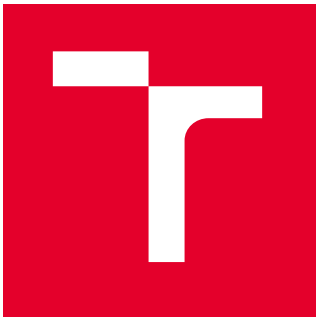


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ENERGETICKÝ ÚSTAV

ENERGY INSTITUTE

STROJOVÉ UČENÍ PŘI DIAGNOSTICE VODNÍCH STROJŮ

MACHINE LEARNING IN WATER MACHINE DIAGNOSTICS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — **Bc. Denis Šebek**

AUTHOR

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

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Energetický ústav
Student:	**Bc. Denis Šebek**
Studijní program:	Energetické a termofluidní inženýrství
Studijní obor:	Fluidní inženýrství
Vedoucí práce:	**doc. Ing. Vladimír Habán, Ph.D.**
Akademický rok:	2022/23

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:

Strojové učení při diagnostice vodních strojů

Stručná charakteristika problematiky úkolu:

Zvuk vyzařovaný hydraulickým strojem snímaný pomocí snímačů akustické emise mikrofonu, nebo pomoci vysokofrekvenčních piezoelektrických smímačů tlaku indikuje provozní stav stroje. S takto naměřených dat, by mělo být možno stanovit průtok, kavitaci případně indikovat poruchu na hydraulickém stroji.

Cíle diplomové práce:

Pomocí nestandartních vysokofrekvenčních měření (zvuku, akustická emise, piezoelektrické snímače tlaku, ….) stanovit provozní stavy vodního stroje. Identifikovat poruchu, kavitaci, průtok. Případně odhadnou možnosti těchto nestandartních metod a jejich 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.

ABSTRAKT

Práca sa zaoberá využitím vysokofrekvenčných metód merania veličín ako tlak, zvuk a akustická emisia a ich aplikovania do strojného učenia pre detekciu preváckových sta- vov vodných strojov. Úlohou tejto práce je navrhnúť spôsob strojného učenia, ktorý by dokázal z nameraných dát vyhodnotiť stavy kavitácie, prietoku a poruchy na testovacej trati. Prvá časť práce sa zaoberá teoretickým úvodom niektorých vysokofrekvenčných metód merania a základným úvodom strojného učenia. V druhej časti je popísaný samotný experiment spolu s vysvetlením úpravy vstupných dát a naprogramovaním neurónovej siete. V závere sú jednotlivé výsletky zhodnotené a porovnané.

KĽÚČOVÉ SLOVÁ

Strojné učenie, Neurónové siete, Kavitácia, Prietok, Tlak, Zvuk, Akustická emisia

ABSTRACT

The thesis focuses on utilizing high-frequency measurement methods for measuring pressure, sound, and acoustic emission, and their application in machine learning for detecting operational states in water machinery. This thesis aims to propose a machine-learning approach capable of evaluating cavitation, flow conditions, and faults based on the collected data.

The first part of the thesis discusses the theoretical introduction to selected high-frequency measurement methods and provides a basic introduction to machine learning. The second part describes the experimental setup, including data preprocessing and the implementation of a neural network. In the conclusion, the individual results are evaluated and compared.

KEYWORDS

Machine learning, Neural network, Cavitation, Fluid flow, Pressure, Acoustic emission

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. Křestní Příjmení, Ph.D.

Vyhlásenie autora o pôvodnosti diela

Meno a priezvisko autora:	Bc. Denis Šebek
VUT ID autora:	201611
Typ práce:	Diplomová práca
Akademický rok:	2023/24
Téma závěrečnej práce:	Strojové učení při diagnostice vodních strojů

Vyhlasujem, že svoju záverečnú prácu som vypracoval samostatne pod vedením vedúcej/cého záverečnej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej záverečnej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto záverečnej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podpisuje iba v tlačenej verzii.

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce doc. Ing. Vladimírovi Habánovy, Ph.D. za odborné vedenie, cenné rady a pripomienky, a rýchlu odozvu na akékoľvek otázky. Tak isto ďakujem ľuďom, ktorí mi pomohli s gramatickou a štylistickou úpravou.

Obsah

Úvod **21**

Ciele práce **23**

1 Vysokofrekvenčné spôsoby merania **25**
1.1 Kapacitné mikrofóny 25
1.2 Akustická emisia 27
1.3 Piezoelektrické snímače tlaku 29

2 Strojné učenie **31**
2.1 Učenie pod dohľadom 31
2.1.1 Lineárna regresia 32
2.1.2 Logistická regresia 33
2.1.3 Rozhodovacie stromy 34
2.1.4 HĺBoké učenie 35
2.2 Učenie bez dohľadu 36
2.2.1 Zhlukovanie 36
2.3 Učenie s posilneným učením 37

3 Experiment **39**
3.1 Meranie dát 39
3.2 Spracovanie dát 44
3.3 Príprava datasetov 48

4 Konvulenčná neuronová sieť **51**
4.1 Vrstvy 51
4.1.1 Konvolučná vrstva 51
4.1.2 Združovacia vrstva 52
4.1.3 Plne prepojená vrstva 53
4.2 Aktivačné funkcie 54
4.2.1 Sigmoid a Logistická aktivačná funkcia 54
4.2.2 Tanh aktivačná funkcia 55
4.2.3 ReLu aktivačná funkcia 55
4.2.4 Leak ReLU 56
4.3 Stratové funkcie 56
4.4 Architektúra CNN 57
4.5 Aplikácia strojného učenia 62
4.5.1 Klasifikácia kavitácie 62

4.5.2 Klasifikácia prietoku 71

Závěr **79**

Literatúra **83**

Zoznam symbolov a skratiek **89**

Zoznam príloh **91**

A Tvorenie datasetu pre binárnu klasifikáciu zo surových dát **93**

B Vytvorenie a uloženie spektografov **95**

C Tvorenie datasetov pre klasifikáciu prietokov zo surových dár **97**

D Tvorenie datasetov pre klasifikáciu prietokov a kavitácie za pomoci spektografov **101**

E CNN pre binárnu klasifikáciu kavitácie zo surových dát **103**

F CNN pre klasifikácu prietokov zo surových dát **107**

G CNN pre binárnu klasifikáciu kavitácie za pomoci spektografov **111**

H CNN pre klasifikáciu prietokov za pomoci spektografov **115**

Zoznam obrázkov

1.1 Schéma kondenzátorového mikrofónu [2] . . . 25
1.2 Základné schéma kapacitného mikrofónu [3] . . . 26
1.3 Schéma elektretového mikrofónu [3] . . . 27
1.4 Akustická emisia [5] . . . 27
1.5 Schéma piezo-elektrického snímača tlaku [7] . . . 29
1.6 Piezoelektrický snimač: a) Prevod sily na napätie, b) Zosilovač elektrického signálu [6] . . . 30
2.1 Schéma procesu učenia pod dohľadom [10] . . . 31
2.2 Príklad lineárnej regresie [12] . . . 32
2.3 Logistická regresia [13] . . . 33
2.4 Príklad rozhodovcieho stromu [15] . . . 34
2.5 Schéma neurónovej siete [17] . . . 35
2.6 Učenie bez dohľadu [18] . . . 36
2.7 Zhlukovanie [19] . . . 36
2.8 Posilnené učenie [21] . . . 37
3.1 Sledovaný úsek meriacej trate . . . 39
3.2 Rozmery lopatky ČKD . . . 40
3.3 Piezoelektrický snímač tlaku Kistler typu 211B4 s tlakovým rozsahom 200 psi [23] . . . 40
3.4 Piezoelektrický mikrofón od PCB typ 130A24 [24] . . . 40
3.5 Magneticko indukčný prietokomer firmy KROHNE typ 4100 v rozsahu 3000 m^3/hod [25] . . . 41
3.6 Snímač akustickej emisie firmy DAKEL typ MDK13AS [26] . . . 41
3.7 Záznam z vysokorýchlostnej kamery . . . 43
3.8 Záznam prietokov pre 0 - 30 kPa . . . 43
3.9 Príklad spektografu . . . 46
3.10 Porovnanie spektografov pred filtráciou (vľavo) a po filtrácii (vpravo) 47
3.11 Graf znázorňujúci hranice kavitácie pre jednotlivé pracovné body . . 48
4.1 Príklad konvulenčnej neurónovej siete [28] . . . 51
4.2 Filter CNN [30] . . . 52
4.3 Max a average pooling [31] . . . 52
4.4 Plne prepojená vrstva [32] . . . 53
4.5 Sigmoidova krivka [34] . . . 54
4.6 Funkcia tanh [35] . . . 55
4.7 Funkcia ReLU [36] . . . 55
4.8 Porovnanie funkcií ReLU [37] . . . 56
4.9 Architektúra VGG16 [39] . . . 57

4.10 Vplyv veľkosti learning rate na nájdenie minima funkcie [40] 58
4.11 Architektúra CNN pre binárnu klasifikáciu zo segmetovaných dát . . 59
4.12 Architektúra CNN pre prípad klasifikácie prietokov za pomoci segmentovaných dát 60
4.13 Architektúra CNN pre prípad klasifikácie kavitácie za pomoci spektografov 61
4.14 Presnosť a strata snímača akustickej emisie pre binárnu klasifikáciu zo segmentovaných dát 62
4.15 Presnosť a strata mikrofónu pre binárnu klasifikáciu zo segmentovaných dát 63
4.16 Presnosť a strata dynamického snímača tlaku pre binárnu klasifikáciu zo segmentovaných dát 64
4.17 Klasifikačné reporty CNN binárnej klasifikácie segmentovaných dát . 65
4.18 Akustická emisia - Kavituje 67
4.19 Akustická emisia - Nekavituje 67
4.20 Kisler - Kavituje 67
4.21 Kisler - Nekavituje 67
4.22 Mikrofón - Kavituje 67
4.23 Mikrofón - Nekavituje 67
4.24 Porovnanie Spektografov Kavituje vs Nekavituje pre všetky 3 senzory 67
4.25 Presnosť a strata snímača akustistej emisie pre binárnu klasifikáciu zo spektografov 68
4.26 Presnosť a strata mikrofónu pre binárnu klasifikáciu zo spektografov . 68
4.27 Presnosť a strata dynamického snímača tlaku pre binárnu klasifikáciu zo spektografov 69
4.28 Klasifikačné reporty CNN binárnej klasifikáciu zo spektografov . . . 70
4.29 Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát 71
4.30 Presnosť a strata dynamického snímača tlaku pre klasifikáciu prietoku zo segmentovaných dát 71
4.31 Presnosť a strata mikrofónu pre klasifikáciu prietoku zo segmentovaných dát 72
4.32 Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát 150 epóch 72
4.33 Nastavenie neurónovej siete pre klasifikáciu prietoku zo spektografov . 74
4.34 Presnosť a strata mikrofónu pre klasifikáciu prietoku zo spektografov 75
4.35 Presnosť a strata dynamického snímača tlaku pre klasifikáciu prietoku zo spektografov 75

4.36 Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát 76
4.37 Presnosť a strata mikrofónu pre klasifikáciu prietoku zo segmentovaných dát (Redukované) 77

Zoznam tabuliek

3.1 Počet dátových záznamov pre jednotlivé pracovné body 42
3.2 Segmentácia dát 44
4.1 Výstupy CNN pre binárnu detekciu kavitácie za pomoci segmentovaných dát pre všetky sensory po 35 epochách 64
4.2 Výstupy CNN pre binárnu detekciu kavitácie za pomoci spektografov pre všetky sensory na koci poslednej epochy. 70
4.3 Porovnanie všetkých snímačov pre klasifikáciu kavitácie 79
4.4 Porovnanie f1 Skóre pre binárnu klasifikáciu kavitácie (0 - Kavituje, 1 - Nekavituje) 80
4.5 Zoznam symbolov a skratiek 90

Úvod

Strojové učenie sa stalo jedným z najdôležitejších a rýchlo sa rozvíjajúcich oblastí v súčasnej dobe. Jeho schopnosť analyzovať veľké objemy dát a odhaľovať skryté vzory umožňuje aplikácie v rôznych priemyselných odvetviach. Jednou z týchto oblastí, ktorá profituje z vývoja strojového učenia, je diagnostika vodných strojov.

Vodné stroje, ako napríklad turbíny, alebo čerpadlá, sú kľúčovými zariadeniami v energetickom sektore a ďalších priemyselných odvetviach. Ich spoľahlivá a efektívna prevádzka je kritická pre zabezpečenie trvalo udržiavaného fungovania a minimalizáciu nákladov spojených s údržbou a opravami. Avšak, neustále monitorovanie a diagnostika vodných strojov je náročná úloha, nakoľko tieto zariadenia sú vystavené rôznym degradáciám a poruchám, ktoré môžu viesť k ich nefunkčnosti alebo zlyhaniu.

Cieľom tejto diplomovej práce je preskúmať možnosti aplikácie strojového učenia pre diagnostiku vodných strojov. Konkrétne, práca sa zameria na vývoj a implementáciu algoritmov strojového učenia, ktoré sú schopné analyzovať dáta zo senzorov, a následne identifikovať príznaky degradácie alebo poruchy vodných strojov. Tieto algoritmy by mali umožniť automatickú detekciu a klasifikáciu anomálií a predikciu budúcich stavov zariadení na základe historických dát.

V rámci práce budeme venovať pozornosť aj výberu relevantných dátových atribútov, spracovaniu dát a trénovaniu strojových modelov. Praktická časť práce bude realizovaná na reálnych dátach získaných zo senzorov vodných strojov a výsledky budú zhodnotené a porovnané s existujúcimi metódami diagnostiky.

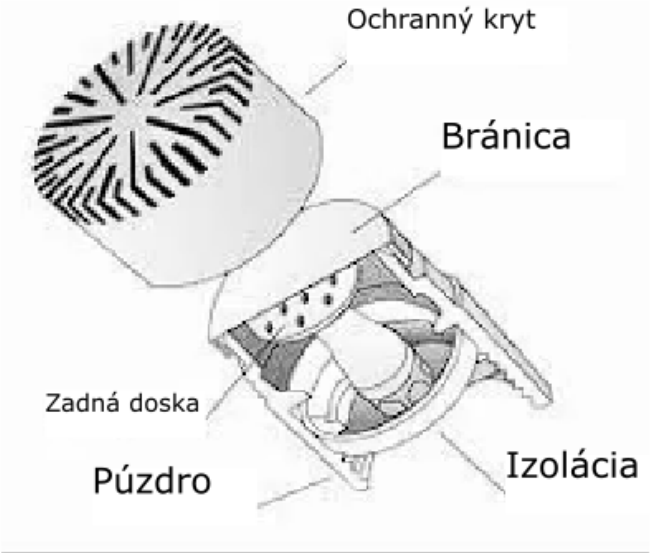
Ciele práce

Za ciele tejto práce považujeme hlavne pomoc strojného učenia na stanovenie prevádzkových stavov vodného stroja, z dát získaných za pomoci vysokofrekvenčních spôsobov merania tlaku, zvuku a akustickej emisie, a taktiež stanovenie nedostatkov týchto metód. Pod prevádzkovými stavmi sú myslené stavy ako kavitácia, prietok a porucha stroja.

1 Vysokofrekvenčné spôsoby merania

1.1 Kapacitné mikrofóny

Mikrofón je akusticko-mechanicko-elektrický menič, ktorým sa sníma kmitanie vonkajšieho prostredia (plynu) a prevádzame ho na elektrický signál. V praxi sa rozoznáva celá rada typov mikrofónov, ktoré dokážu premieňať akustický signál na mechanický či elektrický. Aj napriek tomu sa dnes používajú v technike recipročné meniče (dynamické, elektrostatické, piezoelektrické atď.). Pri serióznom meraní hluku sa v súčasnosti používajú meniče; elektrostaticko-kapacitné mikrofóny [1].



Obr. 1.1: Schéma kondenzátorového mikrofónu [2]

Funkcia týchto snimačov je založená na dvoch základných rovniciach pre kapacitu paralelných dosiek [3].

$$C = \frac{\epsilon A}{d} \tag{1.1}$$

Kde:

- **C** → Kapacita
- ϵ → Permitivita
- **A** → Plocha dosiek
- **d** → Vzdialenosť medzi doskami

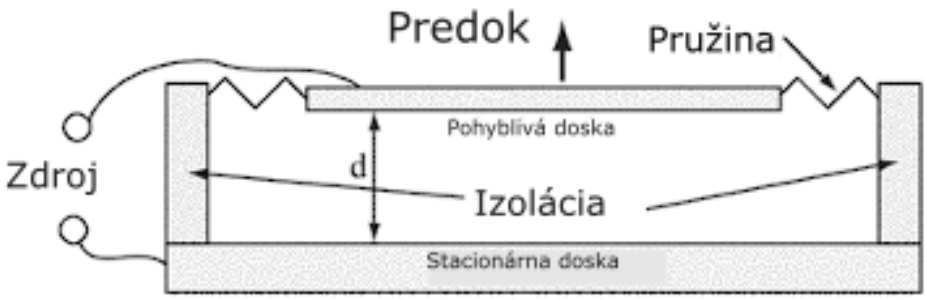
$$C = \frac{Q}{V} \tag{1.2}$$

Kde:

- **C** → Kapacita
- **Q** → Náboj uložený na doskách
- **V** → Napätie na kondenzátore

Kombináciou týchto rovníc je nasledovný vzťah:

$$V = Q\frac{d}{\epsilon A} \tag{1.3}$$



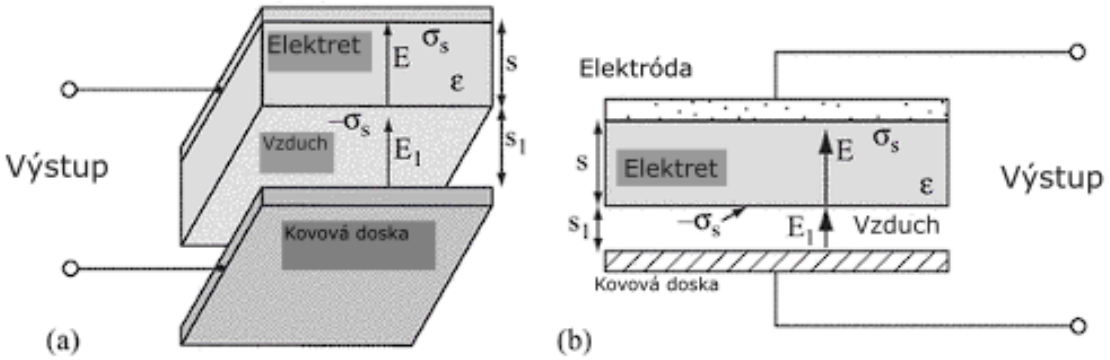
Obr. 1.2: Základné schéma kapacitného mikrofónu [3]

Variáciou týchto mikrofónov sú tzv. elektretové mikrofóny. Sú to kapacitné mikrofóny, ktoré tak isto obsahujú 2 paralelné dosky, ale s vrstvou elektretového materiálu pod vrchnou doskou, ktorý je vyrobený z tenkého materiálu, aby bola umožnená jeho pružnosť. Elektrotový materiál vytvára náboj na ploche vrchnej kapacitnej dosky a záporný náboj na spodnej doske. Toto vygeneruje elektrické pole medzi nimi. Napätie medzi nimi bez akéhokoľvek vonkajšieho stimulu sa dá popísať nasledovnou rovnicou [3].

$$V = \frac{\sigma_s s_1}{\epsilon_0 s + \epsilon s_1} \tag{1.4}$$

Kde:

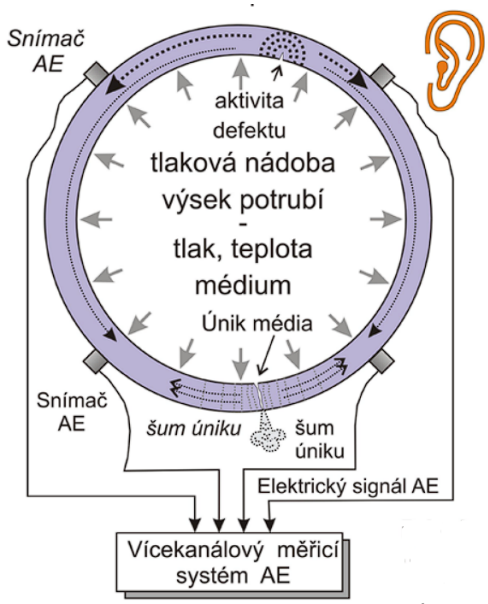
- **V** → Napätie na doskách
- σ_s → Povrchová hustota náboja
- ϵ_0 → Permitivita volného priestoru
- ϵ → Permitivita elektretového materiálu
- **s** → Hrúbka elektretu
- s_1 → Hrúbka voľného priestoru



Obr. 1.3: Schéma elektretového mikrofónu [3]

1.2 Akustická emisia

Pod pojmom akustická emisia sa rozumie fyzikálny jav, kedy plastickou deformáciou kovov dochádza ku akustickému popraskaniu, či akustickému šumu. S týmto fyzikálnym javom sa je možné stretnúť v bežnom živote pomerne často, napr. pri praskaní ľadu na jazere. Merajú sa teda elastické napäťové vlny generované dynamickým uvoľnením mechanického napätia vnútri tělesa, alebo procesom pôsobiacim na vznik elastických napäťových vĺn na povrchu telesa [4].



Obr. 1.4: Akustická emisia [5]

Metóda akustickej emisie sa nazýva metódu detekcie javu, mechanizmu, procesu, alebo zdroja akustickej emisie, následné spracovanie detekovaného signálu a konečne vyhodnotenie parametrov detekovaného signálu. Zdrojom akustiských emisií je akýkoľvek fyzikálny jav, ktorý môže vyvolať v telese vznik elastických napäťových vĺn. Od nespojitých poskokov dislokácií pri plastických deformáciách, až po lom nekovových zložiek štruktúry materiálu. V prípade vodných strojov môže byť týmto javom kavitácia. Aplikácia akustickej emisie je tak isto rôznorodá, ako zdroje akustickej emisie [4].

Vo všeobecnosti sa rozoznávajú 2 tipy zdrojov:

1. **Primárne cieľové zdroje** → na ich detekciu je priamo zameraná akustická emisia
2. **Sekundárne rušivé zdroje** → aplikácia týchto zdrojov ruší, alebo prekrýva detekciu primárnych zdrojov

Ďalším dôležitým pojmom je detekovatelnosť cieľových primárnych zdrojov. Akustická emisia (ďalej definované už len ako AE), detekuje vo frekvenčnom pásme približne od jednotiek kHz do 1 MHz. Pokiaľ sú primárne zdroje pod prahom detekcie akustickej emisie, potom táto metóda stráca pre danú aplikáciu zmysel. Detekovateľnosť zdroja je závislá na mnohých faktoroch, od intenzity zdroja, cez pokles signálu pri jeho šírení od zdroja k snímaču, až po citlivosť detekcie meracej trasy. Spolu s intenzitou zdroja AE a jeho frekvenčného obsahu ovplyvňuje detekovateľnosť aj vplyv šírenia vĺn AE od zdroja ku snímaču, za príčiny znižovania hustoty energie. Disperzia energie pôvodne koncetrovanej v ostrej vlnovej ploche, v ktorej sa každá zložka spektra pulzu šíri inou rýchlosťou spôsobí pôvodne ostrá plocha sa postupne začne šíriť ako balík rôznych odrážaných lúčov. Ďalším dôvodom je zmena mechanickej energie na energiu tepelnú [4].

Vplyvom týchto faktorov je meraný signál veľmi závislý na vzdialenosti snímača od zdroja AE. Ak je v laboratórnych podmienkach vzdialenosť od zdroja k snímaču 0.05m a v praxi 3 m, potom rozdiel v amplitudách môže byť 10-100 násobný [4].

