VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY
INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

SOFTWAROVÉ MOŽNOSTI NASAZENÍ ALGORITMŮ METOD UMĚLÉ INTELIGENCE V PRŮMYSLU

SOFTWARE POSSIBILITIES OF USING ALGORITHMS OF ARTIFICIAL INTELLIGENCE METHODS IN INDUSTRY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. Kristián Karas

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Jiří Kovář, Ph.D.

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Bc. Kristián Karas**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Mechatronika
Vedoucí práce:	**Ing. Jiří Kovář, Ph.D.**
Akademický rok:	2020/21

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:

Softwarové možnosti nasazení algoritmů metod umělé inteligence v průmyslu

Stručná charakteristika problematiky úkolu:

V aktulní době zažíváme velký rozmach použití metod umělé inteligence. Specifikem použití těchto metod v průmyslu je podmínka jejich dlouhodobého použití bez možnosti vnějšího zásahu. Tento fakt klade vysoké nároky na adaptaci používaných metod. Cílem práce je rešerše těchnto adaptačních metod a následně test jedné z nich.

Cíle diplomové práce:

Prostudujte a popiště metody a způsoby adaptace vybraných metod umělé inteligence.
Navrhnětě a realizujte test vybrané metody na modelu technické soustavy.

Seznam doporučené literatury:

MURPHY, K.P, Machine Learning : A Probabilistic Perspective(Hardback) - 2012 Edition, 2012, ISBN-13: 978-0262018029.

BISHOP, M. Ch., Pattern Recognition and Machine Learning (Information Science and Statistics), ISBN-13: 978-0387310732.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

Abstrakt

Práce je zaměřena na využití technik umělé inteligence v průmyslu a v systémech pro monitorování strojů. V praktické části se dále práce zaměřuje na stavbu konvoluční neuronové sítě a její testovaní na reálných datech pro diagnostiku stavu stroje.

Klíčová slova

umělá inteligence, strojové učení, neuronová síť, konvoluční neuronová síť, perceptron, diagnostika, prognostika, monitorovací systémy, klasifikace

Abstract

The work is focused on the use of artificial intelligence techniques in the industry and in systems for monitoring machines. In the practical part, the work focuses on the construction of a convolutional neural network and its testing on real data for diagnosing the state of the machine.

Keywords

artificial intelligence, machine learning, neural network, convolutional neural network, perceptron, diagnostics, prognosis, monitoring systems, classification

Bibliografická citace:

KARAS, Kristián. *Softwarové možnosti nasazení algoritmů metod umělé inteligence v průmyslu* [online]. Brno, 2021 [cit. 2021-05-14]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/131926. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce Jiří Kovář.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma *Softwarové možnosti nasazení algoritmů metod umělé inteligence v průmyslu* jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **20. května 2021** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Jiřímu Kováři, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

Chtěl bych touto cestou také poděkovat i mému otci, který byl pro mě vždy vzorem a který už od mala podporoval můj zápal pro techniku a technické zájmy. Byl jsi mou oporou po celou dobu studia, a i díky tobě jsem tam kde jsem. Děkuji.

V Brně dne: **20. května 2021** ……

podpis autora

Obsah

1 ÚVOD

Teoretická část této práce se zabývá představením technik umělé inteligence (artificial intelligence - AI) a popisem technik učení. Také jsou zde popsány některé aplikace AI technik v samotném průmyslu a bližší popis monitorovacích systémů, které využívají strojové učení pro diagnostiku a prognostiku strojů.

V praktické část této práce je potom popsán princip a stavba konvoluční neuronové sítě, která byla v rámci práce naprogramována v programovacím jazyce python. Pro testovaní této vlastní konvoluční neuronové sítě byl použit dataset spektrogramů, získaný Fourierovou transformací signálu ze senzorů vibrací.

1.1 Cíl práce

Hlavním cílem této práce je přiblížení technik umělé inteligence a zároveň důkladné vysvětlení fungovaní jednoho představitele těchto technik, a to konvoluční neuronové sítě, která byla na závěr této práce otestována reálnými daty ze stroje.

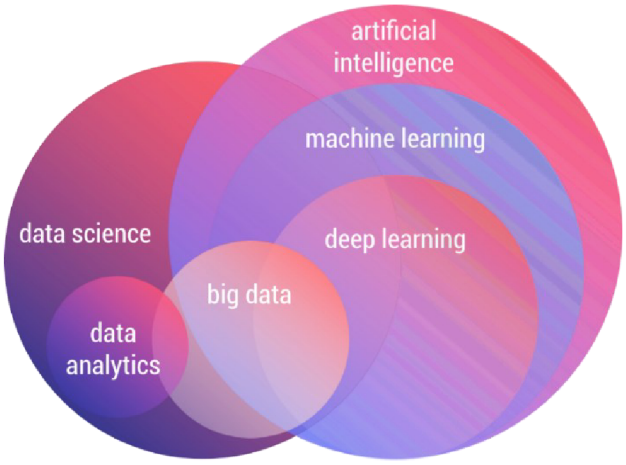
2 PRVKY UMĚLÉ INTELIGENCE V PRŮMYSLU

Ačkoliv prvky artificial intelligence (AI) jsou známy na akademické úrovni už od 50. let minulého století, své popularitu se dočkali až v posledních letech, a to hlavně díky mnohem výkonnější výpočetní technice [1]. Techniky AI se postupným vývojem rozdělily na mnoho podoborů, které jsou však v mnoha literaturách označené zaměnitelně a nepřesně. Podobory, kterými jsou techniky AI, jsou především deep learning, machine learning, big data a data science. Každý z těchto pojmů představuje trošku jiný význam, a proto je potřeba těmto pojmům dobře porozumět.

2.1 Artificial intelligence

Artificial intelligence (AI) je pojem, kterým se zatupují především systémy, které dokážou fungováním napodobit lidskou inteligenci a schopnosti učení. Tyto systémy na základě zkušeností řeší složíte problémy, a postupem času se stávají víc a víc schopnější.

První systémy s technikami AI vznikly v padesátých a šedesátých letech minulého století, kterými byly především neural networks (NN). Tyto systémy se poté v osmdesátých letech rozvinuly do machine learning technik až do současnosti, kdy se k tomu poté přidaly i techniky deep learning [1].



Obr. 2-1 Rozdělení umělé intelligence
(Zdroj: [1])

Machine learning

Machine learning (ML) neboli strojové učení jsou techniky učení neuronových sítí, které dokážou na základě vstupních dat upravit, naučit matematický model tak, aby dokázal řešit i nelineární problémy. Tento model je poté schopný sám se rozhodovat a predikovat určité hodnoty. Samotný ML se dělí dále na 4 techniky učení a každá z těchto technik pracuje na jiném přístupu k datům a samotnému učení neuronových sítí.

Supervised learning

Při procesu učení se využívají data, která musí být označena (label) správnou hodnotou výsledku. Díky této hodnotě dokáže matematický model zpětně určit vlastní chybu predikce a na základě této chyby si upravit parametry modelu tak, aby se tato chyba každou iterací snižovala.

- Unsupervised learning

Při procesu učení nejsou data označená žádnou správnou hodnotou výsledku, a proto nelze určit ani chybu predikce, a ani přírůstky parametrů modelu. Unsupervised learning analyzuje data, hledá si v nich vzory a vlastnosti, díky kterým může tato data sám smysluplně rozdělit do tříd či jinak upravit.

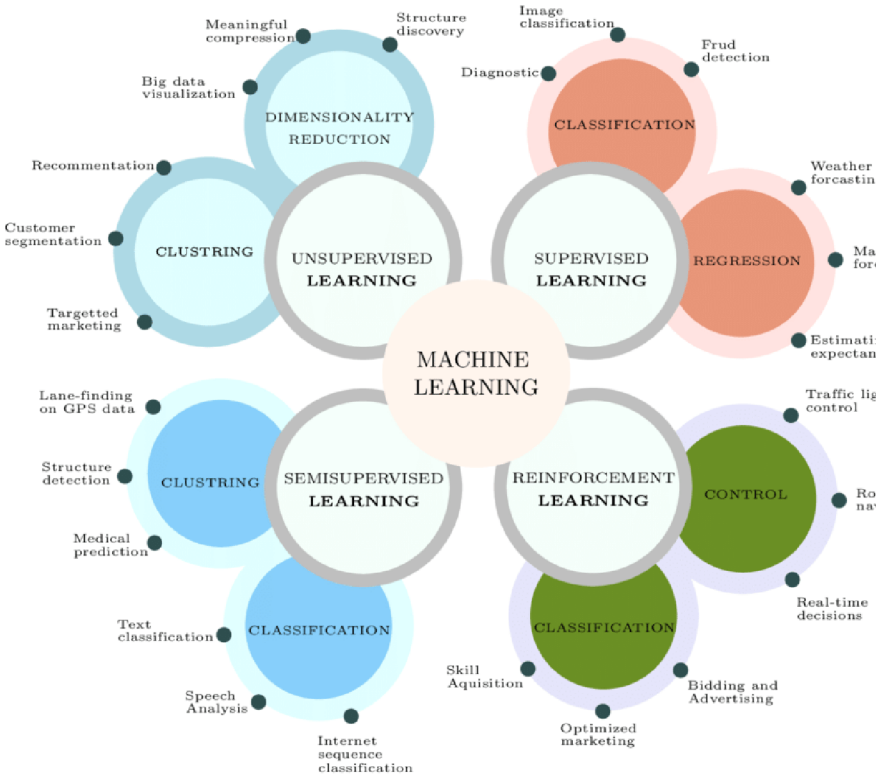
- Semisupervised learning

Tyto systémy jsou kombinací techniky unsupervised a supervised learning. Tuto techniku využívá například deep belief network. Na začátku učení pracuje s neoznačenými daty, a až na konci procesu učení pro doladění parametru, využívá data se správnými hodnotami výsledku jak u supervised learning.

- Reinforcement learning

Reinforcment learning (RL) na rozdíl od předešlých technik učení pracuje a učí se na základě vlastních zkušeností. RL při procesu učení testuje různé způsoby chovaní, za které je ohodnocen od hodnotící funkce číselnou hodnotou, která ho při správném chování ohodnotí kladně a při nesprávném chovaní záporně. RL si tak hledá sám správnou cestu, jak získat nejlepší skóre z hodnotící funkce a vyřešit

daný problém charakterizovaný právě touto funkcí.



Obr. 2-2 Machine learning
(Zdroj: [2])

Deep learning

Deep learning (DL) je podobor samotného ML, do kterého spadají především neuronové sítě, které mají oproti běžným sítím více vrstev perceptronů [1]. Velká struktura neuronové sítě přináší větší výpočetní náročnost a delší čas učení. Ale velkou výhodou je, že dokážou řešit složitější a komplexnější problémy, než běžné neuronové sítě.

Data science

Data science je obor, který se nachází na pomezí oborů AI, znalostních systémů a statistiky. Oproti ML však Data science nemá za cíl vytvářet tréninkové modely, ale jenom vytvářet predikce na základě extrahování informací [1].

Big data

Big data (BD) je obor zaměřený na analýzu velkého množství dat, která není možné analyzovat neuronovými sítěmi z důvodu velkých rozměrů [1]. Neuronové sítě dokážou zpracovávat pouze 1D data, popřípadě 2D data, pokud se jedná o convolution neural network. Systémy analyzující big data, díky velkým rozměrům dat, přinášejí lepší výsledky než jiné systémy zpracovávající méně rozměrná data. Samotný obor big data leží na pomezí AI a Data science.

2.2 Aplikace machine learning v průmyslu

Techniky AI pomalu mění skoro každý průmysl od E-commerce sektoru až po medicínu, kde se využívají AI techniky například na diagnostiku nemocí. Formy AI usnadňují především práci s daty a mnohdy i nahrazují práci samotného člověka.

V další části se tato práce, proto bude zabývá některými aplikacemi AI technik v průmyslu a jejich výhody, které přinášejí.

Kontrola kvality

Kontrola kvality produktu je mnohdy únavná a opakující se práce. Každý kus produktu musí být pozorně prohlédnut a musí dojít ke zjištění, zda nemá žádné nedostatky. Každý kus s nedostatky by se měl vyřadit nebo předělat před samotným finálním zabalením, a proto na tuto práci je většinou potřeba více zaměstnanců, aby se odhalily všechny chybné produkty, které nesmí být odeslány zákazníkům za žádnou cenu.

V mnohých firmách se proto zavádějí chytré systémy s technikami ML, které dokážou samy kontrolovat tvar, váhu nebo i vizuální podobu produktu pomocí senzorů a kamer. Tyto systémy tak kontrolují kvalitu a v případě nedostatečné kvality produkt i samy vyřadí z výrobní linky [3].

Diagnostika a prognostika strojů

Diagnostika a prognostika strojů nám pomáhá zjistit aktuální stav stroje a predikovat vývoj životnosti stroje. Díky těmto informacím se může firma včas

připravit na opravu, výměnu degradujících součástek. Stroje jsou tak pomocí senzorů monitorovány a snímaný signál je analyzován pomocí systému založeném na ML. Tyto analýzy pak dokážou s předstihem určit stav stroje a budoucí vývoj [3].

Optimalizace procesů

- Optimalizace skladových zásob

Hlídaní skladových zásob bývá většinou v režii pracovníka, který kontroluje počty naskladněného zboží. Při nedostatku skladových zásob objednává nové. Tento systém funguje převážné pro malé a střední podniky. V případě podniků, které mají mnohem větší skladové zásoby, je vhodnější rozhodovaní přenechat na AI systémech. Jako příkladem práce chytrého systému hlídající skladové zásoby, může být objednávkový systém supermarketů, který nesleduje pouze počty jednotlivého zboží, ale také předpověď počasí, dny v týdnu, roční období a také reklamní aktivity daného produktu [3]. Všechny tyto faktory totiž částečně ovlivňují chovaní zákazníků, a proto je výhodnější používat chytré systémy, které jsou schopné si všechny tyto údaje pohlídat a predikovat vývoj prodeje. Tyto systémy v případě potřeby dokážou vyhodnocovat a upravovat ceny produktů tak, aby dosáhly nejvyššího zisku.

- Optimalizace spotřeby energie

Při optimalizaci spotřeby energie hraje roli mnoho faktorů, které při běžném fungovaní musí být zváženy, aby se dalo co nejlépe rozhodnout, kdy je nejvýhodnější začít určitý pracovní proces. Faktory, které mohou ovlivňovat rozhodování AI systému, jsou například sazba energie, pracovní vytížení linky, skladové zásoby a další [3].

- Optimalizace pracovních pozic

Podobně, jak předchozí dva případy, tak i optimalizace pracovních pozic neboli rozmístění jednotlivých pracovníků ve výrobě může být mnohdy ovlivněna mnoha faktory. Z tohoto důvodu je tento proces lepší spravovat pomocí chytrého systému,

který dokáže zvažovat faktory předpovědi počasí, počet aktuálních pracovníků, skladové zásoby, urgentnost jednotlivých zakázek a další [3].

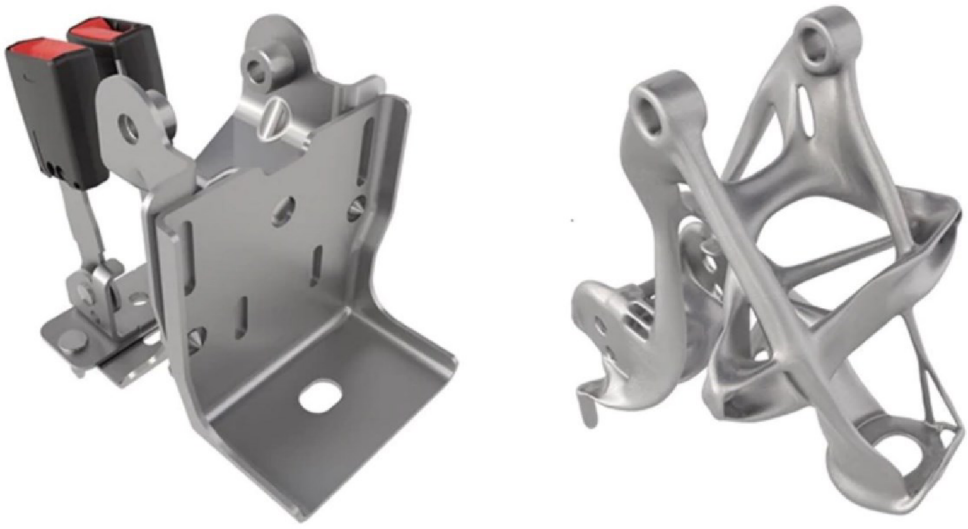
Simulace reálného modelu

Simulace reálných modelů usnadňují projektování a testování výrobních procesu. Díky simulaci lze odhalit mnoho nedostatků ještě před samotnou realizací projektu, a tak ušetřit velké finanční a časové prostředky. Mnoho simulačních systémů funguje i bez ML technik, avšak v poslední době se ukazuje, že implementace ML je pro simulační software krok správným směrem. Pro zjednodušení reálného světa, se v simulačních systémech zavádějí lineární aproximace reálného chování systému. Práce matematického modelu s lineárními závislostmi zrychluje výpočetní procesy, avšak v mnoha případech není lineární aproximace vhodná, a proto se využívá ML pro zastoupení nelineárních procesů v matematickém modelu [4]. ML je také použit pro podporu dynamické vizualizace, do které může uživatel interaktivně zasahovat v průběhu simulace [5].

Generative design

Generative design je systém využívající ML pro tvorbu konstrukčních návrhů splňujících požadované vlastnosti. Je to iterativní proces složený ze dvou neural networks označených jako generátor a diskriminátor. Generátor se snaží v procesu návrhu vytvářet modely výrobku, které poté postupují k diskriminátoru, který hodnotí vytvořený model na základě požadovaných konstrukčních parametrů. Pokud model dostatečně nesplňuje požadované parametry, je označený jako nesprávný. Generátor na základě velikosti nepřesností modelu se zpětně učí a snaží se každý cyklus vytvořit lepší model, který by už prošel kontrolou diskriminátora [6].

Příkladem generative designu je nový úchyt bezpečnostních pásů navržený General motors, který je tvořen pouze z 1 části oceli z původních 8 dílů. Tento nový design je také o 40% lehčí a o 20% pevnější, než jeho předchůdce navržený konstruktéry [7].



Obr. 2-3 Generativ design – držák bezpečnostních pásů
(Zdroj: **[7]**)

3 MONITOROVACÍ SYSTÉMY

V této další části se práce zaměřuje na aplikaci AI v monitorovacích systémech a na celkovou funkci těchto systémů. V návaznosti na tuto část pokračuje i praktická část práce, která demonstruje klasifikační algoritmus diagnostického systému strojů.

Systémy pro sledování stavů strojů se v souvislosti se trendem industry 4.0, staly jednou ze základních věcí všech chytrých firem. Tyto firmy se díky těmto a dalším systémům stávají mnohem efektivnější, spolehlivější a zároveň bezpečnější [8].

Tyto chytré systémy dokážou kromě diagnostiky aktuálního stavu stroje, také predikovat budoucí vývoj stavu stroje, a díky této prediktivní údržbě odhalit poruchu stroje s dostatečným předstihem.

Výzkumy potvrzují, že prediktivní údržba strojů snižuje počet poruch o 70 % a také snižuje náklady na údržbu až o 12 % [9].

Staré monitorovací systémy jsou tzv. pochůzkové systémy neboli offline systémy. Offline systémy obsluhuje vyškolený pracovník, který má za úkol v různých časových intervalech kontrolovat a měřit signály ze strojů [10]. Změřená data poté analyzuje a vyhodnocuje stav daného stroje v příslušném diagnostickém softwaru.

Naproti tomu nové online systémy sbírají signály strojů pomocí senzorů, které jsou umístěné na daném stroji a tento signál neustále odesílají do centrálního uložiště, kde se signál analyzuje [10]. Tyto systémy jsou tak vhodnější pro analýzu signálu pomocí ML, protože přinášejí mnohem více informací o chování daného stroje, než offline systémy.

Systémy prediktivní údržby s technikami strojového učení přináší především:

- Snížení počtu prostojů.
- Vyšší produktivitu a provozní efektivitu.
- Zlepšení bezpečnosti v okolí strojů.
- Rychlejší pochopení hlavních příčin poruch.

Při návrhu systému je potřeba vyřešit také uložiště, které musí být schopné ukládat a zpracovávat velké množství dat. Pro tyto systémy jsou možnosti jak lokálního uložiště, tak i uložiště v cloudu.

Při ukládaní dat do cloudu bývá většinou k dispozici komplexnější systém než na lokálním uložišti. Avšak v některých případech se upřednostňuje rychlost predikce před samotnou komplexností, a proto se v takových případech preferuje tzv. edge computing, který ukládá a analyzuje data v lokálním počítači [11].

Další problém spočívá také v implementaci nového monitorovacího systému s prvky strojového učení. Přechod na nový monitorovací systém, je možné realizovat, až po dostatečném sběru potřebných dat ze strojů pro proces učení. Tato data musí být charakteristická pro řešený problém a také musí být označena správnou hodnotou v případě supervised learningu. Tento fakt je jeden z největších problémů, se kterým se musí podnik implementující nové techniky vypořádat, pokud nemají požadovaná data nebo celou technologii implementovanou ve stroji už od samotného výrobce strojů.

Před implementací monitorovacího systému je zapotřebí zvážit zmíněné problémy a rozhodnou se, zda pro firmu je výhodné do takového systému investovat, nebo zůstat u starého systému s tradičními vyhodnocovacími metodami.

3.1 Procesy monitorovacích systémů

Procesy monitorovacích systému můžeme rozdělit na 3 procesy, které ve většině případu pracují souběžně a zvyšují tak ještě víc spolehlivost.

