



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY

A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

VIZUÁLNÍ DETEKCE ANOMÁLIÍ V PRŮMYSLOVÉ VÝROBĚ

VISUAL ANOMALY DETECTION IN INDUSTRIAL PRODUCTION

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. Jan Hrabica

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Karel Horák, Ph.D.

BRNO 2024

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Jan Hrabica

ID: 220986

Ročník: 2

Akademický rok: 2023/24

NÁZEV TÉMATU:

Vizuální detekce anomálií v průmyslové výrobě

POKyny PRO VYPRACOVÁNÍ:

Práce se zabývá architekturami klasifikátorů vhodných pro unární (jednotřídní) klasifikaci obrazových dat. Cílem práce je navázat na předchozí práce a navrhnout a implementovat úlohu experimentálního ověření jednotřídní klasifikace. Jde tedy o úlohu třídění vstupních vizuálních dat s anotací pouze jedné třídy. V průmyslovém prostředí úloha odpovídá stavu, kdy jsou ve výrobním provozu známy pouze OK kusy a vadné kusy nejsou implicitně ani explicitně definovány (anotací ani modelem).

1. Nastudujte oblast klasifikačních metod pro unární, binární a obecnou multi-class klasifikaci.
2. Sestavte seznam architektur vhodných pro unární klasifikaci se zevrubným popisem funkce a příklady použití.
3. Podle domluvy s vedoucím pořídte vlastní anotovaný dataset průmyslového výrobku čítající 1000+ realizací.
4. Pro vybraný mechanismus proveďte (re)implementaci, tj. sestavte/doplňte aplikaci pro ověření funkce klasifikátoru.
5. Pomocí vytvořené aplikace proveďte na pořízeném datasetu sadu experimentů s různým nastavením hyperparametrů a dělením datasetu.
6. Výsledky experimentů vyhodnoťte formou matice záměn a také z pohledu výpočetní náročnosti.

DOPORUČENÁ LITERATURA:

1. Alla S., Adari S. K. Beginning Anomaly Detection Using Python-Based Deep Learning. 2019, Apress. 416 p. ISBN 9781484251768.
2. One-class classification - Concept-learning in the absence of counter-examples. URL: <http://homepage.tudelft.nl/n9d04/thesis.pdf>

Termín zadání: 5.2.2024

Termín odevzdání: 15.5.2024

Vedoucí práce: Ing. Karel Horák, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být 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.

Abstrakt

Tato práce pojednává o problematice unárních klasifikátorů pro detekci anomálií v průmyslové výrobě. V úvodu je rozebrána klasifikace jako obecný problém, metody klasifikace a některé jejich hodnocení a následně jsou rozebrány hlavní kategorie používaných architektur. V praktické části je popsán proces tvorby scény a následné pořizování datasetu. Vytvořený dataset je použit na naučení klasifikátoru, na kterém jsou v závěru práce provedeny různé experimenty za účelem odhadu výkonnosti.

Klíčová slova

Unární klasifikátor, detekce anomálií, neuronové sítě, autoenkodér, algoritmy umělé inteligence, zpracování obrazu.

Abstract

This thesis deals with the problem of unary classifiers for anomaly detection in industrial production. It starts with a discussion of classification as a general problem, classification methods and some of their evaluations, and then discusses the main categories of architectures used. Practical part describes the process of scene creation for the acquisitions of a dataset. Acquired dataset is then used for teaching a classifier, on which is then performed a number of experiments to determine its performance.

Keywords

Unary classifier, anomaly detection, neural networks, autoencoder, artificial intelligence algorithms, image processing.

Bibliografická citace

HRABICA, Jan. *Vizuální detekce anomálií v průmyslové výrobě* [online]. Brno, 2024 [cit. 2024-05-15]. Dostupné z: <https://www.vut.cz/studenti/zav-prace/detail/159976>. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Karel Horák.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	Jan Hrabica
VUT ID studenta:	220986
Typ práce:	Diplomová práce
Akademický rok:	2023/24
Téma závěrečné práce:	Vizuální detekce anomálií v průmyslové výrobě

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

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a 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: 15. května 2024

podpis autora

Poděkování

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

V Brně dne: 15. května 2024

podpis autora

Obsah

SEZNAM OBRÁZKŮ	9
SEZNAM TABULEK.....	10
ÚVOD	11
1. KLASIFIKÁTORY	12
1.1 METODY KLASIFIKACE	12
1.1.1 Unární klasifikátor.....	13
1.1.2 Binární klasifikátor	14
1.1.3 Obecná multi-class klasifikace.....	14
1.2 METODY HODNOCENÍ KLASIFIKAČNÍCH ALGORITMŮ	14
1.2.1 Matice záměn	14
1.2.2 ROC a AUC křivka.....	16
1.3 NEURONOVÉ SÍTĚ	16
1.3.1 Aktivační funkce.....	17
1.3.2 Konvoluční neuronové sítě.....	18
1.4 UČENÍ NEURONOVÝCH SÍTÍ.....	18
1.4.1 Optimizéry pro učení.....	19
2. ARCHITEKTURY UNÁRNÍCH KLASIFIKÁTORŮ	20
2.1 HLAVNÍ KONCEPCE.....	20
2.1.1 Deep learning for feature extraction.....	20
2.1.2 Learninging feature representations of anomaly.....	21
2.2 ARCHITEKTURY GENERIC NORMALITY FEATURE LEARNING	21
2.2.1 Autoencoders.....	22
2.2.2 Generative Adversarial Networks (GANs).....	24
2.2.3 Predictability modeling.....	25
2.2.4 Self-supervised classification	26
2.3 ARCHITEKTURY ANOMALY-MEASURE DEPENDENT FEATURE LEARNING	26
2.3.1 Distance based measures	26
2.3.2 One-class classification measures	27
2.3.3 Clustering-based measures	27
2.4 END-TO-END ANOMALY SCORE LEARNING (END-TO-END UČENÍ SE SKÓRE ANIMALITY)	28
2.4.1 Prior-driven Models.....	29
2.4.2 Softmax Likelihood Models	29
2.4.3 End-to-end One-class Classification.....	29
3. TVORBA DATASETU	31
3.1 DEFINICE TŘÍD	31
3.1.1 OK kusy.....	31
3.1.2 NOK kusy	31
3.1.3 Normální třída	32
3.1.4 Anomálie	32
3.2 VÝBĚR PRŮMYSLOVÉHO VÝROBKU	32
3.3 SCÉNA	33
3.3.1 Karuselový přípravek.....	34

3.3.2	<i>Tvorba vlastní scény</i>	34
3.3.3	<i>Horní pohled</i>	39
3.3.4	<i>Boční pohled</i>	40
3.3.5	<i>Zhodnocení finální scény</i>	44
3.4	TVORBA DATASETU	46
3.4.1	<i>Proces focení</i>	46
3.4.2	<i>MATLAB script</i>	46
3.5	AUGMENTACE DAT	48
4.	IMPLEMENTACE ALGORITMU	49
4.1	VÝBĚR ALGORITMU	49
4.2	POPIS REIMPLEMENTOVANÉHO ALGORITMU	49
4.2.1	<i>Popis funkce augmentace dat</i>	49
4.2.2	<i>Popis funkce uživatelského rozhraní</i>	50
4.2.3	<i>Popis funkce hlavního programu</i>	51
5.	SADA EXPERIMENTŮ	56
5.1	RŮZNÉ NASTAVENÍ HYPERPARAMETRŮ	56
5.1.1	<i>Nastavitelné hyperparametry</i>	56
5.1.2	<i>Experimenty s hyperparametry</i>	57
5.2	DĚLENÍ DATASETŮ	57
5.3	NOK KUSY V TRÉNINKOVÉM DATASETU	57
5.4	REDUKCE DATASETU	58
5.5	ČASOVÁ NÁROČNOST ALGORITMU	58
6.	ZHODNOCENÍ EXPERIMENTŮ	59
6.1	NALEZENÍ OPTIMÁLNÍCH HYPERPARAMETRŮ	59
6.1.1	<i>Volba rozměru obrázku</i>	59
6.1.2	<i>Volba velikosti latentní dimenze</i>	61
6.1.3	<i>Volba velikosti dávky</i>	62
6.2	POMĚRY DĚLENÍ DATASETŮ	63
6.3	ROBUSTNOST VŮČI ŠPATNÉ ANOTACI	64
6.4	REDUKCE DATASETŮ	65
6.5	NÁCHYLNOST NA PŘEUČENÍ	65
	ZÁVĚR	67
	LITERATURA	69

SEZNAM OBRÁZKŮ

1.1	Příklad multi-class, binární a unární klasifikace (převzáno z [2]).....	13
1.2	Příklad matice záměn	15
1.3	Příklad ROC křivky.....	16
1.4	Ukázky průběhu jednotlivých aktivačních funkcí (převzato z [28])	17
1.5	Příklad konvoluce pomocí kernelu pro zvýraznění vertikálních hran (převzato z [28]).....	18
2.1	Hierarchická taxonomie pro hlubokou detekci anomálií (převzato z [4])	20
2.2	Schéma autoenkoderu (převzato z [6]).....	22
2.3	porovnání iregulárního a regulárního latentního prostoru (převzato a přeloženo z (8)).....	24
2.4	Příklad schématu architektury GAN (převzato [10]).....	25
3.1	Příklady OK kusů (horní řádek boční pohled, spodní řádek horní pohled, 160x160 pixelů)	31
3.2	Příklady NOK kusů (horní řádek boční pohled, spodní řádek horní pohled, 160x160 pixelů, každý sloupek představuje odlišný špendlík).....	32
3.3	Snímané připínáčky HODAN 222 (převzato z [20]).....	33
3.4	Karuselový přípravek	34
3.5	Výchozí stav pracovitě	35
3.6	Horní kamera DFK 21AU04 (pro focení byl použit objektiv computar)	36
3.7	Upravené pracovitě	36
3.8	Boční kamera DFK 41BU02.H s objektivem HLM5V50F13	37
3.9	Světlo horního přísvitů Mic 209 144.....	38
3.10	Finální verze pracoviště, výměna objektivu, přidání horního přísvitů	38
3.11	Příklady obrázků horního pohledu prvotní scény a jejich histogramy	39
3.12	Příklady obrázků horního pohledu finální scény a jejich histogramy	40
3.13	Příklady obrázků bočního pohledu prvotní scény a jejich histogramy	41
3.14	Příklady obrázků bočního pohledu upravené scény a jejich histogramy	42
3.15	Příklady obrázků bočního pohledu finální scény a jejich histogramy	43
3.16	Porovnání histogramů hodných pohledů scény.....	44
3.17	Ukázka nedostatku horního pohledu	45
3.18	Porovnání histogramů bočních pohledů scény	45
3.19	Příklady augmentovaných fotek (první sloupek jsou výchozí data).....	48
4.1	Uživatelské rozhraní s nastavenými parametry	51
4.2	Příklad vizualizace výsledků klasifikace pro různé klasifikátory.....	53
4.3	Příklad vizualizace příznakového prostoru pomocí t-SNE a PCA.	53
4.4	Příklad vizualizace výsledků klasifikace pro různé klasifikátory.....	54
4.5	Příklad vizualizace originálních, zakódovaných, dekodovaných a rozdílových dat.....	55
6.1	Ukázka specifického typu anomálie	59
6.2	Mylně zaklasifikované vzorky (CAE BAE2, Conv3, LD 64, size 128, FN vzorky nahoře FP vzorky dole)	61
6.3	Vzory v dekodovaném obraze pro velkou dávku (CAE BAE2, Conv3, LD 80, size 128, vstupní obrázky nahoře, dekodované dole).....	63
6.4	Srovnání výsledku algoritmů pro počet epoch 1 a 120 (CAE BAE2, Conv3, LD 80, size 128, velikost dávky 32	66
6.5	Průběhy trénovací a validační ztráty učení (CAE BAE2, Conv3, LD 80, size 128, počet epoch 120 velikost dávky 32).....	66

SEZNAM TABULEK

5.1	Vlastností architektur dekoderu.....	56
6.1	Výpis výsledků pro různé architektury cod/dec a velikosti vstupních obrázků (BAE2, LD32).....	59
6.2	Výpis výsledků pro různé nastavení velikosti latentní dimenze (ConvM3, BAE2, 128px)	61
6.3	Výpis výsledků pro různé nastavení velikosti dávky (ConvM3, BAE2, LD80, 128px).....	62
6.4	Výpis výsledků pro různé poměry trénovacího a validačního datasetu (ConvM3, BAE2, LD80, 128px, dávka 40).....	63
6.5	Výsledky zavedení chyby do datasetu (CAE BAE2, Conv3, LD 80, size 128,10 epoch, velikost dávky 32).....	64
6.6	Výpis výsledků pro různé nastavení velikosti vstupních obrázků (ConvM1, BAE2, LD32)	65

ÚVOD

Detekce anomálií, známá také jako detekce odlehlých hodnot nebo detekce novinek, zahrnuje identifikaci datových případů, které se výrazně liší od většiny datových bodů. Tato oblast je aktivně zkoumána již několik desetiletí, přičemž počátky zkoumání sahají až do 60. let 20. století. Vzhledem k rostoucí poptávce a aplikacím v různých oblastech, jako je řízení rizik, dodržování předpisů, bezpečnost, finanční dohled, zdravotní a lékařská rizika a bezpečnost umělé inteligence, nabývá detekce anomálií stále důležitější role. Je ústředním bodem v různých komunitách, včetně dolování dat, strojového učení, počítačového vidění a statistiky [4].

Tato oblast je zvláště důležitá ve výzkumu počítačového vidění, kde úzce souvisí s detekcí odlehlých hodnot, odstranění šumu z obrazu a detekcí anomálií v obrazech i videích. Detekci anomálií lze zařadit do oblasti klasifikace jedné třídy, jejímž cílem je konstrukce klasifikačních modelů v případě, že negativní třída buď chybí, je špatně vzorkována, nebo není jasně definována. V tomto kontextu je negativní třída považována za anomálii (tj. odlehlou hodnotu), zatímco pozitivní (nebo cílová, normální) třída je dobře charakterizována příklady v trénovacích datech [10].

Cílem práce je vysvětlit problematiku klasifikace jako celek, sestavit seznam architektur vhodných pro unární klasifikaci pro získání přehledu nad současnými trendy, sestavit dataset pro naučení vybraného algoritmu, reimplementovat tento algoritmus a provést sadu experimentů s tímto mechanismem s různým nastavením hyperparametrů a dělením datasetu. V závěru práce jsou výsledky experimentů zhodnoceny pomocí různých metrik vycházejících z matice záměn jako precision, recall a F1 a pohledu na výpočetní náročnost.

1. KLASIFIKÁTORY

Existují různé typy technik strojového učení, tato práce se zabývá jednou z nejznámějších a nejpoužívanějších: klasifikaci. Otázky, kterými se bude zabývat, jsou však použitelné pro všechny techniky strojového učení. Klasifikátor je systém, který přijímá vektor diskrétních a/nebo spojitých hodnot příznaků a poskytuje jediný diskrétní výstup – třídu. Například spamový filtr kategorizuje e-mailové zprávy jako "spam" nebo "není spam", přičemž jeho vstup je reprezentován logickým vektorem, například $x = (x_1, \dots, x_j, \dots, x_d)$ kde $x_j = 1$, pokud se v e-mailu vyskytuje j-té slovo ze slovníku, a $x_j = 0$ v opačném případě. Učící se subjekt vezme trénovací množinu příkladů (x_i, y_i) , kde $x_i = (x_{i,1}, \dots, x_{i,d})$ je pozorovaný vstup a y_i je odpovídající výstup, a vytvoří klasifikátor. Učící se subjekt je hodnocen na základě toho, zda tento klasifikátor přesně předpovídá výstup y_t pro budoucí příklady x_t . Například se hodnotí, zda spamový filtr správně klasifikuje dříve neviděné e-mailové zprávy jako spam, nebo ne.

