivační funkce kombinuje vlastnosti funkce *ReLU* s vlastnostmi *dropoutu* a *zoneoutu*, což jsou prvky vedoucí k stochastickému vyřazení neuronu. Podstata této funkce spočívá v tom, že vstup do neuronu je násoben pravděpodobností, že náhodné číslo z normálního rozdělení bude menší než hodnota na vstupu do neuronu. Tedy s větším argumentem je větší šance, že se neuron aktivuje, a naopak s menším se zmenšuje i šance – podle distribuční funkce normálního rozdělení. Aktivace neuronu je tedy sice stochastická, ale zároveň závislá na velikosti vstupu [10].

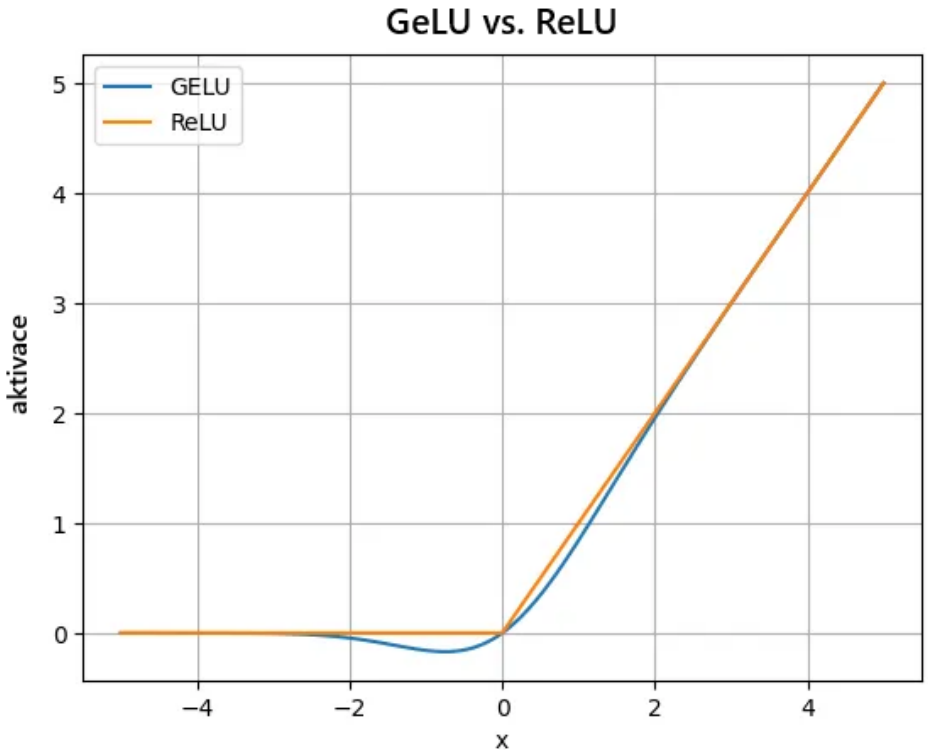
Výhodou této funkce oproti *ReLU* je plynulejší přechod, což přispívá ke kvalitě učení. Funkce je nicméně náročnější na výpočet a složitější na implementaci. Předpis funkce je dán následovně:

$$GeLU(x) = x \cdot \Phi(x) \tag{1-3}$$

kde *Φ(x)* je distribuční funkce normálního rozdělení s předpisem:

(1-4) $$\Phi(x) = \frac{1}{2} \cdot \left[1 + erf\left(\frac{x}{\sqrt{2}}\right)\right]$$

kde *erf* je Gaussova chybová funkce. [10] [8]



Obrázek 1-5: Porovnání grafů funkcí GeLU a ReLU, upraveno dle [11]

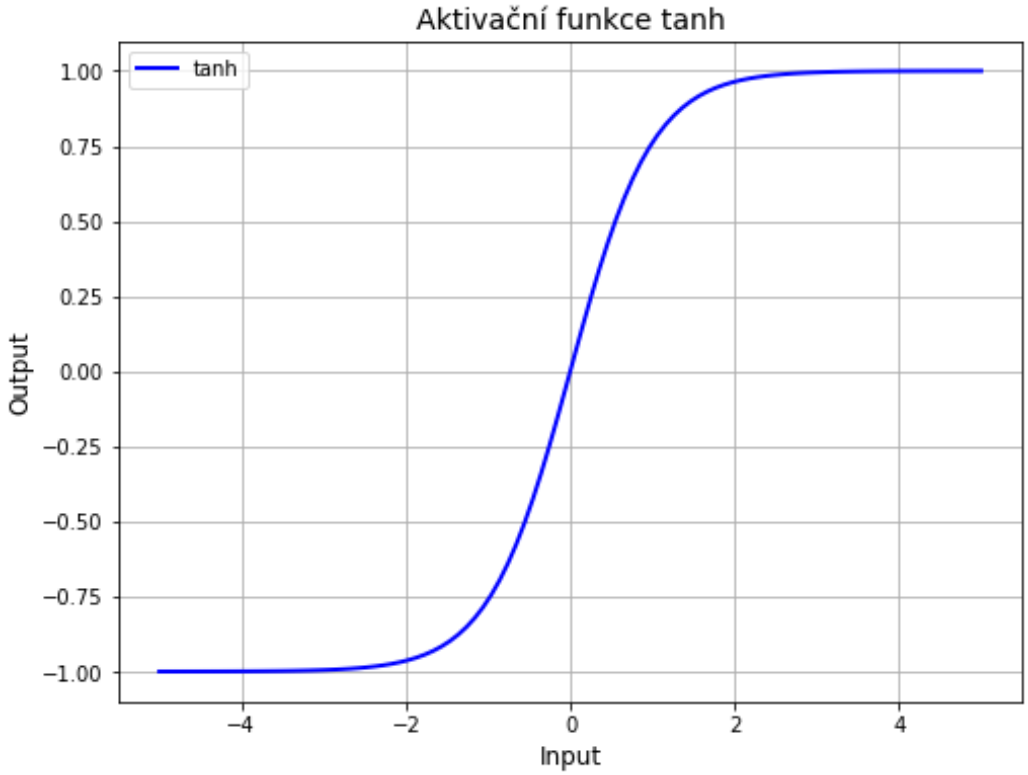
Pro zjednodušení výpočtu se v praxi často využívá zjednodušená aproximovaná funkce [10]:

(1-5) $$GeLU(x) = 0{,}5x\left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}(x + 0{,}044715x^3)\right]\right)$$

- Tanh

Funkce *tanh* je svým tvarem velmi podobná funkci *sigmoid*, se kterou je často srovnávána. Liší se ovšem v důležitých aspektech – její obor hodnot je (-1;1) a je symetrická kolem osy x. Funkce má pro stejné argumenty vyšší gradienty než funkce *sigmoid*, a je tedy odolnější na mizení gradientu a obyčejně konverguje rychleji, čemuž přispívá také symetrie kolem osy x. Předpis funkce je dán předpisem [12]:

(1-6)
$$Tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



Obrázek 1-6: Průběh aktivační funkce tanh(x) vykreslené pomocí knihovny matplotlib

- SoftMax

Funkce *SoftMax* je nejčastěji využívána při vícetřídních klasifikacích. Vstup pro tuto funkci je vektor, stejně tak i výstup funkce. Funkce normalizuje přijaté vstupy tak, aby výstupy bylo možné interpretovat jako pravděpodobnosti. Je tedy splněno, že součet všech prvků výstupního vektoru je roven jedné a obor hodnot každého prvku vektoru je (0;1). Díky těmto vlastnostem bývá často používána v poslední vrstvě neuronové sítě. [13]

Předpis funkce *SoftMax* je dán následovně [13]:

(1-7)
$$SoftMax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{Z} e^{z_j}}$$

Kde z je vektor vstupních hodnot, i a j jsou indexy prvků ve vektoru z a Z je počet prvků ve vektoru z

1.2 Základní algoritmy neuronových sítí

Hlavní princip neuronových sítí spočívá v aproximaci řešení. K aproximaci řešení dochází pomocí optimalizace vah a posunů v jednotlivých neuronech. Princip a účel neuronových sítí tak tedy lze v jistém nadneseném ohledu přirovnat například k metodě nejmenších čtverců, kde se jedná také o aproximaci řešení pomocí optimalizace parametrů a minimalizace ztrátové funkce. Neuronové sítě jsou samozřejmě mnohem komplexnější nástroj využívající komplexnější metody a jsou schopné zachytit složitější vzorce mimo jiné díky nelinearitě.

Smysl učení neuronové sítě tedy spočívá v nalezení správných hodnot vah w_i a posunů b_i. To se děje využíváním algoritmu zpětné propagace (*backpropagation*). V prvním kroku, kdy jsou hodnoty w_i a b_i, dejme tomu, náhodně iniciovány, dojde k výpočtu ztrátové funkce. Ztrátová funkce má obecně řečeno význam kvantifikované odlišnosti predikce a skutečnosti. Ztrátová funkce může být definována v závislosti na problému různě. Mezi nejběžnější ztrátové funkce používané při řešení regrese může patřit *Mean Square Loss Function* a naopak pro problém klasifikace se často využívá *Cross Entropy Loss Function*. *Mean Square Loss Function* je definována následovně:

$$L = \frac{1}{2m}\sum_{i=0}^{m}(h_0(x_0, x_1, \dots, x_n) - y_i)^2 \tag{1-8}$$

Často se objevuje také varianta, kde se před sumou nachází pouze $\frac{1}{m}$.
V rovnici je m počet datových vzorků, y_i je skutečná hodnota výstupu (z datového vzorku), x_i jsou potom vstupy z datového vzorku, a jelikož funkce h_0 je funkcí hypotézy, výraz $h_0(x_0, x_1, \dots, x_n)$ je vyčíslením predikovaného výstupu, n je počet vstupních argumentů do hypotézy. [14]
Cross Entropy Loss Function pro binární klasifikaci je pak definována vztahem:

$$L = -\frac{1}{m}\left[\sum_{i=0}^{m}[y_i log(p_i) + (1 - y_i)\log(1 - p_i)]\right] \tag{1-9}$$

V rovnici je m počet datových vzorků, y_i je skutečná hodnota výstupu (z datového vzorku), p_i jsou potom výstupy z neuronové sítě interpretované jako pravděpodobnost obvykle pomocí funkce *sigmoid*. [15]
Pro optimalizování všech vah a posunů v síti probíhá algoritmus zpětné propagace. Při něm je ztrátová funkce parciálně derivována podle jednotlivých optimalizovaných parametrů a je tedy počítán její gradient. Tyto derivace ve své podstatě vypovídají o tom, jak velký vliv má parametr (podle které se derivuje) na vznik chyby – pokud bude derivace nulová, ve směru konkrétního parametru se nacházíme na minimu, parametr se na chybě podílí minimálně a je zoptimalizovaný. Jelikož jsou tyto derivace obvykle velmi složité, využívá se *takzvaného chain rule* neboli řetězového pravidla, kdy je parciální derivace ztrátové funkce zapsána pomocí vícero jednodušších derivací odpovídající struktuře modelu sítě. Pro nalezení minima ztrátové funkce, a tedy optimálních parametrů je potřeba, aby se gradient, a tedy jednotlivé parciální derivace ztrátové funkce rovnaly nule, respektive se jí blížily.

K tomu je využíván algoritmus gradientního sestupu (anglicky *gradient descent*). Jedná se o princip, kdy se optimalizuje například váha *w* odečtením hodnoty derivace ztrátové funkce podle této váhy *w*, vynásobené learning rate, od původní hodnoty váhy *w* [5]. Tedy:

$$w_{new} = w - \eta \cdot \frac{\partial L(w)}{\partial w} \tag{1-10}$$

Kde η je learning rate, programátorem určená či optimalizovaná hodnota určující, jak velkými kroky se budou měnit optimalizované hodnoty [5]. Hodnota learning rate je důležitým prvkem, který je potřeba zvolit, protože má výrazný vliv na schopnost neuronové sítě učit se. Learning rate je hyperparamter, který v podstatě udává velikost kroku ve směru klesajícího gradientu při určování nových vah. Zvolený learning nesmí být ani příliš velký, ani příliš malý. Příliš velký learning rate může způsobovat divergenci a nemožnost naleznutí minima. Naopak příliš malý learning rate zpomaluje konvergenci a může znamenat uváznutí v lokálním minimu a nenalezení toho globálního [16]. Problematika je znázorněna na obrázku 1-7:

Obrázek 1-7: Znázornění průběhů hledání minima v závislosti na zvoleném learning rate, upraveno dle [16]

Během učení se tímto způsobem spočítají optimální parametry vyskytující se v neuronové síti a síť je pak možné považovat za naučenou. Taková síť pak při dodání neznámých vstupů dokáže vracet výstupy, které jsou blízko reálným hodnotám. [5]

1.3 Klasifikační metriky neuronové sítě

Klasifikační metriky jsou hodnoty, které lze vyčíslit pro konkrétní neuronové sítě, a na základě nich hodnotit kvalitu a relevanci neuronových sítí nebo porovnávat sítě mezi sebou. Pro definici jednotlivých metrik je nutné nejprve nadefinovat 4 pojmy:

- Skutečně pozitivní (*True positivity*) (TP) – počet pozitivních prvků, které algoritmus odhadl správně
- Skutečně negativní (*True negativity*) (TN) – počet negativních prvků, které algoritmus odhadl správně
- Falešně pozitivní (*False positivity*) (FP) – počet negativních prvků, které algoritmus odhadl špatně
- Falešně negativní (*False negativity*) (FN) – počet pozitivních prvků, které algoritmus odhadl špatně

Pomocí různých poměrů těchto skupin je možné určit některé kvality neuronové sítě. [17]
Nelze jednoznačně a obecně určit hranici, kdy lze metriky považovat za dostačující. Uspokojivé hodnoty se určují vždy v kontextu daného problému, počtu dostupných dat k učení a podobně.

1.3.1 Preciznost (*precision*)

Metrika preciznost (*precision*) vypovídá o tom, jak velký podíl z prvků modelem určených jako pozitivní je skutečně pozitivní. Respektive udává poměr skutečně pozitivních prvků ku všem pozitivně určených prvků. Definici je tedy možné zapsat následující rovnicí:

$$Precision = \frac{TP}{TP + FP} \tag{1-11}$$

Preciznost je vhodné maximalizovat v případech, kdy je nutné snižovat falešnou pozitivitu, a tedy je žádoucí, aby bylo označení prvku za pozitivní důvěryhodné a pravděpodobně správné. V případě maximalizace této metriky je tedy možné hovořit o přísném kritikovi, který eventuelně zpochybňuje i to, co nemá. [17]

1.3.2 Úplnost (*recall*)

Atribut úplnost hodnotí schopnost sítě označovat prvky jako pozitivní ve vztahu ke všem pozitivním prvkům, tedy k prvkům skutečně pozitivním a falešně negativním. Vypovídá tedy o tom, jak moc síti unikají další pozitivní prvky. Definice je zapsána následovně:

$$Recall = \frac{TP}{TP + FN} \tag{1-12}$$

Metriku úplnost je vhodné sledovat a maximalizovat v případě, že charakteristika problému vyžaduje záchyt co největšího množství pozitivních případů, a tedy je nutné minimalizovat falešně negativní prvky. Opět je možné hovořit o kritikovi, který je tentokrát laxní a eventuelně nezpochybňuje to, co by měl. [17]

1.3.3 Přesnost (*accuracy*)

Metrika přesnost vypovídá o celkové přesnosti neuronové sítě. Jedná se o poměr celkových správně označených prvků ku všem označeným prvkům. Definice je zapsána následovně:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1-13}$$

Při sledování přesnosti je nutné brát v potaz, že atribut funguje správně pouze v případě, že se jedná o vyváženou datovou sadu, tedy pozitivních prvků není výrazně více, nebo míně než negativních. V opačném případě by mohla být přesnost například vysoká i v případě, že by síť nefungovala dobře. V extrémním případě, kdy by byl v datové sadě o 100 prvcích pouze jeden negativní a síť by všechny prvky označila jako pozitivní, by byla přesnost 99 %, i když síť vůbec nedisponuje schopností negativní prvky rozeznávat. [17]

1.3.4 F1 skóre (*F1 score*)

Přestože metriky preciznost a úplnost jsou v určitém smyslu konkurenční – u jedné hovoříme o laxním kritikovi, u druhé o přísném – může vznikat potřeba dosahovat co nejlepších hodnot u obou těchto atributů. F1 skóre je harmonickým průměrem těchto dvou metrik. Harmonický průměr je případnou nedostatečností jedné z metrik penalizován mnohem víc než aritmetický průměr, proto je vhodný při zkoumání vyváženosti preciznosti a úplnosti. Matematicky je F1 skóre definováno následovně:

$$F1\ Score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{1-14}$$

V různých případech se můžou sledovat také modifikované varianty, které mohou lépe popisovat daný problém – *Micro F1 score* nebo *Sample-weighted F1 score* vhodné pro nevyvážené datové sady, kdy je žádoucí přikládat větší váhu obsáhlejším třídám, *Macro F1 score*, když je naopak třeba brát v potaz každou třídu stejně nezávisle na její velikosti nebo *Fβ score*, když je potřeba přikládat větší váhu preciznosti než úplnosti nebo naopak. Metriky F1 jsou vhodné při porovnávání jednotlivých neuronových sítí mezi sebou, protože je ze všech metrik nejkomplexnější. [18]

1.4 Učení neuronových sítí

Učení neuronových sítí, tedy optimalizace parametrů v sítí, může probíhat třemi základními způsoby – učení s dohledem, učení bez dohledu a posílené učení. Přístupy se liší především typem dat, které modely zpracovávají, a zpětnou vazbou, které modely dostávají.

1.4.1 Učení s dohledem

Při učení s dohledem neuronová síť zpracovává data, která jsou takzvaně označená. Taková datová sada obsahuje u každého prvku také cílovou hodnotu (*target*), například třídu, do které náleží, přesnou číselnou hodnotu a podobně. Učení s dohledem je dobře aplikovatelné na problémy klasifikace a regrese. Zpětnou vazbou pro model je ztrátová funkce, která byla popsána výše v této práci.

Takovéto učení je obvykle přesné a efektivní. Nevýhodou této metody však zůstává nutnost jednak velkého množství dat, které může být časově náročné získat, a jednak nutnost lidského vyhodnocení a označení dat. S tím souvisí také příležitost pro lidskou chybu, neboť v případě, že data budou označena chybně, model se naučí špatný vzorec. [19]

Tento typ učení neuronových sítí bude využit i v této práci, neboť byla naměřena data, která byla měřena záměrně jako data bez vady nebo záměrně jako data s vadou.

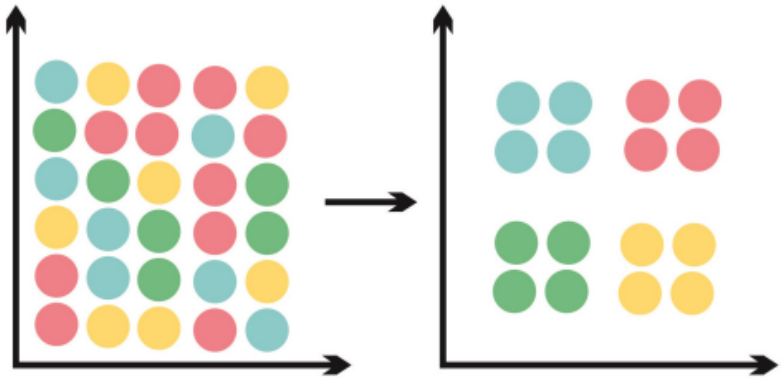
1.4.2 Učení bez dohledu

Rozdíl mezi učením s dohledem a učením bez dohledu je především struktura zpracovávaných dat – při učení bez dohledu nemají data označení. Zpětná vazba probíhá na základě hodnocení interních metrik kvality modelu, které hodnotí, jak se daří plnit cíle. Existuje množství technik, které se při učení bez dohledu využívají:

- Shlukování

Princip učení bez dohledu shlukováním spočívá v rozdělení dat do shluků, kdy je cílem, aby si prvky v jednom shluku byly podobné, a naopak aby byly odlišné od prvků v jiných shlucích. Podobnost se obvykle definuje pomocí vzdálenosti definované metriky. Mezi nejčastěji využívaný algoritmus shlukování se využívá například K-means – algoritmus náhodně volí středy shluků, následně pomocí euklidovské vzdálenosti určité metriky přiřadí prvky k jednotlivým středům. Tyto středy jsou následně přepočítány jako průměry přiřazených bodů. Tento proces se iterativně opakuje.

Existuje několik variant této metody – exkluzivní shlukování, kdy jeden prvek náleží právě do jednoho shluku; hierarchické shlukování, kdy je mezi shluky vytvořena určitá hierarchie; překrývající se shlukování, kdy jeden prvek může náležet i do více shluků. [20]



Obrázek 1-8: Demonstrace shlukování [20]

- Asociace

Při učení asociací se algoritmus snaží nalézt určité vztahy mezi jednotlivými prvky ve zpracovávaných datech. Cílem algoritmu je nalézt určitá pravidla a souvislosti vypovídající o tom, jak mezi sebou různé prvky a kombinace prvků souvisí. Algoritmus se tedy snaží najít vztahy ve smyslu *Když nastane X, pak nastane Y* a podobně. Tyto vztahy jsou pak hodnoceny metrikami, jako je podpora (*support)* či důvěra (*confidence)*. *Support* vyjadřuje v kolika případech z celkového počtu případů nastal případ *Když X, tak Y* (existuje vztah). *Confidence* pak určuje, jak často je to skutečně pravidlem, tedy jak často se vyskytuje vztah *Když X, tak Y* ve všech případech, kdy nastává X. [20]

- Autoenkódery

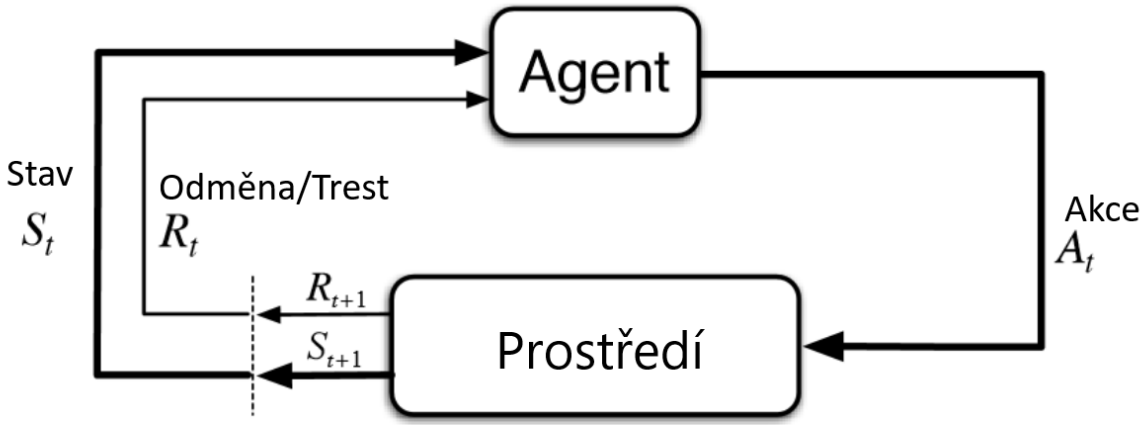
Autoenkódery jsou typ neuronové sítě, která funguje na principu rozkladu a následné zpětné rekonstrukce dat. Autoenkódery se skládají ze dvou hlavních částí – enkóderu a dekóderu. Enkóder rozkládá a transformuje data ze vstupu do nižší dimenze, čímž vytváří takzvanou latentní reprezentaci. Dekóder se potom snaží tuto latentní reprezentaci zpětně rekonstruovat na původní data ze vstupu. Rozdíl mezi původními daty a daty zrekonstruovanými dekóderem z latentní reprezentace popisuje opět ztrátová funkce, často MSE nebo binární Cross-entropy. Vyšší hodnota ztrátové funkce může být způsobena například výskytem anomálie ve zpracovávaných datech. Autoenkodery se tedy hodí i pro binární klasifikaci v případě učení bez dohledu. [20] [21]

Obrázek 1-9: Diagram struktury autoenkóderu [20]

1.4.3 Posílené učení

Posílené učení funguje nadneseně na principu pokus – omyl. Ve schématu takovéto sítě se nachází agent – prvek, který dělá rozhodnutí na základě své politiky (strategie, pravidel) a na základě stavu, ve kterém se nachází. Na základě toho, jestli je výsledkem agentovy akce správný či nesprávný výstup, dostává agent jako zpětnou vazbu odměnu či trest. Cílem algoritmu je maximalizovat výši odměn. Tento druh učení je hojně používaný například

v oblasti autonomních vozidel, robotiky nebo výuce algoritmů k hraní her a k problémům hrám podobným. Kontext a vše, s čím agent může interagovat, tedy provádět zde různé akce, se nazývá environment – prostředí. [22]



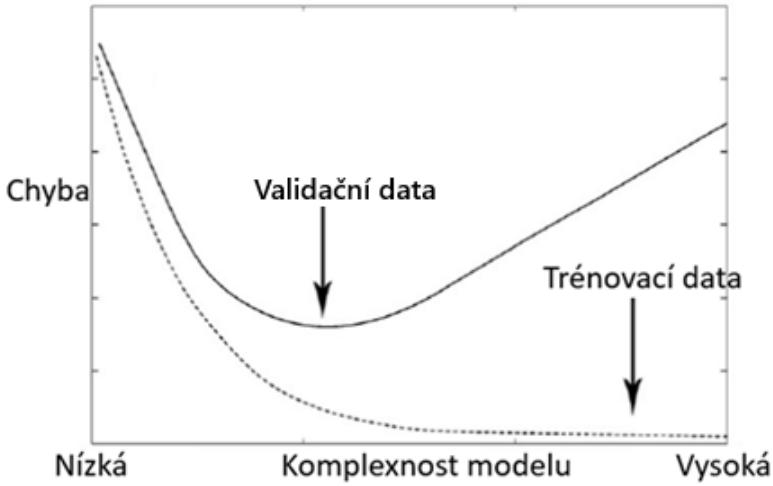
Obrázek 1-10: Schéma posíleného učení [23]

1.5 Typické problémy při učení neuronových sítí

Při trénování neuronové sítě může nastat několik typických problémů, které mají negativní vliv na efektivitu modelu či na jeho celkové fungování.

1.5.1 Přeučení a nedoučení (*overfitting a underfitting*)

Přeučení je problém, který nastává při učení s dohledem v důsledku příliš složitého modelu vzhledem k řešenému problému a množství dat. Přeučení je neschopnost modelu zobecnit daný problém, dojde k perfektnímu vytrénování modelu na tréninková data, kdy jsou ovšem zohledněny i nechtěné jevy včetně šumu a odchylek. Model má následně dobrou výkonnost na trénovacích datech, ale špatnou na datech testovacích a validačních. Příčiny přetrénování modelu neuronové sítě může být jednak přílišná výraznost šumu, kdy dochází právě k jeho zachycení a zanedbání skutečně relevantních vztahů, a jednak při předpokládání příliš složité hypotézy, například při příliš mnoho vstupech, kde mnohé nejsou relevantní. Dalším častou příčinou je nedostatečné množství trénovacích dat. [24]

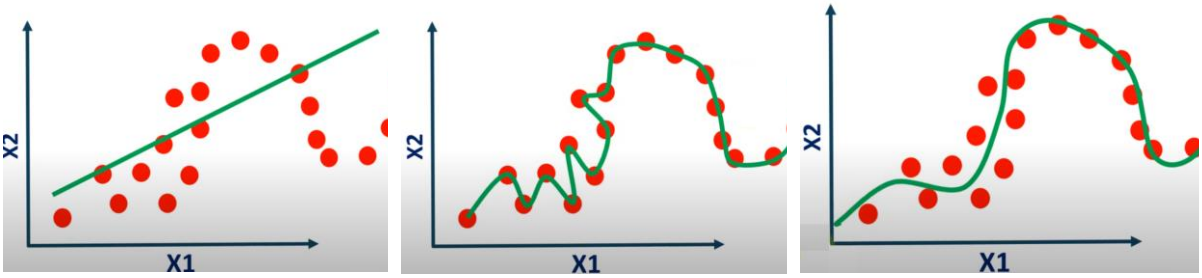


Obrázek 1-11: Znázornění průběhu chyby na trénovacích a validačních datech u přetrénovaného modelu

Na obrázku 1-11 je zřejmé, že model dosahuje nejvyšší výkonosti při určité komplexnosti. Při dalším zvyšování chyba opět narůstá.

Naopak nedoučení je problém, kdy je model až příliš jednoduchý na to, aby dokázal zachytit určité vzory. Může vznikat, pokud je architektura modelu nedostatečná vůči řešenému problému a množství dat. Typickým příkladem vzniku nedoučení může být řešení nelineárního problému lineární regresí.

Cílem dobře vytrénovaného modelu je být složitý tak akorát, aby byly dostatečně obecně zachyceny všechny vzorce, které jsou podstatné, a aby nebyly zachyceny ty vzorce, které jsou nepodstatné. Tohoto je dosaženo přiměřenou složitostí modelu – přiměřeným počtem vrstev a neuronů – také vystavením rozumného množství dat, pracuje se tedy s počtem epoch. Tento proces je iterační a vychází se ze zkušeností, případně se přistupuje k optimalizačním algoritmům. [24] [25]



Obrázek 1-12: Ukázka (z leva) nedotrénovaného modelu, přetrénovaného modelu a modelu optimálního, upraveno [25]

1.5.2 Mizející a explodující gradient (*Vanishing* a *exploding gradient*)

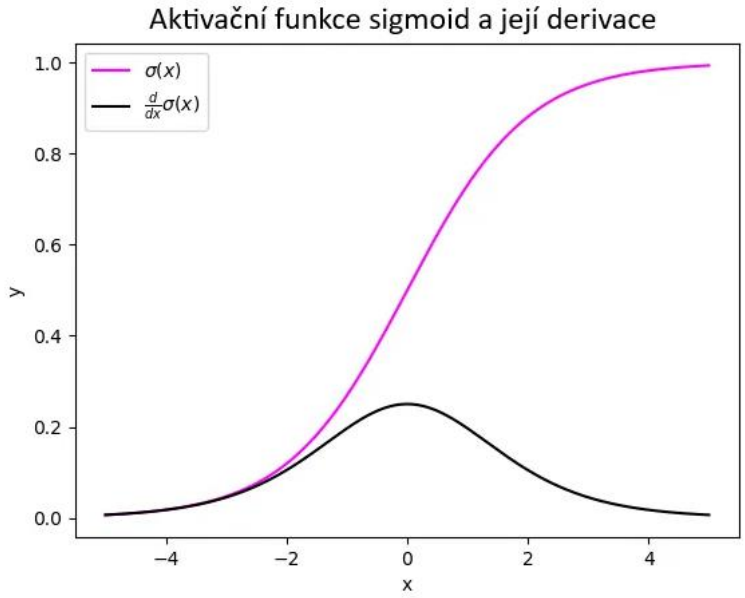
Problém mizejícího gradientu je problém, který může nastat při nevhodném navržení neuronové sítě především potom, pokud obsahuje mnoho skrytých vrstev. Tomuto jevu musí být věnována zvýšená pozornost především u rekurentních sítí. Při zpětném šíření dochází ke zmenšování gradientu tak, že přestane být signifikantní a učení neuronové sítě (hledání minima ztrátové funkce) se drasticky zpomalí nebo zcela ustane. Nejčastěji je tento jev spojovaný s aktivačními funkcemi *sigmoid* a *tanh*. [26]

Pro pozorování toho, jak dochází k vymizení gradientu, je nutné rozepsat gradient ztrátové funkce podle váhy v poslední skryté vrstvě pomocí řetězového pravidla:

$$\frac{\partial L}{\partial w^{(l-1)}} = \frac{\partial L}{\partial \sigma(z^{(l)})} \frac{\partial \sigma(z^{(l)})}{\partial z^{(l)}} \frac{\partial z^{(l)}}{\partial \sigma(z^{(l-1)})} \frac{\partial \sigma(z^{(l-1)})}{\partial z^{(l-1)}} \frac{\partial z^{(l-1)}}{\partial w^{(l-1)}} \tag{1-15}$$

Kde L je ztrátová funkce, σ je aktivační funkce a z je vstup, l označuje index vrstvy. V důsledku použití řetězového pravidla dochází k tomu, že výsledný gradient může být součinem mnoha malých čísel, a tedy může vymizet. Jak již bylo řečeno, do značné míry riziko souvisí s použitou aktivační funkcí, a především pak s jejími derivacemi.

Tento jev nastává v případě, že sklon aktivační funkce, a tedy i její derivace je velice malá. [9]



Obrázek 1-13: Znazornění derivace funkce sigmoid, upraveno [9]

Na obrázku 1-13 je znatelné, že derivace aktivační funkce *sigmoid* dosahuje maximální hodnoty 0,25. V případě, že jsou argumenty vzdálené od nuly, je derivace ještě mnohem menší. Pokud by v průběhu zpětné propagace byly argumenty právě takové, hrozil by problém mizejícího gradientu. Problém může být řešen například normalizací dat na vstupu každé vrstvy navržením vhodnější aktivační funkce nebo například tvrdým omezením velikosti gradientu. [26]

Problém nazývaný exploze gradientu je problém, dá se říct opačný, tedy gradient v průběhu zpětné propagace rychle narůstá. To může způsobit potíže se stabilitou sítě a také numerické potíže. Na rozdíl od problému s mizejícím gradientem, explodující gradient souvisí spíš s počáteční inicializací vah než s volbou aktivačních funkcí. Podobně jako mizející gradient je možné tento problém řešit normalizací dat na vstupu každé vrstvy či tvrdým ořízntutím gradientu, který se dostane nad definovanou hranici. [26]

2 Druhy neuronových sítí

Existuje několik typů neuronových sítí, kdy se každá hodí pro jiný typ problému. Nejdůležitější z nich budou pro kontext práce stručně představeny v této kapitole.

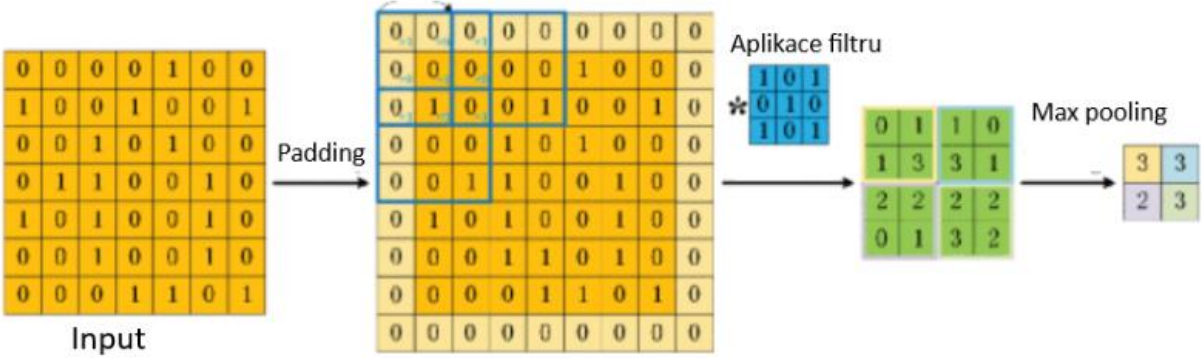
2.1 Plně propojená neuronová síť

Plně propojená neuronová síť je základním typem neuronové sítě, ze které následně vychází další, pokročilejší architektury. Jak napovídá název, jedná se o síť, kde je každý neuron spojen se všemi neurony v předchozí vrstvě a se všemi neurony ve vrstvě následující. Algoritmus a fungování tohoto druhu sítě byl popsán v předchozí kapitole. Jedná se o univerzální variantu, která je vhodná pro řešení širokého spektra problémů, především pro řešení klasifikačních problémů či problémů regrese. Tento druh sítě se naopak nehodí pro zpracování časových řad či obrázků. Ačkoliv tyto problémy řešit dokáže, algoritmus není příliš efektivní, protože kvůli propojenosti všech neuronů je řešení časově náročné. [5]

2.2 Konvoluční neuronová síť

Konvoluční neuronová síť má využití především v počítačovém vidění a pro problémy vyvíjecí se v čase. Konvoluční síť má tři různé druhy skrytých vrstev – konvoluční vrstvy, poolingové vrstvy, a nakonec plně propojené vrstvy. Neurony konvoluční vrstvy mají na rozdíl od plně propojených sítí spojení pouze s některými dalšími neurony, což dokáže urychlit konvergenci. Stejný efekt má i možné sdílení vah skupinou neuronů, což snižuje množství optimalizovaných parametrů. [27]

V konvoluční vrstvě dochází k aplikaci takzvaného filtru neboli jádra, kernelu, na vstupní matici. Ještě před konvoluční vrstvou může být na vstupní matici aplikován *padding*, který pomocí nulových prvků dokáže v případě nutnosti upravit velikost tohoto vstupu. Představme si, že vstup je obrázek. Pro zjednodušení předpokládejme, že je černobílý, a můžeme se tedy omezit na 2D prostor (v případě, že by byl barevný, třetí rozměr by měl tři prvky a obsahoval by RGB hodnoty, takto můžeme říct 1 – černá, 0 – bílá). Potom obrázek bude reprezentován maticí obsahující 1 a 0 o libovolném rozměru. Na tuto matici bude aplikován filtr, tedy typicky násobně menší matice, která obsahuje váhy, kterými budou vstupy násobeny. Filtr je na vstup postupně aplikován. Často tak, aby se vždy v dalším kroku částečně překrýval s krokem minulým. Viz obrázek 2-1. Velikost kroku se nazývá *stride* a čím je větší, tím je nižší hustota konvoluce. Vynásobením vstupů z oblasti, na kterou je filtr aplikován, a vah filtru, vzniká následně menší matice, která se nazývá mapa rysů. Díky aplikaci filtru na vstupní matici tedy získáme matici méně rozměrnou, která zachovává podstatné informace, tedy rysy. [27] [5]



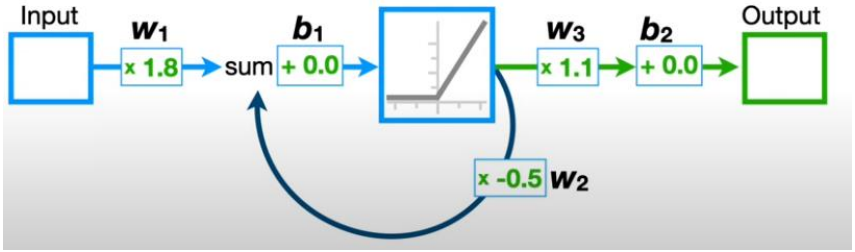
Obrázek 2-1: Znázornění funkce konvuluční a poolingové vrstvy, upraveno [27]

S těmito mapami rysů je dále pracováno v poolingové vrstvě. Mapa rysů může obsahovat mnoho, třeba i redundantních informací, které by mohly způsobovat například přeučení. Tomu se předchází aplikací poolingu. Tedy mapa rysů je redukována za použití dalšího filtru –

nejčastěji se využívá filtru *max pool* nebo *average pool*. Tento filtr zredukuje oblast mapy rysů, na kterou je aplikován, na prvek v další matici. U max poolingu je zredukován na číslo s hodnotou největšího prvku v redukované oblasti, u average poolingu jsou všechny prvky zprůměrovány. Tyto zredukované matice následně vstupují do plně propojené vrstvy. [27]

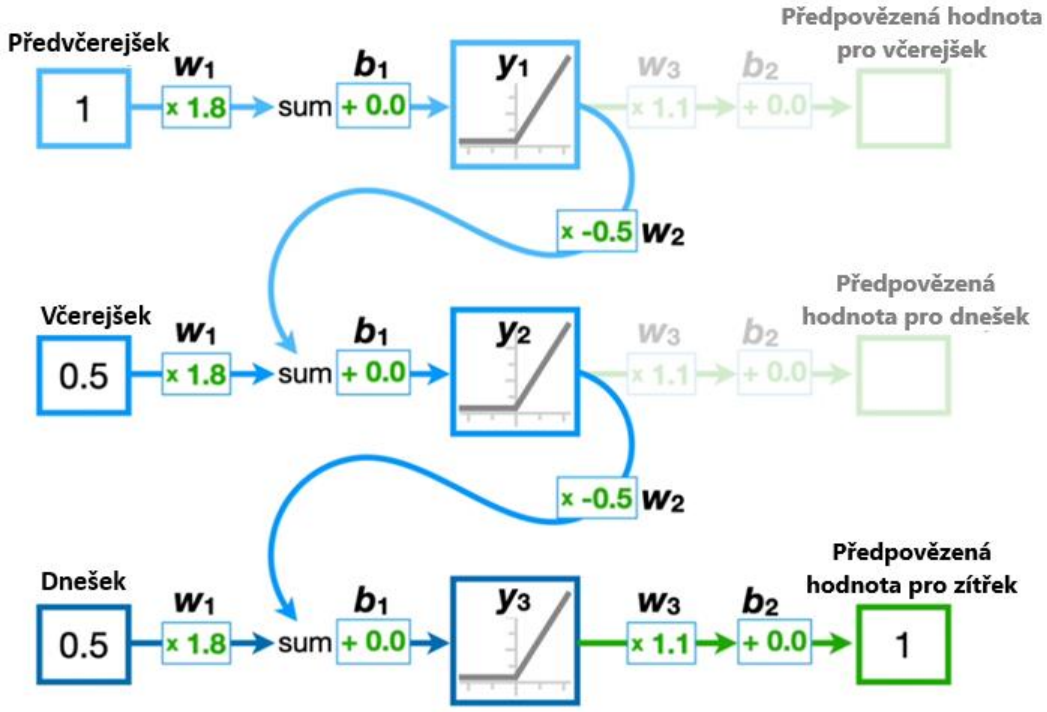
2.3 Rekurentní neuronová síť

Tento typ neuronových sítí je užitečný především ve zpracování sekvenčních dat, jakými můžou být například data v časovém kontextu nebo přirozený jazyk, kdy záleží na pořadí slov ve větě. Rekurentní neuronové sítě zachovávají informace o výstupech, které vznikly s předchozími vstupy. Díky tomu dokáže rekurentní neuronová síť v aktuálním časovém kroku zohlednit stav z předchozích časových kroků, takzvaný skrytý stav [28]. Tato schopnost je realizována pomocí smyčky, která je pro lepší představu znázorněna na obrázku 2-2.