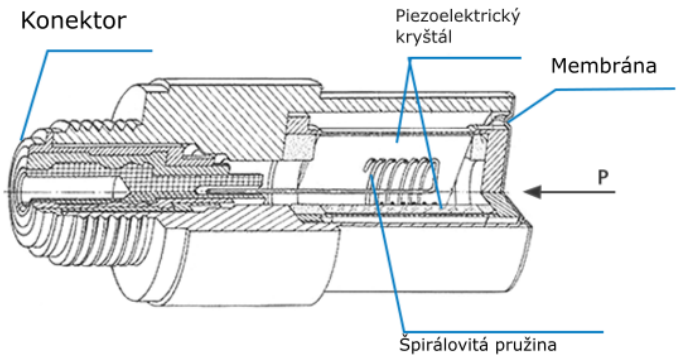
1.3 Piezoelektrické snímače tlaku

Piezoelektrické snímače tlaku sú tzv. priame, aktívne snímače, ktoré pracujú na piezoelektrickom jave. Tento jav spočíva v tom, že v niektorých dielektrikách vzniká pôsobením mechanického namáhania elektrická polarizácia. Piezoelektrický jav sa dá opísať ako lineárna elektromechanická interakcia v materiále, ktorý nemá stred symetrie pre tlakové senzory založené na priamom piezoelektrickom efekte, ktorý môže byť definovaný nasledovne [6].

$$D = \epsilon \times E + d \times X \tag{1.5}$$

Kde:

- D → Indukovaný elektrický posun
- E → Aplikované elektrické pole
- X → Mechanické namáhanie aplikované na materiál
- ϵ_p → Dielektrická konštanta
- d_p → Piezoelektrický koaficient



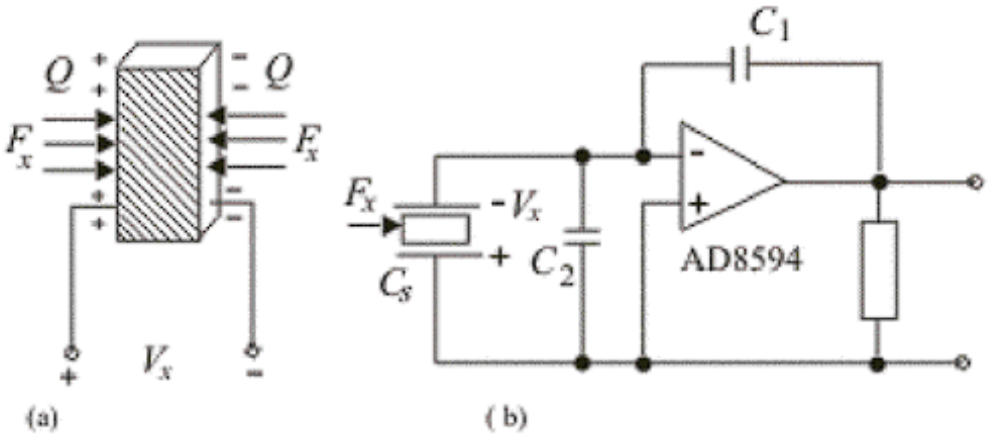
Obr. 1.5: Schéma piezo-elektrického snímača tlaku [7]

Meraný tlak sa privádza cez oddeľovaciu membránu na piezoelekrickú dvojicu kryštálov vyrobených najčastejšie z kremíka alebo turmalínu. Vplyvom mechanickej deformácie vzniká v kryštály elektrická polarizácia, a jej dôsledkom sa na povrchu vytvárajú zdanlivé náboje, ktoré v priložených elektródach uvoľňujú skutočné náboje [6].

Medzi hlavné výhody piezoelektrických snímačov patria [6]:

- Široké spektrum merania
- Vysoké vlastné frekvencie
- Vysoká linearita medzi výstupným signálom a aplikovaným zaťažením
- Široké spektrum preváckových teplôt
- Necitlivita na elektrické a magnetické polia

Jednou z nevýhod týchto snímačov sú ťažkosti pri meraní statiky. Jedná sa skoro o úplne dynamické snímače, nakoľko reagujú iba na zmeny zaťaženia. Pre akúkoľvek dátovú akvizíciu a analýzu signálu je potrebné tento výstup konvertovať na napäťový výstup, prípadne na zosilovač [6].



Obr. 1.6: Piezoelektrický snimač: a) Prevod sily na napätie, b) Zosilovač elektrického signálu [6]

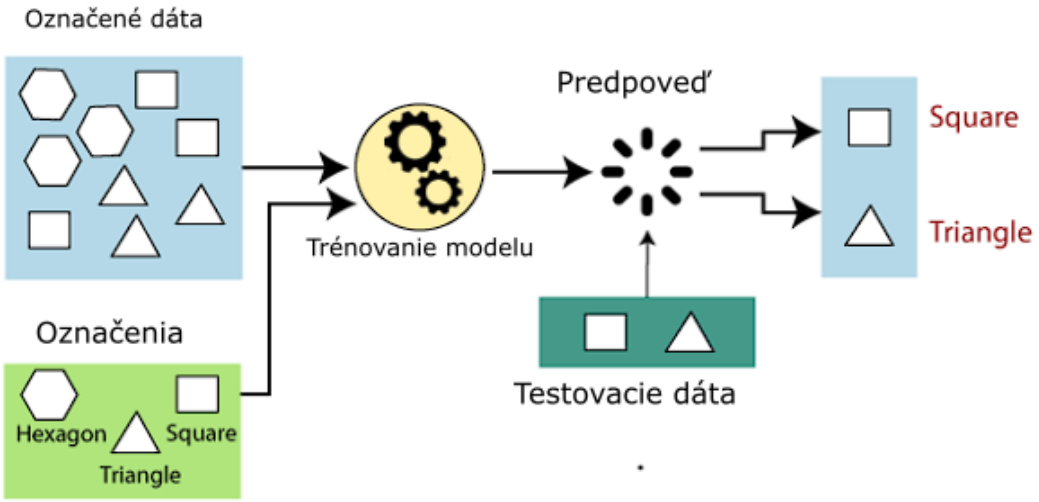
2 Strojné učenie

Pod pojmom strojné učenie sa rozumie súbor algoritmov navrhnutých tak, aby sa snažili napodobniť ľudské myslenie a hlavne jeho schopnosť učiť sa z jeho okolitého prostredia. Techniky založené na strojnom učení sú aplikované v širokom spektre polí, od rozoznania vzorov, cez počítačové videnie, vesmírne inžinierstvo, financie, služby, biológiu, až po biomedicínu a lekárstvo. Vo všeobecnosti, strojné učenie sa snaží naučiť predpovedať (aproximovať), označenie týchto dát, ktoré je závislé na vlastnostiach týchto dát. Táto predpoveď sa získava ako funkčná hodnota tzv. mapy hypotéz, ktorej vstupné argumenty sú vlastnosti dátového bodu. Podľa toho, ako metódy strojného učenia hodnotia kvalitu týchto máp, rozlišujeme 3 hlavné metódy strojného učenia [8].

1. **Učenie pod dohľadom**
2. **Učenie bez dohľadu**
3. **Učenie s posilneným učením**

2.1 Učenie pod dohľadom

Metódy učenia pod dohľadom používajú trénovací súbor údajov, ktorý obsahuje označené dáta ako vstupy, a tak isto požadované výstupy. Takýmto dátam sa hovorí označené dáta, pokiaľ ich označenie poznáme. Tieto údaje sú získané z rozličných zdrojov a sú označené pomocou ľudského vstupu. Preto sa týmto metódam hovorí učenie pod dohľadom, nakoľko vyžadujú ľudmi označené vstupné dáta [9].



Obr. 2.1: Schéma procesu učenia pod dohľadom [10]

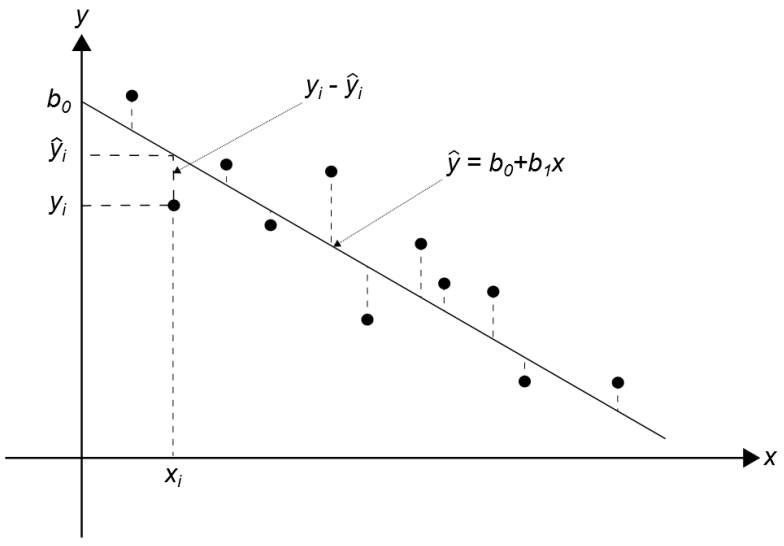
Pri metódach učenia pod dohľadom sa učia hypotézy so základným princípom minimálnej nesúladosti medzi predpoveďou a správnym označením dátového záznamu trénovacieho datasetu. Tieto metódy sa snažia nájsť krivku pre označenie dátových záznamov trénovacieho datasetu. Pre vyhodnotenie takejto krivky je za potrebné stanoviť stratovú funkciu, ktorá kvantifikuje chybu tejto krivky. Komplikácia týchto metód je v spracovaní obrovských množstiev dátových záznamov, každý s potenciálne viacerými označeniami. Ďalšou reštrikciou je, že sa snažia nájsť veľmi nelineárnu funkciu pre "napasovanie" na trénovacie dátové záznamy [8].

Do algoritmov, ktoré spadajú do kategórie učenie pod dohľadom je mnoho a každý má svoje výhody a nevýhody.

2.1.1 Lineárna regresia

Jednoduchý algoritmus používaný pre regresné problémy, kde cieľom je predpovedať spojitú vstupnú premennú. Jedná sa o starú metódu, ale jej používanie môžeme sledovať dodnes aj v komplikovanejších algoritmoch [11].

Uvažuje sa datový zápis, ktorý je charakterizovaný vektorom vlastností $\mathbf{y}$ a numerických označení $\mathbf{x}$. Lineárna regreia sa snaží naučiť hypotézu z lineárneho priestoru hypotéz a predpovedať $\hat{y}$, ktorý by minimalizoval stratu štvorcovej chyby od vstupných dát. [8].



Obr. 2.2: Príklad lineárnej regresie [12]

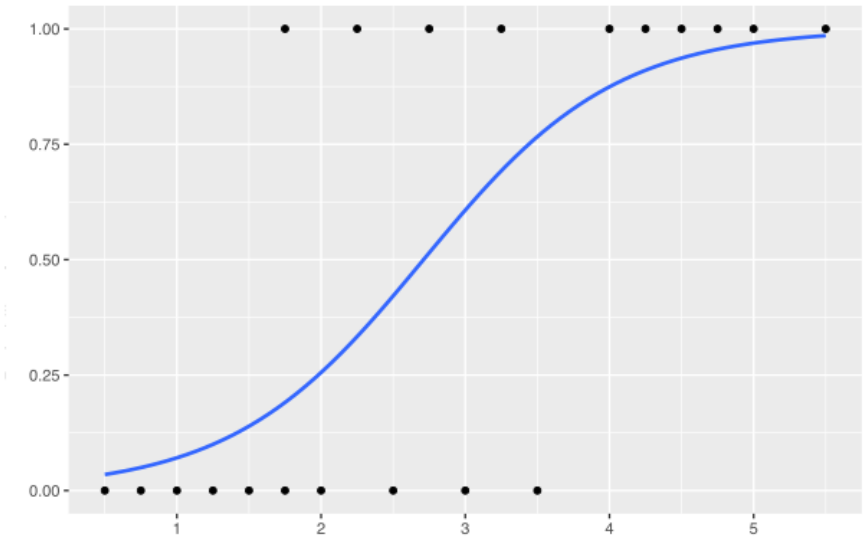
$$\hat{y} = \arg\min_{h \in H(n)} \frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h(x^{(i)}))^2 \tag{2.1}$$

Kde:

- **H** → Priestor hypotéz
- **y** → Reálna hodnota výstupu
- **h** → Predpovedaná hodnota výstupu
- **m** → Počet príkladov
- **argmin** → Matematický výraz špecifikujúci že hladáme h, ktoré minimalizuje hodnotu v zátvorkách

2.1.2 Logistická regresia

Ako pri lineárnej regresii tak aj logistická regresia je metóda, ktorá klasifikuje dátové záznamy, ktoré sú charakterizované vektorom vlastností podľa 2 kategórií, ktoré sú zakódované označením **y**. Logistická regresia sa učí hypotézy z priestoru hypotéz tak isto ako u lineárnej regresie. Logistická regresia využíva logistickú stratu pre vyhodnotenie špecifickej hypotézy. Snaží sa minimalizovať empirický risk (priemerná logistická strata) [8].



Obr. 2.3: Logistická regresia [13]

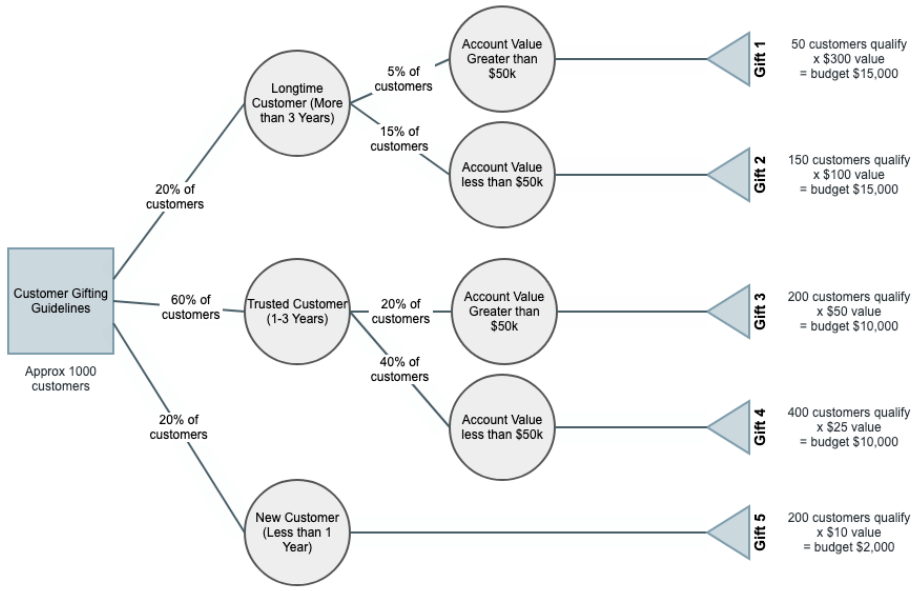
$$\hat{L}(w|D) = \frac{1}{m}\sum_{i=1}^{m}\log(1+\exp(-y^i h^w(x^i))) \tag{2.2}$$

Kde:

- $\hat{L} \rightarrow$ Logistická stratová funkcia
- $\mathbf{w} \rightarrow$ Parametre naučené pri trénovaní
- $\mathbf{D} \rightarrow$ Trénovací data-set
- $\mathbf{y} \rightarrow$ Reálna hodnota výstupu
- $\mathbf{h} \rightarrow$ Predpovedaná hodnota výstupu
- $\mathbf{m} \rightarrow$ Počet príkladov

2.1.3 Rozhodovacie stromy

Rozhodovacie stromy sú v podstate popisy máp podobné vývojovým diagramom $h : X \rightarrow Y$, ktoré mapujú vlastnosti dátových záznamov za účelom predpovedania označenia h [14].



Obr. 2.4: Príklad rozhodovcieho stromu [15]

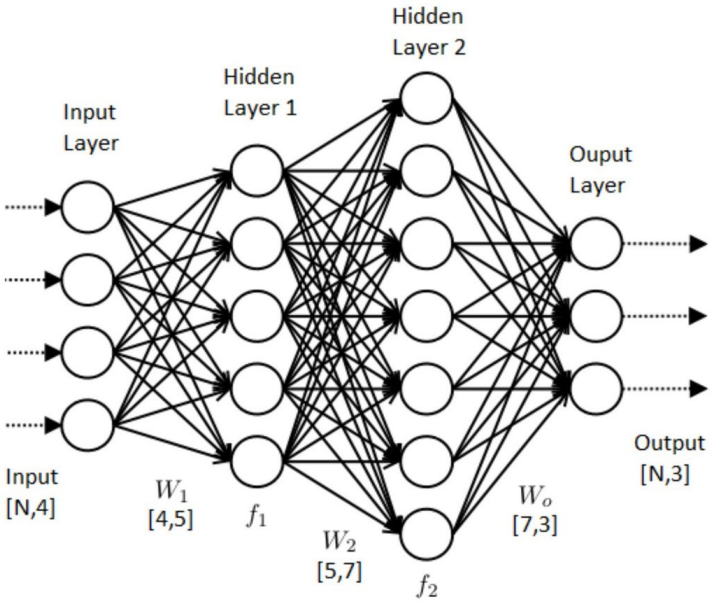
Rozhodovací strom sa skladá z uzlov, ktoré sú prepojené za pomoci smerovaných hrán. O strome rozhodnutí sa dá uvažovať ako o súbore po sebe idúcich inštrukcií pre výpočet funkčnej hodnoty $h(x)$ pre zadané vlastnosti dátového záznamu. Tento výpočet začína v koreňovom uzly a končí v jednom z tzv. listových uzlov. Listový uzol, ktorý neobsahuje žiadne smerované hrany reprezentuje rozhodovaciu zónu v priestore vlastností [16].

Vo všeobecnosti rozhodovacie stromy obsahujú 2 druhy uzlov:

- **Rozhodovacie (testovacie) uzly.** Tieto uzly reprezentujú špecifické testy o vektore vlastností **x** napr.: "Je skalárny súčin **x** väčší ako 10?"
- **Listové uzly**, ktoré zodpovedajú podmnožinám priestoru prvkov

2.1.4 Hĺboké učenie

Jedným z ďalších príkladov priestoru hypotéz, sú tzv. umelé neurónové siete. Vektor vlastností $\mathbf{x} \in R^n$ je vložený do vstupných jednotiek, kde každá číta jednu vlastnosť vstupného vektoru **x**. Tieto vlastnosti sú následne multiplikované váhou w_j, ktorá sa vzťahuje prepojením medzi vstupným neurónom a strednou vrstvou (skrytou) [17].



Obr. 2.5: Schéma neurónovej siete [17]

Výstup z tejto strednej vrstvy je daný [17]:

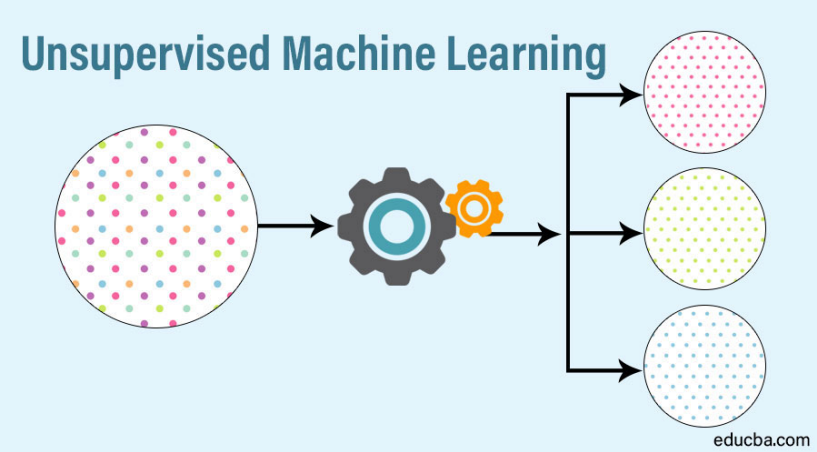
$$s_j = g\left(\sum_{n_j=1}^{n_j} w_{j,n_j} x_{n_j}\right) \tag{2.3}$$

Kde:

- $s_j \rightarrow$ Výstup z neurónu skrytej vrstvy
- $\mathbf{g} \rightarrow$ Aktivačná funkcia
- $n_j \rightarrow$ Počet neurónov v predchádzajúcej vrstve prepojených do neurónu skrytej vtstvy
- $w_j \rightarrow$ Váha medzi dvomi spojenými vrstvami
- $x_j \rightarrow$ Výstup z predchádzajúcej vrstvy

2.2 Učenie bez dohľadu

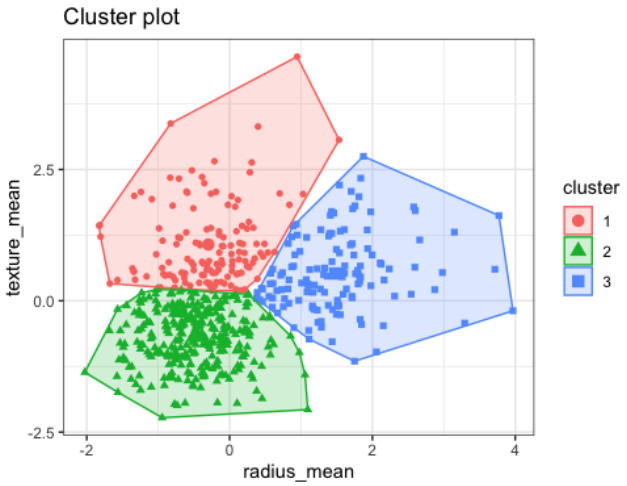
Niektoré metódy strojného učenia nevyžadujú poznať označenie dátových záznamov, a preto sa tieto metódy nazývajú; metódy učenia bez dohľadu. Tieto metódy sa musia spoliehať na vnútornú štruktúru dátových záznamov, aby sa naučili dobrú hypotézu. Na základe toho teda metódy bez dozoru nepotrebujú dedikovanú osobu, ktorá by im poskytovala označené dáta [8].



Obr. 2.6: Učenie bez dohľadu [18]

2.2.1 Zhlukovanie

Jeden z algorytmov učenia bez dohľadu je tzv. zhlukovanie, ktorého cieľom je nájsť klaster vstupných dát. Sleduje tieto vstupy, a podľa ich vlastností ich rozdeľuje do jednotlivých skupín - zhlukov.

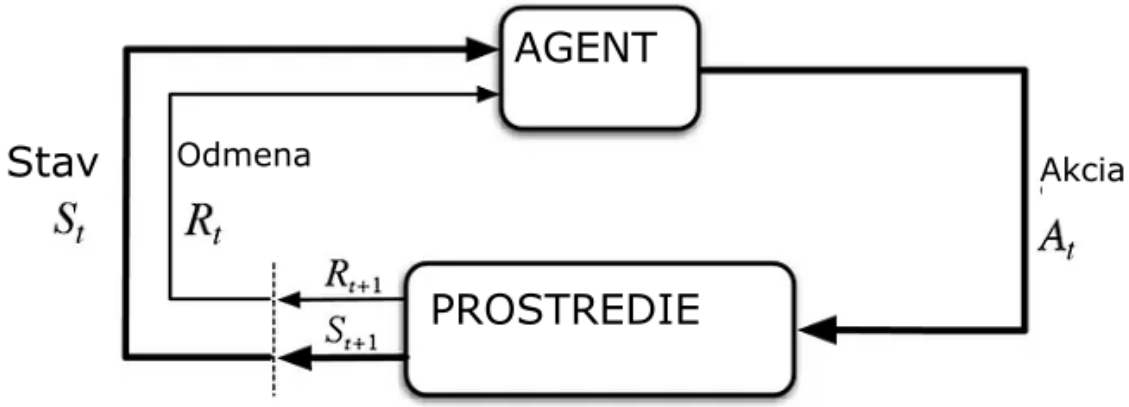


Obr. 2.7: Zhlukovanie [19]

V praxi sa táto metóda používa bežne, napr. na rozdelenie zákazníkov firmy podľa údajov, ku ktorým má firma prístup, na kompresiu obrázkov pomocou zoskupovania pixelov tých istých farieb, alebo pre zoskupovanie dokumentov podľa názvov [20].

2.3 Učenie s posilneným učením

Vo všeobecnosti metódy strojného učenia používajú stratové funkcie pre vyhodnotenie a porovnanie hypotéz. Stratová funkcia pridelí hodnotu straty dvojici hypotézy a dátovému záznamu. Metódy strojného učenia hľadajú hypotézu z typicky veľkého priestoru hypotéz, ktorá spôsobí minimálnu stratu pre akýkoľvek dátový záznam. Učenie s posilneným učením študuje aplikácie, kedy predpoveď získaná hypotézou ovplyvní budúcu generáciu dátových záznamov [8].



Obr. 2.8: Posilnené učenie [21]

Na rozdiel od učenia sa pod dohľadom, kde spätná väzba agentovi je správny súbor akcií pre vykonanie úlohy. V učení s posilneným učením sa používa tzv. odmena a trest, ako signály pre správne alebo nesprávne riešenie. Rozdiel je aj v cieli. Oproti učeniu s dohľadom, ktoré hľadá najlepšiu funkciu na minilazovanie straty, učenie s posilneným učením hľadá najlepší model, ktorý maximalizuje celkový počet odmien [22].