Počáteční výzvou pro aplikaci strojového učení je výběr z obrovského množství učících se algoritmů. V současné době jsou jich k dispozici tisíce a každý rok přibývají stovky dalších. Pro orientaci v tomto rozsáhlém prostředí je nezbytné si uvědomit, že se v podstatě skládá z kombinací tří klíčových složek. Těmito složkami jsou: reprezentace, vyhodnocení a optimalizace

Reprezentace. Klasifikátor musí být vyjádřen ve formálním jazyce, který může počítač zpracovat. Další související otázkou, je způsob reprezentace vstupu, v podstatě určení, které příznaky využít.

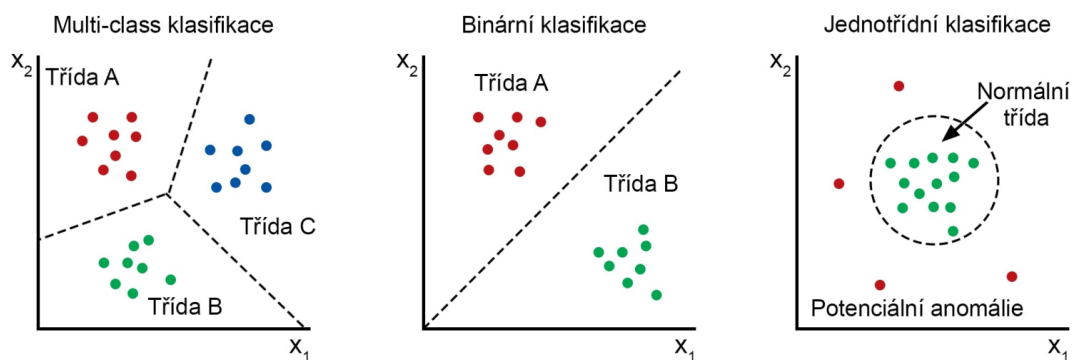
Vyhodnocení. Funkce pro hodnocení neboli odhad přesnosti modelu (označovaná jako objektivní funkce nebo skórovací funkce) je nezbytná k rozlišení efektivních klasifikátorů od neefektivních. Vnitřní hodnotící funkce použitá algoritmem se může lišit od vnější funkce, kterou chceme, aby klasifikátor zlepšil

Optimalizace. Nakonec je zapotřebí technika, která prozkoumá klasifikátory a určí ten s nejvyšším skóre. Výběr optimalizační metody je klíčový pro efektivitu učení a hraje roli při hodnocení vytvořeného klasifikátoru, zejména pokud má vyhodnocovací funkce více optim. Zpočátku nové učící se subjekty často využívají hotové optimalizátory, které mohou být později nahrazeny ne míru navrženými optimalizátory.
[1]

1.1 Metody klasifikace

V disciplíně strojového učení definujeme tři základní typy klasifikačních úloh: unární klasifikace, binární klasifikace a vícetřídní klasifikace.

Výběr správného typu klasifikační úlohy je klíčový pro úspěšné řešení dané úlohy, výběr nesprávného klasifikátoru může vést k vytvoření neobjektivního klasifikátoru z důvodu nesprávnosti předpokladů vlastnosti a povahy úlohy.



Obrázek 1.1 Příklad multi-class, binární a unární klasifikace (převzáno z [2])

1.1.1 Unární klasifikátor

Konvenční přístup ke klasifikaci více tříd je navržen tak, aby neznámý datový objekt zařadil do jedné z několika předem určených skupin, v případě binární klasifikace obvykle do dvou. Problémy nastávají, když neznámý datový objekt nezapadá do žádné z těchto předem definovaných kategorií. Například, když trénovací soubor dat obsahuje příklady ovoce a zeleniny. Použití binárního klasifikátoru je proveditelné při klasifikaci neznámého testovacího objektu v rámci domény ovoce a zeleniny (např. jablko nebo brambora). Pokud však testovaný datový objekt patří do zcela jiné domény, např. kočka z kategorie zvířat, klasifikátor bude kočku chybně klasifikovat buď jako ovoce, nebo jako zeleninu a v obou případech poskytne nesprávné výsledky. Existují případy, kdy cílem klasifikace není přiřadit testovaný objekt do předem definovaných kategorií, ale spíše určit, zda patří do určité třídy, či nikoli.

V kontextu unárních klasifikátorů je jedna z tříd, libovolně označená jako pozitivní nebo cílová třída, dobře definována případy v trénovacích datech. Naopak druhá třída, označená jako negativní nebo odlehlá, buď nemá instance, má jich velmi málo, nebo tyto instance nepředstavují statisticky reprezentativní vzorek negativního konceptu. Pro ilustraci významu jednotřídní klasifikace uvažujme několik scénářů. Například při monitorování poruch strojů může nastat situace, kdy je cílem odhalit abnormální nebo chybné chování stroje. Shromáždění dat o běžném provozu stroje (tréninková data pozitivní třídy) je jednoduché. Získání dat o potenciálních poruchách je však náročné, protože se ve skutečnosti nemusely vyskytnout nebo jsou vzácné, což má za následek minimální nebo žádná trénovací data pro negativní třídu. Navíc čekání na výskyt takových poruch může být nežádoucí z důvodu vysokých nákladů, poruchy stroje nebo potenciálního rizika pro lidskou obsluhu.

Dalším příkladem je automatická diagnóza nemoci. Generování pozitivních dat, která obsahují informace o všech pacientech s „běžnou“ nemocí, je poměrně jednoduché. Získání negativních údajů však představuje výzvu. Pacienty, kteří nebyli dříve testováni, nelze považovat za negativní případy a provedení takových testů může

být nákladné. V případech, kdy je onemocnění považováno za „vzácné“, je shromažďování pozitivních vzorků obtížné, dokud není postižena dostatečně velká skupina, což je nepraktický přístup.[3]

1.1.2 Binární klasifikátor

Binární klasifikace je úloha strojového učení s učitelem, která zahrnuje kategorizaci dat do dvou exkluzivních skupin nebo kategorií. Tyto skupiny lze označit jako 0 a 1, pozitivní a negativní nebo pravdivé a nepravdivé. Modely pro binární klasifikaci se trénují na anotovaných dateštech, kde má každý datový bod přiřazený výsledek. Tímto se model naučí kategorizovat datové body. Aplikace binární klasifikace zahrnují různé oblasti, včetně detekce spamu, podvodů a lékařské diagnostiky. Binární klasifikátor může například rozpoznat, zda je e-mail spam, či nikoli, a to na základě rozpoznání konkrétních klíčových slov a vzorů spojených se spammem. Po naučení může model klasifikovat nové e-maily jako spam nebo ne [5]. Dalším příkladem je klasifikace obrázků jako pes nebo kočka, jak je znázorněno na diagramu neuronové sítě níže.

1.1.3 Obecná multi-class klasifikace

V případě multiclass klasifikace (klasifikace do více tříd) algoritmus vypočítá pravděpodobnost příslušnosti konkrétní jednotky ke každé možné třídě a následovně se použije klasifikační pravidlo k přiřazení třídy. Obvykle je toto pravidlo velmi jednoduché, přičemž nejběžnější přístup přiřazuje jednotku do třídy s nejvyšší pravděpodobností.

Klasifikační model poskytuje pravděpodobnost příslušnosti ke konkrétní třídě pro každou jednotku. Ve scénářích s více třídami existují různé možnosti, přičemž mezi nejčastěji používané techniky patří hodnota nejvyšší pravděpodobnosti a softmax [17].

Na rozdíl od binární nebo unární klasifikace je zde možnost kategorizovat do podrobnějších tříd. Jednoduchý příklad tříd v kontextu průmyslové výroby je klasifikace poškození výrobku jako jsou: vada povrchové úpravy, koroze, poškozený závit, chybějící díra. A z toho vyplývající rozhodnutí, co s dílem udělat: na opravu, doobrobit, šrotovat.

1.2 Metody hodnocení klasifikačních algoritmů

Pro hodnocení klasifikátorů existují různé metody, v této práci budou využity běžné metriky jako matice záměn a ROC křivka.

1.2.1 Matice záměn

Anglicky confusion matrix také nazývána jako nákladová matice slouží jako hodnocení kvality klasifikátoru zařazením výsledků do 4 kategorií. V příkladu hodnocení binárních klasifikátorů představují kategorie veškeré výsledky klasifikace. Data mohou buď do třídy patřit nebo nikoliv (ground truth) a klasifikátor je může vyhodnotit jako patřící či

nepatřící a s těmito čtyřmi možnostmi matice záměn pracuje.

Tyto čtyři možnosti jsou:

- TP – true positive (skutečně pozitivní), data jsou v tréninkové množině pozitivní a klasifikátor rozhodl že jsou pozitivní.
- TN – true negative (skutečně negativní), data jsou v tréninkové množině negativní a klasifikátor rozhodl že jsou negativní.
- FP – false positive (falešně pozitivní, také nazývána chyba prvního typu), data jsou v tréninkové množině negativní a klasifikátor rozhodl že jsou pozitivní.
- FN – false negative (falešně negativní, také nazývána chyba druhého typu), data jsou v tréninkové množině pozitivní a klasifikátor rozhodl že jsou negativní.

		Klasifikátor	
		Pozitivní	Negativní
Dataset	Pozitivní	TP	FN
	Negativní	FP	TN

Obrázek 1.2 Příklad matice záměn

Z těchto čtyř případů se dále počítá řada metrik, mezi které patří:

- Precision – přesnost, počet skutečně pozitivních vůči všem pozitivním dle klasifikátoru

$$Precision = \frac{TP}{(TP + FN)}$$

- TPR – True Positive Rate (Recall, senzitivita), počet skutečně pozitivních vůči všem pozitivním dle datasetu

$$TPR = \frac{TP}{(TP + FN)}$$

- TNR – True Negative Rate (specificita), počet skutečně negativních vůči všem negativním dle datasetu

$$TNR = \frac{TN}{(FP + TN)}$$

- FPR – False Positive Rate, počet falešně pozitivních vůči všem negativním dle datasetu

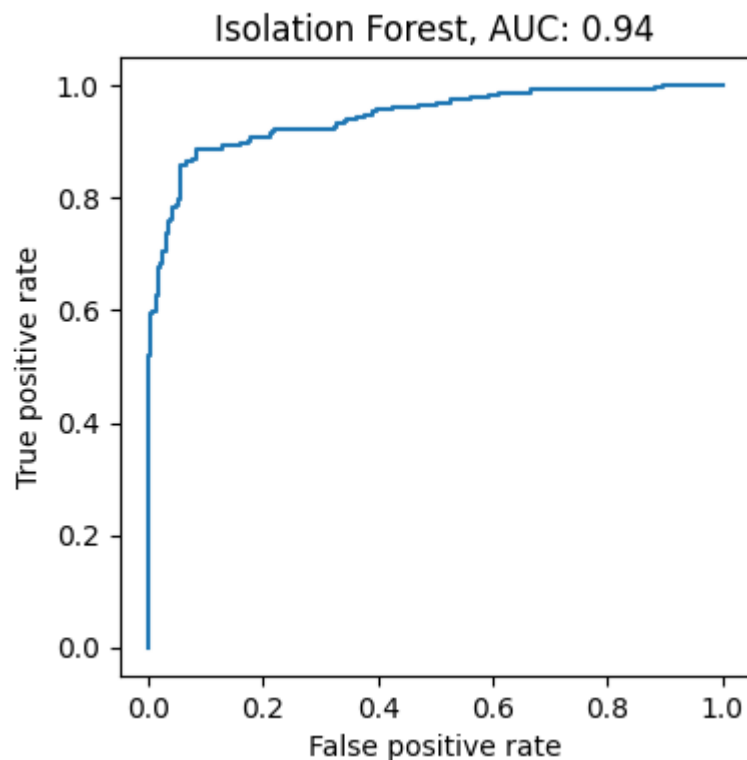
$$FPR = \frac{FP}{(FP + TN)}$$

- F_1 skóre – harmonický průměr Precision a TPR (0-1) hodnotí poměr mezi Precision a TPR

$$F_1 = 2 \frac{\text{Precision} \cdot \text{TPR}}{(\text{Precision} + \text{TPR})}$$

1.2.2 ROC a AUC křivka

ROC křivka je grafické porovnání vycházející z matice záměn. Na ose x je FPR a na ose y TPR, každý bod na této křivce odpovídá jednomu nastavení klasifikátoru. V grafu nadále platí že čím blíže je tento bod nastavení klasifikátoru k levému hornímu rohu tedy minimálnímu FPR a maximálnímu TPR [14].



Obrázek 1.3 Příklad ROC křivky

Další metrikou vycházející z ROC křivky je AUC (Area Under ROC) tedy plocha pod ROC křivkou popisuje kvalitu všech možných klasifikátorů, které lze vytvořit. Tuto hodnotu lze také nazvat predikční potenciál klasifikátoru. Hodnotí tedy klasifikátor jako celek.

1.3 Neuronové sítě

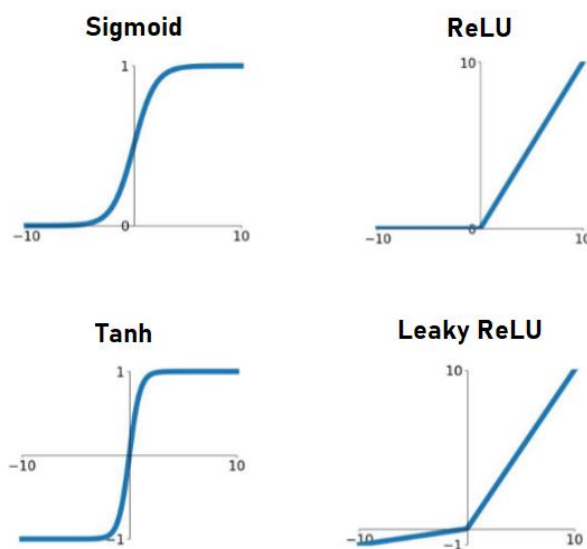
Ve strojovém učení se velmi rozšířeně používá koncept neuronových sítí (NN), tedy algoritmů, které se podobají na funkci mozku. Sítě mohou být jedno nebo vícevrstvé, v takovém případě se první vrstva označuje jako vstupní, následují vnitřní (skryté)

vrstvy následované výstupní vrstvou. Základní stavební jednotkou neuronové sítě je samotný neuron, ten má řadu parametrů, mezi které patří počet vstupních proměnných, váhy jednotlivých vstupních proměnných, aktivační funkce neuronu a výstupní hodnota. Počet vstupních proměnných záleží buď na velikosti vstupních dat, které má síť zpracovávat, pokud se jedná o vstupní vrstvu nebo na počtu neuronů v předešlé vrstvě, různé vrstvy mohou mít různé počty jednotlivých neuronů, a ne všechny musí být plně propojeny.