Obrázek 2-2: Znázornění rekurentní sítě [5]

Jak bylo řečeno, rekurentní sítě mají využití mimo jiné také u dat v časovém kontextu, například při předpovědi počasí, kdy víme, jaké počasí bylo včera a předevčírem, víme, jaké počasí je dnes, a snažíme se předpovědět počasí, jaké bude zítra. V tomto případě smyčka proběhne dvakrát. Pro lepší názornost je neuronová síť rozkreslená pro chod s každým vstupem zvlášť:



Obrázek 2-3: Rozkreslení jednotlivých chodů rekurentní neuronové sítě, upraveno dle [5]

Z předvčerejších dat je možné předpovědět včerejší počasí, tento výstup však není zajímavý, neboť včerejší počasí je již známo. Výstup prvního průchodu se tedy vrací jako zpětná vazba a je brán v potaz při chodu sítě se skutečnými daty o včerejším počasí. Stejně tak tento výstup vstupuje do chodu sítě se skutečnými dnešními daty, který již dává předpověď zítřejšího počasí. Protože síť si sama poskytuje zpětnou vazbu v průběhu časových kroků, dokáže zachytit vzory, které se v čase vyskytují. [28] [5]

Při jednotlivých průchodech neuronovou sítí se zachovávají všechny váhy a posuny, které se v síti vyskytují. To je rizikové vzhledem ke vzniku mizejícího a explodujícího gradientu, neboť při zanořování do časových kroků se stále násobí stejnou váhou. Pokud váha bude příliš malá nebo naopak příliš velká, dojde k jednomu ze jmenovaných problémů. [5]

2.3.1 LSTM neuronová síť

Problém mizejícího gradientu do jisté míry řeší použití LSTM neuronových sítí, což je zkratka z anglického *Long short-term memory*. Tyto sítě kromě skrytého stavu obsahují také stav buňky, který slouží k dlouhodobému uchování informace. Které informace do tohoto stavu vstoupí, a tedy budou uchovány, které budou použity pro výpočet skrytého stavu a které budou zapomenuty, rozhoduje systém bran. Tedy vstupní, výstupní a zapomínající brána. [28]

2.3.2 GRU

GRU sítě pracují na podobném principu jako LSTM, liší se v absenci stavu buňky, jeho funkci zastává přímo skrytý stav, který je regulován pouze dvěma bránami – resetovací a aktualizační. Díky jednodušší architektuře se GRU oproti LSTM rychleji učí a mají tak význam například při řešení problémů v reálném čase. [28]

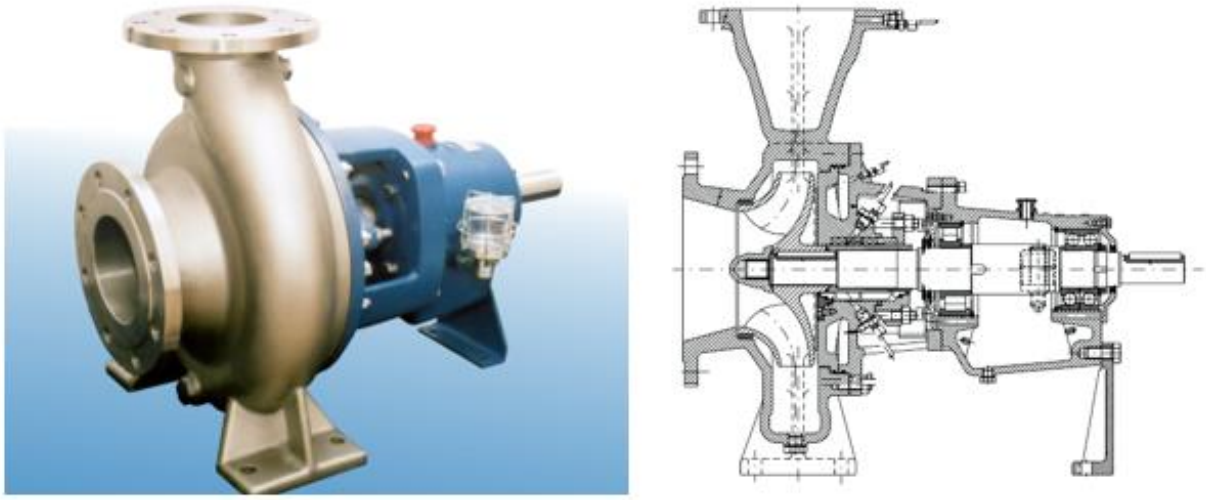
Ačkoliv nejde rekurentní neuronové sítě označit za zastaralé, v dnešní době jsou populárnější pokročilejší architektury, především pak transformery [28].

3 Popis problému a měření

Tato kapitola stručně popisuje měření dat. Struktura a data samotná jsou blíže popsány v následující kapitole. Primárním cílem této práce je snaha využít neuronovou síť pro detekci poruchy na vodním stroji. Data, která posloužila k učení neuronové sítě, byla naměřena na zkušebním hydraulickém okruhu v Těžké hydraulické laboratoři na odboru Fluidního inženýrství na VUT v Brně. Samotné měření nebylo součástí diplomové práce a tato kapitola vychází především ze zdroje [29].

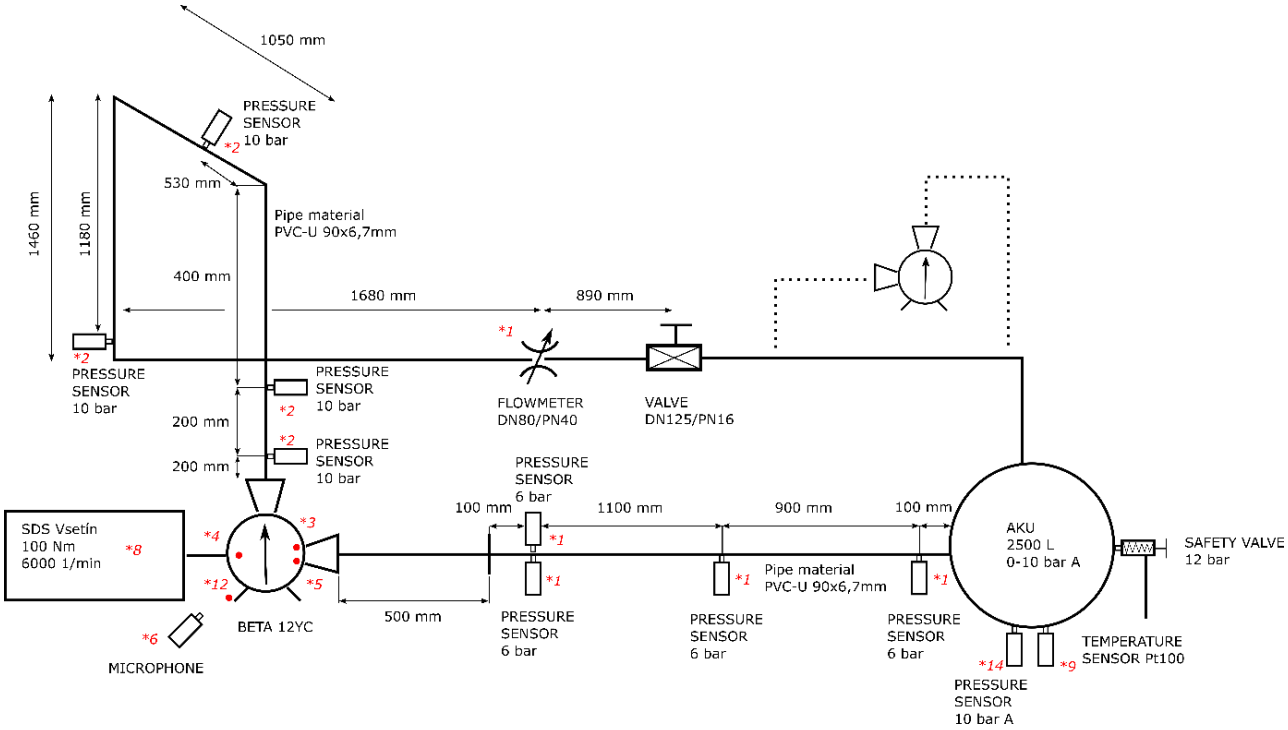
3.1 Popis čerpadla a tratě

Hlavním prvkem měřicího okruhu bylo radiální horizontální čerpadlo s označením ISH 80-50-NHJ-200 (BETA 12YC) a s charakteristickými specifickými otáčkami 78 min^{-1}. Jedná se o jednostupňové čerpadlo s axiálním vstupem a radiálním výstupem, které je určeno pro čerpání kapalin čistých, případně lehce mechanicky znečištěných, s hustotou v rozmezí 600 $kg \cdot m^{-3}$ až 1 900 $kg \cdot m^{-3}$, kinematickou viskozitou do 75 $mm^2 \cdot s^{-1}$ a teplotou od -40 °C do 180 °C. Maximální provozní tlak čerpadla je 25 barů. Pružnou hřídelovou spojkou je na čerpadlo napojen dynamometr. Oběžné kolo čerpadla má celkem 7 lopatek. Oběžné kolo, víko čerpadla, příruba ucpávky, spirála, těsnicí kruh a hřídel jsou vyrobeny z materiálu 1.4308. Lucerna a ložiskové těleso jsou z šedé litiny EN-GJL-200. Toto čerpadlo bylo při experimentu provozováno v čerpadlovém i turbínovém režimu. Kromě tohoto čerpadla bylo v trati zapojeno také podávací čerpadlo. [29]



Obrázek 3-1: Použité čerpadlo a jeho zobrazení v řezu [29]

Čerpadlo bylo zapojeno v měřicím okruhu podle následujícího schématu, červená čísla ve schématu odpovídají svým umístěním pořadovým číslům senzorů v Tabulce 1.



Obrázek 3-2: Schéma zapojení čerpadla v měřícím okruhu [29]

Na sání i výtlaku čerpadla bylo použito plastové potrubí PVC-U s rozměry 90 × 6,7 mm. Přechodový kus, připojený přímo na hrdlo spirály, byl navržen s úhlem rozšíření 8 ° tak, aby se minimalizovala produkovaná akustická emise, která by narušovala měření. [29]

3.2 Použité senzory

Naměřená data obsahují dva typy naměřených dat – data statická a data dynamická. Data statická nebyla z hlediska neuronových sítí zajímavá. Tyto statické kanály zaznamenávaly čas, tlaky v různých částí měřicí tratě, průtok, tlak v kotli, teplotu v kotli, krouticí moment čerpadla a jeho otáčky. Tato data jsou využita při řešení dílčích problémů, ale většinou nevystupují jako vstupy do neuronových sítí. Do neuronových sítí vstupují povětšinou výhradně dynamická data. Ze statických dat se do neuronové sítě vstupovalo pouze v jednom případě při práci s MEL-spektrogramem, a to pouze s otáčkami a jmenovitým průtokem, tato varianta je popsána v kapitole 4.5.5.

Dynamická skupina dat obsahuje záznam zrychlení spirály, tedy vibrace spirály, záznam tlaku ve spirále a na sání čerpadla, záznam akustické emise a záznam hluku z mikrofonu.

Většina veličin je obecně dobře známa, proto bude popsána jen akustická emise. Akustická emise je jev, který je následkem emitace elastických vln v materiálu. Tyto elastické vlny vznikají vnitřními pochody uvnitř materiálu, při kterých dochází k náhlému uvolnění energie, například vznik mikrotrhlin, dislokační skluz v materiálu nebo plastická deformace. Akustická emise tedy může být projevem nějaké formy mechanického namáhání [30]. Akustická emise se může projevovat na širokém frekvenční spektru od 30 kHz do 2 MHz. Na nižších frekvencích probíhá rušení běžným hlukem, což je ovšem zase zachyceno mikrofonem [31].

Data byla měřena se vzorkovací frekvencí 200 kHz. Informace o použitých senzorech jsou uvedeny v následujících tabulkách [29]:

Tabulka 1: Seznam použitých senzorů a informace o nich [29]

1	**Snímač tlaku (na sání čerpadla, savce turbíny) 4 ks**		
Výrobce:	BD Sensors	Měřicí rozsah:	0 – 6 bar A
Typ:	DMP 331	Výstup:	0-20 mA
Přesnost:	±0,25 % z rozsahu		

2	**Snímač tlaku (na výtlaku čerpadla, vstupu do turbíny) 4ks**		
Výrobce:	BD Sensors	Měřicí rozsah:	0 – 10 bar A
Typ:	DMP 331	Výstup:	0-20 mA
Přesnost:	±0,25 % z rozsahu		

3	**Piezoelektrický snímač tlaku (sání čerpadla)**		
Výrobce:	PCB	Měřicí rozsah:	344kPa
Typ:	113B28	Výstup:	14,5 mV / kPa
Přesnost:	±1,0 % z rozsahu		

4	**Piezoelektrický snímač tlaku (spirála)**		
Výrobce:	KISTLER	Měřicí rozsah:	0 - 14 bar
Typ:	211B4	Výstup:	363 mV / bar
Přesnost:	±1,0 % z rozsahu		

5	**Akcelerometr (na spirále)**		
Výrobce:	PCB	Měřicí rozsah:	± 500 g
Typ:	352A60	Frekvenční rozsah:	0,5 Hz – 60 kHz
		Výstup:	10 mV / g

6	**Mikrofon**		
Výrobce:	G.R.A.S.	Měřicí rozsah:	0 – 100 Pa
Typ:	40PH	Frekvenční rozsah:	10 Hz – 20 kHz
		Výstup:	50 mV / Pa

7	**Elektromagnetický průtokoměr**		
Výrobce:	Krohne	Měřicí rozsah:	200 m^3/h
Typ:	Altoflux IFS 4000	Výstup:	0-20 mA
Přesnost:	±0,2 % z měřené hodnoty		

8	**Dynamometr MEZ-Servis**		
Výrobce:	Mez Vsetín	Měřicí rozsah:	100 Nm / 6000 /min
Typ:	SDK50	Výstup:	0-20 mA / 0-20 mA
Přesnost:	±0,2 % z měřené hodnoty		

9	**Snímač teploty**		
Výrobce:	RAWET	Měřicí rozsah:	0 – 50 °C
Typ:	Pt100	Výstup:	4 – 20 mA
Přesnost:	±0,3 % z rozsahu		

11	**Laboratorní napájecí zdroj**		
Výrobce:	Statron		
Typ:	2224.1	Výstup:	0 – 24 V / 6 A

12	**Snímač akustické emise (spirála čerpadla)**		
Výrobce:	DAKEL		
Typ:	MDK-13AS	Zesílení předzesilovače:	0; 32 dB

13	**Aparatura pro zesílení a záznam signálu AE**		
Výrobce:	DAKEL	Frekvenční rozsah:	20 kHz – 15 MHz
Typ:	IPL-3M	Rozlišení převodníku:	18 bitů
Software:	SW IPL	Rozsah zesílení:	0 – 90 dB

14	**Snímač tlaku (sací kotel)**		
Výrobce:	BD Sensors	Měřicí rozsah:	0 – 10 bar A
Typ:	DMP 331	Výstup:	0 – 20 mA
Přesnost:	±0,25 % z rozsahu		

3.3 Provedení měření

Měření probíhalo co nejvíce automatizovaně. Čerpadlový režim byl proměřen následovně. Nejprve byly nastaveny jmenovité otáčky 1500 ot./min. Na těchto jmenovitých otáčkách byl pomocí škrticího ventilu nastavený jmenovitý průtok. Tento průtok je vždy uvedený v názvu souboru. Jmenovitý průtok byl nastavován od 1 l/s do 21 l/s s krokem 3 l/s vyjma přechodu mezi 19 l/s a 21 l/s. Po nastavení všech ventilů, tedy celé tratě, proběhlo proměření všech otáček. Otáčky byly měřeny od zhruba 300 ot./min. do 2300 ot./min. s krokem přibližně 100 ot./min. Otáčky byly měněny pomocí frekvenčního měniče. Toto měření bylo zopakováno pro všechny tři tlaky v kotli 0,6 bar, 1 bar a 3 bar.

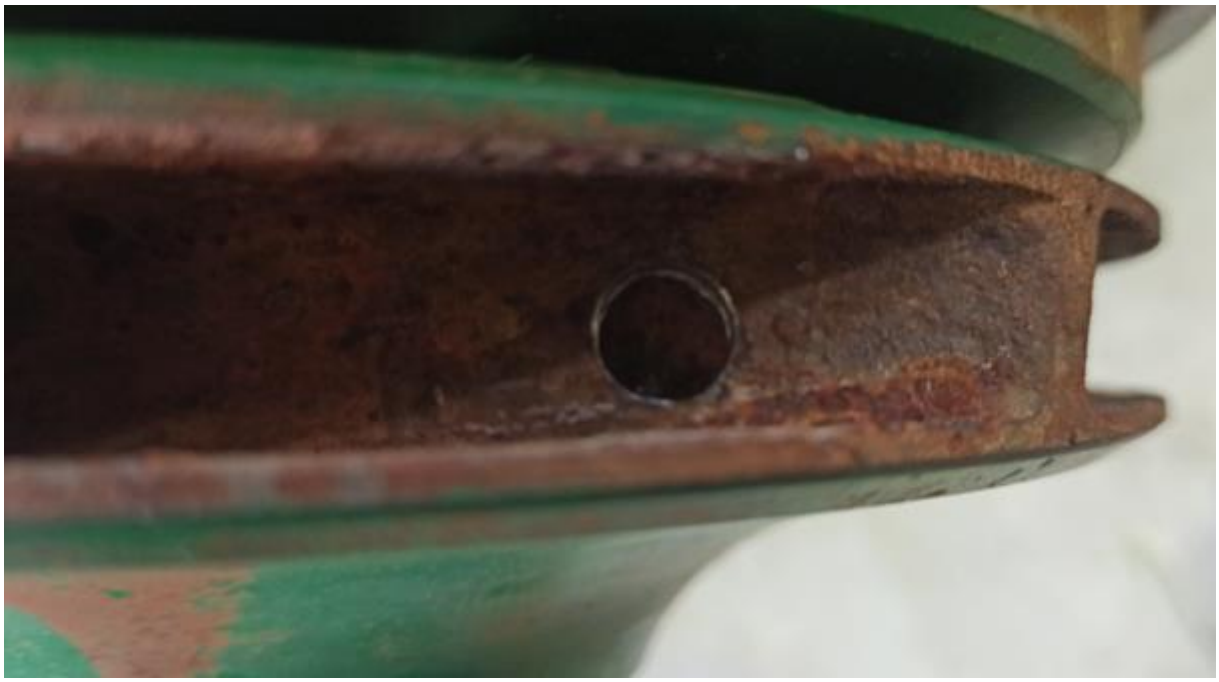
Stejně jako v čerpadlovém režimu i měření turbínového režimu probíhalo za tří stejných tlaků v kotli. V tomto případě byl na turbíně držen konstantní spád 37 m. To se provádělo pomocí podávacího čerpadla. Při tomto spádu byly proměřeny všechny otáčky od 200 ot./min. do 2800 ot./min. Opět s krokem 100 ot./min.

Popsaný proces se provedl s oběžným kolem bez poruchy a následně s uměle vytvořenou poruchou.

3.4 Vytvoření umělé vady na čerpadle

Aby mohla být vytvořena neuronová síť pomocí učení s dohledem, která binární klasifikací rozhodne, zda se jedná o chod stroje s vadou či bez vady, musela být naměřena jednak data, kdy chod stroje běžel bez vady a následně také s vadou. Tato data byla následně neuronové síti předkládána již jako označená.

Vada na lopatce oběžného kola byla vytvořena provrtáním jedné z lopatek. Díra byla vyvrtána 26 mm od výstupní hrany lopatky, v jejím středu, a to s průměrem 8 mm. Původní záměr byl vytvořit kmitáním v lopatce trhlinu. To se ovšem nedařilo, a proto bylo přistoupeno k tomuto způsobu. Díra byla záměrně vyvrtána, vzhledem k šířce lopatky, velká, aby se vada v naměřených datech projevila co nejvíce. Takové provrtání simuluje vadu vzniklou úbytkem materiálu, například oblast poškozenou kavitací. Je pravděpodobné, že taková vada se projeví několika způsoby – zřejmě dojde ke snížení měrné energie čerpadla, dojde ke změně tlakového pole v kanálech u provrtané lopatky, pravděpodobně vzhledem ke vzniklým ostrým hranám dojde i k odtržení proudění a vzniku jevu stall či vzniku Kármánovy vírové stezky. Stejně tak může vada přispět ke vzniku kavitace. Do určité míry může také vyvrtáním vzniknout nevývažek. Pokud je úvaha správná, všechny tyto projevy se propíšou do měřených signálů, ve kterých bude možné je zpětně zachytit. [29]



Obrázek 3-3: Uměle vytvořená vada na čerpadle [29]

4 Řešení neuronové sítě

Tvorba modelů neuronových sítí probíhala především na základě poznatků, diskusí a doporučení vedoucího práce. Při navrhování konkrétních architektur modelů se nejdřív pracovalo s manuálním testováním různých variant, kdy se využila nejlepší z nich. V dalších fázích práce se už přistupovalo i k sofistikovanějším metodám optimalizace.

K samotnému řešení byl využíván programovací jazyk Python, který byl používaný v prostředí Visual Studio Code. Pro snazší práci bylo používáno GUI Anaconda Nvaigator, které usnadňuje instalaci knihoven a tvorbu virtuálních prostředí. Ve valné většině případů byl využíván Python ve verzi 3.11.10. V těchto případech byl používán open source framework Tensorflow od společnosti Google, a to ve verzi 2.19.0 včetně API Keras, který je oficiální součástí. Hojně byla využívána také knihovna Numpy verze 1.26.4. Další využité knihovny mají spíše sekundární význam a jsou uvedeny vždy na začátku kódů v přílohách. V případě implementace optimalizace knihovnou Optuna a zapisování do prostředí Tensorboard byla využita verze Pythonu 3.9.0, a to z důvodu správné synchronizace právě s ostatními knihovnami při zapisování do Tensorboardu. Tensorflow byl v tomto případě použit ve verzi 2.10.1, Optuna ve verzi 4.3.0 a Tensorboard ve verzi 2.10.1.

Primární strategií bylo vstupování do modelů všemi pěti dynamickými snímači najednou. V průběhu výzkumu se ovšem často přistoupilo i k použití jednotlivých snímačů separátně.

Když byla data dělena do trénovací, testovací a validační sady, bylo to obvykle v doporučovaném poměru 7:2:1 [32]. Například při optimalizaci modelu s MEL-spektrogramy v kapitole 4.5.7. byl poměr ovšem důvodně měněn. Takové změny jsou vždy popsány u konkrétního případu.

Protože se v práci pracuje s klasifikačním problémem a vyváženými datasety (tj. záznamů s vadou je přibližně stejně jako bez vady), je vhodnou metrikou ke sledování přesnost – accuracy [33]. Ve všech případech je použita ztrátová funkce binary_crossentropy, která je obecně populární pro klasifikační modely [15]. Jako optimizer byl volen široce využívaný adam.

Při tvorbě kódů dopomáhal jazykový model ChatGPT od společnosti OpenAI ve verzi GPT-4 a GPT-3.5. ChatGPT je citován přímo v příslušných kódech stejně jako další případné zdroje. Všechny relevantní kódy jsou přiloženy v příloze 7. Kódy jsou rozlišeny do složek podle metody, ke které přísluší. V práci jsou popisována pro každou metodu různé nastavení (různé frekvence, různé použité kanály atd.), v přiloženém kódu je vždy nějaké konkrétní nastavení, pokud by měl být kód spuštěn s určitým požadovaným nastavením, je nutné jej nastavit podle textu práce a podle komentářů v kódu.

Bylo testováno několik různých architektur a přístupů k preprocessingu dat. Původní záměr byl učit plně propojenou neuronovou síť surovými daty bez jakéhokoliv předchozího zpracování. Téměř nezávisle na množství zpracovaných dat a nastavení samotné plně propojené neuronové sítě se dosahovalo přesnosti okolo 54 %. Tato přesnost byla zavádějící, protože, ačkoliv pouze lehce, naznačovala, že neuronová síť rozpoznává určité vzorce – pokud by k žádnému nálezu vzorců nedošlo, přesnost by byla statisticky přesně 50 %. Po bližší analýze se však ukázalo, že takovýto přístup je nesmyslný. Hlavním důvodem, proč nejde použít prostou plně propojenou neuronovou síť na surová data je charakteristika vady a tím i její projev. Tím, že se síť téměř nebyla schopna učit, se ukázalo, že vada se v datech ze senzorů zřejmě neprojevuje kontinuálně, ale pouze někdy. Vzhledem k tomu, že vada byla vytvořena provrtáním jedné lopatky, se lze domnívat, že vada se v datech projevuje při určité poloze provrtané lopatky, a tedy s periodou odpovídající jedné otáčce oběžného kola. V prvotním pokusu se v datové sadě tedy nacházela data, která byla sice označena jako data s vadou, ale žádná vada v nich nebyla, respektive se neprojevovala, protože poškozená lopatka v moment

měření zřejmě nebyla v konkrétní, určité poloze. Neuronová síť tedy byla v podstatě učena chybně označenými daty, což vedlo k neschopnosti dále se učit.

Další možností je, že se vada projevuje spíš relativními vztahy mezi po sobě jdoucími datovými záznamy. Například, pro demonstraci myšlenky, pokud by pozorovaná veličina při chodu s vadou oscilovala z vysokých hodnot do nízkých, pozorováním jednotlivých hodnot bychom nutně nemuseli rozeznat vadu – pokud by stejná veličina nabývala stejných hodnot, ale například se zcela jinou frekvencí oscilace či jiným rozdílem, také při chodu bez vady. Je tedy jistě nutné se na data dívat jako na časové řady. Z této úvahy vyplynuly dva postupy, které problém řeší:

- Použití konvoluční neuronové sítě
- Použití statistických charakteristik v rámci preprocessingu dat a následné použití plně propojené sítě

Oba tyto přístupy k datům přistupují s ohledem na jejich charakteristiku. Následující metody zpracovávají data různými způsoby jako časové řady, nikoliv jako nezávislé hodnoty. Je pracováno téměř výhradně s dynamickými senzory: akustické emise, mikrofon, snímač zrychlení a snímač tlaku na sání a ve spirále.

4.1 Metody rozřazení dat do jednotlivých sad

Použitá data se při tvorbě modelu třídí do trénovací, validační a testovací sady. Trénovací sada dat slouží k samotnému tréninku modelu, tedy k optimalizaci vah a posunutí, validační sada dat slouží k průběžnému hodnocení výkonnosti modelu během tréninku a na testovacích datech je plně vytrénovaný model testovaný pro zjištění jeho konečné výkonnosti a hodnot sledovaných metrik. Bylo otázkou, jakým způsobem data třídit. Byly navrženy dva přístupy – do tří sad dělit přímo načítané TDMS soubory nebo nad soubory provést preprocessing (rozdělení do konvolučních oken či výpočet statistických charakteristik) a rozdělit až tyto produkty. Obě varianty mají své výhody a nevýhody.

Roztříděná data po preprocessingu mají nespornou výhodu v tom, že lépe popisují všechny body měření, neboť pochází z mnoha souborů a reprezentují tak statisticky, dejme tomu, celé spektrum zpracovávaných souborů. Oproti tomu při dělení přímo souborů do sad, dochází k výběru několika souborů, které jsou testovány, respektive jsou využity k validaci. Lze tedy říct, že první varianta poskytuje data v jednotlivých sadách různorodější, a lépe tak reprezentují celkový vzorek.

Na druhou stranu při dělení zpracovaných dat jsou si jednotlivé sady vzájemně podobnější, neboť data v nich mohou pocházet ze stejných souborů. Výraznější je tato nevýhoda u zpracování statistikou, neboť u této varianty se pracuje s překryvem jednotlivých oken, ze kterých jsou vyčísleny statistické charakteristiky. Oproti tomu data z dělení souborů pochází vždy z jiného TDMS souboru a odlišnost jednotlivých sad je významnější, a tedy i směrodatnější.

U konvolučních modelů je rozdíl ve výsledné přesnosti na testovací sadě odlišný velmi málo a z hlediska kontextu udává stejnou použitelnost. U turbínových dat je rozdíl zanedbatelný a u čerpadlových dat je v jednotkách procent. Oproti tomu u statistických metod může rozdíl sahat až k devíti procentům. To odpovídá vyřčenému předpokladu.

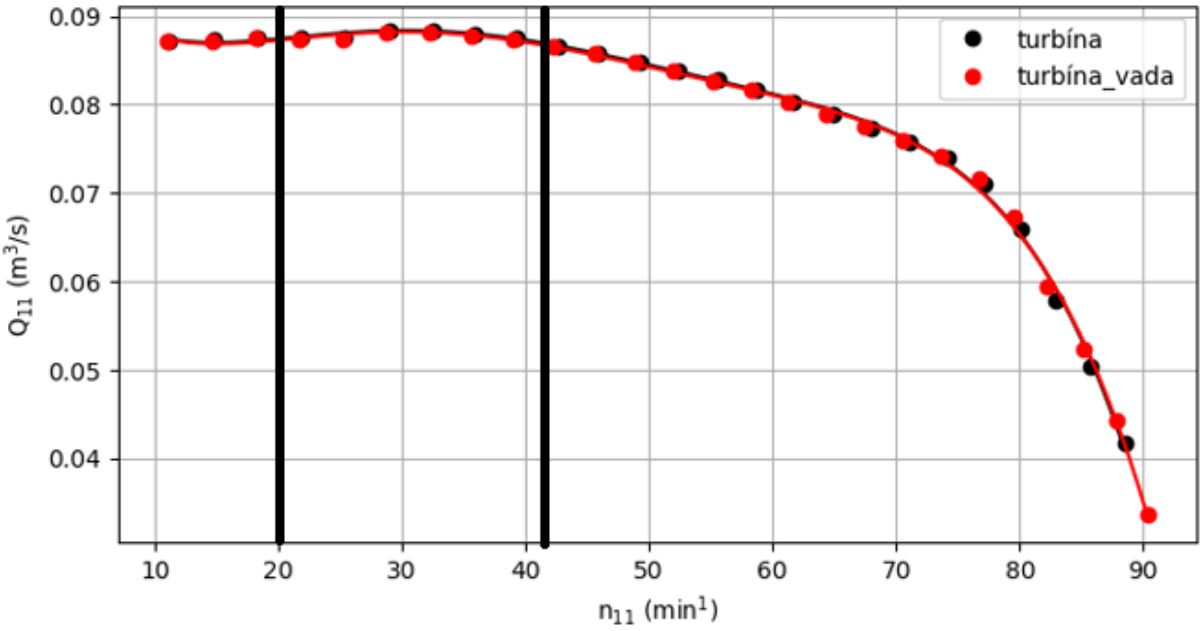
Při zkoumání různých variant byla používána primárně varianta s dělením zpracovaných dat, protože výhoda lepší reprezentace všech dat se zdála významnější. Především pak, protože přesnosti na testovacích sadách jednotlivých modelů byly vnímány zejména relativně jeden k druhému. Navíc byla snaha testovat modely na zcela jiných datech, jak je popsáno dále v práci, což je pak považováno za nejvíce směrodatné. Zmíněná varianta dělení již zpracovaných dat je tedy použita, pokud v textu není uvedeno jinak.

Pokud by se vzaly v potaz výhody, nevýhody a rizika, jsou obě varianty relevantní. Při používání MEL-spektrogramů jsou tyto přístupy totožné, protože jednotlivé MEL-spektrogramy jsou vždy vytvořeny z celého souboru.

4.2 Metody vyřazování skupiny souborů

Byla snaha vytrénované modely testovat nejen na datech z testovací sady, ale také na datech, která byla vyřazena z celého procesu učení/validace/testování. Vyřazení některých souborů navíc bylo v některých případech nutné z důvodu velkého objemu dat, na který nestačila výpočetní kapacita, zejména u čerpadlového režimu. Výhodou těchto dat bylo, že byla vybírána řízeně a jejich charakter byl předem známý. Na základě tohoto charakteru pak bylo možné interpretovat dosažené výsledky. Na základě výsledků je snaha diskutovat, jak efektivní by byl model v případě, že by oběžné kolo obsahovalo jinou vadu než provrtání lopatky na konkrétním místě.

U turbínového režimu bylo výrazně méně možných variant vyřazení souborů, protože naměřených dat bylo celkově méně. Naměřené stavy byly celkem tři – měnil se pouze tlak v kotli – 0,6 bar, 1 bar a 3 bar. Při každém tlaku byl udržován stálý spád 37 m a byly proměřeny všechny otáčky – od zhruba 200 ot./min do zhruba 2800 ot./min s krokem 100 ot./min. Pro každé nastavení otáček se naměřily dva čtyřsekundové záznamy a celkem se tedy pro každý tlak v kotli naměřilo 52 souborů. Celková velikost datových záznamů je 9,94 GB. Pro zkoumání přesnosti modelu na zcela neznámých datech bylo přistoupeno k metodě, kdy byl náhodně vybrán TDMS soubor a následujících 10 minut záznamů se vyřadilo. To odpovídalo 14 vyřazeným souborům při vylučování souborů bez vady, respektive 21 vyřazeným souborům při vylučování souborů s vadou. Byla tedy vyřazena část naměřené charakteristiky pro jeden tlak v kotli. Tato část charakteristiky však byla proměřena na zbývajících dvou tlacích v kotli.

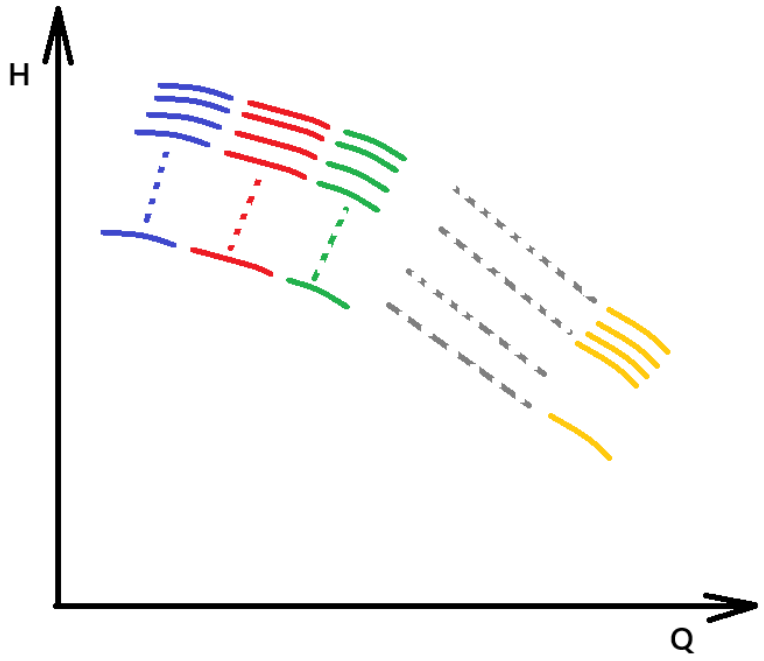


Obrázek 4-1: Závislost jednotkového průtoku na jednotkových otáčkách pro případ turbíny a turbíny s vadou, černými čarami je označena oblast, která byla vyjmuta vyřazením souborů, upraveno dle [29]

Toto rozdělení souborů je vhodné pro zjištění funkce modelu na neznámých datech, je však složité výsledky nějakým způsobem interpretovat vzhledem k použitelnosti v reálném provozu. Pro tyto interpretace je vhodnější měření z čerpadlového režimu, protože bylo naměřeno více dat a bylo možné si dovolit vyřazení více různých souborů. Proto se toto

testování na turbíně provedlo pouze u varianty se surovými daty a dále se pozornost věnovala spíše čerpadlovým datům.

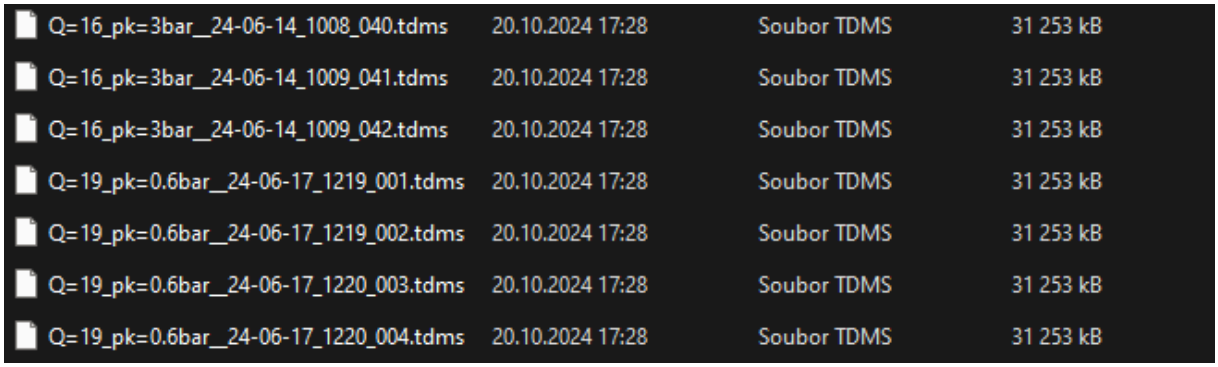
Pro čerpadlový režim byl vždy nastaven tlak v kotli a jmenovitý průtok, následně se proměřily otáčky. O konkrétní kombinaci jmenovitého průtoku a tlaku je v této práci hovořeno jako o stavu měření, respektive jenom jako o stavu. Tlak v kotli byl, stejně jako pro turbínu, 0,6 bar, 1 bar a 3 bar. Jmenovitý průtok byl měřen pro 1 l/s, 4 l/s, 7 l/s, 10 l/s, 13 l/s, 16 l/s, 19 l/s a 21 l/s. Celkem je tedy naměřeno 24 různých stavů. Pro každý stav byly proměřeny otáčky od zhruba 300 ot./min do zhruba 2300 ot./min, s posunutím vždy o 100 ot/min. Každý stav tedy obsahuje 21 měření za různých otáček. Je dobré podotknout, že skutečný průtok není takový, jaký je uvedený v označení souboru (1 l/s, 4 l/s atd.), zde se jedná o průtok při jmenovitých otáčkách 1500 ot./min. Skutečný průtok čerpadlem je samozřejmě dán protnutím charakteristiky čerpadla (ovlivněna otáčkami) a charakteristiky tratě (ovlivněna škrcením nastaveným při jmenovitých otáčkách). Pro každý nastavený jmenovitý průtok je tak naměřena část 21 charakteristik pro různé otáčky. Při nastavení dalšího jmenovitého průtoku jsou naměřeny další navazující části těchto charakteristik. To vše je provedeno třikrát pro různé tlaky v kotli. Pro lepší představu jsou data demonstrována na následujícím obrázku:



Obrázek 4-2: Znázornění čerpadlových dat

Řekněme, že obrázek zachycuje měření při konstantním, konkrétním tlaku v kotli, například 0,6 bar. Každá barva pak zachycuje měření při jednom konkrétním stavu, tedy při jednom konkrétním nastavení jmenovitého průtoku. Modré křivky pochází například ze souborů označených jako *Q = 1, p = 0.6 bar*, červené ze souborů označených jako *Q = 4, p = 0.6 bar*... a žluté ze souborů *Q = 21, p = 0,6 bar*. Křivek každé barvy je 21, každá pro jiné otáčky. Barev, tedy nastavených jmenovitých průtoků, je v obrázku 8. Tato data byla naměřena pro všechny tři tlaky v kotli. Pro každé nastavení statiky, tedy nastavení otáček při konkrétním tlaku v kotli a jmenovitém průtoku, byly dynamickými senzory naměřeny dva čtyřsekundové soubory.

Soubory ve svém názvu nesou informaci o průtoku, tlaku v kotli a pořadí v daném stavu měření, mimo to také datum a čas měření:

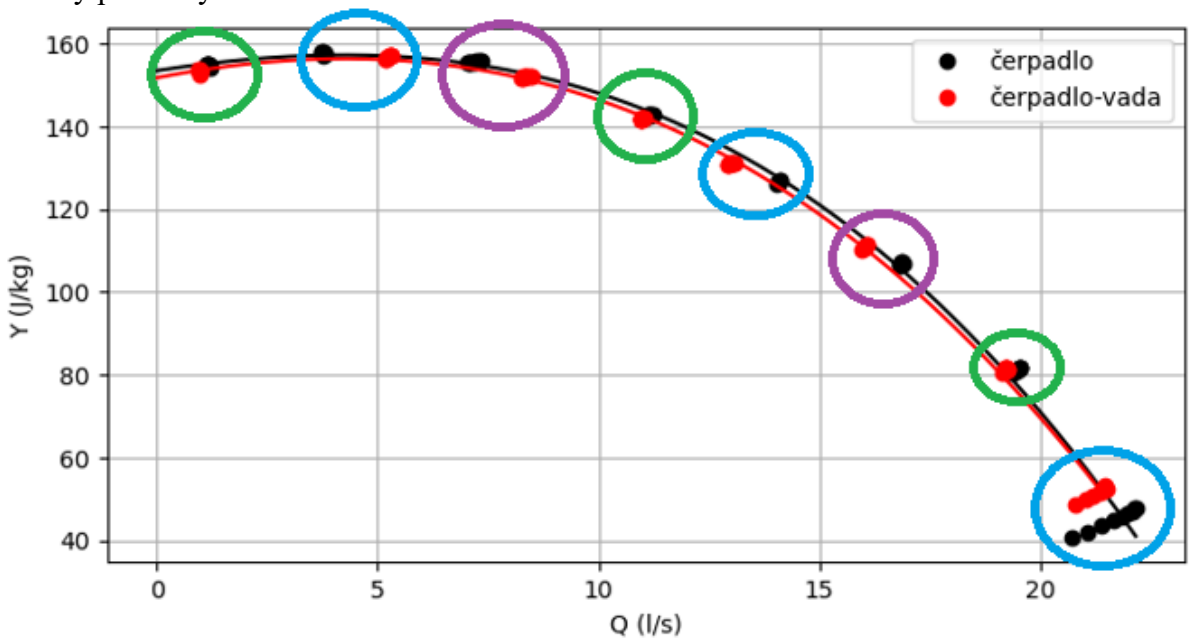


Obrázek 4-3: Ukázka pojmenování souboru

Byly zvoleny dva přístupy redukce – první, vyřazení každého druhého souboru. V tomto případě je vyřazena polovina souborů. Pro každé nastavení statiky – mimo jiné nastavení otáček – byly změřeny vždy dva soubory. To znamená, že každý vyřazený soubor byl naměřen za stejné statiky jako jiný soubor použitý při učení. Přestože jsou tyto dva soubory měřeny při stejných podmínkách, jejich hodnoty se lehce odlišují. Při testování natrénovaného modelu tedy testujeme na datech, která odpovídají známému stavu, ale přitom jsou to data neznámá. Podobných výsledků by bylo pravděpodobně dosaženo na datech naměřených s neznámou vadou, pokud by byl model trénován na širokém spektru vad, z nichž by se některé blízce podobaly vadě na těchto testovacích datech. O tomto přístupu se v práci hovoří jako o testování na neznámých souborech, případně jako o vyřazení poloviny souborů. Oproti tomu se o druhé variantě hovoří jako o testování na neznámých stavech.