Diagnostika aktuálního stavu stroje

Při diagnostice aktuálního stavu stroje se zjišťuje, zda stroj bude mít poruchu v krátkém časovém rozmezí. Ve strojovém učení tenhle případ vede ke klasifikačnímu problému [12].

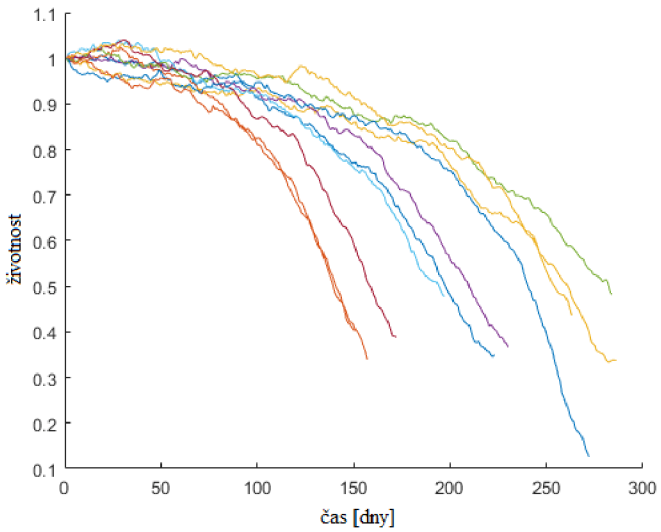
Pro proces učení je potřeba mít označená tréninková data jako normální stav i poruchový stav.

Pro ověření správnosti tréninkových dat je možné vizualizovat tato data a porovnat rozdíly, zda jsou lehce rozpoznatelné nebo ne. V případě, kdy jsou rozdíly mezi jednotlivými daty značné, jedná se o vhodný dataset pro klasifikační problém. V případech, kdy nelze jednoznačně rozpoznat rozdíly mezi jednotlivými třídami, se může jednat o nevhodný dataset nebo dataset pouze se slabými rozdíly.

Pokud se vyhodnotí poruchový stav stroje jako pozitivní, je možné vytvořit i systém, který zjistí, o jakou poruchu se vlastně jedná. V tomto případě je potřeba pro proces učení získat i data ze strojů s konkrétními poruchami, které má systém rozpoznávat. Pokud však firma nemá k dispozici stroj s danou poruchou je potřeba tuto poruchu experimentálně vytvořit nebo získat tato data od výrobce stroje.

Predikce stavu stroje

Zjištění zbývající životnosti stroje vede ve strojovém učení především na regresivní problém. Pro vyřešení tohoto problému je potřeba mít správný dataset s označenými hodnotami jednotlivých fází degradace stroje a strojních součástí od 100%, 80%,70% až po 1% životnosti stroje [11].



Obr. 3-1 Degradující proces
(Zdroj: [13])

Monitoring abnormálních stavů

V tomto případě se sleduje také i abnormální stav stroje, který může indikovat aktuálně vzniklou poruchu. Tato porucha může dále vést k řetězci dalších poruch a až k úplnému zastavení stroje či výrobní linky. Proto se tyto stavy také sledují a v případě výskytu identifikují včas poruchu, upozorní obsluhu, nebo zastaví běh stroje.

3.1.1 Postup analýzy dat

Při diagnostice stroje jsou potřeba tyto tři základní kroky:

- Získaní signálu.
- Extrakce dat.
- Diagnostika a prognostika stroje.

Získaní signálu

Při monitorování strojů můžeme monitorovat mnoho fyzikálních veličin jako jsou vibrace, teplota, hluk, tlak, napětí, odpor, proud, mechanické namáhání a další.

Výběr monitorovacích veličin u stroje je závislý především na charakteru daného stroje, přístupnosti jednotlivých částí stroje a dostupnosti měřících senzorů.

Jako příklad monitorovacích veličin mohou být vibrace strojů. Samotná analýza vibrací strojů se nazývá vibrodiagnostika a je v dnešní době jeden z nejvýznamnějších a nejvíce se rozvíjejícím oborem technické diagnostiky [14]. Vibrodiagnostika nachází uplatnění především mezi rotačními stroji. Pokud rotační stroj pracuje správně, vibrace stroje jsou malé a konstantní, pokud nastane změna v dynamice stroje, projeví se tato změna okamžitě na jeho vibracích. Stroje s komplexní mechanickou strukturou produkují oscilace, které se přenáší přes části stroje. Díky tomuto procesu vytváří každý stroj unikátní frekvenční spektrum, které tento stroj charakterizuje. Pokud se tedy některá důležitá část stroje poškodí, projeví se tato porucha ihned na jeho frekvenčním spektru [15].

Samotné vibrace lze u strojů snímat 3 druhy senzorů, a to senzorem výchylky, senzorem rychlosti a senzorem zrychlení.

Extrakce dat

Extrakce dat je proces, kdy se ze získaného signálu vybírají nejdůležitější vlastnosti, které tento signál reprezentují. Extrakce dat také plní funkci redukce dat, extrahované informace mají totiž mnohem nižší rozměr než původní signál.

Získané signály ze senzorů lze zpracovávat v časové, frekvenční nebo časově-frekvenční oblasti [16]. Před samotným zpracováním dat v některé z uvedených oblastí, je potřeba však jako první odstranit rušení, které vzniká v měřeném signále z důvodu vnějšího okolí měřícího senzoru nebo poruchou samotného senzoru.

Některé rušivé složky v signálu představují vysoko frekvenční šum, který lze odstranit lehce pomocí dolní propustí. Některé rušivé složky mohou mít nízké frekvence, podobné i frekvencím měřeného signálu. V tomto případě je vhodné použít více měřících senzorů a pokusit se identifikovat zdroj rušení.

- **Časová oblast**

Ze signálů v časové oblasti lze získat mnoho statistických hodnot jako například střední hodnota signálu, koeficient špičatosti, koeficient šikmosti, směrodatná odchylka a mnoho dalších hodnot, které mohou být dále použity pro diagnostiku stroje.

- **Frekvenční oblast**

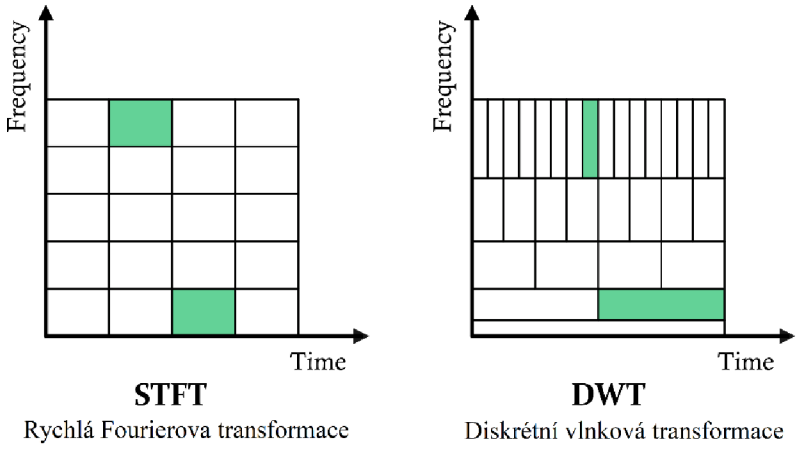
Ve frekvenční oblasti se zkoumá periodicita signálu. Pomocí frekvenční analýzy lze nalézt charakteristické frekvence poruch daného stroje, avšak diagnostika může být komplikovanější, je-li periodicita signálu potlačena. Ve frekvenční oblasti lze také hledat rušivé složky a rezonanční děj. Hlavní nevýhoda frekvenční oblasti je však časová nezávislost [15].

- **Časově – frekvenční oblast**

Signál lze transformovat také do časově-frekvenční oblasti pomocí rychlé

Fourierovy transformace, Wigner-Ville distribuce a Vlnkové transformace. Tyto metody transformace vytváří 3D datové pole, které nám udává složení frekvenčního spektra signálů v závislosti na čase. Toto datové pole lze pomocí barevné škály reprezentovat i ve 2D poli [15].

Vlnková transformace se od krátkodobé Fourierovy transformace liší především tím, že zvyšující se měřenou frekvencí se časové rozdělení úseků více a více diskretizuje. Význam této diskretizace může být vysvětlen na lidském sluchu, který při poslechu signálu s nízkou frekvencí dokáže docela snadno rozpoznat malou změnu frekvence v signálu. Avšak při poslechu signálu s mnohem vyšší frekvencí, nedokáže lidský sluch přesně rozpoznat stejnou změnu frekvence. Díky tomu jsou systémy s vlnkovou transformací citlivé i na změnu ve vyšších frekvencích než při diskrétní Fourierovy transformaci.



Obr. 3-2 Popis diskretizace v časově-frekvenční oblasti
(Zdroj: [17])

3.2 Machine learning modely

Před samotnou implementací ML modelu do systému je potřeba vždy vyzkoušet, který z modelů je pro daný problém vlastně nejvhodnější. Pro většinu řešených problémů je totiž k dispozici vždy více než jen jeden model.

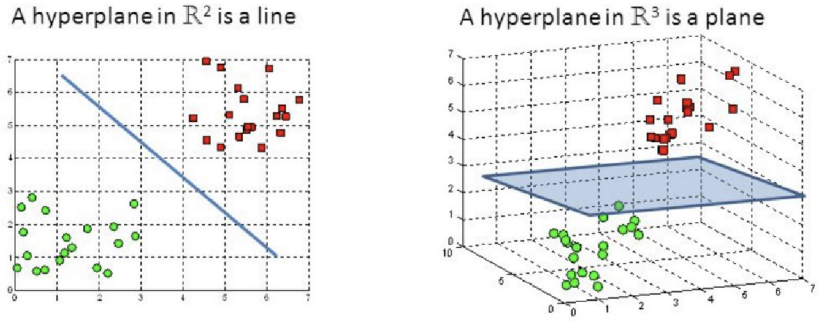
Tabulka 1 Machine learning modely

Klasifikace	Multiclass klasifikace	Regrese	Detekce anomálie
RNN	RNN	RNN	Autoencoder
CNN	CNN	CNN	Anomaly detection
DNN	DNN	DNN	
Random Forest	Random Forest		
SVM	SVM		
Decision trees	Decision trees		

RNN - Recurrent neural network, CNN - Convolution neural network, DNN - Deep neural network, SVM - Support vector machine[1]

Support Vector Machine

Metoda podpůrných vektorů, je metoda strojového učení s učitelem pro binární klasifikaci, klasifikaci do dvou tříd. Tato metoda se snaží pomocí hyperroviny rozdělit N-dimenzionálního prostoru na dva zcela rozdělené prostory. Rozdělit v N-dimenzionálním prostoru body jednou hyperrovinou je ale možné nekonečně mnoha způsoby, avšak tento algoritmus hledá takovou hyperrovinu, která má největší vzdálenost od obou tříd všech bodů [18].



Obr. 3-3 Support vector machine
(Zdroj: [19])

[1] Recurrent neural network = Rekurentní neuronová síť, Convolution neural network = Konvoluční neuronová síť (KNS), Neural network = neuronová síť (NS)

Deep neural networks

Rozdíl mezi neural networks a deep neural networks je ve velikosti vrstev a jejich počtu. Základní stavební jednotkou obou těchto sítích je však perceptron. Perceptron je uzel v neuronové síti, ve které probíhá výpočetní proces závislý na parametrech daného perceptronu. Tyto vrstvy několika perceptronů poté vytváří sítě, které během procesu učení vstřebávají množství data, v závislosti na samotné velikosti sítě. Vstupní data do NS jsou data v 1D struktuře.

Convolutional neural networks

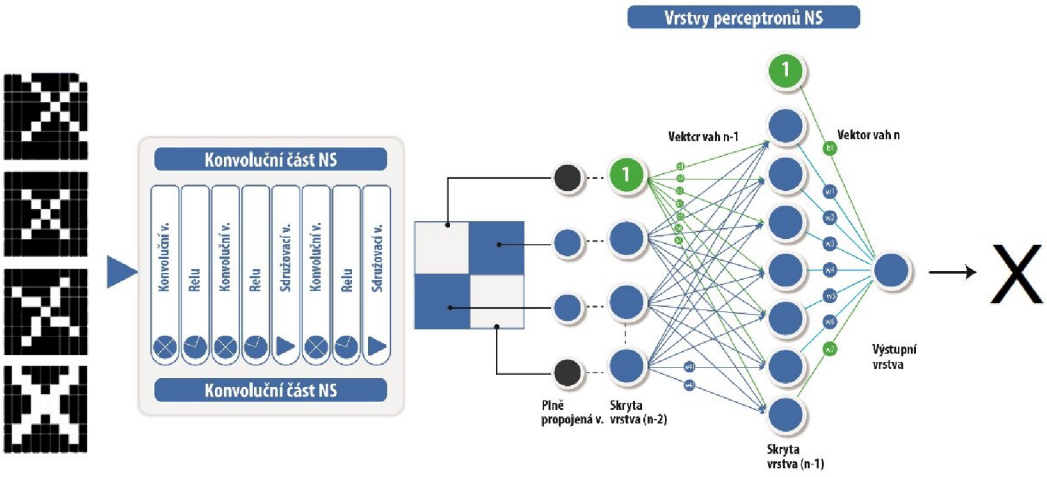
Konvoluční neuronová síť je podobná neuronové sítí s tím rozdílem, že na začátku před vrstvami s perceptrony má tato síť další vyhodnocovací strukturu. Do této struktury vstupují před samotnou NS vstupní data. V této struktuře se podobně, jako v NS nachází několik vrstev, avšak tyto vrstvy jsou specializované na manipulaci s 2D daty. Tyto představují nejčastěji obrázky (fotografie) nebo jiná data, která mají 2D strukturu. Při manipulaci v těchto vrstvách dochází k vyhledávaní vzorců ve 2D struktuře a k redukci rozměru 2D dat. V poslední vrstvě konvoluční části dochází dokonce k transformaci 2D dat na 1D data, která pak vstupují do vrstev perceptronů, které jsou stejné jak v běžné NS.

Recurrent neural network

Rekurentní neuronové sítě jsou takové neuronové sítě, které dokážou pracovat s časově posloupnými daty. Tyto neuronové sítě jsou použity především pro automatický překlad nebo pro rozpoznávání řeči. Svou schopnost pracovat s časově posloupnými informacemi mají především díky tzv. vnitřní paměti sítě. Aktuální vstup do rekurentní sítě je totiž vždy ovlivněn předešlým vstupem a dává tak do souvislosti všechny informace v s danou posloupností [20].

4 STAVBA KONVOLUČNÍ NEURONOVÉ SÍTĚ (KNS)

Pro praktickou část této práce byla vybrána stavba konvoluční neuronové sítě (KNS), která díky schopnosti analyzovat 2D data dokáže řešit problémy, s kterými si běžná NS lehce neporadí. KNS se používají převážně pro analýzu obrazu, avšak v této práci je použita na klasifikaci spektrogramů vytvořených z dat vibrací. Díky analýze spektrogramů, lze tak lépe klasifikovat i problémy komplexních systémů, které při degradaci či menší poruše nevykazují velké změny chování.



Obr. 4-1 Konvoluční neuronová síť

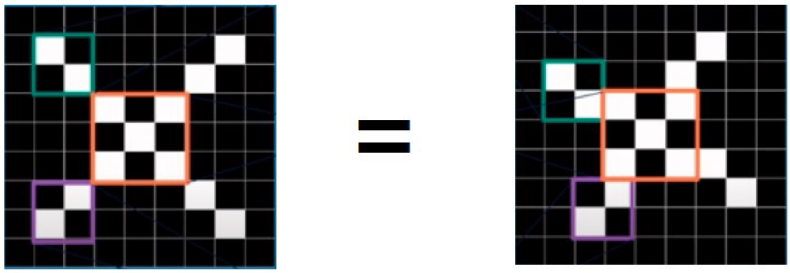
Jak už bylo zmíněno v předešlé kapitole, tak vstup do konvoluční neuronové sítě jsou 2D data, které nejčastěji reprezentují obrázky. Samotná práce KNS je ukázána na černobílých obrázcích o rozměrech 9x9 pixelů, které jsou přivedeny na vstup naučené KNS. Tomuto procesu se říká dopředná propagace. Dopředná propagace má za úkol klasifikovat daná vstupní data do tříd a v tomto případě rozpoznat písmeno z obrazu.

4.1 Princip KNS

KNS pracuje na principu postupného vyhledávaní segmentů (vzorů) v obraze. V první vrstvě se KNS snaží vyhledat základní segmenty, segmenty čar, křivek, bodů a dalších menších útvarů. Tyto segmenty se dále v dalších vrstvách skládají dohromady do větších tvarů, které se KNS učí také identifikovat. Například KNS naučená na rozpoznávání obličeje hledá v první vrstvě segmenty čár a bodů, které se poté v dalších vrstvách skládají dohromady do tvaru oka, nosu, ucha. Tyto

charakteristické rysy se poté dále skládají do podoby obličeje, které se díky tomuto postupu lehce identifikují.

Na Obr. 4-2 můžete vidět barevně zvýrazněné segmenty hledané v prvních vrstvách KNS. Díky rozdělení písmene X na jednotlivé segmenty je snazší pro KNS rozpoznat, zda je nakreslené skutečně písmeno X nebo něco jiného.



Obr. 4-2 Hledané segmenty KNS

Samotný proces vyhledaní segmentů probíhá hlavně v konvoluční vrstvě s pomocí tzv. konvolučních filtrů. Dále jsou pak v KNS další 4 typy vrstev - vrstvy sdružovací (pooling layers), aktivační vrstvy, vrstvy perceptronů a také jedna plně-propojená vrstva (fully connected layer). Tato vrstva transformuje 2D vstupní data na 1D strukturu. Tato transformovaná data dále vstupují do vrstev perceptronů, které jsou známe už i z normálních NS.

4.2 Konvoluce

Operace konvoluce se označuje znakem $*$. Konvoluce získává výslednou hodnotu v bodě (x, y) jako vážený průměr operace konvoluce nad vstupní funkcí f a váhovou funkcí h, která reprezentuje konvoluční filtr:

$$g(x,y) = h(x,y) * f(x,y) \tag{1}$$

$$g(x,y) = \frac{1}{N}\sum_{a=-n}^{n}\sum_{b=-m}^{m} h(a,b)f(x-a,y-b) \tag{2}$$

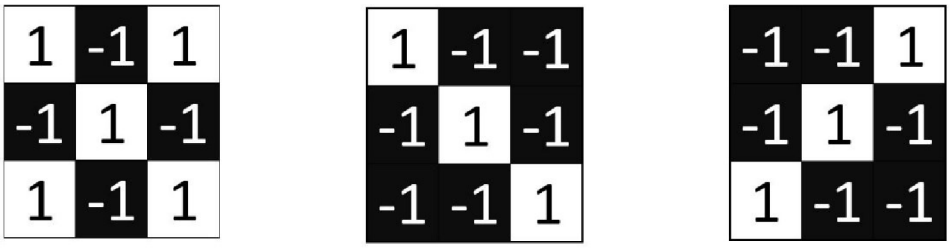
kde pro konvoluční filtr stejných rozměrů se $n = m$, hodnota N označuje počet diskrétních hodnot filtru, funkce $f(x,y)$ je vstupní obraz, $h(a,b)$ je konvoluční

filtr a funkce $g(x, y)$ je pak výstupní obraz z konvoluce.

4.3 Konvoluční vrstva

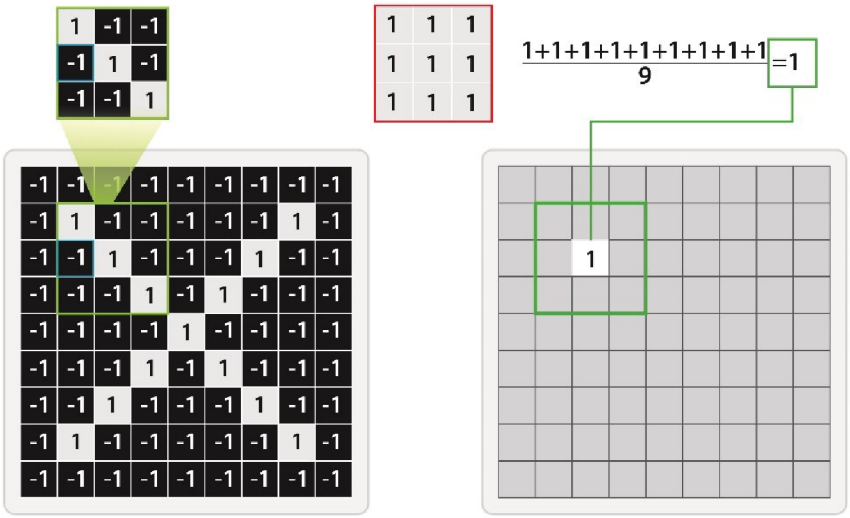
Základem konvoluční vrstvy je několik filtrů, jejichž počet se volí na základě charakteru řešeného problému. Každá hodnota filtru se na začátku volí náhodně v rozmezí 0 až 1. V procesu učení se tyto hodnoty jednotlivých filtrů upravují podobně, jak váhy perceptronů. Funkce těchto filtrů spočívá v identifikovaní charakteristických segmentů ve vstupních datech, které se v dalších vrstvách skládají do větších celků a lze je poté identifikovat jako určitý objekt [21]. Čím více je řešený problém komplexnější, tím více by mělo být KV a filtrů v těchto vrstvách.

Příkladem konvolučních filtrů ve KV mohou být filtry na Obr. 4-3, kde první filtr hledá segment kříže a další dva filtry hledají segmenty horizontálních čar.