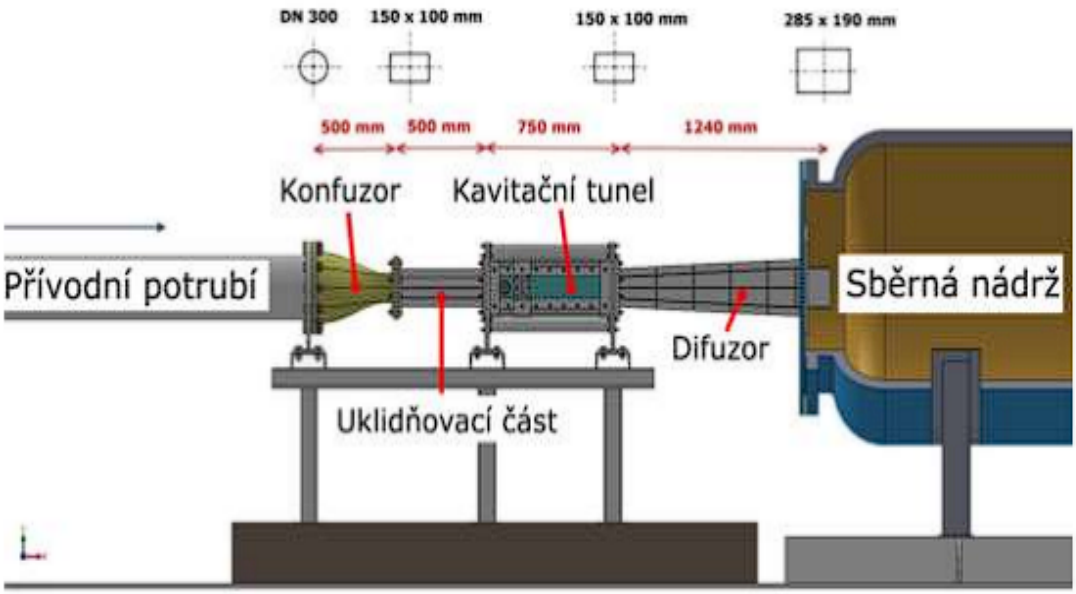
3 Experiment

Celkovo sa táto praktická časť práce dá rozdeliť na 3 časti:

- Meranie dát
- Spracovanie dát
- Aplikovanie strojného učenia

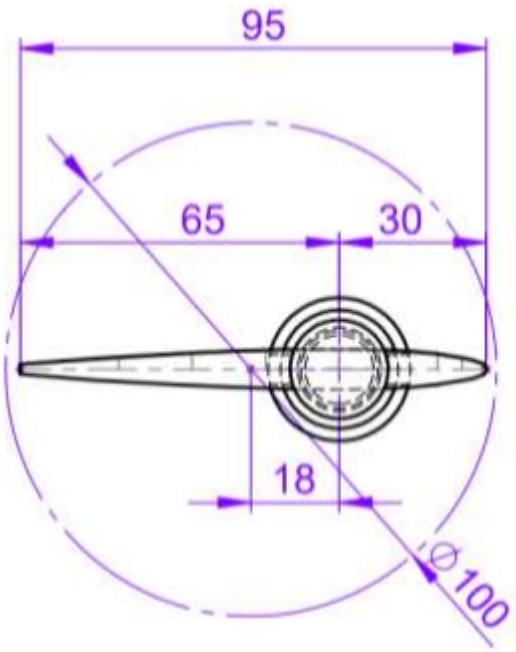
3.1 Meranie dát

Cieľom tejto práce je stanoviť prevádzkové stavy vodných strojov ako porucha, kavitácia a prietok. Pre tieto účely bola použitá už existujúca meracia trať v laboratóriách odboru fluidného inžinierstva VUT. Táto meriaca trať pozostáva z obehového čerpadla, ktoré zaisťuje prietok kvapaliny, zbernej nádrže, v ktorej je možné regulovať tlak sledovanej zóny s priehradným kavitačným tunelom a potrubnej trasy.



Obr. 3.1: Sledovaný úsek meriacej trate

Na Obr. 3.1 môže byť videný model sledovanej zóny s kavitačným tunelom, prívodným potrubím, difúzorom, ukludňovacou časťou a konfuzorom. Do kavitačného tunela bola následne umiestnená lopatka NACA profilu. Táto lopatka bola upevnená na obidvoch koncoch s možnosťou nastavenia jej sklonu od 0° po 7,5°.



Obr. 3.2: Rozmery lopatky ČKD

Ďalej bola na meracom úseku umiestnená samotná meracia technika pozostávajúca z nasledovných snímačov a techniky:

- **Piezoelektrický snímač tlaku**

Obr. 3.3: Piezoelektrický snímač tlaku Kistler typu 211B4 s tlakovým rozsahom 200 psi [23]

- **Piezoelektrický mikrofón PCB**

Obr. 3.4: Piezoelektrický mikrofón od PCB typ 130A24 [24]

- **Magneticko indukčný prietokomer**

Obr. 3.5: Magneticko indukčný prietokomer firmy KROHNE typ 4100 v rozsahu 3000 m^3/hod [25]

- **Snímač akustickej emisie**

Obr. 3.6: Snímač akustickej emisie firmy DAKEL typ MDK13AS [26]

Na trati je okrem týchto snímačov, ktoré predstavujú snímače vstupov do neurónových sietí aj ďalšia pomocná technika - snímače absolútneho tlaku v kotly, snímače teploty, vibrometre aj vysokorýchlostná kamera, ktorá je namierená práve na profil lopatky a používaná pre vyhodnotenie stavu kavitácie a nekavitácie.

Priebeh merania bol navrhnutý tak, aby sa prešlo niekoľko pracovných bodov, kde regulovateľné parametre boli; uhol nábehu lopatky, prietok a tlak v zbernej nádrži. Celkovo boli nastavené 4 uhly nábehu lopatky, a to: 0°, 2,5°, 5° a 7,5°. Tlaky v zbernej nádrži boli regulované od 30 kPa po 350 kPa.

Pre každú kombináciu uhlu nábehu lopatky a tlaku v zbernej nádrži bolo následne nastavených 35 prietokov, vyjadrených ako stredná rýchlosť v kavitačnom tunely. Meranie každého prietoku bolo opakované desaťkrát s výnimkov bodov 2,5°- 50 kPa, 5°- 130 kPa, 7,5°- 300 kPa a 7,5°- 350 kPa, kde meranie nebolo opakované desaťkrát, z dôvodu obavy o poškodenie trate následkom plne rozvinutej kavitácie.

Celkový počet pracovných bodov pre kombinácie uhol nábehu a tlak je **13 945**. Rozloženie týchto bodov môžeme vidieť v nasledovnej tabulke.

P	[kPa]	30	50	70	90	130	170	210	250	300	350
O	0°	350	350	350	350	350	350	350	350	350	350
O	2,5°	350	345	350	350	350	350	350	350	350	350
O	5°	350	350	350	350	330	350	350	350	350	350
O	7,5°	350	350	350	350	350	350	350	350	340	330

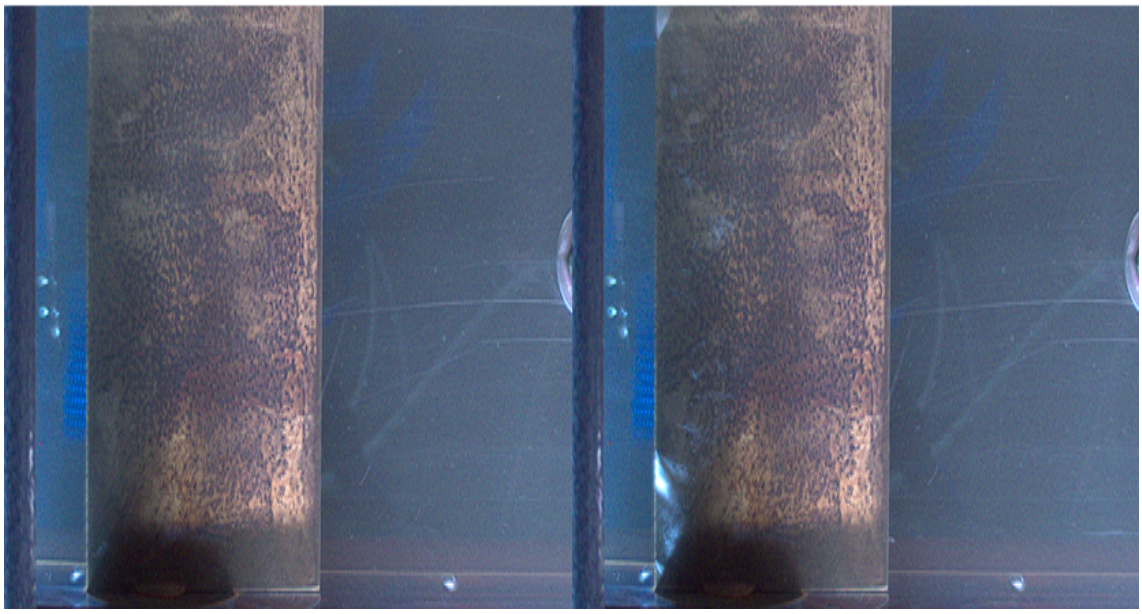
Tab. 3.1: Počet dátových záznamov pre jednotlivé pracovné body

$$D = S \times f = 13945 \times 200000 = 2789000000 \tag{3.1}$$

Kde:

- D → Celkový počet datových záznamov
- S → Počet súborov
- f → Počet datových záznamov v jednom súbore

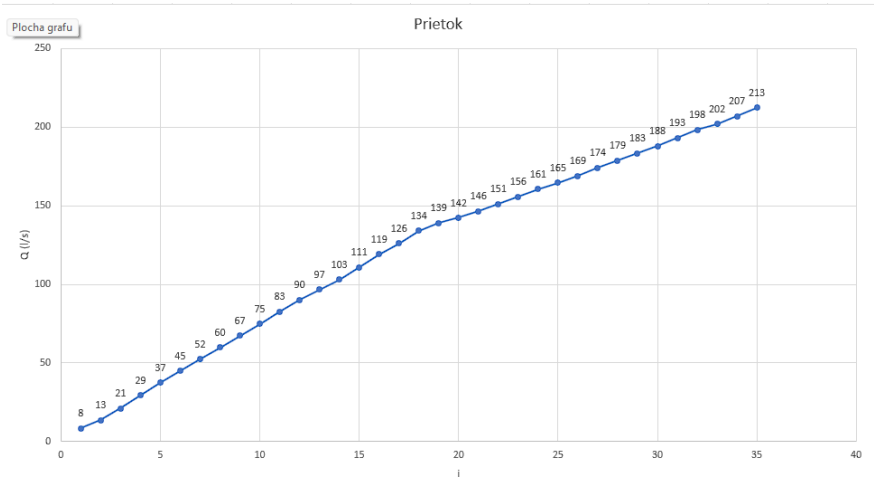
Každý pracovný bod bol meraný po časový úsek 1 sekundy, so vzorkovacou frekvenciou 200 kHz. Každý pracovný bod v sebe obsahuje 200 000 časových záznamov pre meracie snímače okrem vysokorýchlostnej kamery, kde nebolo za potrebné takej veľkej vzorkovacej frekvencie, nakoľko je jej meranie hlavne informatívne a za jej pomoci sa následne v úprave dát rozdeľujú na body s označením Kavituje a Nekavituje. Príklad záznamu z vysokorýchlostnej kamery môže byť videný na obr. 3.7 kde na ľavej strane je záznam bez kavitácia a vpravo záznam, ktorý kavituje



Obr. 3.7: Záznam z vysokorýchlostnej kamery

Samotný prietok bol v tomto prípade zvažovaný ako plávajúci parameter a mení sa na základe podmienok, meniacich sa počas merania, teda kavitácia a uhol nábehu lopatky. Menenie prietoku bolo realizované za pomoci budiacej frekvencie motoru podávajúceho čerpadla. Boli vo všeobecnosti nastavené medze stredných rýchlostí, v ktorých sa prietok bude pohybovať a to 0,5 m/s až 10 m/s.

Tu vznikol problém, kdeideálne by medzi všetkými bodmi bola konštantná zmena prietoku. Ale za prítomnosti odporov na trase a nepresnosti prietokomeru, ktorý pre prietoky pod 35 l/s trpí veľkou neistotou. Vykreslený graf prietokov v jednotlivých časových záznamoch môžeme vidieť v obrázku 3.8.



Obr. 3.8: Záznam prietokov pre 0 - 30 kPa

Pre hodnoty prietoku pod 35 l/s sa vyskytuje nelinearita, spôsobená nepresnosťou prietokomeru pre nízke prietoky. Ďalšou problematickou oblasťou je pásmo medzi prietokmi 119 - 142 l/s. Za vznik tejto nelinearity sa predpokladajú 2 príčiny, a to kavitácia a možná nepresnosť pri zadávaní otáčok čerpadla.

3.2 Spracovanie dát

Pre spracovanie dát boli využité dve metódy. Prvou metódou bol vstup s tzv. surovými dátami. Pod týmto je možné chápať dáta, ktoré neboli akokoľvek matematicky ošetrené a sú zachované ich pôvodné hodnoty. Pri zvažovaní vstupu surových dát ale musia byť tieto záznamy skrátené, aby sa uľahčila práca s nimi a ich manipulácia. Z týchto dôvodov boli časové záznamy segmentované na 10 častí, každý úsek po 20 000 časových záznamov ako je zobrazené v tabuľke 3.2.

20 000	20 000	20 000	20 000	20 000	20 000	20 000	20 000	20 000	20 000
1	2	3	4	5	6	7	8	9	10

Tab. 3.2: Segmentácia dát

Z takto rozdelených dátových záznamov boli následne vybrané stredné záznamy (5. záznam od začiatku) pre nasledovné využitie. Dôvod výberu záznamov zo stredu bolo zamedzenie vplyvu počiatku merania na počítači. Táto dátová segmentácia bola prevedená pomocou programovacieho jazyku Python a jednotlivých modulov. Program prechádzal všetky súbory z merania a v nastavenom rozsahu dát ich uložil do programu Microsoft Excel. Tento proces bol opakovaný celkovo trikrát, raz pre tlakový snímač, mikrofón a snímač akustickej emisie.

Týmto spôsobom bol dosiahnutý súbor dát o počte súborov **13 945** každý s časovým záznamom v dĺžke 0,1 sekundy s 20 000 dátovými bodmi.

$$D = S \times f = 13945 \times 20000 = 278900000 \tag{3.2}$$

Kde:

- D → Celkový počet dátových záznamov
- S → Počet súborov
- f → Počet dátových záznamov v jednom súbore

Druhá metóda úpravy dát bola vykonaná za pomoci spektografov. V tejto metóde už sú upravované hodnoty originálnych dát z merania.

Spektograf je grafická reprezentácia signálu, ktorý odhaduje rozloženie energie signálu v časovo frekvenčnej doméne. Môže byť matematicky vyjadrený nasledujúcou rovnicou [27].

$$SPEC_{x(n)}(n, w) = |STFT_{x(n)}(n, w)^2| \tag{3.3}$$

Kde:

- **STFT** → Rýchla Fourierova transformácia
- **x** → Časový index signálu
- **w** → Frekvenčný index signálu

Na základe rovnice 3.3, spektograf je vlastne rýchla Fourierova transformácia na druhú. Rýchla Fourierova transformácia, ďalej iba STFT, je prvá modifikácia klasickej Fourierovej transformácie, ktorá dáva možnosť analyzovať nestacionárne signály v časovo frekvenčnej doméne. Bola prvýkrát predstavená Gaborom v roku 1946. STFT časového signálu, *x(t)*, $STFT_{x(t)}(t, f)$ môže byť vyjadrená nasledovnou rovnicou [27].

$$STFT_{x(t)}(t, w) = \int_{-\infty}^{+\infty} x(t)w(t - \tau)exp(-jwt)d\tau \tag{3.4}$$

Kde:

- **w**(τ) → Oknová funkcia
- τ → Časová premenná
- **j** → Imaginárna jednotka

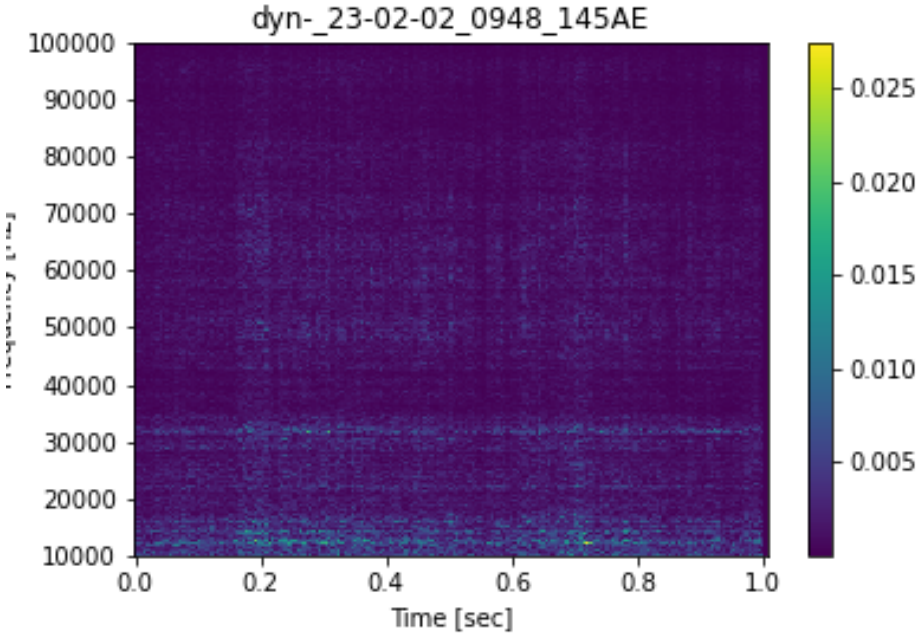
Základný princíp STFT je namiesto spočítania diskrétnej Fourierovej transformácie celého signálu, kde je signál rozdelený na kratšie segmenty tej istej dĺžky za použitia časovo lokalizovanej oknovej funkcie ako Gausovské okno alebo Hammingovo okno. Následne sa prevedie diskrétna Fourierova transformácia osobitne na každý segment originálneho signálu, ktoré dokopy vytvoria časovo-frekvenčné spektrum signálu [27].

Diskrétna Fourierova transformácia sa dá vyjadriť nasledovne:

$$x(w) = \int_{-\inf}^{+\inf} x(t)e^{-jwt}dt \tag{3.5}$$

Kde:

- **x(t)** → Signál
- **w** → Frekvencia
- **j** → Imaginárna jednotka



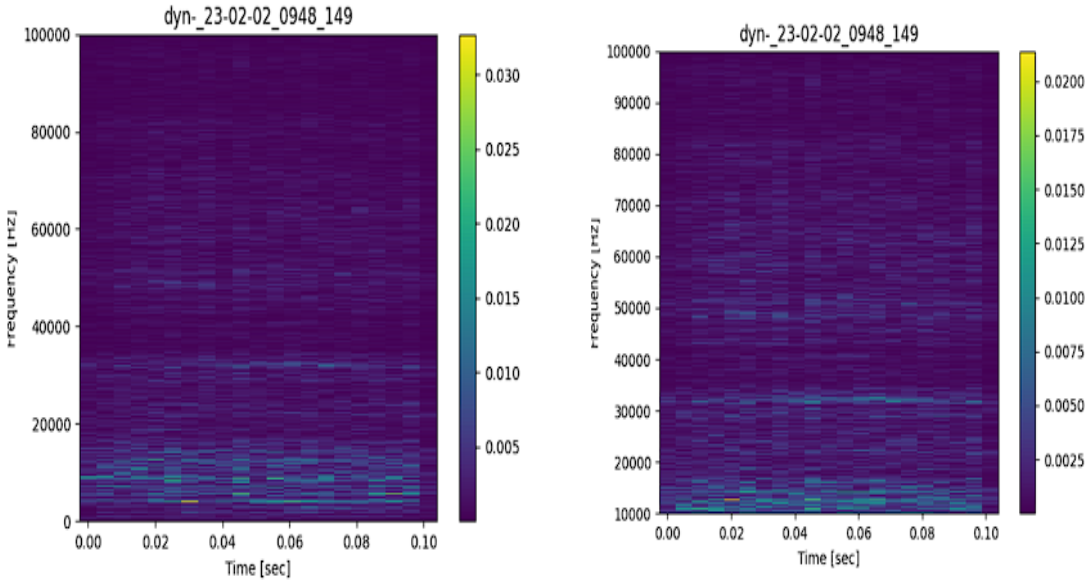
Obr. 3.9: Príklad spektografu

Úprava dát pomocou spektografov bola realizovaná v programovom jazyku Python, za pomoci modulu **scipy.signal STFT**, ktorý umožnuje vypočítať STFT a následne vypočítať spektograf.

Následne boli nastavené parametre pre výpočet. **Fs** reprezentuje vzorkovaciu frekvenciu vstupného signálu a bola nastavená na 200 kHz čo je priamo vzorkovacia frekvencia snímačov použitých v meraní. Ako oknová funkcia bola ponechaná **Hammingova** funkcia, ktorá je v module prednastavená. Táto funkcia je široko používaná a poskytuje kompromis medzi frekvenčným rozlíšením a amplitúde signálu, ktorý sa prenesie z jedného segmentu do ďalšieho. **Nperseg** predstavuje veľkosť samotného okna a po otestovaní rozličných veľkostí, kde bola vybraná veľkosť okna 2048, z dôvodu dobrého rozlíšenia, ale aj požiadavky vkladania veľkosti okien v rozmedzí mocním čísla 2, a **noverlap** čo predstavuje počet bodov, ktoré sa prekrývajú medzi segmentami a bola nastavená ako polovica hodnoty nperseg, teda 50 % prekrívania.

Výstupom z tohto programu sú 3 matice. Matica **f** obsahujúca vzorkovacie frekvekcie, matica **t** obsahujúca časy segmentov a matica **Sxx**, teda 2D matica komplexných čísel, kde reálna zložka čísla predstavuje amplitúdu a imaginárna zložka zase fázu. Pre účely tejto úlohy sú fázy nepotrebné a boli uložené iba absolútne hodnoty tohto spektra. Tieti hodnoty predstavujú pravú amplitúdu a niesú normalizované podľa maximálnej amplitúdy.

Pred uložením boli ešte odstránené frekvencie pod 10 kHz, nakoľko pod touto frekvenciou nájdeme frekvenciu samotného obehového čerpadla, ktorá by mohla spôsobovať problémy pri trénovaní siete. Z týchto matíc bola uložená iba matica **Sxx** pre ďalšie použitie v samotných datasetoch.

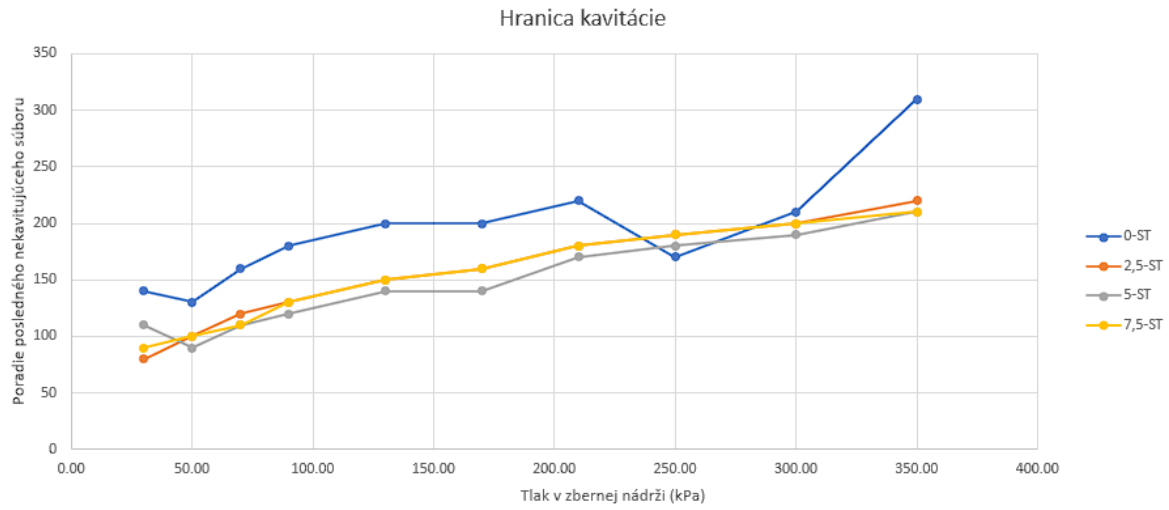


Obr. 3.10: Porovnanie spektografov pred filtráciou (vľavo) a po filtrácii (vpravo)

Ako vstupné dáta boli znova použité dáta z troch snímačom (Tlakový senzor, Akustická emisia, Mikrofón). Namiesto redukcie počtu dát segmentovaním, ako v prvom prípade dátového spracovania sa v tomto prípade realizovala redukciu dát spôsobom redukcie počtu súborov. Z pôvodných 13 945 súborov bol vybraný každý piaty súbor, a na tento bol aplikovaný kód generujúci spektografy. Je získaný teda **2 789** súborov, každý s 200 000 dátovými záznamami. Celkový počet dátových záznamov sa rovná **557 800 000**. Výsledná veľkosť matíc frekvenčno-časového spektra je **(913, 199).**

3.3 Príprava datasetov

Po úprave vstupných dát je potrebné tieto dáta rozdeliť podľa požadovaného spôsobu. V tejto práci sa vyskytujú 2 typy datasetov. Prvý bol zostavený pre prípad detekcie kavitácie, konkrétne prípad **Kavituje** ,**Nekavituje**. Dáta zo všetkých senzorov boli rozdelené na 2 skupiny za pomoci záznamov z vysokorýchlostnej kamery. Využitím záznamov z tejto kamery, ktoré obsahovali čas záznamu snímku, bola identifikovaná hranica – kavitácia, a tá bola pridelená k dátam. Tento spôsob sa veľmi osvedčil pre väčšie natočenie lopatky. Podľa záznamov pre uhly natočenia od 2,5°a vyššie, bola zreteľná hranica počiatku kavitácie na lopatke. Pre uhol natočenia 0° to bolo komplikovanejšie. Nie vždy bolo jasne zreteľné kedy kavitácia nastávva a kedy nie. Pre hodnoty prietoku bolo rozdelenie dát jednoduchšie, nakoľko vieme že každý prietok bol opakovaný desaťkrát pre každú kombináciu tlaku v zbernej nádrži a natočenia lopatky (s vínimkou bodov, kedy bolo meranie predbežne zastavené z dôvodu hrozby poškodenia trate). Obr. 3.11 znázorňuje na osi **Y** poradie posledného dátového súboru, kedy kavitácia nebola viditeľná a tlak v kotly na osi **X**.