Druhý přístup – vyřazování jednotlivých stavů. V tomto případě byly vyřazeny dvě třetiny dat. Algoritmus vyřazování souborů byl následovný – pro každý jmenovitý průtok byl k tréninku ponechán pouze jeden tlak v kotli. Byly ponechány kombinace 1 l/s a 0,6 bar, 4 l/s a 1 bar, 7 l/s a 3 bar atd. Pro každou kombinaci byly ponechány všechny soubory, tedy celé spektrum otáček. Lze tedy říct, že model je sice trénován na kompletních charakteristikách čerpadla, ovšem jednotlivé části konkrétní charakteristiky jsou měřeny za jiných tlaků v kotli. Model tak sice dostává informace o celém chodu čerpadla, ale nedostává informace o všech stavech. Pro lepší představu jsou na následujícím obrázku znázorněna data, na kterých je model trénován. Graf znázorňuje charakteristiku čerpadla pro konkrétní otáčky a barevnými kroužky jsou rozlišeny tlaky v kotli, při kterých byl úsek měřený. Je nutno zdůraznit, že obrázek slouží pouze pro demonstraci přístupu.

Konkrétní data v grafu odpovídají jmenovitým otáčkám 1500 ot./min a jsou měřena při konstantním tlaku v kotli. Stejně tak si zle představit, že na obrázku 4-2 jsou jednotlivé barvy měřeny při různých tlacích v kotli.



Obrázek 4-4: Závislost měrné energie na průtoku pro případ čerpadla a čerpadla s vadou, každá barva označuje jiný tlak v kotli, upraveno dle [29]

Při testování na vyřazených datech pak model naráží na stavy, které zná, ovšem s nějakou odlišností. Tento přístup by mohl simulovat situaci, kdy je model trénován na méně rozmanitém vzorku vad a vada nacházející se v provozu je odlišná, nicméně stále nese určité společné rysy s vadami z tréninku. Jak bylo řečeno, o tomto přístupu se v práci hovoří nejčastěji jako o testování na neznámých stavech apod.

Podle použité metodiky je tedy vyřazena určitá skupina souborů s vadou a určitá skupina souborů bez vady. Natrénovaný model je následně testován na těchto dvou skupinách zvlášť. Díky tomu je získáván přehled o tom, jak dobře model funguje při rozeznání obou chodů stroje.

I když byl model trénován se záměrem testování na vyřazených datech, byla data určená ke tvorbě modelu i tak rozdělena na trénovací, validační a testovací sadu. Velký význam byl pak přisuzován porovnání přesnosti na testovací sadě dat a přesnosti na vyřazených datech. Ukázalo se totiž, že na testovací sadě lze dosáhnout velmi vysokých přesností, a výzvou tedy zůstalo tyto přesnosti replikovat na neznámých datech. Přesnosti na testovacích sadách klesají s redukcí dat použitých k tréninku. Pokud se však na neznámých datech přesnosti (třeba že o něco nižší) budou blížit, značí to využitelnost konkrétního přístupu.

Při vyhodnocování přesnosti modelu binární klasifikace se pracuje s tzv. thresholdem neboli prahem, který určuje, k jaké třídě bude výstup přiřazen. Pokud target 0 znamená chod bez vady a target 1 chod s vadou a threshold bude nastaven na 0,5, pak bude při výstupu z modelu vyhodnocováno, zda je výsledek <0,5, pak bude vzorek označen za chod bez vady, nebo >0,5, pak bude označen za vadu. Základní nastavení v Tensorflow je threshold 0,5 [34] [35]. Toto nastavení bylo v případě testování na testovací sadě dat ponecháno. Při testování na vyřazených datech byl skript napsán tak, aby jako vada byly označeny výstupy nad 0,6 a za chod bez vady výstupy pod 0,4. Prvky, které dávaly výstup mezi 0,4 a 0,6, byly označeny jako nejednoznačné, aby se lépe poznalo, jak jistý model je. Při vyčíslení přesnosti jsou nejednoznačné výsledky konzervativně považovány za nesprávně určené.

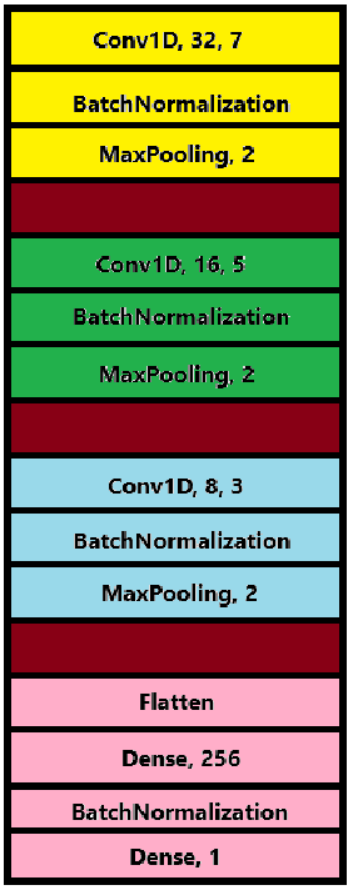
4.3 Zpracování surových dat konvoluční neuronovou sítí

Zpracování surových dat má řadu výhod i nevýhod. Jedná se o robustní, konzervativní metodu, při které nedochází k redukci informací předchozím zpracováním. Na druhou stranu je to metoda výpočetně náročná především ve smyslu vysokých nároků na výpočetní paměť a výpočetní čas při tréninku. Zároveň pro model neuronové sítě může být náročnější nalézt vzory, pokud neproběhne v předzpracování dat jejich zdůraznění tak, jak k tomu dochází například při použití MEL-spektrogramů.

V tomto přístupu byl použit model využívající 1D konvoluci. Protože do modelu bylo vstupováno primárně všemi pěti kanály zároveň, připadalo v úvahu využít také 2D konvoluci. 1D konvoluce je méně náročná na výpočetní paměť i na čas výpočtu. Na druhou stranu 2D konvoluce dokáže zachytit vzorce spočívající ve vazbách mezi jednotlivými kanály, oproti tomu 1D konvoluce zpracovává každý kanál zvlášť [36] [37].

V případě použití konvoluční neuronové sítě jsou data dělena do oken, která se mohou vzájemně částečně překrývat. Takto vytvořená okna představují časové řady, které jsou potom předkládány jako vstupy neuronové síti. V těchto časových řadách následně konvoluční síť hledá vzory pomocí různého počtu filtrů v jednotlivých vrstvách. Ve všech vrstvách byla použita aktivační funkce *ReLU*, až na poslední vrstvu, kde byla použita funkce *sigmoid*, aby se výsledek dal interpretovat jako pravděpodobnost výskytu poruchy.

Schéma použité konvoluční neuronové sítě je znázorněno na následujícím obrázku. Tato síť byla použita, pokud není uvedeno jinak. Popis konvoluce je ve tvaru „počet filtrů, kernel size“. Číslice u MaxPoolingu značí pool size. Pokud není uvedeno jinak, byl použit learning rate 0,001 a batch size 256.



Obrázek 4-5: Schéma použité konvoluční neuronové sítě

Velikost jednak oken a jednak velikost překryvu je jedna z významných otázek, která byla při řešení problému zkoumána. Protože byla vada vytvořena právě na jedné lopatce, zdálo se ideální, aby okno zahrnovalo celou otáčku, protože tak by zcela jistě obsahovalo projev přítomné vady. V případě turbíny byly nejnižší měřené otáčky zhruba 215 1/min, tedy zhruba 3,6 1/s. Data byla vzorkována s frekvencí 200 kHz a jeden soubor byl měřen 4 sekundy (obsahuje 800 000 záznamů), jedna sekunda tedy zahrnuje 200 000 záznamů, jedna sekunda zároveň odpovídá nejméně třem celým otáčkám oběžného kola – aby byla jistota, že v datovém okně je zachycena minimálně jedna otáčka, musí mít datové okno velikost alespoň 60 000 záznamů. Dále se ovšem ukázalo, že to nutné není, jak je popsáno dále v práci. Hodnota 60 000 je zároveň poněkud vysoká a může nastat problém s nedostatkem dat, respektive s nedostatkem těchto oken. Velikost datové sady je navyšována právě překryvem. Posuv však nesmí být příliš malý, protože v takovém případě může docházet ke generaci velmi podobných datových oken, což může vést k přeučení neuronové sítě. Volba této hodnoty je tedy otázkou iterace. Volba velikosti okna a velikosti posuvu byla volena tak, aby se okna tvořila vždy pouze z dat z jednoho souboru.

Také musí být vzata v potaz prostorová náročnost při učení modelu, neboť ta roste nepřímo úměrně velikosti kroku posuvu datového okna. Prostorovou náročnost lze vyjádřit následovně:

$$S\left(\left(\frac{(T-L)}{S}+1\right)\times L\times C\right) \tag{4-1}$$

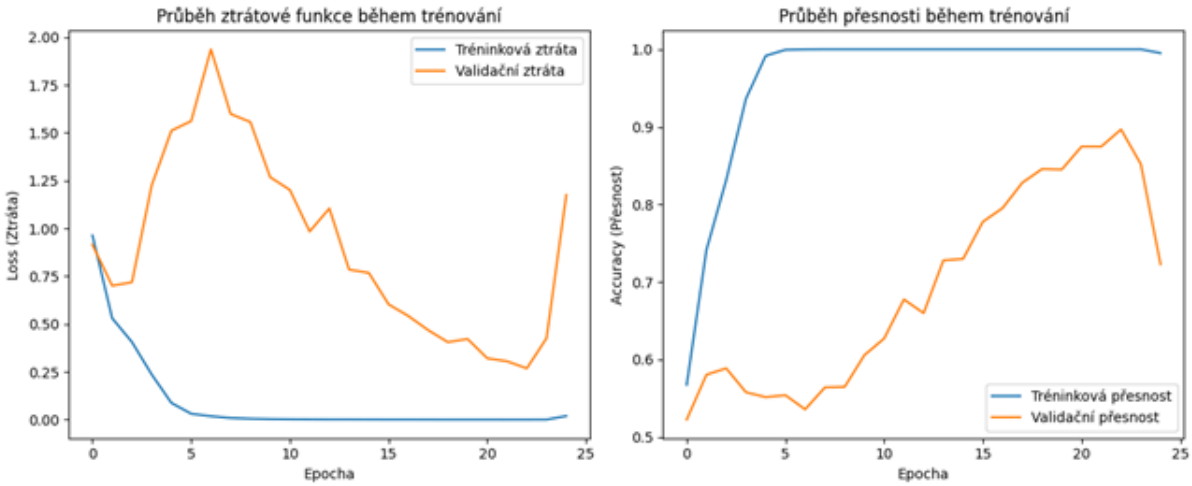
Kde T je celkový počet záznamů, L je délka datového okna, S je velikost kroku posunutí a C je počet kanálů. Problém s nedostatkem pracovní paměti se projevoval jak při vyhodnocování dat měřených v čerpadlovém, tak turbínovém režimu. Významnější byl však tento problém u čerpadlových dat, protože měla celkovou velikost 60,0 GB, zatímco turbínová data pouze 9,94 GB. Na druhou stranu, vzhledem k většímu objemu dat mohly být na čerpadlových datech testovány různé redukční metody, například podvzorkování z původních 200 kHz. U turbíny již nebylo efektivní data redukovat.

V následujících iteracích, kdy bylo hledáno funkční nastavení neuronové sítě, jsou v první variantě nejprve načteny všechny soubory a z těch jsou následně vytvořena okna o určitém rozměru. Tato okna vstupují do 1D konvoluce. V těchto případech probíhá třídění do testovací, validační a trénovací sady dat až po vytvoření těchto oken – jsou tedy tříděna už jednotlivá připravená okna. Dále v práci je testována také varianta, kdy jsou data tříděna do sad již na úrovni souborů.

K této variantě se vztahují kódy ve složce „Konvoluce surových dat“ v Příloze 7. Kódy jsou pojmenovány intuitivně, aby šla znát jejich funkce, ta je také shrnuta v úvodním komentáři přímo v kódu.

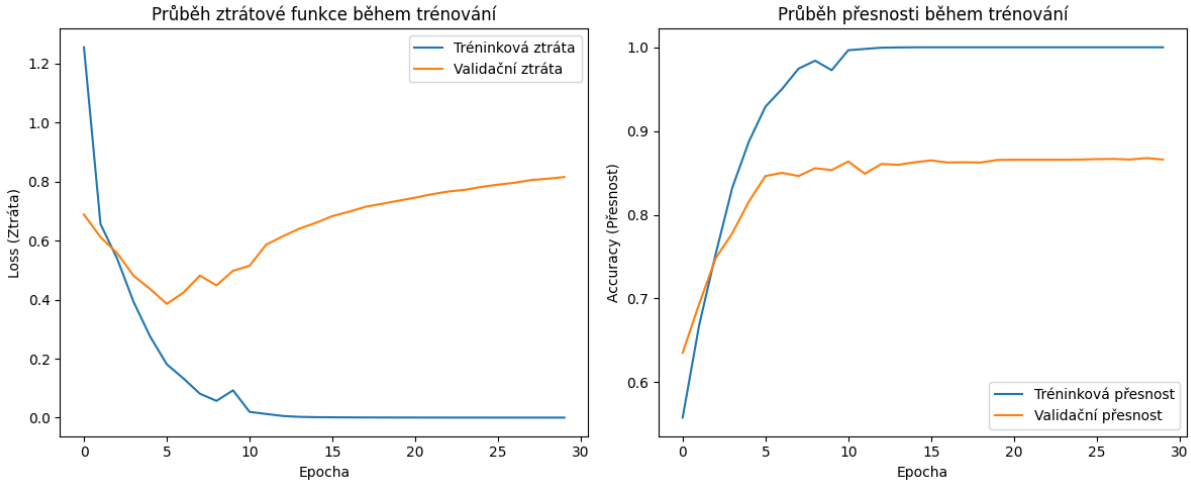
4.3.1 Data z turbínového režimu

Nejprve byla spočítána varianta, kdy datové okno mělo rozměr 50 000 záznamů a délka kroku posunu 25 000. Síť dosáhla na testovací sadě přesnost 72,3 %. V závěru učení došlo k prudkému nárustu ztrát a poklesu přesnosti na validačních datech. Zřejmě se tedy model začal rapidně přeučovat.



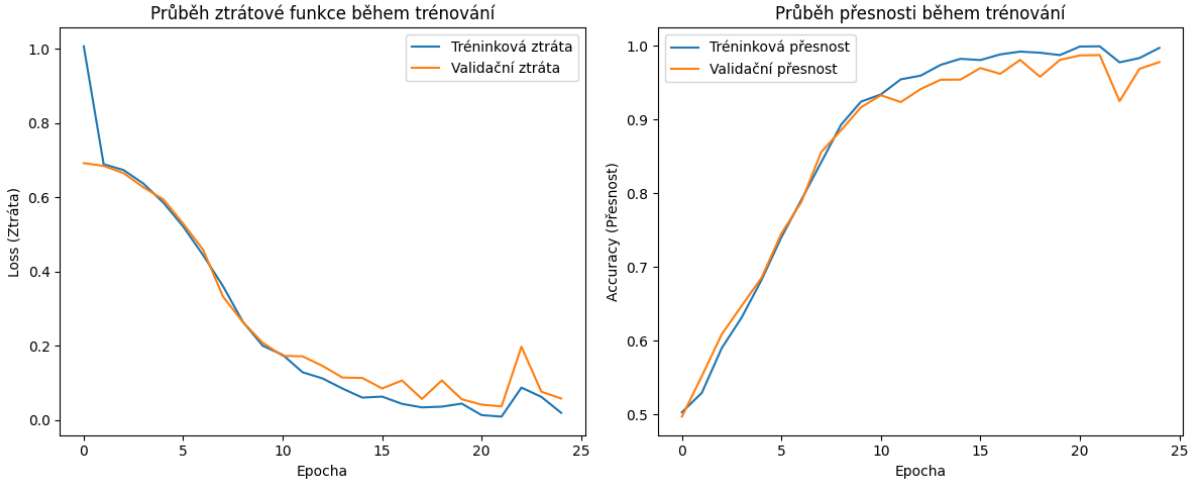
Obrázek 4-6: Průběh ztrátové funkce a přesnosti při velikosti datového okna 50 000 a posuvu okna 25 000; 256 neuronů v plně propojené vrstvě

Proti tezi o minimální délce okna jdou pokusy s neuronovou sítí pracující s délkou okna 10 000 a bez překryvu (tedy s krokem posuvu také 10 000). Pravděpodobně tedy nedochází k projevu vady pouze v jednom okamžiku například při průchodu poškozené lopatky kolem jazýčku voluty, ale déle po dobu jakéhosi doznívání, případně cyklicky, ale častěji než jednou za otáčku. S tímto nastavením byla síť aplikována v několika variantách. První varianta obsahovala v plně propojené vrstvě za konvolucí 512 neuronů. V tomto případě se dosáhlo relativně dobrých výsledků, na testovací sadě byla přesnost 87,4 %. Při učení sítě bylo nicméně znatelné přeučení:



Obrázek 4-7: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; 512 neuronů v plně propojené vrstvě

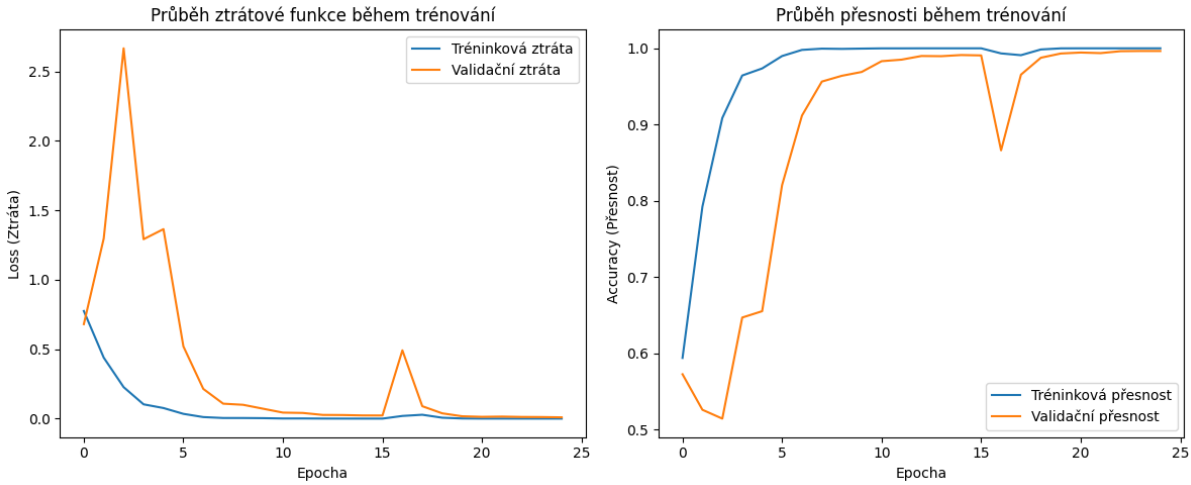
Model s 512 neurony byl zřejmě poněkud složitý, proto docházelo ke zřetelnému přeučení modelu. Byl tedy snížen počet neuronů na 256. Přeučování tím bylo zřejmě zamezeno. Přesnost dosahovala na testovací sadě 97,9 %:



Obrázek 4-8: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; 256 neuronů v plně propojené vrstvě

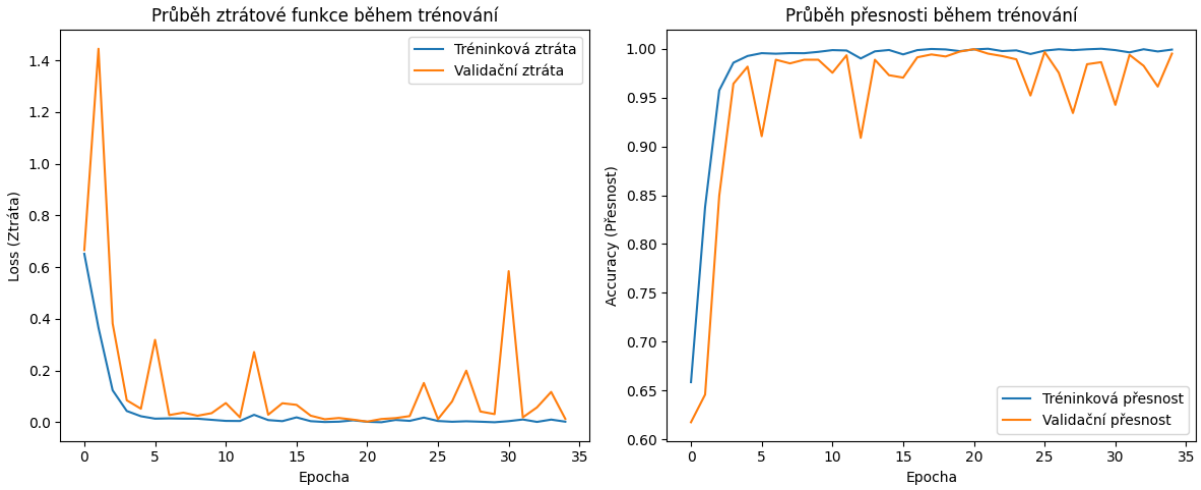
V poslední variantě byla ještě implementována normalizace batchů. V tomto případě se při učení model nepřeučoval, křivka ztrát a přesností méně oscilovala a model byl naučený po menším počtu epoch. Jediný náznak problému nastal okolo šestnácté epochy.

Na grafech lze pozorovat „zub“ na validačních křivkách. Model se ovšem okamžitě sám zase stabilizoval, a to hned při další epoše.



Obrázek 4-9: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; 256 neuronů v plně propojené vrstvě; Batch normalizace

Takový model se zdá s přesností na testovací sadě 99,8 % velice úspěšný. V následující fázi byl tedy model testován na datových setech, které vznikly nejprve rozdělením samotných souborů na skupiny trénovací, testovací a validační, a až posléze byla vytvořena konvoluční okna. Na testovací sadě pak bylo dosaženo přesnosti 99,6 %.



Obrázek 4-10: Průběh ztrátové funkce a přesnosti při rozdělení samotných souborů

Nastala však otázka, jak přesný by model byl v případě, že by byl vystaven datům, která odpovídají sekvenci po sobě jdoucích souborů, které model při tréninku neviděl. Proto byl naprogramován program, který náhodně vybral TDMS soubor (800 000 záznamů během 4 sekund), a vyřadil jej a další soubory, které byly měřeny v následujících 10 minutách. To odpovídalo 14 vyřazeným souborům při vylučování souborů bez vady, respektive 21 vyřazeným souborům při vylučování souborů s vadou. Na zbytku souborů byl model vytrénován a následně uložen. Poté se uložený model aplikoval na vyřazené soubory, přičemž se sledovalo, jak přesně předpoví, zda se jedná o soubory s vadou nebo bez vady – při vyřazení souboru bylo samozřejmě známo, zda se vyřazují soubory s vadou či bez vady. Pochopitelně, a jak se očekávalo, přesnost byla v takovém případě nižší – během trénování modelu byla určena testovací přesnost 97,0 % při vyřazení souborů bez vady a 92,6 % při vyřazení souborů s vadou. Snížení nastalo zřejmě snížením množství trénovacích dat. Modely byly testovány na vyřazených souborech. V případě rozeznávání vady bylo jako vada rozpoznáno 1216 datových oken, 81 bylo neurčitých a 383 bylo chybně označeno jako chod bez poruchy. Tedy s prahem rozhodování 0,6 je přesnost 72,4 %. Při rozeznávání stavu bez poruchy bylo správně označeno 1074 datových oken, nesprávně za vadu 33 datových oken a ve 13 případech byl výsledek neurčitý. Tedy s přesností 95,9 %.
Jak se očekávalo, na datech, která model při trénování nepotkal, je přesnost nižší.

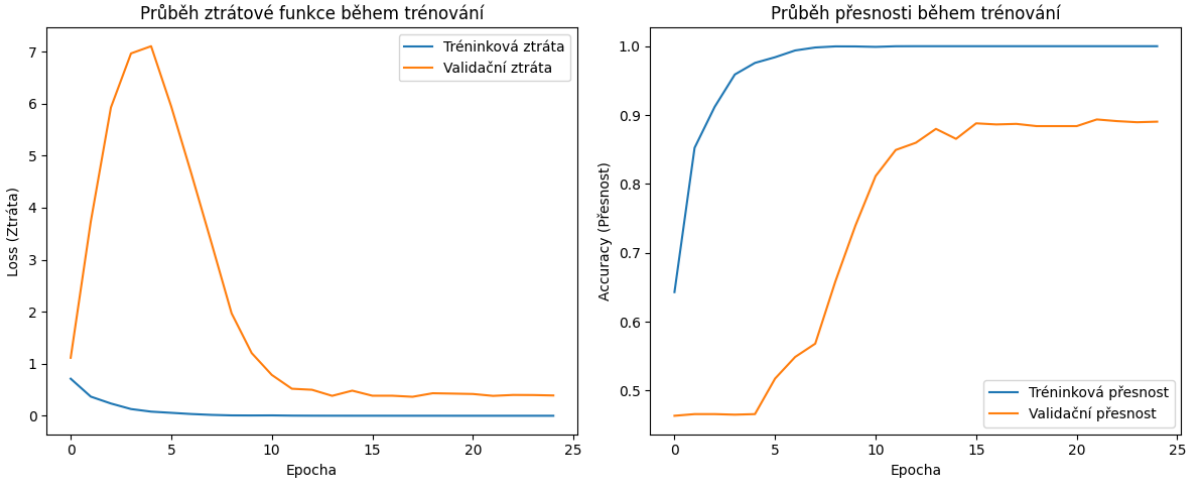
4.3.2 Data z čerpadlového režimu

Protože dat pro čerpadla bylo výrazně více než pro turbínu, musel být zvolen jiný přístup v jejich zpracování. Nebylo možné zpracovat kompletně všechny naměřené hodnoty tak jako u turbínových dat. Data tedy musela být zredukována vhodným způsobem. Při redukci musí být proveden vhodný kompromis mezi snížením prostorové náročnosti a zachováním podstatných informací, které samozřejmě s redukcí zanikají.

Nastavení sítě bylo stejné jako v případě poslední varianty u turbínového režimu – tedy tak, jak je v úvodu zobrazeno na schématu.

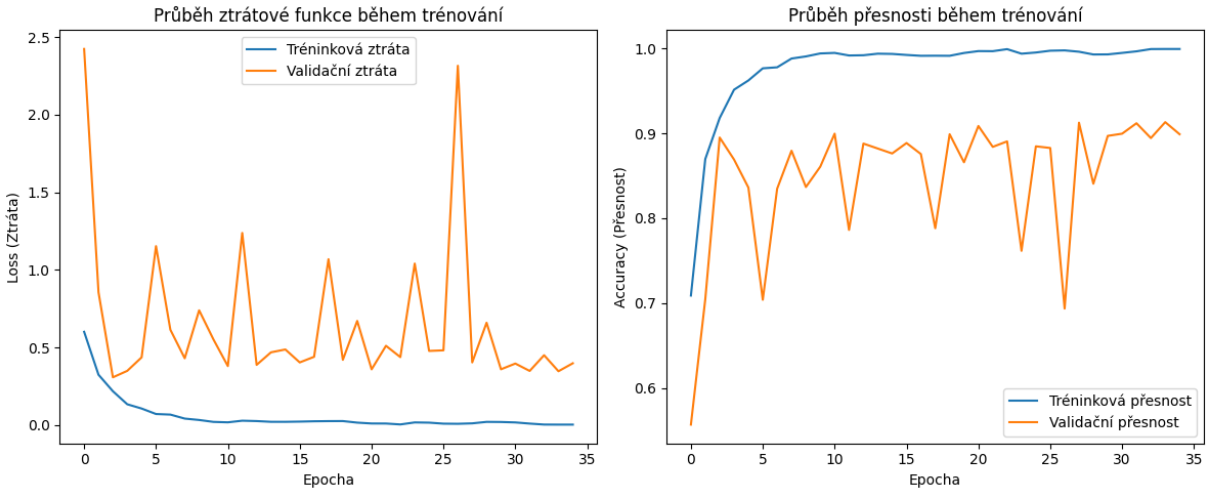
Prvním způsobem redukce bylo podvzorkování. Data byla převzorkována z původních 200 kHz na 20 kHz. Výhodou této metody je zachování rovnoměrnosti dat ze všech částí měření. Data budou pocházet ze všech bodů charakteristiky čerpadla a ze všech variant tlaků v kotli. Další výhoda spočívá v usnadnění splnění hypotetické podmínky zahrnutí otáčky do

datového okna, ačkoliv tato podmínka zřejmě není nutná k funkčnímu modelu. Stejně dlouhé datové okno při podvzorkování z 200 kHz na 20 kHz zahrnuje desetkrát delší časový úsek, a tedy desetkrát víc otáček. Naopak nevýhodou metody je ztráta potenciálních jevů, které se vyskytují na vyšších frekvencích.
Výsledná přesnost na testovací sadě je u této metody 91,5 %, což je poměrně uspokojivá hodnota.



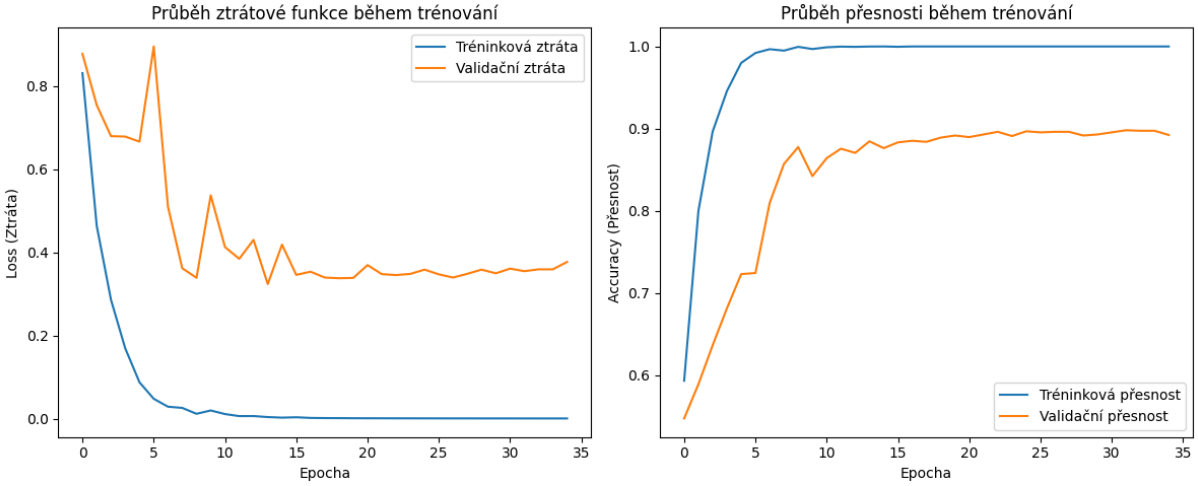
Obrázek 4-11: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Podvzorkování na 20 kHz

Totéž bylo provedeno i s variantou, kdy byly do testovací, validační a trénovací sady děleny přímo soubory, přesnost na testovací sadě byla v tomto případě 87,2 %. Je však zřetelné kmitání validačních křivek.



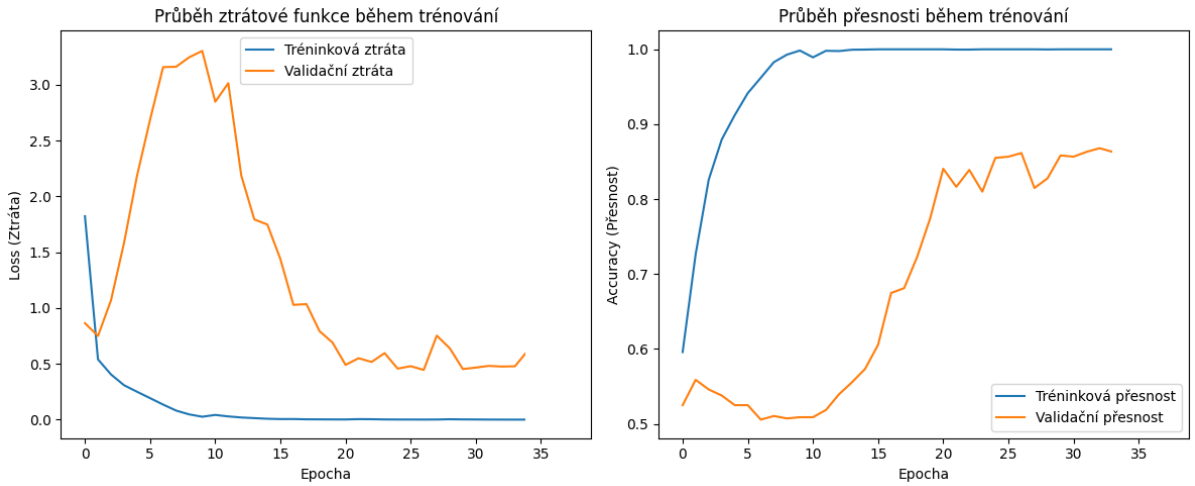
Obrázek 4-12: Průběh ztrátové funkce a přesnosti při podvzorkování na 20 kHz, při rozdělení samotných souborů

Takovéto kmitání validační ztráty může být důsledkem příliš velkého learning rate, ten byl v tomto případě 0,001. Došlo tedy ke snížení learning rate na 0,0005, což přineslo následující průběh učení. Přesnost na testovací sadě také narostla na 89,8 %:



Obrázek 4-13: Průběh učení při snížení learning rate na 0,0005

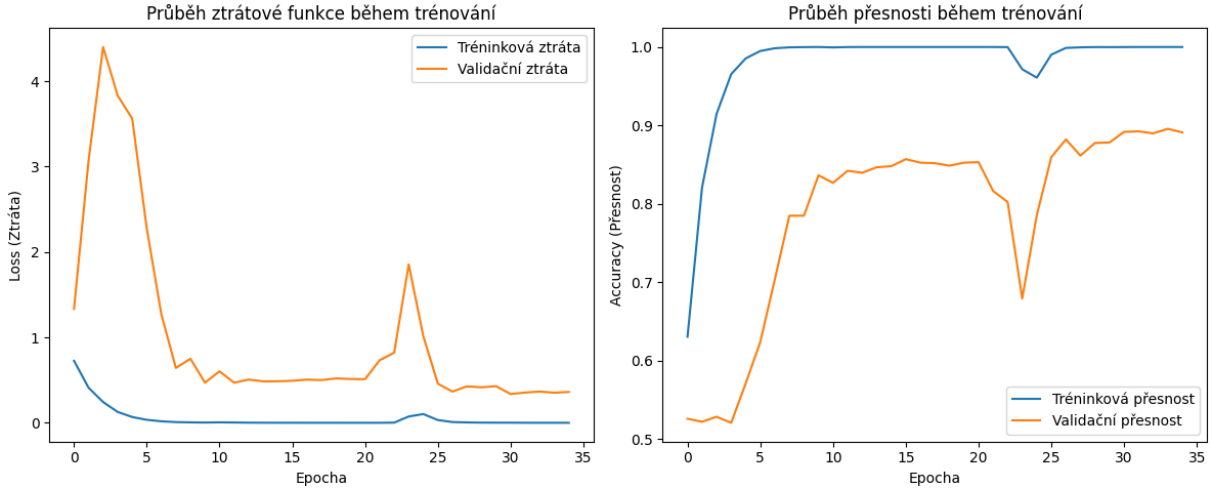
Další možnost, jak velikost dat zredukovat, byla vybrat pouze některé soubory. Tento přístup byl nejprve zkombinován s podvzorkováním na 20 kHz pro porovnání se zpracováním všech souborů. Jak bylo popsáno výše v práci, byl k učení vybrán každý druhý soubor. Model se tedy při učení setkal se všemi tlaky v kotli a celou charakteristikou, nicméně s méně daty v každém bodě. Testovací přesnost modelu byla 83,1 %. Data, která nebyla využita při učení, byla následně vyhodnocena uloženým modelem. Vada byla správně rozeznána ve 2 858 případech, neurčitý výsledek byl ve 196 případech a chybně byl prvek označen za provoz bez poruchy v 978 případech. Tedy správně byla vada zachycena v 70,9 % případů. Provoz bez vady byl správně určen v 2 859 případech, neurčitý byl výsledek ve 189 případech a v 672 případech byl prvek chybě určen jako vada. Přesnost je tedy 76,9 %.



Obrázek 4-14: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Podvzorkování na 20 kHz v kombinaci se selekcí souborů (každý druhý)

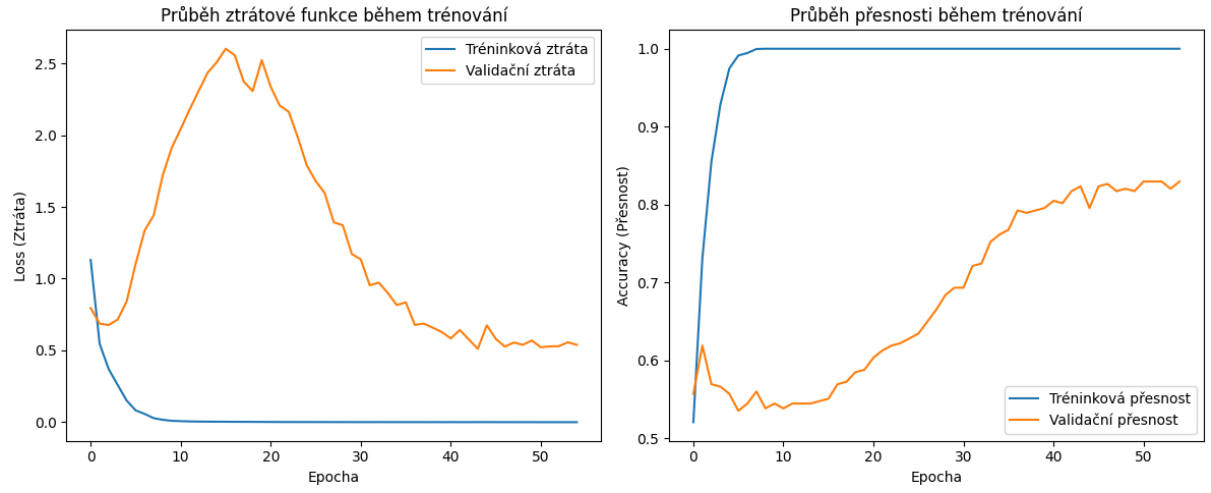
Stejný výběr souborů byl následně aplikován při podvzorkování na 50 kHz. Při tomto nastavení byla přesnost při učení na testovací sadě 90,5 %. Při rozeznávání vady bylo správně označeno 8 662 vzorků, neurčitě 354 vzorků a nesprávně 1 064 vzorků. Přesnost je tedy 85,9 %.

Při rozeznávání provozu bez poruchy bylo správně označeno 8 156 vzorků, neurčitě 241 vzorků a nesprávně jako vada bylo označeno 903 vzorků. Přesnost je tedy 87,7 %.



Obrázek 4-15: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Podvzorkování na 50 kHz v kombinaci se selekcí souborů (každý druhý)

V další iteraci byly vybrány soubory tak, aby byla pokryta celá charakteristika čerpadla, ale každá její část byla měřena za jiného tlaku v kotli. V tomto případě byly vyřazeny dvě třetiny dat, které byly opět využity k testování natrénovaného modelu. Nejprve byl tento případ opět zkombinován s podvzorkováním na 20 kHz. V tomto případě už šlo znát, že dat není optimální množství. Model se natrénoval na přesnost na testovací sadě 82,4 %. Na zahozených stavech byl správně určen provoz bez poruchy v 3 528 případech, neurčitý výsledek byl zaznamenán v 293 případech a chybně byla určena vada v 1 611 případech. Lze tedy hovořit o přesnosti 64,9 %. Při testování na datech s poruchou byla vada správně rozeznána v 4 689 případech, neurčitý výsledek nastal v 288 případech a v 1 071 případech byl chybně určen chod bez poruchy. Lze hovořit o přesnosti 77,5 %.

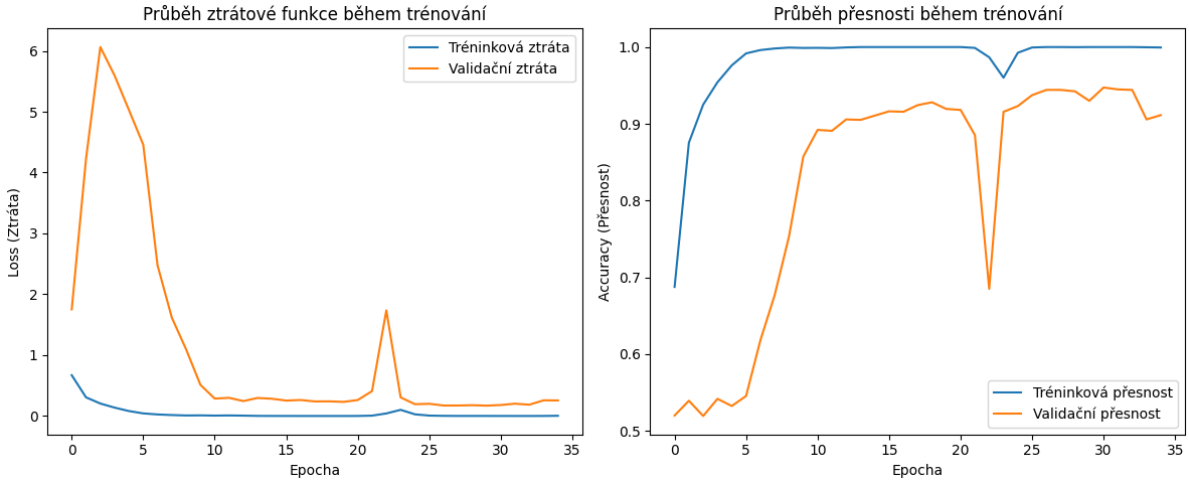


Obrázek 4-16: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Podvzorkování na 20 kHz v kombinaci se selekcí souborů – třetina souborů

Protože použité výpočetní kapacity nestačily na výpočet takového rozdělení souborů bez podvzorkování, byl tento případ zkoumán dále dvěma cestami. První cesta znamenala vzít již vyselektované soubory a vzít pouze každý druhý. Tedy celkem šestinu původních souborů. Druhá cesta vede přes snížení podvzorkování na 100 kHz. Obě varianty jsou prostorově stejně

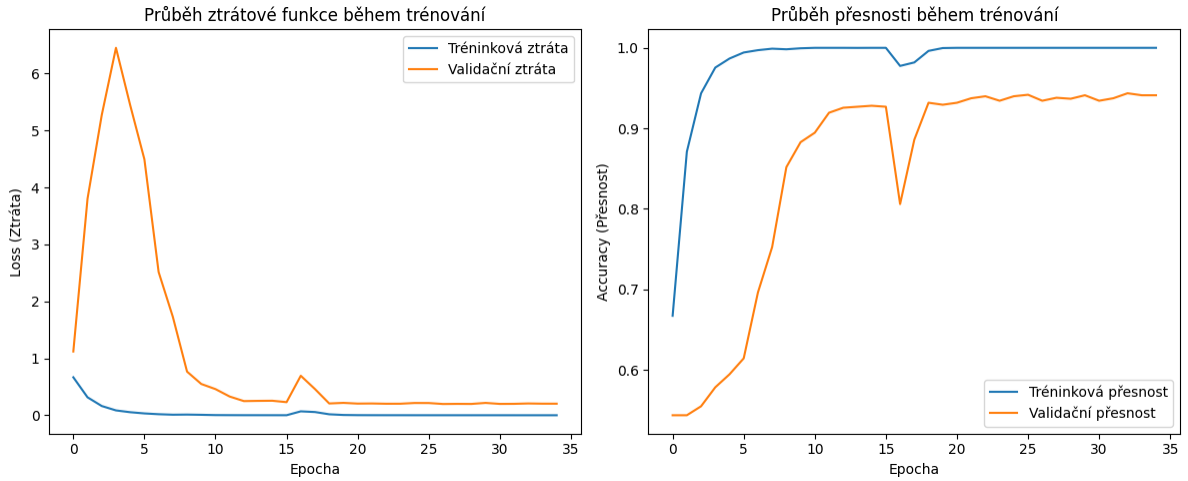
náročné. Využití pouze každého druhého souboru z již vyselektované třetiny zachovává možnost detekovat jevy, které se vyskytují s frekvencí až 200 kHz.