Výhoda vícevrstvých neuronových sítí spočívá ve velkém množství nastavitelných parametrů, jelikož každá váha každého neuronu může být nastavena. Díky této vlastnosti můžou být algoritmy tohoto typu použity na nepřeberné množství aplikací. Příkladem neuronové sítě může být algoritmus na rozpoznání psaných číslic nebo písmen. Kde do sítě vstupuje obrázek nějakého rozměru a výsledkem může být množina desetinných hodnot jistot s jakou síť rozpoznala danou číslici.

1.3.1 Aktivační funkce

Existuje velká řada aktivačních funkcí v následující podkapitole jsou rozebrány aktivační funkce využívané v praktické části. Popis funkcí je převzat z dokumentace knihovny Keras [29] a [28]



Obrázek 1.4 Ukázky průběhu jednotlivých aktivačních funkcí (převzato z [28])

Sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$ spíše zastaralá funkce, její nevýhody jsou že není centrována kolem 0, její vlastnost saturace ruší gradienty a je náročná na výpočet.

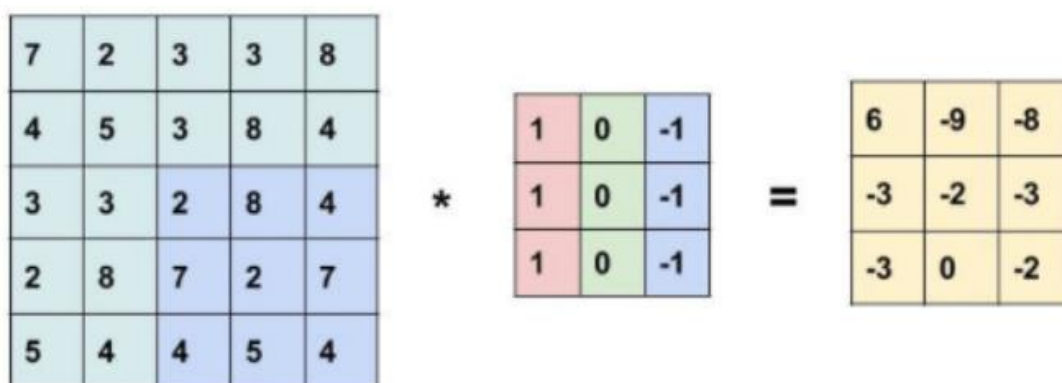
Tanh: $f(x) = \frac{e^{2x}-1}{e^{2x}+1}$ oproti funkci sigmoid je Tanh centrována kolem nuly ale stále zůstávají stejná omezení jako saturace výstupu a výpočetní náročnost kvůli exponenciále.

ReLU: $f(x) = \max(0, x)$ velmi rozšířená funkce, nesaturuje výstup, je jednoduchá na derivaci ale „zabíjí neurony“, jelikož nijak neváhuje záporné hodnoty a tím nezpůsobí žádnou změnu během trénování

Leaky ReLU modifikace ReLU která řeší problém „zabíjení neuronů“

1.3.2 Konvoluční neuronové sítě

V konvolučních neuronových sítích (CNN) jsou jednotlivé vrstvy obvykle uspořádány do posloupnosti a vstupní data postupují od vstupní k výstupní vrstvě. Propojení mezi jednotlivými vrstvami je oproti NN realizováno různými matematickými operacemi jako je na příklad konvoluce nebo poolování (redukce rozměru). Konvoluční vrstvy CNN obsahují kernely neboli matice, skládající se z vah, kernel se postupně posunuje v obraze a na každé pozici dojde k vynásobení jednotlivých vah s pixely vstupního obrázku a následnému sečtení těchto hodnot, výsledná hodnota tvoří jeden pixel do další vrstvy sítě. Výhodou je že existuje v podstatě nekonečné množství různých kernelů, se kterými můžeme vstupní obraz kondolovat, mezi které patří například hledání hran, hledání textur nebo detekce významných bodů.



Obrázek 1.5 Příklad konvoluce pomocí kernelu pro zvýraznění vertikálních hran (převzato z [28])

1.4 Učení neuronových sítí

Učení neuronových vah probíhá obecně následovně: na začátku algoritmu se inicializují váhy celé sítě, na vstup se potom přivedou data z tréninkové množiny a nechají se sítě proběhnout, z výstupní vrstvy sítě se získají výstupy, které jsou porovnány s požadovanými hodnotami a na základě toho se vypočítá chyba, chyba se následně šíří sítě zpět ke vstupu a vypočítávají se gradienty chyby. Na základě těchto gradientů a učicího koeficientu, po případě optimizéru jsou následně adaptovány váhy. Celý proces se opakuje, dokud není dosaženo požadované úrovně chyby nebo určitého kritéria.

1.4.1 Optimizéry pro učení

Optimizéry slouží jako algoritmy používané při učení pro nalezení optimálních parametrů sítě, tak, aby vznikala co možná nejmenší chyba. Mezi hlavní funkce patří aktualizace vah, nalezení minima chybové funkce, volba kroku učení tento paramter udává jak moc se mají váhy v závislosti na chybě měnit nebo případné upravování kroku učení v průběhu tréninku.

V následující podkapitole jsou rozepsány vlastnosti různých optimizérů použitelných pro strojové učení.

Adaptive Moment Estimation (Adam): jedná se o nejvíce používaný optimalizační algoritmus, optimizér používá průměr druhých momentů gradientů a výkonově se jedná o kompromis mezi rychlostí a generalizací.

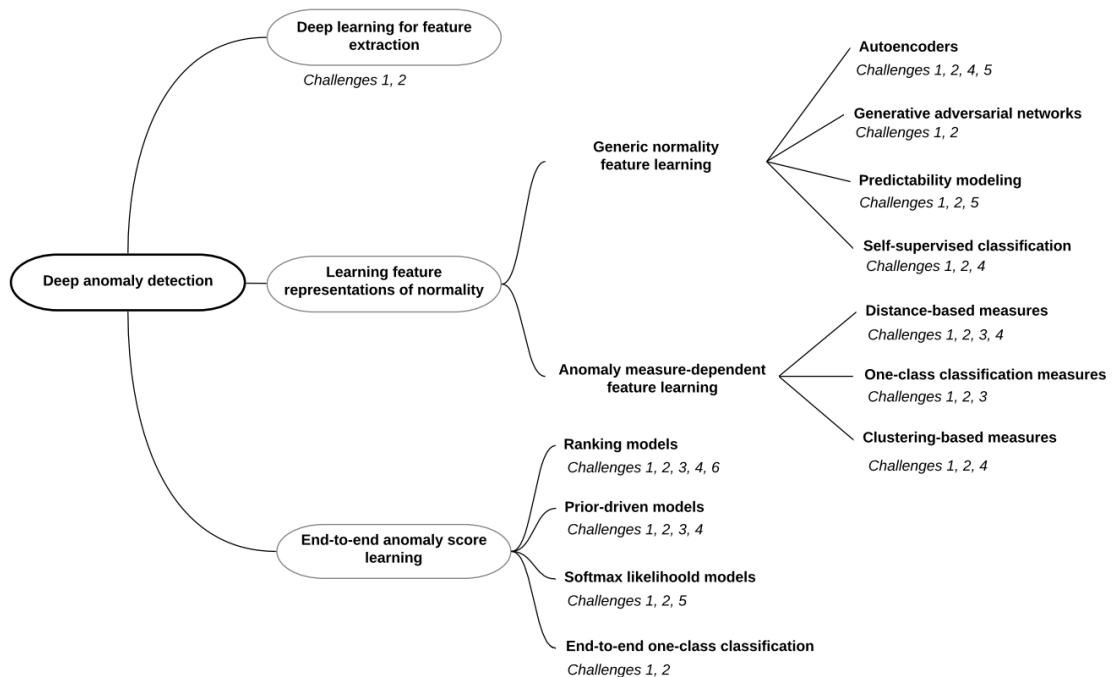
Stochastic gradient descent (SGD): jedná se o variantu gradientního sestupu, která se zaměřuje na častější aktualizaci parametrů sítě. Mezi hlavní výhody patří rychlejší konvergence díky častějším aktualizacím a nižší paměťové nároky, což umožňuje efektivnější výpočty, ale metoda je náchylná na vysoké variabilitě parametrů modelu.

Root Mean Square Propagation (RMS-Prop): princip spočívá v použití exponenciálně klesajícího průměru gradientů. Pro výpočet zahrnuje pouze nejnovější gradienty, což umožňuje modelu flexibilně adaptovat koeficient učení. Nevýhodou je že učení je relativně pomalé.

2. ARCHITEKTURY UNÁRNÍCH KLASIFIKÁTORŮ

Hluboké neuronové sítě použité pro detekci anomálií lze rozdělit do 3 hlavních kategorií a 11 podrobných kategorií z perspektivy modelování. Konkrétně se hluboké detekce anomálií skládají ze tří koncepčních paradigmat [4].

- Deep learning for feature extraction (hluboké učení pro získání příznaků)
- Learning feature representations of anomaly (učení příznaků reprezentující normální třídu)
- End-to-end anomaly score learning (end-to-end učení se skóre anomaly)



Obrázek 2.1 Hierarchická taxonomie pro hlubokou detekci anomálií (převzato z [4])

2.1 Hlavní koncepce

2.1.1 Deep learning for feature extraction

V překladu hluboké učení pro získání příznaků, tato kategorie metod se zaměřuje na využití hlubokého učení k odvození nízko rozměrných reprezentací příznaků z dat, která jsou vysoko rozměrná a/nebo nejsou lineárně oddělitelná. Extrakce příznaků a skórování anomálií jsou v této kategorii zcela oddělené a fungují nezávisle na sobě. Komponenty hlubokého učení proto slouží výhradně k redukci dimenzionality.

V porovnání s jinými populárními metodami pro redukci dimenzionality v oblasti detekci anomálií například principal component analysis (PCA) (analýza hlavních

komponent) a random projection (náhodná projekce) ukázaly metody hlubokého učení podstatně lepší schopnost v extrakci sémanticky bohatých příznaků a vztahy mezi nelineárními příznaky. Jeden z přístupů zahrnuje přímé využití známých předtrénovaných modelů hlubokého učení, jako jsou AlexNet, VGG a ResNet, pro extrakci nízkorozměrných příznaků [4].

Výhody: k dispozici je velké množství pokročilých, předtrénovaných hlubokých modelů a snadno dostupných hotových detektorů anomálií. Hluboká extrakce příznaků poskytuje robustnější redukci dimenzionality ve srovnání s široce používanými lineárními metodami.

Nevýhody: oddělenost extrakce příznaků a skórování anomálií obvykle vede k suboptimálnímu skórování anomálií. Předtrénované hluboké modely jsou typicky limitovány na specifické typy dat.

Tato kategorie metod zahrnuje mapování vysokorozměrných nebo nesamostatných dat na podstatně méně rozměrný prostor, což umožňuje použití stávajících metod detekce anomálií na jednodušší datové prostory. Reprezentace s nižšími rozměry často pomáhá odhalit skryté anomálie a minimalizovat falešně pozitivní výsledky. Je však důležité poznamenat, že tyto metody nemusí zachovat dostatečné informace pro detekci anomálií, protože projekce dat je zcela nezávislá na procesu detekce anomálií. Kromě toho tento přístup umožňuje využití různých typů příznaků a vývoj sémanticky bohatých detekčních modelů což přispívá ke snížení počtu falešně pozitivních výsledků [4].

2.1.2 Learninging feature representations of anomaly

V překladu učení příznaků reprezentující normální třídu, v této kategorii metody do určité míry integrují učení příznaků se skórováním anomálií, čímž se odchyľují od úplného oddělení těchto dvou modulů, jak bylo uvedeno v předchozí kategorii. Tyto metody obvykle patří do dvou hlavních skupin: generic normality feature learning (obecné učení příznaků) a anomaly-measure dependent feature learning (učení příznaků závislé na míře anomálie) [4].

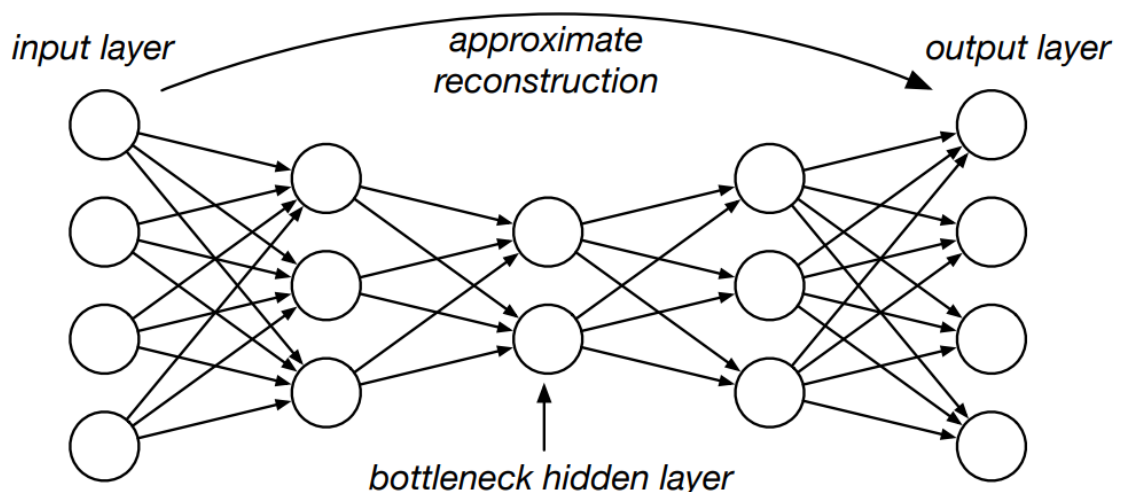
2.2 Architektury generic normality feature learning

V překladu obecné učení příznaků, metody v této kategorii získávají reprezentace datových instancí optimalizací obecné účelové funkce učení příznaků. Tato funkce není specificky přizpůsobena pro detekci anomálií, ale získané reprezentace mohou zlepšit schopnosti detekce anomálií, protože jsou nuceny zachytit základní zákonitosti dat. Tento přístup zahrnuje metody řízené více perspektivami, jako je rekonstrukce dat, generativní modelování, modelování předvídatelnosti a samokontrolovaná klasifikace. Modelování předvídatelnosti i klasifikace s vlastním dohledem vycházejí z metodik učení s vlastním dohledem, ale pracují s odlišnými předpoklady a mají své výhody a nevýhody [4].