Obr. 3.11: Graf znázorňujúci hranice kavitácie pre jednotlivé pracovné body

Prietok v tomto prípade nebol nastavený ako presný parameter, ako bolo zmienené v predošlej kapitole. Z dôvodov nepresnosti prietokomeru a vzniku samotnej kavitácie všetky hodnoty prietoku nemusia odpovedať jednému prietoku v skupine desiatich. Dáta zo všetkých snímačov boli rozdelené do 35-tich súborov, jeden súbor pre jednu skupinu prietoku.

Takto rozdelené dáta bolo následne potrebné rozdeliť do množín. Neurónové siete, ktoré budú aplikované na tieto dáta, patria do skupiny učenia pod dozorom a využívajú sa v nich 3 tipy dátových množín:

- Trénovacia množina
- Validačná množina
- Testovacia množina

Trénovacia množina predstavuje najväčšiu časť dát. Táto množina v sebe obsahuje predpracované dáta s prideleným označením. Takto označené dáta slúžia pre natrénovanie neurónovej siete. Tieto dáta sú prvé, ktoré neurónová sieť vidí a snaží sa upraviť svoje trénovacie parametre tak, aby minimalizovala trénovaciu chybu.

Validačná množina sa používa iba nepriamo pri trénovaní testovacej siete, pre výber najlepšieho vygenerovaného modelu a pomáha zabrániť preplneniu neurónovej siete.

Testovacia množina je množina, ktorá obsahuje dáta, ktoré neurónová sieť ešte nikdy nevidela. Táto množina v rámci strojného učenia a neurónových sietí reprezentuje takzvané cudzie dáta alebo dáta, ktoré neurónovej sieti doposiaľ neboli predstavené. V reálnej aplikácii predstavujú dáta, ktoré sú neurónovej sieti predstavené po natrénovaní a uložení.

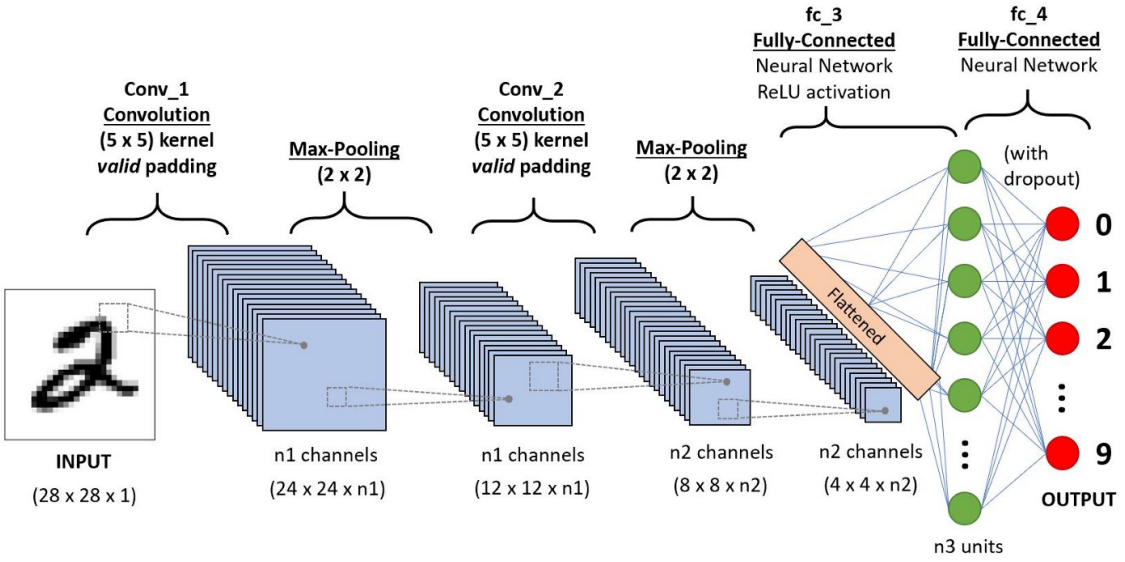
Toto rozdelenie bolo prevedené za pomoci programu Pycharm v programovom jazyku Python a jeho jednotlivých modulov a to hlavne Pandas, Numpy, Random a Tensorflow. Dáta boli najprv načítané z ich príslušných zložiek. Dáta z týchto súborov boli načítané do programu a vložené do prázdnej množiny a príslušne označené číselnou hodnotou napríklad $0 \rightarrow Nekavituje$ a $1 \rightarrow Kavituje$. Všetky dáta boli následne načítané do jednej veľkej množiny, kde boli za pomoci funkcie random.shuffle zamiešané, aby neboli v poradí, v akom sa nachádzali v zložkách. Takto zamiešané dáta boli následne rozdelené do už zmienených 3 množín, a to v pomere:

- $0.7 \rightarrow$ Trénovacia množina
- $0.2 \rightarrow$ Validačná množina
- $0.1 \rightarrow$ Testovacia množina

Celý takýto dataset bol následne uložený vo formáte .npz, čo je komprimovaný súbor Pythonu, ktorý môže byť priamo načítaný do neurónovej siete. Pre prípad záznamov zo snímačov, ktoré boli iba segmentované, tieto množiny v sebe obsahujú vektorové dáta, kde každý vektor obsahuje jedno značenie či kavitáciu, alebo hodnotu prietoku. Pre prípad spektografov sú tieto množiny tvorené maticami frekvenčno-časovej závislosti, kde každá matica má zase jedno označenie.

4 Konvulenčná neuronová sieť

V tejto kapitole bude predstavená takzvaná konvolučná neurónová sieť, ktorá bola zvolená ako voľba pre riešenie danej problematiky. Konvulenčné neurónové siete, od teraz už len ako CNN, sa učia abstrakné vlastnosti z alternajúcich a vrstvených konvolunčných vrstiev a tzv. pooling funkcií.



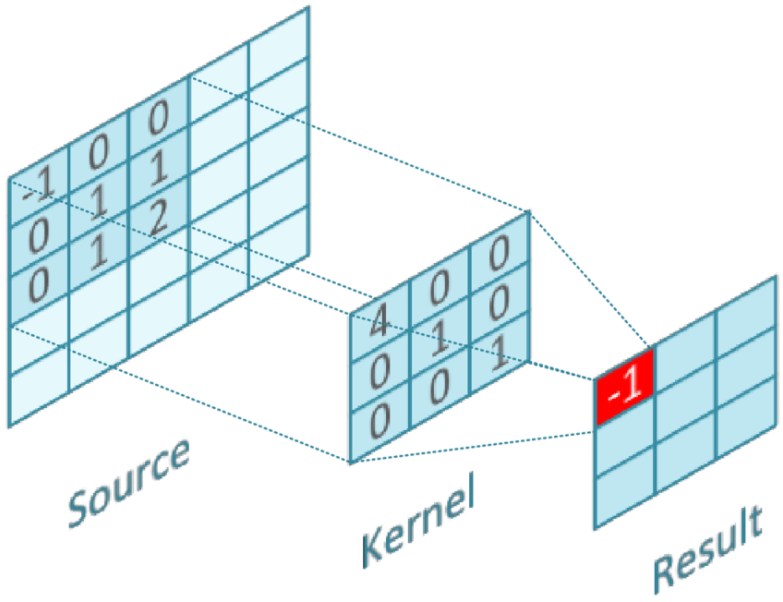
Obr. 4.1: Príklad konvulenčnej neurónovej siete [28]

4.1 Vrstvy

Neurónové siete boli už predstavené v teoretickej časti tejto práce. Konvolučné neurónové siete sú predovšetkým charakterizované vrstvami, ktoré sa v nej používajú.

4.1.1 Konvolučná vrstva

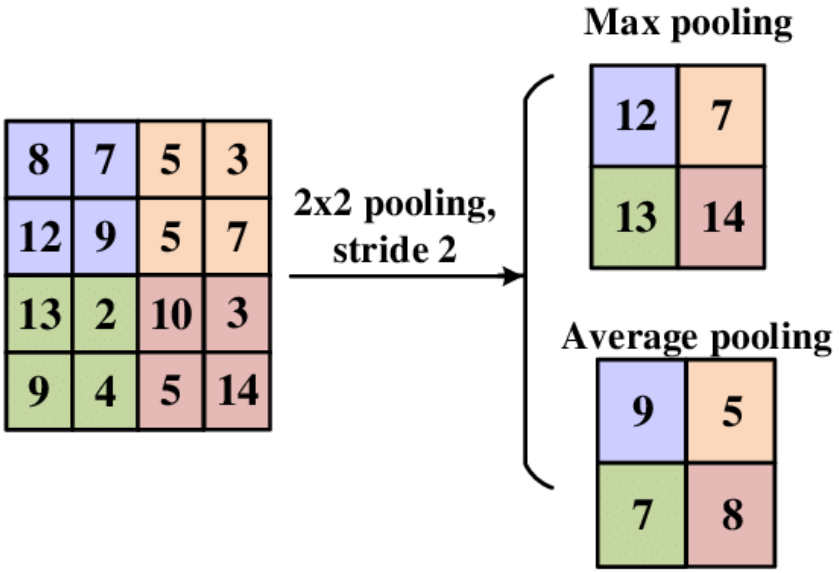
Konvulenčné vrstvy sú základom CNN a uskutočnuje sa v nich majorita procesov CNN. Vyžaduje vstupné dáta, filter a mapu vlastností. Samotný filter je 2D matica hmotností w. Tento filter je následne aplikovaný na vstupné dáta. Medzi vstupnými dátami a filtrom sa vypočíta bodový súčin. Tento produkt sa potom privedie do výstupného poľa. Filter sa následne posunie o krok a proces sa opakuje, kým filter neprejde cez celé vstupné dáta. Konečný výstup zo série bodových súčinov zo vstupných dát a použitého filtra sa nazýva mapa vlastností Výstupy z konvolunčných vrstiev následne vstupuje do aktivačných funkcií [29].



Obr. 4.2: Filter CNN [30]

4.1.2 Združovacia vrstva

Združovacie vrstvy slúžia na postupnú redukciu rozmerov dát, a tým redukujú počet parametrov a čas potrebný pre výpočet daného modelu. Združovacia vrstva operuje cez aktivačnú mapu vo vstupoch a upravuje jej rozmery za pomoci funkcie **MAX** alebo **AVERAGE**.

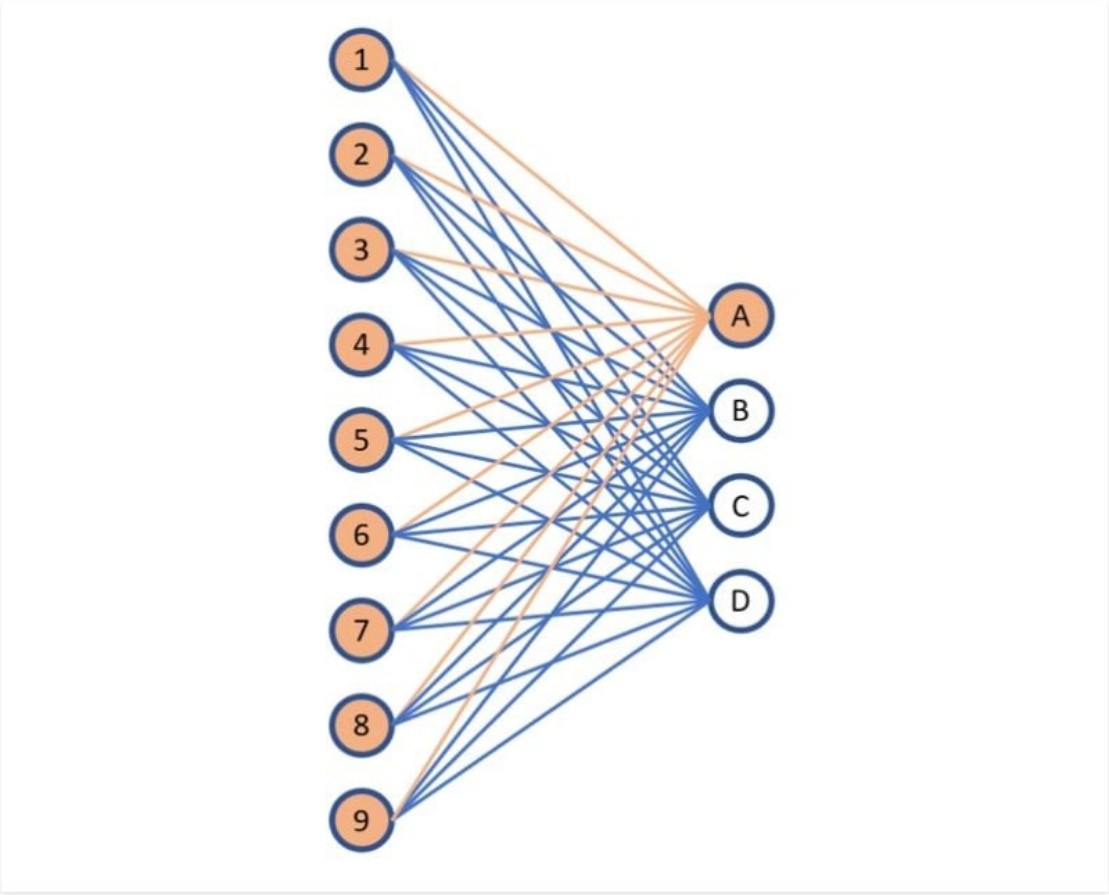


Obr. 4.3: Max a average pooling [31]

Max pooling vyberá na veľkosti jeho filtra maximálnu hodnotu a average vyberá strednú hodnotu medzi všetkými hodnotami na veľkosti jeho filtra. Vo väčšine CNN sa najčastejšie používa funkcia **MAX** a je ju možné vidieť vo forme združovacích - max vrstvách alebo max-pooling layers s filtrom o rozmere 2x2, ktorý je aplikovaný s krokom o veľkosti 2 po priestorových rozmeroch vstupu. Toto by redukovalo originálnu veľkosť vstupov na 25 %. Rozmery tohto filtra môžu byť upravované pre väčšiu dimenzionálnu redukciu dát. Nevýhodou týchto vrstiev je, že síce redukuje rozmery dát, čo sa prejaví na redukcii výpočtového času ,ale strácame tu nejaké informácie, ktoré by dáta mohli niesť [29].

4.1.3 Plne prepojená vrstva

Ako meno napovedá, plne prepojené vrstvy sa skladajú z neurónov, ktoré sú plne prepojené s predchádzajúcou vrstvou. Táto vrstva slúži ako klasifikačná na základe vlastností vybraných zo vstupných dát a vrstiev a rozličných filtrov [29].



Obr. 4.4: Plne prepojená vrstva [32]

4.2 Aktivačné funkcie

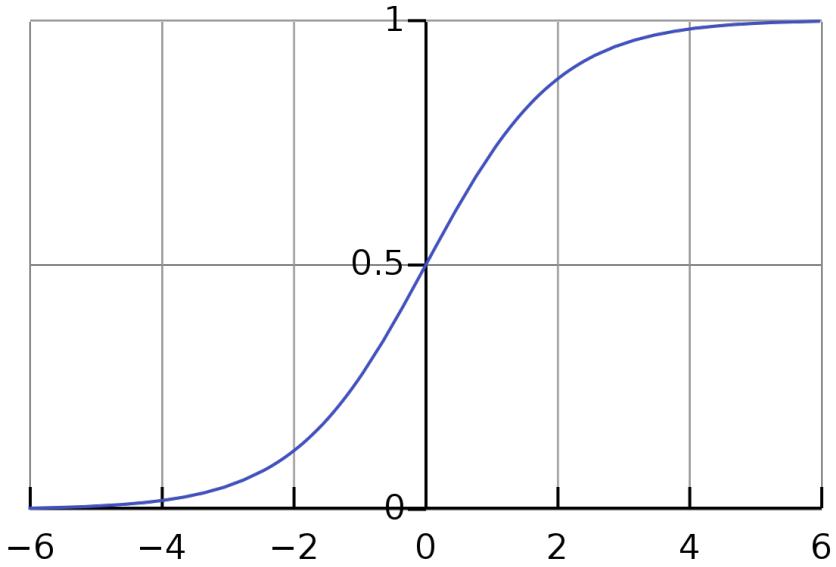
Po výstupe z konvulenčnej vrstvy dáta vstupujú do tzv. aktivačných funkcií. Aktivačné funkcie sú funkcie, ktoré determinujú výstup z neurónovej siete ako napr. **Kavituje**, **Nekavituje**.
Mapuje výsledné hodnoty a pridáva im hodnotu v závislosti na použitej aktivačnej funkcii. Rozlišujú sa 2 tipy aktivačných funkcií.

1. Lineárne aktivačné funkcie
2. Nelineárne aktivačné funkcie

V praxi sa využívajú hlavne nelineárne aktivačné funkcie, a to hlavne z dôvodu nelinearity sledovaného deja.

4.2.1 Sigmoid a Logistická aktivačná funkcia

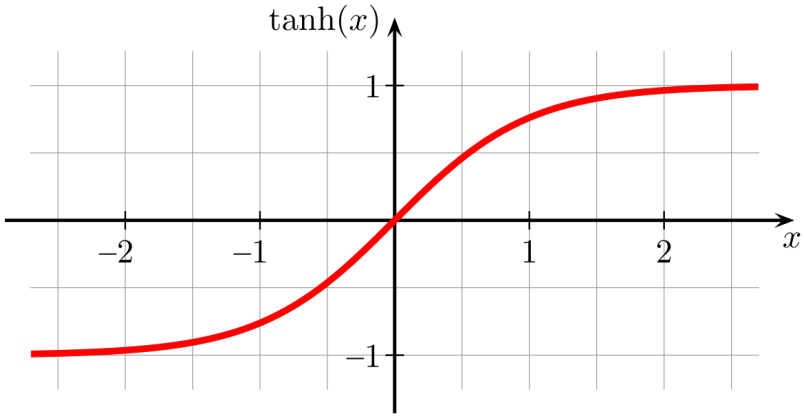
Sigmoidová funkcia existuje v rozmedziach <0,1>. Používa sa predovšetkým pri modeloch, kde predpovedáme pravdepodobnosť ako výstup. Táto funkcia je diferencovatelná, čo znamená, že môže byť nájdené stúpanie sigmiodovej krivky medzi akýmikoľvek dvomi bodmi. Táto funkcia môže drasticky spomaliť trénovanie modelu, preto sa často nahradzuje aktivačnou funkciou **softmax** používanou pre viac triedovú klasifikáciu [33].



Obr. 4.5: Sigmoidova krivka [34]

4.2.2 Tanh aktivačná funkcia

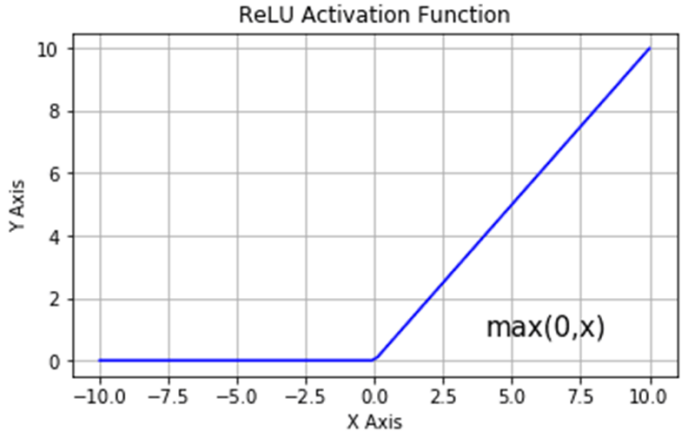
Tanh je podobná Sigmoidovi s rozdielom v hraniciach funkcie. Kým pri Sigmoidovi sú hranice <0,1> v Tanh sú tieto hranice <-1,1>. Výhodou tejto funkcie je, že záporné hodnoty môžu byť mapované ako záporné a 0 môžu byť mapované blízko 0. Táto funkcia sa používa hlavne pre binárnu klasifikáciu [33].



Obr. 4.6: Funkcia tanh [35]

4.2.3 ReLu aktivačná funkcia

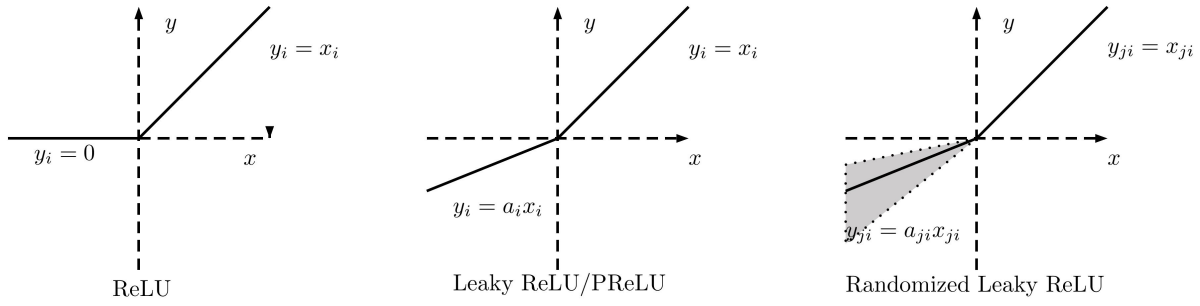
ReLu patrí k najpoužívanejším aktivačným funkciám. Používa sa takmer vo všetkých konvolunčných neurónových sieťach alebo hĺbkovom učení. Problém pri tejto funkcii sa vyskytuje pri hodnotách pod 0, čo limituje možnosti tohto modelu trénovať na dátach. Toto v praxi znamená, že akékolvek záporné hodnoty, ktoré vstúpia do ReLU sa premenia na 0 v grafe, čo vo výsledku ovplyvní výsledný graf [33].



Obr. 4.7: Funkcia ReLU [36]

4.2.4 Leak ReLU

Funkcia Leak ReLU je vylepšená verzia klasickej funkcie ReLU, ktorá zlepšuje jej rozsah. Tento rozsah je väčšinou zväčšený o 0.01, ak nie, táto funkcia sa volá Randomize ReLU a jej teoretické hranice sú v rozmedzí $< -\inf, +\inf >$



Obr. 4.8: Porovnanie funkcií ReLU [37]

4.3 Stratové funkcie

V strojnom učení sa používajú stratové funkcie pre vyjadrenie, ako dobre je model schopný predikovať alebo klasifikovať výstupy z daných vstupov. Cieľom strojného učenia je túto chybu minimalizovať. Tieto stratové funkcie môžu byť rozdelené do troch skupín

1. Regresné stratové funkcie
2. Binárne klasifikačné stratové funkcie
3. Viac triedne klasifikačné stratové funkcie

V tejto práci sú použité stratové funkcie v kategórii 2 a 3. Pre prípady binárnej klasifikácie je použitá takzvaná binárna krížová entropická strata, ktorá je popísaná nasledovnou rovnicou [38].