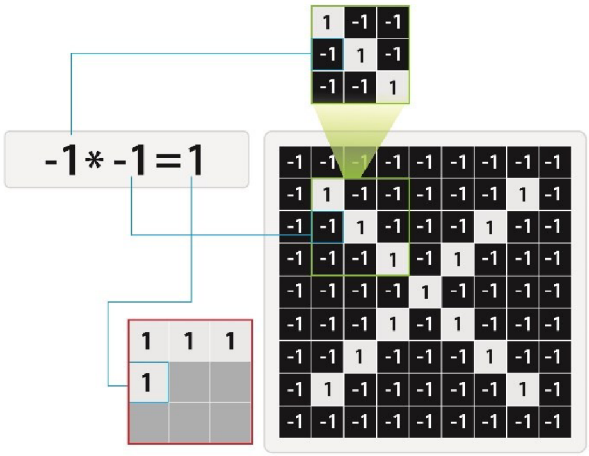
Obr. 4-3 Příklad konvolučních filtrů

Proces procházení obrazu pomocí filtru lze vidět na Obr. 4-4 a Obr. 4-5, kde filtr postupně prochází přes všechny pixely obrazu. Na obr. č. 4 lze vidět jeden okamžik, kdy filtr prochází určitou část pixelů. V tomto kroku se násobí jednotlivé hodnoty filtru s hodnotami pixelů v dané části obrazu. Na příkladě lze vidět, že filtr horizontální čáry se shoduje s danou části obrazu, a proto všechny násobky pixelu mají hodnotu 1. Kdyby tomu tak nebylo, hodnota by byla menší než jedna.



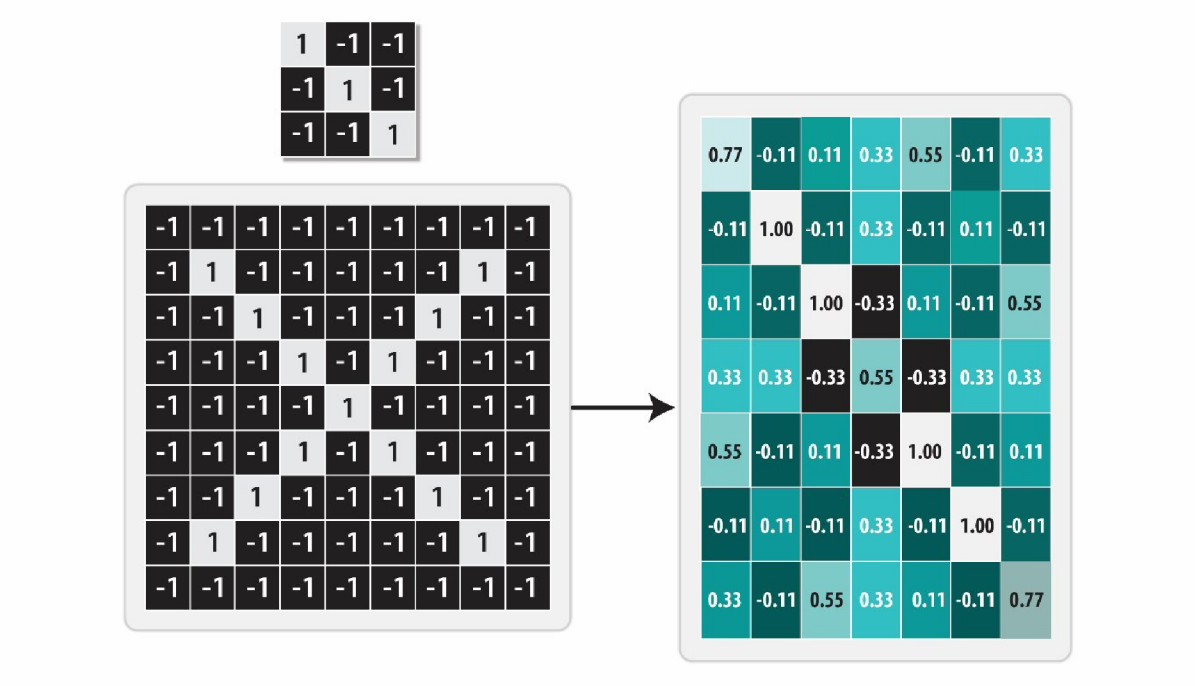
Obr. 4-4 Konvoluční vrstva – proces filtrace obrazu 1.část

Výsledek výstupního pixelu z KV se získá podílem součtu všech násobku ku počtu pixelů filtru.



Obr. 4-5 Konvoluční vrstva – proces filtrace obrazu 2.část

Výsledný vyfiltrovaný obraz je zobrazen na Obr. 4-6. Lze zde vidět, že na horizontální čáře jsou největší hodnoty pixelu, protože byl použit právě filtr na hledaní(identifikovaní) segmentů horizontálních čar. Všechny ostatní mají mnohem nižší hodnoty a některé pixely jsou dokonce záporné.



Obr. 4-6 Výstup z konvoluční vrstvy

Z daného postupu v konvoluční vrstvě lze zpozorovat, že podle počtu filtrů v konvoluční vrstvě vzniká stejný počet vyfiltrovaných obrazů na výstupu z vrstvy. Když tedy je v konvoluční vrstvě 20 filtrů, tak z jednoho vstupního obrazu vznikne 20 výstupních obrazů. Díky této vlastnosti se zvyšuje počet výpočetních procesů, které zpomalují běh samotné KNS. Tato vlastnost je částečně redukována sdružovacími vrstvami, které zmenšují rozměry obrazů.

Při práci s KNS je potřeba si uvědomit také, že díky konvolučním filtrům, které mají tzv. sdílené váhy, není počet vah až tak velký s porovnáním počtu vah ve vrstvách perceptronů.

Rozměr výstupního obrazu z KV se změní díky konvoluci podle rovnice:

$$n_g = n_f - n_h + 1 \qquad (3)$$

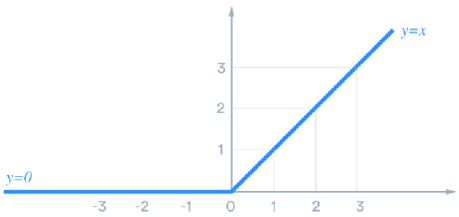
kde n_f je rozměr čtvercového obrazu, n_h je rozměr čtvercového filtru a n_g je rozměr výstupního obrazu.

Pokud má KNS mnoho konvolučních vrstev, a v každé konvoluční vrstvě se obraz zmenší, je dobré před vstupem do Konvoluční vrstvy rozšířit rozměry obrazu o nulové pixely. Tato úprava se nazývá tzv. „Padding“, a díky ní se rozměry vstupního obrazu do KV zachovávají.

4.4 Aktivační funkce

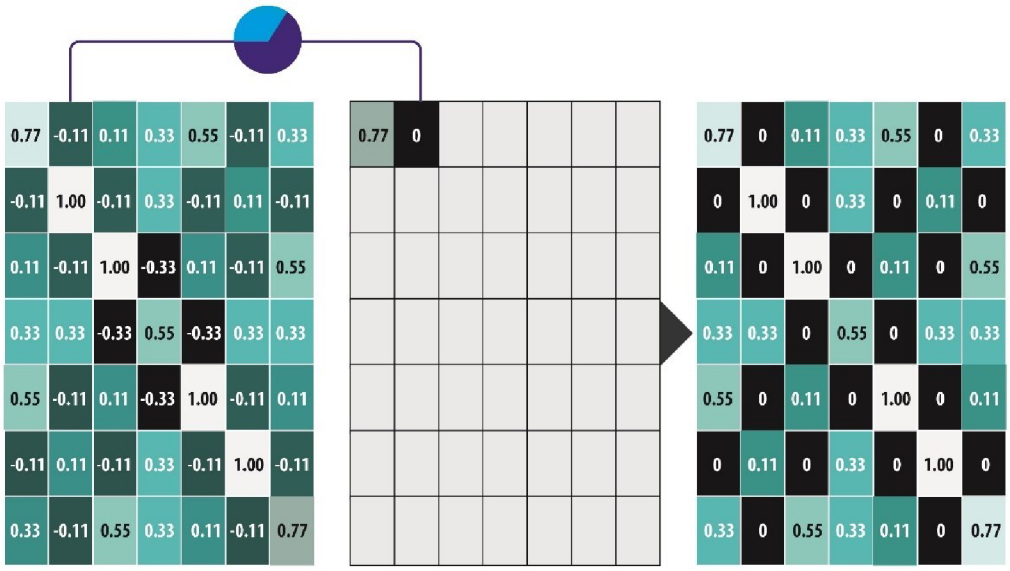
Záporné hodnoty výstupního obrazu z konvoluční vrstvy jsou ještě před vstupem do další vrstvy upraveny pomocí metody vnitřní normalizace. Tato metoda upraví hodnoty pixelů pomocí aktivační funkce. Nejčastěji se používá pro tento účel funkce relu, která má v celé negativní části osy x výstupní hodnotu 0 (nula) a v kladné časti osy x funkce zachovává identitu.

$$y = f(x) = max(0, x) \tag{4}$$



Obr. 4-7 Funkce Relu
(Zdroj: [22])

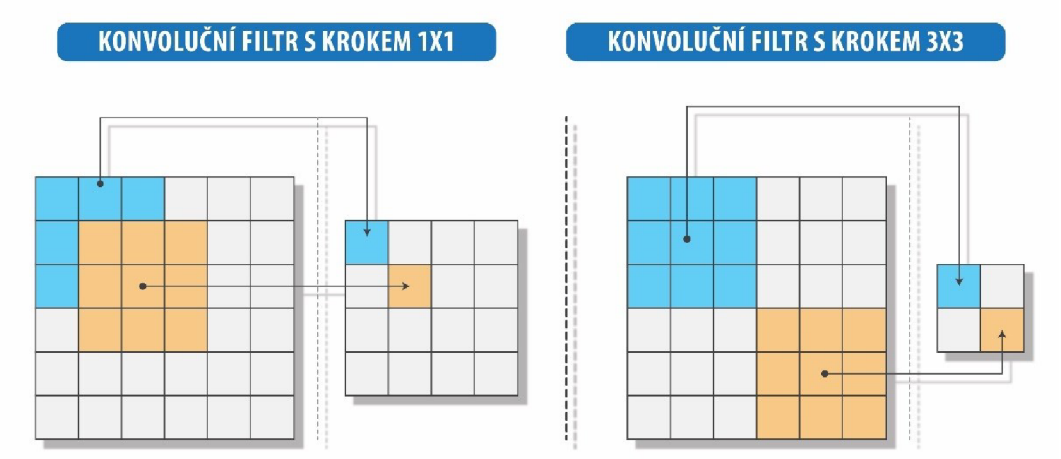
Tato normalizace dat kompenzuje rozdíly hodnot mezi jednotlivými pixely výstupního obrazu, díky tomu všechny pixely mají stejnou váhu při dopředné propagaci. Normalizace dat se proto uplatňuje na výstupu každé KV.



Obr. 4-8 Výstupní obraz po aplikaci Normalizace

4.5 Krok filtru

Krok filtru je vlastnost jednotlivých filtrů, která určuje o kolik pixelů se filtr posune při konvoluci. Krok filtru tak ovlivňuje velikost výstupního obrazu. Čím větší je krok, tím menší je výstupní obraz z KV. Při zvolení příliš velkého kroku je možné, že filtr začne ztrácet schopnost správně rozpoznat charakteristické objekty v obrazci, proto je potřeba krok filtru volit v rozumné velikosti.



Obr. 4-9 Krok filtru (step)

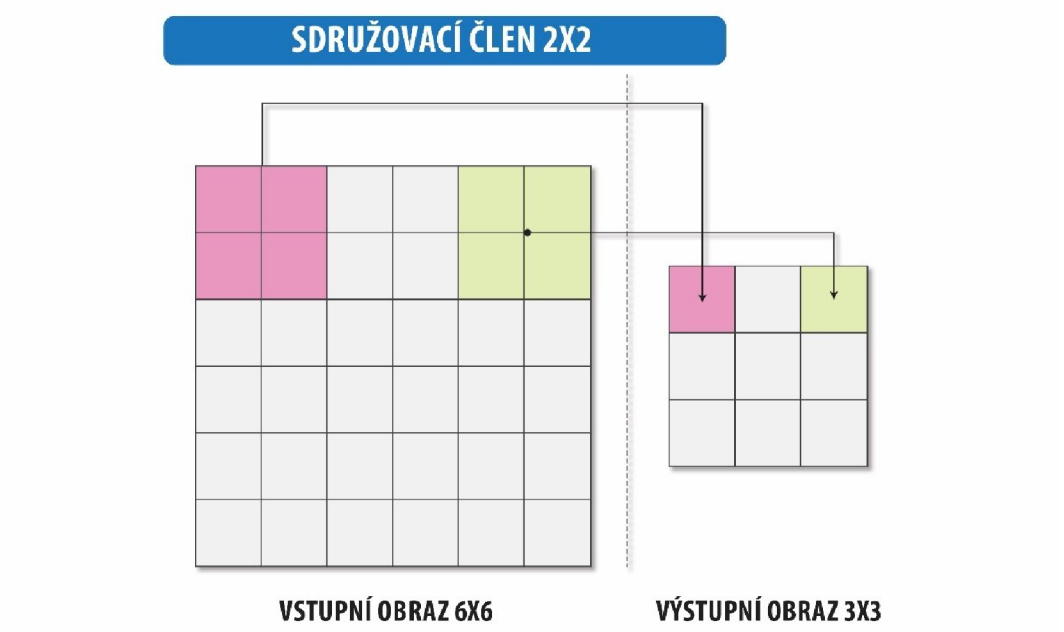
4.6 Sdružovací vrstva

Sdružovací vrstva (pooling layer) se většinou vkládá mezi konvoluční vrstvy tak, aby redukovala rozměry obrazů. Tato vrstva prochází podobně jako konvoluční vrstva obraz pomocí sdružovacího okna, který však nenásobí jednotlivé pixely, ale hledá pixel s maximální hodnotou nebo průměruje hodnoty pixelů v sdružovacím okně a ty dále předává na výstup sdružovací vrstvy.

$$p_{mean}(x,y) = \frac{1}{N}\sum_{a=-n}^{n}\sum_{b=-m}^{m} f(x-a, y-b) \tag{5}$$

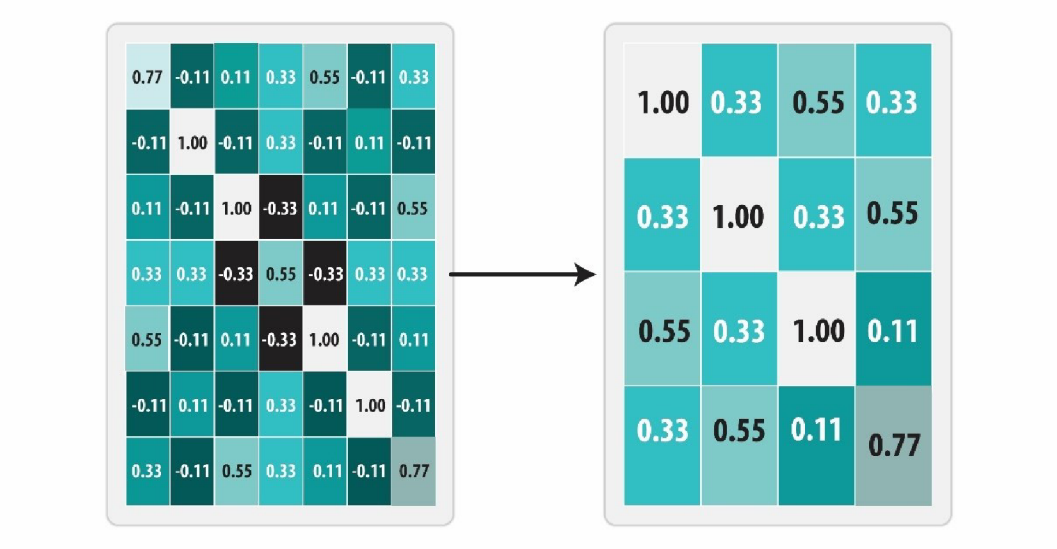
$$p_{max}(x,y) = p_{max}(f(x,y)) \tag{6}$$

kde $f(x,y)$ je funkce obrazu a N reprezentuje počet sdružovacích hodnot v sdružovacím okně. Na Obr. 4-10 lze vidět redukci vstupního obrazu 6x6 na obraz 3x3.



Obr. 4-10 Princip sdružovací vrstvy

Na Obr. 4-11 lze vidět příklad výstupního obrazu ze sdružovací vrstvy s maximální hodnotou. Je vidět, že i po redukci velikosti obrazu, lze rozeznat umístění hledaných segmentů v obraze, v tomto případě umístění horizontální čáry.

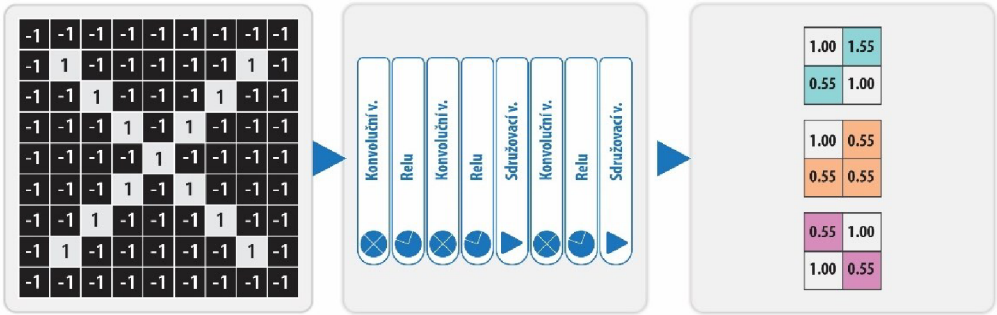


Obr. 4-11 Výstupní obraz ze sdružovací vrstvy

4.7 Posloupnost vrstev:

Posloupnost jednotlivých vrstev je závislá na několika faktorech, především na rozměrech vstupních dat, charakteru dat a komplexnosti řešeného problému. Například KNS, která rozeznává pouze psaný text, bude mít menší počet jednotlivých vrstev, než KNS která rozpoznává pozici auta v obraze. Proto složení a posloupnost vrstev se v každé KNS liší a při řešení problémů se zkouší různá kombinace vrstev.

Na Obr. 4-12 lze vidět jedna kombinace vrstev, které na výstupu má 3 obrazy. Tyto obrazy, které dále postupují do poslední části KNS a to do části s plně propojené vrstvy (fully connected layer), která je vstupní vrstvou pro vrstvy perceptronů.

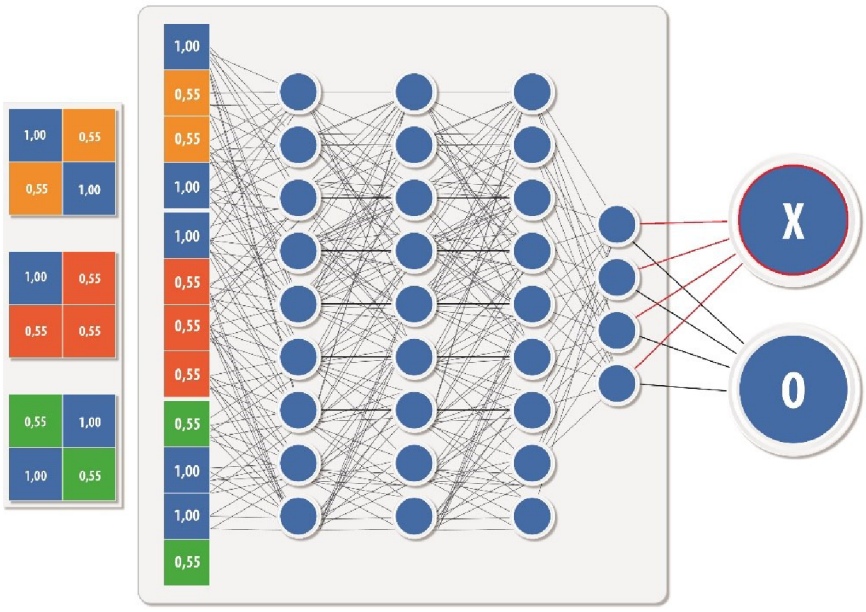


Obr. 4-12 Posloupnost a složení jednotlivých vrstev

4.8 Vrstvy perceptronů

Tato poslední část KNS je stejná, jak běžná neuronová síť složená z vrstev perceptronů. Na začátku se výstupní obrazy z konvoluční části transformují z 2D na 1D rozměr a poté vstupují do vrstev perceptronů.

Samotná konvoluční část měla za úkol rozpoznat určité segmenty v obraze, avšak pro samotnou predikci je potřeba i rozhodovací struktura vrstev perceptronů. Tyto vrstvy dokážou na základě identifikovaných segmentů z konvoluční části určit, o jakou třídu či predikci se jedná. Na Obr. 4-13 můžete vidět dopřednou propagaci vrstvami perceptronů a správnou klasifikaci písmene X.



Obr. 4-13 Plně propojené vrstvy

Na začátku procesu učení jsou všechny váhy perceptronů a konvolučních filtrů náhodně zvolené. Při procesu učení jsou tyto váhy upravovány tak, aby dokázaly vyřešit daný problém, na který byly vytrénovány.

4.9 Ztrátová funkce

Ztrátová funkce definuje, jak moc dobrá či špatná byla predikce z KNS. Jako vstup do ztrátové je samotná predikce KNS a označení skutečného výsledku (label). Díky tomu je ztrátová funkce schopna porovnat tyto dvě hodnoty.

Ztrátová funkce $Z(y, p)$ je funkce s názvem Cross-entropy.

Měrná suma ztrátových funkci všech dat, určených pro proces učení je pak J.

$$Z(y,p) = -\left[\mathrm{y}\log\left(p\right) + (1-\mathrm{y})\log\left(1-\mathrm{p}\right)\right] \tag{7}$$

$$J = -\frac{1}{m}\sum_{i=1}^{m}\left[Z(y_i, p_i)\right] \tag{8}$$

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

kde hodnota p je predikce KNS (odhad) a y je skutečná hodnota řešeného problému.