Varianta bez podvzorkování, kde byla při učení použita pouze šestina souborů, dosáhla na testovací sadě přesnost 91,6 %. Na vyřazených datech pak při testování na datech bez poruchy bylo správně rozpoznáno 8 896 vzorků, neurčitě 587 a v 4 117 případech byl chod označen chybně jako chod s vadou. Přesnost je tedy 65,4 %. Při testování na datech s poruchou bylo správně určeno 13 358 vzorků, neurčitých bylo 363 a nesprávně bylo označeno 1 399 vzorků. Přesnost je tedy 88,3 %.



Obrázek 4-17: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Vzorkování 200 kHz v kombinaci se selekcí souborů – šestina souborů

Dále byla řešena varianta s podvzorkováním dat na 100 kHz. V tomto případě byla přesnost při tvorbě modelu 93,9 %. Při testování na vyřazených stavech byl při chodu bez poruchy tento stav správně označen v 19 089, neurčitého výsledku bylo dosaženo v 833 případech a chybně byla zaznamenána porucha v 7 238 případech. Přesnost při určování stavu bez poruchy je tedy 70,1 %. Při chodu s poruchou bylo správně označeno 25 939 případů, neurčitý výsledek byl v 632 případech a v 3 669 případech byl chod určen chybně jako bez poruchy. Přesnost na těchto datech je tedy 85,8 %.



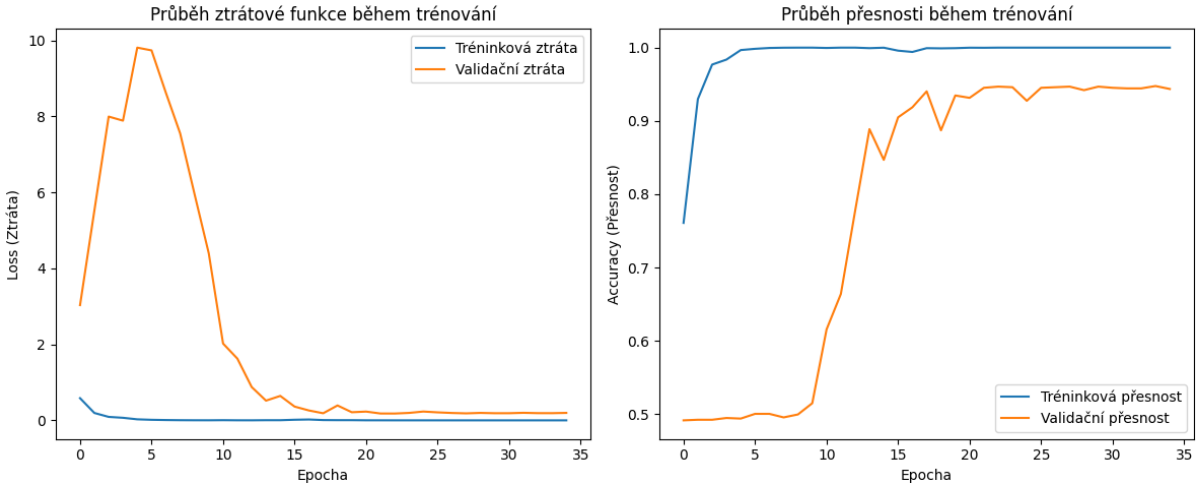
Obrázek 4-18: Průběh ztrátové funkce a přesnosti při velikosti datového okna 10 000 a posuvu okna 10 000; Podvzorkování na 100 kHz v kombinaci se selekcí souborů – třetina souborů

Další způsob je analyzovat, které kanály ovlivňují schopnost detekce vady nejvíce a které nejméně. Pro porovnání jednotlivých kanálů s variantou se všemi využitými kanály, byly všechny vyhodnoceny také s podvzorkováním na 20 kHz. Přesnosti modelů získaných za využití veškerých souborů jsou uvedeny v následující tabulce:

Tabulka 2: Přesnosti modelu při použití jednotlivých kanálů při podvzorkování na 20 kHz

Název kanálu	Přesnost modelu (testovací sada)
Všechny kanály zároveň	91,5 %
AE (akustická emise)	85,9 %
Mic-GRAS40PH (Mikrofon)	93,9 %
a-spiral (Snímač zrychlení na spirále)	75,7 %
p-PCB113B28-spiral (Snímač tlaku ve spirále)	62,2 %
p-PCB113B28-sani (Snímač tlaku na sání)	68,1 %

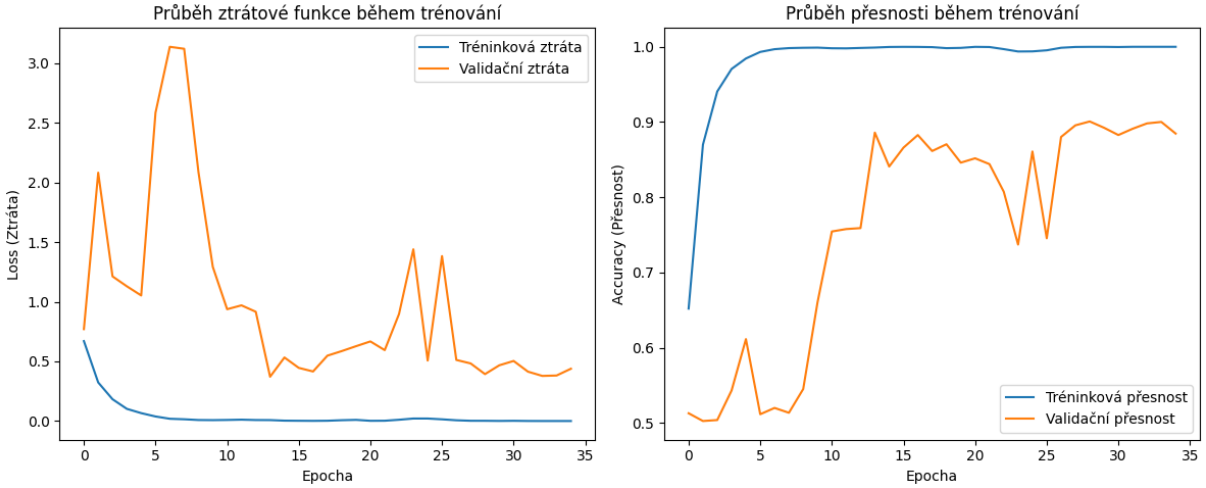
Nejvyšší přesnosti bylo dosáhnuto při použití mikrofonu a také při použití snímače akustické emise. Mikrofon samotný dosahuje dokonce vyšší přesnosti než použití všech kanálů zároveň. Rozdíl není markantní, může být částečně způsoben náhodnými jevy, ale určitý vliv může mít i potenciální nadbytečnost snímačů tlaků, které samy o sobě dosahují nižších hodnot přesnosti. Jak bylo řečeno, tyto kanály mohou přinášet šum, kterému se model také přizpůsobuje, ačkoliv není relevantní.



Obrázek 4-19: Průběh učení při použití pouze mikrofonu s podvzorkováním na 20 kHz

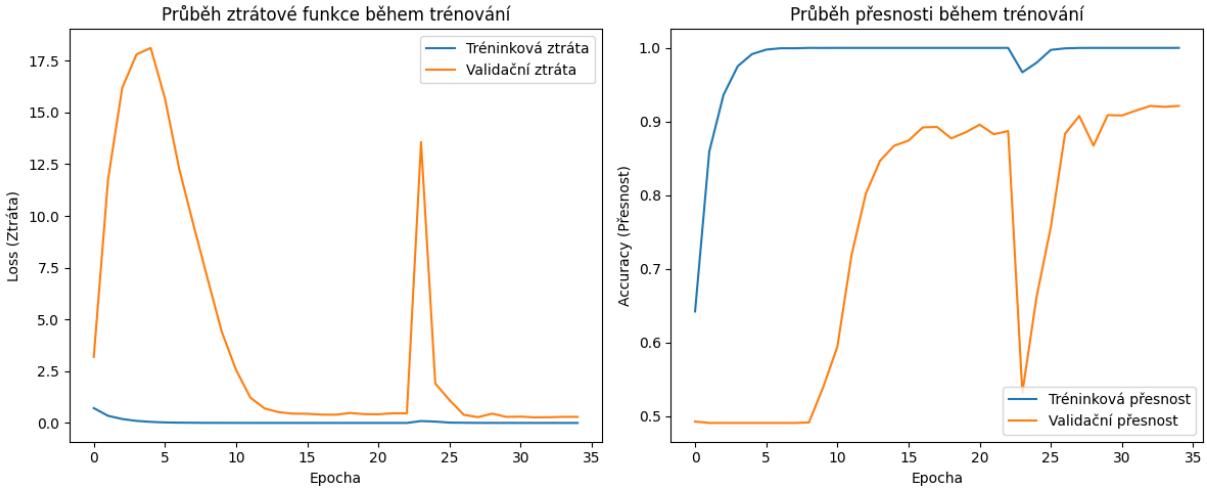
Protože mikrofon dosahoval nejlepšího výsledku, bylo testováno, jak model bude přesný na souborech a stavech, které nebyly zahrnuty v učení. Byl použit data set s polovinou souborů, kde byl každý druhý vyřazen k testování. V tomto případě bylo použito podvzorkování na 50 kHz, aby byl opět výsledek porovnatelný s případem, kdy byly použity všechny kanály. Během tréninku bylo na testovací sadě dat dosaženo přesnosti 89,6 %. Při testování na zahozených souborech byl provoz bez poruchy správně rozeznán v 8 386 případech, v 151 případech byl výsledek nejednoznačný a chybně byl výsledek označen v 763 případech. Přesnost je tedy v případě rozeznání chodu bez poruchy 90,2 %.

Vada byla správně rozeznána v 8 233 případech, v 377 případech byl výsledek nejednoznačný a v 1 470 byl chod označen nesprávně jako chod bez vady. Přesnost rozeznání poruchy je tedy 81,7 %.



Obrázek 4-20: Průběh učení při použití mikrofonu na polovině souborů a podvzorkování na 50 kHz

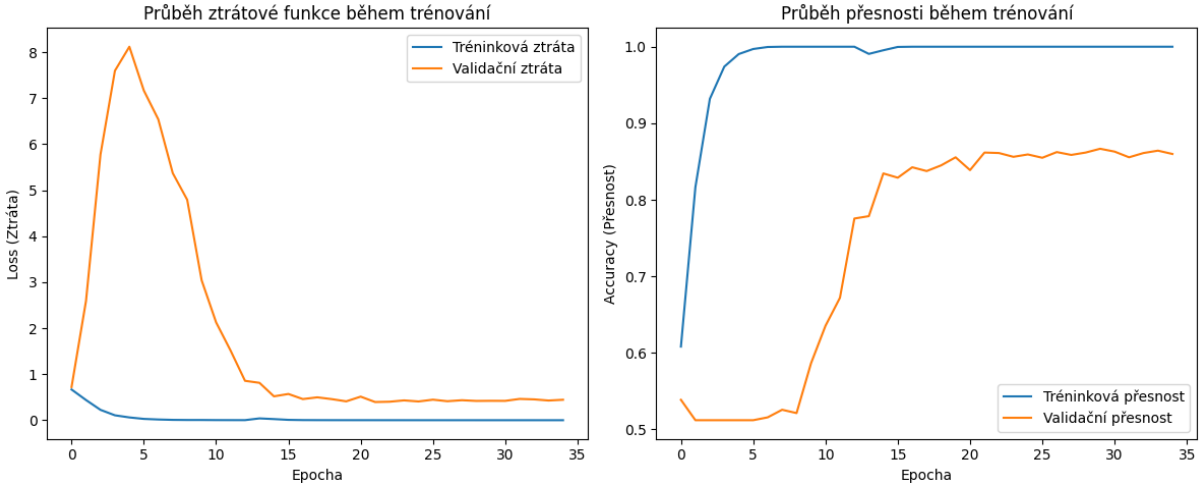
Následně, podobně jako v předchozích případech, byl model trénován na třetině dat při frekvenci 100 kHz. Na testovací sadě dat byla přesnost 91,6 %. Na zahozených stavech, které model při učení neviděl, však byly výsledky nesmyslné a téměř v každém případě byl výstup označen jako vada. Takový jev může všeobecně nastat v několika případech. První problém, který je potřeba očekávat při takovém výsledku, je nevyváženost dat. To je ovšem možné v našem případě vyloučit. Stejně tak lze vyloučit také problém se základním principem architektury modelu, protože model pracoval v předchozích případech spolehlivě. Problém, který se v našem případě zdál nejpravděpodobnější, bylo přeučení, ačkoliv se neprojevilo na validačních datech. Protože je použita pouze třetina souborů, pouze jeden kanál a k tomu podvzorkování na 100 kHz, nemusí být objem dat optimální [38] [39].



Obrázek 4-21: Průběh učení při použití mikrofonu na třetině souboru a podvzorkování na 100 kHz

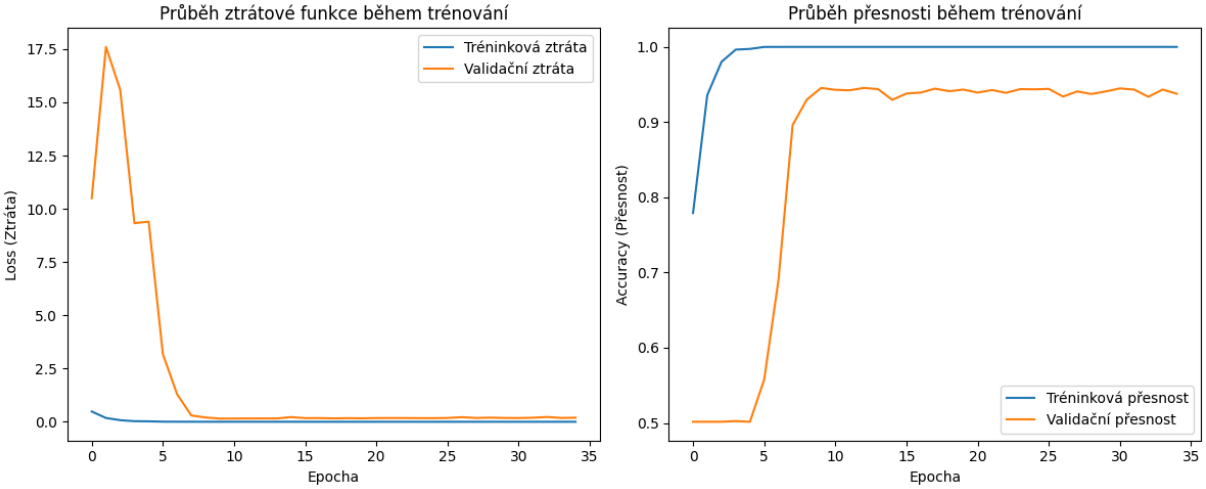
Na základě tohoto předpokladu byl zjednodušen model neuronové sítě – byly použity pouze dvě konvoluční vrstvy, kdy první používá pouze 16 filtrů namísto původních 32 a druhá používá pouze 8 filtrů namísto původních 16. Plně propojená vrstva pak byla zmenšena na 64 neuronů z 256. Tyto úpravy zřejmě zamezily přeučování. Na testovací sadě dat po trénování

modelu byla přesnost vyhodnocena na 86,6 %. Při zpracování dat s vadou z vyřazených souborů bylo správně určeno 25 171 segmentů, neurčitých výstupů bylo 1 074 a chybně model predikoval výstup v 3 995 případech. Přesnost byla tedy 83,7 %. Při rozeznávání chodu bez poruchy byla predikce správná v 20 464 případech, neurčitá v 983 případech a chybná v 5 713 případech. Přesnost tedy byla 75,3 %.



Obrázek 4-22: Průběh učení při použití mikrofonu na třetině souboru a podvzorkování na 100 kHz, použití zjednodušeného modelu

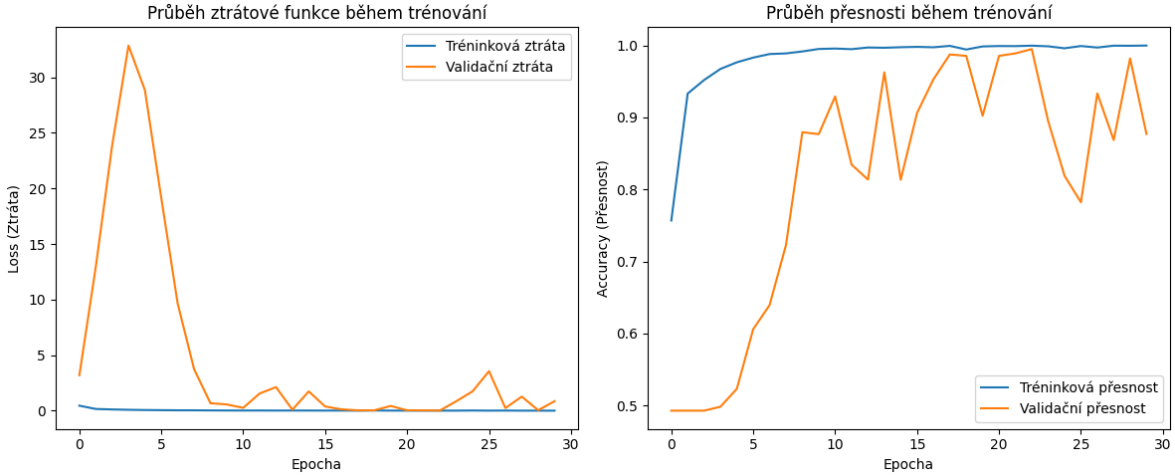
Zmiňovaný problém to tedy vyřešilo. Přesto byl původní model vytrénován znovu, tentokrát bez podvzorkování. Tedy se vzorkovací frekvencí 200 kHz a s původním nastavením. Vynecháním podvzorkování se zvyšuje objem dat, což může také zamezit přeučování. Na testovací sadě dat byla dosažena přesnost 94,1 %. Je vidět očekávaný nárust oproti situaci s podvzorkováním na 100 kHz. Po zvýšení frekvence se problém přeučování zdál být překonaný. Při rozpoznání stavu bez poruchy byl model správný v 43 895 případech, v 1 614 případech byl výsledek neurčitý a chybná predikce nastala v 8 811 případech. Přesnost při rozpoznání chodu bez poruchy byla tedy 80,8 %. Při rozeznání poruchy bylo správně rozeznáno 55 634 vzorků, neurčitý výstup byl v 818 případech a v 4 028 případech byla predikce chybná. Přesnost je tedy 92,0 %.



Obrázek 4-23: Průběh učení při použití mikrofonu na třetině souboru a bez podvzorkování

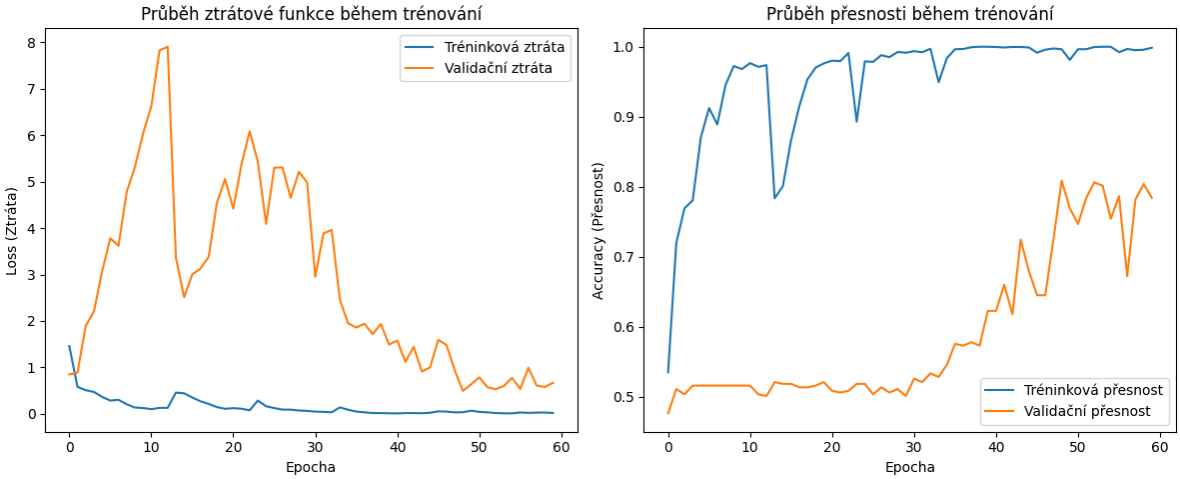
Pro porovnání byl při vzorkovací frekvenci 200 kHz na třetině souborů vytrénován i model pouze z dat ze senzoru akustické emise. Při tréninku se model opět potýkal s výrazným

oscilováním validačních ztrát, což částečně vyřešilo snížení learning rate na 0,0001. Přesnost na tréninkové sadě dat byla 87,57 %. Při predikci stavů bez vady byla přesnost 75,3 % (správně určeno 40 901 vzorků, neurčitě 1 155 a nesprávně 12 264 vzorků). Při predikci stavů s vadou byla přesnost 83,5 % (správně určeno 50 478 vzorků, neurčitě 3 519 a nesprávně 6483 vzorků).



Obrázek 4-24: Průběh učení při použití AE na třetině souboru a bez podvzorkování

V úvodu bylo řečeno, že použití více kanálů zároveň se sebou nese možnost aplikovat 2D konvoluci namísto 1D konvoluce. 2D konvoluce neaplikuje filtry neboli kernely, pouze přes záznam v časové dimenzi nad daty z jednoho kanálu, ale zároveň i přes tyto jednotlivé kanály. Vícero kanálů je tedy možné považovat za další dimenzi. Tím, že jsou filtry aplikovány i přes jednotlivé kanály, dokážou mezi těmito kanály zachytit případné vztahy, tedy další vzorce, které mohou dopomáhat v učení [36] [37]. Původní model využívající 1D konvoluci (znázorněný na schématu na úvodu kapitoly) byl převeden analogicky do 2D. Neuronová síť byla vytrénována na datech všech senzorů s podvzorkováním na 20 kHz na přesnost na testovací sadě 82,0 %. To je nižší hodnota, než jaké dosáhla 1D konvoluce. To je výsledek, který jde proti předpokladu, že vyhledání vztahů mezi jednotlivými senzory dopomůže učení. Výpočetní náročnost problému se výrazně zvýšila jak z hlediska času, tak z hlediska využití paměti. Z toho důvodu, a protože se výsledky nejevily slibně, se nezdálo rentabilní v této variantě pokračovat. Nelze ovšem vyloučit zlepšení, pokud by se například optimalizoval model jako takový.



Obrázek 4-25: 2D konvoluce nad všemi senzory, 20 kHz

4.3.3 Shrnutí

Metoda využívající surová data zpracovaná konvoluční sítí se jeví jako relativně dobře využitelná. Ukázalo se, že modely dokážou rozpoznat v surových datech vzory, které charakterizují vadu a dokážou tyto vzory poměrně dobře generalizovat. To dokazují výsledky na vyřazených datech jak v turbínovém, tak i čerpadlovém režimu.

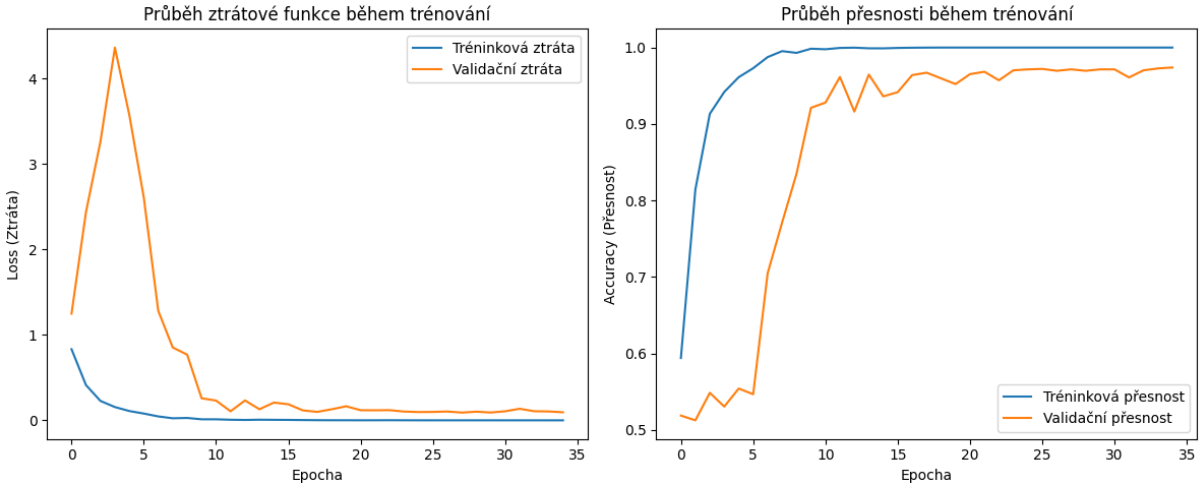
Před zhodnocením přesnosti na vyřazených datech (princip vyřazení byl diskutován výše v práci) je dobré pohlédnout na vliv na přesnost na testovací sadě dat v případě tréninku na všech souborech, polovině a třetině. Ke srovnání byly vyhodnocovány data podvzorkovaná na 20 kHz. Významný je pokles obou redukcí vzhledem k přesnosti bez redukce, méně významný rozdíl je potom mezi jednotlivými variantami redukce – 91,5 % ku 83,1 % a 82,4 %. Záměr testování na vyřazených datech byl, kromě redukce objemu dat kvůli výpočetní paměti, alespoň částečně získat povědomí o schopnostech predikce modelu při jiných podmínkách, než na jakých byl trénován, což přibližuje situaci, kdy se model pokouší predikovat jinou vadu, než na kterou byl trénován.

Při tréninku na polovině dat, na testování na neznámých souborech byla při vzorkovací frekvenci 50 kHz přesnost 87,7 % na datech bez poruchy, respektive 85,9 % na datech s poruchou. To je při přesnosti na testovací sadě dat 90,5 % velice dobrý výsledek, protože se přesnosti na vyřazených souborech blíží přesnosti na testovací sadě dat. Zároveň je vidět nárůst v přesnosti oproti vzorkovací frekvenci 20 kHz, tedy lze předpokládat, že s vyšším výpočetním výkonem při využití vyšší vzorkovací frekvence by výsledek mohl být ještě lepší.

Na neznámých stavech byl výsledek také nejlepší při nejvyšší použité frekvenci, tentokrát 100 kHz. Přesnosti na vyřazených stavech byly 70,1 % na datech bez vady a 85,8 % s vadou. Oproti testovací přesnosti 93,9 % jsou to hodnoty nižší, což je pochopitelné, neboť vzory nalezené při učení se mohou v datech z jiných stavů měření jevit pozměněné a pro model ne tak zřejmé. I tak se ovšem dá říct, že i na těchto datech dokáže model rysy vady zachytit. Zajímavé také je, že model spíše strádá při rozpoznání stavu bez poruchy a tyto soubory častěji falešně označuje za vadu než naopak.

Nutno podotknout, že výsledky by šly zlepšit také optimalizací architektury modelu samotného. To bylo v této fázi provedeno pouze ručně bez automatizace – důmyslnější optimalizací se práce zabývá při využití MEL-spektrogramů. Pro demonstraci byla snaha optimalizovat model s použitím třetiny souborů a s podvzorkováním na 100 kHz. Byly použity celkem čtyři konvoluční 1D bloky – s 16 filtry o velikosti 7, s 32 filtry o velikosti 5 a dva s 64 filtry o velikosti 3. Všechny bloky obsahují batch normalizaci a maxpooling s pool size dva. Plně propojená vrstva obsahovala tentokrát pouze 64 neuronů a byl přidán dropout bránící přeučení. Poslední vrstva obsahuje pouze jeden neuron s aktivační funkcí *sigmoid*, jinak byla všude použita aktivační funkce *ReLU*. Při této variantě se dosáhlo lehkého zvýšení přesnosti na testovací sadě. Oproti 93,9 % u původního modelu, bylo nyní dosaženo přesnosti na testovací sadě 95,9 %. Při rozpoznávání chodu bez vady se dosáhlo přesnosti 77,9 % (původně 70,1 %) (správně určeno 21 175, neurčitě 774 a chybně 5 211 vzorků). Při chodu s vadou bylo dosaženo přesnosti 90,9 % (původně 85,8 %) (správně určeno 27 489, neurčitě 351 a chybně 2 403 vzorků).

Ukázalo se tedy, že nárust na neznámých stavech byl s vyšší komplexitou modelu vyšší než nárůst na testovací sadě dat.



Obrázek 4-26: Průběh učení s optimalizovanou architekturou modelu

Obecně se ukazuje, že při zvýšení vzorkovací frekvence je možné dosáhnout lepších výsledků přesnosti.

Je potřeba také zmínit variantu, kdy byl použit výhradně mikrofon. Samotným mikrofonem bylo dosaženo velmi dobrých výsledků. Ukázalo se, že v případě mikrofonu se výsledky s podvzorkováním na 20 kHz a bez podvzorkování příliš neliší. To je dáno frekvenčním rozsahem mikrofonu, který je 10 Hz až 20 kHz, tedy zhruba rozsah slyšitelného zvuku. Výsledky při tréninku na třetině dat jsou dokonce vyšší než při použití všech pěti kanálů. Je ovšem nutné zdůraznit, že nejvyšší frekvence, která byla použita se všemi pěti kanály, byla z důvodu výpočetní kapacity 100 kHz. Jiné kanály mají na rozdíl od mikrofonu rostoucí tendence s rostoucí vzorkovací frekvencí (kromě snímače zrychlení s hranicí 60 kHz). Není tak možné vyloučit, že by při frekvenci 200 kHz dosáhla kombinace všech kanálů vyšší hodnoty. Nicméně faktem zůstává, že použití samotného mikrofonu může být v použití s konvolucí na surových datech funkční a efektivní metoda. Ovšem měření mikrofonem se sebou nese v praxi určitá rizika například v možnosti narušení měření. Oproti tomu se například snímače tlaku jeví jako naopak velmi spolehlivé. Ani využití samotného senzoru akustické emise se nejeví jako nereálné. Výsledky jsou o něco horší než u mikrofonu, ale měření akustické emise je spolehlivější z hlediska narušení měření než mikrofon. Výsledky byly pro přehlednost shrnuty v následující tabulce.

Tabulka 3: Shrnutí hlavních výsledků konvolučních sítí se surovými daty pro turbínový režim

Turbínový režim – Konvoluce surových dat					
Využitá data ke tvorbě modelu:	Přesnost na testovací sadě při učení:	Testovací přesnost při učení s vyřazenými daty bez poruchy:	Testovací přesnost při učení s vyřazenými daty s poruchou:	Přesnost na vyřazených datech bez vady:	Přesnost na vyloučených datech s vadou:
Veškerá dostupná data – Rozdělení datových oken:	99,8 %	-	-	-	-
Veškerá dostupná data – Rozdělení souborů:	99,6 %	-	-	-	-
Data s vyřazenými 10 minutami záznamu:	-	97,0 %	92,6 %	95,9 %	72,4 %

Tabulka 4: Přehled přesnosti významných modelů pro Čerpadlový režim s použitím konvolučních sítí a surových dat

Čerpadlový režim – Konvoluce surových dat				
Využitá data ke tvorbě modelu:	Vzorkování:	Přesnost na testovací sadě:	Přesnost na vyřazených datech bez vady:	Přesnost na vyloučených datech s vadou:
Veškerá dostupná data – Rozdělení datových oken:	20 kHz	91,5 %	-	-
Veškerá dostupná data – Rozdělení souborů:	20 kHz	89,8 %	-	-
Polovina souborů, použit každý druhý:	20 kHz	83,1 %	76,9 %	70,9 %
	50 kHz	90,5 %	87,7 %	85,9 %
Třetina souborů, soubory voleny tak, aby byl pokryt celý chod	20 kHz	82,4 %	64,9 %	77,5 %
	100 kHz	93,9 %	70,1 %	85,8 %
Šestina souborů, volen každý druhý soubor z předchozí třetiny	200 kHz	91,6 %	65,4 %	88,3 %
Veškeré soubory - Pouze mikrofon	20 kHz	93,9 %	-	-
Polovina souborů - Pouze mikrofon	50 kHz	89,6 %	90,2 %	81,7 %
Třetina souborů, soubory voleny tak, aby byl pokryt celý chod - Pouze akustická emise	200 kHz	87,6 %	75,3 %	83,5 %
Třetina souborů, soubory voleny tak, aby byl pokryt celý chod - Pouze mikrofon	200 kHz	94,1 %	80,8 %	92,0 %

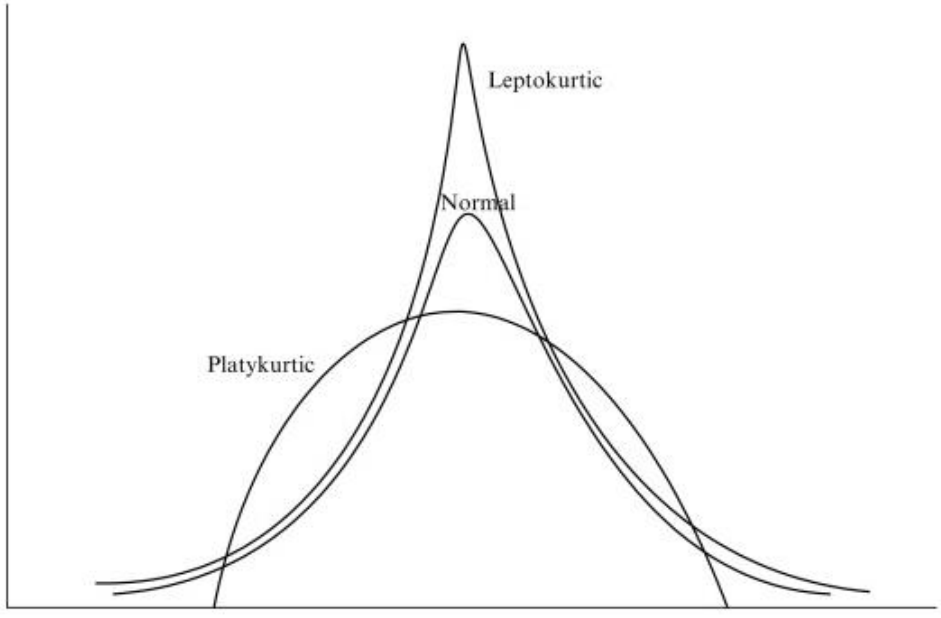
4.4 Využití statistických metod při preprocessingu dat

Druhou cestou, která se jevila jako využitelná, bylo zpracovat data statistickými metodami ještě před vstupem do neuronové sítě, která díky tomu může být jednodušší a pouze plně propojená, namísto konvoluční. Myšlenka této varianty je pojmenovat sekvenci dat pomocí vhodné statistické charakteristiky. Tím, že několika veličinami bude popsána celá sekvence, budou tyto veličiny závislé i na tom, zda se během této doby projevila lopatka s poruchou, či nikoliv. Konkrétní statistická charakteristika musí být volena vhodně, protože je pravděpodobné, že ne každá popisuje sekvenci dat tak, že se ve výsledku vada projeví. Nejnadějněji se jevila charakteristika kurtosis, která byla nadále kombinována ještě s dalšími – šikmost a percentil, respektive rozdíl percentilů. Vstupem do neuronové sítě je tak především popis rozložení dat.

- Kurtosis, tedy špičatost, je statistická míra charakterizující rozložení dat v jednotlivých oblastech rozdělení, tedy v oblasti špičky a ocasních částech. Vypovídá o tom, zda jsou data soustředěna kolem průměru a jak často se vyskytují extrémní hodnoty. To je důležité pro tuto aplikaci, protože extrémní hodnoty mohou být spojeny s anomálií, jakou může být porucha [40]. Pro výpočet špičatosti byla v Pythonu využita knihovna SciPy. Byl počítán Fisherův kurtosis, který definuje normální rozdělení výsledkem 0, nikoliv 3. Výsledný vzorec je tedy [41]:

(4-2)
$$K = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2\right)^2} - 3$$

Kde n je počet hodnot ve vyhodnocovaném segmentu dat, x_i je konkrétní hodnota v segmentu a $\bar{x}$ je průměrná hodnota segmentu.
Lze říct, že hodnotou kurtosis lze data zařadit do typu rozdělení podle obrázku:



Obrázek 4-27: Druhy rozdělení podle hodnoty kurtosis [42]

Data, která mají Fisherův kurtosis vyšší než 0, jsou data leptokurtická, pokud jsou přesně rovna 0, pak odpovídají normálnímu rozložení a pokud je kurtosis menší, jedná se o rozdělení platykurtické [42].
Kurtosis se jevil velice nadějně, protože ve výsledcích ve zprávě [29] vykazoval značně rozdílné hodnoty za provozu s poruchou a bez poruchy.

- Šikmost, tedy skewness, je podobně jako kurtosis statistická míra, která popisuje rozdělení dat. Šikmost vypovídá o tom, zda je rozdělení symetrické kolem průměru, případně na jakou stranu od průměru je situováno více dat [43].
  Šikmost byla počítána opět pomocí knihovny SciPy a lze spočítat následovně [44]:

(4-3)
$$Sk = \frac{m_3}{m_2^{3/2}}$$

$$m_i = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^i$$

Kde m_i jsou centrální momenty a x_n jsou jednotlivé prvky v sadě. Druhý centrální moment m_2 je rozptyl a třetí centrální moment m_3 zachycuje asymetrii rozdělení. Šikmost může být kladná (pravostranná) či záporná (levostranná). V případě pravostranné šikmosti je většina prvků soustředěna na levé straně osy, extrémní hodnoty se nachází na straně pravé. To je situace, kdy je většina hodnot nižších a jen několik hodnot je výrazně vyšších. Jedná se o situaci, kdy průměr může být vyšší než medián. Tato situace je znázorněna na následujícím obrázku. [43].

Obrázek 4-28: Ukázka rozložení při kladné šikmosti, upraveno dle [43]

Šikmost, stejně jako špičatost, nezávisí přímo na absolutních hodnotách dat. To je výhodné pro účely této metody vzhledem k použitým senzorům.

- Percentil je statistická hodnota, která udává, kde se daná hodnota nachází v konkrétním datovém vzorku. Například percentil(95) je hodnota, která je vyšší než 95 % vzorků v sadě a nižší než zbývajících 5 % [45].
  K rozdílů percentilů bylo přistoupeno právě z důvodu, že hodnota percentilu je závislá na měření absolutních hodnot. Vzdálená inspirace proběhla také metodou peak to peak používanou ve vibrodiagnostice.

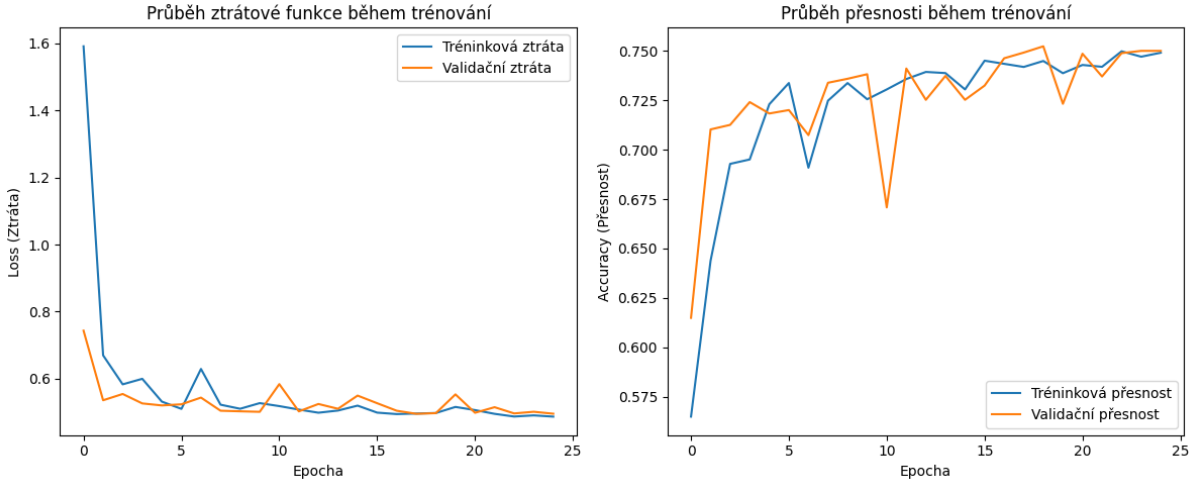
Nesporná výhoda této varianty je její nižší prostorová náročnost, neboť na rozdíl od varianty s konvoluční sítí je každý segment, datové okno, zredukován do několika jednotek hodnot. To je výhodné především při učení neuronové sítě, protože byla větší volnost při nastavování preprocessingu dat, například velikosti segmentu a překryvu těchto segmentů.