Konkrétní architektury, které budou rozebrány dále v této kapitole jsou: Autoencoders (Autoenkodéry), Generative Adversarial Networks (GANs) (Generativní adverzní sítě), Predictability Modeling (Modelování předvídatelnosti), Self-supervised Classification (Klasifikace s vlastním dohledem)

2.2.1 Autoencoders

Autoenkodéry jsou specializovaná forma vícevrstvé neuronové sítě určená k hierarchickému a nelineárnímu snižování dimenzionality dat. Snižování dimenzionality lze interpretovat jako kompresi dat, kdy kodér komprimuje data (z výchozího prostoru do zakódovaného prostoru, nazývaného také latentní prostor) [8]. Výstupní vrstva má obvykle stejný počet uzlů jako vstupní vrstva a architektura má vrstevnatou a symetrickou strukturu. Hlavním cílem autoenkodéru je natrénovat výstupní vrstvu tak, aby z latentního prostoru co nejpřesněji rekonstruovala vstupní data. Uzly ve středních vrstvách jsou méně početné, a proto jediným způsobem, jak rekonstruovat vstup, je nastavit váhy sítě tak, aby mezilehlé výstupy uzlů ve středních vrstvách představovaly zmenšené reprezentace. Obrázek 1 znázorňuje plně propojený autoenkodér, jehož výstupy z úzké vrstvy slouží jako redukovávaná reprezentace.[6]



Obrázek 2.2 Schéma autoenkodéru (převzato z [6])

Protože autoenkodér vytváří zhuštěnou reprezentaci dat, je přirozenou metodou pro detekci odlehlých hodnot. Základní koncepce spočívá v tom, že odlehlé hodnoty jsou v této zhuštěné podobě výrazně náročnější na přesnou reprezentaci v porovnání s normální třídou. V důsledku toho bývá při rekonstrukci odlehlého bodu chyba značná, což nabízí přirozený prostředek k hodnocení datového bodu. Zásadní problém při aplikaci neuronových sítí pro tento účel vyplývá ze zahrnutí odlehlých hodnot do samotného trénovacího modelu, což vede ke zvýšenému riziku přeučení na daný trénovací model. To přispívá k relativně omezenému úspěchu neuronových sítí ve srovnání s jinými metodami redukce dimenzionality při detekci odlehlých hodnot [6].

Convolutional autoencoder (CAE)

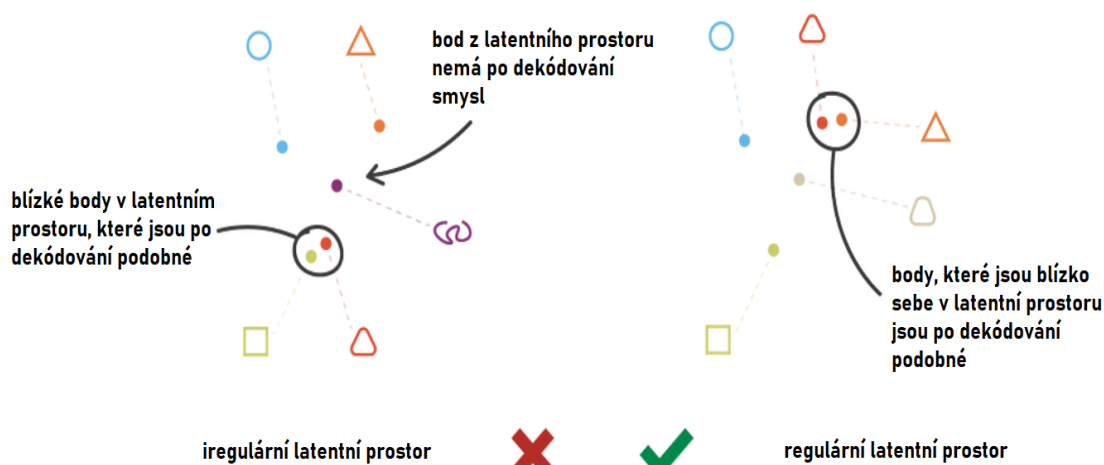
Konvoluční autoenkodérové modely, jsou modely založené na neuronových sítích a určené pro redukci dimenzionality a učení reprezentace v různých úlohách. Na rozdíl od výpočetních nákladů spojených s odstraňováním šumu z obrazu řeší modely CAE tuto úlohu tím, že ji zasazují do statistického rámce regrese, což nabízí lépe zvládnutelný výpočet. Konkrétně je proces odstraňováním šumu z obrazu pomocí modelu CAE koncipován jako problém učení během trénování modelu a odhad parametrů se provádí pomocí modifikace známého algoritmu backpropagation (zpětného šíření). Inovace, kterou CAE zavádí do zpracování obrazu, spočívá v začlenění konvolučních vrstev, které jsou schopny vytvářet abstraktnější reprezentace výchozích vstupů eliminací šumu a redundantních informací. V důsledku toho jsou tyto vrstvy označovány za významnou hranici v oblasti hlubokého učení a analýzy obrazu [7].

Začlenění konvolučních vrstev umožňuje CAE účinně eliminovat šum a konstruovat robustní a stabilní reprezentace příznaků. Zároveň tyto vrstvy zmenšují velikost vstupních rozměrů, takže CAE jsou vhodné pro práci s obrázky s velkým rozměrem a šumem. Na rozdíl od klasických autoenkodérů, u nichž často dochází k výrazné ztrátě informací při skládání a řezání dat, CAE tento proces zvládají efektivněji. Konvoluční vrstvy v CAE zachovávají prostorovou informaci vstupních obrazových dat a zároveň šetrně extrahují relevantní detaily. CAE mají v podstatě schopnost učit se komprimované latentní reprezentace obrazu, čímž zachovávají prostorovou lokalizaci vstupních dat podobně jako jiné konvoluční neuronové sítě (CNN) [7].

Variational autoencoder (VAE)

Podobně jako u běžného autoenkodéru je variační autoenkodér struktura zahrnující kodér a dekodér. Cílem jeho trénování je minimalizovat chybu rekonstrukce mezi zakódovanými a dekodovanými daty a původními daty. Abychom však zavedli určitou regularizaci latentního prostoru, přistoupíme k mírné modifikaci procesu kódování a dekodování: namísto kódování vstupu jako jednoho bodu jej zakódujeme jako rozložení v latentním prostoru. [8]

Regularita, která se od latentního prostoru očekává, aby byl generativní proces možný, lze vyjádřit dvěma hlavními vlastnostmi: spojitostí (dva blízké body v latentním prostoru by po dekodování neměly poskytovat dva zcela odlišné výsledky) a úplností (pro zvolené rozdělení by měl bod vybraný z latentního prostoru po dekodování poskytovat "smysluplný" obsah).



Obrázek 2.3 porovnání iregulárního a regulárního latentního prostoru (převzato a přeloženo z (8))

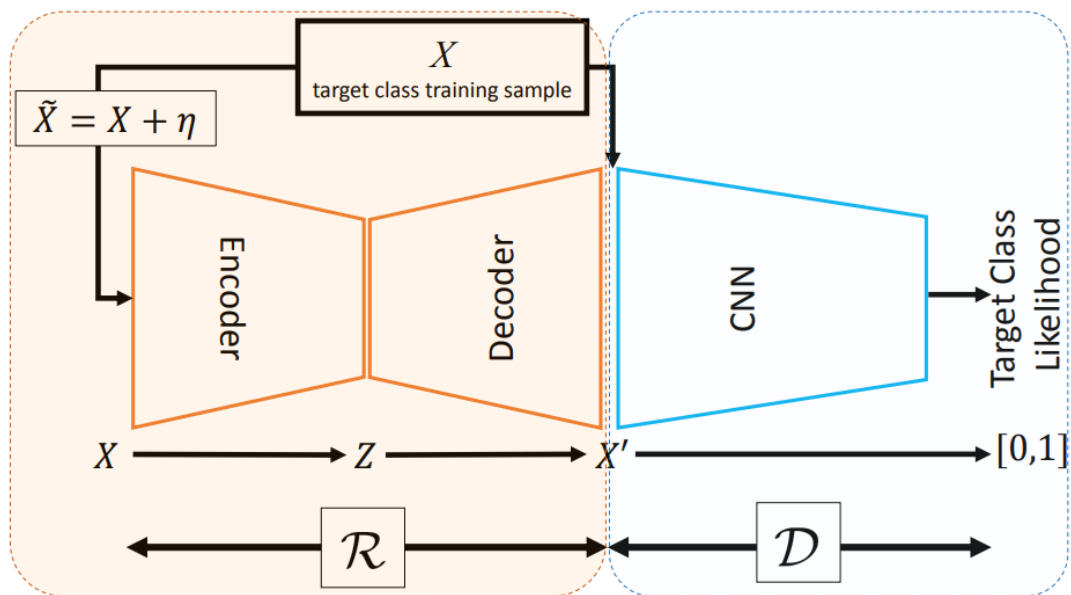
V praxi se kódovaná rozdělení obvykle volí tak, aby odpovídala normálnímu rozdělení. Tato volba umožňuje kodéru naučit se a vrátit střední hodnotu i kovarianční matici, které charakterizují tato Gaussova rozdělení. Důvod pro kódování vstupu jako rozdělení s rozptylem, nikoliv jako jediného bodu, spočívá v jeho přirozeném vyjádření regularizace latentního prostoru. Rozdělení vytvořená kodérem jsou omezena tak, aby se těsně shodovala se standardním normálním rozdělením. Tento přístup zajišťuje jak lokální, tak globální regularizaci latentního prostoru - lokální díky kontrole rozptylu a globální díky kontrole střední hodnoty.[8]

Adversarial autoencoder (AAE)

Adversariální autoenkodér (AAE) funguje podobně jako VAE na latentní reprezentaci je představena rozložením. Zatímco VAE využívá Kullback-Leiblerovu divergenci k dosažení posteriorní hodnoty latentního vektoru s apriorním rozdělením, AAE zahrnuje koncept generativních adverzních sítí (GAN) [9], které budou vysvětleny v další kapitole.

2.2.2 Generative Adversarial Networks (GANs)

Generativní protichůdné sítě, tyto sítě jsou založeny na soupeření mezi dvěma sítěmi, které procházejí souběžným tréninkem bez dohledu. První síť slouží jako generátor (G) a jejím cílem je vytvářet realistická data, například obrázky. Současně druhá síť přebírá roli diskriminátoru (D) a snaží se rozlišovat mezi reálnými daty a daty generovanými sítí G. Specifickým typem GAN, jsou podmíněné sítě GAN. V této variantě přijímá G jako vstup obraz X a generuje nový obraz X' . Mezitím se D snaží rozlišit mezi X a X' , zatímco G se snaží oklamat D tím, že vytváří stále realističtější obrazy [10]. GANs mohou být využity například pro detekci neobvyklých událostí ve videu [11]



Obrázek 2.4 Příklad schématu architektury GAN (převzato [10])

Navrhovaný rámec se skládá ze dvou základních modulů: síť \mathcal{R} , která slouží jako krok předzpracování a rekonstrukce, a síť \mathcal{D} , která je zodpovědná za detekci (diskriminaci). Obě sítě jsou trénovány soupeřivým a nekontrolovaným způsobem v rámci nastavení end-to-end. Síť \mathcal{R} rekonstruuje vstup X , generuje X' a snaží se oklamat síť \mathcal{D} aby klasifikovala X' jako původní vzorek, nikoli rekonstruovaná data. Naopak síť \mathcal{D} má přístup k původnímu souboru dat, což jí umožňuje rekonstruované vzorky odmítnout. Tyto dvě sítě mezi sebou soupeří a díky tréninkové fázi, kdy jsou modelu předkládány vzorky z cílové třídy, se \mathcal{R} zdokonalí v rekonstrukci vzorků z cílové třídy s minimální chybou, aby úspěšně oklamala \mathcal{D} . Výsledkem tréninkového postupu je dvojice sítí, \mathcal{R} a \mathcal{D} , které se obě učí distribuci cílové třídy. Tyto moduly jsou trénovány v rámci adverzního učení ve stylu GAN a tvoří komplexní systém pro klasifikaci jedné třídy a detekci anomálií. Pro zvýšení odolnosti metody vůči šumu a poškozeným vstupním vzorkům se ke vstupním trénovacím vzorkům přidává Gaussův šum a přivádí se do \mathcal{R} [10].

2.2.3 Predictability modeling

Cílem metod založených na modelování předvídatelnosti je získat reprezentace příznaků předpovídáním aktuálních případů dat s využitím reprezentací předchozích případů v rámci časového okna jako kontextu. V tomto kontextu se datové instance vztahují k jednotlivým prvkům v sekvenci, například k videosnímškům ve videosekvenci. Tento přístup se běžně používá pro učení reprezentací sekvencí a predikci [12]. Pro dosažení přesných předpovědí jsou reprezentace navrženy tak, aby zachytily časové, sekvenční a rekurentní závislosti v rámci dané délky sekvence. Normální případy se obvykle dobře drží těchto závislostí a lze je efektivně předpovídat, zatímco anomálie tyto závislosti

často porušují a jsou nepředvídatelné. Chyby predikce proto slouží jako základ pro definování skóre anomálií [3].

2.2.4 Self-supervised classification

Klasifikace pod vlastním dohledem zahrnuje učení reprezentací normality prostřednictvím konstrukce samonaváděcích klasifikačních modelů. Případy, které těmto klasifikačním modelům neodpovídají, označuje jako anomálie. Tato metodika vychází z tradičních přístupů založených na křížové analýze příznaků nebo příznakových modelech [13]. Tyto mělké metody posuzují normalitu datových instancí na základě jejich konzistence se sadou predikčních modelů. Každý model je vycvičen k predikci jednoho příznaku na základě zbývajících příznaků. Na rozdíl od těchto studií, které se zaměřují na tabulková data a vytvářejí příznakové modely pomocí původních dat, se hloubková detekce anomálií založená na konzistenci zaměřuje na obrazová data a vytváří prediktivní modely pomocí rozšířených dat založených na transformaci příznaků. Aby bylo možné efektivně rozlišovat transformované instance, jsou klasifikační modely nuceny učit se rysy klíčové pro popis základních vzorů v instanci trénovacích dat. V důsledku toho vykazují normální případy obecně silnější shodu s klasifikačními modely.

2.3 Architektury Anomaly-measure dependent feature learning

V překladu učení příznaků závislé na míře anomálie. Odlišně od obecného přístupu učení příznaků, který počítá skóre anomálií pomocí heuristiky po získání naučených reprezentací, tento směr výzkumu integruje již existující míru anomálie do cílové funkce učení příznaků. Cílem je optimalizovat reprezentace příznaků specificky pro danou míru anomálie [4].

Konkrétní architektury, které budou rozebrány dále v této kapitole jsou: Distance-based measures (metriky na základě vzdálenosti), One-class classification measures (metriky jednotřídní klasifikace) a Clustering-based measure (metriky na bázi shlukování).

2.3.1 Distance based measures

V překladu metriky na základě vzdálenosti, algoritmy detekce anomálií založená na vzdálenosti se snaží získat reprezentace příznaků přizpůsobené pro specifické míry anomálií založené na vzdálenosti. Techniky založené na vzdálenostech jsou známé svou jednoduchostí a snadnou implementací, přičemž byly zavedeny různé účinné míry anomálií, jako jsou DB (distance based) odlehle hodnoty, vzdálenost k nejbližších sousedů, průměrná vzdálenost k nejbližších sousedů, relativní vzdálenost a náhodná vzdálenost nejbližšího souseda. Výraznou nevýhodou těchto konvenčních měr anomálií

založených na vzdálenostech je jejich neefektivnost ve vysokodimenzionálních datech v důsledku prokletí dimenzionality [14]. Hluboká detekce anomálií založená na vzdálenosti řeší toto omezení tím, že před použitím míry vzdálenosti promítne data do nízkorozměrného prostoru, což umožňuje efektivní zpracování vysokorozměrných dat za cenu ztráty informace [4].