Další známou ztrátovou funkci je i funkce Mean squared error (MSE), neboli Střední kvadratická chyba.

$$mse(y,p) = -(\mathrm{p}-\mathrm{y})^2 \tag{10}$$

$$MSE = -\frac{1}{m}\sum_{i=1}^{m}(p_i - y_i)^2 \tag{11}$$

4.10 Aktivační funkce

Aktivační funkce mají jedny z nejdůležitějších rolí v každé neuronové síti, protože hlavně díky těmto funkcím a jejím nelinearitám jsou NS schopné vlastně řešit veškeré nelineární problémy.

Nelinearita funkce

Kdyby byly za aktivační funkce zvoleny pouze lineární funkce, byl by výstup NS vždy lineárně závislý na vstupu. Nezáleželo by ani na tom, jak mnoho by bylo jednotlivých vrstev. V tomto případě by nebylo možné řešit žádné nelineární

problémy.

Aktivační funkce

Aktivační funkci je třeba zvolit, jak do konvolučních vrstev, tak i do vrstev perceptronů. Podle zvolených aktivačních funkcí je potřeba upravit také vstupní data, které by měly vstupovat do KNS. Vstupní data by měly mít stejný číselný rozsah, jak aktivní část aktivačních funkcí. Aktivní rozsah aktivační funkce se myslí ta část, která má derivaci značně různou od nuly. Například u funkce tanh (tangents hyperbolický) má aktivní část rozsah vstupních hodnot od -1 do 1. Je potřeba tedy vstupní data ještě před samotným procesem učení upravit tak, aby měla stejný rozsah, jak zvolená aktivační funkce. Díky tomu se zajistí rychlejší a stabilnější průběh učení.

V případě funkce sigmoid je aktivní část funkce v rozsah od 0 do 1, stejně tak i u funkce relu. Dobrým zvykem při volbě aktivační funkce je zvolit si jednu aktivační funkci pro všechny vrstvy. Je možné v některých případech kombinovat i více aktivačních funkcí, avšak tyto kombinované funkce by měly mít, alespoň stejný aktivní rozsah, aby se zabránilo nestabilnímu chování.

Jedinou výjimkou při volbě aktivační funkce zůstává poslední výstupní vrstva, která se volí v závislosti na řešeném problému. V případě binární klasifikace, kdy je výstupem pouze jedna hodnota z perceptronu, se jako výstupní aktivační funkce volí většinou funkce sigmoid. Tato funkce udává pravděpodobnost příslušnosti k jednotlivé třídě, kde hodnota blížící se k 0 vyjadřuje příslušnost do třídy A a hodnota jdoucí k 1 zas příslušnost třídy B. V případě klasifikace do více tříd, kde se počet výstupních hodnot rovná počtu klasifikačních tříd, se využívá funkce softmax.

4.11 Backpropagation

Backpropagation neboli zpětná propagace (ZP), je proces učení celé NS. Právě díky zpětné propagaci jsou NS schopné upravovat své váhy a řešit tak nelineární problémy. Tyto úpravy vah probíhají na základě zpětně propagované chyby získané ze ztrátové funkce.

Postupnou zpětnou propagací se propaguje chyba každou vrstvou od výstupní vrstvy až k vstupní vrstvě. Tedy v opačném pořadí, než jak to bylo u dopředné propagaci. A při této zpětné propagaci dochází v každé vrstvě perceptronů a v konvolučních vrstvách k výpočtu přírůstků vah, které svou změnou ovlivní příští dopřednou propagaci a při správné funkci sníží tak i chybu predikce.

Proces ZP začíná od derivací ztrátové funkce a poté postupuje jednotlivými vrstvami dále [2]. V průběhu zpětné propagace je hodnota derivace ovlivněna jak jednotlivými váhami vrstev, tak i derivacemi aktivačních funkcí.

Derivace ztrátové funkce

Derivace funkce Cross-entropy $Z(y,p)$, vytváří vstupní hodnoty do zpětné propagace.

$$\frac{\partial Z(y,p)}{\partial \mathrm{p}} = \frac{\partial(-\mathrm{y}\log(p) - (1-y)\log(1-p))}{\partial \mathrm{p}}$$

$$= \frac{\partial(-y\log(p))}{\partial \mathrm{p}} + \frac{\partial(-(1-y)\log(1-p))}{\partial \mathrm{p}} \tag{12}$$

$$= -\frac{y}{p} + \frac{1-y}{1-p} = \frac{p-y}{p(1-p)}$$

V případě použití funkce $mse(y,p)$, je derivace zpětné funkce o něco méně složitá.

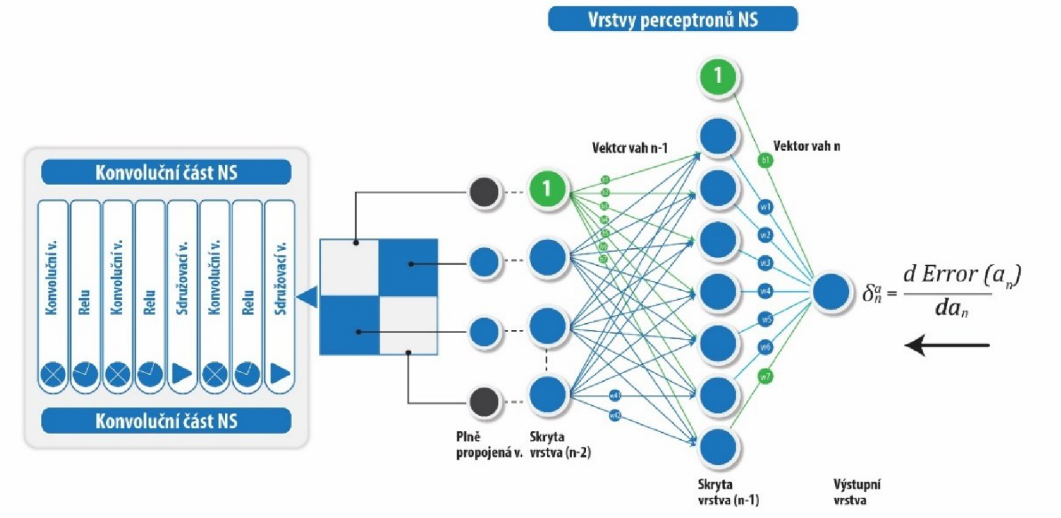
$$\frac{\partial \mathrm{mse}(y,p)}{\partial \mathrm{p}} = \frac{\partial(-(p_i - y_i)^2)}{\partial \mathrm{p}} = -2\,(p_i - y_i) \tag{13}$$

Zpětná propagace ve vrstvách perceptronů

Pro samotnou zpětnou propagaci chyby nestačí pouze derivace ztrátových funkcí a derivace aktivačních funkcí. Situace je o něco složitější, jelikož je zapotřebí mít uložené i všechny hodnoty z dopředné propagace NS. To znamená, že

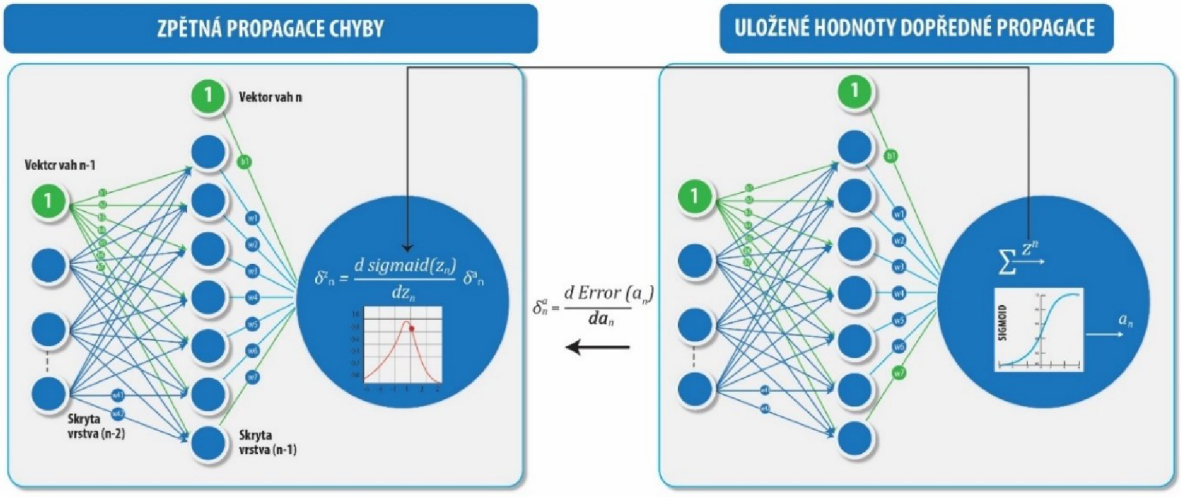
[2] Jelikož se pomocí zpětné propagace prochází od koncové (výstupní) vrstvy ke první (vstupní) vrstvě jsou indexy jednotlivých vrstev označovány písmenem **n,** každá další vrstva ve směru zpětné propagace je pak označená indexem **n-1,2,3**…

naprogramovaný systém pro práci s NS musí být schopný, si při dopředné propagaci ukládat každou propagaci vstupních hodnot přes jednotlivé vrstvy. Každý průchod jakoukoliv vrstvou musí by tedy uložen, aby bylo možné vytvořit správně fungující backpropagation proces.



Obr. 4-14 Zpětná propagace

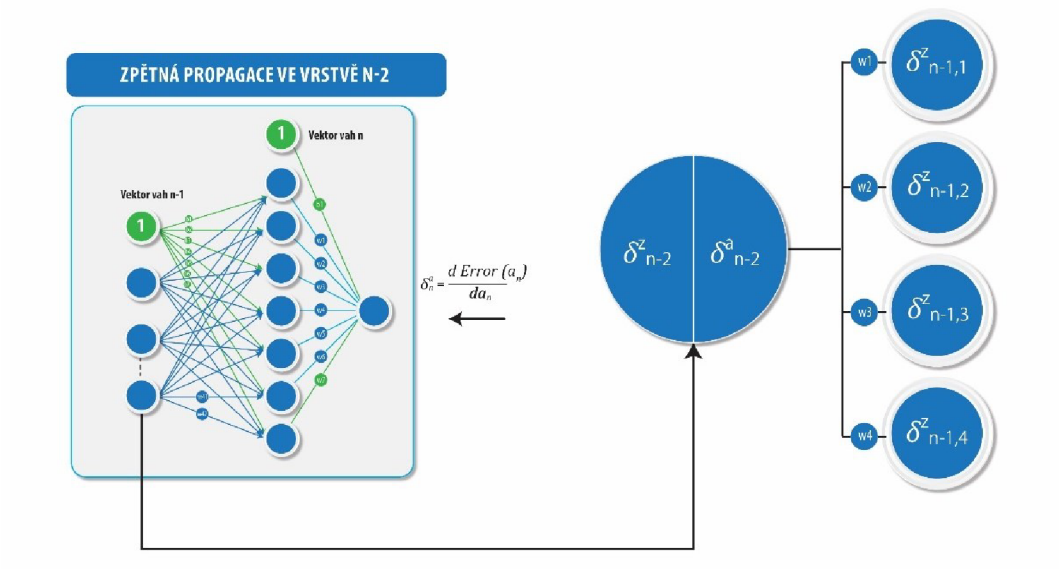
Backpropagation začíná derivací ztrátové funkce, poté tato derivace postupuje do n-1 vrstvy.



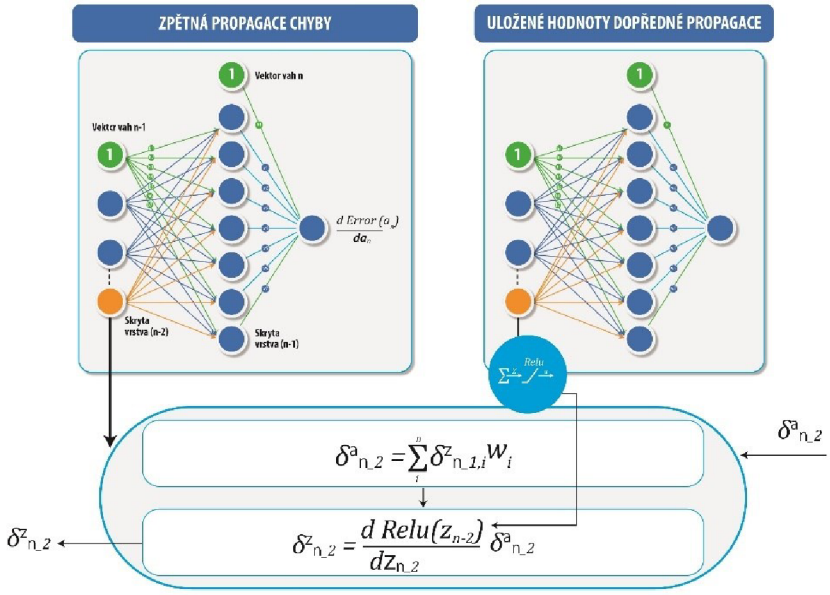
Obr. 4-15 Zpětná propagace v poslední vrstvě

Na Obr. 4-15 lze vidět ZP přes aktivační funkci sigmoid, jak je naznačeno na obrázku, pro tuto ZP je potřeba také hodnota perceptronu Z_n získaná z dopředné

propagace vrstvou n.



Obr. 4-16 Zpětná propagace ve skryté vrstvě perceptronů - 1.část

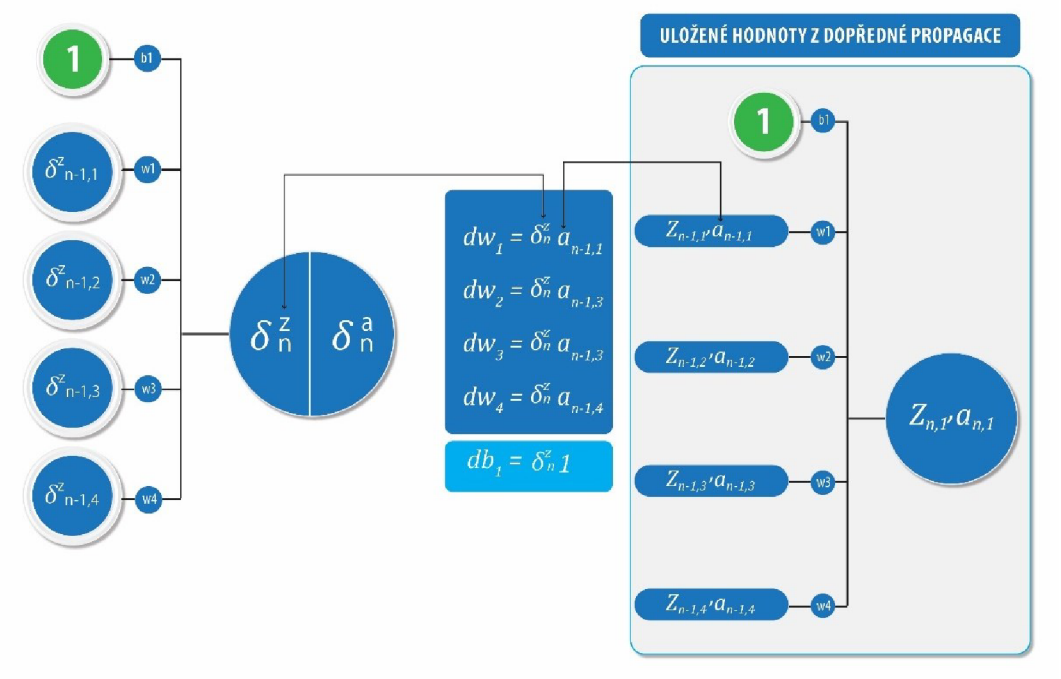


Obr. 4-17 Zpětná propagace ve skryté vrstvě perceptronů - 2.část

Na Obr. 4-16 a Obr. 4-17 můžete vidět ZP ve skryté vrstvě perceptronů. Při průchodu každou skrytou vrstvou perceptronů, je první krok výpočet pomocí sumy hodnot δ^a, která je ovlivněna hodnotami vah předešlé vrstvy. Druhý krok ZP se vypočítá přes derivaci aktivační funkce hodnota δ^z, která přechází dále pak do další vrstvy. Derivace aktivační funkce je počítána v hodnotě Z, z dopředné

propagace.

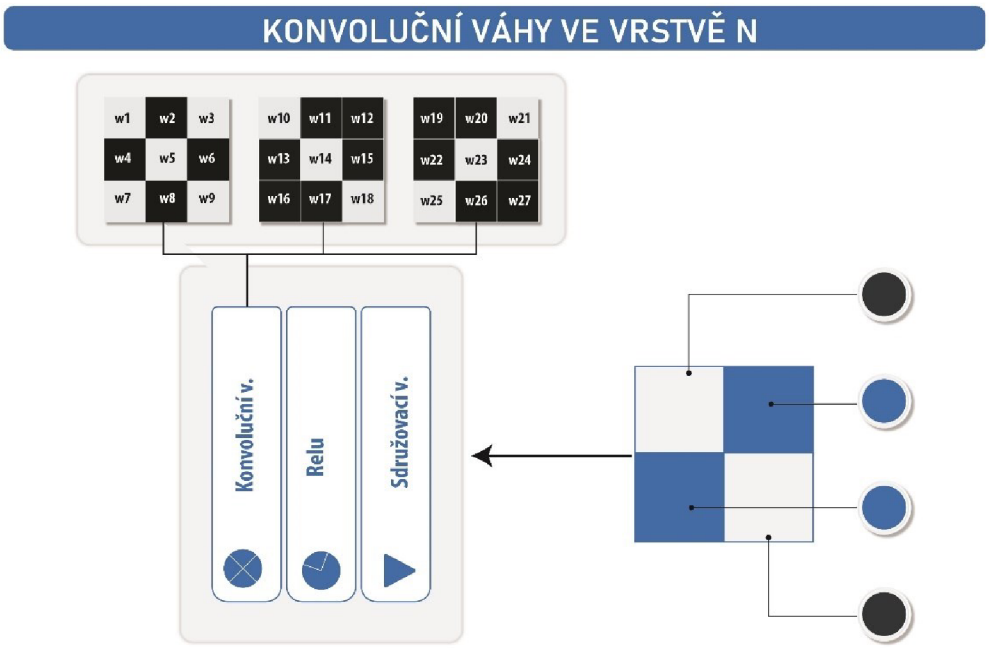
Ještě před ZP v další vrstvě dochází k výpočtu přírůstků vah v dané skryté vrstvě. Podobně jako zpětně propagovaná hodnota, tak i výpočet těchto přírůstků vah potřebuje jako vstup uložené hodnoty z dopředné propagace a hodnoty z aktivační funkce z vrstvy o jednu menší, v případě výpočtu ve vrstvě **n** z vrstvy **n-1**.



Obr. 4-18 Výpočet přírůstku vah perceptronů

Po průchodu poslední perceptronovou vrstvou vstupují zpětně propagované hodnoty do plně propojené vrstvy, kde se tyto hodnoty transformují z 1D rozměru zpět na 2D rozměr. Tento proces je potřeba pro zpětnou propagaci v konvoluční částí, jelikož tato část umí pracovat pouze 2D daty.

Z plně propojené vrstvy přecházejí data do konvoluční části, kde přecházejí jak aktivačními, sdružovacími tak i konvolučními vrstvami. V každé této vrstvě se hodnoty zpětné propagace upravují v závislosti na typu dané vrstvy. Pouze v konvoluční vrstvě se vypočítávají také přírůstky vah konvolučních filtrů, které ovlivňují samotnou dopřednou propagaci NS.



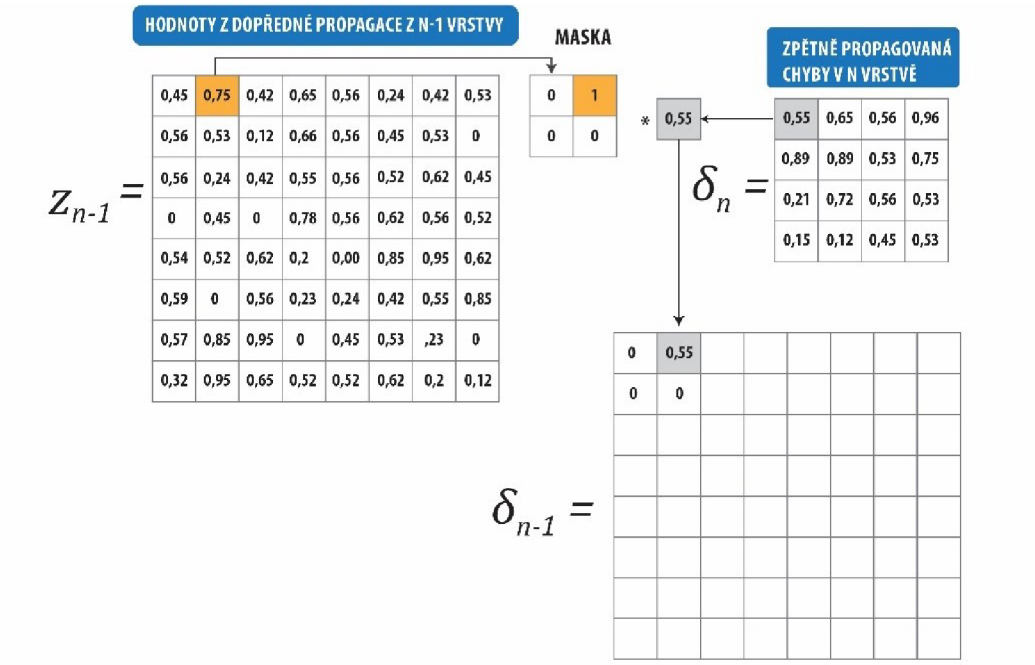
Obr. 4-19 Plně propojená vrstva a konvoluční váhy

Zpětná propagace sdružovací vrstvou

Sdružovací vrstva (SV) v dopředné propagaci slouží k redukci vstupních dat, což znamená ke zmenšení rozměrů daného obrazu. Naopak při zpětné propagaci chyby dochází k obnově původní velikosti obrazu. Jako sdružovací vrstvy byly představeny dva typy a každá z těchto vrstev má odlišnou formu samotné zpětné propagace. Proto je potřeba rozlišovat, zda v dané vrstvě byla použita SV s maximální nebo průměrnou hodnotou.