Jak bylo řečeno, byla použita ne příliš složitá, plně propojená neuronová síť. Počet neuronů na vstupu odpovídal násobku počtu kanálů (v tomto případě výhradně 5) a počtu využitých statistických charakteristik. Následují čtyři plně propojené vrstvy s 64, 128, 256 a 128 neurony, všechny jsou řešeny s *ReLU* aktivační funkcí, která se v tomto případě projevila jako vhodná. Výstupní vrstva má přirozeně jeden neuron, a protože výstup je v podobě, dejme tomu, pravděpodobnosti poruchy, využívá aktivační funkci *sigmoid*. Pokud není uvedeno jinak byl použit learning rate 0,001 a batch size 256.

K této části práce se vztahují kódy v příloze 7 ve složce „Statistika“. Kódy jsou pojmenovány intuitivně, aby šla znát jejich funkce, ta je také shrnuta v úvodním komentáři přímo v kódu.

4.4.1 Data z turbínového režimu

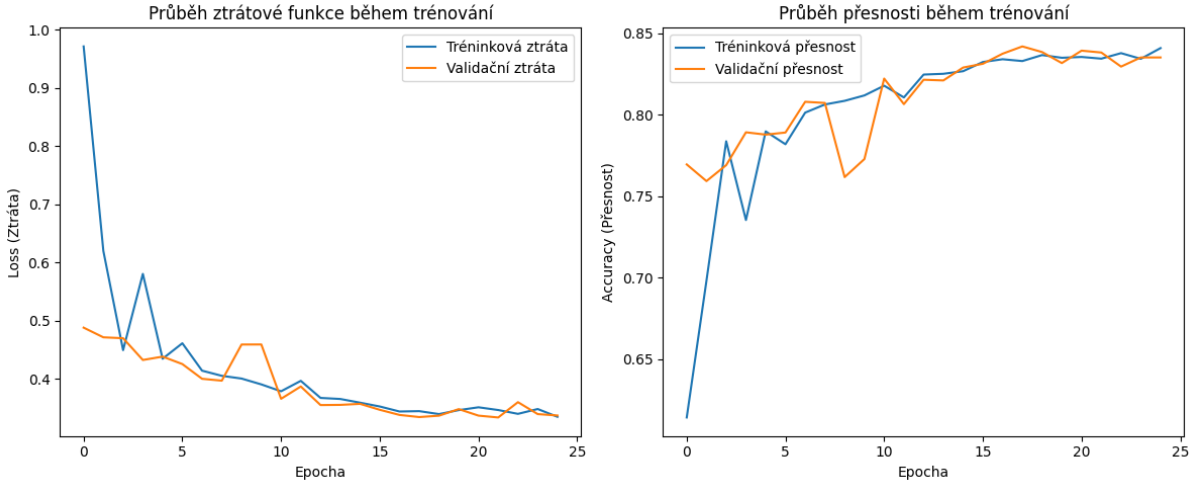
Jako první byl využit pouze kurtosis, který se na základě zprávy [29] jevil jako nejvhodnější. Nejprve byl pro porovnání s předchozí variantou se surovými daty spočítán případ, kde se kurtosis počítal ze segmentů bez překryvu a s délkou 10 000. V tomto případě neuronová síť dosáhla přesnosti na testovacích datech 75,0 %. Což je nižší než při použití konvoluce na surová data.



Obrázek 4-29: Průběh ztrátové funkce a přesnosti při velikosti segmentu 10 000 a posuvu 10 000; pouze kurtosis, 200 kHz, turbínová data

Díky nižší prostorové náročnosti bylo možné spočítat také téměř ideálně požadovanou velikost segmentu 50 000. Taková hodnota téměř zaručuje, že v segmentu bude zahrnuta celá otáčka oběžného kola. Nevýhodou je, že redukcí tak velkých segmentů do jedné statistické hodnoty dostaneme jen velmi málo vzorků k učení. Proto byl nutný značný překryv těchto segmentů. S takovým překryvem je potřeba mít na paměti, že jednotlivé segmenty si mohou být podobné s několika dalšíma. V tomto případě je tedy obzvlášť žádoucí porovnávat přesnost

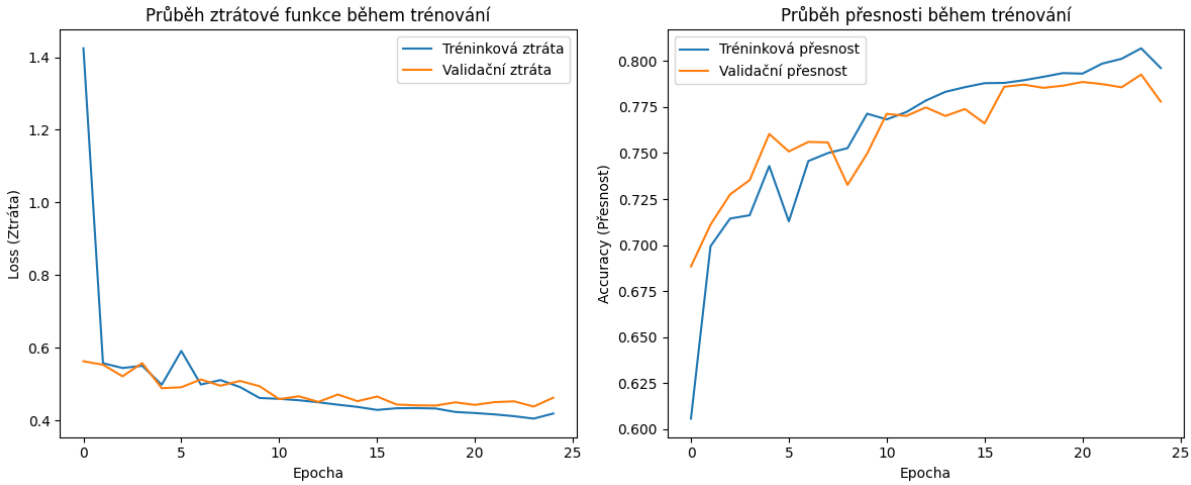
při rozdělení do trénovací/testovací/validační sady na úrovni těchto segmentů a souborů nebo s přesností na neznámých datech. Posun byl zvolený na 5 000 záznamů. V tomto případě se za použití pouze kurtosis model naučil na testovací přesnost 83,7 % při rozdělení na úrovni segmentů.



Obrázek 4-30: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 5 000; pouze kurtosis, 200 kHz, turbínová data

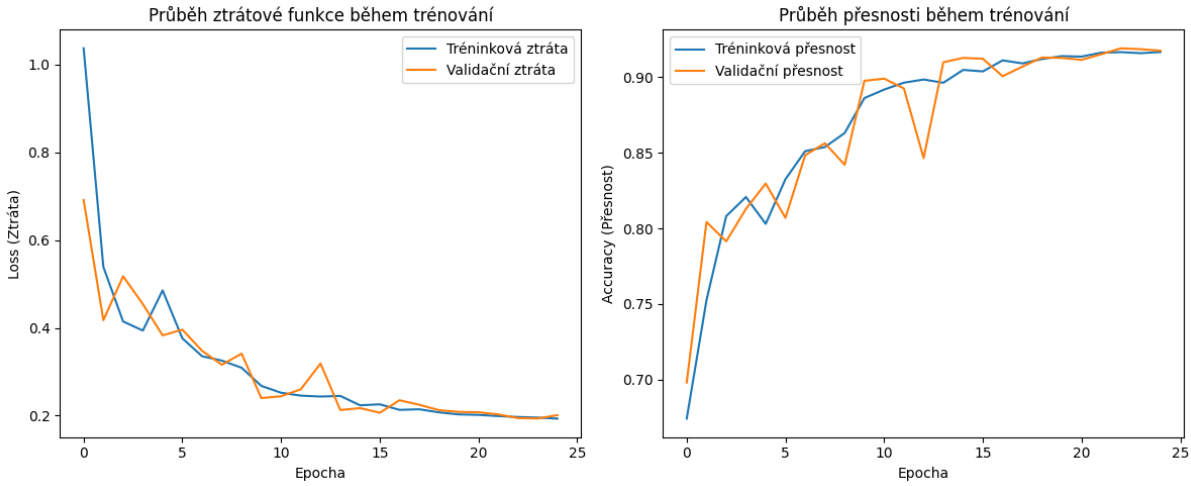
Výkonnost neuronové sítě lze zvýšit, pokud bude kurtosis doplněn dalšími statistickými charakteristikami. Byly využity další charakteristiky – koeficient křivosti a několikrát percentil. Bylo zkoumáno, jaké argumenty použít pro funkci percentil. Na základě vyhodnocení několika variant byla zvolena kombinace percentil(90) a percentil(50). Byla také vyzkoušena varianta, kdy se přidával ještě třetí percentil. Tento krok ovšem neměl žádný pozitivní vliv na výsledek.

Pro tuto variantu byl pro porovnání opět vyhodnocen případ, kdy byly statistické metody vyhodnocovány ze segmentu o velikosti 10 000 a bez překryvu. V tomto případě byla testovací přesnost 79,1 %. Přesnost je tedy vyšší než u použití samotného kurtosis, ale stále nižší než u použití konvoluční sítě.



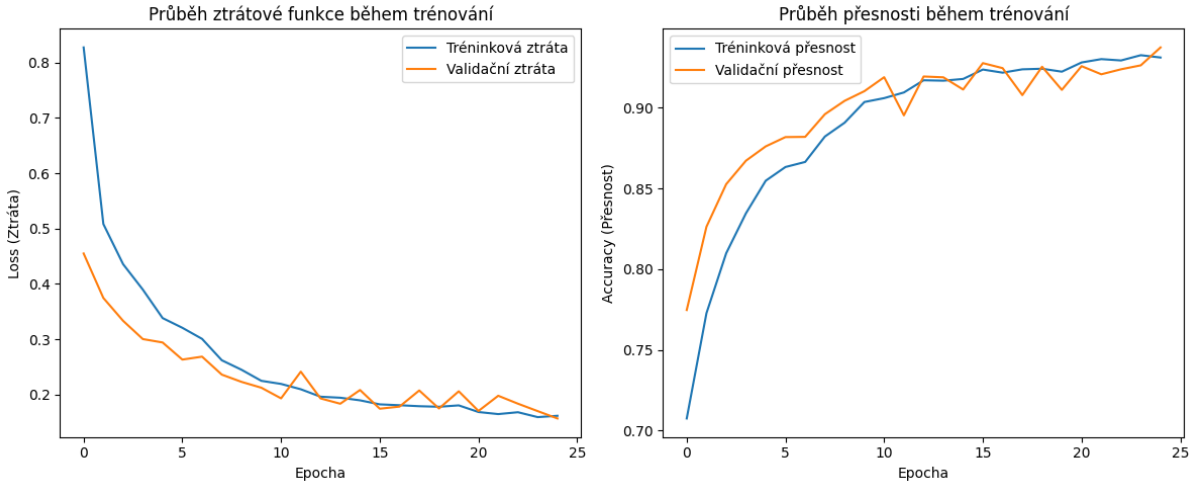
Obrázek 4-31: Průběh ztrátové funkce a přesnosti při velikosti segmentu 10 000 a posuvu 10 000; kurtosis, šikmost, percentil(90) a percentil(50), 200 kHz, turbínová data

Při velikosti segmentu 50 000 a posunu segmentu o 5 000 byla přesnost modelu na testovací sadě 91,5 %.



Obrázek 4-32: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 5 000; kurtosis, šikmost, percentil(90) a percentil(50), 200 kHz, turbínová data

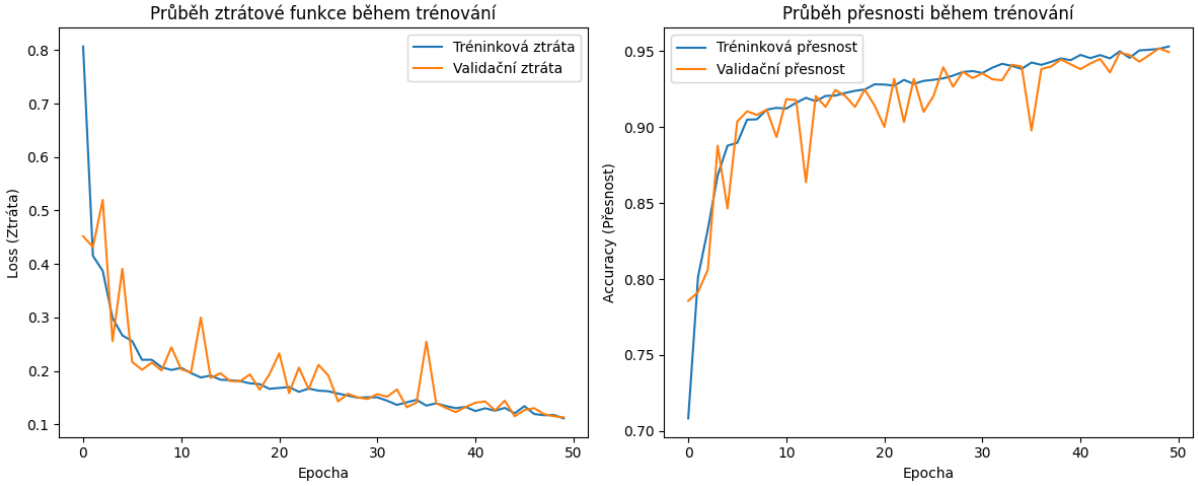
Proběhlo ještě další zjemnění kroku posunu segmentu, a tedy zvýšení překryvu. Velikost segmentu byla stále 50 000 a posun 2 500. V tomto případě se přesnost sítě ještě zlepšila na 93,4 %. K dalšímu zjemňování už se nepřistupovalo.



Obrázek 4-33: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost, percentil(90) a percentil(50), 200 kHz, turbínová data

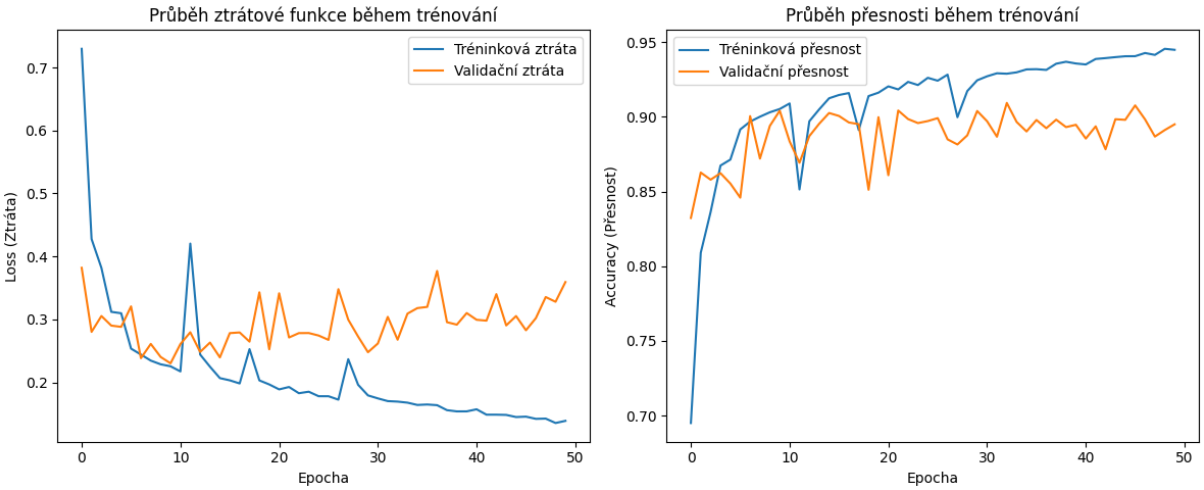
Protože u této metody nebyly tolik limitující nároky na paměť počítače, byla spočítána i varianta s velikostí segmentu 60 000 a velikostí posuvu 3 000. Jedná se o plně korektní variantu, kdy je stoprocentně v segmentu zachycena alespoň jedna otáčka. Nicméně se ukazuje, že rozdíl mezi délkou segmentu 60 000 a 50 000 není signifikantní. S touto délkou bylo dosáhnuto přesnosti na testovací sadě 93,2 %, tedy defacto stejné hodnoty jako při délce segmentu 50 000.

Jak bylo popsáno a zdůvodněno v úvodu popisu této metody, byla testována také varianta s kurtosis, šikmostí a rozdílu percentil(95) a percentil(5), přesnost na testovací sadě pak byla 95,5 %.



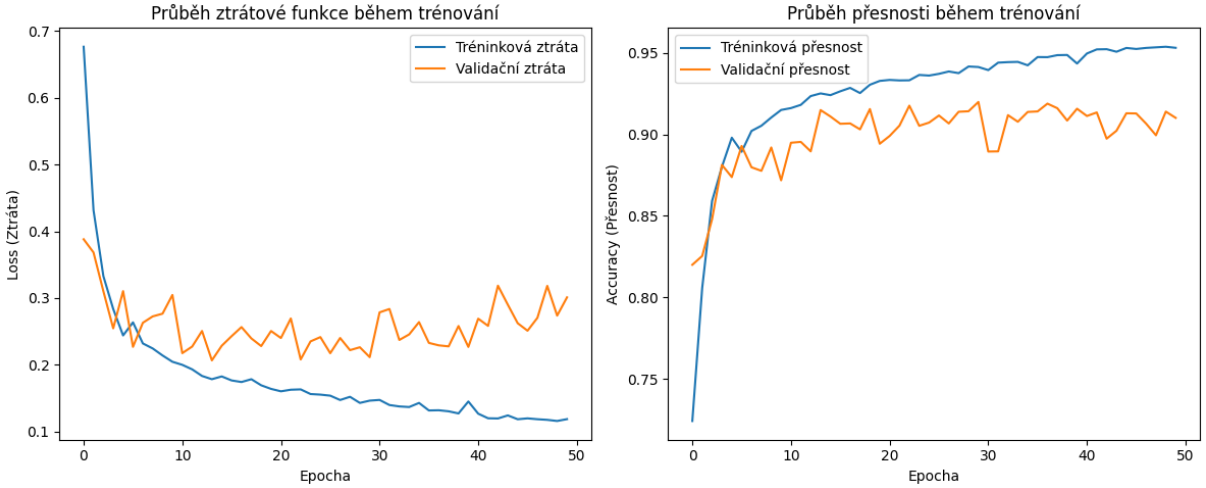
Obrázek 4-34: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; šikmost, kurtosis a percentil(95)-percentil(5), 200 kHz, turbínová data

Podobně jako při použití konvoluční sítě se surovými daty, byla i u statistických metod testována varianta, kde byly do trénovací/validační/testovací sady dat rozřazovány přímo soubory, nikoliv už segmenty, respektive vypočtené statistické veličiny. V následujících případech byla použita délka segmentu 50 000 a překryv 2 500. Přesnost na testovací sadě dat pro variantu s kurtosis s šikmostí a dvěma percentily byla 87,2 %. Pokles je zde o něco vyšší než u konvolučního modelu, což zřejmě souvisí s tím, že na rozdíl od konvolučních oken jsou statistické veličiny počítány nad segmenty, které se vzájemně překrývají. Jednotlivé segmenty, které mohou být v jiných sadách (trénovacích, testovacích či validačních), si mohou být výrazně podobné. Přesnost při rozdělení na úrovni segmentů je pak znatelně vyšší než při rozdělení na úrovni souborů. Při dělení na úrovni souborů se pak také projevovalo přeučení, jak je znát na průběhu učení.



Obrázek 4-35: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost, percentil(90) a percentil(50), 200 kHz, turbínová data; Rozdělení na úrovni souborů

Takto byla testována i varianta s kurtosis, skewness a rozdílu percentil(95) a percentil(5), přesnost při rozdělení dat přímo na souborech byla 89,2 %. Jedná se tedy o obdobný efekt.



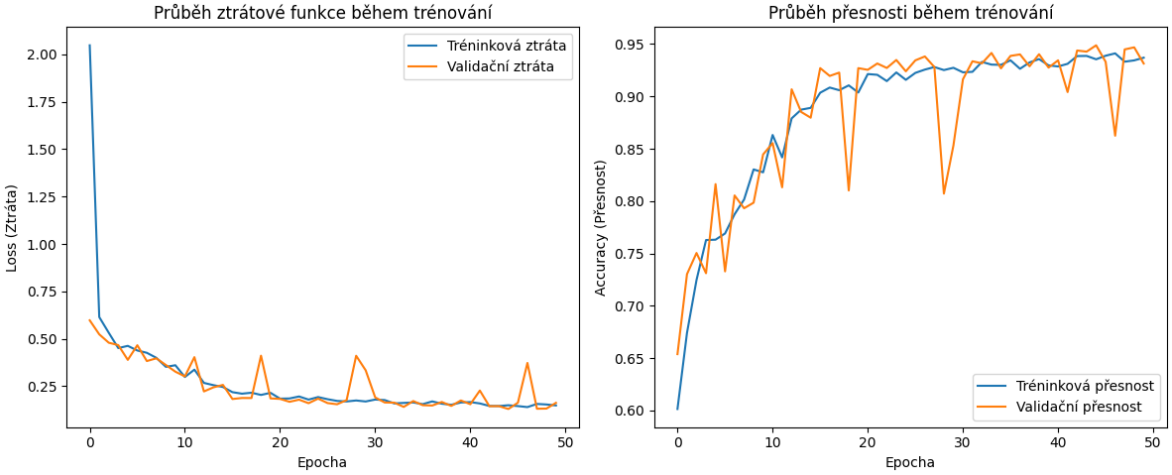
Obrázek 4-36: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost a percentil(95)- percentil(5), 200 kHz, turbínová data; Rozdělení na úrovni souborů

4.4.2 Data z čerpadlového režimu

Varianta modelu, která se nejlépe osvědčila na datech z turbínového režimu, byla aplikována na data z režimu čerpadlového. Tedy varianta s délkou segmentu 50 000 a posuvem 2500 za využití kurtosis, šikmosti a dvou percentilů nebo rozdílu percentilu(95) a percentilu(5). Zde byl model opět trénován pouze na části dat, aby byl zredukován jejich počet. Soubory byly voleny stejně jako v případě u konvoluční neuronové sítě se surovými daty. Protože vyřazených dat bylo mnoho a vyhodnocení trvalo dlouho, nebyla k testování na vyřazených datech vždy použita všechna vyřazená data, ale pouze různé části tak, aby zůstalo zachováno jejich rozložení.

Nejprve bylo použito podvzorkování na 100 kHz při využití třetiny dat k tréninku. Během trénování byl model využívající kurtosis, šikmost a dva percentily natrénován na přesnost na testovací sadě 93,9 %. Na vyřazených stavech byl následně v případě dat bez poruchy určen správný stav v 17 796 případech, neurčitý výsledek byl v 1 320 případech a v 8 065 případech byl výsledek určen chybně. Přesnost na neznámých stavech je tedy v případě rozeznání chodu bez poruchy 65,5 %.

Při rozeznávání dat s vadou byl správný výsledek určen v 23 988 případech, neurčitý výsledek byl zaznamenán v 1 575 případech a 4 658 vzorků bylo označeno nesprávně. Přesnost při rozeznání poruchy je tedy 79,4 %.

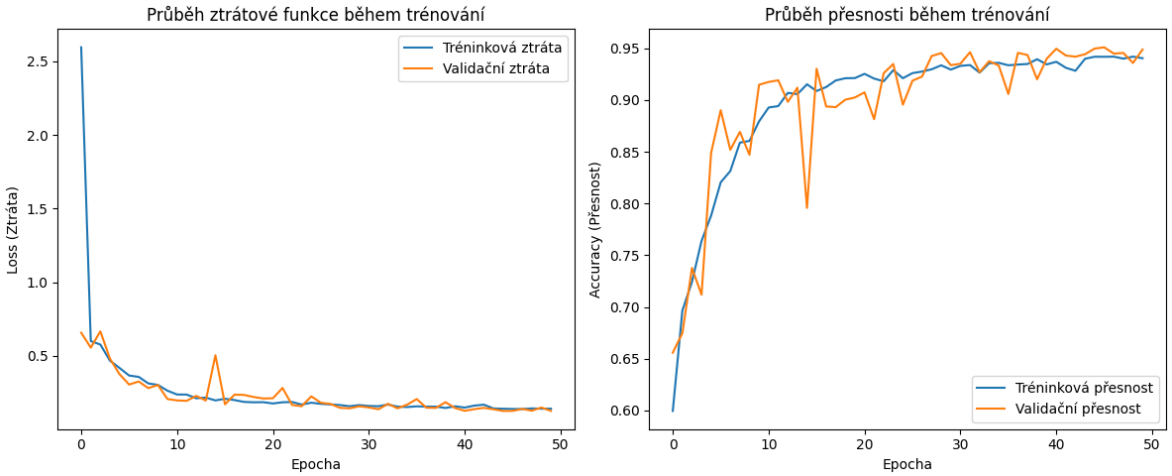


Obrázek 4-37: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost, percentil(90) a percentil(50); Data z čerpadlového režimu; 100 kHz; třetina souborů

Dále byl vytrénován model využívající kurtosis, šikmost a dva percentily na polovině naměřených souborů se vzorkovací frekvencí 50 kHz, kdy byl použit opět každý druhý soubor. Během učení byla dosažena přesnost na testovací sadě 90,0 %. Při testování na vyřazených souborech obsahujících vadu bylo správně označeno 32 970 vzorků, 3 034 neurčitě a 4 297 bylo označeno chybně. Přesnost je tedy 81,8 %. Při rozeznávání chodu bez poruchy byl model správný v 31 234 případech, k neurčitému výsledku došel v 2 153 případech a v 3 794 případech bylo označení chybné. Přesnost je tedy 84,0 %.

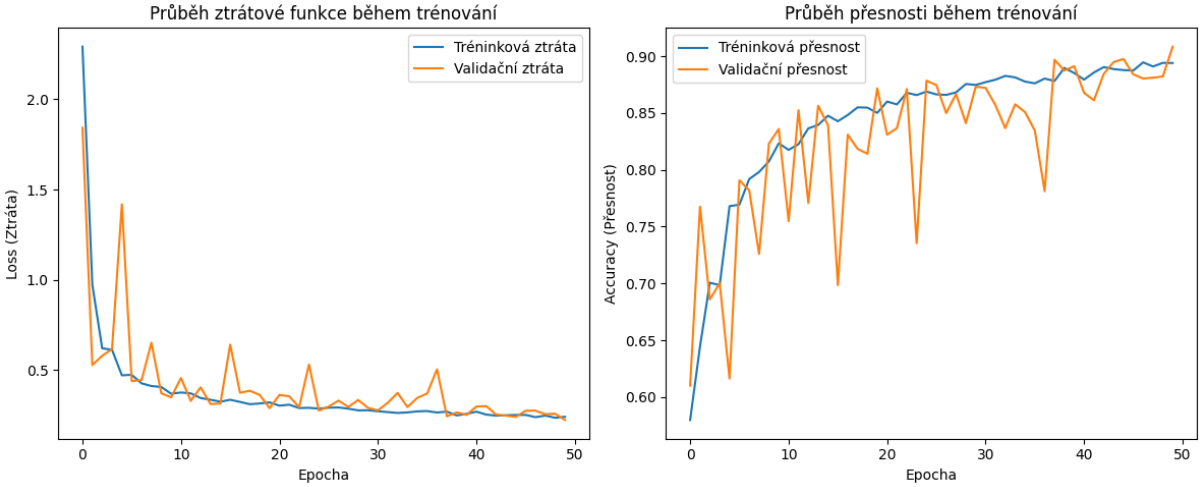
Totéž bylo provedeno s variantou, kdy byla použita šikmost, špičatost a rozdíl percentilů 95 a 5. Při tréninku na třetině dat s frekvencí 100 kHz byla dosažena přesnost na trénovací sadě 94,7 %. Během testování dat bez vady bylo správně určeno 9 591 segmentů, neurčitě 757 a chybně bylo za vadu označeno 5726 vzorků. Přesnost byla tedy 59,7 %.

Při predikci dat obsahující vadu bylo správně určeno 15 134 segmentů, neurčitě bylo označeno 720 segmentů a chybně 1912. Přesnost byla tedy 85,2 %.



Obrázek 4-38: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost a percentil(95)- percentil(5); Data z čerpadlového režimu; 100 kHz; třetina souborů

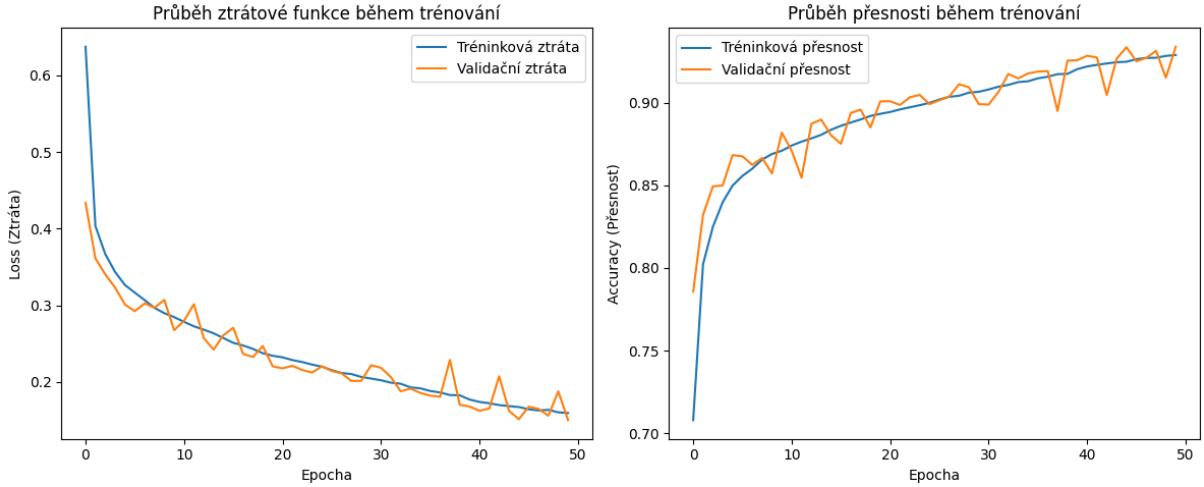
Při využití poloviny souborů a vzorkovací frekvenci 50 kHz při učení byla přesnost na testovací sadě 90,7 %. Při určování dat bez poruchy byl model přesný v 3 489 případech, v 526 byl výsledek neurčitý a chybný byl v 743 případech. Přesnost při rozpoznání chodu bez poruchy je tedy 73,3 %.
Při rozpoznávání chodu s poruchou bylo správně určeno 4 576 segmentů, neurčitě 230 a chybně 318 segmentů. Přesnost je tedy 89,3 %.



Obrázek 4-39: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; kurtosis, šikmost a percentil(95)- percentil(5); Data z čerpadlového režimu; 50 kHz; polovina souborů

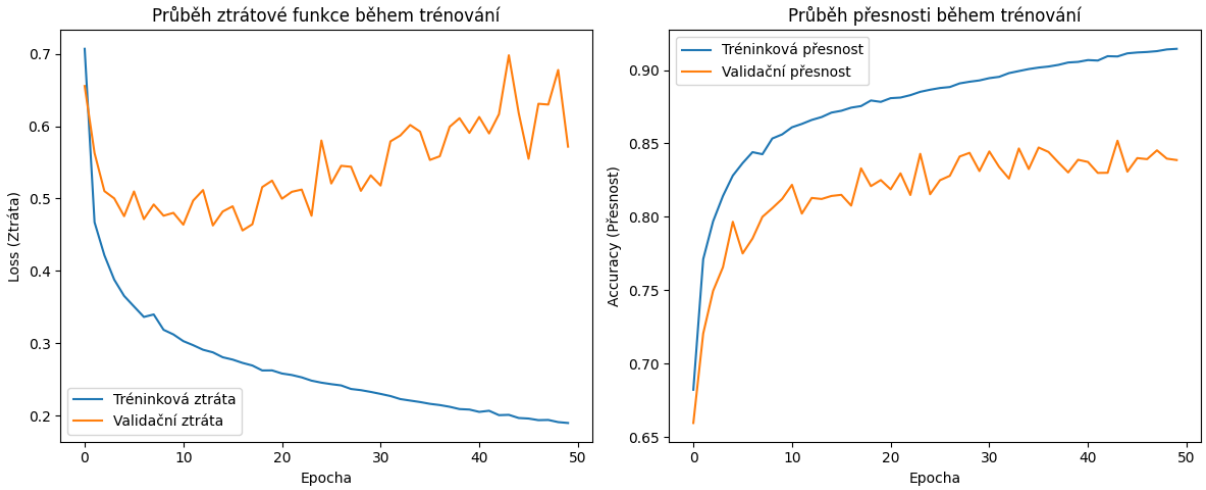
Protože na rozdíl od konvoluční metody není nutné mít načtena všechna data zároveň, byla naprogramována varianta kódu, která soubory načítala postupně. V paměti byl tedy vždy načtený pouze jeden soubor, ze kterého byly uloženy statistické charakteristiky, což umožnilo načítání bez podvzorkování.

Přesnost na testovací sadě při použití veškerých dat bez podvzorkování při aplikaci kurtosis, šikmosti a percentilu(90) a percentilu(50) byla 93,3 % při rozdělení do sad na úrovni segmentů.



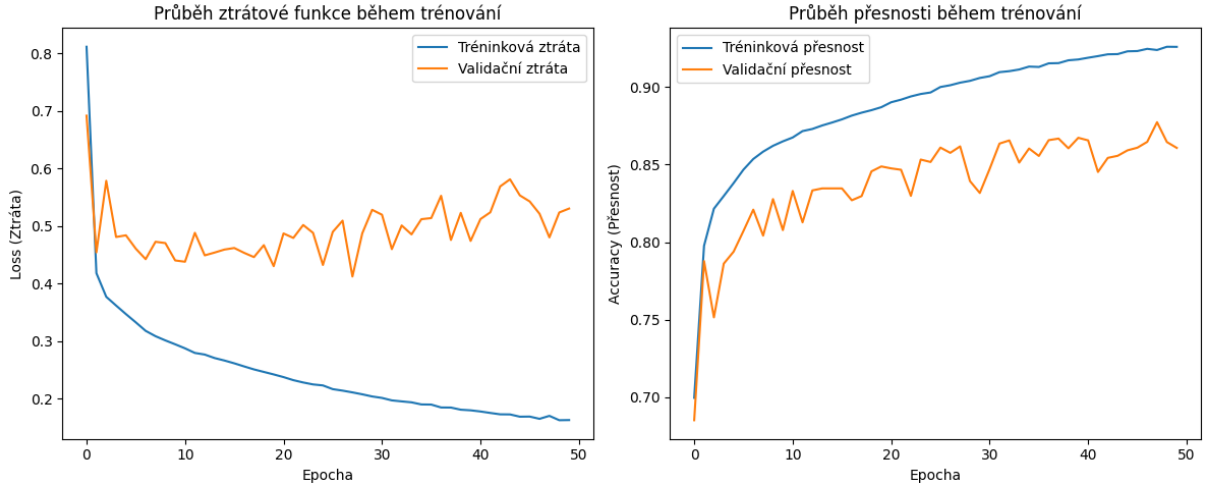
Obrázek 4-40: Průběh ztrátové funkce a přesnosti při 200 kHz, kurtosis, šikmost, percentil(90) a percentil(50), čerpadlová dat

Při vytvoření sad přímo ze souborů byla přesnost na testovací sadě 84,7 %. Opět bylo znát přeučení.



Obrázek 4-41: Průběh ztrátové funkce a přesnosti při 200 kHz, kurtosis, šikmost, percentil(90) a percentil(50), dělení na úrovni souborů, čerpadlová data

Při 200 kHz byla testována i varianta s kurtosis, skewness a rozdílem percentil(95) a percentil(5), která při rozdělení dat do sad již na úrovni souborů dosáhla přesnosti na testovací sadě 86,1 %.



Obrázek 4-42: Průběh ztrátové funkce a přesnosti při velikosti segmentu 50 000 a posuvu 2 500; Skewness, Kurtosis a percentil(95)-percentil(5); Rozdělení přímo souborů; 200 kHz

4.4.3 Shrnutí

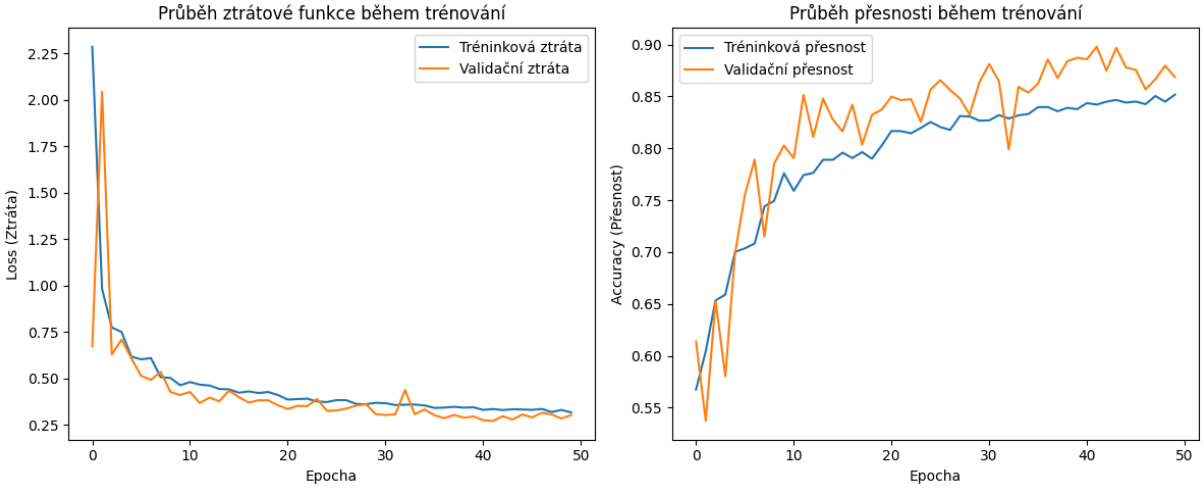
V případě využití výpočtu statistických charakteristik v rámci přípravy dat byly využívány veličiny kurtóza, šikmost a percentil. Nejlépe se osvědčilo využití percentilu 90 a percentilu 50. Nevýhodou využití percentilu jako takového je fakt, že na rozdíl od kurtózy a šikmosti závisí na absolutních hodnotách. To není zcela ideální vzhledem k použitým snímačům. Proto bylo dále využito rozdílu percentilu 95 a percentilu 5. Tento způsob nevýhodu eliminuje.

Využití statistických metod v kombinaci s plně propojenou neuronovou sítí se ukázalo jako možné, avšak metoda by pro praktickou aplikaci potřebovala další výzkum. Výsledkem

zkoumání této metody je závěr, že je skutečně možné extrahovat ze segmentu dat informaci o poruše za použití statistických charakteristik, metoda v této fázi provedení ovšem není schopná dostatečně generalizovat problém tak, aby byla efektivní i v rozpoznání dat ze stavů, které nebyly použity při učení. Problém se ukázal především ve špatném rozeznávání chodu bez poruchy. Na neznámých stavech se zde přesnost pohybovala okolo 60 %.

Na rozdíl od zpracování surových dat konvoluční sítí v předchozí metodě se u této varianty lišila přesnost při rozdělení do sad na úrovni segmentů a na úrovni souborů. Jak již bylo řečeno, je to zřejmě následek použití významného překryvu jednotlivých segmentů. Při dělení do sad na úrovni souborů se zároveň objevovalo výrazné přeučení modelu. Což poukazuje na problém s generalizací. Problém s generalizací se následně potvrdil při testování na neznámých stavech. Výsledky na neznámých souborech ze známých stavů ovšem dosáhly relativně obstojných výsledků. To lze interpretovat jako potvrzení závěru, že metoda v tomto nastavení a provedení dokáže najít jisté vzory, ale nedokáže je příliš dobře zobecnit.

Pro zajímavost byl na třetině dat s frekvencí 200 kHz vytrénován ještě model, který pracoval se segmentem 10 000 a bez překryvu segmentů. Rozdělení do testovací/trénovací/validační sady proběhlo již na úrovni souborů. Tím byl eliminován nedostatek s překrývajícími se datovými segmenty. Zabránilo se přeučení a přesnost při rozeznání dat s vadou byla poměrně uspokojivá – 72,1 % (6574 určena vada, 720 neurčitý výsledek, 1826 chybně predikován chod bez vady). Nicméně naopak při analýze dat bez vady byla přesnost pouze 56,3 % (5676 určen chod bez vady, 1286 neurčitý výsledek, 3118 chybně predikován chod s vadou). Výsledky jsou opět shrnuty do tabulek.



Obrázek 4-43: Průběh ztrátové funkce a přesnosti při velikosti segmentu 10 000 a posuvu 10 000; Skewness, Kurtosis a percentil(95)-percentil(5); Rozdělení přímo souborů; 200 kHz

Tabulka 5: Shrnutí hlavních výsledků s využitím statistických metod pro turbínový režim

Turbínový režim – Statistické metody	
Využitá data:	Přesnost na testovací sadě:
Veškerá data - Kurtosis, skewness a 2× percentil	93,2 %
Veškerá data, rozdělení přímo souborů - Kurtosis, skewness a 2× percentil	87,2 %
Veškerá data - Kurtosis, skewness a rozdíl percentilů	95,5 %
Veškerá data, rozdělení přímo souborů - Kurtosis, skewness a rozdíl percentilů	89,2 %

Tabulka 6: Shrnutí hlavních výsledků s využitím statistických metod pro čerpadlový režim

Čerpadlový režim – Statistické metody				
Využitá data:	Vzorkování:	Přesnost na testovací sadě:	Přesnost na vyřazených datech bez vady:	Přesnost na vyloučených datech s vadou:
Třetina souborů, Kurtosis, skewness a dva percentily	100 kHz	93,9 %	65,5 %	79,4 %
Třetina souborů Kurtosis, skewness a rozdíl percentilů	100 kHz	94,7 %	59,7 %	85,2 %
Polovina souborů, Kurtosis, skewness a dva percentily	50 kHz	90,0 %	84,0 %	81,8 %
Polovina souborů Kurtosis, skewness a rozdíl percentilů	50 kHz	90,7 %	73,3 %	89,3 %
Veškeré soubory, Kurtosis, skewness a dva percentily	200 kHz	93,3 %	-	-
Veškeré soubory - Kurtosis, skewness a dva percentily - Dělení po souborech	200 kHz	84,7 %	-	-
Veškeré soubory - Kurtosis, skewness a rozdíl percentilů - Dělení po souborech	200 kHz	86,1 %	-	-

4.5 Využití MEL-spektrogramů

4.5.1 Tvorba spektrogramů

Další možností zpracování dat před vstupem do neuronové sítě je použití spektrogramu, respektive MEL-spektrogramu. Spektrogram jako takový je vhodný pro grafickou reprezentaci signálu. Spektrogram je vytvářen následujícím algoritmem:

- Nejprve je časový signál digitalizován podle určené vzorkovací frekvence. Při volbě vzorkovací frekvence je potřeba pamatovat na Nyquist-Shannonův teorém, který hovoří o tom, že pokud chceme správně rekonstruovat signál o frekvenci f Hz, musí být rekonstruován z bodů, které byly měřeny alespoň s frekvencí 2f Hz [46]. V případě této práce, kdy byly hodnoty zaznamenávány s frekvencí 200 kHz, bude maximální frekvence ve spektrogramu, respektive MEL-spektrogramu, 100 kHz.
- Další krok tvorby spektrogramu spočívá v rozdělení časového signálu do krátkých časových úseků. Tyto časové úseky se obvykle překrývají, často například o 50 %. Nad těmito krátkými úseky časového signálu je dále prováděna Fourierova transformace [47].
- Před aplikováním Fourierovy transformace je vhodné aplikovat na úseky časového signálu takzvané okenovací funkce. Pronásobení signálu s těmito funkcemi v rozdělených oknech zmírňuje problematiku takzvaného spektrálního úniku. Jedná se o problém, kdy se energie frekvence rozprostře mezi sousední frekvence namísto toho, aby byla soustředěna v jedné konkrétní. Tento problém je důsledkem toho, že Fourierova transformace předpokládá signál nekonečný [48]. Existuje mnoho různých funkcí – obdélníkové okno, Blackmanovo okno nebo Hanningovo okno, což je funkce, která je přednastavena při využívání knihovny Librosa [48] [49].
  Hanningovo okno neboli Hann funkce je dána následujícím předpisem [50]:

$$w(n) = 0{,}5 \cdot \left(1 - cos\left(\frac{2\pi n}{N-1}\right)\right), 0 \leq n \leq N-1 \tag{4-4}$$

  Kde: *w(n)* je hodnota okna v konkrétním časovém vzorku *n*, *n* je tedy index vzorku v rámci časového okna a *N* je jejich celkový počet.