2.3.2 One-class classification measures

V překladu metriky jednotřídni klasifikace. Tato sada metod je určena k získání reprezentací příznaků přizpůsobených pro následnou detekci anomálií na základě klasifikace jedné třídy. Klasifikace jedné třídy zahrnuje úkol sestavit popis množiny datových případů, aby bylo možné rozlišit, zda nové případy odpovídají vzorům pozorovaným v trénovacích datech. Je to jedna z převládajících metodik detekce anomálií. Mnoho modelů v klasifikaci jedné třídy čerpá inspiraci ze Support Vector Machines (SVM) (stroje podpůrných vektorů), například dva hojně využívané modely: jedno-třídni SVM (OC-SVM) a Support Vector Data Description (SVDD) (popis dat podpůrnými vektory). Klíčová oblast výzkumu v rámci této kategorie zahrnuje učení reprezentací, které jsou specificky optimalizovány pro tyto konvenční modely klasifikace jedné třídy [4].

OC-SVM

Metoda detekce anomálií bez dohledu OC-SVM je rozšířením metody podpůrných vektorů, která se běžně používá při klasifikaci. Na rozdíl od klasické SVM, která hledá nadrovinu pro maximalizaci rozpětí mezi datovými body, se OC-SVM zaměřuje na učení nadroviny, která efektivně odděluje datové body od počátku. SVM obecně disponují schopností zachytit nelinearitu díky použití kernelů. Metoda kernelů usnadňuje mapování datových bodů ze vstupního prostoru příznaků v R^d do prostoru vyšších dimenzí R^D , kde se data stanou lineárně separovatelnými prostřednictvím transformace $R^d \rightarrow R^D$ [15].

DeepSSVD

Deep Support Vector Data Description (Deep SVDD)(hluboký popis dat s podpůrnými vektory) je metoda, která zahrnuje trénování neuronové sítě za současné minimalizace objemu hyperkoule obklopující síťové reprezentace dat. Minimalizací objemu hyperkoule je síť nucena extrahovat obvyklé variační faktory, protože musí těsně mapovat datové body do středu koule. Výsledkem je, že normální příklady dat jsou mapovány těsně ke středu hyperkoule, zatímco anomální příklady jsou mapovány dále od středu nebo mimo hyperkouli. Tím získáme kompaktní popis normální třídy. Minimalizace velikosti sféry tento proces učení vynucuje. [16].

2.3.3 Clustering-based measures

Detekce anomálií založená na shlukování se snaží získat reprezentace, které v nově naučeném prostoru zřetelně odlišují anomálie od shluků. Shlukování a detekce anomálií

jsou ze své podstaty propojené úlohy, což vedlo k četným studiím zaměřeným na využití výsledků shlukování pro definici anomálií. Tyto studie zahrnují metody, jako je velikost shluku, vzdálenost ke středům shluků, vzdálenost mezi středy shluků a členství ve shluku.

Hlavní myšlenkou těchto metod je předpoklad že normální případy vykazují větší shodu se shluky ve srovnání s anomáliemi. Klíčovým prvkem tohoto přístupu k detekci anomálií je hluboké shlukování, jehož cílem je získat reprezentace příznaků optimalizované pro konkrétní shlukovací algoritmus. Hlavní důvod vychází z pozorování, že výkon shlukovacích metod je významně ovlivněn vstupními daty. Naučit se reprezentace příznaků explicitně přizpůsobené pro shlukovací algoritmus zajišťuje robustní výkonnost v různých souborech dat [4].

Obecně lze říci, že tento koncept je založen na dvou hlavních bodech: dobré reprezentace zlepšují výsledky shlukování a dobré výsledky shlukování mohou poskytnout cenné podněty pro učení reprezentací; a reprezentace vyladěné pro konkrétní algoritmus shlukování nemusí být univerzálně použitelné pro jiné algoritmy shlukování vzhledem k odlišným základním předpokladům, které jsou vlastní každému algoritmu [4].

2.4 End-to-end anomaly score learning (end-to-end učení se skóre anomaly)

V překladu end-to-end učení se skóre anomaly. Tento směr výzkumu se zaměřuje na získávání skalárních skóre anomálií end-to-end způsobem. Na rozdíl od učení příznaků závislých na míře anomálie je v tomto přístupu skórování anomálií nezávislé na již existujících mírách anomálií ale má neuronovou síť, která se přímo učí skóre anomálií. K řízení sítě pro skórování anomálií jsou obvykle nutné jedinečné ztrátové funkce.

Na rozdíl od metod hlubokého učení pro získání příznaků, které využívají heuristiku pro výpočet skóre anomálií po získání naučených reprezentací, se techniky v této kategorii učí současně jak reprezentace příznaků, tak skóre anomálií. Tento souběžný proces učení výrazně zlepšuje optimalizaci skóre anomálií a/nebo řazení anomálií. V tomto ohledu sdílejí některé podobnosti s metodami učení příznaků reprezentující normální třídu. Metody učení příznaků závislé na míře anomálie se však často potýkají s přirozenými omezeními spojenými se začleněnými mírami anomálie, což je nedostatek, který metody v této kategorii nemají. Kromě toho tyto metody představují dva naprosto odlišné přístupy k návrhu modelu: jeden se zaměřuje na to, jak syntetizovat už existující míry anomálií a modely neuronových sítí, zatímco druhý se soustřeďuje na vývoj nových ztrátových funkcí pro přímé učení skóre anomálií.[4]

Konkrétní architektury, které budou rozebrány dále v této kapitole jsou: prior-driven models (modely řízené prioritou), softmax likelihood models a end-to-end one-class classification models.

2.4.1 Prior-driven Models

V překladu modely řízené prioritou. Tento přístup využívá k zakódování a řízení učení skóre anomálií rozdělení předpovědí. Jelikož se skóre anomálií učí způsobem end-to-end, lze prioritu použít buď na vnitřní modul, nebo na výstup učení funkce učení skóre τ .

Zavedená priorita zahrnuje přirozenou (ab)normalitu souboru dat. Integraci priority do interní skórovací funkce anomálie ilustruje zkoumání bayesovského přístupu inverse reinforcement learning (IRL) (inverzního učení s posilováním) [18]. Základní myšlenka spočívá v tom, že pokud je agentovi jako vstup poskytnuta řada sekvenčních dat, lze jeho pravidelné chování charakterizovat pomocí latentní funkce odměny. V důsledku toho je testovací sekvence identifikována jako anomální, pokud jí agent přiřadí nízkou odměnu. V této souvislosti se k odvození funkce odměny používají přístupy IRL založené na vzorcích, které zvyšují efektivitu procesu učení [4].

Mezi výhody tohoto přístupu patří že optimalizaci skóre anomálií lze přímo sladit se zadanou prioritou. Tento rámec umožňuje flexibilitu při integraci různých rozdělení priorit do procesu učení skóre anomálií. Pro efektivní detekci anomálií lze přizpůsobit různé techniky bayesovského hlubokého učení. Začlenění priority může přispět ke skóre anomálií, které je ve srovnání s jinými metodami lépe interpretovatelné.

Mezi nevýhody těchto modelů patří následující. Navrhnout univerzálně účinný prior pro různé scénáře aplikací detekce anomálií je náročné, ne-li nemožné. Účinnost modelů se může snížit, pokud priorita není dobře sladěna se základním rozdělením [4].

2.4.2 Softmax Likelihood Models

Tato metoda se zaměřuje na získávání skóre anomálií maximalizací pravděpodobnosti událostí přítomných v trénovacích datech. Vzhledem k tomu, že normální a anomální případy jsou spojeny s častými a vzácnými vzory, očekává se, že normální případy budou představovat události s vysokou pravděpodobností, zatímco anomálie budou spíše události s nízkou pravděpodobností [4]. V důsledku toho lze skóre anomálie přirozeně definovat jako inverzní hodnotu pravděpodobnosti události. Při realizaci tohoto cíle se osvědčily jako účinné a efektivní softmaxové pravděpodobnostní modely, zejména prostřednictvím technik, jako je noise contrastive estimation (NCE) (kontrastní odhad šumu) [19].

2.4.3 End-to-end One-class Classification

Tato sada metod se věnuje trénování klasifikátoru způsobem end-to-end. Na rozdíl od metrik jednotřídní klasifikace se tento přístup nespolehá na zavedené klasifikační míry jedné třídy, jako je OC-SVM nebo SVDD. Místo toho vzniká integrací GAN a konceptu soupeřivého učení. Základní koncepce spočívá v trénování diskriminátoru jedné třídy pro normální případy, který je účinně odlišuje od adverzně generovaných pseudoanomálií. Tento přístup se výrazně liší od metod založených čistě na GAN v

části ve dvou zásadních aspektech. Zaprvé, metody založené na GAN se snaží naučit generativní rozdělení pro maximální aproximaci skutečného rozdělení dat, čímž vytvářejí generativní model, který efektivně zachycuje normalitu trénovacích instancí. Naopak metody v této části se snaží optimalizovat diskriminační model pro oddělení normálních instancí od adverzně generovaných okrajových instancí. Za druhé, metody založené na GAN určují skóre anomálií na základě reziduí mezi skutečnými instancemi a odpovídajícími generovanými instancemi, zatímco metody zde přímo využívají diskriminátor ke klasifikaci anomálií [4]. Tento přístup funguje za předpokladu že datové body, které se přibližují anomáliím mohou být efektivně syntetizovány a veškeré instance normální třídy mohou být shrnuty diskriminačním jednotřídním modelem.

Koncepce adversarially learned one-class classification (ALOCC) (protichůdně naučená klasifikace jedné třídy) byla původně zkoumána v [10]. Základní přístup zahrnuje trénování dvou hlubokých sítí, kde jedna síť slouží jako model jedné třídy a rozlišuje normální případy od anomálií. Současně je druhá síť trénována tak, aby zvýrazňovala normální případy a generovala zkreslené odlehlé hodnoty. Obě sítě jsou instancovány a optimalizovány pomocí přístupu GAN. Model jedné třídy je postaven na diskriminační síti, zatímco generátorová síť je konstruována na základě denoisingového autoenkodéru (AE) [4]. Hlavní myšlenkou této architektury je že AE dokáže zdařile rekonstruovat, a dokonce zvýraznit instance normální třídy ale při předložení anomálií na vstup generuje vysoce zkreslené výstupy. Prostřednictvím procesu optimalizace minimaxem zlepšuje diskriminátor D svou schopnost rozlišit normální případy od odlehlých ve srovnání s použitím původních datových případů. V důsledku toho lze D použít přímo pro detekci anomálií [4].

3. TVORBA DATASETU

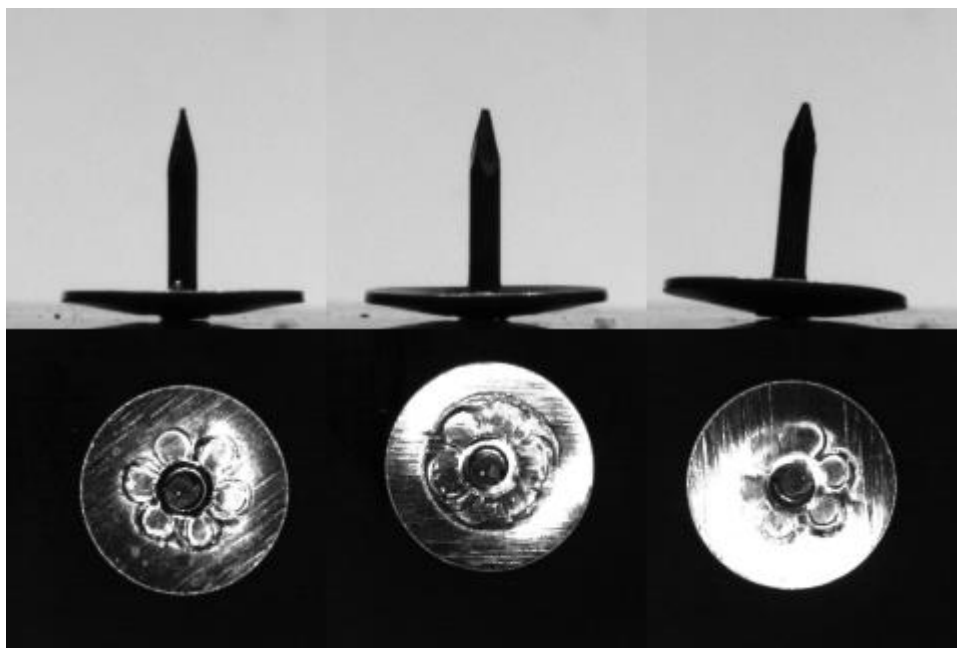
V úvodu jsou vysvětleny pojmy týkající se zbytku kapitoly, dále výběr průmyslového výrobku, vytváření scény a samotné pořizování datasetu, na závěr je popsán proces augmentace dat.

3.1 Definice tříd

V této podkapitole jsou vysvětleny pojmy týkající se datasetu, se kterými bude následně pracováno.

3.1.1 OK kusy

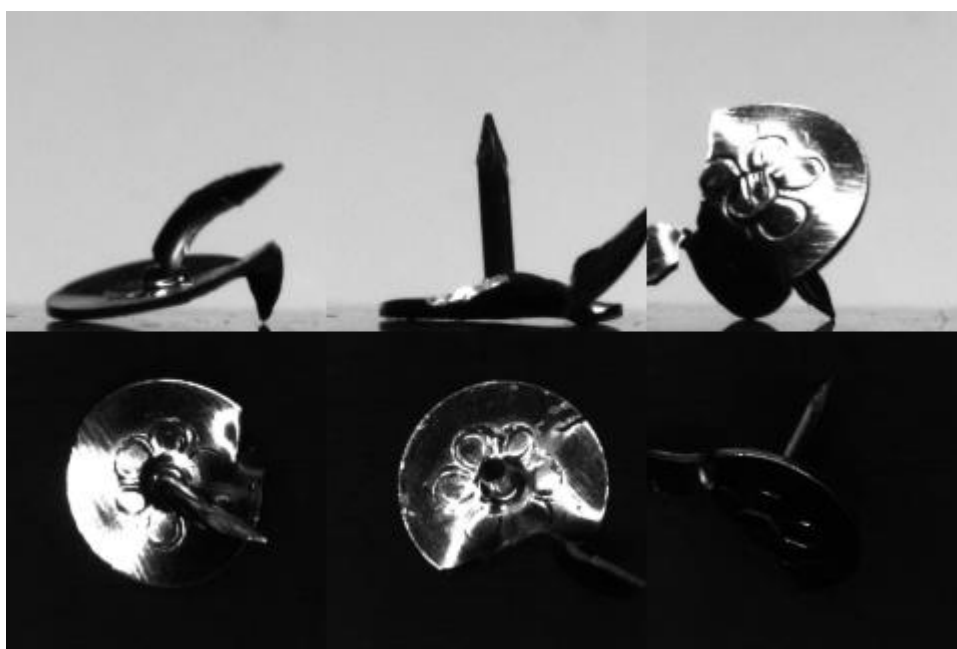
Datová reprezentace výrobků, které by podle definice výrobce prošly kontrolou kvality. Jedná se o běžnou třídu. Na obrázku 3.1 jsou příklady OK kusů, sloupečky představují vždy jeden stejný připínáček.



Obrázek 3.1 Příklady OK kusů (horní řádek boční pohled, spodní řádek horní pohled, 160x160 pixelů)

3.1.2 NOK kusy

Datová reprezentace výrobků, které by naopak neprošly kontrolou kvality výrobce. Jedná se o zájmovou třídu. Na obrázku 3.2 jsou příklady NOK kusů, sloupečky představují vždy jeden stejný připínáček.