- Sdružovací vrstva s maximální hodnotou

Při zpětné propagaci s maximální hodnotou se při prvním kroku musí vytvořit maska na základě maximální hodnoty redukčního okna z dopředné propagace z dané vrstvy a poté se tato maska vynásobí hodnotou ze zpětné propagace.

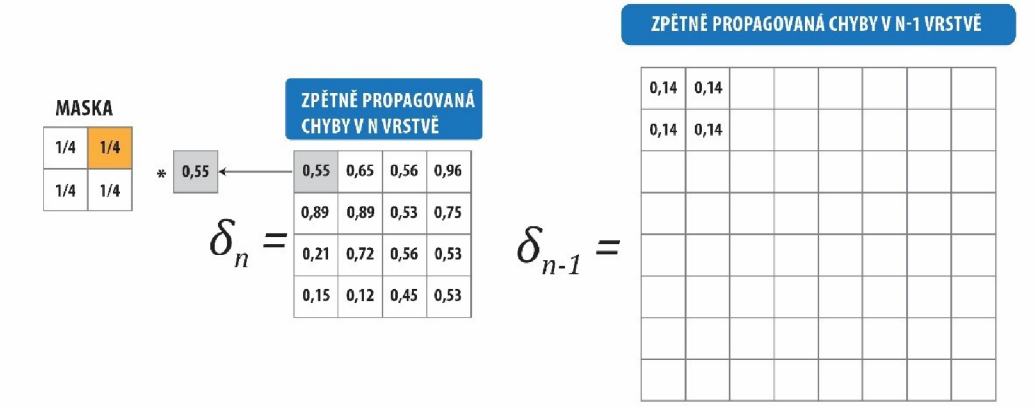


Obr. 4-20 Zpětná propagace sdružovací vrstvou – Max

Na Obr. 4-20 lze vidět, že při tvorbě masky se hledá vždy maximální hodnota z hodnot dopředné propagace z daného okna. Tato maximální hodnota byla totiž zachována při redukci dat sdružovací vrstvou, a proto se propaguje i nazpět. Samotná maska se tedy poté vynásobí hodnotou ze zpětné propagace. Tato zpětně propagovaná hodnota se bude nacházet na té stejné pozici jako zachovalá hodnota z dopředné propagace.

- Sdružovací vrstva s průměrnou hodnotou

V případě ZP v sdružovací vrstvě s průměrnou hodnotou je maska tvořena na základě velikosti redukčního okna vrstvy.

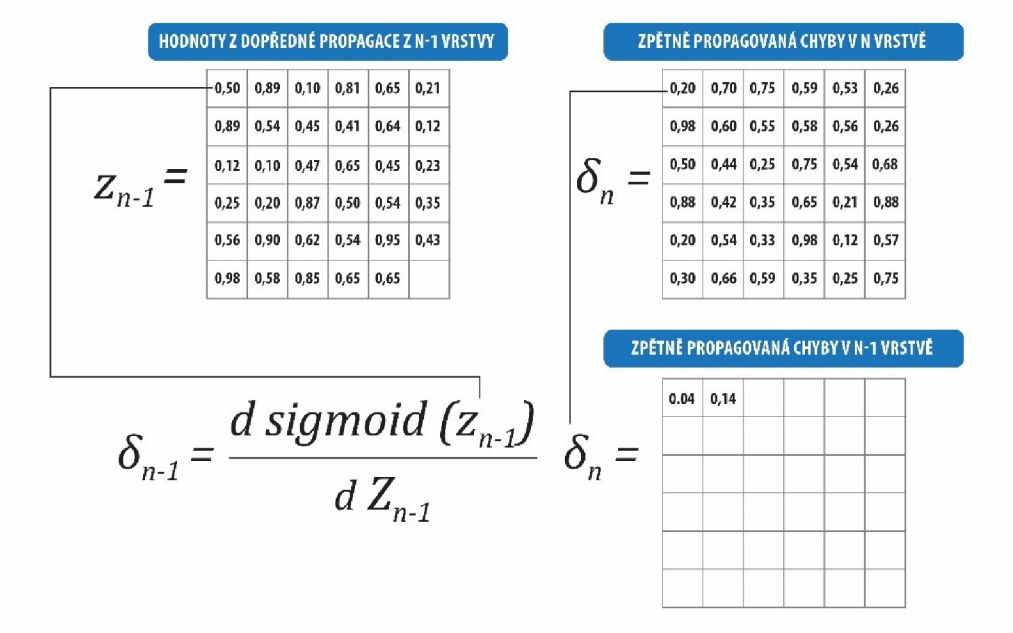


Obr. 4-21 Zpětná propagace sdružovací vrstvou – Průměrná funkce

Na Obr. 4-21 lze vidět příklad, kdy redukční okno mělo velikost 2x2 (4 pole), a proto všechny hodnoty masky mají hodnotu ¼. Tato maska se poté vynásobí hodnotou ze ZP podobně, jak u sdružené vrstvy s maximální hodnotou. Na rozdíl od maximální hodnoty se zpětně propagovaná chyba nepřenáší pouze do pole s maximální hodnotou, ale do všech polí daného redukčního okna rovnoměrně.

Zpětná propagace aktivační vrstvou

Zpětná propagace (ZP) aktivační vrstvou pracuje na podobném principu, jako ZP aktivační funkci ve vrstvách perceptronů. Jako vstupní hodnoty do tohoto procesu vstupují zpětně propagovaná hodnota δ z předchozí vrstvy, a také uložené hodnoty Z, z dopředné propagace dané vrstvy.

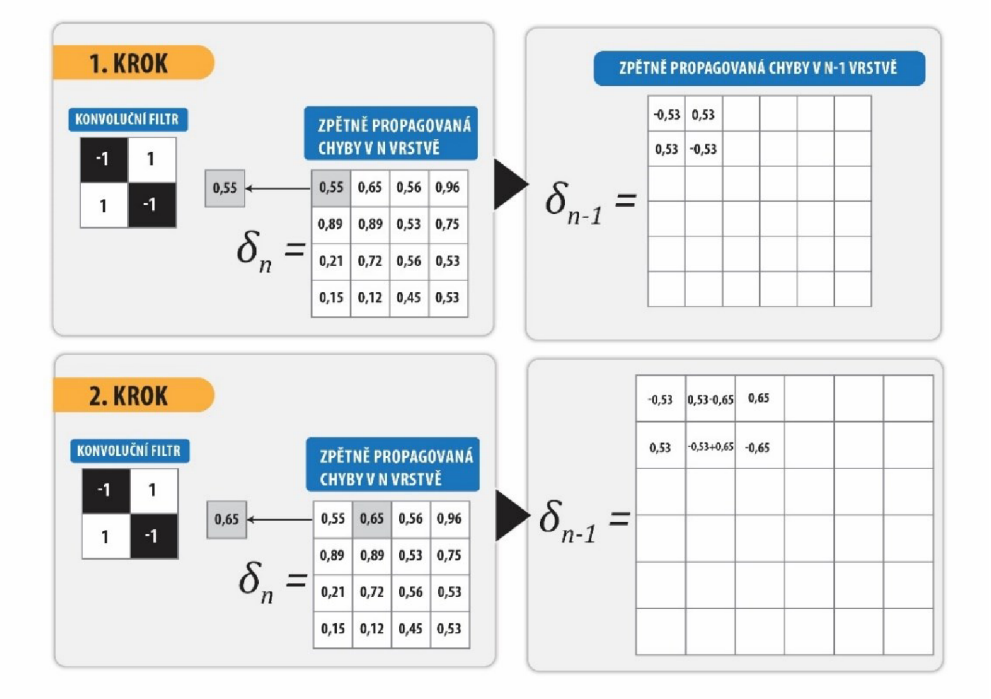


Obr. 4-22 Zpětná propagace aktivační vrstvou

Na Obr. 4-22 lze vidět vizuální zobrazení ZP z vrstvy n do vrstvy n-1. Po výpočtu zpětné propagované chyby postupuje hodnota dále do dalších vrstev.

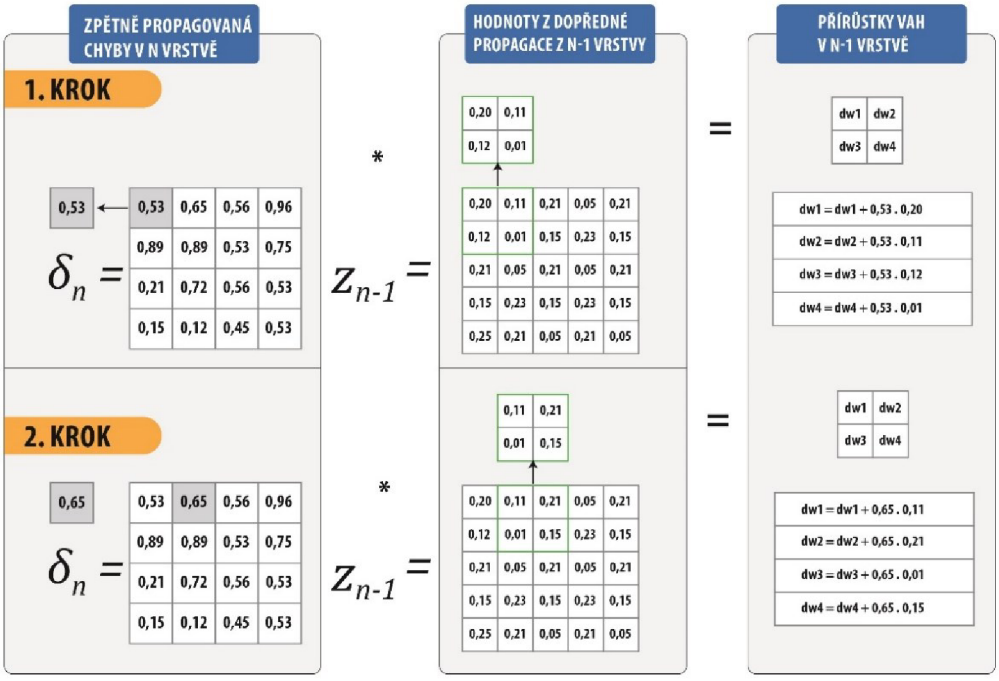
Zpětná propagace konvoluční vrstvou

Při ZP konvoluční vrstvou je situace o něco málo složitější, jelikož v této vrstvě se kromě zpětné propagace chyby vypočítávají také přírůstky vah konvolučních filtrů.



Obr. 4-23 Zpětná propagace konvoluční vrstvou

Na Obr. 4-23 lze vidět 2 kroky ZP. Lze si i všimnout, že krok filtru v konvoluční vrstvě ovlivňuje také i samotnou ZP, a v tomto případě krok konvolučního filtru byl zvolen jedna [3].



Obr. 4-24 Výpočet přírůstku konvolučních vah

[3] Vliv „paddingu“ není zde popsán

Na Obr. 4-24 jsou zobrazeny 2 kroky výpočtu přírůstku vah filtru v konvoluční vrstvě. Právě díky těmto úpravám dokážou samotné filtry rozpoznávat určité segmenty v obrazu, které jsou pro řešený problém specifické.

Aktualizace vah

Po průchodu celé KNS zpětnou propagací se hodnoty vah ve vrstvách perceptronů a ve vrstvách konvolučních aktualizují. Tyto nové váhy vstupují potom dále do další dopředné propagace. Velikost úpravy vah, neboli rychlost učení, je navíc ovlivněna i koeficientem učení, který má hodnotu od 0 do1. Tímto koeficientem se zpomaluje proces učení v případech, že proces učení probíhá příliš rychle a hrozí nestabilní chování.

$$w = w - dw \,.\, k_{lr} \tag{14}$$

$$b = b - db \,.\, k_{lr} \tag{15}$$

5 PROCES UČENÍ A OPTIMALIZACE KNS

Ještě před samotným procesem učení je zapotřebí vždy stanovit mnoho parametrů sítě, které mají zásadní vliv na samotný průběh učení. Mezi tyto parametry patří například počet vrstev NS, pořadí jednotlivých vrstev, koeficient učení, druh aktivační funkce, velikosti filtrů a mnoho dalších. Při každé nové aplikaci NS je vždy náročné uhodnout správné nastavení těchto parametrů, a je potřeba formou pokus omyl vždy vyzkoušet několik nastavení. První nastavení probíhá většinou na základě vlastních zkušeností z předchozích aplikacích NS.

Hned po prvním běhu učení nově nastavené NS se prokáže, jak dobře si toto nastavení vedlo. Z výsledků lze pak usazovat do jisté míry jaké parametry je třeba pozměnit, a co by mělo zůstat zachováno. Například jestliže proces učení probíhá velice pomalu, je třeba zvýšit koeficient učení, ale je potřeba pohlídat, aby nebyl zase příliš velký. To by vedlo k nestabilnímu učení, ve kterém by ztrátová funkce stoupala, stagnovala nebo oscilovala. Ne všechny potřebné úpravy lze rozpoznat z výsledků. Některé parametry je potřeba nastavit a ověřit si jejich vliv. Úprava parametrů se provádí iterativně do konečné fáze, kdy je dosaženo, co možná nejlepších výsledků učení dané NS.

5.1 Vstupní data

Dalším důležitým aspektem, který ovlivňuje zásadně učení NS jsou i samotná vstupní data. Vstupní data by měla mít stejné rozsahy, stejnou strukturu a stejný formát pro všechny vstupní hodnoty. Není vhodné tím pádem, aby některé obrázky byly v barevném formátu RGB a jiné zas ve formátu HSV. Také je potřeba zajistit, aby všechny obrázky měly stejný počet vrstev. Nelze tedy použít ani černobílé obrázky s barevnými.

5.2 Příprava vstupních dat

Příprava vstupních dat zahrnuje především techniky normalizace a standardizace. Tyto dvě techniky jsou založené na posuvu a škálování vstupních dat do požadovaného rozsahu, který zajistí vhodný průběh procesu učení NS.

Normalizace

Normalizace je změna rozsahu vstupních dat z původního rozsahu do rozsahu od 0 do 1. Při normalizaci je potřeba správně určit minimální a maximální hodnotu, kterou mohou nabývat vstupní hodnoty.

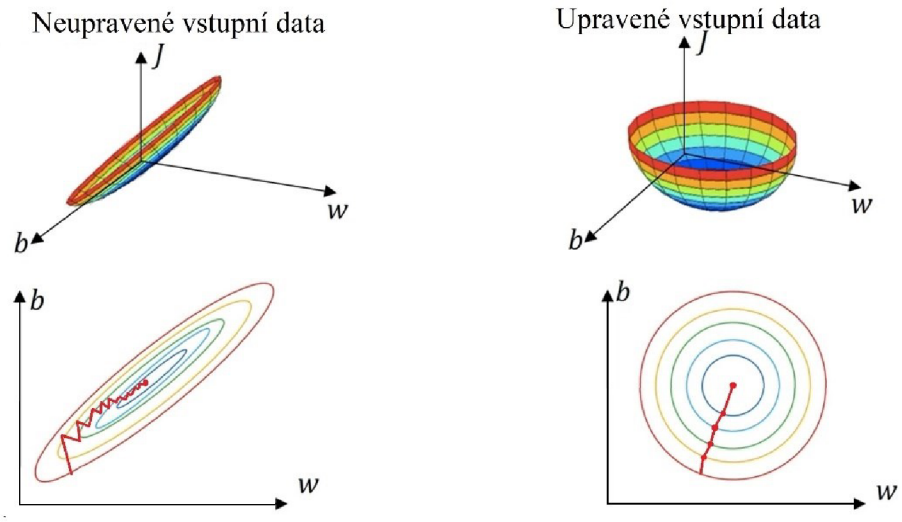
Proces normalizace se skládá ze dvou kroků, jako první se odečte minimální hodnota od všech vstupních hodnot, a poté se tato hodnota vydělí původním rozsahem (maximum – minimum).

$$\mathbf{X}_{\text{new}} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{16}$$

Standardizace

Standardizace vstupních dat je podobně jako normalizace, změna rozsahu vstupních dat. Pro správný proces standardizace je potřeba určit střední hodnotu a směrodatnou odchylku vstupních dat. Jako první se od vstupních hodnot odečte střední hodnota, a poté se tyto hodnoty vydělí směrodatnou odchylkou [23].

$$z = \frac{x_i - \mu}{\sigma} \tag{17}$$



Obr. 5-1 Vliv úpravy vstupních dat na proces učení
(Zdroj: [24])

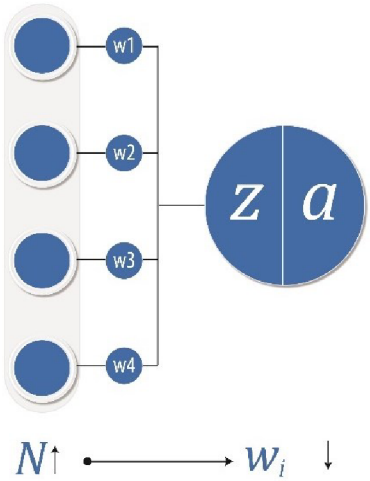
Na obrázku Obr. 5-1 lze vidět jak normalizace / standardizace vstupních dat upravuje n-dimenzionální prostor. Jsou-li data neupravená před vstupem do NS, je proces učení neboli proces hledaní minima ztrátové funkce většinou mnohem pomalejší, než když jsou data upravená správně. Při procesu učení s neupravenými daty může dojít až ke stagnaci neboli uvíznutí v lokálním minimu ztrátové funkce.

Promíchání vstupních dat

Dobrým zvykem je také samotná vstupní data promíchat, aby se zajistilo nezávislosti učení na posloupnosti vstupních dat.

5.3 Inicializace vah

Při inicializaci vah NS je zapotřebí zvolit takové hodnoty vah, díky kterým bude zpětná propagace spadat do nenulové derivace aktivačních funkcí. U aktivační funkce relu a sigmoid je tato oblast v rozmezí od 0 do1. Proto je potřeba znát počet perceptronů v předešlé vrstvě.



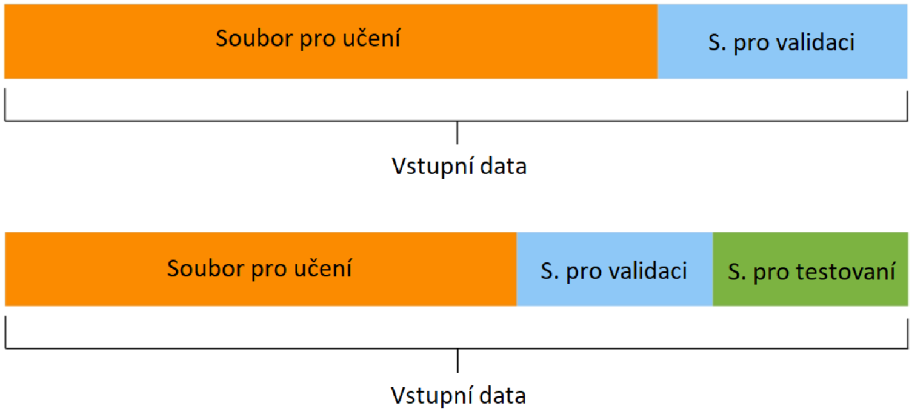
Obr. 5-2 Vliv vstupních perceptronů na inicializaci vah

Na Obr. 5-2 můžete vidět schéma perceptronu, do kterého vstupují 4 hodnoty z perceptronů vynásobené váhami. V případě, že by se počet perceptronů (N) na vstupu zvětšoval, je potřeba snižovat inicializované váhy, aby byla zachována nenulová derivace.

Váhy perceptronů se proto volí pro aktivační funkce sigmoid a relu s rozptylem $\frac{1}{N}$ a střední hodnotou 0,5, kde N je počet perceptronů na vstupu. V případě Konvoluční vrstvy počet N udává počet polí v daném filtru.

5.4 Rozdělení vstupních dat

Před učením je také vhodné si vstupní data rozdělit na dvě nebo tři kategorie pro samotné učení NS, a pak pro validaci výsledků. Největší část ze vstupních dat musí tvořit soubor dat pro proces učení. Další část je pak soubor dat pro validaci NS a třetí část dat, která se někdy vynechává, je soubor dat pro testovaní.



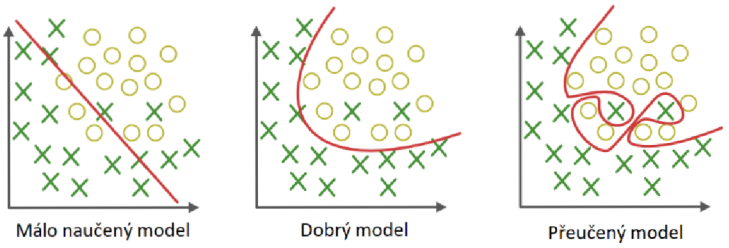
Obr. 5-3 Rozdělní vstupních dat

Pro validaci NS se v první fázi návrhu používá vždy pouze soubor dat pro validaci. Na závěr, když je stav naučení NS dostačující, je pro objektivnější posouzení použit i soubor dat pro testování.