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \times \log(p(y_i)) + (1 - y_i) \times \log(1 - p(y_i)) \tag{4.1}$$

Kde:

- $\mathbf{y} \rightarrow$ Je označenie
- $\mathbf{p(y)} \rightarrow$ Je pravdepodobnosť že bod bude patriť do triedy
- $\mathbf{N} \rightarrow$ Počet bodov

Druhou použitou stratovou funkciou použitej pre klasifikáciu s viacerými triedami je riedka krížová entropická strata. Rozdiel od viactriednej krížovej entropickej straty je, že pre riedku krížovú entropickú stratu sú kategórie označované celými číslami [38].

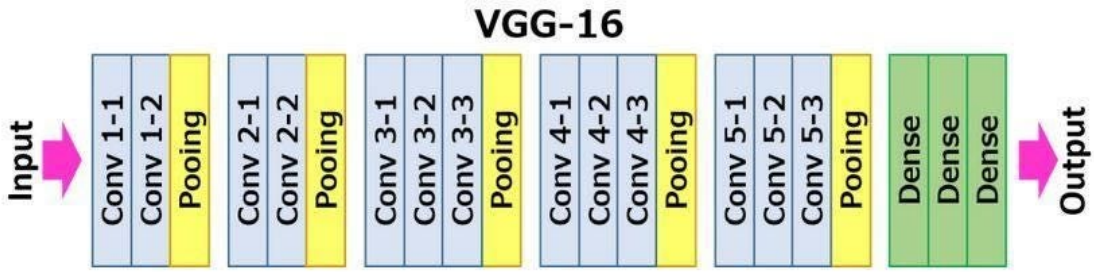
$$J(w) = -\frac{1}{N}\sum_{i=1}^{N}[\hat{y}_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{4.2}$$

Kde:

- $\mathbf{w} \rightarrow$ Váhy parametrov modelu
- $\mathbf{y} \rightarrow$ Správne označenia bodov
- $\hat{y} \rightarrow$ Predpovedané označenia bodov

4.4 Architektúra CNN

Pre architektúru CNN bola použitá takzvaná VGG16 architektúra. VGG16 je typ CNN, ktorá je uvažovaná ako jedna z najlepších modelov, bola prvý krát publikovaná v roku 2015. VGG znamenajúc vizuálna geometrická skupina objavuje efekty zvyšovacej hĺbky konvolunčných sietí na ich presnosť. Sú to architektúry s veľmi malými filtrami 3x3, ktoré ukazujú značné zlepšenie v porovnaní s inými konfiguráciami. CNN bola naprogramovaná za pomoci jazyku Python a modulov tensorflow a keras. Pre prípad klasifikácie prípadu Kavituje a Nekavituje je architektúra nasledovná Celá architektúrá sa skladá z 13 konvolučných 1D vrstiev, 5 1D združovacích vrs-



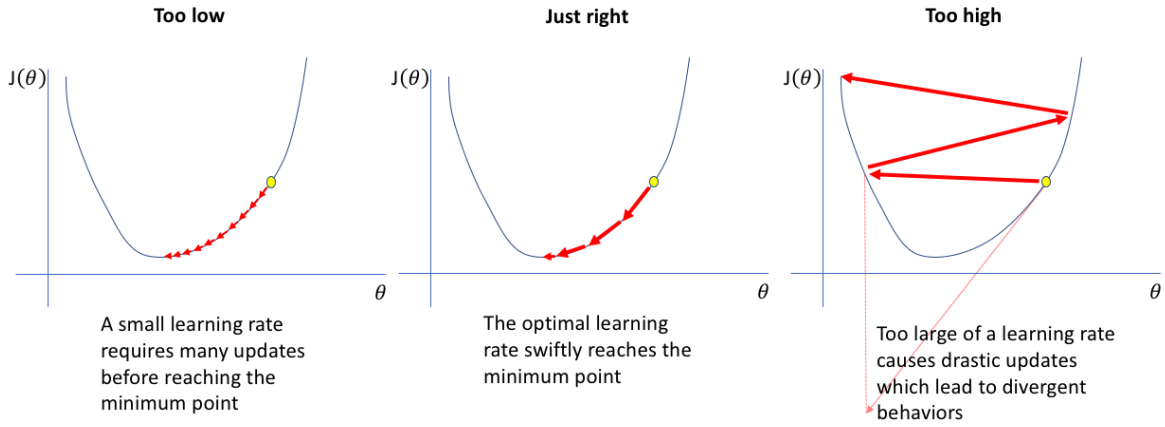
Obr. 4.9: Architektúra VGG16 [39]

tiev a 3 plne prepojených vrstiev (pre prípad dvojrozmerných vstupných dát sú tieto vrstvy premenené na 2D. Toto je viac popísané neskôr). Velkosti všetkých filtrov pre konvolunčné vrstvy sú 3. Kedže vstupné dáta predstavajú tenzory 1. rádu nie je za potrebné definovať druhú dimenziu filtra. Združovacie vrstvy, ako bolo už zmienené sa používajú pre redukciu veľkosti dát. V tomto prípade všetky združovacie vrstvy využívajú velkosť filtru 4 s maxpoolingom. Konvulenčné vrstvy sú nastavené tak, aby vytvárali pyramídový efekt a tak sa zväčšuje počet ich filtrov smerom dole a to

od 4 po 64 v poslednej vrstve.

Všetky vrstvy používajú aktivačnú funkciu relu okrem poslednej plne prepojenej, ktorá využíva funkciu Sigmoid z dôvodu, že sa jedná o binárnu klasifikáciu a pre iné aktivačné funkcie bola generovaná chyba, alebo nebola vyhodnocovaná samotná strata. Jedná sa o prípad binárnej klasifikácie takže bola aplikovaná binárna krížová entropická strata.

Jedným z cieľov je porovnať jednotlivé snímače a metódy úpravy a ich vplyv na presnosť CNN. Z tohto dôvodu boli nastavené hyperparametre rovnaké pre všetky prípady. Hyperametre ako learning rate bol nastavený na hodnotu 0.00001. Learning rate sa dá predstaviť ako krok, ktorý funkcia zoberie pre hľadanie optima. Ak je krok príliš veľký, može sa stať, že toto optimum preskočí, alebo sa k nemu nikdy nedostane.. V opačnom prípade, ak je moc malý, predĺži to dobu nájdenia optima modelu.



Obr. 4.10: Vplyv veľkosti learning rate na nájdenie minima funkcie [40]

Ďalším parametrom sú epochy, ktoré predstavujú jednu iteráciu cez celý dataset počas trénovania. A Batch size, ktoré predstavujú rozdelenie datasetov na menšie úseky. Batch size v tomto prípade udáva veľkosť týchto segmentov, ktoré sú preposlané cez CNN po požadovaný počet epoch. Počet epoch bol v tomto prípade nastavený na 35 s batch size 150.

Layer (type)	Output Shape	Param #
conv1d_39 (Conv1D)	(None, 19999, 4)	16
conv1d_40 (Conv1D)	(None, 19997, 4)	52
max_pooling1d_15 (MaxPooling1D)	(None, 4999, 4)	0
conv1d_41 (Conv1D)	(None, 4997, 8)	104
conv1d_42 (Conv1D)	(None, 4995, 8)	200
max_pooling1d_16 (MaxPooling1D)	(None, 1248, 8)	0
conv1d_43 (Conv1D)	(None, 1246, 16)	400
conv1d_44 (Conv1D)	(None, 1244, 16)	784
conv1d_45 (Conv1D)	(None, 1242, 16)	784
max_pooling1d_17 (MaxPooling1D)	(None, 310, 16)	0
conv1d_46 (Conv1D)	(None, 308, 32)	1568
conv1d_47 (Conv1D)	(None, 306, 32)	3104
conv1d_48 (Conv1D)	(None, 304, 32)	3104
max_pooling1d_18 (MaxPooling1D)	(None, 76, 32)	0
conv1d_49 (Conv1D)	(None, 74, 64)	6208
conv1d_50 (Conv1D)	(None, 72, 64)	12352
conv1d_51 (Conv1D)	(None, 70, 64)	12352
max_pooling1d_19 (MaxPooling1D)	(None, 17, 64)	0
flatten_3 (Flatten)	(None, 1088)	0
dense_9 (Dense)	(None, 512)	557568
dense_10 (Dense)	(None, 128)	65664
dense_11 (Dense)	(None, 1)	129

Total params: 664,389
Trainable params: 664,389
Non-trainable params: 0

Obr. 4.11: Architektúra CNN pre binárnu klasifikáciu zo segmetovaných dát

Pre prípad segmentácie dát a určovanie prietokov sa samotná architektúra moc nemení. Je dôležité uvedomiť si, že počet neurónov v poslednej plne prepojenej vrstve odpovedá počtu označení v našich datasetoch. V prípade binárnej klasifikácie sa jedná o jeden neurón, ale pre prípad kedy je cieľom klasifikovať jednotlivé prietoky, sa počet neurónov zmení na 35. Ďalej, nakoľko sa nejedná už o binárnu klasifikáciu, je zmenená stratová funkcia z binárnej krížovej entropickej straty na riedku krížovú entropickú stratu.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 19999, 4)	16
conv1d_1 (Conv1D)	(None, 19997, 4)	52
max_pooling1d (MaxPooling1D)	(None, 4999, 4)	0
conv1d_2 (Conv1D)	(None, 4997, 8)	104
conv1d_3 (Conv1D)	(None, 4995, 8)	200
max_pooling1d_1 (MaxPooling1D)	(None, 1248, 8)	0
conv1d_4 (Conv1D)	(None, 1246, 16)	400
conv1d_5 (Conv1D)	(None, 1244, 16)	784
conv1d_6 (Conv1D)	(None, 1242, 16)	784
max_pooling1d_2 (MaxPooling1D)	(None, 310, 16)	0
conv1d_7 (Conv1D)	(None, 308, 32)	1568
conv1d_8 (Conv1D)	(None, 306, 32)	3104
conv1d_9 (Conv1D)	(None, 304, 32)	3104
max_pooling1d_3 (MaxPooling1D)	(None, 76, 32)	0
conv1d_10 (Conv1D)	(None, 74, 64)	6208
conv1d_11 (Conv1D)	(None, 72, 64)	12352
conv1d_12 (Conv1D)	(None, 70, 64)	12352
max_pooling1d_4 (MaxPooling1D)	(None, 17, 64)	0
flatten (Flatten)	(None, 1088)	0
dense (Dense)	(None, 512)	557568
dense_1 (Dense)	(None, 128)	65664
dense_2 (Dense)	(None, 35)	4515

Total params: 668,775
Trainable params: 668,775
Non-trainable params: 0

Obr. 4.12: Architektúra CNN pre prípad klasifikácie prietokov za pomoci segmentovaných dát

Najväčšia zmena je zaznamenaná v prípade vyhodnocovania kavitácie a prietoku za pomoci spektografov. V tomto prípade sú vstupné dáta do CNN v tvare tenzoru druhého rádu. Z tohto dôvodu je za potrebné prepísať všetky vrstvy z 1D na 2D a definovať druhé rozmery všetkých filtrov. Pre prípad stratových funkcií a aktivačných funkcií, sú zmeny v CNN také isté, ako pre prípad segmentácie. Tak isto bola pre združovacie vrstvy definovaná možnosť same. Táto možnosť nahradzuje eliminované dáta nulami, aby boli zachované rozmery matice dát.

Layer (type)	Output Shape	Param #
conv2d_13 (Conv2D)	(None, 911, 197, 4)	40
conv2d_14 (Conv2D)	(None, 909, 195, 4)	148
max_pooling2d_5 (MaxPooling 2D)	(None, 454, 97, 4)	0
conv2d_15 (Conv2D)	(None, 452, 95, 8)	296
conv2d_16 (Conv2D)	(None, 450, 93, 8)	584
max_pooling2d_6 (MaxPooling 2D)	(None, 225, 47, 8)	0
conv2d_17 (Conv2D)	(None, 223, 45, 16)	1168
conv2d_18 (Conv2D)	(None, 221, 43, 16)	2320
conv2d_19 (Conv2D)	(None, 219, 41, 16)	2320
max_pooling2d_7 (MaxPooling 2D)	(None, 110, 21, 16)	0
conv2d_20 (Conv2D)	(None, 108, 19, 32)	4640
conv2d_21 (Conv2D)	(None, 106, 17, 32)	9248
conv2d_22 (Conv2D)	(None, 104, 15, 32)	9248
max_pooling2d_8 (MaxPooling 2D)	(None, 52, 8, 32)	0
conv2d_23 (Conv2D)	(None, 50, 6, 64)	18496
conv2d_24 (Conv2D)	(None, 48, 4, 64)	36928
conv2d_25 (Conv2D)	(None, 46, 2, 64)	36928
max_pooling2d_9 (MaxPooling 2D)	(None, 23, 1, 64)	0
flatten_1 (Flatten)	(None, 1472)	0
dense_3 (Dense)	(None, 512)	754176
dense_4 (Dense)	(None, 128)	65664
dense_5 (Dense)	(None, 1)	129

Total params: 942,333
Trainable params: 942,333
Non-trainable params: 0

Obr. 4.13: Architektúra CNN pre prípad klasifikácie kavitácie za pomoci spektografov

4.5 Aplikácia strojného učenia

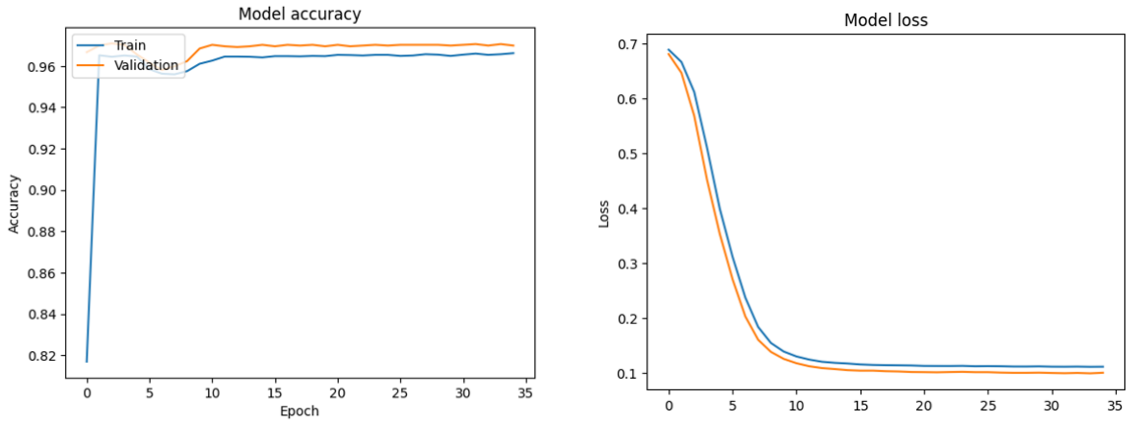
Všetky neurónové siete boli trénované a testované v online prostredí Google Colab. Toto prostredie umožňuje externe sa pripojiť na vzdialené počítačové zariadenie a využiť jeho prostriedky pre riešenie daného problému. Neplatené prostredie Google Colabu poskytuje užívateľovi 12,7 GB operačnej pamäte a 107,7 GB miesta na externom disku. Program za pomoci modulov keras a tesorflow trénuje, validuje a testuje danú neurónovú sieť a zároveň ukladá najlepší možný model pre ďalšie použitie.

4.5.1 Klasifikácia kavitácie

V prvom prípade boli cez neurónku pustené segmentované dáta pre všetky 3 snímače. Výstupom neurónovej siete sú 2 grafy..

Snímač akustickej emisie

V prvom grafe môžeme vidieť presnosť modelu, a to konkrétne validačná a trénovacia, a na druhom grafe trénovacia a validačná strata pre tento prípad. Ako bolo už zmienené, je to binárna krížová entropická strata. Grafy pre jednotlivé snímače môžeme vidieť nižšie.



(a) Presnosť

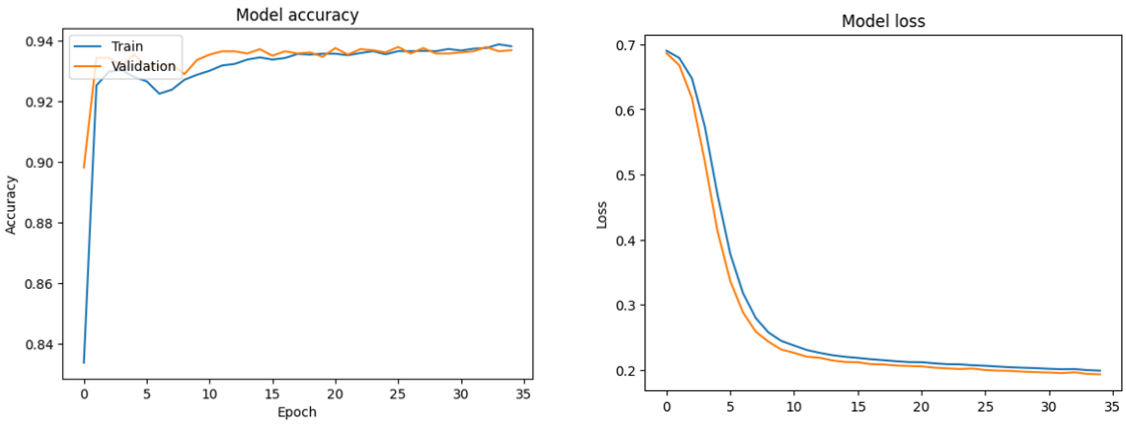
(b) Straty

Obr. 4.14: Presnosť a strata snímača akustickej emisie pre binárnu klasifikáciu zo segmentovaných dát

Z grafov 4.14 je možné vidieť, že pre snímač akustickej emisie bola neurónová sieť schopná dosiahnuť vysokého percentilu natrénovanosti a validácie, za malý počet prejdených epóch. Z dát programu môže byť vyčítané, že už po prvej epoche bola validačná presnosť vo výške 96 % a trénovacia presnosť cez 81 %.

Trénovacia presnosť na konci 35. epochy je 96 % a validačná presnosť dosiahla maximum 97,13 %. Po natrénovaní bol cez neurónku prehnaný testovací dataset, binárna krížová entropická strata bola 0,092 a presnosť 97,13 %.

Mikrofón



(a) Presnosť

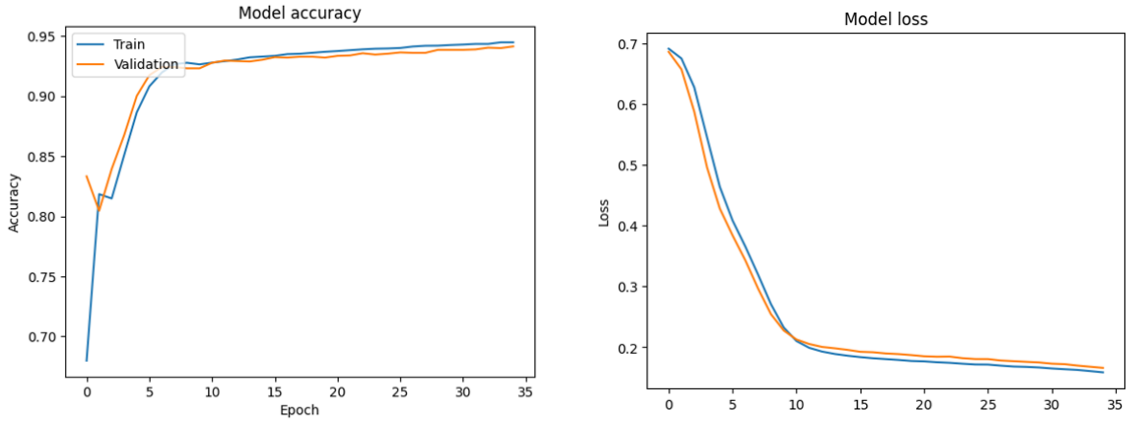
(b) Straty

Obr. 4.15: Presnosť a strata mikrofónu pre binárnu klasifikáciu zo segmentovaných dát

Na rozdiel od snímača akustickej emisie sa CNN pri použití dát z mikrofónu tak rýchlo neustálila, hlavne jej presnosti ako môže byť videné na grafe 4.15. Validačná a testovacia presnosť dosiahli po prvej epoche hodnoty 89,82 % a 83,38 %. Trénovacia presnosť po 35. epochách dosiahla hodnotu 94,19 % a maximálna validačná presnosť bola 93,8 %. Hodnota trénovacej presnosti na konci 35. epóch ale nepredstavuje maximálnu presnoť nakoľko obidve presnosti kolísali signifikantne na začiatku, kde bol zaznamenaný veľký pád presnotí, a pomalé ustálenie v ostatku priebehu. Testovacia presnosť nadobudla hodnoty 94,19 % a binárna krížová entropická strata hodnoty 0,1836.

Dynamický snímač tlaku

Validačná a trénovacia presnosť pre dynamický snímač tlaku dosiahla najmenšie



(a) Presnosť (b) Straty

Obr. 4.16: Presnosť a strata dynamického snímača tlaku pre binárnu klasifikáciu zo segmentovaných dát

presnosti a to 83,32 % a 67,99 %. Validačná a testovacia presnosť po 35. epochách dosiahli hodnoty 94,13 % a 94,47 %. Testovacia presnosť modelu je 95,42 % a binárna krížová entropická strata 0,1416. Na rozdiel od ostatných snímačov z grafov pre dynamický snímač tlaku môžeme sledovať potenciálny trend ďalšieho rastu presnosti a zníženie strát. V tabuľke 4.1 je možne vidieť dosiahnuté validačné, trénovacie a testovacie presnosti a straty pre všetky 3 snímače.

	Validácia		**Trénovanie**		**Testovanie**	
	Presnosť [%]	Straty	Presnosť [%]	Straty	Presnosť [%]	Straty
p_{kis}	94,13	0,1660	94,47	0,1586	95,42	0,1416
AE	96,99	0,1008	96,62	0,1119	97,13	0,092
SP	93,69	0,1929	93,82	0,1986	94,19	0,1836

Tab. 4.1: Výstupy CNN pre binárnu detekciu kavitácie za pomoci segmentovaných dát pre všetky sensory po 35 epochách

Následne bolo rozhodnuté, že by stálo za pokus pustiť na CNN doposiaľ nevidené dáta. Túto funkciu plní testovací dataset, z ktorého doposiaľ výstup bol iba trénovacia presnosť a strata. Za pomoci funkcie predict, ktorá vezme testovací dataset vyberie z neho iba samotné dáta a využije model na jeho klasifikáciu, v module tensorflow bol načítaný model s najlepšími dosiahnutými parametri pre korešpondujúcu CNN a spustený. Tieto hodnoty dávajú väčší náhľad do toho, ako sa bude správať

CNN pre použitie v praxi na neoznačených datasetoch. Výstupom tejto funkcie je takzvaná klasifikačná matica. Tieto matice môžu byť videné v 4.17.

Classification Report:

	precision	recall	f1-score	support
0	0.98	0.97	0.97	661
1	0.97	0.98	0.98	734
accuracy			0.97	1395
macro avg	0.97	0.97	0.97	1395
weighted avg	0.97	0.97	0.97	1395

(a) Akustická emisia

Classification Report:

	precision	recall	f1-score	support
0	0.94	0.96	0.95	626
1	0.97	0.95	0.96	771
accuracy			0.95	1397
macro avg	0.95	0.95	0.95	1397
weighted avg	0.95	0.95	0.95	1397

(b) Dynamický snímač tlaku

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.96	0.94	655
1	0.96	0.93	0.94	740
accuracy			0.94	1395
macro avg	0.94	0.94	0.94	1395
weighted avg	0.94	0.94	0.94	1395

(c) Mikrofón

Obr. 4.17: Klasifikačné reporty CNN binárnej klasifikácie segmentovaných dát

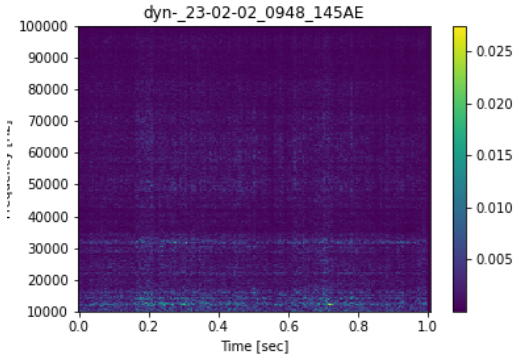
Kde:

- **Precision**(Precíznosť) → Predstavuje schopnosť modelu neoznačiť dáta nesprávnym označením a je definovaná ako pomer správne označených dát k sume správne a nesprávne označených dát jednej triedy.
- **Recall**(Odvolanie) → Schopnosť modelu nájsť všetky dáta jednej triedy (0 - Nekavituje, 1 - Kavituje). Je definovaná ako pomer Správne označených dát k sume správne označených dát jednej skupiny a nesprávne označených dát skupiny druhej.
- **f1 score**(f1 skóre) → Je vážený harmonický priemer preciznosti a odvolania, kde najlepšie skóre je 1 a najhoršie 0. Tento parameter sa vo všeobecnosti

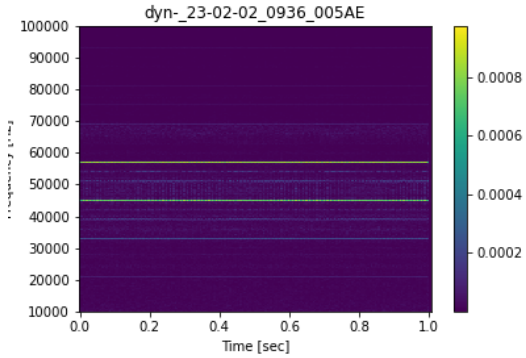
používa ako porovnávací parameter medzi modelmi.
- **Support**(podpora) → Prectavuje počet dát patriacich do danej triedy

Z týchto reportov môže byť videné že najlepšie f1 skóre má snímač akustickej emisie.