- V dalším kroku je již konečně provede Fourierova transformace. Fourierova transformace je transformace, pomocí které je možné převést časový signál na signál frekvenční. Je dána následujícím předpisem [47]:

$$x(f) = \int_{-inf}^{inf} x(t) e^{-jft} dt \tag{4-5}$$

Pro případ diskrétního signálu, jako je zpracováván v případě tvorby spektrogramu, je možné napsat definici diskrétní Fourierovy transformace následovně [51]:

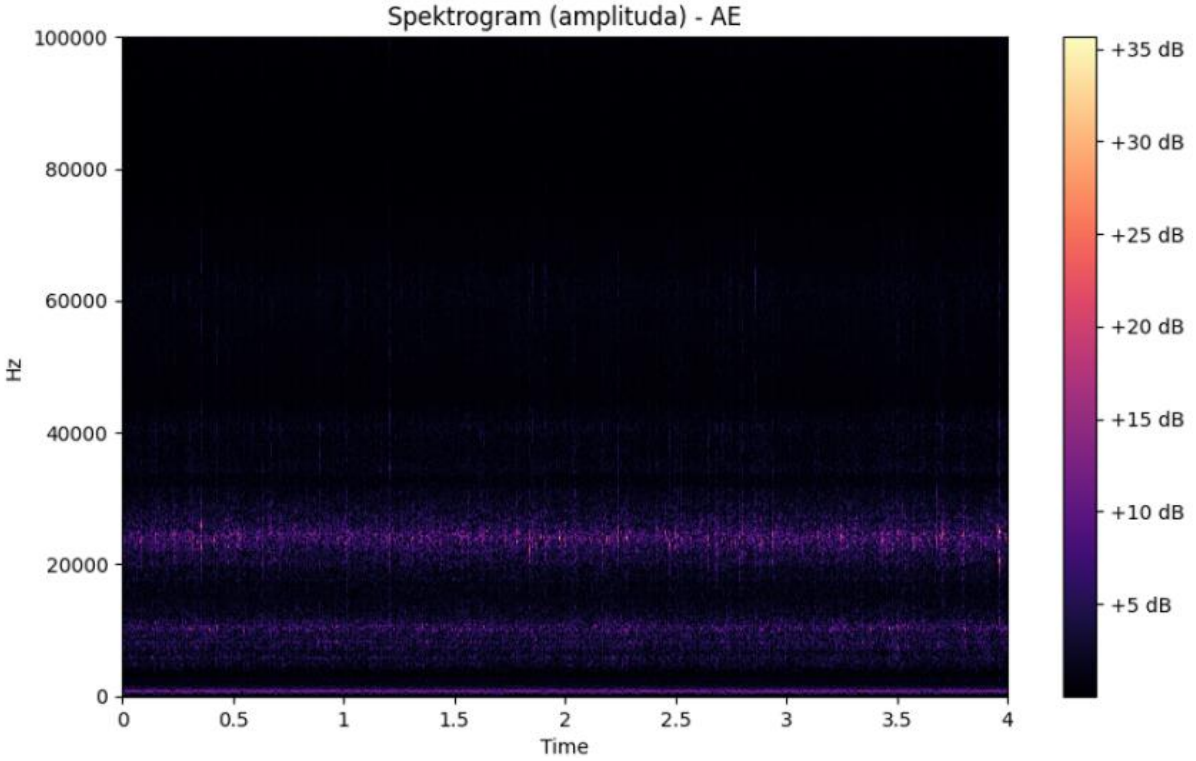
(4-6) $$\hat{x}(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi k}{N}n}$$

Kde *x(t)* je časový signál, *f* je frekvence a k je index frekvence, respektive *x(f)* a $\hat{x}(k)$ je frekvenční signál a *j* je imaginární jednotka. *N* je celkový počet časových vzorků a *n* je index konkrétního časového vzorku.

- Následně proběhne už pouze výpočet výkonového spektra jako čtverec absolutní hodnoty Fourierova spektra [47]:

(4-7) $$P(k) = |\hat{x}(k)^2|$$

- Spektrogram je následně zobrazen tak, že na ose x je časová osa, na ose y je osa frekvencí a barvami je vyznačen výkon, respektive intenzita signálu.



Obrázek 4-44: Ukázka spektrogramu pro signál akustické emise z turbínového režimu

4.5.2 Tvorba MEL-spektrogramů

MEL-spektrogramy jsou modifikací běžných spektrogramů. Oproti běžným spektrogramům jsou MEL-spektrogramy uzpůsobeny vnímání zvuku lidským sluchem. Nejčastěji jsou MEL-spektrogramy používány právě při zpracování lidské řeči, hudby a podobně. Lidský sluch je vyvinutý tak, že jsou lépe rozlišovány nižší frekvence a rozdíly mezi vyššími frekvencemi jsou vnímány méně výrazně. Výraznou výhodou MEL-spektrogramů

v kontextu této práce je ovšem fakt, že vytvoření MEL-spektrogramu na rozdíl od běžného spektrogramu redukuje data. Hodí se tak pro využití při zpracování velkých datových sad. [52]

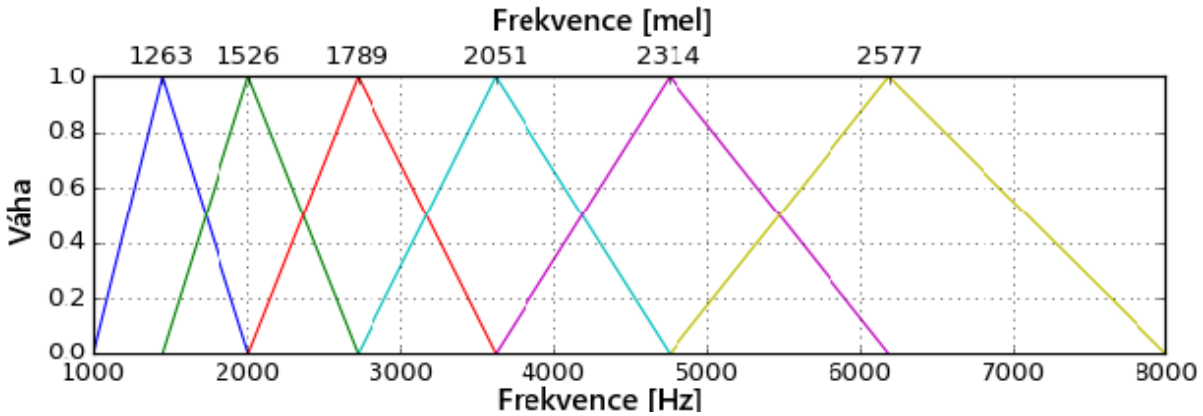
Tvorba MEL-spektrogramů probíhá stejně jako tvorba standardních spektrogramů popsaná výše. Rozdíl spočívá především v aplikaci MEL-filtrů a v použití logaritmické škály pro amplitudy. Postup je následovný: nejprve se frekvence získané Fourierovou transformací převedou do MEL stupnice pomocí následujícího vztahu [53]:

(4-8)
$$f_{mel} = 2595 \cdot \log\left(1 + \frac{f}{700}\right)$$

Kde f_{mel} jsou frekvence v MEL stupnici a f jsou frekvence získané z Fourierovy transformace. Tímto převodem je zajištěn efekt, kdy jsou nižší frekvence rozlišovány jemněji než frekvence vyšší. Následně je frekvenční rozsah rovnoměrně rozdělený a tyto určené body jsou přeneseny zpět na Hertzovu stupnici [53]:

(4-9)
$$f = 700 \cdot \left(10^{\frac{f_{MEL}}{2595}} - 1\right)$$

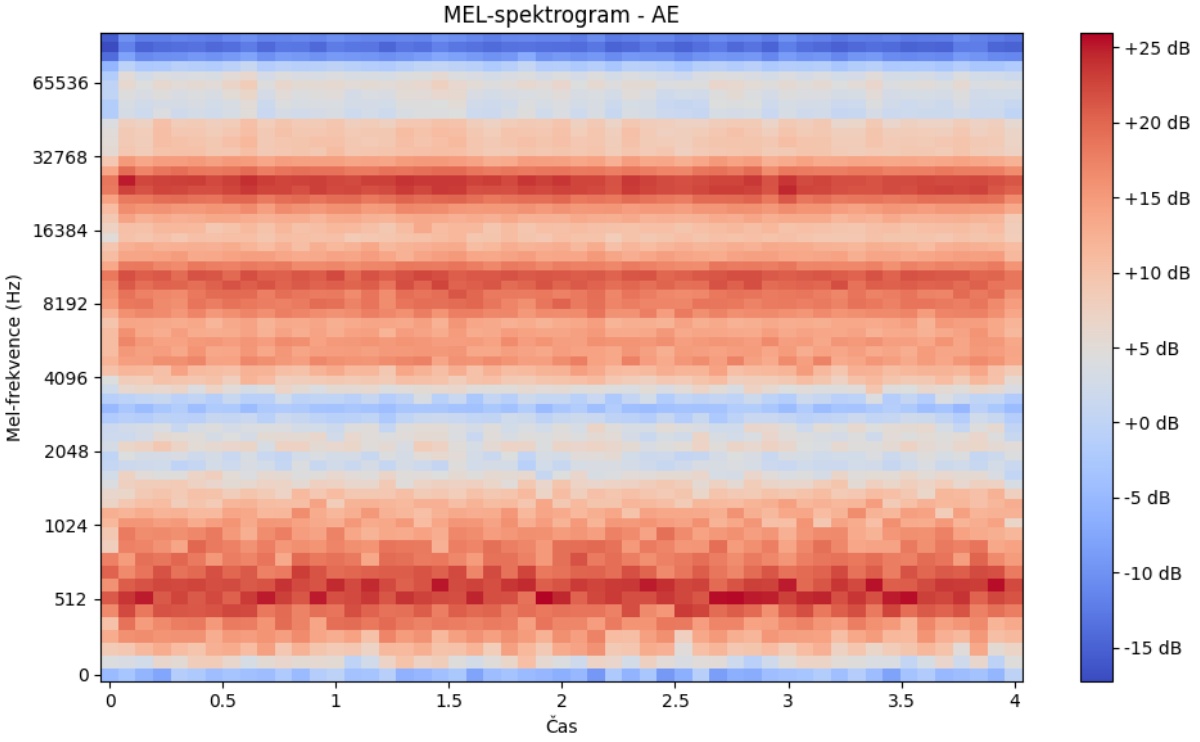
Na vytvořených bodech jsou následně naneseny trojúhelníkové filtry. Jejich počet je předem definovaný. Každý filtr je definován třemi body: levým dolním, vrcholem a pravým dolním. Filtry se navzájem překrývají podle definovaného požadovaného překrytí. Na ose y se nachází takzvané váhy, které určují, jak moc jsou jednotlivé frekvence zastoupeny v daném pásmu [53].



Obrázek 4-45: Vizualizace trojúhelníkových filtrů, upraveno [54]

Rozmístění filtrů odpovídá MEL měřítku a je tedy přibližně logaritmické. Na nižších frekvencích jsou filtry řidší, zatímco na vyšších jsou hustší. To má za následek přesně efekt lidského sluchu, kdy jsou lépe rozlišovány nižší frekvence, zatímco vyšší frekvence splývají do celků.

Lze tedy říct, že nižší frekvence mají vyšší rozlišení, zatímco vyšší frekvence mají rozlišení nižší. [52] [53]



Obrázek 4-46: Ukázka MEL-spektrogramu pro signál akustické emise z turbínového režimu

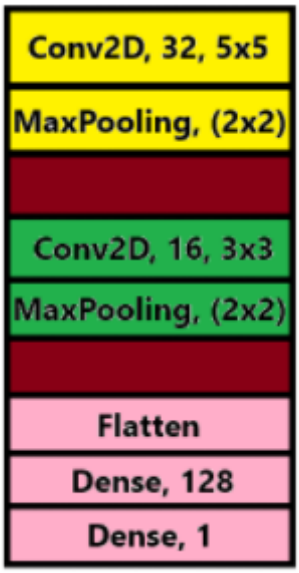
4.5.3 Aplikace MEL-spektrogramů jako vstupů do neuronové sítě

Stejně jako v předchozích dvou přístupech bylo i nyní vstupováno do modelu primárně všemi pěti kanály souběžně. Vstup do 2D konvoluční neuronové sítě měl tedy podobu pětice matic o rozměrech 64×53. Jednotlivé MEL-spektrogramy byly vždy tvořeny z celého souboru a obsahují tedy záznam z celých čtyř sekund. V první fázi byla použita plná vzorkovací frekvence 200 kHz. Následně bylo nutné zvolit rozměr okna, nad kterým bude vyhodnocena Fourierova transformace, velikost posuvu, která určuje překrytí jednotlivých oken, a počet použitých MEL-filtrů. Velikost okna je nutné volit tak, aby velikost byla mocninou čísla dva, a velikost posuvu je žádoucí zvolit s ohledem na tuto hodnotu, aby byl pokryt celý soubor s co nejmenší ztrátou dat. Volba těchto parametrů byla do jisté míry zautomatizována výpočtem přímo v kódu. Velikost okna pro FFT byla volena při frekvenci 200 kHz 16 384. Při snížení frekvence se pak i velikost okna snižovala poměrně. Dále do výpočtu vstupuje počet dílků na ose x, který byl zvolen 53. Pokud by mělo být MEL-spektrogramy vstupováno do autoenkóderů, bylo by vhodné, aby měly čtvercový rozměr, tedy například 64×64. Je dopočítávána velikost posuvu, v tomto případě 15 365. Počet MEL-filtrů byl volen 64. To charakterizuje rozměr ve směru osy y.

Rozřazení do testovací, trénovací a validační sady probíhá z této podstaty na úrovni spektrogramů, tedy celých souborů. Podobně jako předchozí přístupy i u této varianty byla testována přesnost na neznámých datech.

Ukázalo se, že preprocessing v podobě MEL_spektrogramu je velmi významným procesem, který rozpoznání stavu neuronové síti značně usnadňuje. Testováním modelů různé složitosti se ukázalo, že velice dobrých výsledků je dosahováno i s jednoduchými modely.

Konkrétně, pokud není řečeno jinak, byl použit model znázorněný na následujícím schématu, v konvolučních vrstvách jsou opět používány osvědčené aktivační funkce *ReLU*, dále je v nich využit padding, který v případě nutnosti doplňuje nuly. Pokud není uvedeno jinak byl použit learning rate 0,001 a batch size 32.

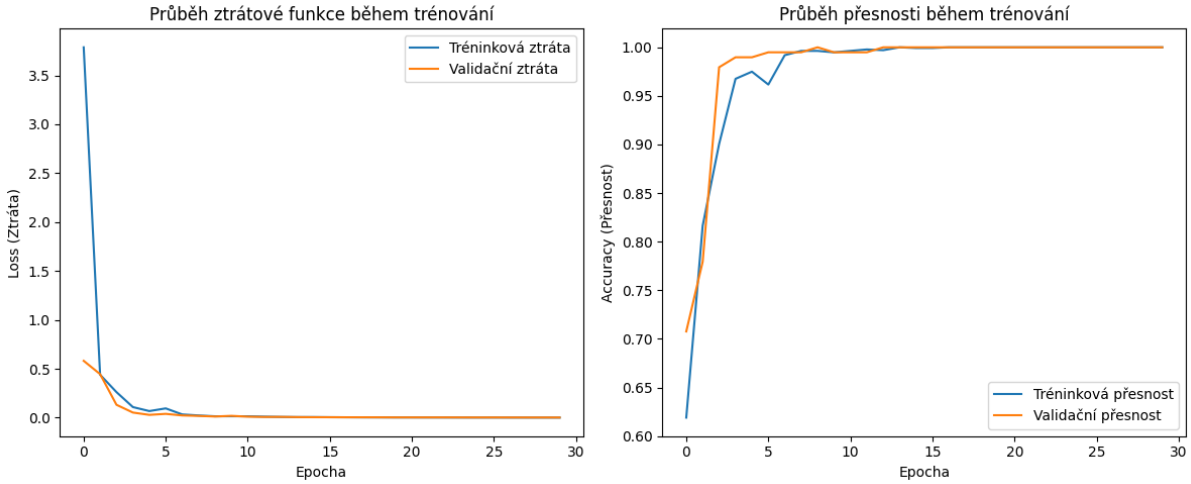


Obrázek 4-47: Schéma konvoluční sítě využité ke zpracování MEL-spektrogramů

K této části práce se vztahují kódy v příloze 7 ve složce „MEL“. Nejprve byly vytvořeny samotné MEL-spektrogramy, a to kódem „vytvoreni-MEL-spektrogramu.py“. Tyto MEL-spektrogramy byly následně načteny a byly na nich vytrénovány modely. K tomu slouží kód „MEL-trenovani-modelu.py“. K vyhodnocení natrénovaného modelu na neznámých datech pak slouží kód „MEL_vyhodnoceni_modelu.py“.

4.5.4 Data z čerpadlového režimu

V případě čerpadlového režimu bylo na tomto modelu se všemi kanály při frekvenci 200 kHz dosaženo přesnosti 100 %.



Obrázek 4-48: Průběh ztrátové funkce a přesnosti při zpracování MEL-spektrogramů v čerpadlovém režimu

Na výsledek bylo pohlíženo skepticky, ale po analýze nebyl nalezen důvod, proč by měl být výsledek neplatný. To se potvrdilo i na testování na vyřazených datech – na testovací sadě

bylo dosaženo přesnosti 100 % při redukci dat na polovinu i na třetinu. To probíhalo stejným způsobem jako v předchozích přístupech. Při vyřazení poloviny dat bylo na vyřazených souborech správně určeno 464 MEL-spektrogramů neobsahujících vadu, neurčitě nebyl označen žádný takový spektrogram a vada byla nesprávně odhalena pouze v jednom případě. Přesnost takového modelu je tedy 99,78 %. V případě testování na MEL-spektrogramech s vadou bylo správně určeno všech 504 vzorků a přesnost je tedy 100 %. Při testování na neznámých, vyřazených, stavech (při tréninku modelu na třetině dat) bylo při testování MEL-spektrogramů bez vady správně určeno 638 MEL-spektrogramů, neurčitá byla predikce v 6 případech a chybná v 35 případech. Přesnost je tedy 93,96 %. Při predikci spektrogramů s vadou bylo správně určeno 744 vzorků, neurčitě 10 a zcela špatně 2 vzorky. Přesnost je tedy 98,41 %.

Protože se výsledky jevily velmi slibně, byly testovány varianty, které uvažovaly různé redukce. Tyto redukce nebyly prováděny z nutnosti, kvůli prostorové náročnosti, jako například při využívání konvoluce nad surovými daty, ale především pro možnost efektivnějšího využití v praxi. Při využití v provozu by bylo nejvýhodnější využít co nejméně senzorů a co nejnižší vzorkovací frekvenci kvůli nižším pořizovacím nákladům. Byly testovány různé kombinace vzorkování a využití senzorů, přesnosti jsou uvedeny v následující tabulce:

Tabulka 7: Přesnosti na testovacích sadách pro různé kombinace vzorkovacích frekvencí a použitých senzorů

Použité senzory: / Vzorkovací frekvence:	200 kHz	100 kHz	50 kHz	25 kHz
Všechny	100 %	99,74 %	99,74 %	99,75 %
AE	100 %	100 %	98,19 %	94,56 %
mic-GRAS40PH	99,74 %	99,74 %	100 %	99,20 %
a-spiral	96,38 %	97,46 %	97,42 %	93,00 %
p-PCB113B28-spiral	91,12 %	94,83 %	92,76 %	92,02 %
p-PCB113B28-sani	87,95 %	93,05 %	86,04 %	86,56 %
p-PCB113B28-spiral a p-PCB113B28-sani	90,96 %	96,89 %	93,02 %	94,12 %

Průběhy učení jsou přiloženy v Příloze 1 – Průběhy učení při použití MEL-spektrogramů a veškerých dat v příloze diplomové práce „*Příloha 1-6*“. Grafy jsou roztříděny podle frekvence a jsou pojmenovány intuitivními názvy s názvem použitého senzoru.

Varianty, které se jevily nejvýhodněji, byly dále testovány na neznámých stavech a neznámých souborech (varianty, jejichž přesnosti jsou označeny zeleně). Nejvhodnější varianty byly zvoleny ty, které dosahovaly vysoké, ale ne nutně nejvyšší přesnosti, a to za co nejnižší vzorkovací frekvence. Kromě jednotlivých senzorů byly také testovány všechny senzory pohromadě a senzory tlaků pohromadě. Další kombinace, například kombinace mikrofonu a akustické emise, testovány nebyly, neboť u obou senzorů se předpokládala i tak vysoká přesnost, což se potvrdilo. Pro konzistenci s předchozími přístupy byl model opět trénován na polovině dat a na zbytku byl testován. Výsledky jsou uvedeny v tabulce. Pod přesností jsou vždy uvedeny počty vyhodnocených MEL-spektrogramů ve formátu bez poruchy:neurčitý výsledek:vada.

Tabulka 8: Přesnosti vybraných variant na neznámých souborech

Senzor:	Frekvence:	Testovací přesnost:	Přesnost rozpoznání poruchy:	Přesnost rozpoznání provozu bez poruchy:
Všechny	25 kHz	97,91 %	99,21 % (4:0:500)	98,28 % (457:3:5)
AE	100 kHz	100 %	100 % (0:0:504)	99,35 % (462:1:2)
mic-GRAS40PH	25 kHz	99,48 %	99.60 % (2:0:502)	99,78 % (464:1:0)
a-spiral	50 kHz	95,88 %	97,22 % (8:6:490)	95,91 % (446:4:15)
p-PCB113B28-spiral	100 kHz	89,69 %	84,52 % (46:32:426)	86,67 % (403:25:37)
p-PCB113B28-sani	100 kHz	82,99 %	87,30 % (41:23:440)	77,63 % (361:20:84)
p-PCB113B28-spiral a p-PCB113B28-sani	100 kHz	94,33 %	95,63 % (13:9:482)	92,04 % (428:10:27)

Průběhy učení jsou přiloženy v Příloze 2 – Průběhy učení při použití MEL-spektrogramů – polovina souborů v příloze diplomové práce „*Příloha 1-6*“. Grafy jsou pojmenovány intuitivními názvy s názvem použitého senzoru a použité frekvence.
Přesnost na neznámých stavech (trénink na třetině dat a testování na zbytku) byla vyhodnocována jednak konvenční metodou používanou v této práci – tedy se statickým thresholdem definovaným intervalem 0,4 až 0,6, kde hodnoty uvnitř tohoto intervalu byly označeny jako neurčité – a také metodou s optimalizovaným thresholdem. Metoda a přínos je popsán dále v této kapitole. K optimalizaci thresholdu byl využit kód „MEL-optimalizace-thresholdu.py“.

Tabulka 9: Přesnosti vybraných variant na zcela neznámých stavech

Senzor:	Frekvence:	Testovací přesnost:	Přesnost rozpoznání poruchy:	Přesnost rozpoznání provozu bez poruchy:
Všechny	25 kHz	99,00 %	87,30 % (69:27:660)	95,88 % (651:9:19)
Optimalizovaný threshold	0,433	99,00 %	89,81 %	96,32 %
AE	100 kHz	99,00 %	99,21 % (3:3:750)	98,23 % (667:4:8)
Optimalizovaný threshold	0,676	100 %	99,07 %	99,26 %
mic-GRAS40PH	25 kHz	100 %	99,07% (5:2:749)	98,67 % (670:2:7)
Optimalizovaný threshold	0,434	100 %	99,21 %	98,67 %
a-spiral	50 kHz	95,99 %	87,70 % (71:22:663)	80,41 % (546:14:119)
Optimalizovaný threshold	0,433	97,00 %	90,08 %	81,00 %
p-PCB113B28-spiral	100 kHz	87,00 %	72,35 % (156:53:547)	73,93 % (502:29:148)
Optimalizovaný threshold	0,541	89,00 %	74,34 %	76,58 %
p-PCB113B28-sani	100 kHz	85,00 %	80,56 % (102:45:609)	70,84 % (581:31:167)
Optimalizovaný threshold	0.217	90,00 %	91,11 %	65,68 %
p-PCB113B28-spiral a p-PCB113B28-sani	100 kHz	93,00 %	81,22 % (120:22:614)	72,61 % (493:20:166)
Optimalizovaný threshold	0,376	95,00 %	84,26 %	72,02 %

Průběhy učení modelů na třetině dat jsou přiloženy v Příloze 3 – Průběhy učení při použití MEL-spektrogramů – třetina souborů v příloze diplomové práce „*Příloha 1-6*“. Grafy jsou pojmenovány intuitivními názvy s názvem použitého senzoru a použité frekvence.

Jako nejefektivnější z hlediska přesnosti se jeví senzor akustické emise a mikrofon. Oba senzory dosahují velmi vysokých přesností a modely založené na datech z nich se zdají být

velice relevantní. Je potřeba zdůraznit, že především u mikrofonu je nutné senzor používat správně. Měření mikrofonem může narušit jakýkoliv hluk v okolí.

Dobrých výsledků se dosahuje také u snímače zrychlení a při použití všech senzorů najednou s nízkou frekvencí vzorkování. Samotný mikrofon při stejné vzorkovací frekvenci však dosáhl lepších výsledků. Nejméně přesné se ukazují senzory tlaku, nejhůře se pak jeví snímač tlaku umístěný na sání čerpadla. Poměrně uspokojivých výsledků lze dosáhnou při kombinaci obou tlakových snímačů.

4.5.5 Data z turbínového režimu

Na datech z turbínového režimu byla provedena analýza jednotlivých senzorů pouze při frekvenci 200 kHz. Projevují se zde stejné trendy, které jsou podrobněji rozebrány na čerpadlových datech.

Tabulka 10: Přesnosti na testovacích sadách pro různé kombinace použitých senzorů – turbínový režim

Použité senzory:	Vzorkovací frekvence:	200 kHz
Všechny		100 %
AE		100 %
mic-GRAS40PH		100 %
a-spiral		98,39 %
p-PCB113B28-spiral		91,94 %
p-PCB113B28-sani		79,03 %
p-PCB113B28-spiral a p-PCB113B28-sani		88,71 %

Průběhy učení jsou přiloženy v Příloze 4 – Průběhy učení při využití MEL-spektrogramů v turbínovém režimu v příloze diplomové práce „*Příloha 1-6*“. Grafy jsou pojmenovány intuitivními názvy s názvem použitého senzoru.

4.5.6 Modifikace MEL-spektrogramů

Při tvorbě MEL-spektrogramů existuje mnoho možných modifikací, které můžou ovlivnit i následné učení neuronové sítě. Na vybraném snímači bylo testováno několik z nich a byl sledován vliv na přesnost na testovací sadě. Pro zkoumání byl použit kanál p-PCB113B28-spiral a frekvence vzorkování 200 kHz. Tento senzor a frekvence byla vybrána, protože původní výsledky dávají prostor k projevení změny při tvorbě vstupů.

Bylo zkoumáno především ořízntutí dolních frekvencí, rozměry MEL-spektrogramů a použití normování hodnot podle maximální hodnoty ve spektrogramu. Dolní hranice použitých frekvencí byla volena od 500 Hz do 10 kHz. Při zkoumání rozměrů MEL-spektrogramů byl namísto původního rozměru 64×53 použit čtvercový rozměr 32×32, 64×64 a 128×128. První rozměr je určený počtem MEL-filtrů, který musí být roven mocnině dvou.

Tabulka 11: Přesnosti modelů na testovací sadě při úpravách MEL-spektrogramů

Úprava:	Originál	f_min = 500 Hz	f_min = 2500 Hz	f_min = 5000 Hz	f_min = 10000 Hz
Přesnost:	91,12 %	89,41 %	90,44 %	88,37 %	78,29 %
Úprava:	Originál	ref = max	MEL 32×32	MEL 64×64	MEL 128×128
Přesnost:	91,12 %	92,76 %	85,28 %	93,28 %	90,44 %

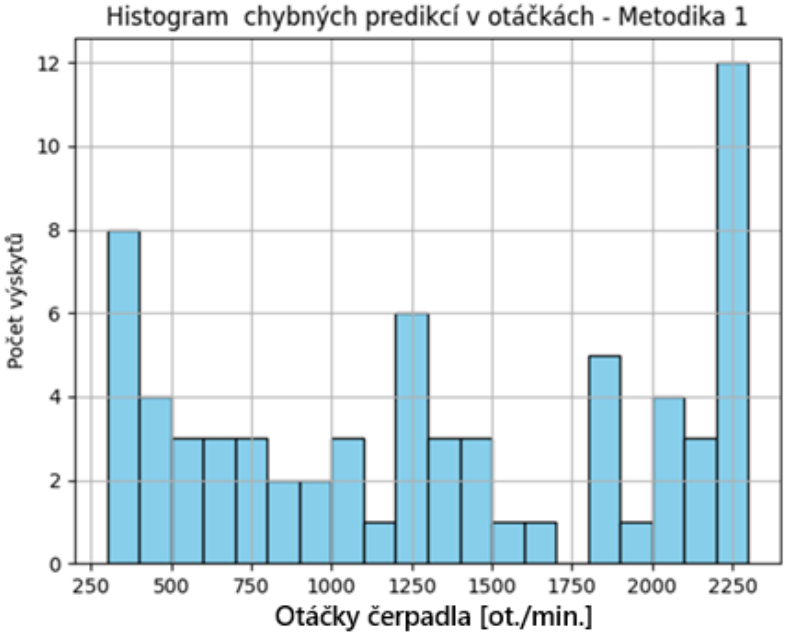
Průběhy učení jsou přiloženy v Příloze 5 – Průběhy učení při použití úprav MEL-spektrogramů v příloze diplomové práce „*Příloha 1-6*“. Grafy jsou pojmenovány intuitivními názvy s názvem popisujícím úpravu MEL-spektrogramu. Změny byly prováděny přímo při tvorbě MEL-spektrogramů, tedy v kódu „vytvoreni_MEL-spektrogramu.py“.

Zkoumané úpravy neměly ve většině případů významný vliv. Lehce pozitivní vliv byl pozorovaný při použití rozměru MEL-spektrogramu o rozměru 64x64 a při normování podle maxima. Významnější negativní vliv mělo pouze oříznutí frekvencí pod 10 kHz a použití rozměru 32x32. Tyto úpravy příliš redukují zpracovávané informace a schopnost zachytit důležité vzory se snižuje. Zbytek úprav měl jen lehce negativní efekt.

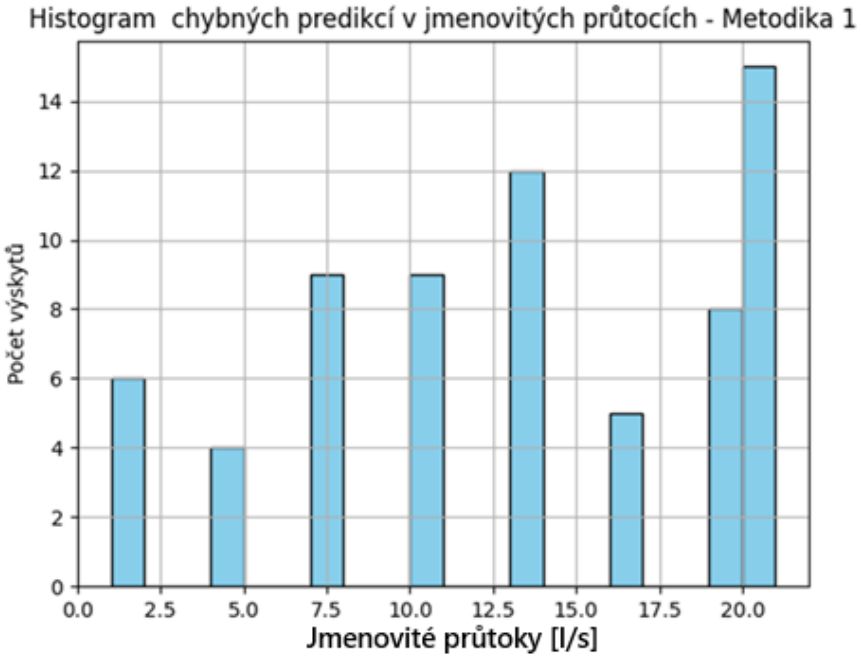
4.5.7 Přidání otáček a jmenovitého průtoku

Jak již bylo nastíněno v úvodu práce, protože byla provrtána právě jedna z lopatek, předpokládalo se, že projev poruchy v datech bude cyklický. Podle této hypotézy by se tedy porucha vyskytovala s určitou frekvencí, která by zřejmě nějakým způsobem mohla být navázána na frekvenci otáčkovou a tím pádem i otáčky samotné. Rozeznání vzorů by tak mohlo být posíleno vstoupením otáček do modelu. Pro ověření této varianty bylo přistoupeno ke dvěma variantám, jak otáčkami do neuronové sítě vstoupit. Otáčky byly uloženy v souborech s údaji o statice hydraulického systému. Pro výzkum této varianty byl použit, stejně jako v předchozím případě, senzor tlaku na spirále se vzorkovací frekvencí 200 kHz. Bude zkoumána i možnost vstupu jmenovitým průtokem.

Nejprve, pro lepší pochopení dat a funkčnosti modelu na nich, byly vytrénovány modely, kterými byla následně vyhodnocována data, a do histogramu byly vyneseny jednak otáčky a jednak jmenovité průtoky těch spektrogramů, které byly predikovány chybně. To bylo provedeno dvěma metodikami – Metodika 1 byla metodika, kdy byla k tréninku využita všechna data a tato data byla následně také vyhodnocena.

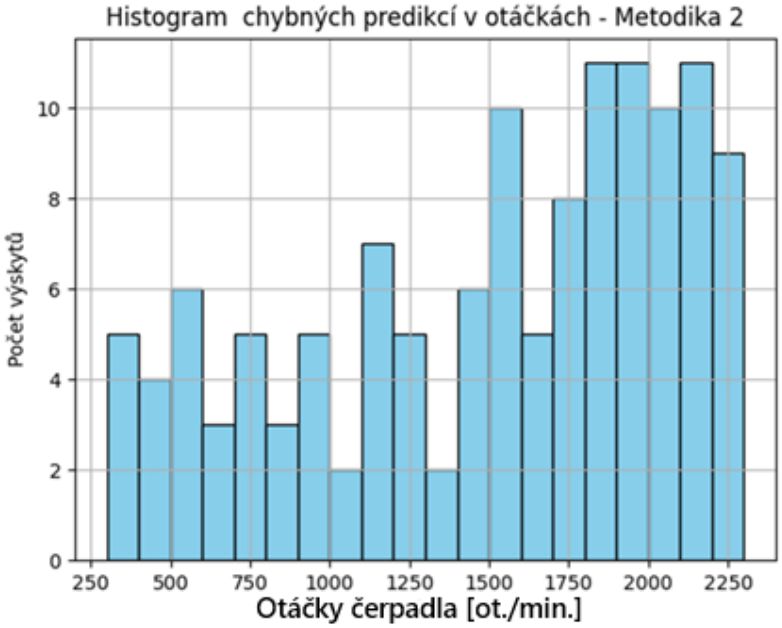


Obrázek 4-49: Histogram znázorňující počet výskytů chybné predikce ve spektru otáček podle Metodiky 1

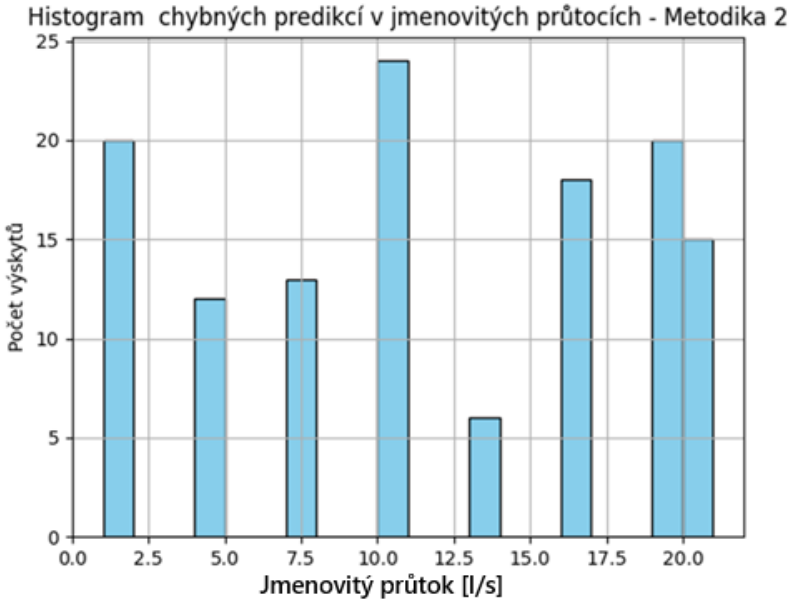


Obrázek 4-50: Histogram znázorňující počet výskytů chybné predikce ve spektru jmenovitých průtoků podle Metodiky 1

Nevýhoda Metodiky 1 je fakt, že kvůli predikování stejných dat, na kterých byl model tvořen, je celkový počet chybných predikcí nízký a histogramy nejsou tak statisticky relevantní. Proto byla navržena Metodika 2, která k tréninku používá polovinu dat a vyhodnocuje druhou polovinu. Polovina byla vytvořena stejně jako v případě testování modelů na neznámých souborech – tedy vyřazení každého druhého souboru. V tomto případě už bylo chybných predikcí víc a je možné lépe sledovat výskyt ve spektrech sledovaných veličin.



Obrázek 4-51: Histogram znázorňující počet výskytů chybné predikce ve spektru otáček podle Metodiky 2



Obrázek 4-52: Histogram znázorňující počet výskytů chybné predikce ve spektru jmenovitých průtoků podle Metodiky 2

V histogramech je poměrně problematické najít nějaké závislosti. Nanejvýš lze říct, že k chybným predikcím dochází spíše při vyšších otáčkách. Ve vztahu k průtokům žádné takové tvrzení vyřknout nelze.

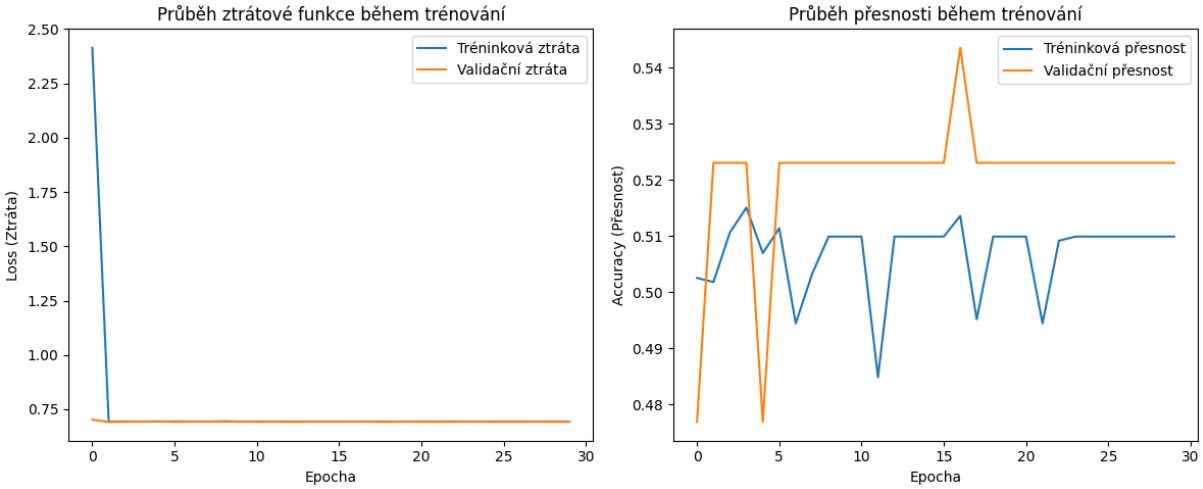
K tomuto vyhodnocení byl využit kód „MEL-vyhodnoceni-chybovosti.py“, který zpracovává již natrénovaný model.