Obrázek 3.2 Příklady NOK kusů (horní řádek boční pohled, spodní řádek horní pohled, 160x160 pixelů, každý sloupek představuje odlišný špendlík)

3.1.3 Normální třída

Normální třída představuje soubor dat, který reprezentuje kusy s dostatečnou kvalitou dle definice výrobce. Jde tedy o OK kusy, výrobky, které chceme zachovat a algoritmus by je tedy neměl zařadit mezi anomálie. V reálných aplikacích se může stát, že normální třída obsahuje jednu či více NOK kusů, například při špatné anotaci datasetu, tomuto problému se věnuje experiment v [5.3](#).

3.1.4 Anomálie

Anomálie představují soubor dat, který reprezentuje NOK kusy podle definice výrobce. S ohledem na klasifikaci se jedná o zájmovou třídu, kterou chceme mezi kvalitními kusy vyhledávat a z výrobního procesu vyřadit. Opět se může stát, že v praktických aplikacích dojde ke špatnému anotování či přiřazení dat do této skupiny.

3.2 Výběr průmyslového výrobku

V praxi se běžně kamerami kontroluje velice široká škála různých výrobků, od nábojnic, matiček, keramických filtrů pro odlévání hliníku až po netkané textilie a například léky.

Hlavní kritériem při výběru produktu pro tuto práci byla možnost dostatečné vizuální rozlišitelnosti OK/NOK kusů. Jedná se o vizuální kontrolu, obecně nebude

možné parametry, které neovlivňují rozměry, povrch či vlastnosti světla odráženého od vyšetřovaného kusu vyhodnotit.

Dalším kritériem byly rozměry výrobku, výběr miniaturních součástek jako například SMD kondenzátorů či rezistorů by pravděpodobně vyžadovaly návrh upínacího zařízení pro zajištění opakovatelnosti scény a představoval by problém při manipulaci. Naopak příliš velké výrobky by mohly vyžadovat více kamer, více pohledů, obtížnější manipulaci nebo problémy s uskladněním.

Spolu s rozměry se taky do jisté míry váže cena, dataset má dle zadání čítat alespoň 1000 realizací, z toho důvodu byla kusová cena držena co možná nejnižší. Nehledě na to, že část výrobků musela být pro účely pořízení NOK kusů zničena. Dále by se mělo jednat o běžně dostupný výrobek.

Na základě předešlých kritérií byl, jako snímaný objekt vybrán kovový niklovaný připínáček HODAN 222 s průměrem 10 mm a délkou 8 mm.



Obrázek 3.3 Snímané připínáčky HODAN 222 (převzato z [20])

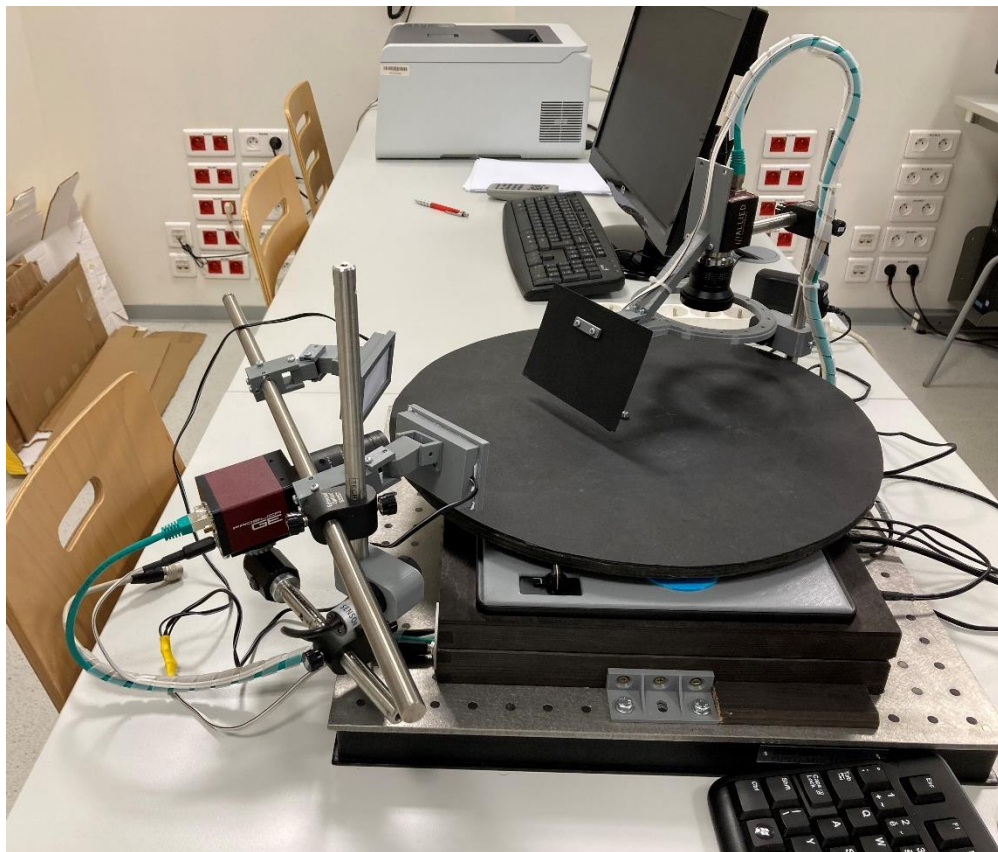
Připínáčky byly vybrány na základě jednoduchého geometrického tvaru, rozlišitelnosti OK/NOK kusu, relativně malým rozměrům, jednoduché skladnosti, malé pořizovací ceny, 100ks lze pořídit asi za 10kč a jednoduché dostupnosti, připínáčky lze pořídit online nebo prakticky v každém papírnictví.

3.3 Scéna

Tato podkapitola popisuje postup tvorby scény a výběr komponentů, které ji tvoří.

3.3.1 Karuselový přípravek

Podle prvotních předpokladů bylo zamýšleno že dataset bude pořízen na přípravku dostupném v laboratoři.

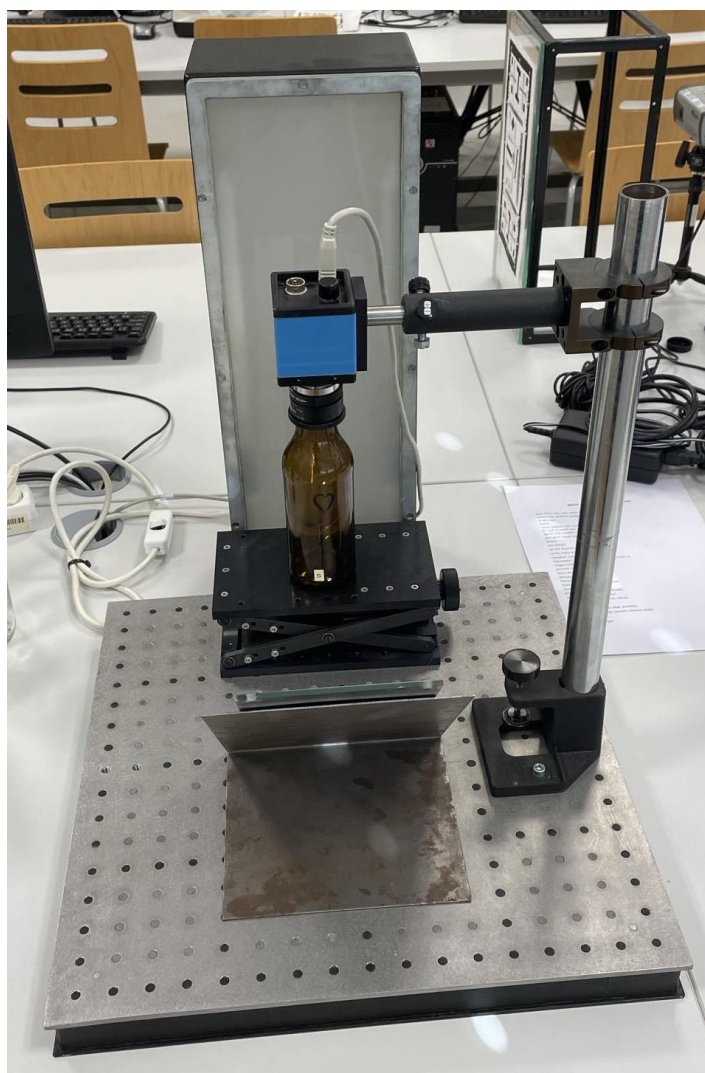


Obrázek 3.4 Karuselový přípravek

Zařízení se ale ukázalo jako nespolehlivé, pro práci s přípravkem není žádný návod, instrukce v souborech obsáhlých v adresářích ovládacího programu nedostačovaly na jednoduché zprovoznění systému. Systém Anubis často padá a následně je nutno celé zařízení restartovat a opět oživit pomocí příkazového řádku. Extrakce obtížně pořízených dat také představuje problém, kdy se pořízené fotky musí extrahovat pomocí příkazového řádku z Docker kontejneru, na toto ovšem musí uživatel přijít úplně sám. Po opakovaných pokusech a mírných úpravách provedených pracovníky ze skupiny strojového vidění bylo rozhodnuto jít vlastní cestou a sestavit vlastní scénu pro pořízení datasetu.

3.3.2 Tvorba vlastní scény

Za účelem tvorby datasetu bylo modifikováno pracoviště na defektoskopii v laboratoři SE 2.149



Obrázek 3.5 Výchozí stav pracoviště

Pracoviště bylo vybráno kvůli zadnímu podsvícení, které dobře zvýrazňuje siluetu připínáčku a disponuje nastavitelným stolem pro precizní nastavení polohy a pozice kusu a kamery. Pracoviště nadále již obsahovalo kameru IMAGING SOURCE DFK 21AU04, která byla použita na pořízení snímku z vrchní strany připínáčku, tato kamera byla osazena objektivem COMPUTAR 12 mm 1:1,2 1/2". Kamera DFK 21AU04 disponuje formátem 640x480 pixelů, maximální frekvencí až 60 snímků za sekundu, připojitelností přes USB a dobou expozice 1/10000 až 30 s podrobnější specifikace jsou v katalogu výrobce [21]. Objektiv computar disponuje ohniskovou vzdáleností 12 mm a světelností 1:1,2.

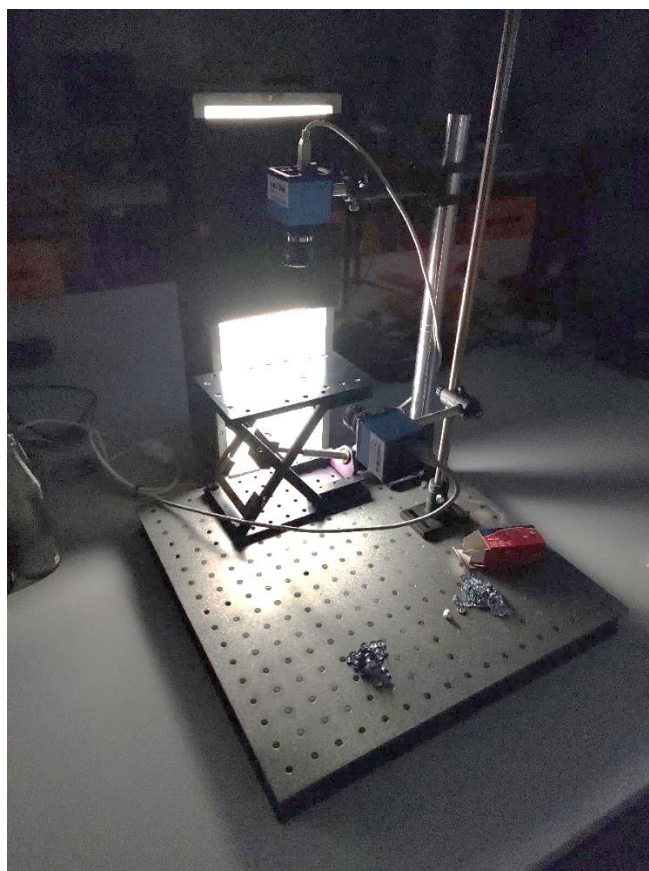
Pracoviště bylo následně upraveno, bylo odstraněno zrcadlo a přidán stojan spolu s kamerou, která snímá půdorys připínáčku z bočního pohledu. Z dispozice laboratoře byla vybrána kamera IMAGING SOURCE DFK 41BU02.H, kamera disponuje formátem 1280x960 pixelů, maximální frekvencí 15 snímků za sekundu, připojitelností

přes USB a dobou expozice 1/10000 až 30 s podrobnější specifikace jsou v katalogu výrobce [22].

Na kameru byl namontován teleobjektiv Ricoh s ohniskovou vzdáleností 12 mm a světelností 1:1,2.



Obrázek 3.6 Horní kamera DFK 21AU04 (pro focení byl použit objektiv computer)



Obrázek 3.7 Upravené pracoviště

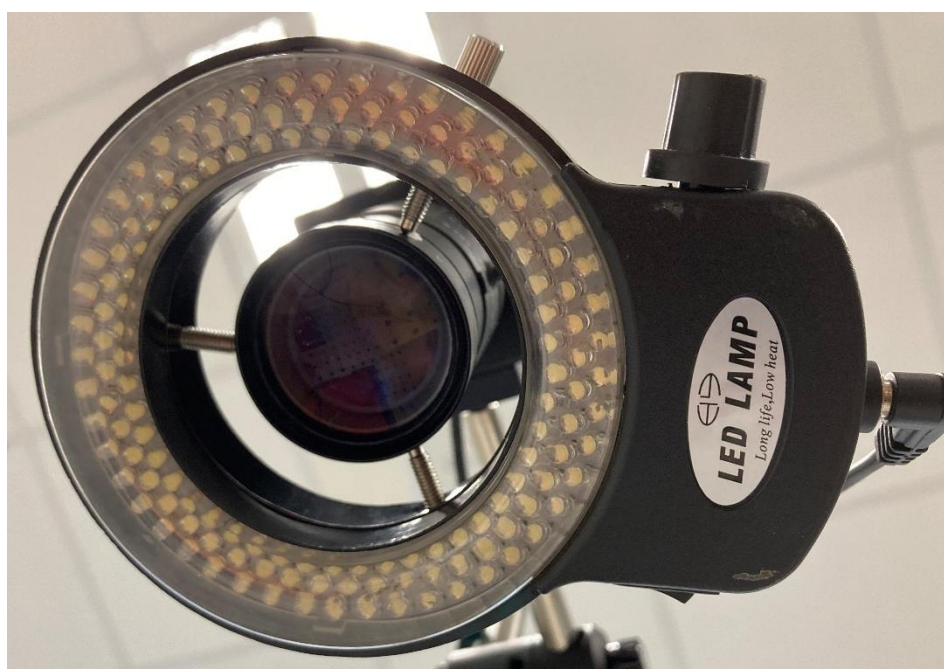
Jak je vidět na níže uvedených histogramech, tato konfigurace nenabízí uspokojivé výsledky. Objektiv byl vyměněn za objektiv Honeywell DFK HLM5V50F13. Tento objektiv disponuje ohniskovou vzdáleností 5-50 mm, clonou F 1,3-zavřeno bližší parametry jsou v katalogy výrobce [23]. Objektiv se ukázal vhodnější mimo jiné díky plynulé možnosti přiblížení, která umožňuje lépe využít celý obraz kamery a povedlo se s ním eliminovat rozostření části podstavy.