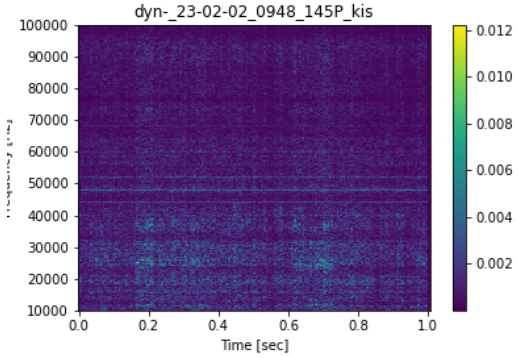
V ďalšom prípade bola binárna klasifikácia aplikovaná na spektografy, ktorých tvorba bola už opísaná v predošlej kapitole. Je tu ale za potrebné poznamenať, že kým natrénovanie CNN pre binárnu klasifikáciu zo segmentovaných dát trvalo vo všeobecnosti hodinu, v prípade binárnej klasifikácie za použitia spektografov sa tento čas predĺžil rámcovo na zhruba 4 hodiny. Tak isto stojí za povšimnutie vykreslenie spektografov pre stavy kavituje a nekavituje, kde je jasne viditeľný rozdiel medzi týmito grafmi a môže v budúcnosti byť použitý ako vstup do CNN pre rozoznávanie kavitácie z obrazov.



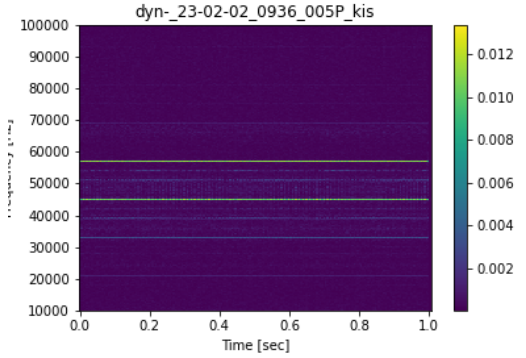
Obr. 4.18: Akustická emisia - Kavituje



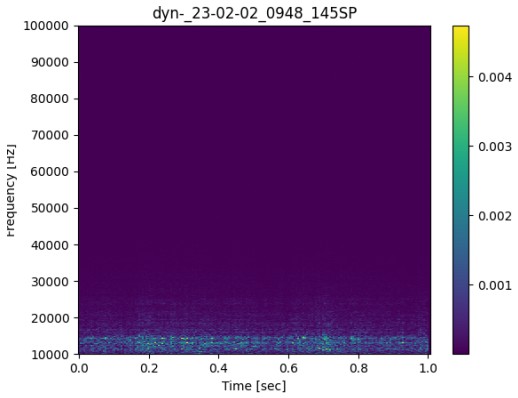
Obr. 4.19: Akustická emisia - Nekavituje



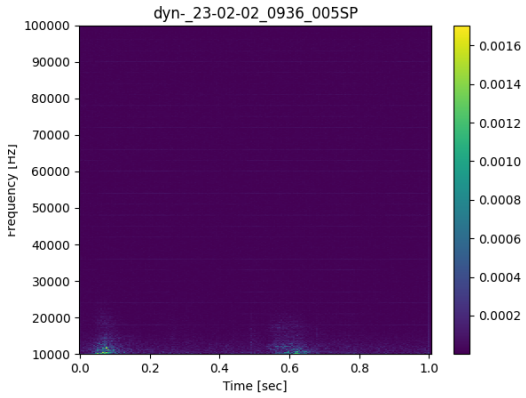
Obr. 4.20: Kisler - Kavituje



Obr. 4.21: Kisler - Nekavituje



Obr. 4.22: Mikrofón - Kavituje

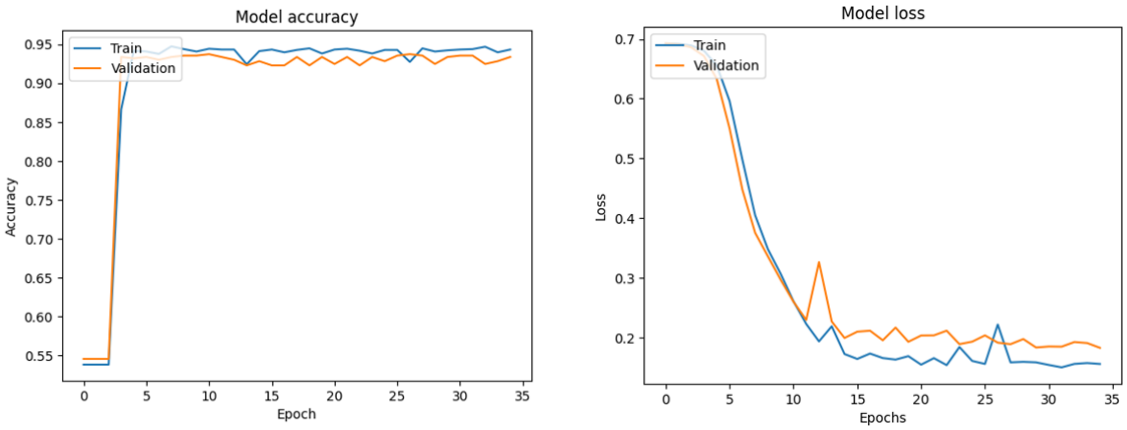


Obr. 4.23: Mikrofón - Nekavituje

Obr. 4.24: Porovnanie Spektografov Kavituje vs Nekavituje pre všetky 3 senzory

Snímač akustickej emisie

Pri porovnaní presnosti modelu pre binárnu klasifikáciu kavitácie s použitím spekto-

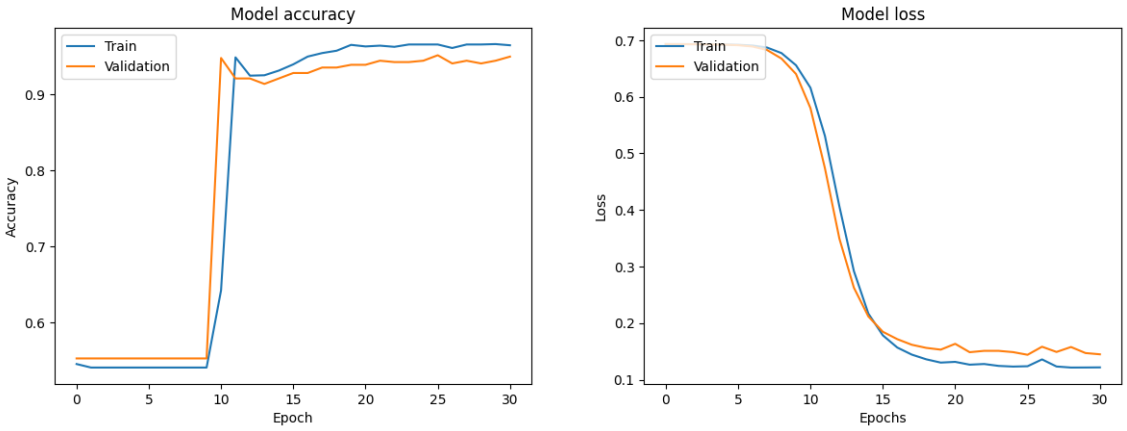


(a) Presnosť (b) Straty

Obr. 4.25: Presnosť a strata snímača akustistej emisie pre binárnu klasifikáciu zo spektografov

grafov oproti segmentácii môže byť videné počiatočné ustálenie na nízkych presnostiach a potom obrovský nárast v samotnej presnosti. Maximálna validačná presnosť pre snímač akustickej emisie je 93,72 % a trénovacia presnosť po 35tich epochách je 94,34 %. Celková testovacia presnosť a straty sú 95 % a 0,176.

Mikrofón



(a) Presnosť (b) Straty

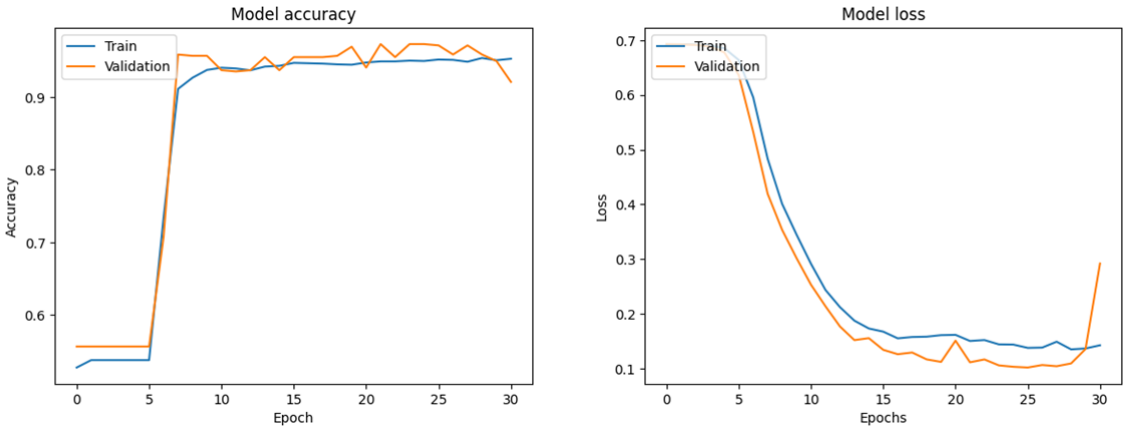
Obr. 4.26: Presnosť a strata mikrofónu pre binárnu klasifikáciu zo spektografov

Tak, ako v prípade snímača akustickej emisie, aj v tomto prípade môže byť

sledované počiatočné ustálenie na nízkych presnostiach avšak trvajúce až po dobu desiatich epóch. Maximálna validáčná presnoť tohto modelu dosiala hodnotu 95,15 % a trénovacia presnsoť po 31 epochách 96,47 %. Testovacia presnosť a straty 96,07 % a 0,1407. Pre tento model bolo dosiahnutých iba 31 epóch oproti plánovanému počtu 35, z dôvodu nemeniacej sa validačnej presnosti po dobu 5 epóch. Následne bol systémom terminovaný výpočet a uložený najlepší model.

Dynamický snímač tlaku

Maximálna dosiahnutá validačná presnosť modelu bola 97,31 %. Trénovacia pres-



(a) Presnosť (b) Straty

Obr. 4.27: Presnosť a strata dynamického snímača tlaku pre binárnu klasifikáciu zo spektografov

nosť po 31 epochách bola 95,29 %. Testovacia presnsoť a straty nadobudli hodnoty 90,36 % a 0,3509. Ako je možné vidieť z grafu, tak isto ako pre mikrofón, výpočet bol ukončený systémom po 31 epochách, z dôvodu nemeniacej sa validačnej presnoti. Porzoruhodný je aj nárast strát validačných na poslednej epoche a nim odpovedajúci pokles validačnej presnosti.

V tabuľke 4.2 je možno vidieť dosiahnuté validačné, trénovacie a testovacie presnosti a straty pre všetky 3 snímače.

	Validácia		**Trénovanie**		**Testovanie**	
	Presnosť [%]	Straty	Presnosť [%]	Straty	Presnosť [%]	Straty
p_{kis}	93,10	0,2918	95,29	0,1423	90,36	0,0,3509
AE	93,36	0,1827	94,31	0,1556	95,00	0,1760
SP	94,97	0,1448	96,47	0,1217	96,07	0,1407

Tab. 4.2: Výstupy CNN pre binárnu detekciu kavitácie za pomoci spektografov pre všetky sensory na koci poslednej epochy.

Klasifikačné matice pre spektografy môžu byť videnné nižšie.

Classification Report:

	precision	recall	f1-score	support
0.0	0.93	0.95	0.94	132
1.0	0.95	0.94	0.95	148
accuracy			0.94	280
macro avg	0.94	0.94	0.94	280
weighted avg	0.94	0.94	0.94	280

(a) Akustická emisia

Classification Report:

	precision	recall	f1-score	support
0.0	0.95	0.98	0.96	137
1.0	0.98	0.95	0.96	143
accuracy			0.96	280
macro avg	0.96	0.96	0.96	280
weighted avg	0.96	0.96	0.96	280

(b) Dynamický snímač tlaku

Classification Report:

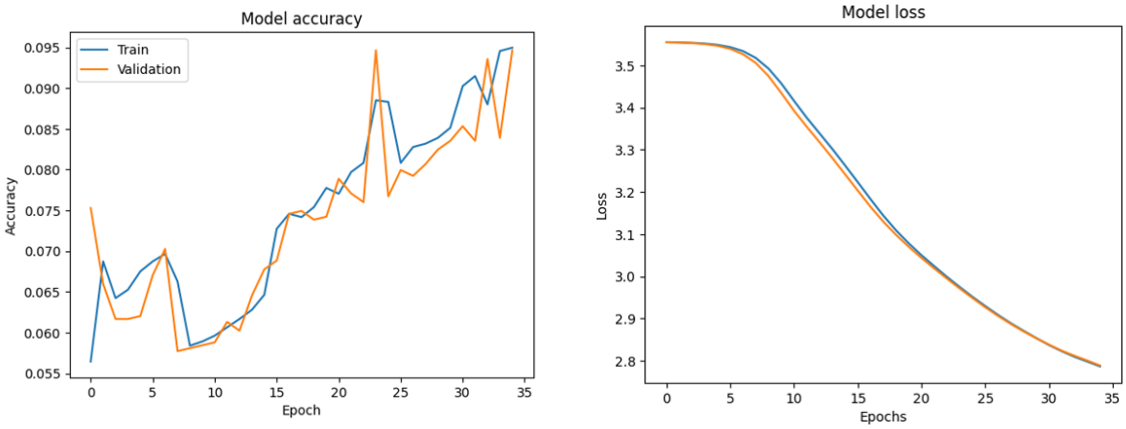
	precision	recall	f1-score	support
0.0	0.96	0.96	0.96	141
1.0	0.96	0.96	0.96	139
accuracy			0.96	280
macro avg	0.96	0.96	0.96	280
weighted avg	0.96	0.96	0.96	280

(c) Mikrofón

Obr. 4.28: Klasifikačné reporty CNN binárnej klasifikáciu zo spektografov

4.5.2 Klasifikácia prietoku

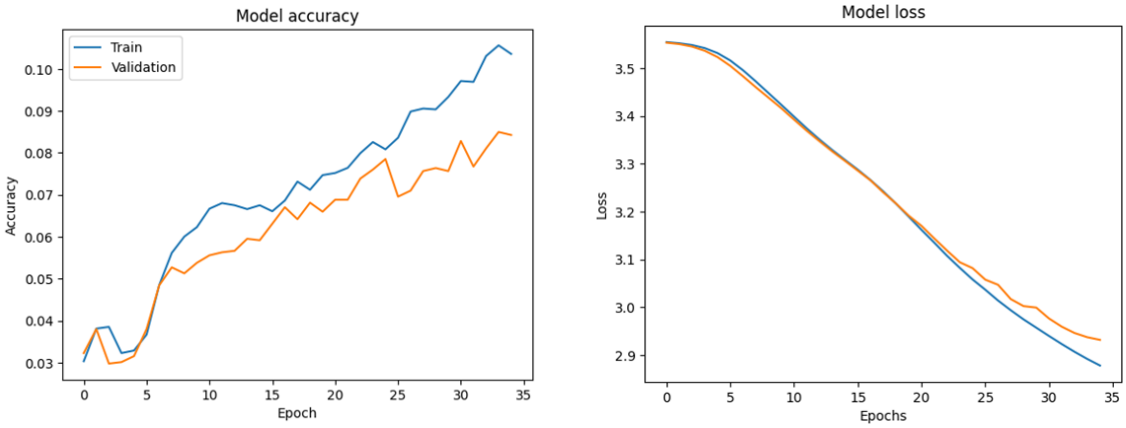
Pre prípad klasifikácie prietoku nastal značný problém v samotnej detekcii a klasifikácii prietoku. Nakoľko prietok bol veľmi ovplyvňovaný viacerými parametrami, ako presnosť snímača, odpory trate, kavitácia a množstvo klasifikovaných prietokov. Toto môže byť videné v nasledovných grafoch 4.29, 4.30 a 4.31.



(a) Presnosť

(b) Straty

Obr. 4.29: Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát

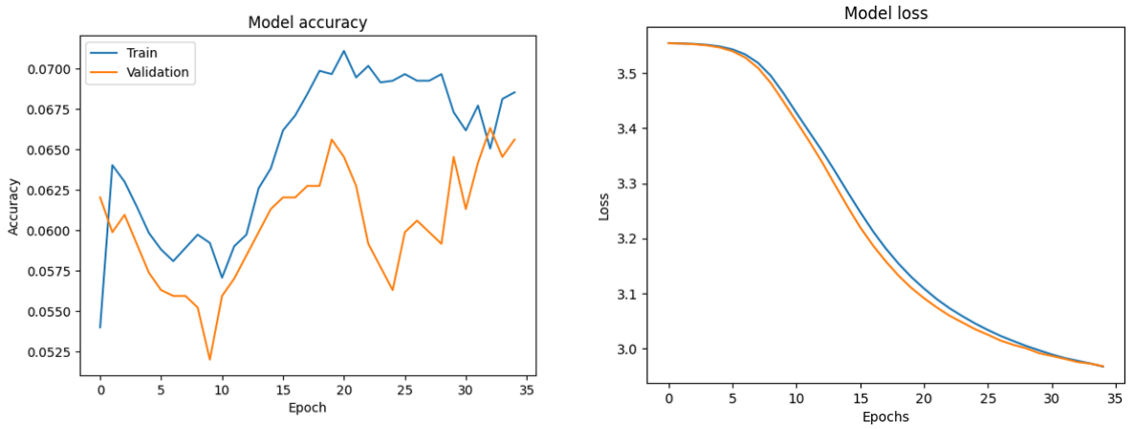


(a) Presnosť

(b) Straty

Obr. 4.30: Presnosť a strata dynamického snímača tlaku pre klasifikáciu prietoku zo segmentovaných dát

Z grafov 4.29, 4.30 a 4.31 je možné vidieť, že po 35 epochách bola dosiahnutá malá presnosť. To znamená, že systém nebol schopný nájsť uspokojivé riešenie s daným nastavením parametrov neurónovej siete. Pre kompletnosť riešenia bolo následne

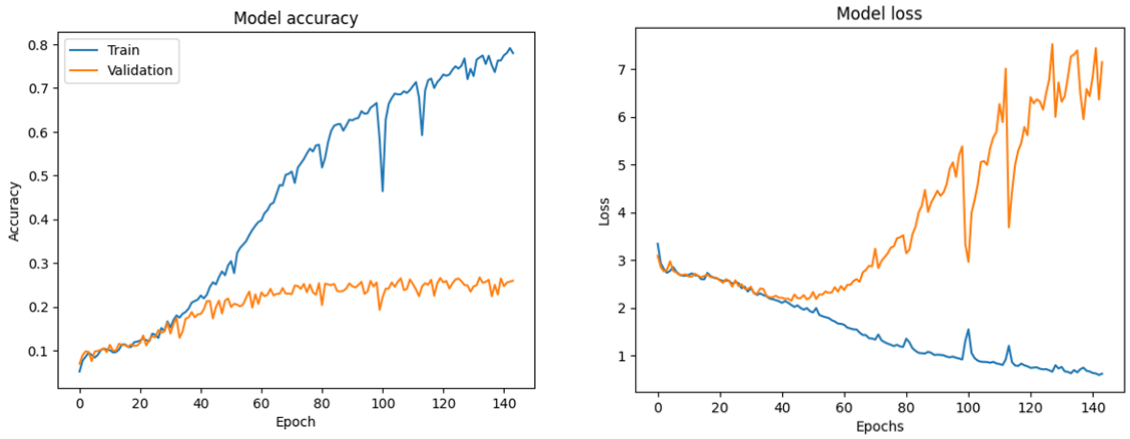


(a) Presnosť

(b) Straty

Obr. 4.31: Presnosť a strata mikrofónu pre klasifikáciu prietoku zo segmentovaných dát

otestované spustenie systému, pre väčší počet epóch. Výsledky môžu byť videné v Obr. 4.32.



(a) Presnosť

(b) Straty

Obr. 4.32: Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát 150 epóch

Z týchto výsledkov je evidentné, že dochádza k tzv. preplnení neurónovej siete. V grafoch 4.32 môžu byť videné presnosti pre snímač akustickej emisie pre 150 epóch, kde hodnota trénovacej presnosti značne presahuje hodnotu validačnej presnosti. To znamená že, neurónová sieť namiesto toho, aby sa učila na trénovacích dátach, si ich začína pamäta. Tento fakt potvrdzuje aj veľká hodnota validačnej straty oproti trénovacej strate.

Pre klasifikáciu prietoku za pomoci spektografov boli znova za pomoci programu v Pythone vytvorené spektografy. Z týchto spektografov boli následne uložené matice obsahujúce frekvenčno-časovú závislosť. Spektografy boli vytvorené pre každý piaty dátový záznam. V porovnaní so spektografmi pre klasifikáciu kavitácie, kde bolo odfiltrovaných prvých 10 kHz, v tomto prípade nebola filtrácia prevedená, nakoľko cieľom je klasifikovať prietoky a toto frekvenčné spektrum obsahuje frekvenciu čerpadla, ktoré môže napomôcť pri klasifikácii prietokov.

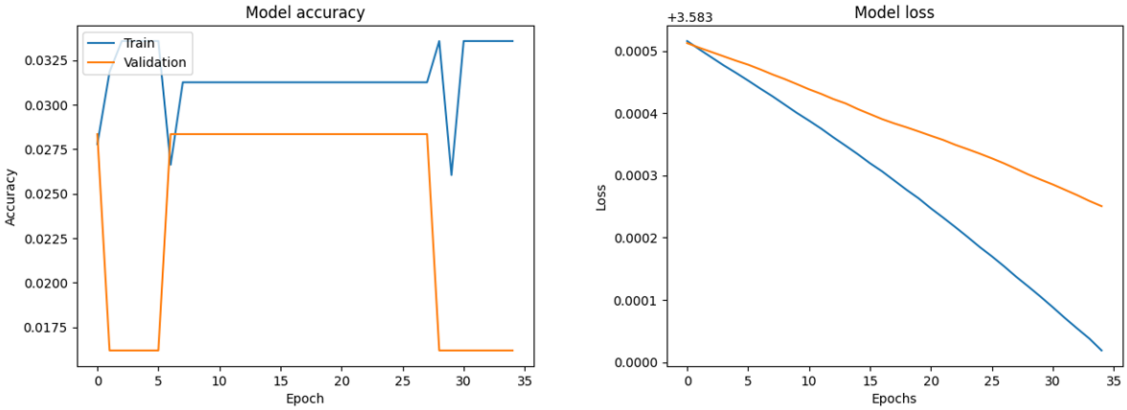
Výsledné matice majú tvar 1015, 199, 1. Číslo 1 reprezentuje počet kanálov, pre príklad, farebný obrázok by mal počet vstupných kanálov 3 - červená, zelená a modrá a čiernobiely zase 2. Tieto matice sú následne načítané ďalším programom a zložené do 3D matíc. Tieto matice sú označené podľa súboru, v ktorom sa nachádzajú, v tomto prípade 1 - 35 zamiešané a rozdelené na trénovacie a trénovacie datasety.