Jak bylo řečeno, přidání statických veličin proběhlo dvěma způsoby. V první variantě bylo otáčkami vstupováno již při tvorbě MEL-spektrogramů, to se provádělo v kódu „vytvoreni-MEL-spektrogramu-s-otackami.py“. V tomto případě byl načtený signál (každý prvek signálu) podělený kvadrátem příslušných otáček při kterých byl naměřen:

(4-10)
$$signál = \frac{původní\ signál}{otáčky^2}$$

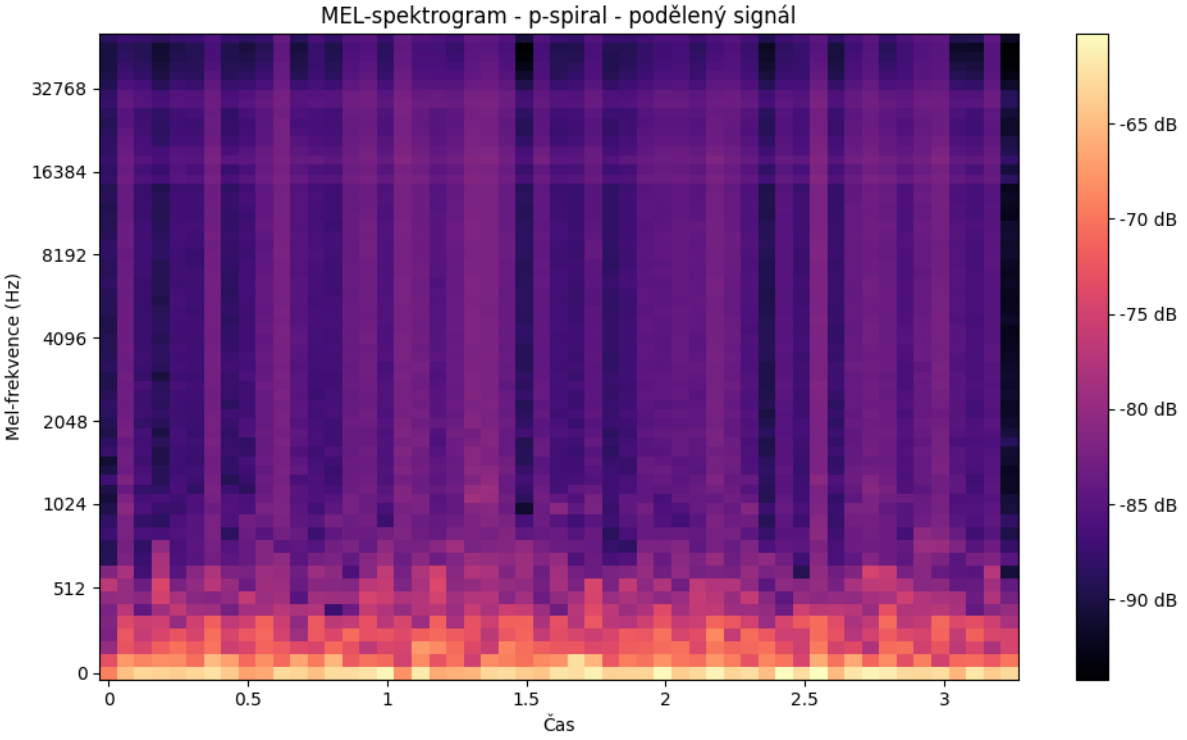
Podělení kvadrátem otáček bylo aplikováno, protože obecně platí, že tlakové pulzace jsou úměrné dopravní výšce, která zase podle afinních vztahů odpovídá druhé mocnině otáček. Toto tvrzení platí pro běžné tlakové pulzace, otázkou však zůstává, zda také tlakovým pulzacím na takto vysoké frekvenci.

Výsledek tohoto přístupu se zdál být nevyužitelný, neboť model se přestal zcela učit a výsledná přesnost se pohybovala jen lehce nad 50 %.

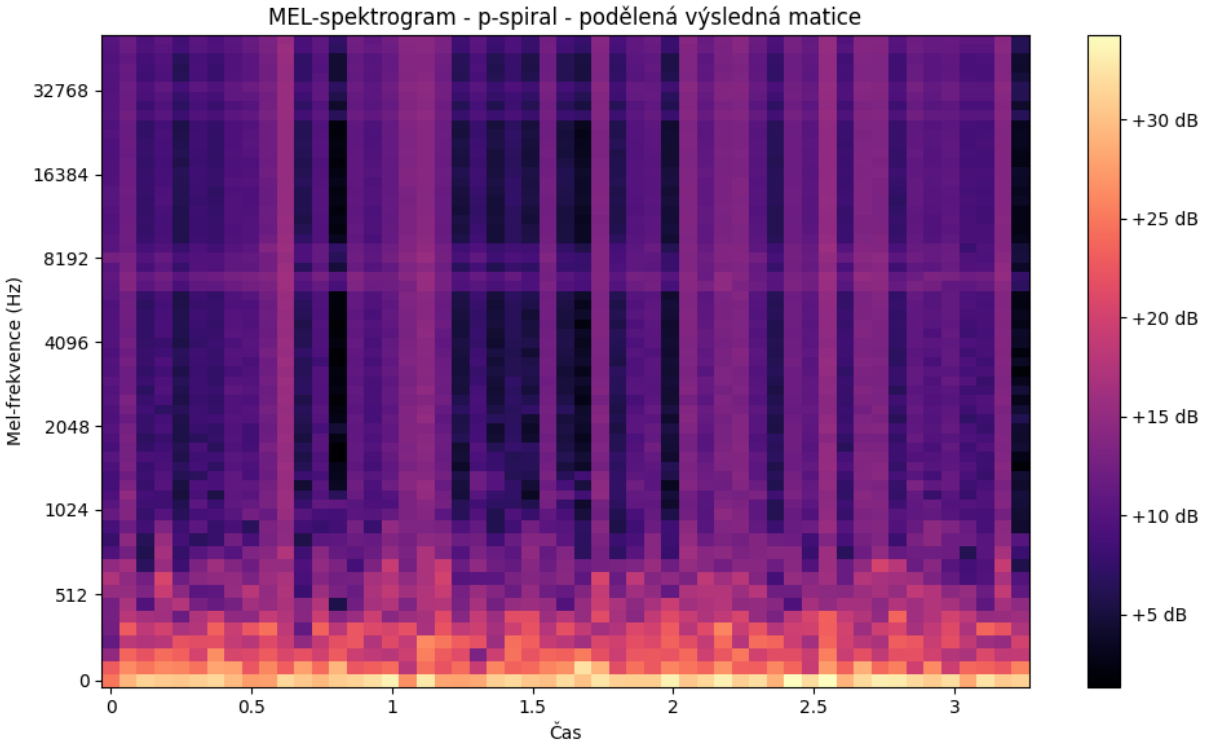


Obrázek 4-53: Průběh ztrátové funkce a přesnosti při zpracování MEL-spektrogramů vytvořených se signálem poděleným kvadrátem otáček

Pro porovnání jsou na následujících dvou obrázcích vzniklé MEL-spektrogramy – první je MEL-spektrogram vzniklý ze signálu poděleného kvadrátem otáček. Druhý, pro porovnání, vznikl z původního signálu. Jak je z obrázků zřejmé, liší se především v amplitudách, zatímco tvarem se jeden druhému blíží.

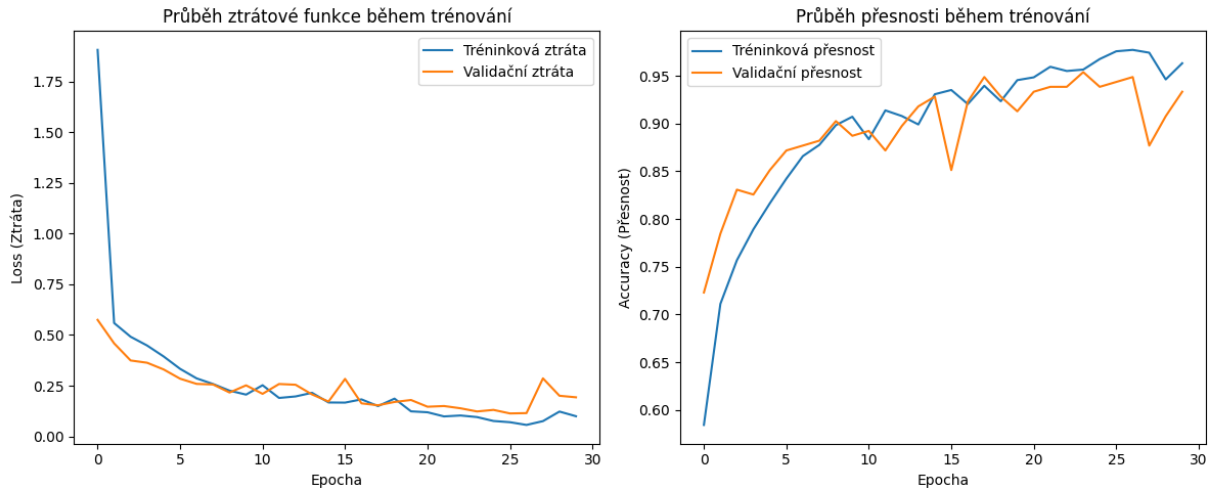


Obrázek 4-54: MEL-spektrogram vytvořený s přidáním otáček



Obrázek 4-55: Původní MEL-spektrogram

Druhým způsobem, jak zahrnout otáčky k dynamickému signálu, bylo použití vícero vstupů. K tomuto účelu slouží v tensorflow *layers.Concatenate()*. Kromě otáček byl tímto způsobem zároveň přidán také jmenovitý průtok. Nejprve byla provedena konvoluce nad samotnými maticemi MEL-spektrogramů, poté byla provedena operace *flatten*, tedy převedení na jednorozměrný vektor. K tomuto vektoru pak byla připojena hodnota otáček. Otáčky i jmenovitý průtok byly v tomto případě normalizovány na hodnoty od nuly do jedné. Trénink takového modelu se provádí v kódu „MEL-trenovani_modelu_s_pridanim_otacek.py“. Přesnost na testovací sadě v takovém případě byla 91,73 %. Tedy hodnota v podstatě stejná jako bez použití otáček.



Obrázek 4-56: Průběh ztrátové funkce a přesnosti při zpracování MEL-spektrogramů s přidáním dalšího vstupu v podobě otáček

Protože bylo podezření, že jediné dvě hodnoty normalizovaných otáček a jmenovitého průtoku se ve vektoru ztrácí, byla posílena jejich váha vynásobením konstantami. Byly zkoušeny řádově různé hodnoty těchto konstant od 1 do 100 000. Tento krok v žádném případě nepřinesl zlepšení, naopak u vyšších hodnot konstant již docházelo k prudkému zhoršení učení. Přidání otáček ani průtoku do modelu nepřináší výrazné zlepšení a zřejmě se jedná o redundantní informaci.

4.5.8 Bayesovská optimalizace thresholdu

Při určování přenosnosti, a tedy i použitelnosti, modelu hraje významnou roli hodnota thresholdu. Jedná se o hranici, se kterou je v kontextu binární klasifikace porovnáván výstup z posledního neuronu. Protože v posledním neuronu je zpravidla použita funkce *sigmoid,* výstup náleží do intervalu od 0 do 1. Ve stejném intervalu se musí nacházet i threshold. Pokud je výstup vyšší než threshold, je výsledek označen v této práci za vadu, pokud nižší, tak za chod bez poruchy. Je tedy zřejmé, jak threshold výslednou predikci ovlivňuje. Doposud byla v práci použita metoda, kdy byla hranice pevně definována jako 0,5. V případě vyhodnocování vyřazených dat pak byly za chod bez poruchy považovány výsledky, které byly nižší než 0,4, za chod s poruchou výsledky nad 0,6 a uvnitř intervalu byly výsledky označeny jako neurčité (tyto výsledky byly při vyčíslování přesnosti konzervativně považovány za chybné). Díky prostřednímu intervalu bylo možné alespoň částečně říct, jak je model jistý ve svých predikcích. Tento způsob není zcela optimální – představme si, že by data a model byly takové, že všechny vzorky, které vadu neobsahují, dávají na výstupu z modelu hodnoty do 0,2, vyšší hodnoty jsou pak výsledkem vzorků s vadou. V takovém případě by byla původní volba thresholdu nevhodná a zbytečně by se snižovala přesnost modelu. V tomto případě se tedy snažíme nalézt maximum funkce přesnosti s argumentem v podobě thresholdu. Předem neznáme tvar této funkce.

Nastíněný problém řeší optimalizační algoritmy – těch je několik. Často využívaný je například algoritmus grid search, který systematicky prochází všechny možné varianty v zadaném intervalu a přímo je vyhodnocuje. Výhodou je, že projitím všech variant nastavení dokáže najít to skutečně nejlepší. Ovšem tím, že musí všechny vyhodnotit, je výpočetně náročný [55]. Další možností je random search, který vyhodnotí několik náhodných variant. Tento algoritmus přináší pravděpodobnostní konvergenci, ale záruka najití globálního optima neexistuje [56]. Algoritmus využitý k optimalizaci thresholdu v této práci je algoritmus Bayesovské optimalizace. Předpokládáme neznámou funkci, která je na výpočet náročná, avšak je v každém bodě vyčíslitelná. Nejprve se zvolí několik náhodných parametrů, v tomto případě thresholdů, a vyhodnotí se cílová funkce, v našem případě přesnost modelu. Na základě těchto vyhodnocených hodnot se vytvoří náhradní model, takzvaný apriorní model. Jedná se o jakýsi prvotní odhad toho, jak cílová funkce vypadá. Nejčastěji se k tomu využívá Gaussovský proces. Cílová funkce je nyní předpokládaná jako pravděpodobnostní model, tedy model se střední hodnotou a nejistotami. Pomocí akviziční funkce se určují další body, další thresholdy, které se předpokládají, že budou zajímavé k vyhodnocení. Nejčastěji využívanou akviziční funkcí je *Expected Improvment* [57]*:*

(4-11)
$$EI_{y^*} = \int_{-\infty}^{\infty} \max\left(y^* - y, 0\right) p_M(y|x)\, dy$$

Kde y^* je nejlepší dosud nalezená hodnota cílové funkce (nejlepší přesnost), y hodnota cílové funkce při x (hodnota přesnosti při novém thresholdu), x je vstupní parametr (threshold), $p_M(y|x)$ je pravděpodobnostní model vypovídající, jaká je pravděpodobnost, že pro parametr x dostaneme výsledek y.

Po této volbě se tyto body vyhodnotí a na základě získaných hodnot se aktualizuje náhradní model – získáváme aposteriorní funkci. Tento krok se opakuje do chvíle, než dojde k provedení požadovaného množství iterací či naplnění jiného zastavovacího kritéria [58] [57].

Pro implementaci do kódu musela být naprogramována funkce vyhodnocující přesnost s použitím konkrétního thresholdu. Ta predikci modelu porovnává s tímto thresholdem – pokud je větší, označí vzorek za vadu, jinak za chod bez poruchy. Pokud se označení shoduje se skutečným stavem, je predikce označena jedničkou, pokud se neshoduje, tak nulou. Zprůměrováním těchto hodnot pro všechny predikce je pak získána přesnost pro konkrétní threshold. Bayesovskou optimalizací je pak hledáno maximum této funkce. Dále je nastavován interval, ve kterém má být threshold hledán. Interval byl nastaven od 0.1 do 0.9. Tento interval slouží k tomu, aby algoritmus nehledal nesmyslné hodnoty – výstup z modelu je díky funkci *sigmoid* vždy mezi 0 a 1. Dále je definován počet náhodných bodů vypočtených na začátku procesu a celkový počet iterací. To bylo iteračně zvoleno jako 11 a 16 [59].

Výsledky s optimalizovaným thresholdem jsou uvedeny v tabulce 9 vedle výsledků získaných předchozím způsobem. Oproti původnímu přístupu je s optimalizovaným přístupem znát zlepšení někdy i v jednotkách procent.

4.5.9 Optimalizace architektury pomocí knihovny Optuna

Optuna je framework umožňující snadnou optimalizaci hyperparametrů. Využívá Bayesovskou optimalizaci, která byla popsána výše [60]. Doposud byly v práci využívány modely neuronové sítě, který vznikaly manuálním zkoušením různých variant, z nichž byla vybrána ta nejlépe funkční. Framework Optuna umožňuje nalezení optimální kombinace hyperparametrů, podobně jako bylo popsáno, jak je hledán vhodný threshold. Optuna byla aplikována v kódu „MEL_opitimalizace_Optuna_logovani_tensorboard.py“. Výhodou využití Optuny je možnost nejen optimalizovat nastavení vrstev modelu, ale také dynamická definice počtu těchto vrstev. Poměrně jednoduše tak může být nalezen optimální model. Sledovaným parametrem, který se algoritmus snažil zmaximalizovat, byla opět přesnost modelu. V poslední fázi se pak přistoupilo k multiobjektivní optimalizaci, kdy byla maximalizována přesnost, a naopak minimalizována ztráta. Tento krok znamenal lepší kontrolu nad optimalizovanými modely. Bez optimalizace ztráty by se mohlo stát, že bude navržen model, který bude mít sice vysokou přesnost, ale ztráty relativně vysoké. To by mělo za následek sice dobrou funkci na nynějších datech, ale nízkou míru přenositelnosti. V případě že je optimalizována i ztráta, dochází k lepšímu zobecnění modelu. Díky tomu je pak možné takové nastavení modelu použít jako výchozí při optimalizaci modelu na jiných datech, například na datech ze skutečné lokality, což proces optimalizace výrazně urychlí. Návrhy, se kterými Optuna pracovala, vycházely z předchozích využitých modelů. Princip architektury byl zachován – uvažovala se 2D konvoluční síť s následnými plně propojenými vrstvami. Optimalizace probíhala s daty ze senzoru tlaku ve spirále p-PCB113B28-spiral a bylo použito podvzorkování na 100 kHz, protože ze všech variant se senzorem tlaku vycházela tato nejlépe a byla snaha vytvořit z dat z tlakového senzoru model stejně dobrý jako například z dat z mikrofonu. Následuje výčet a popis optimalizovaných hyperparametrů:

- Počet konvolučních vrstev

  Vždy byly aplikovány dvě 2D konvoluční vrstvy. Stejně jako v původním modelu. Následovala optimalizace počtu dalších přidaných vrstev, tento počet se nacházel v intervalu od 0 do 10. Celkem se tedy počet konvolučních vrstev mohl pohybovat od 2 do 12.

- Počet plně propojených vrstev

  Podobně jako u konvolučních vrstev vždy byla použita první plně propojená vrstva. Následně algoritmus optimalizoval počet dalších od 0 do 10. Celkem se tedy počet plně propojených vrstev pohyboval mezi 1 a 11.

- Počet filtrů v konvolučních vrstvách

  Pro všechny i dynamicky vytvořené konvoluční vrstvy se volil počet filtrů. Je vhodné, aby byl počet mocninou 2, proto se volilo z mocnin dvou od 4 do 512.

- Rozměr filtrů

  Rozměr kernelů v konvolučních vrstvách byl volen mezi rozměry (3,3), (5,5), (7,7) a (9,9)

- Počet neuronů v plně propojených vrstvách

  Ve všech použitých plně propojených vrstvách se optimalizoval počet neuronů. Podobně jako u počtu filtrů, byl i počet neuronů v plně propojených vrstvách mocninou 2 – od 8 do 512.

- Dropout

  Metoda slouží k získání vyšší robustnosti modelu, což mimo zabraňování přeučení může vést i k lepším výsledkům. Princip spočívá ve vypnutí, vyřazení náhodných neuronů v každé iteraci. Tím jsou neurony nuceny vytvářet a posilovat další kombinace propojení. U této regulační techniky je nutné vhodně určit procento vypnutých neuronů, což je opět věc optimalizace [61]. Dropout byl umístěn za každou plně propojenou vrstvu a optimalizace probíhala mezi 20 % a 50 % vypnutých neuronů, což jsou obecně doporučované hodnoty [62].

- Batch size

  Batch size byla volena z hodnot 32, 64 a 128.

- Learning rate

  Dobře nastavený learning rate je nutný pro správný trénink modelu. Jak bylo popsáno již v úvodní kapitole této práce, příliš vysoký learning rate může bránit konvergenci, zatímco příliš nízký learning rate může trénink výrazně zpomalovat. Nejprve byl interval learnining rate nastaven na (1e-6; 0,1). Výsledky s vyššími learning raty z intervalu nebyly použitelné ani slibné. Naopak dobrých výsledků se dosahovalo s learning rate v řádu 1e-4 až 1e-5. Interval byl proto v dalších iteracích upraven na (5e-6; 1e-4).

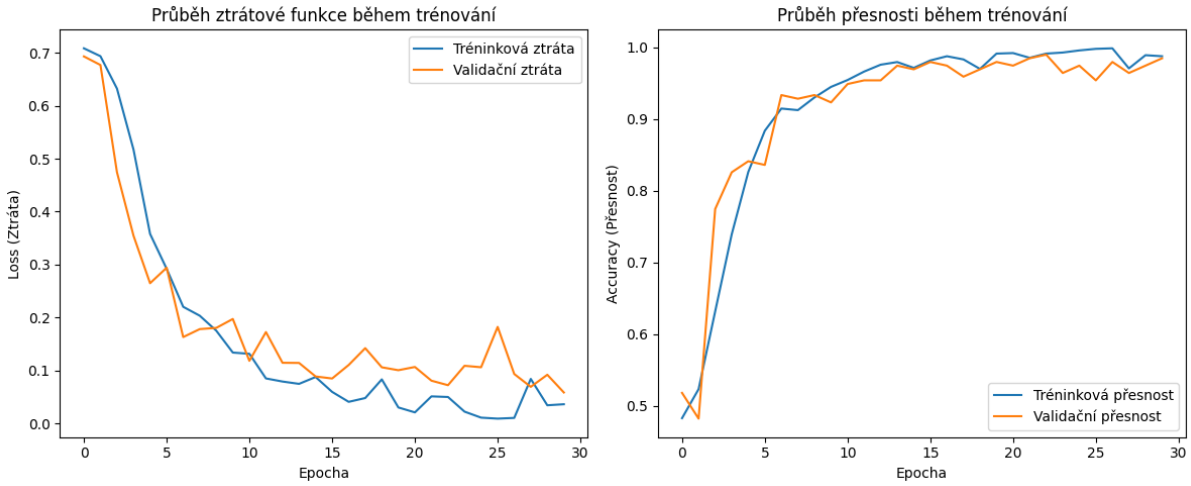
- Early stopping

  Jedná se o regulační techniku, která zastavuje proces tréninku modelu v momentě, kdy se model přestává učit, tedy ve chvíli, kdy křivka ztráty přestává klesat. Tím se zabraňuje přeučování modelu. Jiný způsob, jak může algoritmus rozeznat vhodný moment k zastavení učení, je sledování změny vektoru vah. V případě, že se vektor vah přestává výrazně měnit, může být proces učení ukončen [61].

U early stopping se volí takzvaná patience – tedy „trpělivost“ – je to počet epoch, při kterém, když nepoklesne sledovaná metrika (často, a i v tomto případě ztráta) proběhne ukončení tréninku. Patience u early stoppingu nebyla optimalizována pomocí optuny automaticky, ale hodnota byla volena ručně podle průběhů učení vykreslených v reportu generovaným při každé optimalizaci. Byla testována patience 3, 5 a 10. Pro konečnou iteraci byla zvolena patience 10, aby bylo dostatek prostoru pro řádné vytrénování. Volba této hodnoty, jak se zdálo z křivek učení, zároveň zabraňovala přeučování modelů.

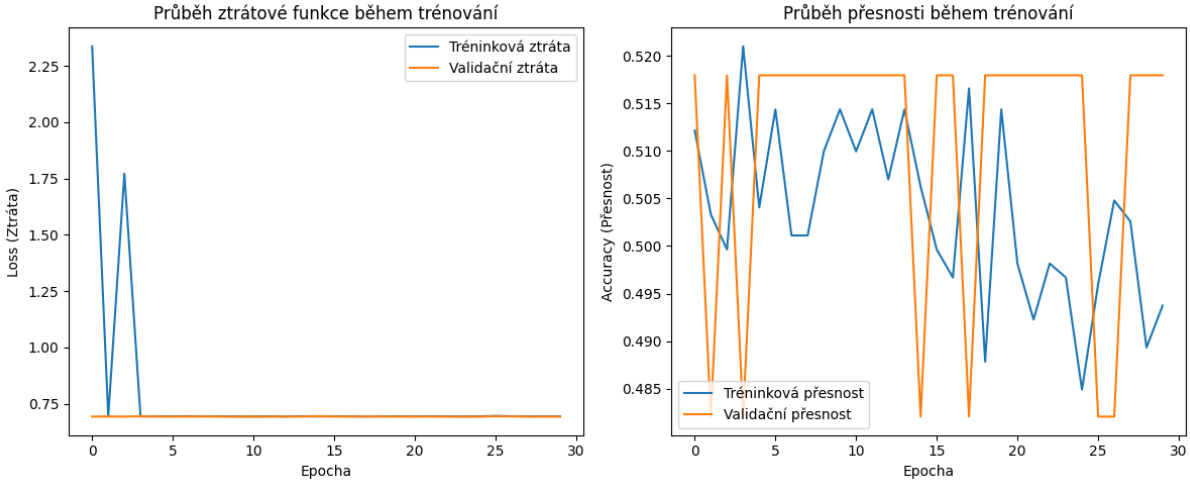
Výsledky jednotlivých iterací optimalizace byly zapisovány do nástrojové sady TensorBoard. Jedná se o nástroj, který umožňuje vizualizaci různých procesů jako například sledování křivky ztrát a přesnosti, vizualizace modelu pomocí diagramů nebo sledovaní hodnot vah atp. V případě kombinace s Optunou je možné vykreslovat také závislosti výsledné hodnoty sledované metriky (přesnost a ztráta) na volbě hodnot jednotlivých hyperparametrů. Díky tomu je možné sledovat, zda existuje nějaké ideální schéma modelu nebo například zda a jak jsou výsledné metriky závislé nebo naopak nezávislé na některých hyperparametrech.

Jak bylo naznačeno, optimalizace probíhala několikrát. Při každém spuštění optimalizace proběhlo 60 trialů, tedy 60 iterací optimalizace hyperparametrů. V průběhu se volily limity pro některé hyperparametry tak, aby se zlepšila efektivita hledání optima. Nejprve byl hledán interval pro volbu learning rate. Původní interval byl široký (1e-6; 0,1). Šířka intervalu zaručovala, že se v něm nachází i vhodné hodnoty, ale zároveň množství možností ztěžovalo jejich hledání. Při analýze reportu první iterace byla nalezena souvislost mezi learning rate a výslednou přesností. Ukázalo se, že křivky učení vypadají slibně pouze u learning rate řádově 1e-4 a menší. To, že je křivka „slibná“ se v tomto případě rozumí, když je na křivkách znát trend učení. Například následující graf z první iterace, který znázorňuje učení při learning rate 3,23e-4:



Obrázek 4-57: Ukázka slibného průběhu učení

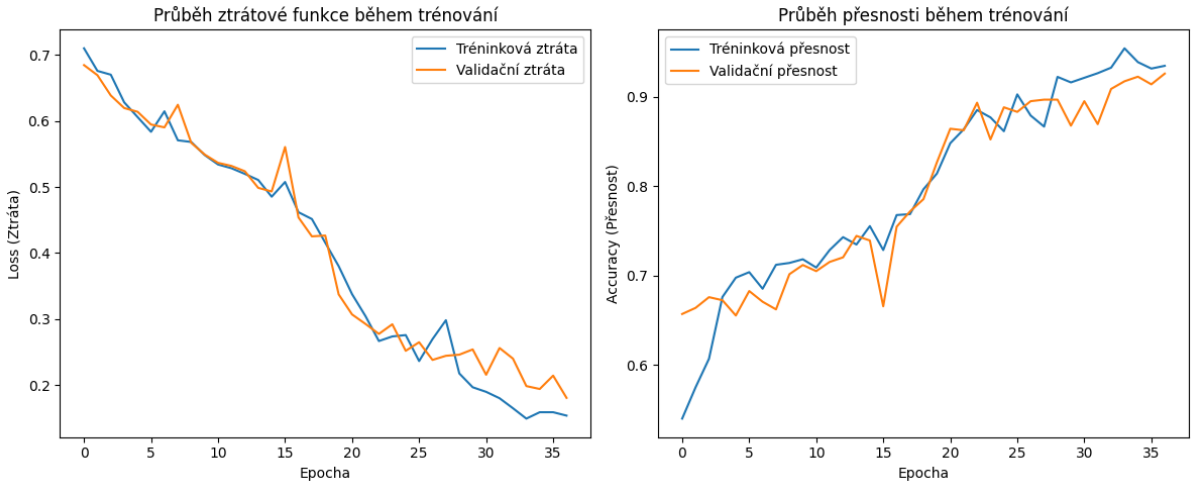
Naopak pokud na křivkách není učení zřetelné, ztráta se nesnižuje a přesnost osciluje okolo nízké hodnoty, pak v tomto kontextu takovou křivku rozumíme jako neslibnou. Na ukázku je uvedena varianta s learning rate 0,045 také z první iterace:



Obrázek 4-58: Ukázka průběhu učení, který není slibný

Na základě tohoto poznatku byl v dalších iteracích použit nový užší interval pro volbu learning rate (5e-6;1e-4). Tento interval v sobě obsahuje pouze vhodné hodnoty, a proto může algoritmus efektivněji optimalizovat další hyperparametry. Protože na některých křivkách bylo znát nedostatečné množství validačních dat, které se projevovalo například nižší validační ztrátou, než byla ztráta tréninková na počátku procesu učení, bylo změněno rozdělení dat do jednotlivých sad. V druhé iteraci bylo proto použito rozdělení trénovací:testovací:validační na 5:2:3, ve třetí iteraci 5:2,5:2,5 a ve finálních iteracích pak opět 5:2:3 (původní rozdělení bylo 7:2:1). To zajistilo větší množství validačních dat a zmírnění problému. Dále bylo v iteracích pracováno s hodnotou patience v early stopping tak, jak bylo popsáno výše při popisu samotného early stopping. Tedy s hodnotami 3, 5 a 10. Early stopping jako takový byl přidán především proto, aby varianty mohly být trénovány na větším počtu epoch. V první iteraci bylo použito 30 epoch, což byla pevná hodnota. To znamenalo, že ať se varianta jevila slibná nebo ne, proběhlo 30 epoch. Byla vůle slibné varianty trénovat déle s více epochami, abychom zjistili skutečný potenciál dané varianty. Naopak u varianty, která nepřinášela žádný výsledek a nebyla slibná, bylo i 30 epoch příliš a akorát byl mařen čas a výpočetní výkon. Počet epoch byl tedy nastaven na maximum 300 s tím, že byl použit právě early stopping.

Erly stopping s patience 3 i 5 se podle křivek učení zdálo málo, protože při pohledu na některé grafy šlo vidět, že by v dalších epochách mohlo dojít ke zlepšení:



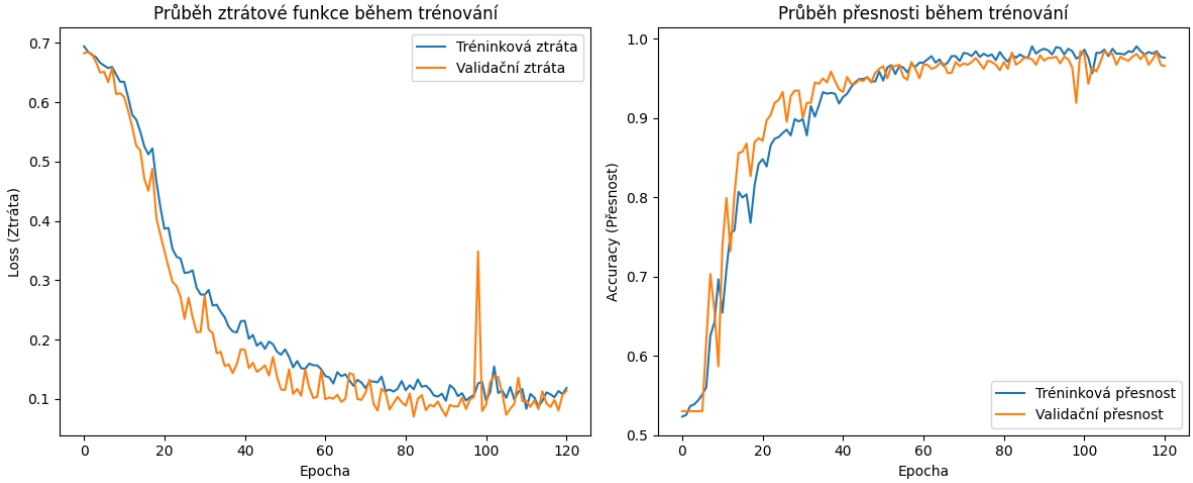
Obrázek 4-59: Průběh učení při nízké hodnotě patience

Poté, co se nalezl vhodný interval pro learning rate a vhodná patience, se přešlo z optimalizace přesnosti na multiobjektivní optimalizaci přesnosti a ztráty. Důvod a výhody byly popsány výše. Tato finální iterace byla tedy iterace s multiobjektivní optimalizací, s patience 10 epoch, s learning rate v užším intervalu a s rozdělením dat do sad v poměru 5:2:3. Oproti předchozí iteraci se tedy liší pouze multiobjektivní optimalizací. Nutno dodat, že přidání tohoto kroku prodloužilo dobu optimalizace dvakrát až třikrát. Reporty z iterace 4 (maximalizace přesnosti) a iterace 5 (multibojektivní optimalizace) jsou k nahlédnutí v Příloze 6 – Automatické reporty z Optuny v příloze diplomové práce „*Příloha 1-6*“. V reportech je možné sledovat průběh tréninku pro všechny varianty, které algoritmus vyhodnocoval.
Modely s nejlepšími výsledky z posledních dvou iterací jsou zpracovány dále v tabulce:

Tabulka 12: Výsledky přesnosti a ztráty nejlépe optimalizovaných modelů

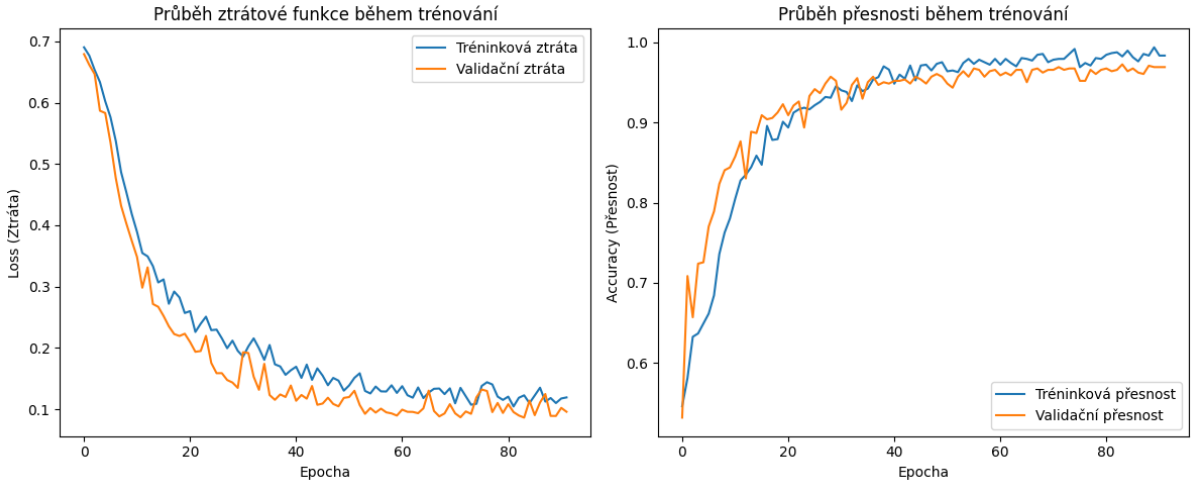
	Hodnocení podle přesnosti	Hodnocení podle ztráty
Iterace 4: optimalizace přesnosti	Model A 97,16 % 0,1304	Model B 96,90 % 0,1068
Iterace 5: Optimalizace přesnosti a ztráty	Model C 96,64 % 0,1499	Model D 95,61 % 0,1384

Model A: 5 konvolučních vrstev s počty filtrů: 64, 64, 64, 4, 64 o rozměrech filtrů 9x9, 7x7, 5x5, 9x9, 5x5, pouze 1 plně propojená vrstva s 8 neurony a dropoutem s koeficientem 0,285. Learning rate byl 5,60e-5 a batch size 32:



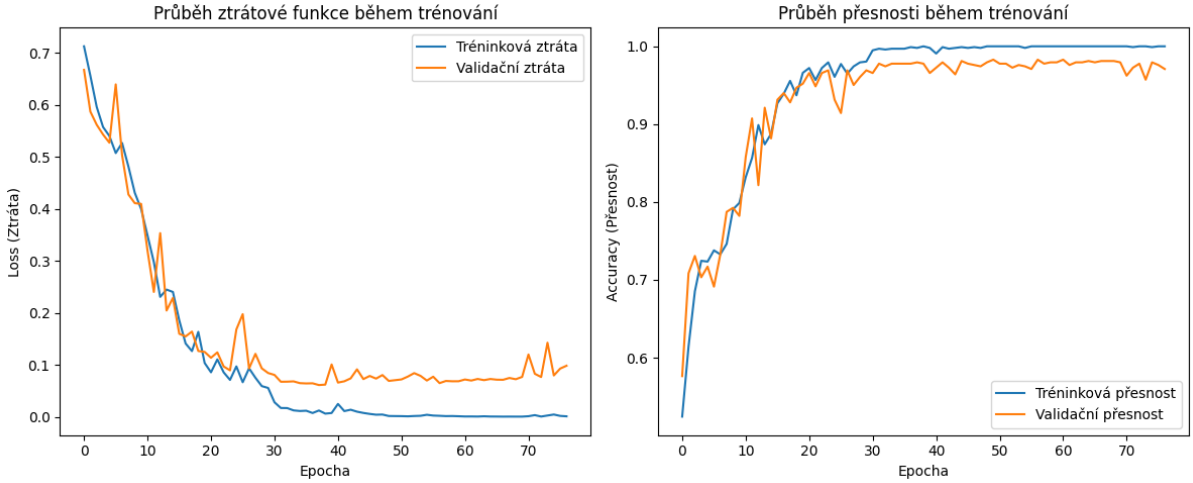
Obrázek 4-60: Průběh učení modelu A

Model B: 5 konvolučních vrstev s počty filtrů: 64, 64, 64, 4, 64 o rozměrech filtrů 9x9, 7x7, 5x5, 9x9, 5x5, pouze 1 plně propojená vrstva s 8 neurony a dropoutem s koeficientem 0,301. Learning rate byl 5,54e-5 a batch size 32:



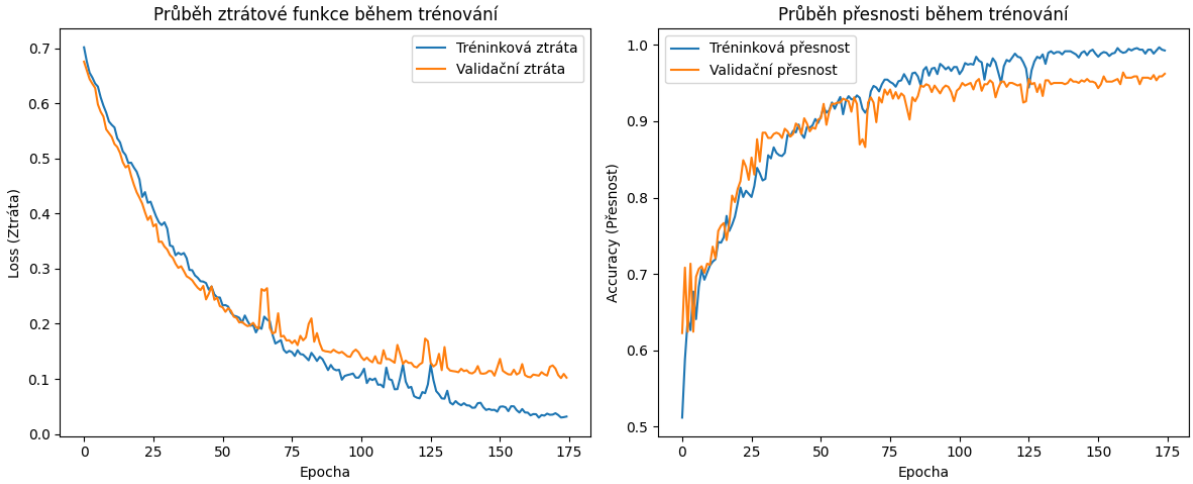
Obrázek 4-61: Průběh učení modelu B

Model C: 5 konvolučních vrstev s počty filtrů: 128, 128, 128, 128, 64 o rozměrech filtrů 7x7, 5x5, 7x7, 9x9, 3x3, pouze 1 plně propojená vrstva s 256 neurony a dropoutem s koeficientem 0,426. Learning rate byl 5,54e-5 a batch size 64:



Obrázek 4-62: Průběh učení modelu C

Model D: 5 konvolučních vrstev s počty filtrů: 32, 8, 256, 16, 64 o rozměrech filtrů 7x7, 7x7, 7x7, 9x9, 5x5, pouze 1 plně propojená vrstva s 32 neurony a dropoutem s koeficientem 0,365. Learning rate byl 3,43e-5 a batch size 128:

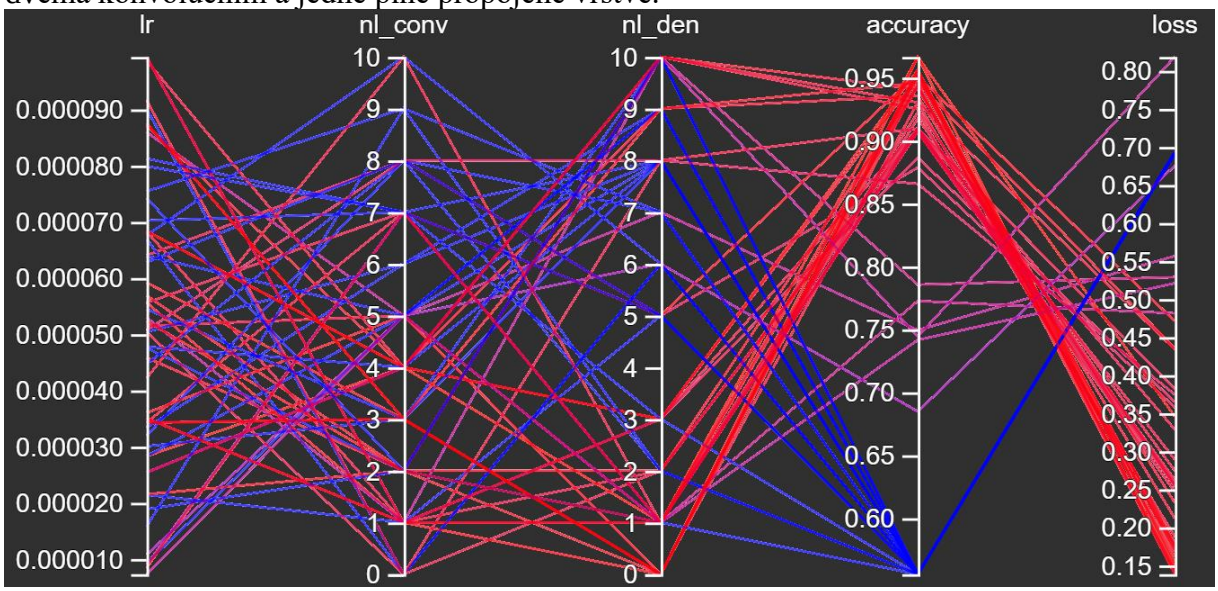


Obrázek 4-63: Průběh učení modelu D

Přestože poslední iterace, tedy iterace s optimalizací obou metrik, by měla být korektnější, tak z hlediska jak přesnosti, tak ztráty, se dosáhlo lepších výsledků ve variantě s optimalizací pouze přesnosti. Výsledky nejsou nikterak zásadní, v obou případech je minimálně struktura modelu stejná – vždy se skládá z pěti konvolučních vrstev a pouze jedné plně propojené. To, že se dosahuje lepších výsledků při méně korektní optimalizaci, zřejmě souvisí s náročností multiobjektivní optimalizace. Pro nalezení optimálního řešení je zřejmě potřeba více trialů než v případě jednoduché optimalizace. Z důvodu časové náročnosti multiobjektivní optimalizace, která je na použitém počítači v řádu dní, již navýšení trialů neproběhlo.