Nadále bylo přidáno kruhové světlo Mic 209 144, světlo disponuje plynulým nastavením intenzity osvětlení a šrouby pro uchycení na objektiv kamery. Světlo bohužel nebylo možno upevnit na objektiv horní kamery, jelikož průměr objektivu horní kamery není dostatečně velký. Pro uchycení světla byl ze součástek dostupných v laboratoři zkonstruován stojan.

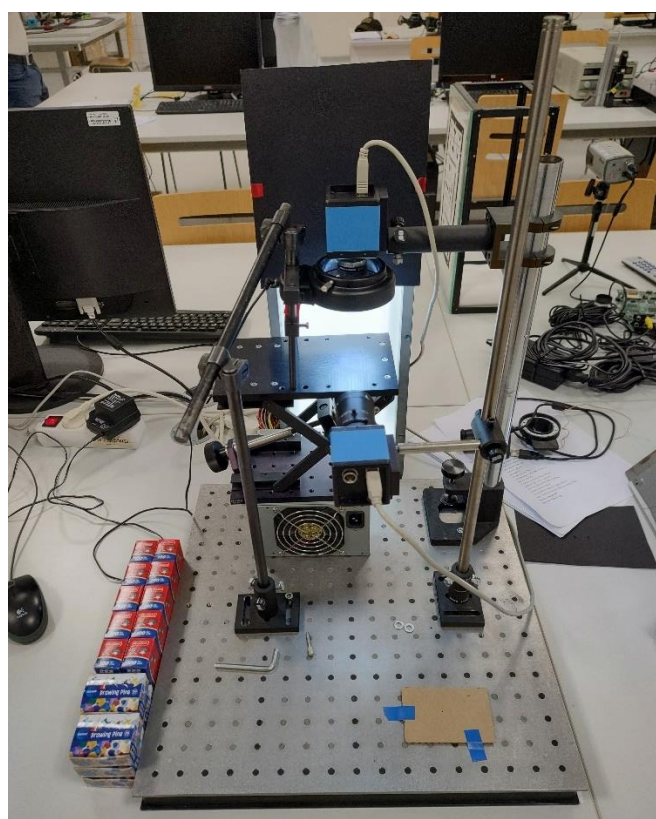
Postupným upravováním bylo dosaženo optimálního nastavení scény. Primárně šlo o různá nastavení expozice, zesílení čipu kamery, intenzity osvětlení a polohy samotného snímaného objektu.



Obrázek 3.8 Boční kamera DFK 41BU02.H s objektivem HLM5V50F13



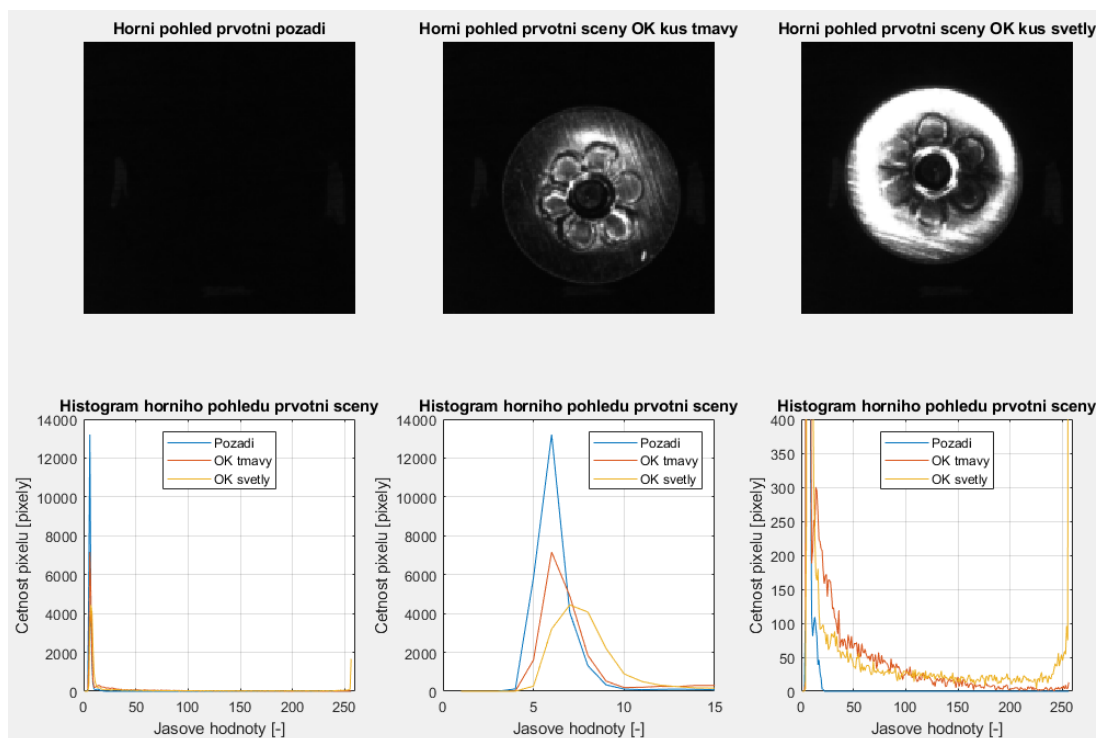
Obrázek 3.9 Světlo horního přívětu Mic 209 144



Obrázek 3.10 Finální verze pracoviště, výměna objektivu, přidání horního přívětu

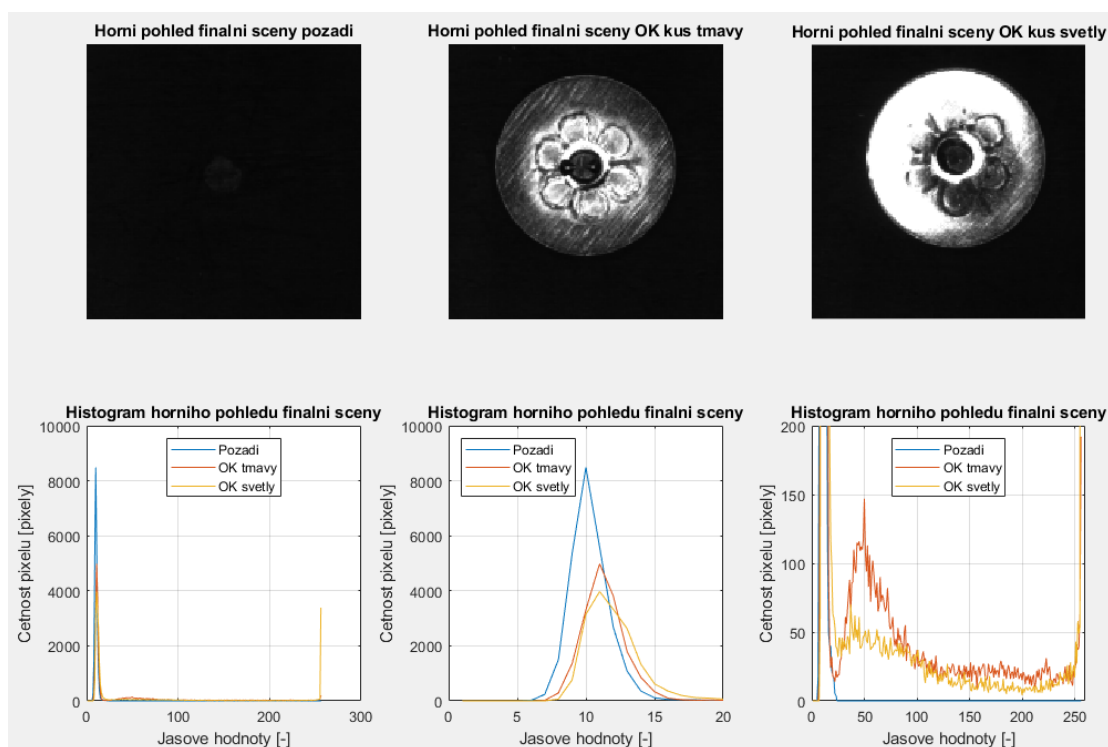
3.3.3 Horní pohled

V této podkapitole je rozebrán postupný vývoj horního pohledu na zájmový objekt s ukázkou histogramů fotek. U horního pohledu představovaly odlesky daleko větší problém než u bočního pohledu.



Obrázek 3.11 Příklady obrázků horního pohledu prvotní scény a jejich histogramy

Ze snímků byly ručně vybrány extrémy představující velmi tmavý, a naopak velmi světlý snímek s odrazem na připínáčku. Průběhy histogramů jsou spíše ploché s nejednoznačným odstupem zájmového objektu a pozadí. Scéna byla následně upravena přidáním horního přísvitů ve formě kruhového světla.

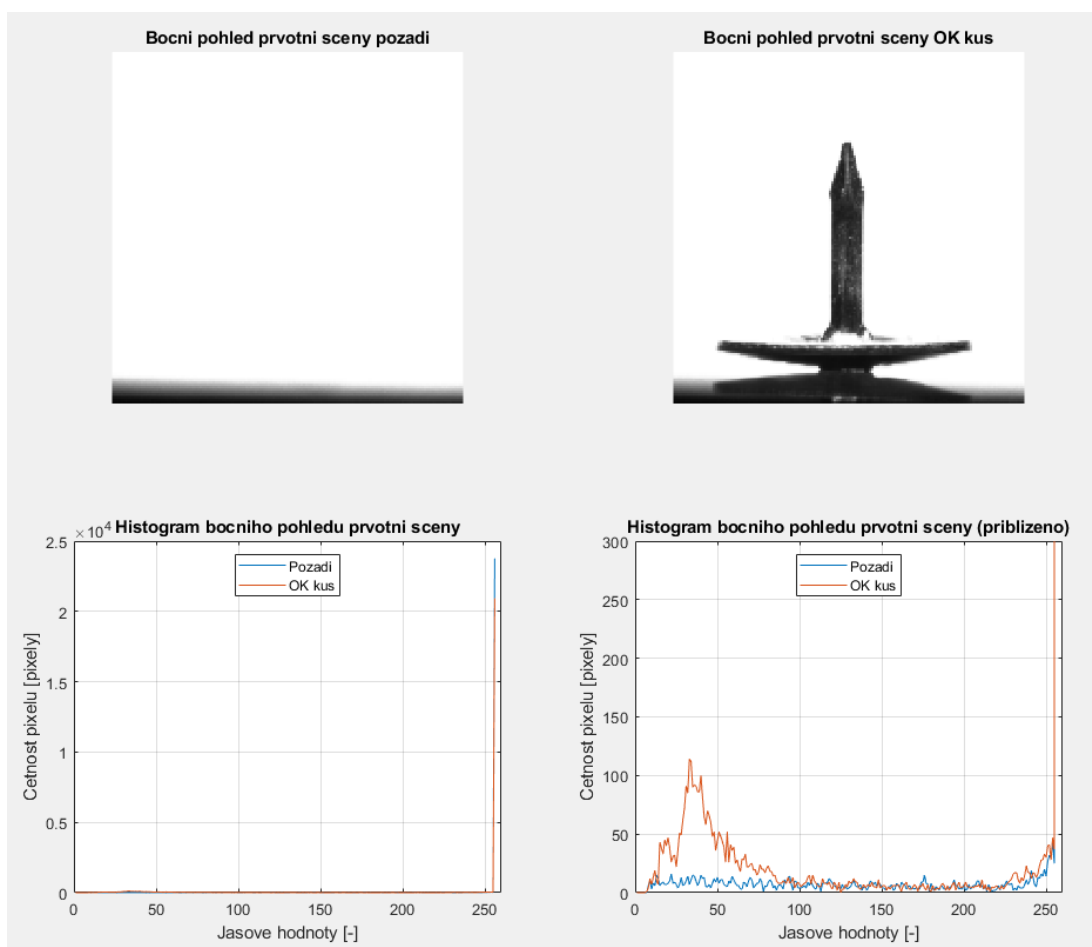


Obrázek 3.12 Příklady obrázků horního pohledu finalní scény a jejich histogramy

Přidání přísvitu pomohlo odstupu zájmového objektu a pozadí a také pomohlo eliminaci velmi tmavých snímků. Odrazy z pohledu horní kamery se nepovedlo odstranit, ale neměly by představovat až tak závažný problém, jelikož se jedná spíše o určení správného geometrického tvaru objektu než kvality jeho povrchu.

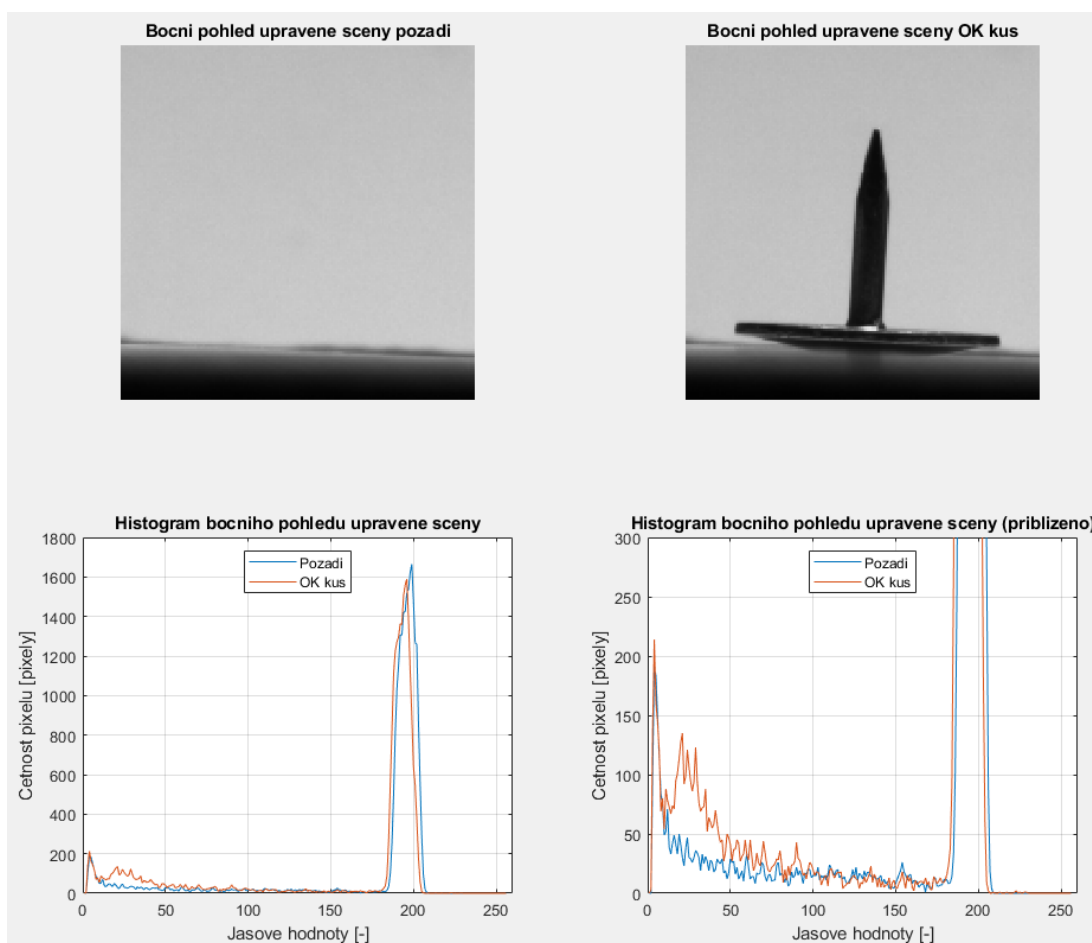
3.3.4 Boční pohled

V této podkapitole je rozebrán postupný vývoj bočního pohledu na zájmový objekt s ukázkou histogramů fotek.