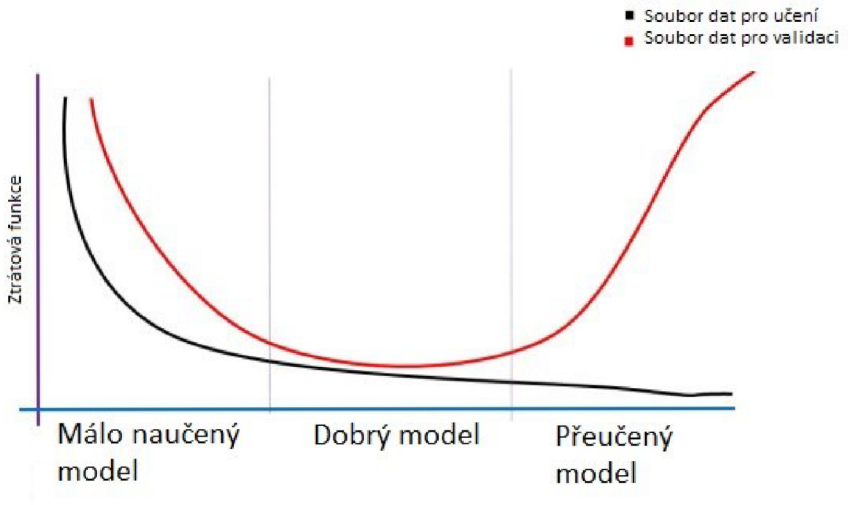
V zásadě je běžnou praxí rozdělovat vstupní data v podílu většinou 70%, 30% . V případě rozdělení vstupních dat na 3 soubory je potom rozdělení 70%, 15%, 15%, kde největší podíl je vždy soubor pro proces učení. Takové rozdělení je rozumné při počtu vstupních dat okolo 1000 nebo 10 000, avšak při velkém počtu vstupních dat není potřeba k otestovaní už celých 20%, ale stačí i mnohem menší podíl.

5.5 Délka učení

Při nastavení všech parametrů je potřeba také zajistit správnou délku učení NS. Při učení nastávají postupně tři stavy, jako první stav je stav, kdy neuronová síť je málo naučená (underfitting). Druhý stav je tzv. dobrý model NS (good model) a poslední stav je stav, kdy je NS přeučená (overfitting). Přeučení NS způsobí, že si NS až moc dobře pamatuje jednotlivé vzory ze souboru dat pro učení, a díky tomu správně dokáže identifikovat pouze vzorky z tohoto souboru. Ale schopnost identifikovat vzorky i z validačního a testovacího souboru se rapidně snižuje.



Obr. 5-4 Proces učení – stavy
(Zdroj: [25])



Obr. 5-5 Proces učení – ztrátová funkce
(Zdroj: [25])

Na Obr. 5-5 lze zpozorovat, jak se v první části grafu NS správně učí, a ztrátová funkce nad souborem dat pro učení i pro validaci konverguje k nule. Na konci procesu učení je vidět, jak ztrátová funkce nad souborem dat pro učení ještě více konverguje k nule. Ztrátová funkce nad souborem dat pro validaci naopak diverguje od nuly. Tento stav NS je pro aplikaci už nevhodný a v takovém případě je potřeba proces učení NS provést znova od začátku.

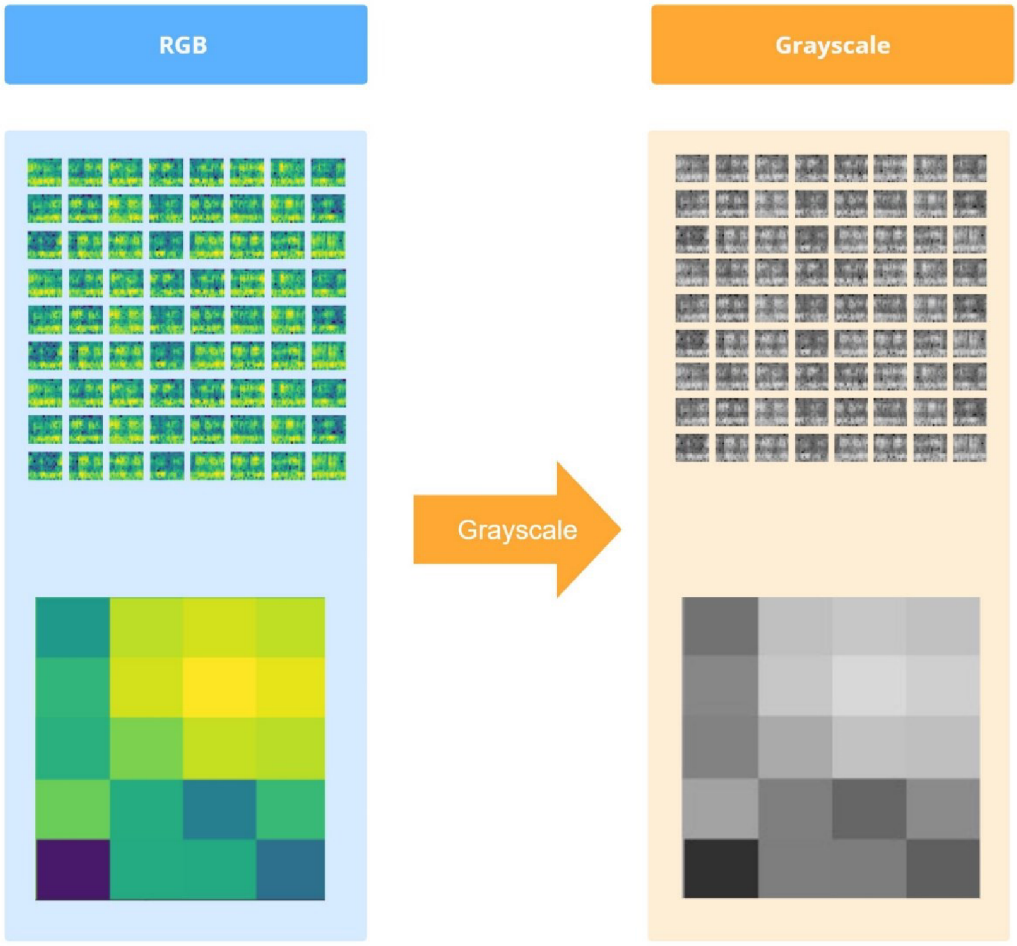
6 TESTOVÁNÍ KNS

KNS byla sestavena v programovacím jazyce python a pro její správné fungovaní byly použité pouze základní knihovny python jako jsou například Numpy, Matplotlib a Pillow knihovna.

Systém byl naprogramován pomocí vlastních funkci z důvodu, aby uživatel měl pod kontrolou každý proces propagace a mohl tak upravovat jednotlivé postupy a funkce sítě.

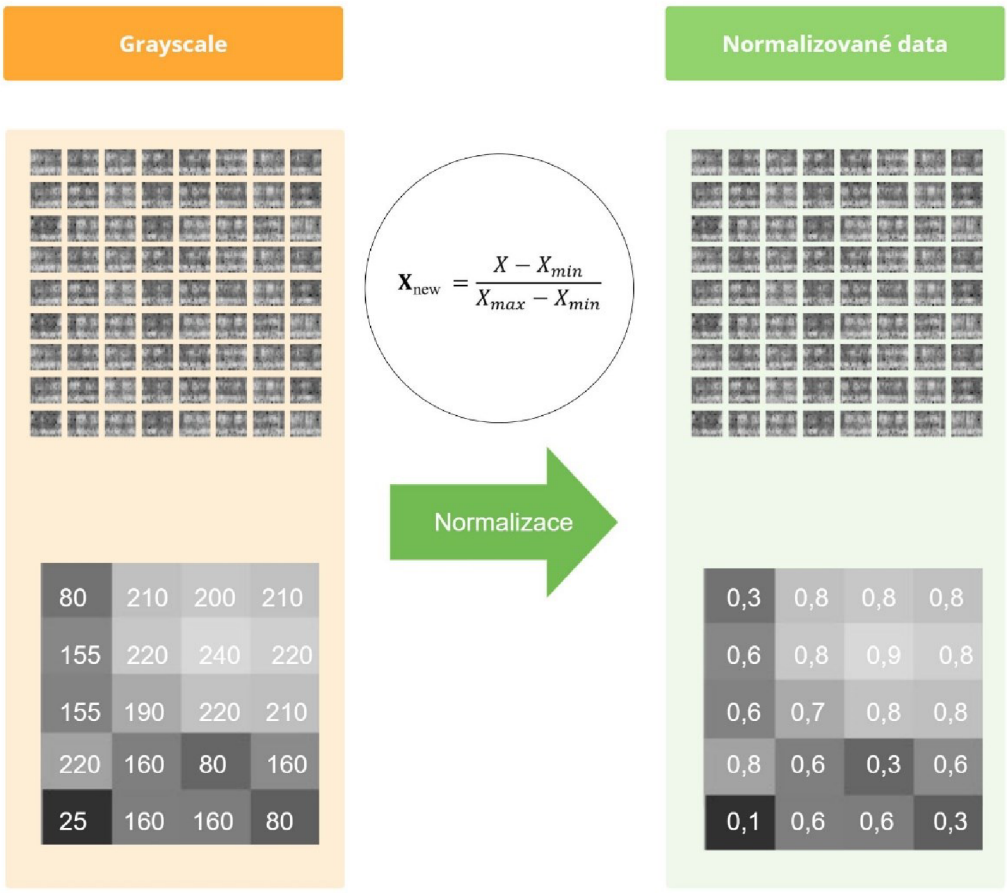
6.1 Proces učení

Před samotným procesem učení je potřeba upravit vstupní data. Jako první úprava datasetu je převedení všech spektrogramů z RGB vrstev na jednu černobílou vrstvu pomocí funkce grayscale.



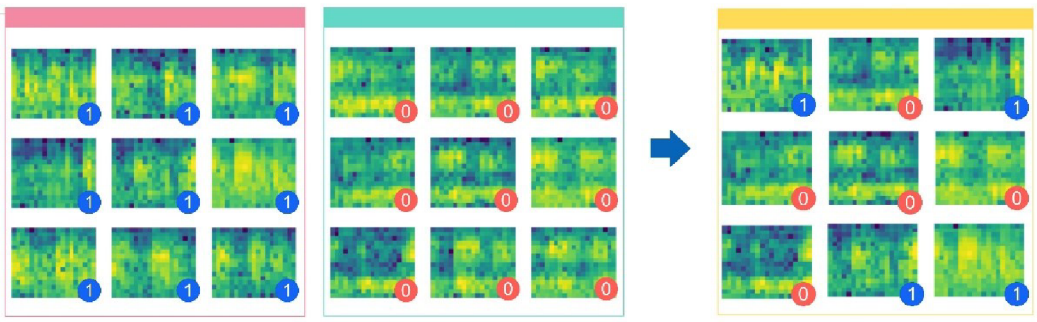
Obr. 6-1 Převod vstupních vah do stupňů šedi

Jako druhý krok úpravy dat byla normalizace všech hodnot pixelů neboli škálování rozsahu z 0 až 255 na rozsah 0 až 1.



Obr. 6-2 Normalizace vstupních dat

Posledním krokem úpravy dat bylo i promíchaní všech spektrogramů, díky čemuž proces učení není závislý na posloupnosti dat.



Obr. 6-3 Promíchání vstupních dat

Proces učení probíhá iteračně s 5 základními kroky. Při prvním běhu si proces učení načte upravená data a přiřazené označení (labels). Poté probíhá první dopředná

propagace všech dat, která vytváří predikci KNS. Tato predikce dále vstupuje do ztrátové funkce, která vypočítavá chyby predikce. V dalším kroku, pokud je chyba menší než 0,08, tak iterativní proces končí a výstupem celého procesu učení je hotový model KNS. V případě nesplnění podmínky následuje zpětná propagace chyby a úprava dat nového modelu, který vstupuje do nové iterace. V případě nestabilního chovaní KNS je potřeba doplnit algoritmus i o podmínku k zastavení po n iterací.

Algoritmus 1 Proces učení

Vstup Vstupní hodnoty, Labels, Koeficient učení, Parametry sítě

Výstup Model sítě

1. Zatímco iterace je menší Maximální počet iterací:

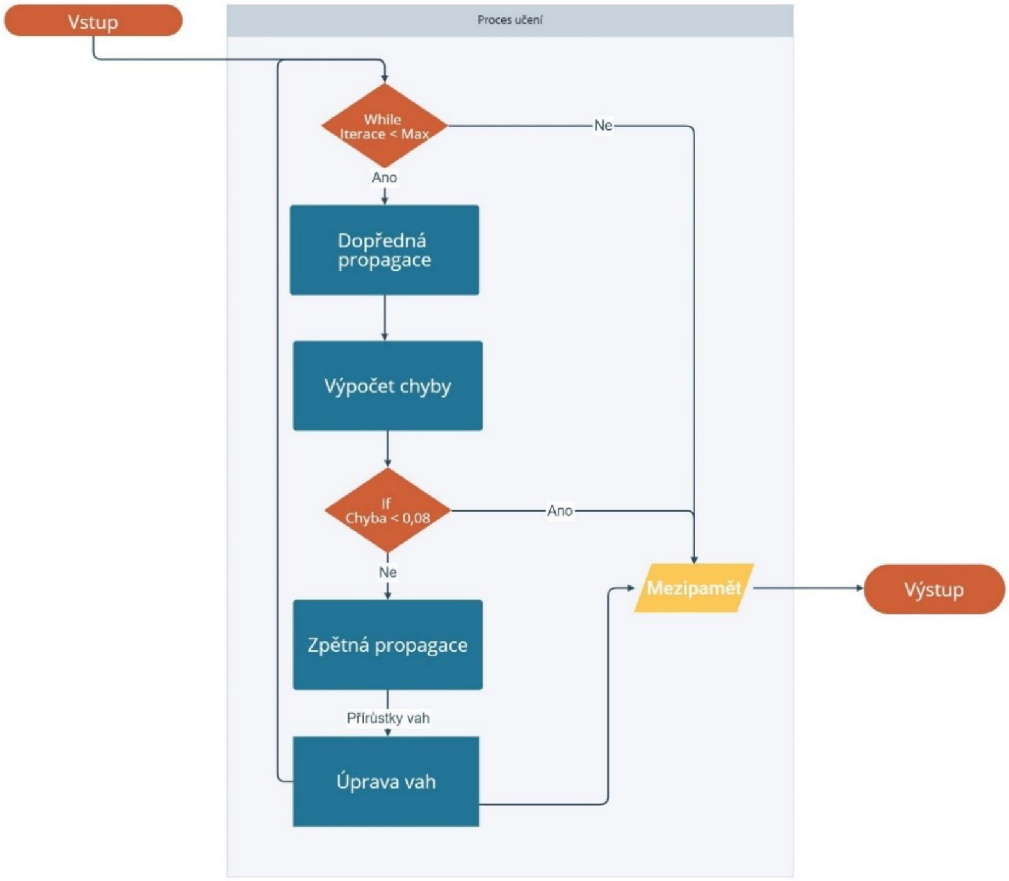
    1. Predikce, Hodnoty DP ← **Dopředná_propagace**(Parametry sítě , Vstupní hodnoty)

    2. Chyba ← Výpočet_chyby(Predikce, Labels)

    3. Pokud Chyba je menší než 0,08:
        1. Ukončení iterace

    4. Přírůstky vah ← **Zpětná_propagace**(Predikce, Labels, HodnotyDP, Koeficient učení, Chyba, Parametry sítě)

    5. Nové parametry sítě ← **Úprava_vah**(Přírůstky vah)



Obr. 6-4 Vývojový diagram – proces učení

Další část je zaměřena především na algoritmy dopředné a zpětné propagace, které jsou klíčové pro správnou funkci KNS.

Dopředná propagace vytváří predikci v závislosti na váhách a vstupních hodnotách. V první části dopředné propagace vstupují hodnoty do konvoluční části, kde se propagují konvolučními a sdružovacími vrstvami. V druhé části pak propagované hodnoty z konvoluční části vstupují do vrstev perceptronů.

Na začátku každé propagace algoritmus zjišťuje, zda není už v poslední vrstvě konvoluční části a pokud ne pokračuje dál v nové iteraci. Poté se zjišťuje, která vrstva je aktuálně na řadě pro propagaci a jaké parametry jsou zvoleny v dané vrstvě. V poslední vrstvě konvoluční části se provede propagace naposled a poté se propagovaná hodnota předává přes plně propojenou vrstvu do vrstev perceptronů.

Algoritmus 2 Dopředná propagace – Konvoluční část

Vstup Parametry sítě, Vstupní hodnoty

Výstup Výstup z konvolučních v. , Uložené hodnoty z DP

1. Zatímco Index vrstvy je menší než n:

    1. Aktuální vrstva ← Parametry sítě **[0] [Index vrstvy]**

    2. Case Aktuální vrstva == konvoluční v. :

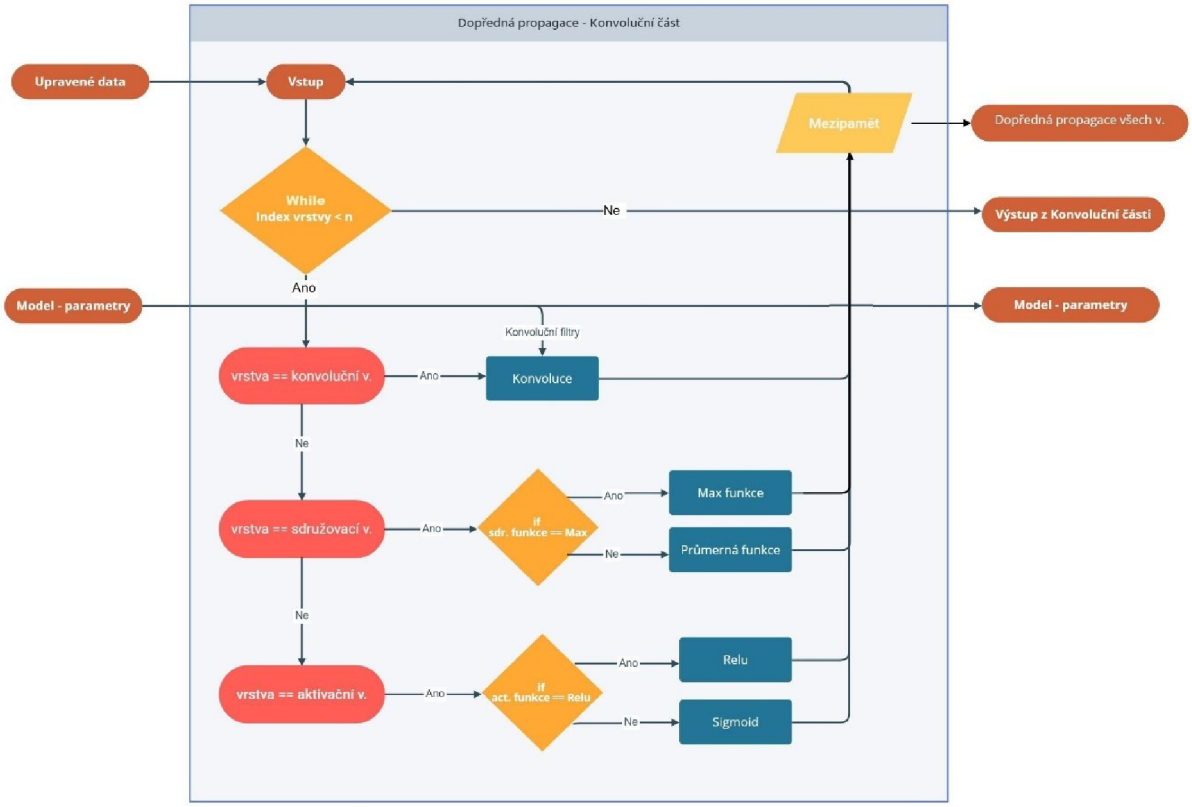
        1. Propagované hodnoty ← Propagace_konvoluční_vrstvou(Parametry sítě)

    3. Case Aktuální vrstva == aktivační v. :

        1. Propagované hodnoty ← Propagace_aktivační_vrstvou(Parametry sítě)

    4. Case Aktuální vrstva == sdružovací v. :

        1. Propagované hodnoty ← Propagace_sdružovací_vrstvou(Parametry sítě)



Obr. 6-5 Vývojový diagram – Dopředná propagace Konvoluční částí

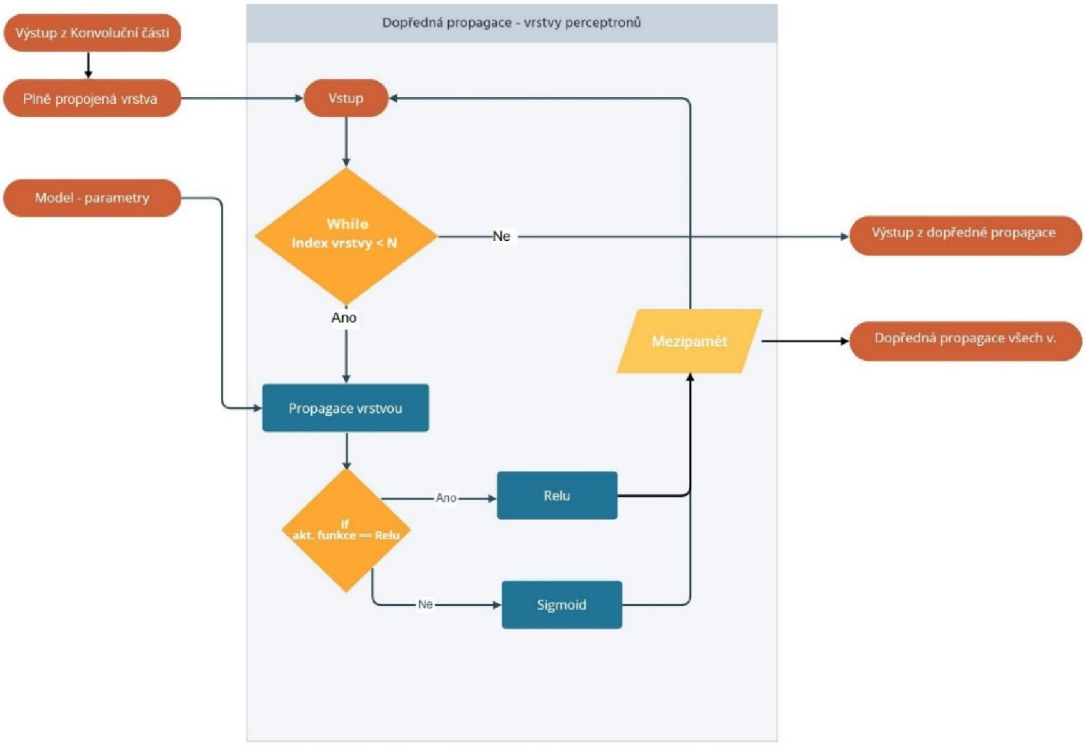
Z plně propojené vrstvy vstupují hodnoty do vrstev perceptronů, kde se v první podmínce hlídá, zda není propagovaná hodnota v poslední vrstvě. V případě, kdy podmínka neplatí, probíhá propagace vrstvou perceptronů do doby, kdy neprojdou všechny vrstvy perceptronů. Poté jako výstup z celé dopředné propagace je predikce, která vstupuje dále do ztrátové funkce.