Jak bylo zmíněno, poslední iterace byla zapisována do prostředí tensorboard. Je tedy možné sledovat graficky závislosti. Následující obrázky znázorňují spojnicemi volby jednotlivých hyperparametrů a výslednou hodnotu metrik, a to z poslední iterace optimalizace.

Je možné pozorovat například závislost výsledku na volbě learning rate a počtu konvolučních a plně propojených vrstev. Při tomto pozorování je dobré připomenout, že veličiny nl_conv a nl_den značí počet konvolučních, respektive plně propojených vrstev navíc oproti základu – dvěma konvolučním a jedné plně propojené vrstvě.



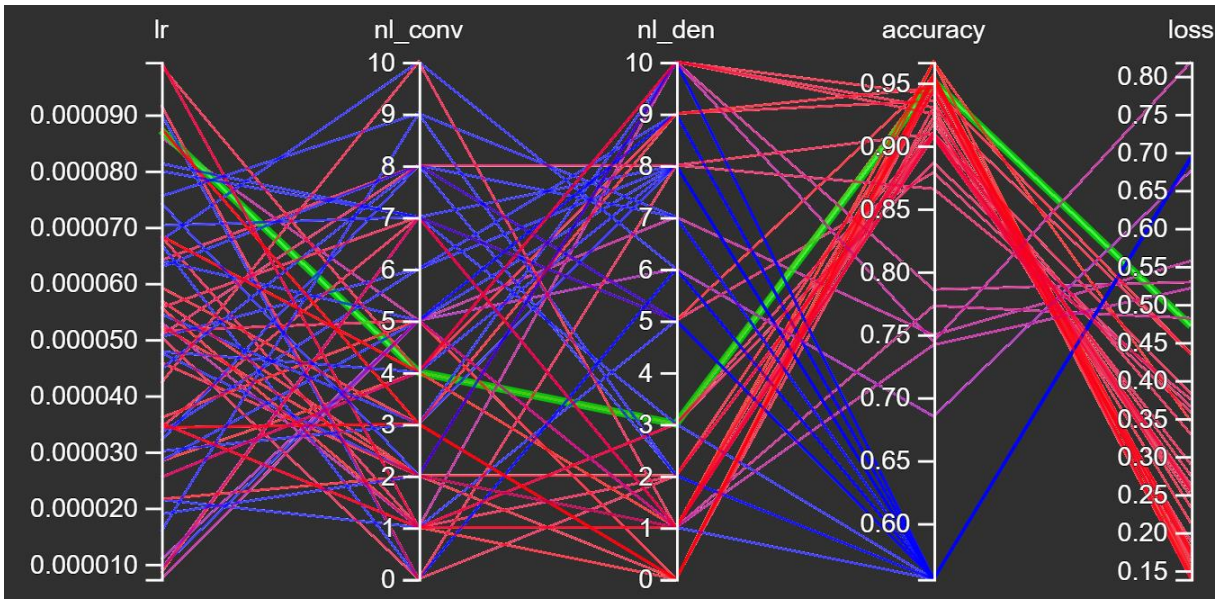
Obrázek 4-64: Závislost výsledných metrik na počtu dalších vrstev a learning rate

Ze spojnic je dobře pozorovatelné, že vzhledem ke vhodně zvolenému intervalu volby pro learning rate (lr), který byl poměrně úzký, byly výsledné metriky nezávislé na hodnotě tohoto hyperparametru. Rovnoměrné rozmístění červených a modrých spojnic po celém intervalu značí, že s každou hodnotou z tohoto intervalu lze dosáhnout jak dobrého, tak špatného výsledku, což se tedy odvíjí od dalších hyperparametrů.

Další pozorovatelný trend souvisí s množstvím konvolučních a plně propojených vrstev. V obou případech jsou spojnice hyperparametrů úspěšných trialů koncentrovány spíše na nižších počtech vrstev. Výraznější je tento trend u plně propojených vrstev. Zajímavé také je, že varianty, které měly vysoký počet plně propojených vrstev, a i přesto dosahují vysoké přesnosti, dosahují vyšších ztrát než varianty s nižším počtem plně propojených vrstev s obdobnou přesností. Respektive nejlepší varianta s počtem dalších plně propojených vrstev vyšších než 5 dosáhla při přesnosti 93,54 % ztráty 0,212 (při 9 přidaných plně propojených vrstvách, tedy 10 celkem). Varianty s 5 a méně dalšími vrstvami dosahovaly ztráty běžně pod 0,150.

Na tomto obrázku je dobré se zaměřit také na vztah mezi výslednými metrikami, tedy na vztah mezi přesností a ztrátou. Vidíme, že ačkoliv lze obecně říct, že vyšší přesnost znamená nižší ztrátu, neplatí to absolutně. Je sice vidět, že modré spojnice dosahující nízkých přesností mají vysoké ztráty a červené spojnice zase ztráty nižší.

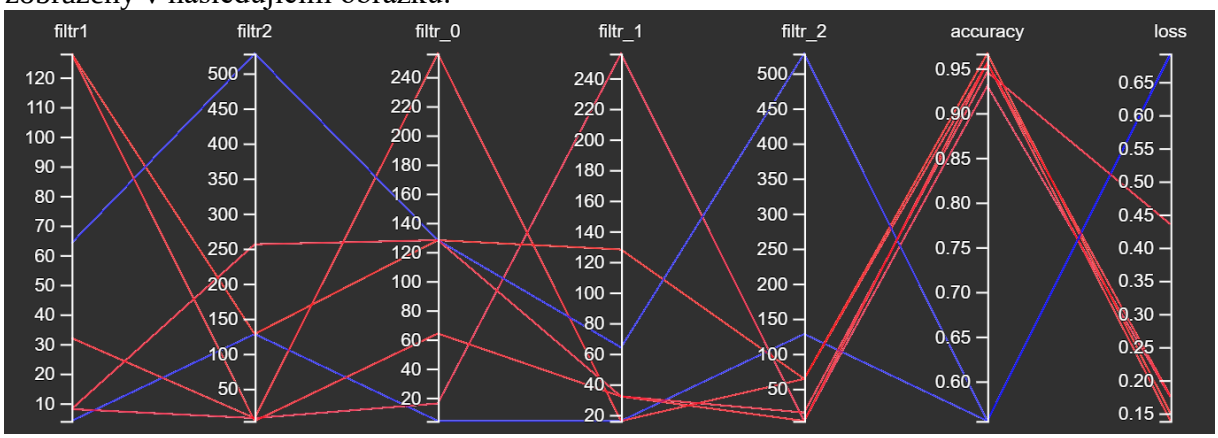
V grafu se ovšem nachází také spojnice s vysokou přesností, ale také s relativně vysokou ztrátou. Tj. například varianta se zelenou spojnicí s přesností 95,09 % a ztrátou 0,471:



Obrázek 4-65: Znázornění varianty s vysokou přesností a vysokou ztrátou

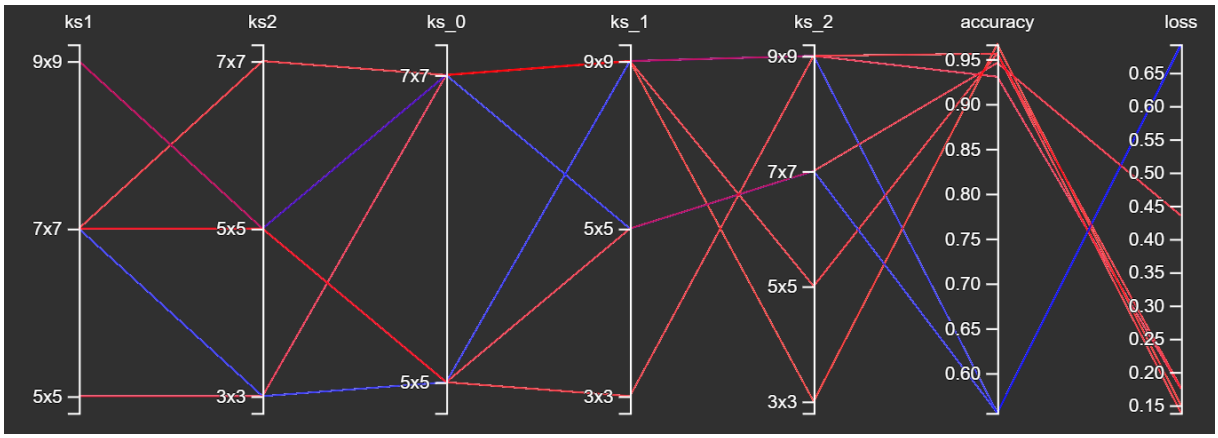
Tyto modely hůře zobecňují problematiku a jsou méně přenositelné.

Dále je možné sledovat vliv počtu filtrů v jednotlivých konvolučních vrstvách. V tomto případě tensorboard zobrazuje jen ty varianty, které obsahují všechny navolené hyperparametry. Pokud se tedy rozhodneme sledovat vliv počtu filtru například v 8. konvoluční vrstvě, zobrazí se pouze ty varianty, které obsahují alespoň 8 konvolučních vrstev. Nejlepších přesností a ztrát dosahovaly modely, které obsahovaly celkem 5 konvolučních vrstev. Jednotlivé počty filtrů v těchto variantách (které mají právě přesně 5 konvolučních vrstev) jsou zobrazeny v následujícím obrázku:



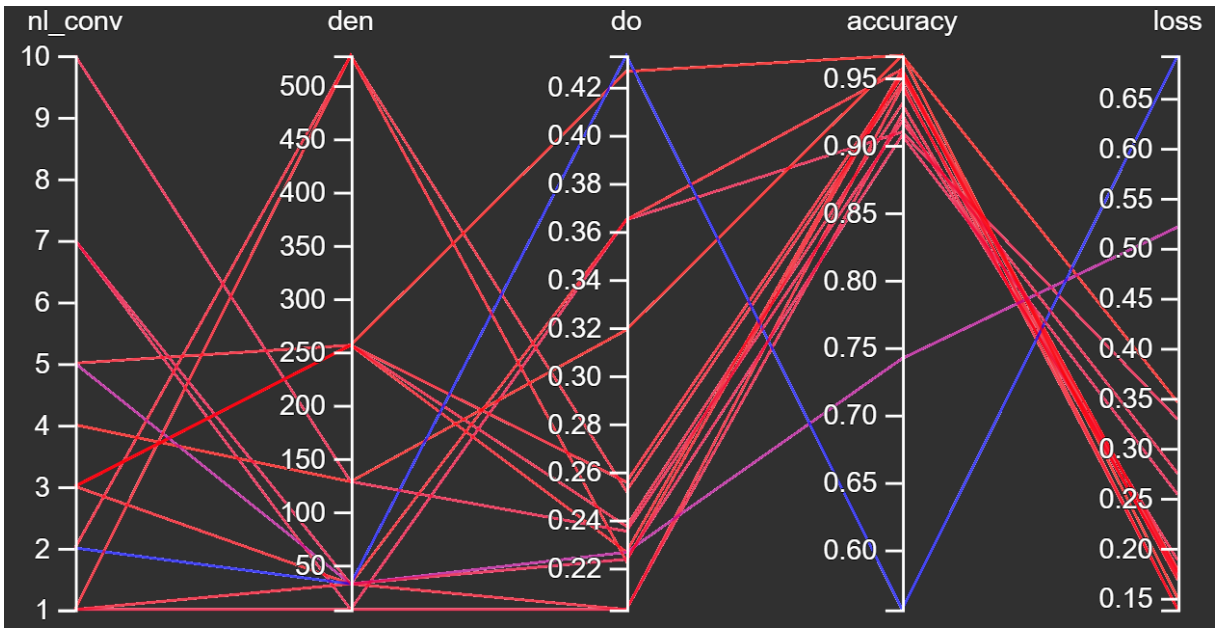
Obrázek 4-66: Znázornění vlivu počtu filtrů v jednotlivých konvolučních vrstvách v modelech s 5 konvolučními vrstvami

Totéž je možné provést i s rozměry těchto kernelů:



Obrázek 4-67: Znázornění vlivu rozměru filtrů v jednotlivých konvolučních vrstvách v modelech s 5 konvolučními vrstvami

Zdá se, že nalézt obecnou závislost nebo vzorec v nastavení filtrů pro modely s konkrétním počtem konvolučních vrstev je problematické a závisí spíše na konkrétní kombinaci než na nějakém znatelném trendu. Opět je potřeba zdůraznit, že také pro zvýšení relevance interpretace by bylo dobré napočítat vyšší vzorek trialů. Jak již bylo řečeno, kvůli vysoké časové a výpočtové náročnosti tohoto procesu se k dalšímu napočítání nepřistoupilo.
Podobně nebyl žádný výrazný trend pozorován ani u počtu neuronů v plně propojených vrstvách. Nejlepších výsledků se dosahovalo, pokud byla použita pouze jedna plně propojená vrstva následující po konvoluci. Byly proto analyzovány varianty s právě jednou plně propojenou vrstvou. Sledoval se počet neuronů v této vrstvě (den), koeficient dropoutu (do) a pro kontext počet konvolučních vrstev před touto plně propojenou vrstvou:



Obrázek 4-68: Znázornění vlivu počtu neuronů v plně propojené vrstvě, koeficientu dropoutu a počtu dalších konvolučních vrstev v modelech s právě jednou plně propojenou vrstvou

Ukazuje se, že valná většina variant modelů s pouze jednou plně propojenou vrstvou dosahuje dobrých výsledků s přesností nad 90 %. Dvě varianty dosáhly výrazně horších výsledků. Opět ovšem není nasnadě nějaký trend.

Nakonec byla ještě pro variantu s nejlepší výslednou přesností, tedy pro model A, provedena optimalizace thresholdu. Po optimalizaci byla přesnost 97,67 % s použitým thresholdem 0,437. Oproti původní variantě s neoptimalizovaným modelem ani thresholdem je tedy zlepšení přesnosti modelu založeném na senzoru p-PCB113B28-spiral o necelá 3 %.

Ačkoliv je tedy při optimalizaci znát zlepšení, při využití jiných senzorů, jako například snímače AE, se dosáhlo lepšího výsledku i při použití původního, neoptimalizovaného modelu. Volba použitých dat, respektive použitého senzoru, zůstává z hlediska úspěšnosti výsledku zcela zásadní a kritická. Nicméně optimalizačními algoritmy, ať už architektury modelu, či thresholdu, lze výsledky dále zlepšovat.

4.5.10 Shrnutí

Jak v turbínovém, tak v čerpadlovém režimu lze při preprocessingu pomocí MEL-spektrogramů dosáhnout stoprocentní nebo téměř stoprocentní přesnosti, a to i na neznámých souborech, a dokonce i stavech. Těchto výsledků lze dosáhnout i pouze za samostatného využití snímače akustické emise nebo mikrofonu.

Spolehlivost mikrofonu v provozu je předmětem diskuse, protože může snadno dojít k narušení měření. Z tohoto důvodu se jako v praxi nejvyužitelnější varianta jeví snímač akustické emise. Model vytrénovaný výhradně na tomto snímači se při rozpoznávání MEL-spektrogramů z neznámých souborů tak i stavů, mýlil pouze v jednotkách případů.

Dále bylo pracováno se snímačem tlaku na spirále, protože tento senzor nabízel prostor pro zlepšení. Byly provedeny různé modifikace MEL-spektrogramů ve smyslu změny jeho rozměrů a oříznutí dolních frekvencí. Ukázalo se, že těmito kroky nelze dosáhnout znatelného nárůstu, naopak při přílišném oříznutí dolních frekvencí a při zmenšení rozměru MEL-spektrogramu na 32×32 se již výsledky zhoršují zřejmě proto, že dochází k příliš významné redukci informací. Pozitivní výsledky nepřineslo ani přidání otáček a jmenovitých průtoků do procesu učení ať už při vnesení při tvorbě MEL-spektrogramů, tak ani při vnesení přímo v modelu neuronové sítě, jak je popsáno výše. Co však pozitivní efekt mělo, byla aplikace optimalizace – optimalizován byl jednak threshold a jednak hyperparametry modelu.

5 Závěr

Práce se v prvních kapitolách věnuje vysvětlení základních principů a algoritmů fungování modelů neuronových sítí. Znalosti a poznatky z těchto teoretických kapitol byly následně aplikovány při tvorbě konkrétních systémů na řešených datech. V další kapitole byl popsán konkrétní řešený problém – tedy detekce poruchy na čerpadle BETA 12YC zapojeném v hydraulické trati v hydraulické laboratoři na Fakultě strojního inženýrství VUT v Brně na Odboru fluidního inženýrství Viktora Kaplana. V této kapitole je blíže popsána uměle vytvořená porucha a také použité snímače, pomocí kterých byla získána analyzovaná data.

Hlavním cílem této práce bylo navrhnout systém strojového učení pro stanovení poruchy na hydraulickém stroji. K řešení tohoto problému byly použity tři hlavní přístupy, které jsou porovnávány dále v závěru. Stejně tak jsou v závěru blíže komentována omezení těchto systémů, a to i v souvislosti s využitím v praxi. Obecně však lze říct, že systém byl úspěšně navrhnut.

5.1 Shrnutí výsledků a srovnání jednotlivých variant

Problém binární klasifikace učením s dozorem byl řešen třemi přístupy – konvoluční síť pracující se surovými daty a s MEL-spektrogramy a plně propojená síť pracující se statistickými charakteristikami. Všechny tři metody byly v menší či větší míře funkční. Ačkoliv záleží i na nastavení hyperparametrů samotného modelu neuronové sítě, největší vliv měl na konečné výsledky právě preprocessing použitých dat. To je významná myšlenka, která by měla být mimo jiné výrazným výstupem a zkušeností z této práce. Neuronové sítě jsou jistě výkonným nástrojem, ovšem teprve se správným přístupem se stávají funkčními.

Ze všech tří přístupů vychází jednoznačně nejlépe konvoluční síť pracující s MEL-spektrogramy. Přesnosti modelů jsou vyšší než u zbývajících dvou metod na veškerých datech. Přesnosti dosahovaly na některých senzorech (senzor AE a mikrofon) až 100 %, na neznámých datech se pak přesnosti na těchto senzorech pohybují okolo 98 % až 99 %. O něco hůře, ale stále uspokojivě, funguje samotný mikrofon při využití surových dat a konvoluční sítě. Tady se přesnosti pohybují až okolo 94 % a na neznámých datech pak okolo 81 % až 92 %. Akustická emise dosahuje výsledků o něco horších. Naopak nejhůře se jeví využití plně propojené sítě zpracovávající statistické charakteristiky, kde se modely ukazují jako nevhodné pro využití na neznámých datech.

Pro aplikaci v praxi, v případě využití učení s dozorem, je tedy využití MEL-spektrogramů jasná volba. Tato varianta má sice oproti metodě „hrubé síly“ se surovými daty teoretickou nevýhodu, kdy vyhodnocení probíhá se zpožděním vyšším o čas, který je potřebný ke tvorbě MEL-spektrogramu. Při podvzorkování na 100 kHz však vychází doba potřebná na vytvoření jednoho MEL-spektrogramu na zhruba 0,020 s a při 200 kHz 0,027 s. Čas na tvorbu MEL-spektrogramu tedy není drastický a vzhledem k předpokládanému využití a k tomu, že samotné měření probíhá 4 sekundy, je rozhodně zanedbatelný

Obecně nejlepších výsledků je dosáhnuto při použití všech snímačů zároveň, při použití samotného mikrofonu a při samotném použití snímače akustické emise. Jak v případě surových dat, tak v případě MEL-spektrogramů je samotný mikrofon a o něco méně (v případě surových dat) i akustická emise rovnocenná, co se přesnosti týče, k použití všech pěti snímačů zároveň. V praxi je tedy zbytečné, a tedy i finančně nevýhodné aplikovat všechny snímače společně. Zároveň by se celý systém musel odstavit při poruše pouze jednoho snímače, což je také neefektivní. Snímač akustické emise je oproti mikrofonu lépe rezistentní vůči externímu narušení. V případě použití mikrofonu musí být přísně nastaven a vymáhán provozní řád, protože jakýkoliv hluk v okolí může měření znehodnotit. Z těchto argumentů tedy vyplývá, že nejlepší k využití je konvoluční síť pracující s MEL-spektrogramy vytvořenými z dat snímače

akustické emise. Při bližším zkoumání se ukázalo, že dostačující vzorkovací frekvence je v tomto případě 100 kHz.

Na zkoumání MEL-spektrogramů ze signálu snímače tlaku ve spirále se zahrnutím dat ze statických snímačů otáček a jmenovitého průtoku se ukázalo, že tyto hodnoty jsou pro modely redundantní a jejich přidání nevede ke zlepšení výsledných metrik. Stejně tak se ukázalo, že chybovost predikce zkoumaného modelu není závislá na hodnotě nastaveného jmenovitého průtoku a na otáčkách je závislá jen velmi lehce až nevýznamně.

5.2 Omezení a využitelnost modelů

Největším omezením či nedostatkem, se kterým se tato práce potýkala, byla jednolitost naměřených dat. Tento nedostatek má dvě roviny. Jak bylo popsáno v předchozích kapitolách, pro čerpadlová data bylo naměřeno 21 charakteristik při různých otáčkách pro různé tlaky v kotli. První rovina nedostatku spočívá v tom, že tyto charakteristiky byly měřeny pouze jednou. Bylo by vhodné, pokud by všechny charakteristiky byly měřeny víckrát, třeba v jiné dny, což by přineslo více dat ke zpracování umožňující kvalitnější trénink. Vícero měření by totiž do dat zaneslo různé stochastické jevy, které by pomohly zobecnit modely. Druhá rovina nedostatku spočívá v tom, že byla naměřena pouze jedna vada. Zajímavé by bylo, pokud by bylo naměřeno více typů vad, například vyvrtání různých průměrů děr, vyvrtání děr v různých polohách lopatky nebo třeba provrtání různého počtu lopatek. Tato data by umožnila sledovat, jak model dokáže fungovat na poruchách, které nezná z tréninku. V turbínovém režimu bylo měřeno ještě výrazně méně dat. Pozornost byla proto věnována především čerpadlovému režimu.

Určitým pokusem o eliminaci tohoto nedostatku v této práci je nevyužití veškerých dat k tréninku, ale rozdělení dat na část použitou k tvorbě modelu a na část použitou k vyhodnocení. Byly použity dvě metodiky, z nichž se každá pokouší simulovat jinou situaci. První metodika spočívá ve vyřazení každého druhého souboru, a tedy tvorba modelu probíhá na celých charakteristikách, které jsou pouze řidší. Druhá metodika pak vyřazuje konkrétní části charakteristik. Blíže jsou metodiky popsány v předchozích kapitolách. Ačkoliv je jisté, že tento přístup zcela nenahrazuje další měření, minimálně získává pohled na práci modelů na datech, která se nějakým způsobem liší od dat použitých při jeho tvorbě, a dodávají tím alespoň částečnou informaci o přenositelnosti metody. Využití MEL-spektrogramů dosahovalo na vyřazených datech téměř totožných výsledků jako na testovacích sadách při tvorbě modelu. Lze tedy vyslovit předpoklad, že by metoda dobře fungovala i na datech s jinou vadou, případně na jiném stroji, například na konkrétní lokalitě vodní elektrárny či čerpací stanice. Relativně dobré shody v tomto ohledu dosahovaly i modely pracující se surovými daty. Naopak při využití statistických charakteristik se tyto výsledky poněkud lišily a přenositelnost je tedy spíše nepravděpodobná.

Jak již bylo zmíněno, s přihlédnutím na spolehlivost a ekonomickou stránku věci se pro využití v praxi jeví nejlépe využití snímače akustické emise s frekvencí vzorkování alespoň 100 kHz. Přesnosti takového modelu se pohybují v rozmezí 98 % - 100 %. Lze předpokládat, že přesnost by se v ostrém provozu spíše snížila, i přesto jsou výsledky nadějné a metoda by mohla sloužit k dodatečnému dohledu na chod stroje doplňující běžnou diagnostiku.

5.3 Další výzkum

Nevýhodou systému pracující na základě učení s dohledem je potřeba mít dostatek naměřených dat pro chod s poruchou i bez poruchy. Zatímco naměřit data bez poruchy problematické není, měření dat s poruchou již problematické být může, protože chod s poruchou je samozřejmě na skutečné lokalitě zcela nežádoucí. Z tohoto důvodu by bylo dobré směřovat další výzkum k metodám učení bez dohledu, například k autoenkóderům. Na základě výsledků této práce lze předpokládat, že bude výhodné do autoenkóderů vstupovat právě MEL-spektrogramy.

SEZNAM POUŽITÝCH ZDROJŮ

[1] HYDROGRID. *Hydrogrid.ai* [online]. [cit. 2025-04-23]. Dostupné z: https://www.hydrogrid.ai/

[2] VOITH. How artificial intelligence works. *Voith.com* [online]. 2019 [cit. 2025-04-23]. Dostupné z: https://voith.com/corp-en/news-room/stories/how-artificial-intelligence-works.html

[3] Búðarháls Power Station. *Landsvirkjun.com* [online]. [cit. 2025-04-23]. Dostupné z: https://www.landsvirkjun.com/powerstations/budarhalsstod

[4] *AI vs. machine learning vs. deep learning vs. neural networks: What's the difference?* [online]. In: . 2023 [cit. 2024-09-13]. Dostupné z: https://www.ibm.com/think/topics/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks

[5] STARMER, Josh. *Neural Networks / Deep Learning by StatQuest* [online]. [cit. 2024-09-14]. Dostupné z: https://www.youtube.com/playlist?list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1

[6] UZAIR, Muhammad a Noreen JAMIL. Effects of Hidden Layers on the Efficiency of Neural networks. *IEEE Xplore* [online]. 2020, (23) [cit. 2024-09-16]. Dostupné z: https://ieeexplore.ieee.org/abstract/document/9318195

[7] BAHETI, Pragati. *Activation Functions in Neural Networks [12 Types & Use Cases]* [online]. [cit. 2024-09-16]. Dostupné z: https://www.v7labs.com/blog/neural-networks-activation-functions#h1

[8] DUBEY, Shiv Ram, Satish Kumar SINGH a Bidyut Baran CHAUDHURI. *Activation functions in deep learning: A comprehensive survey and benchmark* [online]. [cit. 2024-09-17]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0925231222008426?casa_token=3sroTuf-l4cAAAAA:_NVNeSwftuI0lAS7kCiLWBH3VH9k1RZsegbvgLcqe_xRHGgF8hX-6ZtMx2UCu3ZKZmYXLyNYt990

[9] SHARMA, Kushan. *Vanishing/Exploding Gradients Problem* [online]. 2023 [cit. 2024-09-23]. Dostupné z: https://medium.com/@kushansharma1/vanishing-exploding-gradients-problem-1901bb2db2b2

[10] HENDRYCKS, Dan a Kevin GIMPEL. *GAUSSIAN ERROR LINEAR UNITS (GELUS)* [online]. 2023 [cit. 2024-09-24]. Dostupné z: https://arxiv.org/abs/1606.08415v5

[11] Beyond ReLU: Discovering the Power of SwiGLU. *Medium.com* [online]. 2024 [cit. 2025-05-14]. Dostupné z: https://medium.com/@jiangmen28/beyond-relu-discovering-the-power-of-swiglu-%E8%B6%85%E8%B6%8A-relu-%E5%8F%91%E7%8E%B0-swiglu-%E7%9A%84%E5%8A%9B%E9%87%8F-9dbc7d8258bf

[12] Introduction to Activation Functions in Neural Networks. *Datacamp.com* [online]. [cit. 2024-09-24]. Dostupné z: https://www.datacamp.com/tutorial/introduction-to-activation-functions-in-neural-networks?dc_referrer=https%3A%2F%2Fwww.google.com%2F

[13] BELAGATTI, Pavan. *Understanding the Softmax Activation Function: A Comprehensive Guide* [online]. 2024 [cit. 2024-09-23]. Dostupné z: https://www.singlestore.com/blog/a-guide-to-softmax-activation-function/

[14] CUI, Nan. Applying Gradient Descent in Convolutional Neural Networks. *Journal of Physics: Conference Series* [online]. 2018 [cit. 2024-09-26]. Dostupné z: doi:10.1088/1742-6596/1004/1/012027

[15] PYKES, Kurtis. *Cross-Entropy Loss Function in Machine Learning: Enhancing Model Accuracy* [online]. 2024 [cit. 2024-09-28]. Dostupné z: https://www.datacamp.com/tutorial/the-cross-entropy-loss-function-in-machine-learning

[16] JORDAN, Jeremy. Setting the learning rate of your neural network. *Jeremyjordan.me* [online]. 2018 [cit. 2025-05-07]. Dostupné z: https://www.jeremyjordan.me/nn-learning-rate/

[17] *Precision vs. recall vs. accuracy in neural networks* [online]. [cit. 2024-09-30]. Dostupné z: https://www.educative.io/answers/precision-vs-recall-vs-accuracy-in-neural-networks

[18] KUNDU, Rohit. *F1 Score in Machine Learning: Intro & Calculation* [online]. 2022 [cit. 2024-10-01]. Dostupné z: https://www.v7labs.com/blog/f1-score-guide

[19] ALI, Moez. Supervised Machine Learning. *Datacamp.com* [online]. [cit. 2024-10-11]. Dostupné z: https://www.datacamp.com/blog/supervised-machine-learning

[20] NAEEM, Samreen, Aqib ALI, Sania ANAM a Munawar AHMED. *An Unsupervised Machine Learning Algorithms: Comprehensive Review* [online]. 2023 [cit. 2024-10-12]. Dostupné z: doi:10.12785/ijcds/130172

[21] Autoencoders -Machine Learning. *Geeksforgeeks.org* [online]. 2023 [cit. 2024-10-12]. Dostupné z: https://www.geeksforgeeks.org/auto-encoders/

[22] Reinforcement learning. *Geeksforgeeks.org* [online]. 2024 [cit. 2024-10-12]. Dostupné z: https://www.geeksforgeeks.org/what-is-reinforcement-learning/

[23] *An introduction to Reinforcement Learning* [online]. [cit. 2024-10-12]. Dostupné z: https://deepanshut041.github.io/Reinforcement-Learning/notes/00_Introduction_to_rl/

[24] POTHUGANTI, Swathi. Review on over-fitting and under-fitting problems in Machine Learning and solutions. *Researchgate* [online]. 2018 [cit. 2024-10-13]. Dostupné z: doi:10.15662/IJAREEIE.2018.0709015

[25] AHMED, Ryan. *What is Overfitting & Underfitting in Machine Learning?* [online]. 2021 [cit. 2024-10-13]. Dostupné z: https://www.youtube.com/watch?v=jnAeZ8j0Ur0

[26] Vanishing and Exploding Gradients Problems in Deep Learning. *Geeksforgeeks.org* [online]. 2024 [cit. 2024-10-13]. Dostupné z: https://www.geeksforgeeks.org/vanishing-and-exploding-gradients-problems-in-deep-learning/

[27] LI, Zewen, Fan LIU, Wenjie YANG, Shouheng PENG a Jun ZHOU. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. In: *IEEE Xplore* [online]. 2021, s. 6999 - 7019 [cit. 2024-10-14]. Dostupné z: doi:10.1109/TNNLS.2021.3084827

[28] STRYKER, Cole. What is a recurrent neural network? *Ibm.com* [online]. 2024 [cit. 2024-10-17]. Dostupné z: https://www.ibm.com/topics/recurrent-neural-networks

[29] HUDEC, Martin, Vladimír HABÁN a Pavel RUDOLF. *Zpráva o průběhu řešení projektu komplexní diagnostiky – část 1.3*. 2024.

[30] Acoustic Emission. *Chemeurope.com* [online]. [cit. 2025-05-14]. Dostupné z: https://www.chemeurope.com/en/encyclopedia/Acoustic_Emission.html?utm_source=chatgpt.com

[31] ONO, Kanji a Thomas ROSSING. Acoustic Emission. In: *Springer Handbook of Acoustics* [online]. 2014, s. 1209-1229 [cit. 2025-05-14]. ISBN 978-1-4939-0754-0. Dostupné z: https://www.researchgate.net/publication/302546333_Acoustic_Emission

[32] Training, Validation, Test Split for Machine Learning Datasets. *Encord.com* [online]. 2024 [cit. 2025-04-08]. Dostupné z: https://encord.com/blog/train-val-test-split/

[33] CHORNYI, Andrii. How to Choose the Right Metric for Your Model. *Codefinity.com* [online]. 2023 [cit. 2025-05-05]. Dostupné z: https://codefinity.com/blog/How-to-Choose-the-Right-Metric-for-Your-Model

[34] Tf.keras.metrics.BinaryAccuracy. *Tensorflow.org* [online]. [cit. 2025-02-05]. Dostupné z: https://www.tensorflow.org/api_docs/python/tf/keras/metrics/BinaryAccuracy

[35] NYANDWI, Jean de Dieu. Neural Networks for Classification with TensorFlow. *Nyandwi.com* [online]. [cit. 2025-02-05]. Dostupné z: https://nyandwi.com/machine_learning_complete/27_neural_networks_for_classification_with_tensorflow/

[36] MADAN, Nishant. Lecture 06 | 1D 2D 3D CNNs. *Youtube.com* [online]. 2023 [cit. 2025-04-05]. Dostupné z: https://www.youtube.com/watch?v=2YmR4iwmoTk

[37] LANE, Thom. Multi-Channel Convolutions explained with… MS Excel!. *Medium.com* [online]. 2018 [cit. 2025-04-05]. Dostupné z: https://medium.com/apache-mxnet/multi-channel-convolutions-explained-with-ms-excel-9bbf8eb77108

[38] *What could be the reason for a model predicting only one class in a multiclass classification in neural networks?* [online]. [cit. 2025-05-22]. Dostupné z: https://www.quora.com/What-could-be-the-reason-for-a-model-predicting-only-one-class-in-a-multiclass-classification-in-neural-networks

[39] *Binary Model only predicting one class* [online]. [cit. 2025-05-22]. Dostupné z: https://www.reddit.com/r/learnmachinelearning/comments/1c5j7qm/binary_model_only_predicting_one_class/

[40] KENTON, Will, Gordon SCOTT a Tomothy LI. *Kurtosis: Definition, Types, and Importance* [online]. 2024 [cit. 2025-03-29]. Dostupné z: https://www.investopedia.com/terms/k/kurtosis.asp

[41] Scipy.stats. kurtosis. *Docs.scipy.org* [online]. [cit. 2025-03-29]. Dostupné z: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kurtosis.html

[42] IBE, Oliver C. *Fundamentals of Applied Probability and Random Processes*. Elsevier, 2014. ISBN 978-0-12-800852-2.

[43] TURNEY, Shaun. Skewness | Definition, Examples & Formula. *Scribbr.com* [online]. 2023 [cit. 2025-03-29]. Dostupné z: https://www.scribbr.com/statistics/skewness/

[44] Scipy.stats. skew. *Docs.scipy.org* [online]. [cit. 2025-03-29]. Dostupné z: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.skew.html

[45] BISWAL, Avijeet. What Is Percentile in Statistics and How to Calculate it? *Simplilearn.com* [online]. 2024 [cit. 2025-03-29]. Dostupné z: https://www.simplilearn.com/tutorials/data-analytics-tutorial/percentile-in-statistics

[46] BRUNTON, Steve. *Shannon Nyquist Sampling Theorem* [online]. 2020 [cit. 2025-02-15]. Dostupné z: https://www.youtube.com/watch?v=FcXZ28BX-xE

[47] ŠEBEK, Denis a Vladimír HABÁN. *STROJOVÉ UČENÍ PŘI DIAGNOSTICE VODNÍCH STROJŮ*. Brno, 2023. Diplomová práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ, FAKULTA STROJNÍHO INŽENÝRSTVÍ.

[48] HARRIS, Frederic J. On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform. *IEEE*. 1978, 51 - 83.

[49] Source code for librosa.feature.spectral. *Librosa.org* [online]. [cit. 2025-02-15]. Dostupné z: https://librosa.org/doc/main/_modules/librosa/feature/spectral.html#melspectrogram

[50] Hann (Hanning) window. *Mathworks.com* [online]. [cit. 2025-02-15]. Dostupné z: https://www.mathworks.com/help/signal/ref/hann.html

[51] FFT and Spectrogram. *Princeton.edu* [online]. [cit. 2025-02-15]. Dostupné z: https://www.princeton.edu/~cuff/ele201/files/spectrogram.pdf

[52] FAYEK, Haytham. Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between. *Haythamfayek.com* [online]. 2016 [cit. 2025-02-15]. Dostupné z: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[53] VELARDO, Valerio. *Mel Spectrograms Explained Easily* [online]. [cit. 2025-02-16]. Dostupné z: https://www.youtube.com/watch?v=9GHCiiDLHQ4

[54] Mel Filter Bank. *Siggigue.github.io* [online]. [cit. 2025-05-21]. Dostupné z: https://siggigue.github.io/pyfilterbank/melbank.html

[55] Grid Search. *Dremio.com* [online]. [cit. 2025-04-12]. Dostupné z: https://www.dremio.com/wiki/grid-search/

[56] ZABINSKY, Zelda B. *Random Search Algorithms* [online]. In: . 2009 [cit. 2025-04-12]. Dostupné z: https://courses.washington.edu/inde510/516/AdapRandomSearch4.05.2009.pdf

[57] WANG, Wei. Bayesian Optimization Concept Explained in Layman Terms. *Medium.com* [online]. 2020 [cit. 2025-04-12]. Dostupné z: https://medium.com/data-science/bayesian-optimization-concept-explained-in-layman-terms-1d2bcdeaf12f

[58] KOSTOVČÍK, Petr a Petr LACHOUT. *Bayesovská optimalizace*. Praha, 2017. DIPLOMOVÁ PRÁCE. Univerzita Karlova, Matematicko-Fyzikální Fakulta.

[59] NOGUEIRA, Fernando. *Bayesian Optimization* [online]. 2014 [cit. 2025-04-12]. Dostupné z: https://github.com/bayesian-optimization/BayesianOptimization

[60] NAJIB, Taeef. *Hyperparameter Tuning Using Optuna* [online]. 2023 [cit. 2025-04-13]. Dostupné z: https://medium.com/@taeefnajib/hyperparameter-tuning-using-optuna-c46d7b29a3e

[61] PRIYA C, Bala. Regularization in Neural Networks. *Pinecone.io* [online]. 2023 [cit. 2025-02-12]. Dostupné z: https://www.pinecone.io/learn/regularization-in-neural-networks/

[62] RANJAN, Sumit. *What are the best practices for avoiding dropout in deep learning?* [online]. [cit. 2025-05-22]. Dostupné z: https://www.linkedin.com/advice/3/what-best-practices-avoiding-dropout-deep-learning-86fwe

Seznam použitých veličin

i,j	[1]	indexy	
x	[-]	obecný argument matematické funkce	v rovnicích v kapitole1
z	[-]	vstupní vektor aktivační funkce	v rovnicích v kapitole 1
Z	[1]	celkový počet prvků ve vektoru z	v rovnici (1-7)
m	[1]	počet datových vzorků	v rovnicích v kapitole 1
n	[1]	počet vstupních argumentů do hypotézy	v rovnici (1-8)
y	[1]	target datového vzorku	v rovnicích v kapitole 1
p	[1]	výstup neuronové sítě interpretovaný jako pravděpodobnost	
w	[1]	hodnota váhy v modelu	v rovnicích v kapitole 1
x	[-]	konkrétní hodnota ze segmentu dat	v rovnicích (4-2), (4-3)
n	[1]	počet hodnot v segmentu dat	v rovnici (4-2)
K	[1]	Fisherův kurtosis	v rovnici (4-2)
Sk	[1]	šikmost	v rovnici (4-3)
m_x	[-]	x-té centrální momenty	v rovnicích (4-3)
N	[1]	celkový počet prvků okně/segmentu dat	v rovnicích (4-3), (4-4)
n	[1]	index datového vzorku v rámci okna	v rovnici (4-4)
$x(f)$	[-]	frekvenční signál	v rovnici (4-5)
$x(t)$	[-]	časový signál	v rovnici (4-5)
t	[s]	čas	v rovnici (4-5)
j	[1]	imaginární konstanta	v rovnici (4-5)
$x\hat{}(k,m)$	[-]	frekvenční spektrum z diskrétní FFT	v rovnici (4-6)
k	[1]	index frekvence	v rovnici (4-6), (4-7)
$P(k,m)$	[-]	výkonové spektrum	v rovnici (4-7)
f_{mel}	[mel]	frekvence v MEL stupnici	v rovnicích (4-8), (4-9)
f	[Hz]	frekvence v Hertzově stupnici	v rovnicích (4-5), (4-8), (4-9)
y	[1]	hodnota cílové funkce	v rovnici (4-11)
y^*	[1]	dosavadní nejlepší hodnota cílové funkce	v rovnici (4-11)
$p_M(y\|x)$	[1]	pravděpodobnostní model	v rovnici (4-11)

Označení jednotky [-] značí, že konkrétní jednotka se odvíjí od konkrétního použití a zpravidla souvisí s jednotkou zpracovávaných dat.

Seznam příloh

1 Přílohy1-6
   a. Příloha 1 – Průběhy učení při použití MEL-spektrogramů a veškerých dat
   b. Příloha 2 – Průběhy učení při použití MEL-spektrogramů – polovina souborů
   c. Příloha 3 – Průběhy učení při použití MEL-spektrogramů – třetina souborů
   d. Příloha 4 – Průběhy učení při využití MEL-spektrogramů v turbínovém režimu
   e. Příloha 5 – Průběhy učení při použití úprav MEL-spektrogramů
   f. Příloha 6 – Automatické reporty z Optuny
2 Příloha 7 – Python programy