Na koľko sú tu vstupy 2D matice o relatívne veľkých rozmeroch, bolo za potrebné redukovať počet trenovateľných parametrov, aby bolo neurónovú sieť možné spustiť a to z dôvodu veľkého zaťaženia hardwaru. Nastavenia a počet trénovateľných parametrov je možné vidieť v Obr. 4.33.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 1013, 197, 2)	20
conv2d_1 (Conv2D)	(None, 1011, 195, 2)	38
max_pooling2d (MaxPooling2D)	(None, 506, 98, 2)	0
conv2d_2 (Conv2D)	(None, 504, 96, 2)	38
conv2d_3 (Conv2D)	(None, 502, 94, 2)	38
max_pooling2d_1 (MaxPooling2D)	(None, 251, 47, 2)	0
conv2d_4 (Conv2D)	(None, 249, 45, 4)	76
conv2d_5 (Conv2D)	(None, 247, 43, 4)	148
conv2d_6 (Conv2D)	(None, 245, 41, 4)	148
max_pooling2d_2 (MaxPooling2D)	(None, 123, 21, 4)	0
conv2d_7 (Conv2D)	(None, 121, 19, 8)	296
conv2d_8 (Conv2D)	(None, 119, 17, 8)	584
conv2d_9 (Conv2D)	(None, 117, 15, 8)	584
max_pooling2d_3 (MaxPooling2D)	(None, 59, 8, 8)	0
conv2d_10 (Conv2D)	(None, 57, 6, 16)	1168
conv2d_11 (Conv2D)	(None, 55, 4, 16)	2320
conv2d_12 (Conv2D)	(None, 53, 2, 16)	2320
max_pooling2d_4 (MaxPooling2D)	(None, 14, 1, 16)	0
flatten (Flatten)	(None, 224)	0
dense (Dense)	(None, 32)	7200
dense_1 (Dense)	(None, 64)	2112
dense_2 (Dense)	(None, 36)	2340

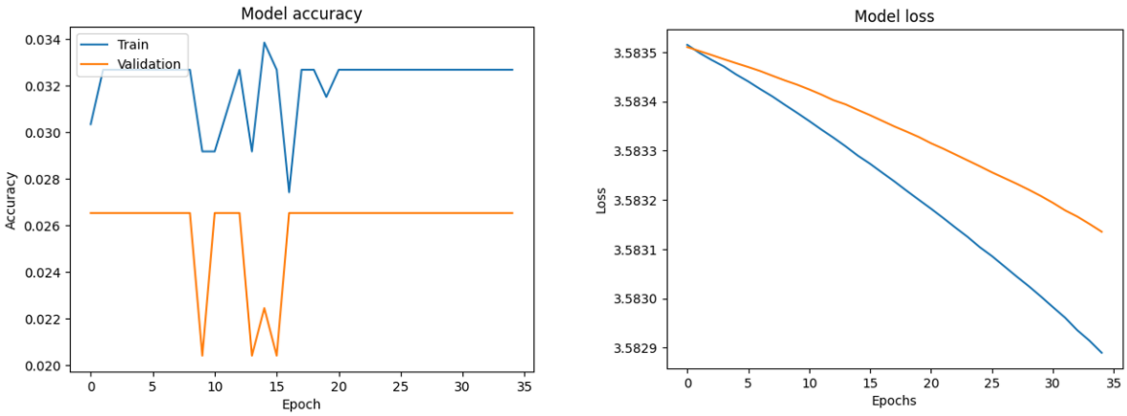
Total params: 19,430
Trainable params: 19,430
Non-trainable params: 0

Obr. 4.33: Nastavenie neurónovej siete pre klasifikáciu prietoku zo spektografov



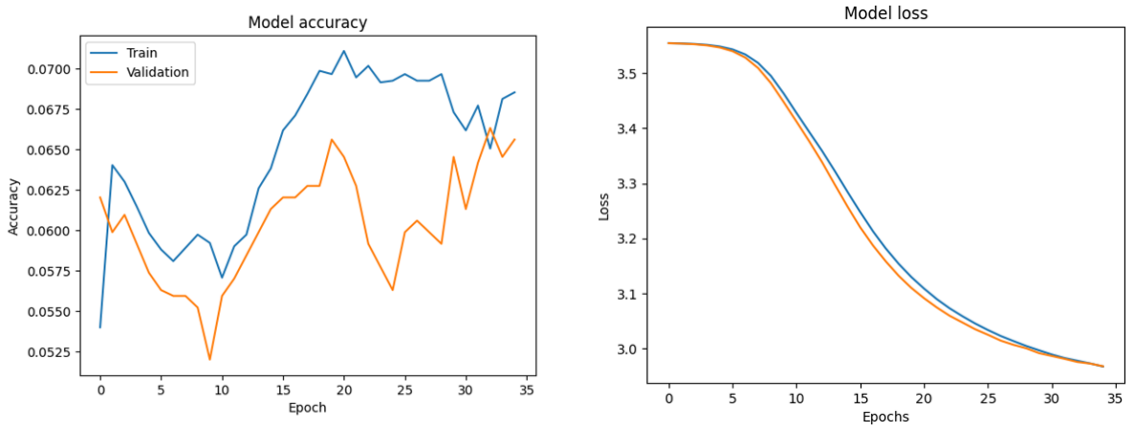
(a) Presnosť (b) Straty

Obr. 4.34: Presnosť a strata mikrofónu pre klasifikáciu prietoku zo spektografov



(a) Presnosť (b) Straty

Obr. 4.35: Presnosť a strata dynamického snímača tlaku pre klasifikáciu prietoku zo spektografov



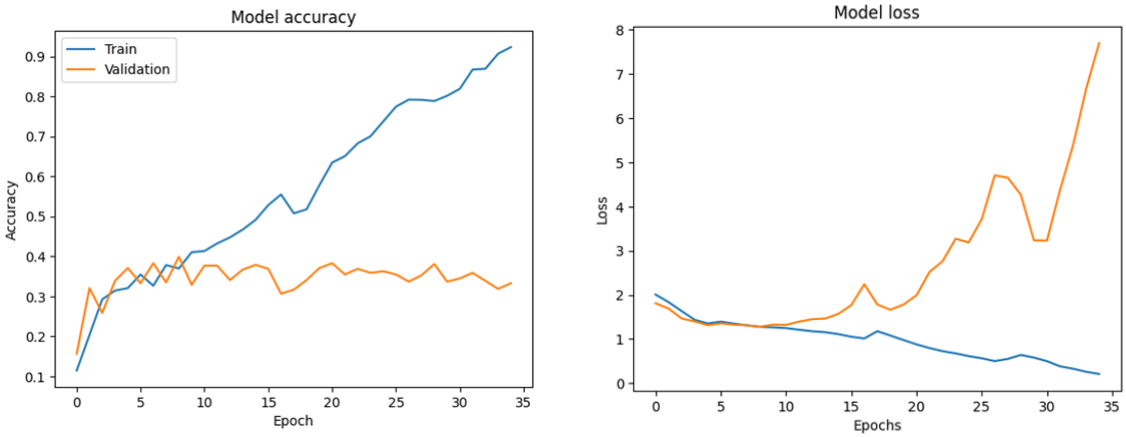
(a) Presnosť

(b) Straty

Obr. 4.36: Presnosť a strata snímača akustickej emisie pre klasifikáciu prietoku zo segmentovaných dát

Z týchto výsledkov je vidieť podobnosť s výsledkami pre kasifikáciu prietokov zo surových dát. Žiadna validačná presnosť pre akýkoľvek zo senzorov nepresiahla 5 %. Uvažovanými príčinami týchto nízkych presností sú 2 možnosti; Ako prvá je samotná testovacia trať, ktorá bola predovšetkým konceptovaná na sledovanie kavitácie. Jej rozmernosť a celková dĺžka predstavuje veľký odpor pre dané prietoky a tak isto aj nepresnosť prietokomeru predstavuje problém. Táto presnosť je ovplyvňovaná odpormi v trase, ale aj samotnou kavitáciou, ktorá tu nastáva. Druhou príčinou sú samotné dáta. Kvôli nepresnosti snímača sú jednotlivé merané prietoky (celkovo 35) nepresné. Dochádza k prelínaniu hodnôt prietoku z jedného bodu do druhého. Toto je tak isto spôsobené tým, že samotný prietok bol nastavený ako plávajúci parameter meniaci sa s okolitými podmienkami, a preto sa hodnoty pre jeden bod nezhodujú s tým istým bodom pre iný tlak v zbernej nádrži.

Táto druhá príčina môže byť overená relatívne jednoducho, a to redukciou počtu prietokov. Boli vzaté originálne prietoky pre snímač akustickej emisie a redukované na jednu pätinu. To znamená, že bol vzatý každý piaty prietok z pôvodne zvyžovaných 35. Tento prístup by mal odstrániť prelínanie záznamov prietokov.



(a) Presnosť

(b) Straty

Obr. 4.37: Presnosť a strata mikrofónu pre klasifikáciu prietoku zo segmentovaných dát (Redukované)

Z grafov 4.37 je možné vidieť, že metóda redukcie počtu klasifikačných tried nespravila obrovský rozdiel v celkovej presnosti. Dosiahnutá validačná presnosť dosiahla maximálnu hodnotu 39,88 % a testovaciu presnosť 40,8 %. Aj keď tieto hodnoty sú vyššie o zhruba 15 % oproti hodnotám v grafe 4.32 je táto presnosť pre reálnu aplikáciu stále moc nízka.

Záver

Ako bolo zmienené, cieľom tejto diplomovej práce bolo využitie strojného učenia pre diagnostiku vodných strojov. Klasifikácia Kavituje-Nekavituje a klasifikácia prietoku v meranej trase. Cielom ttejto práce bolo tak isto využitie strojného učenia pre detekciu poruchy na vodnom stroji, pohužial počas meriacej kampane zameranej na kavitácia sa pri najvýžších natočenia lopatka vplivom prietoku prehla a nebolo možné ju naďalej využívať. Meranie celej dátovej sady trvalo približne 20 pracovných dní.
Pre splnenie týchto cieľov bola za pomoci online prostredia Google Colab vytvorená neurónová sieť, konkrétne konvolunčná neurónová sieť, ktorá je často používaná pre klasifikáciu obrazov. V tomto prípade bola ale aplikovaná na rozličné dáta získané z 3 senzorov na meracej trase. Tieto dáta boli vpustené do neurónovej siete dvomi spôsobmi, a to ako surové a za pomoci ich úpravy vo forme spektografov.

Výsledky pre klasifikáciu kavitácie môžeme vidieť v obrázkoch 4.14, 4.15, 4.16 pre klasifkáciu kavitácie zo surových dát a obrázkoch 4.25, 4.26 a 4.27. Z týchto grafov presností a strát pre jednotlivé snímače je možné vidieť, že neurónová sieť funguje dobre pre binárnu klasifikáciu kavitácie pre surové dáta aj spektografy. Testovacie presnosti všetkých snímačov presiahli 90 %. Porovnanie všetkých snímačov a metód môže byť videné v nasledovnej tabulke.

	Validácia		**Trénovanie**		**Testovanie**	
	Presnosť [%]	Straty	Presnosť [%]	Straty	Presnosť [%]	Straty
Surové dáta						
p_{kis}	94,13	0,1660	94,47	0,1586	95,42	0,1416
AE	96,99	0,1008	96,62	0,1119	97,13	0,092
SP	93,69	0,1929	93,82	0,1986	94,19	0,1836
Spektofrafy						
p_{kis}	93,10	0,2918	95,29	0,1423	90,36	0,0,3509
AE	93,36	0,1827	94,31	0,1556	95,00	0,1760
SP	94,97	0,1448	96,47	0,1217	96,07	0,1407

Tab. 4.3: Porovnanie všetkých snímačov pre klasifikáciu kavitácie

Z tabuľky 4.3 je možné vidieť, že snímač s najlepšou testovacou presnosťou po 35 epochách bol snímač akustickej emisie zo surových dát s presnosťou 97,13 %. Kombinácia s najmenšou dosiahnutou presnosťou je snímač dynamického tlaku a

spektografy, ktorá dosiahla presnosť 90,36 %. Porovnanie f1 skóre pre tak isto binárnu klasifikáciu kavitácie je možné vidieť v tabuľke 4.4.

Snímač	**f1 Skóre**	
	0	1
Surové dáta		
AE	0,97	0,98
P_kis	0,95	0,96
SP	0,94	0,94
Spektografy		
AE	0,94	0,95
P_kis	0,96	0,96
SP	0,96	0,96

Tab. 4.4: Porovnanie f1 Skóre pre binárnu klasifikáciu kavitácie (0 - Kavituje, 1 - Nekavituje)

Táto tabuľka potvrdzuje, že daná neurónová sieť našla najlepšie riešenie pre surové dáta zo snímača akustickej emisie s hodnotou f1 skóre 0,98. Toto je logický výsledok, nakoľko celý princíp snímania akustickej emisie je meranie voľnej energie v tuhom telese. Táto energia je v tomto prípade uvoľnená do potrubia pri implózii kavitačných bublín, ktoré vyvolávajú tlakovú vlnu a táto následne namáha potubnú trasu. Toto samozrejme neznamená, že ostatné kombinácie snímačov a metód sú nepoužitelné, práve naopak. Všetky kombinácie dosiahli f1 skóre nad 0,94 a z technického pohľadu môžu byť používané v praxi s minimálnou predpokladanou chybou.

V prípade klasifikácie prietoku, výsledky tejto práce neboli ani z ďaleka také pozitívne, ako v prípade klasifikácie kavitácie. Celkovo sa prešlo 35 prietokov počas celej etapy merania. Jedným z najväčších problémov pre klasifikáciu prietokov z nameraných dát, bol spôsob nastavenia prietoku. Na rozdiel od hodnôt tlaku v zbernej nádrži a uhlu natočenia lopatky, ktoré boli nastavované priamo, bol prietok nastavovaný ako plávajúci parameter, závislý na okolitých podmienkach ako tlak v zbernej nádrži a kavitácii. Z tohto dôvodu nie sú hodnoty prietokov rovnaké pre všetky merané pracovné body. Ďalším problémom bola samotná kavitácia, ktorá prinášala do merania prietoku nelinaritu a prelínanie jednotlivých prietokov. Grafy presností a strát z obrázkov 4.29, 4.31, 4.30, 4.36, 4.34 a 4.35 je možné vidieť nízka presnosť a vysoká strata pre všetky kombinácie snímačov a úpravy dát. Táto presnosť v žiadnom prípade nedosiahla ani hodnoty 10 %. V surových dátach oproti

spektografom, ale bol pozorovaný trend rastu presnosti. Z tohto dôvodu bolo odskúšané pustenie neurónovej siete pre väčší počet epóch a to 150. Výsledný graf 4.32 poukazuj,e že aj keď trénovacia presnosť narástla na hodnoty pohybujúce sa okolo 80 %, samotná testovacia presnosť nestúpla na 30 %. Tento fakt poukazuje na skutočnosť, že neurónová sieť sa nesnaží nájsť najlepšie riešenie, ale začína si pamätať vstupné dáta. Inými slovami, počet epóch už je natoľko vysoký, že neurónová sieť si zapamätala kompletný trénovací dataset a dochádza k takzvanému pretrénovaniu neurónovej siete.

Ďašie potenciálne riešenie, ktoré bolo odskúšané, bola redukcia počtu klasifikovných prietokov z 35 na 7. Logika tohto prístupu bolo zamedzenie prelínania jednotlivých prietokov a tým lepšia klasifikácia. Z grafov 4.37 môže byť videné, že bola dosiahnutá vyššia celková testovacia presnosť 40 %. Musí sa vziať do úvahy, že originálny počet klasifikačných tried bol redukovaný na pätinu, a tým pádom nie je dosiahnutá klasifikácia celkového prietoku.

Ako bolo už zmienené, dané meranie nebolo priamo orientované na presné meranie prietoku. Pre budúce možné riešenie tejto problematiky by bolo ideálnejšie sa zamerať čisto na meranie prietoku, kde by bola zostavená menšia testovacia trať, v ktorej by sa prešli požadované prietoky a celková doba merania každého prietoku by bola dlhšia ako v tejto práci. Ďalšou cestou môže byť aj iný princíp úpravy dát, či už jednoduchou Fourierovov transformáciou, ktorá by potencionálne mohla jednoduchšie zachytiť frekveciu čerpadla ako spektografy, alebo priemerovaním hodnôt prietoku v každom pracovnom bode a rozdelenie týchto priemerovaných hodnôt do tried.

Pri vypracovaní tejto diplomovej práce boli opísané výhody použitia neurónových sietí pre klasifikačné úlohy za pomoci prostredí ako Google Collab a modulov ako tensorflow a keras, kde ktokoľvek s miernou znalosťou neurónových sietí schopný si naprogramovať vlastnú neurónovú sieť a aplikovať ju na požadované dáta. Boli však objavené aj nedostatky. Jedným z nich je potrebné množstvo hlavne trénovacích dát potrebných pre dobré natrénovanie neurónovej siete a druhým je hardwarové zaťaženie. Aj keď Google Collab poskytuje isté zdroje, viackrát sa prejavila skutočnosť, kedy musel byť redukovaný počet trénovateľných parametrov z dôvodu nedostatku pamäte RAM. Celkovo hardwarové zaťaženie je závislé na daných požiadavkách, ktoré na neurónku pokladáme. Celkovo, aj keď Google Collab poskytuje dostatočné množstvo prostriedkov pre bežnú aplikáciu, v profesionálnej aplikácii, kde kvantum dát by presiahalo terabajty by bolo odporúčané zostaviť vlastnú počítačovú stanicu s dostatočnými prostriedkami pre hodnotenie danej úlohy.

Ako bolo zmienené, cieľom tejto diplomovej práce bolo stanovenie prevádzkových parametrov (Kavitácia, Prietok, Porucha) vodných strojov. Klasifikácia Kavituje alebo Nekavituje dosiahla veľmi dobrých výsledkov pre obidve metódy úpravy dát aj všetky 3 použité snímače. Klasifikácia Prietok dosiahla určitých presností, ale nie dostatočných pre akúkoľvek aplikáciu v praxi. Ale predpokladá sa, že lepším spôsobom merania a úpravou dát a aj využitím inej architektúry neurónovej siete by sa mohlo dosiahnuť aj akceptovateľných presností.

Literatúra

[1] SMETANA, Ctirad. *Hluk a vibrace: měření a hodnocení*. 1. Praha: Sdělovací technika, 1998. ISBN 80-901-9362-5.

[2] The most common instrumentation microphone, a condenser microphone, operates on a capacitive design. In: *Www.ni.com* [online]. Austin, Texas, USA: © 2023 NATIONAL INSTRUMENTS, 2023 [cit. 2023-04-28]. Dostupné z: https://www.ni.com/cs-cz/shop/data-acquisition/sensor-fundamentals/measuring-sound-with-microphones.html

[3] IDA, Nathan. *Ensors, Actuators, and Their Interfaces - A Multidisciplinary Introduction* [online]. Institution of Engineering and Technology (The IET), 2014 [cit. 2023-04-12]. ISBN 978-1-61353-006-1. Dostupné z: https://app.knovel.com/hotlink/khtml/id:kt0110NT22/sensors-actuators-their/capacitive-microphones

[4] PŘIBÁN, Miroslav. *Akustická emise (AE) - teorie a praxe provozních kontrol konstrukcí*. [online]. 25.5.2012, 1 - 25 [cit. 2023-02-18]. Dostupné z: http://www.crr.vutbr.cz/offline/SYSTEM/FILES/BROZURA_06_1205.PDF

[5] KOPEC, Bernard. *Nedestruktivní zkoušení materiálů a konstrukcí: (nauka o materiálu IV)*. Brno: Akademické nakladatelství CERM, 2008. ISBN ISBN978-80-7204-591-4.

[6] *Sensor technology handbook* [online]. Burlington: Elsevier, c2005 [cit. 2023-04-12]. Newnes. ISBN 978-0-7506-7729-5. Dostupné z: https://app.knovel.com/hotlink/pdf/id:kt0047U5G1/sensor-technology-handbook/longitudinal-transverse

[7] Piezoelectric pressure sensor. In: *Kistler* [online]. Winterthur, Švajčiarsko: Kistler Group 2023, 20233 [cit. 2023-04-28]. Dostupné z: https://www.kistler.com/CZ/en/piezoelectric-pressure-sensor/C00000138

[8] JUNG, Alexander. *Machine Learning*. Singapore: Springer, 2022. ISBN 9789811681929.

[9] SOROKIN, Alexander a David FORSYTH. Utility data annotation with Amazon Mechanical Turk. In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops* [online]. 2008 [cit. 2023-03-22]. ISBN 9781424423408. Dostupné z: https://experts.illinois.edu/en/publications/utility-data-annotation-with-amazon-mechanical-turk

[10] Supervised Machine Learning. In: *Javatpoint* [online]. Noida, UP, 201301, India: JavaTpoint [cit. 2023-04-28]. Dostupné z: https://www.javatpoint.com/supervised-machine-learning

[11] JAMES, Gareth, Daniela WITTEN, Trevor HASTIE a Robert TIBSHIRANI. *An introduction to statistical learning: with applications in R* [online]. New York: Springer, [2013] [cit. 2023-03-23]. Springer texts in statistics. ISBN 978-1-4614-7138-7. Dostupné z: https://static1.squarespace.com/static/5ff2adbe3fe4fe33db902812/t/6009dd9fa7bc363aa822d2c7/1611259312432/ISLR+Seventh+Printing.pdf

[12] MARČEK, Milan, Lucia PANČÍKOVÁ a Dušan MARČEK. *Ekonometria a soft computing*. Žilina: Žilinská univerzita v Žiline, 2008. ISBN 978-80-8070-7460.

[13] Logistic regression. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2023-04-28]. Dostupné z: https://en.wikipedia.org/wiki/Logistic_regression

[14] HASTIE, Trevor, Robert TIBSHIRANI a Jerome FRIEDMAN. *The elements of statistical learning: data mining, inference, and prediction* [online]. 2nd ed. New York: Springer, c2009 [cit. 2023-04-07]. Springer series in statistics. ISBN 978-0-387-84858-7.

[15] What is Decision Tree Analysis? How to Create a Decision Tree. In: *Gliffy* [online]. Perforce Software, March 17, 2021 [cit. 2023-04-28]. Dostupné z: https://www.gliffy.com/blog/decision-tree-analysis

[16] Decision Trees. *Decision Trees* [online]. Armonk, New York, USA: IBM Corporation [cit. 2023-04-30]. Dostupné z: https://www.ibm.com/topics/decision-trees

[17] The Artificial Neural Networks handbook: Part 1. In: *Datasciencecentral* [online]. TechTarget, 2018 [cit. 2023-04-28]. Dostupné z: https://www.datasciencecentral.com/the-artificial-neural-networks-handbook-part-1/

[18] A Beginners Guide to Unsupervised Learning. In: *Medium* [online]. Analytics Vidhya, 2019 [cit. 2023-04-28]. Dostupné z: https://medium.com/analytics-vidhya/beginners-guide-to-unsupervised-learning-76a575c4e942

[19] K-means clustering. In: *Bookdown* [online]. Sydney: A.Teixeira-Pinto, 2021 [cit. 2023-04-28]. Dostupné z: https://bookdown.org/tpinto_home/Unsupervised-learning/k-means-clustering.html

[20] ALPAYDIN, Ethem. *Introduction to machine learning* [online]. 2nd ed. Massachusetts: MIT Press, c2010 [cit. 2023-03-28]. ISBN 978-0-262-01243-0. Dostupné z: https://dl.matlabyar.com/siavash/ML/Book/Ethem%20Alpaydin-Introduction%20to%20Machine%20Learning-The%20MIT%20Press%20(2014).pdf

[21] Reinforcement Learning 101. In: *Medium* [online]. Towards Data Science, 2018 [cit. 2023-04-28]. Dostupné z: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

[22] BHATT, Shweta. Reinforcement Learning 101: 1. What is Reinforcement Learning? How does it compare with other ML techniques?. *Towards Data Science* [online]. 2018 [cit. 2023-04-30]. Dostupné z: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

[23] Pressure sensors: High pressure sensors, M10 front-sealing, for high speed events up to 6000 bar (87000 psi), AEP-97 / 6215. In: *KISTLER* [online]. Winterthur , Switzerland: Kistler, 2023 [cit. 2023-05-11]. Dostupné z: https://www.kistler.com/INT/en/cp/high-pressure-sensors-m10-front-sealing-for-high-speed-events-up-to-6000-bar-87000-psi-aep-97-6215/P0000443

[24] ICP® ELECTRET ARRAY MICROPHONE: Model 130A24. In: *PCB Piezoelectronics* [online]. Huckelhoven, Germany: PCB, 2023 [cit. 2023-05-11]. Dostupné z: https://www.pcb.com/products?m=130A24

[25] OPTIFLUX 4100: Electromagnetic flowmeter for standard applications with abrasive and aggressive liquids. In: *OPTIFLUX* [online]. Duisburg Germany: OPTIFLUX, 2022 [cit. 2023-05-11]. Dostupné z: https://krohne.com/en/products/flow-measurement/flowmeters/electromagnetic-flowmeters/optiflux-4100

[26] Snímač AE: typ MDK-13 / MDK-13AS. In: *Dakel* [online]. Rpety: Dakel, c 2006-2023 [cit. 2023-05-11]. Dostupné z: https://www.dakel.cz/index.php?pg=prod/sens/mdk13

[27] HOSAMELDIN, Ahmed a Asoke K. NANDI. *Condition monitoring with vibration signals: compressive sampling and learning algorithms for rotating machines*. Hoboken: Wiley, 2020. ISBN 978-1-119-54462-3.