Algoritmus 3 Dopředná propagace – vrstvy perceptronů

Vstup Parametry sítě, Vstupní hodnoty

Výstup Predikce, Uložené hodnoty z DP

1. Zatímco Index vrstvy je menší než N:
    1. W ← Parametry sítě [0][Index vrstvy]
    2. b ← Parametry sítě [1][Index vrstvy]
    3. Mód ← Parametry sítě [2][Index vrstvy]
    4. Pokud Mód == 'relu':
        1. Propagované hodnoty ← Propagace_vrstvou(W, b, 'Relu')
    5. Jinak:
        1. Propagované hodnoty ← Propagace_vrstvou(W, b, 'Sigmoid')



Obr. 6-6 Vývojový diagram – Dopředná propagace vrstvy perceptronů

U zpětné propagace postupujeme vrstvami v opačném pořadí než při dopředné propagaci. Zpětná propagace má za úkol vypočítat všechny přírůstky vah všech učících se parametrů, což jsou váhy perceptronů a váhy konvolučních filtrů. Do zpětné propagace vstupuje samotná chyba predikce a také uložené hodnoty z dopředné propagace.

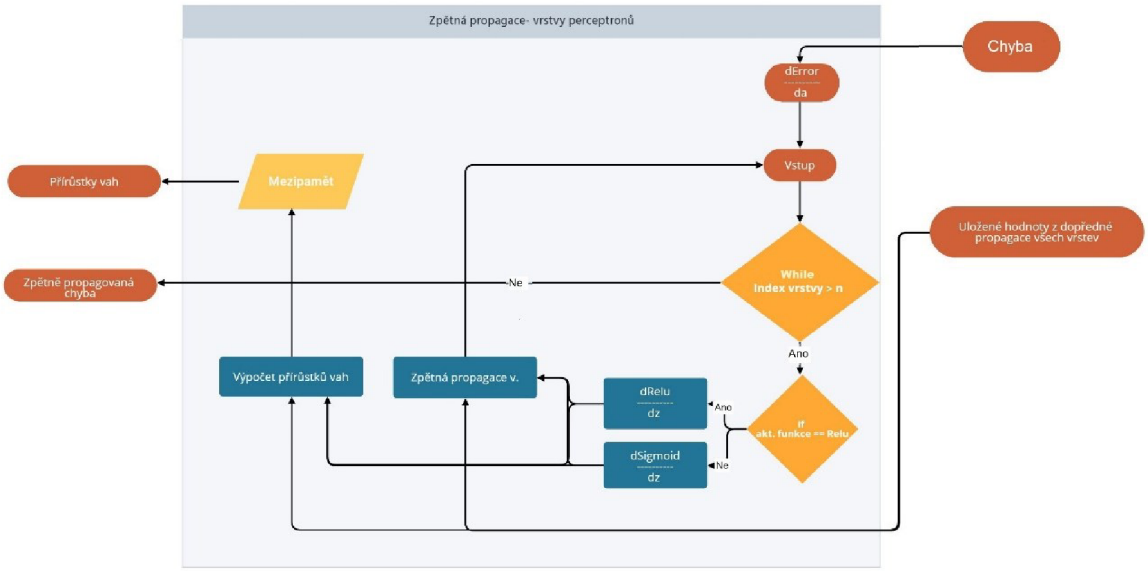
Jako první vstupuje chyba do vrstev perceptronů, kde prochází všechny vrstvy a přes plně propojenou vrstvu pokračuje dále i do konvoluční části. Pro výpočet přírůstků vah jsou potřeba i hodnoty dopředné propagace z jednotlivých vrstev, jak je naznačeno na schématech.

Algoritmus 4 Zpětná propagace – vrstvy perceptronů

Vstup Chyba, Uložené hodnoty z DP, Parametry sítě

Výstup Přírůstky vah, Zpětně propagované hodnoty

1. Propagované hodnoty ← Zpětná_propagace_ztrátovou_funkcí(Chyba)
2. Zatímco Index vrstvy je větší než n:
    1. W ← Parametry sítě [0][Index vrstvy]
    2. b ← Parametry sítě [1][Index vrstvy]
    3. Mód ← Parametry sítě [2][Index vrstvy]
    3. hodnoty z DP ← Uložené hodnoty z DP [Index vrstvy]
    4. Pokud Mód == 'relu':
        1. Propagované hodnoty, Přírůstky vah ← **ZP_vrstvou**(W, b, 'Relu', hodnoty z DP)
    5. Jinak:
        1. Propagované hodnoty, Přírůstky vah ← **ZP_vrstvou**(W, b, 'Sigmoid', hodnoty z DP)



Obr. 6-7 Vývojový diagram – Zpětná propagace vrstvy perceptronů

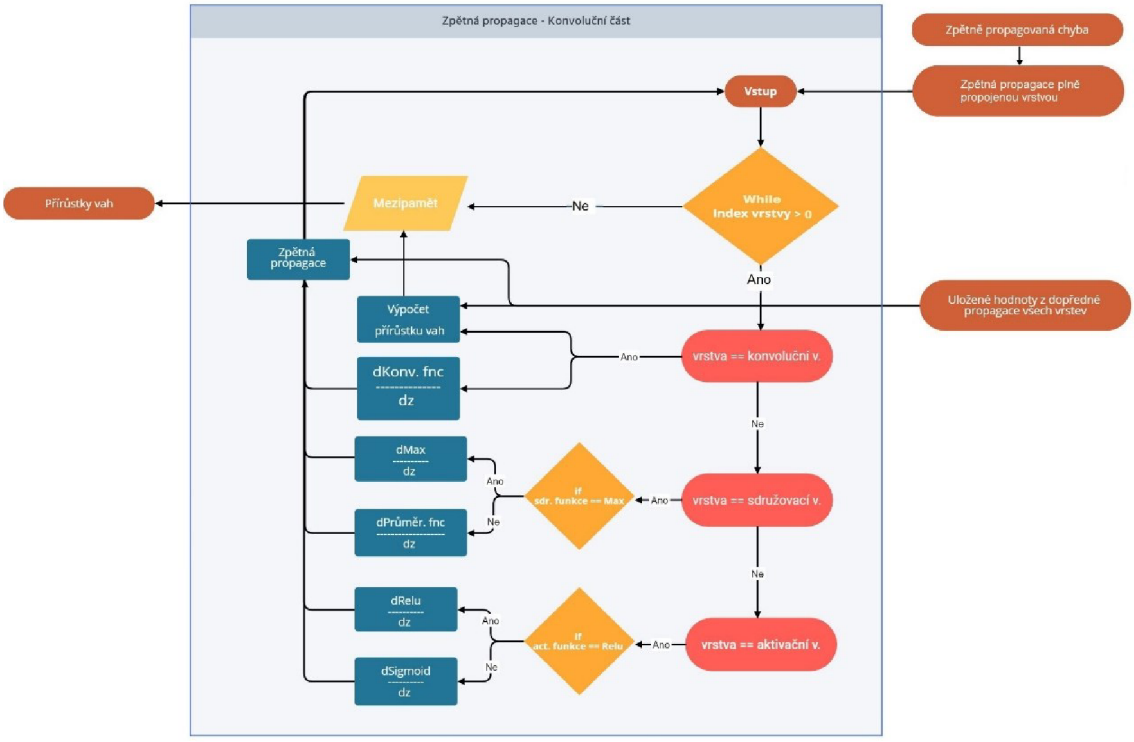
Poslední část zpětné propagace, je zpětná propagace konvoluční částí, která propaguje zpětnou chybu až do první vrstvy KNS.V každé konvoluční vrstvě vypočítává přírůstky vah konvolučních filtrů.

Algoritmus 5 Zpětná propagace – Konvoluční část

Vstup Zpětně propagované hodnoty, Uložené hodnoty z DP, Parametry sítě

Výstup Přírůstky vah

1. Propagované hodnoty ← Zpětně propagované hodnoty
2. Zatímco Index vrstvy je větší než 0:
    1. Aktuální vrstva ← Parametry sítě **[0] [Index vrstvy]**
    2. hodnoty z DP ← Uložené hodnoty z DP **[Index vrstvy]**
    3. **Case** Aktuální vrstva == konvoluční v. :
        1. Propagované hodnoty, Přírůstky vah ← **ZP_konv_v**(Parametry sítě, hodnoty z DP)
    4. **Case** Aktuální vrstva == aktivační v. :
        1. Propagované hodnoty ← **ZP_aktivační_v** (Parametry sítě, hodnoty z DP)
    5. **Case** Aktuální vrstva == sdružovací v. :
        1. Propagované hodnoty ← **ZP_sdružovací_v** (Parametry sítě, hodnoty z DP)



Obr. 6-8 Vývojový diagram – Zpětná propagace Konvoluční částí

6.2 Dataset

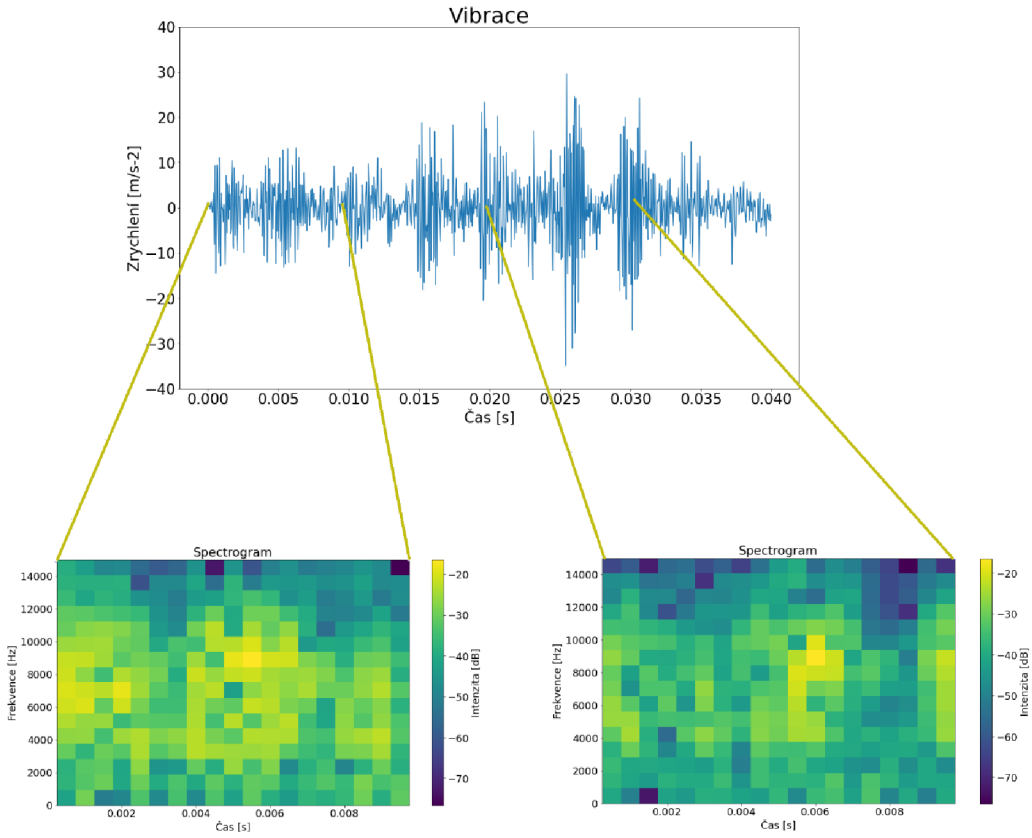
Pro testovaní sestavené Konvoluční neuronové sítě byl použit dataset vibrací snímaný z experimentálně připravené převodovky za běhu. Vibrace byly zaznamenány pomocí snímače vibrací umístěného na vnější konstrukci převodovky a otáčky výstupní hřídele byly 1800 ot/min a snímací frekvence senzoru byla 30 kHz [26]. Samotná data byla zaznamenána ve dvou stavech:

1. Převodovka bez poruchy.
2. Převodovka se zlomeným zubem ozubení.

Naměřená vibrační data, aby se dala použít pro učení KNS, bylo potřeba převést ještě i z 1D signálu na 2D signál. Tento převod se uskutečnil pomocí krátké Fourierovy transformace, která transformuje signál z časové oblasti do časově-frekvenční oblasti a vytváří tzv. spektrogramy. Spektrogramy jsou vizuální

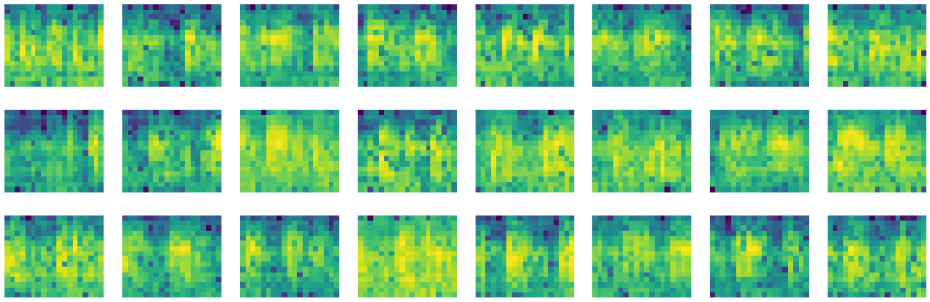
znázornění spektra frekvencí v krátkém časovém úseku. Samotný spektrogram má trojrozměrnou datovou strukturu, a pomocí barev se spektrogram vizualizuje do 2D struktur.

Převod signálu na spektrogram byl realizován po 10ms a bylo tak vytvořeno 120 vzorků spektrogramu pro každý stav převodovky, kde 100 vzorků je použito pro tréninkový dataset a 20 vzorků pro validační dataset. Časový úsek 10ms odpovídá při frekvenci otáčení převodovky 30 Hz (1800 ot/min) zhruba 1/3 otáčky.

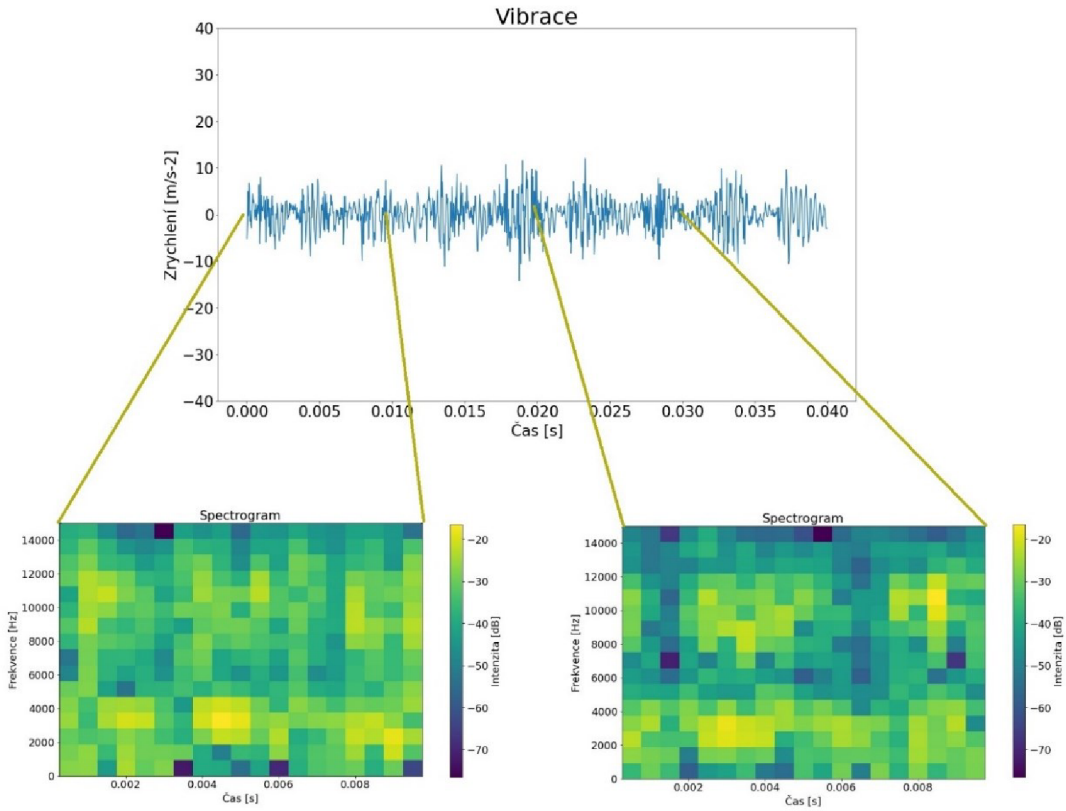


Obr. 6-9 Spektrogramy 1. stavu (převodovka bez poruchy)

Na obrázku Obr. 6-9 lze vidět dva spektrogramy transformované ze signálu vibrací prvního stavu převodovky (Stav bez poruchy). V daných spektrogramech lze podle barev jednotlivých elementů rozpoznat nejvíce zastoupené frekvence signálu v jednotlivých časových úsecích.

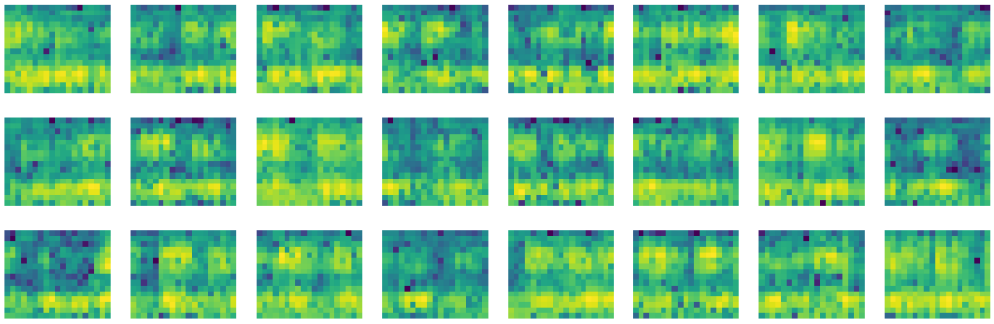


Obr. 6-10 Soubor spektrogramů 1.stavu



Obr. 6-11 Spektrogramy 2. stavu (Převodovka se zlomeným zubem)

Na obrázku Obr. 6-11 lze vidět signál druhého stavu převodovky se zlomeným zubem. Už od pohledu lze částečně rozeznat menší rozdíly mezi spektrogramy 1. a 2.stavu.



Obr. 6-12 Soubor spektrogramů 2.stavu

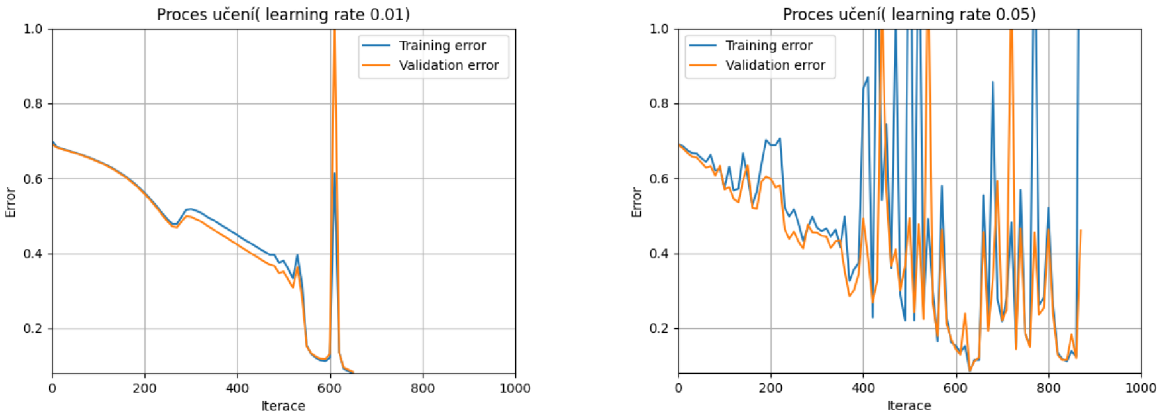
6.3 Výsledky testovaní KNS

Dataset byl testován na několika modelech KNS s různými parametry a strukturou. Pro KNS byly zvolené za aktivační funkce funkce relu a sigmoid, jako ztrátová funkce pak cross-entropy a mean squared error. Výsledky procesu učení jednotlivých KNS jsou reprezentovány grafem ztrátové funkce v závislosti na počtu iterací. Hodnoty ztrátové funkce byly zaznamenány v průběhu učení jak pro tréninkový, tak i pro validační dataset. Maximální délka učení všech NS byla stanovena na 3000 iterací s podmínkou předčasného ukončení procesu učení v případě, že ztrátová funkce klesne pod hodnotu 0,08, při této hodnotě vykazují KNS 100% úspěšnost na validačním datasetu.

Jako první byla otestována NS bez konvoluční části, tím pádem se nejedná o KNS, ale jen NS.

Tabulka 2 Stavba 1. architektury sítě

Phase 1	Vrstvy	Rozměry	Parametry	Počet učicích se p.
Část perceptronů	Perceptronová	608	Relu	608+1
	Perceptronová	201	Relu	201+1
	Perceptronová	52	Relu	52+1
	Perceptronová	1	Sigmoid	Perceptronová v.
Součet parametrů				864



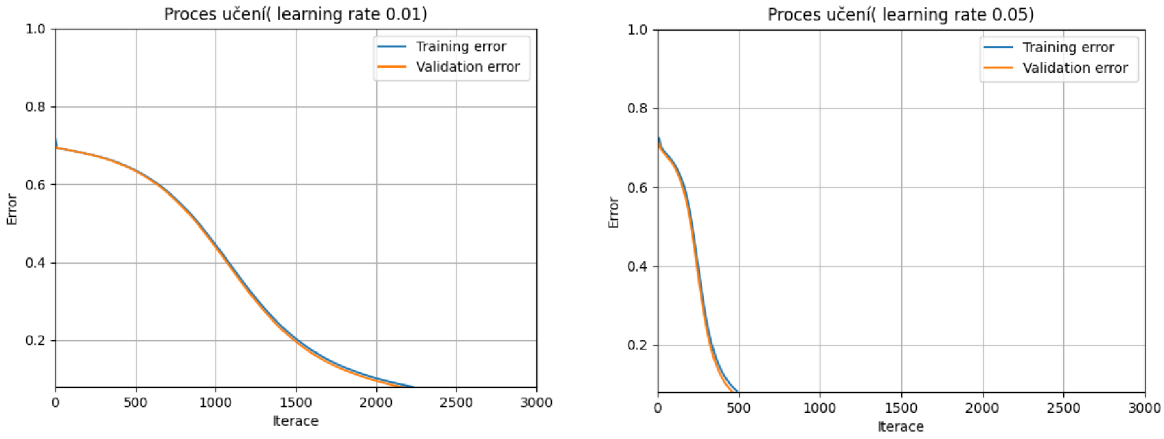
Obr. 6-13 Procesy učení 1. architektury

Z výsledku 1.architektury je patrné, že při koeficientu učení 0,05 NS vykazuje velké známky nestabilního chování, ale i přes tyto skoky nestability, dokázala NS dokonvergovat pod limitní hodnoty ztrátové funkce 0,08. V případě běhu NS s koeficientem učení 0,01 je NS mnohem stabilnější a také rychlejší.

Jako 2. architektura byla zvolena KNS s jednou konvoluční vrstvou.

Tabulka 3 Stavba 2. architektury sítě

	Vrstvy	Rozměry	Parametry	Počet učicích se p.
Část konvoluční	Konvoluční	(16,19,1)	3 filtry (3x3)	27
	Aktivační	(16,19,3)	Sigmoid	
Část perceptronů	Plně propojená	912		
	Perceptronová	912	Relu	912+1
	Perceptronová	1	Sigmoid	
Součet parametrů				940



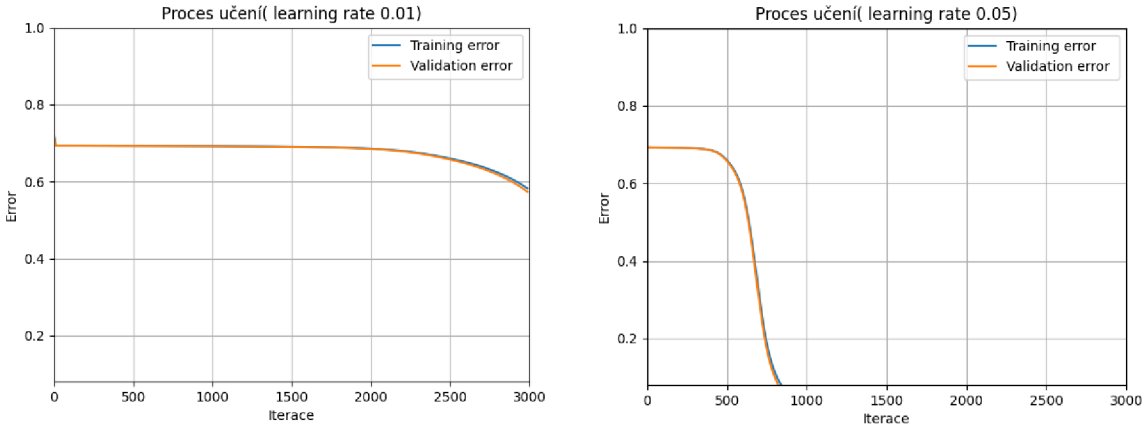
Obr. 6-14 Procesy učení 2. architektury

Ve výsledných grafech 2.architektury, lze vidět mnohem stabilnější chování. V případě procesu učení s koeficientem 0,05 je učení kratší než u 1. architektury. Z průběhu obou těchto grafů lze zpozorovat, že nejrychlejší proces učení probíhá vždy v polovině samotného procesu učení, derivace funkce je největší.

Jako poslední architektura byla otestována architektura s dvěma konvolučními vrstvami a jednou sdružovací vrstvou.

Tabulka 4 Stavba 3. architektury sítě

	Vrstvy	Rozměry	Parametry	Počet učicích se p.
Část konvoluční	Konvoluční	(16,19,1)	3 filtry (3x3)	9
	Aktivační	(16,19,1)	Sigmoid	
	Sdružovací	(16,19,1)	Max	
	Konvoluční	(15,18,1)	1 filtr (3x3)	9
	Aktivační	(15,18,1)	Sigmoid	
Část perceptronů	Plně propojená	270		
	Perceptronová	270	Relu	912+1
	Perceptronová	1	Sigmoid	
Součet parametrů				940

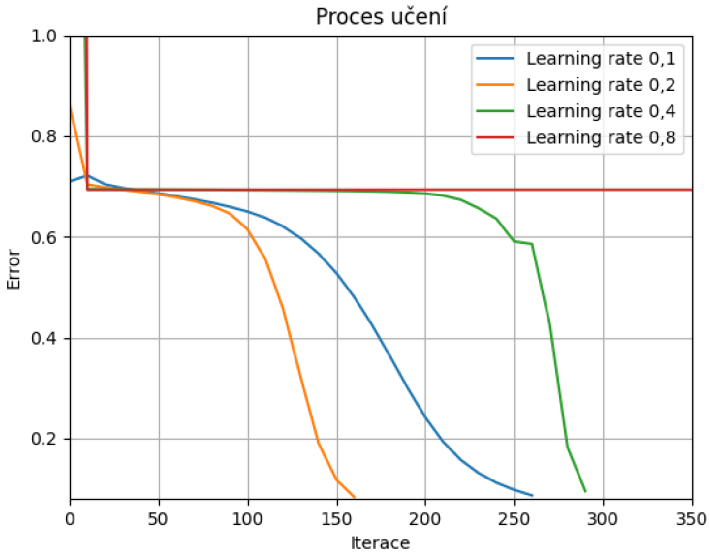


Obr. 6-15 Procesy učení 3. architektury

U grafu s koeficientem učení 0,01 lze vidět, že proces učení byl ukončen limitním počtem iterací, který byl nastaven na 3000 iterací a samotný proces učení nebyl tak ani dokončen. V době ukončení procesu učení byla hodnota ztrátové funkce ještě zhruba 0,58. V druhém případě proces učení dokonvergoval poměrně rychle, avšak oproti 2.architektuře je tato KNS zhruba o 300 iterací pomalejší.

Vliv koeficientu zrychlení

K další části testování byl vybrána pouze 2. architektura, která byla ze všech 3 nejrychlejší. Tato architektura byla testována se zvyšujícím se koeficientem učení a zjišťoval se tak vliv velikosti koeficientu na samotném procesu učení.

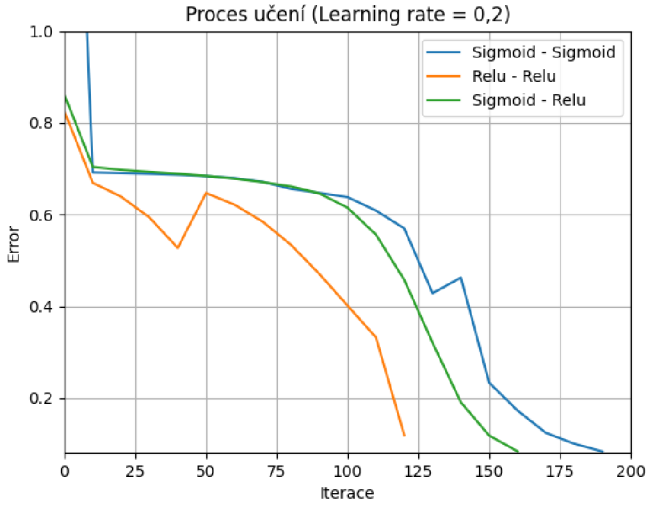


Obr. 6-16 Vliv koeficientu učení

Z výsledků lze vidět, že do hodnoty 0,2 koeficientu se proces učení zrychloval, ale po překročení této hodnoty se začaly projevovat známky nestability a proces učení pomalu zpomaloval. V případě hodnoty 0,8 dokonce hodnota chyby stagnovalo na počáteční hodnotě po celou dobu učení.

Vliv aktivační funkce

V dalším testu byl pozorován vliv aktivační funkce na procesu učení. Aktivační funkce se volí jak do konvoluční části, tak i do vrstev perceptronů, proto první funkce v legendě označuje aktivační funkci v konvoluční části a druhá zase ve vrstvách perceptronů.

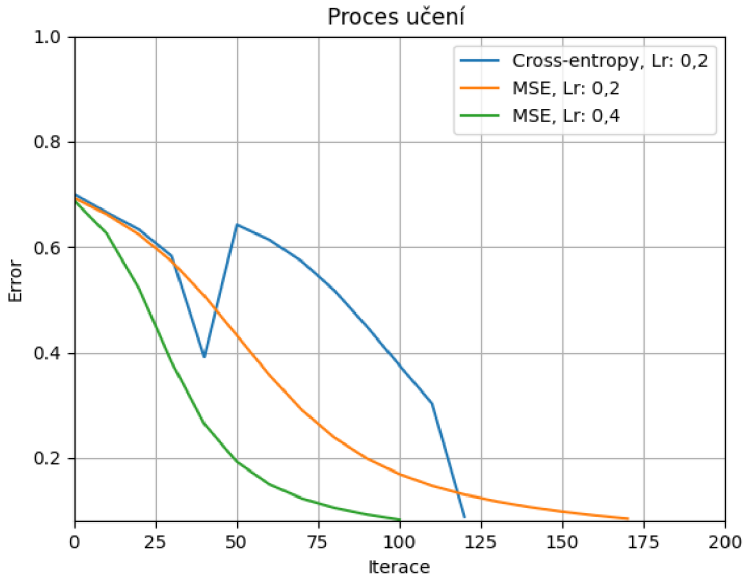


Obr. 6-17 Vliv aktivační funkce

Nejrychlejší ze všech modelů nakonec byl model pouze s aktivačními funkcemi Relu. Ten s porovnáním s modelem Sigmoid – Relu vykazuje sice větší míru nestability chovaní, ale i přesto dokonvergoval během 120 iterací pod limitní hodnotu.

Vliv ztrátové funkce

V posledním testu byla vyzkoušena také změna ztrátové funkce cross-entropy za funkci mean squared error (MSE), která je jedna z nejvíce používaných ztrátových funkcí.



Obr. 6-18 Vliv ztrátové funkce

Proces učení se ztrátovou funkcí cross-entropy vykazuje mnohem větší nestabilitu, než MSE, který dokonce s koeficientem učení 0,4 dokázal dokonvergovat během 100 iterací.

6.4 Diskuse výsledků

Z výsledků testovaní je patrně, že skoro většina modelů byla schopná dokonvergovat pod limitní hodnotu 0,08, což je hodnota při, které model vykazuje 100% úspěšnost klasifikace. Proto v případech, kdy nezáleží na rychlosti učení, může být zvolen, kterýkoliv z těchto modelů. V případech, kdy je pro aplikaci AI systému důležitý i samotný čas učení, například pro vytvoření nového modelu, je vhodnější před samotnou implementací otestovat vždy několik architektur a nastavení vnitřních parametrů sítě.

Z testování také vyplynulo, že velký vliv na samotný proces učení mají parametry neuronové sítě jako jsou volba aktivační funkce, ztrátové funkce nebo i hodnota koeficientu učení, který zásadně ovlivňuje rychlost samotného procesu učení sítě. Při příliš velké hodnotě koeficientu učení může nastat i stav nestabilní chování modelu.

ZÁVĚR

První část této diplomové práce je zaměřena na přiblížení AI technik v průmyslu a bližší popis implementace těchto technik v monitorovacích systémech. Je zde také popsáno několik příkladů aplikací AI systémů v průmyslu a jejich výhody, které přinášejí oproti běžně používaným systémům. Při popisu monitorovacích systémů jsou popsány základní kroky monitorování a metody založené na technikách strojového učení, které jsou použity na diagnostiku a prognostiku stavu stroje.

V další části se tato práce zabývá představením konvoluční neuronové sítě, optimalizací procesu učení a testovaní neuronových sítí. Jsou zde pospány především základní principy konvoluční neuronové sítě, kterými jsou dopředná propagace, výpočet chyby, následná zpětná propagace a aktualizace vah sítě. V části optimalizace procesu učení bylo vysvětleno několik souvislosti okolo procesu učení, jako je forma vstupních dat, délka učení nebo inicializace vah neuronové sítě.

V průběhu testování byly vyzkoušeny 3 architektury neuronové sítě na klasifikačním problému a poté byl ověřen vliv několika parametrů sítě, jako je počet vrstev, výběr aktivační funkce, ztrátové funkce nebo velikost koeficientu učení. Nejlepší výsledek procesu učení byl zaznamenán u konvoluční neuronové sítě s jednou konvoluční vrstvou, rele aktivačními funkcemi a ztrátovou funkcí mean squared error. Tento model dokázalo dokonvergovat za zhruba 100 iterací do limitní hodnoty chyby klasifikace 0,08.

SEZNAM ZDROJŮ

[1] *4 Intersecting Domains That You Can Easily Confuse with Artificial Intelligence, Orhan G. Yalçın* [online]. US: Towards Data Science, 2020 [cit. 2021-05-11]. Dostupné z: https://towardsdatascience.com/4-intersecting-domains-that-you-can-easily-confuse-with-artificial-intelligence-2233cb6ad7d1

[2] *A panoramic view of Machine Learning and its applications, Tariq M. Khan* [online]. 2019 [cit. 2021-05-11]. Dostupné z: https://www.researchgate.net/publication/345821742_Machine_Learning_Quantum_Vs_Classical/figures?lo=1

[3] *6 Ways Machine Learning is Revolutionizing Manufacturing in 2019* [online]. US: Rapidminer, 2019 [cit. 2021-05-11]. Dostupné z: https://rapidminer.com/blog/6-ways-machine-learning-revolutionizing-manufacturing/

[4] *Simulation, AI, Optimization and Complexity, Chris Nicholson* [online]. Pathmind, 2019 [cit. 2021-05-21]. Dostupné z: https://wiki.pathmind.com/simulation-optimization-ai

[5] *Integrating Artificial Intelligence with Simulation Modeling, Gavin Wilkinson* [online]. Anylogic, 2018 [cit. 2021-05-13]. Dostupné z: https://www.anylogic.com/blog/integrating-artificial-intelligence-with-simulation-modeling/

[6] *Begginers guide to generative design, Cat McClintoc* [online]. ptc, 2020 [cit. 2021-05-13]. Dostupné z: https://www.ptc.com/en/blogs/cad/beginner-guide-generative-design

[7] *Driving a lighter, more efficient future of automotive part design* [online]. US: Autodesk, 2020 [cit. 2021-05-11]. Dostupné z: https://www.autodesk.com/customer-stories/general-motors-generative-design

[8] *Industry 4.0: Condition monitoring use cases in detail* [online]. Bosch [cit. 2021-05-15]. Dostupné z: https://blog.bosch-si.com/industry40/industry-4-0-condition-monitoring-use-cases-in-detail/

[9] MOBLEY, R. Keith. *An introduction to predictive maintenance*. 2nd ed. New York: Butterworth-Heinemann, 2002. ISBN 07-506-7531-4.

[10] *Vibrační diagnostika* [online]. Adash, 2019 [cit. 2021-05-13]. Dostupné z: https://adash.com/cs/vibracni-diagnostika/vibracni-diagnostika/

[11] *IoT Learning Algorithms and Predictive Maintenance, Dr. Taşkın Deniz* [online]. 2019 [cit. 2021-05-13]. Dostupné z: https://medium.com/iot-and-cloud/iot-learning-algorithms-and-predictive-maintenance-2-iot-architecture-7cbf20dc468f

[12] *Condition-based Maintenance and Machine Diagnostics*. _. _: Chapman and Hall, 1992. ISBN 9780412465000.

[13] *NARX Time Series Model for Remaining Useful Life Estimation of Gas Turbine Engines*. 2016.

[14] Bearing fault diagnosis based on wavelet transform and fuzzy inference. LOU, X. a K. LOPARO. *Mechanical Systems and Signal Processing*. 2004, s. 1077–1095.

[15] MARWALA, T. *Fault Identification using Neural Networks and Vibration Data*.

[16] *Mechanical Systems and Signal Processing* [online]. [cit. 2021-03-12].

[17] *Short -time Fourier Transform vs. Wavelet Transform* [online]. USA: _, 2019 [cit. 2021-03-12]. Dostupné z: https://imgur.com/gallery/jXhI5AP

[18] *Support Vector Machine — Introduction to Machine Learning Algorithms* [online]. _: _, 2018 [cit. 2021-03-12]. Dostupné z: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[19] *Support Vector Machine — Introduction to Machine Learning Algorithms* [online]. _: _, 2018 [cit. 2021-03-12]. Dostupné z:

https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[20] *Recurrent Neural Networks* [online]. US: IBM Cloud Education, 2020 [cit. 2021-05-11]. Dostupné z: https://www.ibm.com/cloud/learn/recurrent-neural-networks

[21] *Explained: Neural networks* [online]. USA, 2017 [cit. 2021-03-08]. Dostupné z: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

[22] *A Practical Guide to ReLU, Danqing Liu* [online]. Indie, 2017 [cit. 2021-05-12]. Dostupné z: https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7

[23] *Bearing fault diagnosis based on wavelet transform and fuzzy inference*.

[24] *Deep learning - Batch Normalization* [online]. Shephexd, 2019 [cit. 2021-05-12]. Dostupné z: https://shephexd.github.io/deep%20learning/2019/01/28/Deep_learning(8)-Batch_normalization.html

[25] *Underfitting and Overfitting in Machine Learning* [online]. Geeksforgeeks, 2020 [cit. 2021-05-12]. Dostupné z: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

[26] *Gearbox Fault Diagnosis Data* [online]. Uk: OpenEI, 2019 [cit. 2021-04-11]. Dostupné z: https://openei.org/datasets/dataset/gearbox-fault-diagnosis-data

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

AI	…	Artificial intelligence (umělá inteligence)
BD	…	Big data
CNN	…	Convolutional neural network (konvoluční NS)
D	…	Dimenze
DNN	…	Deep neural network
DP	…	Deep learning
DWT	…	Discrete wavelet transform
KNS	…	Konvoluční neuronová síť
ML	…	Machine learning (Strojové učení)
MSE	…	Mean squared error
NN	…	Neural network
NS	…	Neuronová síť
RGB	…	Red green blue
RL	…	Reinforcement learning
RNN	…	Recurrent neural network (Rekurentní NS)
STFT	…	Short-time Fourier transform
SV	…	Sdružovací vrstva
SVM	…	Support vector machine
VUT	…	Vysoké učení technické v Brně
ZP	…	Zpětná propagace

Symboly:

a	…	Propagovaná hodnota aktivační funkcí
db	…	Přírůstek biasu
dw	…	Přírůstek vah
f	…	Vstupní obraz
g	…	Výstupní obraz
h	…	Konvoluční filtr
J	…	Měrná ztrátová funkce
k_{lr}	…	Koeficient učení
log	…	Přirozený logaritmus
N	…	Počet diskrétních hodnot filtru,
n_f	…	Rozměr čtvercového obrazu
n_g	…	Rozměr výstupního obrazu
n_h	…	Rozměr čtvercového filtru
p	…	predikce
p_{max}	…	Sdružovací maximální okno
p_{mean}	…	Sdružovací průměrné okno
$\mathbf{X}_{\text{new}}$	…	Maximální hodnota souboru dat
X_{min}	…	Minimální hodnota souboru dat
X	…	Aktuální hodnota souboru dat
y	…	výstup z aktivační funkce
w	…	váha
z	…	Propagovaná hodnota váhami perceptronů
Z	…	Ztrátová funkce

δ^a	…	Zpětně propagovaná hodnota váhami perceptronů
δ^z	…	Zpětně propagovaná hodnota aktivační funkcí
δ	…	Zpětně propagovaná hodnota
μ	…	Střední hodnota
σ	…	Směrodatná odchylka

SEZNAM PŘÍLOH

Příloha 1 – Zdrojový kód programu je uložen na přiloženém CD