[28] SHAHZAIB, Noor. The Convolution Parameters Calculation. In: *Medium* [online]. London: Medium, 2020 [cit. 2023-05-11]. Dostupné z: https://medium.com/@shahzaibnoor40/the-convolution-parameters-calculation-b2394da8dd59

[29] What are convolutional neural networks?. *IBM* [online]. New York: Copyright IBM Corporation 1994, 2023. [cit. 2023-04-16]. Dostupné z: https://www.ibm.com/topics/convolutional-neural-networks

[30] PISIT, J. DL : Basic Concept of CNN: Part 4.1 of Deep Learning Specialization. In: *Medium* [online]. London: Sum up As A Service, 2020 [cit. 2023-05-11]. Dostupné z: https://medium.com/s-a-a-s/dl-basic-concept-of-cnn-2ef4fc9b039b

[31] YINGGE, Huo, Imran ALI a Kang-Yoon LEE. Deep Neural Networks on Chip - A Survey. *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)* [online]. IEEE, 2020, 2020, 589-592 [cit. 2023-05-11]. ISBN 978-1-7281-6034-4. Dostupné z: doi:10.1109/BigComp48618.2020.00016

[32] UNZUETA, Diego. Fully Connected Layer vs. Convolutional Layer: Explained. In: *Builtin* [online]. Chicago: Built In, c2023 [cit. 2023-05-11]. Dostupné z: https://builtin.com/machine-learning/fully-connected-layer

[33] Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *Cornell University* [online]. 2018, **2018** [cit. 2023-04-30]. Dostupné z: doi:https://doi.org/10.48550/arXiv.1811.03378

[34] Logistic function. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2023-05-11]. Dostupné z: https://en.wikipedia.org/wiki/Logistic_function

[35] Hyperbolický tangens. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2023-05-11]. Dostupné z: https://cs.wikipedia.org/wiki/Hyperbolick%C3%BD_tangens

[36] Tensorflow-keras-flask-app. In: *GitHub* [online]. GitHub, 2017 [cit. 2023-05-11]. Dostupné z: https://github.com/Verdagio/Tensorflow-keras-flask-app

[37] Initialization Of Deep Networks Case of Rectifiers. In: *Jefkin* [online]. jefkin, 2016 [cit. 2023-05-11]. Dostupné z: https://www.jefkine.com/deep/2016/08/08/initialization-of-deep-networks-case-of-rectifiers/

[38] Understanding Loss Function in Deep Learning. In: *Analytics Vidhya* [online]. 2022, 2022 [cit. 2023-05-25]. Dostupné z: https://www.analyticsvidhya.com/blog/2022/06/understanding-loss-function-in-deep-learning/

[39] ROHINI, G. Everything you need to know about VGG16. In: *Medium* [online]. London: Medium, 2021 [cit. 2023-05-11]. Dostupné z: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918

[40] JORDAN, JEREMY. Setting the learning rate of your neural network. In: *Jeremy Jordan* [online]. Jeremy Jordan, 2018 [cit. 2023-05-11]. Dostupné z: https://www.jeremyjordan.me/nn-learning-rate/

Zoznam symbolov a skratiek

Symbol	Význam	Jednotka	Jednotka (Slovne)
C	Kapacita	F	Farad
ϵ	Permitivita materiálu	F/m	Farad na meter
A	Plocha dosiek kondenzátora	m^2	Meter štvorcový
d	Vzdialenosť dosiek kondenzátora	m	Meter
Q	Náboj na doskách kondenzátora	C	Coloumb
V	Napätie na kondenzátore	V	Volt
σ_s	Povrchová hustota náboja	C/m^2	Coloumb na meter štvorcový
ϵ_0	Permitivita volného prostredia	F/m	Farad na meter
s	Hrúbka elektretu	m	Meter
s_1	Hrúbka volného priestoru	m	Meter
D	Indukovaný elektrický posuv	C/m^2	Coloumb na meter štvorcový
E	Aplikované elektrické pole	N/C	Newton na coloumb
X	Mechanické namáhanie materiálu	NA	Neaplikovatelné
d_p	Piezoelektrický koaficient	C/N	Coloumb na newton
ϵ_p	Dielektrická konštanta	NA	Neaplikovatelné
H	Priestor hypotéz	NA	Neaplikovatelné
y	Reálna hodnota výstupu	NA	Neaplikovatelné
h	Predpovedaná hodnota výstupu	NA	Neaplikovatelné
m	Počet príkladov	NA	Neaplikovatelné
argmin	Matematická funkcia	NA	Neaplikovatelné
$\hat{L}$	Logistická stratová funkcia	NA	Neaplikovatelné
w	Parametre naučené pri trénovaní	NA	Neaplikovatelné
D	Trénovací dataset	NA	Neaplikovatelné
s_j	Výstup z neurónu skrytej vrstvy	NA	Neaplikovatelné
g	Aktivačná funkcia	NA	Neaplikovatelné
n_j	Počet neurónov v predchádzajúcej vrstve	NA	Neaplikovatelné
w_j	Váha medzi dvomi prepojenými vrstvami	NA	Neaplikovatelné
x_j	Výstup z predošlej vrstvy	NA	Neaplikovatelné
D_j	Celkový počet dátových záznamov	NA	Neaplikovatelné
S	Počet súborou	NA	Neaplikovatelné
f	Počet datových záznamov v súbore	NA	Neaplikovatelné

STFT	Rýchla Fourierova transformácia	NA	Neaplikovatelné
x	Časový index signálu	NA	Neaplikovatelné
w	Frekvenčný index signálu	NA	Neaplikovatelné
$w(\tau)$	Oknová funkcia	NA	Neaplikovatelné
τ	Časová premenná	NA	Neaplikovatelné
j	Imaginárna jednotka	NA	Neaplikovatelné
x(t)	Signál	NA	Neaplikovatelné
w	Frekvencia	Hz	Hertz
CNN	Konvolunčná neurónová sieť	NA	Neaplikovatelné
p(y)	Pravdepodobnosť že bod patrí do triedy	NA	Neaplikovatelné
N	Počet Bodov	NA	Neaplikovatelné
$\hat{y}$	Predpovedané označenie bodov	NA	Neaplikovatelné

Tab. 4.5: Zoznam symbolov a skratiek

Zoznam príloh

A Tvorenie datasetu pre binárnu klasifikáciu zo surových dát 93

B Vytvorenie a uloženie spektografov 95

C Tvorenie datasetov pre klasifikáciu prietokov zo surových dár 97

D Tvorenie datasetov pre klasifikáciu prietokov a kavitácie za pomoci spektografov 101

E CNN pre binárnu klasifikáciu kavitácie zo surových dát 103

F CNN pre klasifikácu prietokov zo surových dát 107

G CNN pre binárnu klasifikáciu kavitácie za pomoci spektografov 111

H CNN pre klasifikáciu prietokov za pomoci spektografov 115

A Tvorenie datasetu pre binárnu klasifikáciu zo surových dát

```
    import pandas as pd
import numpy as np
import os
import random
import tensorflow as tf

# Cesta k súborom
kavituje_dir_path = r"D:\all\p_kis\Kavituje\\"
nekavituje_dir_path = r"D:\all\p_kis\Nekavituje\\"

# zoznam všetkých súborov
kavituje_file_names = os.listdir(kavituje_dir_path)
nekavituje_file_names = os.listdir(nekavituje_dir_path)

#  Načítanie dát z excelov v Kavituje
kavituje_data = []
for i, file_name in enumerate(kavituje_file_names):
    file_path = os.path.join(kavituje_dir_path, file_name)
    df = pd.read_excel(file_path, header=None)
    data = np.concatenate(df.values)
    label = "kavituje"
    kavituje_data.append((data, label))
    print("Reading file {} of {}: {} ({})".format(i+1, len(kavituje_file_names),
    file_name, label))

# Načítanie dát z excelov v Nekavituje
nekavituje_data = []
for i, file_name in enumerate(nekavituje_file_names):
    file_path = os.path.join(nekavituje_dir_path, file_name)
    df = pd.read_excel(file_path, header=None)
    data = np.concatenate(df.values)
    label = "nekavituje"
    nekavituje_data.append((data, label))
    print("Reading file {} of {}: {} ({})".format(i+1,
    len(nekavituje_file_names), file_name, label))
```

```
# Zlúčenie kavituje a neavituje
data = kavituje_data + nekavituje_data

# Zamiešanie
random.shuffle(data)

# Calculate the maximum length of the data
max_length = max([len(d) for d, _ in data])

# Rozdelenie
X = np.array([np.pad(d, (0, max_length - len(d)), mode='constant')
for d, _ in data])
y = np.array([label for _, label in data])

#
X = np.expand_dims(X, axis=-1)

# SRozdelenie na datasety
train_split = 0.7
val_split = 0.2
test_split = 0.1
train_size = int(train_split * len(data))
val_size = int(val_split * len(data))
train_X = X[:train_size]
train_y = y[:train_size]
val_X = X[train_size:train_size+val_size]
val_y = y[train_size:train_size+val_size]
test_X = X[train_size+val_size:]
test_y = y[train_size+val_size:]

# Info
print("Dataset shape: {}".format(X.shape))
print("Training set shape: {}".format(train_X.shape))
print("Validation set shape: {}".format(val_X.shape))
print("Test set shape: {}".format(test_X.shape))
```

B Vytvorenie a uloženie spektografov

```
    import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import stft
from tqdm import tqdm

# Funkcia pre uloženie datasetov
def save_spectrogram(file_path, f, t, Sxx):
    dir_path = os.path.dirname(file_path)
    file_name = os.path.splitext(os.path.basename(file_path))[0]
    np.save(os.path.join(dir_path, file_name + "_f"), f)
    np.save(os.path.join(dir_path, file_name + "_t"), t)
    np.save(os.path.join(dir_path, file_name + "_Sxx"), Sxx)

# Cielová zloška
main_directory = r"D:\all\Prietok\SPEK\AE\FOURň"

# STFT parametre
nperseg = 2028
noverlap = nperseg // 2

# Orezanie frekvencie
trim_freq = 0  # 10 kHz


for root, dirs, files in os.walk(main_directory):
    for file_name in tqdm(files, desc='Processing files', unit='file'):
        if file_name.endswith(".xlsx"):

            file_path = os.path.join(root, file_name)
            df = pd.read_excel(file_path, header=None, engine='openpyxl')


            x = df.values.squeeze()

            # Spočítanie spektografov
```

```
f, t, Sxx = stft(x, fs=200000, nperseg=nperseg, noverlap=noverlap)

# Orezanie
trim_idx = np.argmax(f >= trim_freq)
f = f[trim_idx:]
Sxx = Sxx[trim_idx:, :]

# Uloženie
dir_path = os.path.dirname(file_path)
file_name = os.path.splitext(os.path.basename(file_path))[0]
np.save(os.path.join(dir_path, file_name + "_Sxx"), np.abs(Sxx))

# Vykreslenie
plt.pcolormesh(t, f, np.abs(Sxx))
plt.xlabel('Time [sec]')
plt.ylabel('Frequency [Hz]')
plt.title(file_name)
plt.ylim(trim_freq, np.max(f))
plt.colorbar()
plt.savefig(os.path.join(dir_path, file_name + "no trim.png"))
plt.close()
```

C Tvorenie datasetov pre klasifikáciu prietokov zo surových dár

```
    import numpy as np
import os
import random
import tensorflow as tf

# Súbory obsahujúce rozdelené prietoky
data_dirs = [r"D:\all\Prietok\SPEK\SP\1",
             r"D:\all\Prietok\SPEK\SP\2",
             r"D:\all\Prietok\SPEK\SP\3",
             r"D:\all\Prietok\SPEK\SP\4",
             r"D:\all\Prietok\SPEK\SP\5",
             r"D:\all\Prietok\SPEK\SP\6",
             r"D:\all\Prietok\SPEK\SP\7",
             r"D:\all\Prietok\SPEK\SP\8",
             r"D:\all\Prietok\SPEK\SP\9",
             r"D:\all\Prietok\SPEK\SP\10",
             r"D:\all\Prietok\SPEK\SP\11",
             r"D:\all\Prietok\SPEK\SP\12",
             r"D:\all\Prietok\SPEK\SP\13",
             r"D:\all\Prietok\SPEK\SP\14",
             r"D:\all\Prietok\SPEK\SP\15",
             r"D:\all\Prietok\SPEK\SP\16",
             r"D:\all\Prietok\SPEK\SP\17",
             r"D:\all\Prietok\SPEK\SP\18",
             r"D:\all\Prietok\SPEK\SP\19",
             r"D:\all\Prietok\SPEK\SP\20",
             r"D:\all\Prietok\SPEK\SP\21",
             r"D:\all\Prietok\SPEK\SP\22",
             r"D:\all\Prietok\SPEK\SP\23",
             r"D:\all\Prietok\SPEK\SP\24",
             r"D:\all\Prietok\SPEK\SP\25",
             r"D:\all\Prietok\SPEK\SP\26",
             r"D:\all\Prietok\SPEK\SP\27",
             r"D:\all\Prietok\SPEK\SP\28",
             r"D:\all\Prietok\SPEK\SP\29",
```

```
            r"D:\all\Prietok\SPEK\SP\30",
            r"D:\all\Prietok\SPEK\SP\31",
            r"D:\all\Prietok\SPEK\SP\32",
            r"D:\all\Prietok\SPEK\SP\33",
            r"D:\all\Prietok\SPEK\SP\34",
            r"D:\all\Prietok\SPEK\SP\35"]
labels = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]

# Vytvorenie prázdnych zoznamov
data = []
labels_list = []

# Načítanie dát
for i, data_dir in enumerate(data_dirs):
    file_names = os.listdir(data_dir)
    for j, file_name in enumerate(file_names):
        file_path = os.path.join(data_dir, file_name)
        with np.load(file_path) as f:
            data_row = f["data"]
        label = labels[i]
        data.append(data_row)
        labels_list.append(label)
        print("Reading file {} of {}: {} ({})".format(j+1, len(file_names),
        file_name, label))

# Zlúčenie dát
data = list(zip(data, labels_list))

# Zamiešanie
random.shuffle(data)


max_length = max([d.shape[1] for d, _ in data])

# Rozdelenie na dáta a atribúty
X = np.vstack([np.pad(d, ((0, 0), (0, max_length - d.shape[1])), mode='constant')
for d, _ in data])
y = np.array([label for _, label in data])
```

```
X = np.expand_dims(X, axis=-1)

# Rozdelenie na datasety
train_split = 0.7
val_split = 0.2
test_split = 0.1
train_size = int(train_split * len(data))
val_size = int(val_split * len(data))
train_X = X[:train_size]
train_y = y[:train_size]
val_X = X[train_size:train_size+val_size]
val_y = y[train_size:train_size+val_size]
test_X = X[train_size+val_size:]
test_y = y[train_size+val_size:]

# Info
print("Dataset shape: {}".format(X.shape))
print("Training set shape: {}".format(train_X.shape))
print("Validation set shape: {}".format(val_X.shape))
print("Test set shape: {}".format(test_X.shape))

# Uloženie
save_path = r"D:\all\Prietok\SP\DATASET"

np.savez_compressed(save_path, train_X=train_X, train_y=train_y, val_X=val_X,
val_y=val_y, test_X=test_X, test_y=test_y)
```

D Tvorenie datasetov pre klasifikáciu prietokov a kavitácie za pomoci spektografov

```
import os
import numpy as np
from sklearn.model_selection import train_test_split

# Cesta k súborom
data_path = r"D:\all\Prietok\SPEK\reduced"

# Zoznam súborov obsaujúcich 2D matice
folders = os.listdir(data_path)

# Prázdne zoznamy
data = []
labels = []

# Prejsť všetky súbory
for folder in folders:
    folder_path = os.path.join(data_path, folder)
    files = os.listdir(folder_path)


    for file in files:
        file_path = os.path.join(folder_path, file)
        array = np.load(file_path, allow_pickle=True)

        # Pridelenie atribútu podla složky, v ktorej je umiestnený
        label = int(folder)


        data.append(array)
        labels.append(label)

# Zlúčenie
data = np.stack(data, axis=0)
```

```
labels = np.array(labels)

# Zamiešanie
indices = np.random.permutation(data.shape[0])
data = data[indices]
labels = labels[indices]

# Rozdelenie
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.1,
random_state=42)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2,
train_size=0.7, random_state=42)

# Uloženie
np.savez("DATASET_Q_SPEK_AE_RED.npz", X_train=X_train, y_train=y_train, X_val=X_val,
y_val=y_val, X_test=X_test, y_test=y_test)


print("Shape of X_train:", X_train.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of X_val:", X_val.shape)
print("Shape of y_val:", y_val.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of y_test:", y_test.shape)
print("Number of unique labels:", len(np.unique(labels)))
```

E CNN pre binárnu klasifikáciu kavitácie zo surových dát

```
import numpy as np
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Conv1D, MaxPooling1D, Flatten
from keras.models import Sequential
from keras.optimizers import Adam
from keras.models import load_model

# Cesta k súborom
dataset = np.load(r"/content/drive/MyDrive/DP/DATASET_PK.csv.npz")
print(dataset.files)


# Načítanie datasetov
train_X = dataset['train_X']
train_y = dataset['train_y']
val_X = dataset['val_X']
val_y = dataset['val_y']
test_X = dataset['test_X']
test_y = dataset['test_y']


train_data_shape = train_X.shape
train_labels_shape = train_y.shape



print(f"Training data shape: {train_data_shape}")
print(f"Training labels shape: {train_labels_shape}")

# Definovanie archytektúry
model = Sequential()
model.add(Conv1D(filters=4, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=4, kernel_size=3, activation='relu'))
```

```
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=8, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=8, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=1, activation='sigmoid'))


learning_rate = 0.00001
optimizer = Adam(lr=learning_rate)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])


checkpoint = ModelCheckpoint('best_P_kis_KAV_SEG.h5', monitor='val_accuracy',

verbose=1,
save_best_only=True, mode='max')
early_stopping = EarlyStopping(monitor='val_loss', patience=5, mode='min')

# Trénovanie
history = model.fit(train_X, train_y, validation_data=(val_X, val_y), epochs=35,

batch_size=150, callbacks=[checkpoint, early_stopping])

# Vyhodnotenie
```

```
test_loss, test_acc = model.evaluate(test_X, test_y)
print('Test loss:', test_loss)
print('Test accuracy:', test_acc)

model.summary()
# Grafy
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel
```

F CNN pre klasifikácu prietokov zo surových dát

```
import numpy as np
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Conv1D, MaxPooling1D, Flatten
from keras.models import Sequential
from keras.optimizers import Adam

# Načítanie datasetu
dataset = np.load(r"/content/drive/MyDrive/DP/Pretok/DATASET_SP.npz")
print(dataset.files)

# Načítanie datasetov
train_X = dataset['train_X']
train_y = dataset['train_y']
val_X = dataset['val_X']
val_y = dataset['val_y']
test_X = dataset['test_X']
test_y = dataset['test_y']


train_data_shape = train_X.shape
train_labels_shape = train_y.shape



print(f"Training data shape: {train_data_shape}")
print(f"Training labels shape: {train_labels_shape}")


# Archytektúra
model = Sequential()
model.add(Conv1D(filters=4, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=4, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
```

```
model.add(Conv1D(filters=8, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=8, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=16, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=35, activation='sigmoid'))


learning_rate = 0.001
optimizer = Adam(lr=learning_rate)
model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy',

metrics=['accuracy'])




checkpoint = ModelCheckpoint('best_model.h5_prietok_SEG_SP', monitor='val_accuracy',
verbose=1, save_best_only=True, mode='max')
early_stopping = EarlyStopping(monitor='val_loss', patience=100, mode='min')



# Trénovanie
history = model.fit(train_X, train_y, validation_data=(val_X, val_y), epochs=150,
batch_size=400, callbacks=[checkpoint, early_stopping])
```

```
model.summary()
# Vyhodnotenie
test_loss, test_acc = model.evaluate(test_X, test_y)
print('Test loss:', test_loss)
print('Test accuracy:', test_acc)


# Grafy
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel("Epochs")
```

G CNN pre binárnu klasifikáciu kavitácie za pomoci spektografov

```
import numpy as np
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from keras.models import Sequential
from keras.optimizers import Adam

# Cesta k datasetu
dataset = np.load(r"/content/drive/MyDrive/DP/KAV_BEST/DATASET_SPEK_P_KIS.npz")
print(dataset.files)

# Rozdelenie
train_X = dataset['train_X']
train_y = dataset['train_y']
val_X = dataset['val_X']
val_y = dataset['val_y']
test_X = dataset['test_X']
test_y = dataset['test_y']


train_data_shape = train_X.shape
train_labels_shape = train_y.shape


print(f"Training data shape: {train_data_shape}")
print(f"Training labels shape: {train_labels_shape}")

# Archytektúra
model = Sequential()
model.add(Conv2D(filters=4, kernel_size=(3, 3), activation='relu',
input_shape=(913, 199, 1)))
model.add(Conv2D(filters=4, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(filters=8, kernel_size=(3, 3), activation='relu'))
```

```
model.add(Conv2D(filters=8, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Conv2D(filters=16, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=16, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=16, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Flatten())
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=1, activation='sigmoid'))


learning_rate = 0.00001
optimizer = Adam(lr=learning_rate)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])


checkpoint = ModelCheckpoint('best_model_P_KIS_SPEK.h5', monitor='val_accuracy',
verbose=1, save_best_only=True, mode='max')
early_stopping = EarlyStopping(monitor='val_loss', patience=100, mode='min')

# Trénovanie
history = model.fit(train_X, train_y, validation_data=(val_X, val_y), epochs=50,
batch_size=50, callbacks=[checkpoint, early_stopping])

# Vyhodnotenie
test_loss, test_acc = model.evaluate(test_X, test_y)
print('Test loss:', test_loss)
print('Test accuracy:', test_acc)

model.summary()
```

```
# Grafy
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel("Epochs")
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

H CNN pre klasifikáciu prietokov za pomoci spektografov

```
import numpy as np
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from keras.models import Sequential
from keras.optimizers import Adam

# Cesta k datasetu
dataset = np.load(r"/content/drive/MyDrive/DP/DATASET_Q_SPEK_AE_RED.npz")
print(dataset.files)


train_X = dataset['X_train']
train_y = dataset['y_train']
val_X = dataset['X_val']
val_y = dataset['y_val']
test_X = dataset['X_test']
test_y = dataset['y_test']


train_data_shape = train_X.shape
train_labels_shape = train_y.shape



print(f"Training data shape: {train_data_shape}")
print(f"Training labels shape: {train_labels_shape}")


# Archytektúra
model = Sequential()
model.add(Conv2D(filters=2, kernel_size=3, activation='relu',
input_shape=(1015, 199, 1)))
model.add(Conv2D(filters=2, kernel_size=3, activation='relu'))
```

```
model.add(MaxPooling2D(pool_size=2,padding='same'))
model.add(Conv2D(filters=2, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=2, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2, padding='same'))
model.add(Conv2D(filters=4, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=4, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=4, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2, padding='same'))
model.add(Conv2D(filters=8, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=8, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=8, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2, padding='same'))
model.add(Conv2D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=16, kernel_size=3, activation='relu'))
model.add(Conv2D(filters=16, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=4, padding='same'))
model.add(Flatten())
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=36, activation='sigmoid'))


learning_rate=0.000001
optimizer = Adam(lr=learning_rate)
model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy',
metrics=['accuracy'])




checkpoint = ModelCheckpoint('best_model_Q_SPEK_AE', monitor='val_accuracy',
verbose=1, save_best_only=True, mode='max')
early_stopping = EarlyStopping(monitor='val_loss', patience=100, mode='min')



# Trénovanie
history = model.fit(train_X, train_y, validation_data=(val_X, val_y), epochs=100,
batch_size=200, callbacks=[checkpoint, early_stopping])
```

```
model.summary()
# Vyhodnotenie
test_loss, test_acc = model.evaluate(test_X, test_y)
print('Test loss:', test_loss)
print('Test accuracy:', test_acc)


# Grafy
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel("Epochs")
```