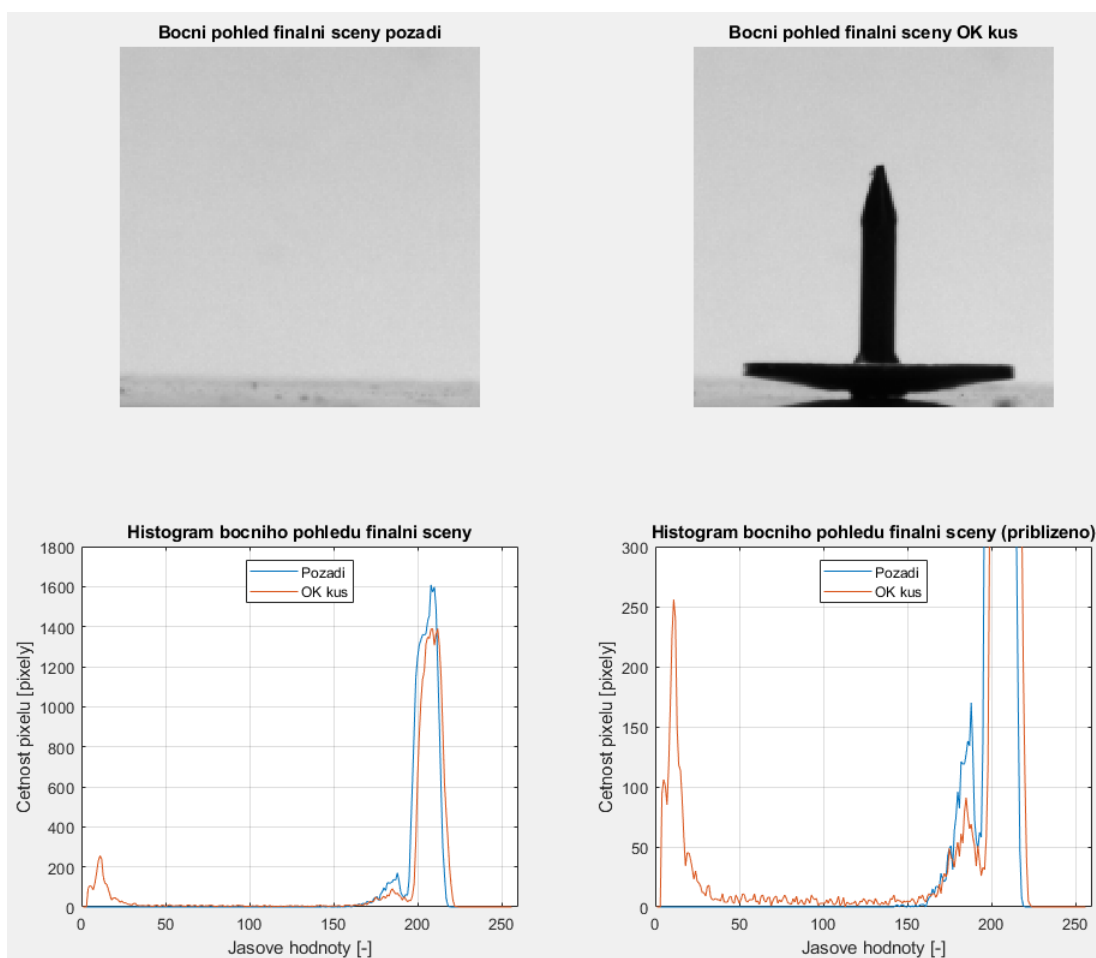
Obrázek 3.13 Příklady obrázků bočního pohledu prvotní scény a jejich histogramy

Z přibližného histogramu je znatelné že četnosti jasových hodnot zájmového objektu a pozadí nejsou příliš vzdálené, oblast v histogramu představující kus je relativně široká, obraz je přesvícený a některé oblasti na výrobku splývají s pozadím jako například spojení mezi kloboučkem a hrotem připínáčku.



Obrázek 3.14 Příklady obrázků bočního pohledu upravené scény a jejich histogramy

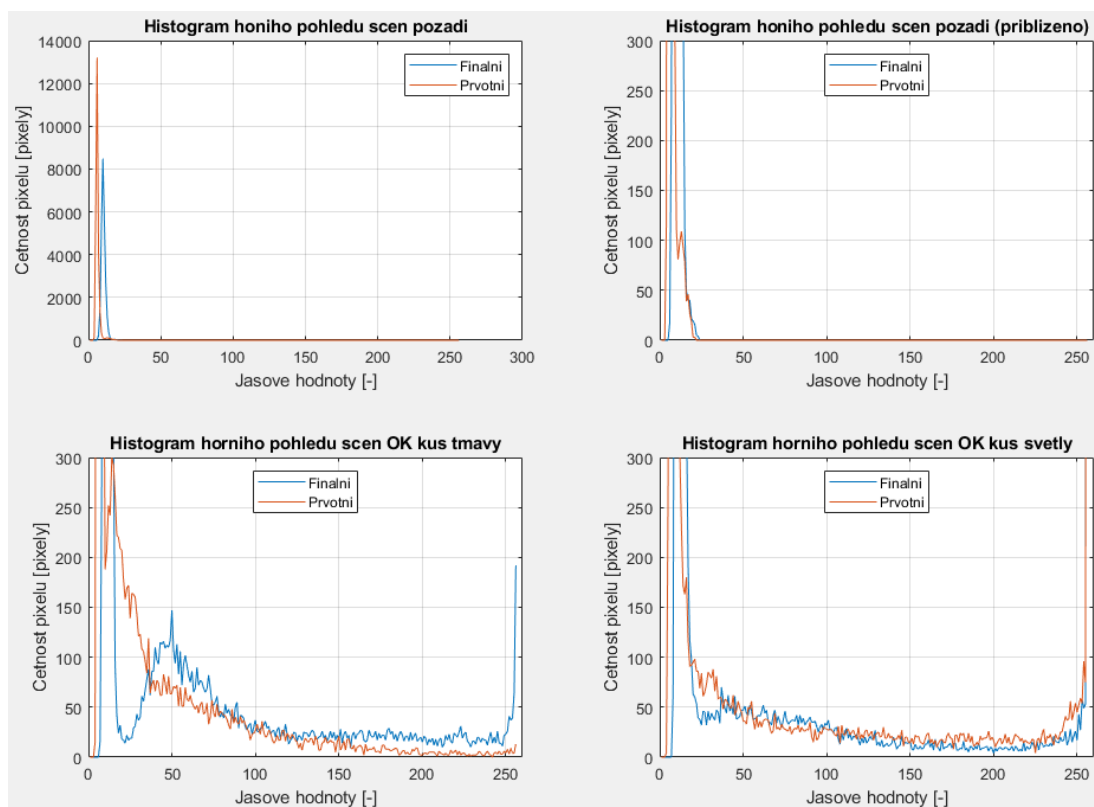
Upravením scény prakticky zmizela přesvícená místa na zájmovém objektu, která by splývala s pozadím, ale naopak vznikl nový problém, kdy na některých fotkách část kloboučku připínáčku začala splývat s rozostřenou částí podstavu. Histogramy opět neukazují velmi značný rozdíl v porovnání mezi scénou s OK kusem a samotným pozadím. Další problém představovala skutečnost, že mohlo dojít ke znehodnocení fotky přítomností cizího objektu, kamera snímala frekvencí kolem 3 snímků za sekundu a někdy došlo k tomu že po umístění připínáčku na podstavu a pořízení fotky byla na uložené fotce vidět část nebo celá ruka. Příčina nebyla s jistotou zjištěna ale zvýšení snímkové frekvence tento problém odstranilo.



Obrázek 3.15 Příklady obrázků bočního pohledu finální scény a jejich histogramy

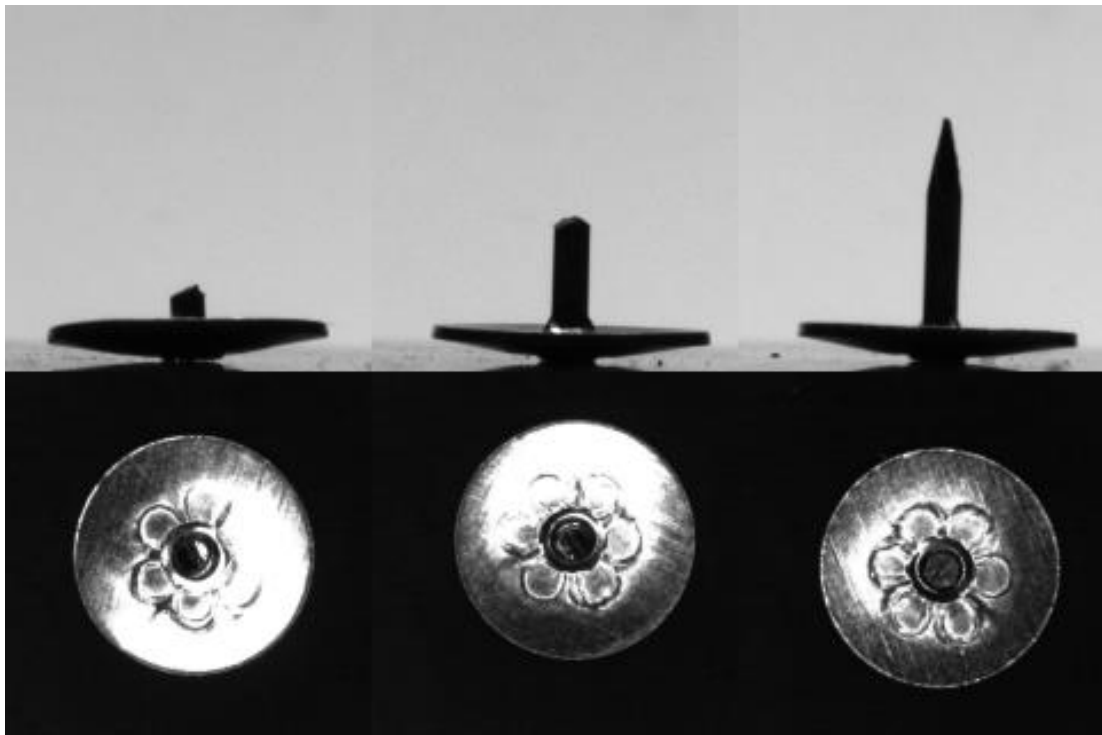
Upravením scény do finální podoby byly eliminovány prakticky všechny odlesky a nastavením kamery dosažen co možná největší kontrast mezi snímaným objektem a pozadím, v předchozích iteracích představovala podstava veliký problém, ale to se povedlo eliminovat vhodnou pozicí kamery, kdy se část světla z podsvitu odráží od podstavy směrem ke kameře. Z histogramu samotného pozadí jde poznat že v oblasti jasových hodnot 0-150 nejsou v podstatě žádné pixely, a právě v této oblasti se v histogramu nachází reprezentace snímaného objektu.

3.3.5 Zhodnocení finální scény

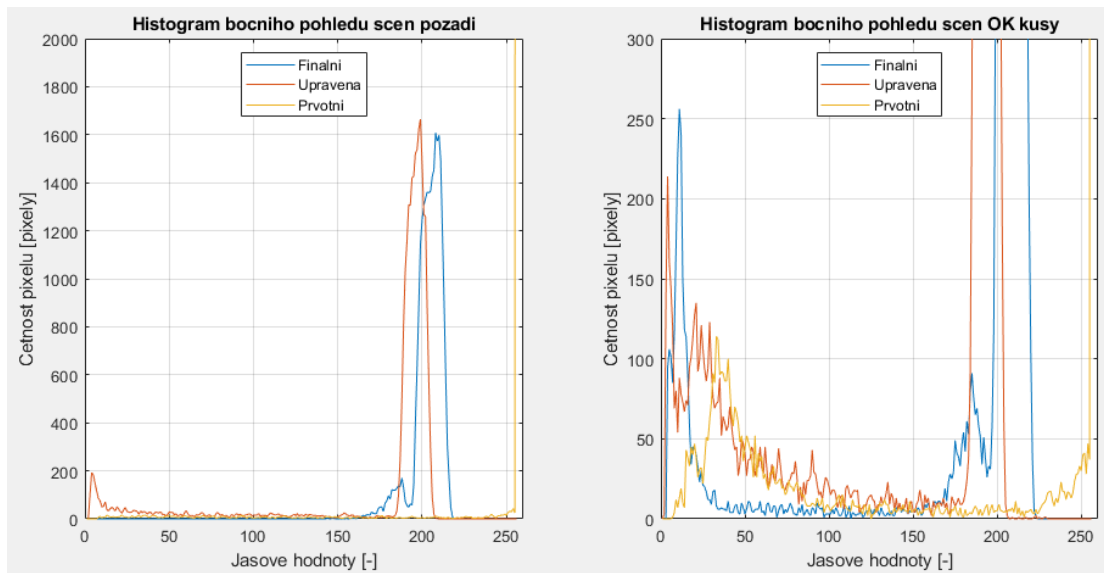


Obrázek 3.16 Porovnání histogramů hodních pohledů scény

Upravením osvětlení na scéně se povedlo zvýšit rozestup mezi zájmovým objektem a pozadím. I přesto že dochází k odleskům je reprezentace geometrického tvaru připínáček zachována. Pro úlohu vyhodnocování kvality povrchu výrobku by odlesky pravděpodobně představovaly velký problém, úloha rozebíraná v této diplomové práci se ale zabývá spíše geometrickými tvary připínáček. Horní pohled bohužel neumožňuje plně vyhodnotit kvalitu připínáčku. Předpokládá se že vyhodnocení kvality, konkrétněji úplnosti hrotu nebude možno algoritmem rozlišit, z horního pohledu totiž nemusí být tento parametr vůbec zřetelný, jak ukazuje obrázek 3.17 níže, každý sloupek představuje jeden připínáček, boční pohled nahoře a horní pohled dole. První dva připínáčky představují NOK kusy a poslední OK kus.



Obrázek 3.17 Ukázka nedostatku horního pohledu



Obrázek 3.18 Porovnání histogramů bočních pohledů scény

Prvotní scéna měla sice nejlepší reprezentaci pozadí, většina pixelů obrazu byla bílá ale docházelo ke zbytečnému přesvícení snímaného objektu a jeho následnému splývání s pozadím. Upravená scéna nenabízela tak úzkou definici v histogramu a spodní část připínáčku v některých případech splývala s podstavou. Z výše uvedených grafů je v

porovnání s ostatními vidět že finální scéna má nejužší a nejvyšší vrchol reprezentace zájmového objektu v histogramu a prakticky veškeré pixely zájmového objektu mají jasovou hodnotu pod 50.

3.4 Tvorba datasetu

V následující podkapitole je vysvětleno, jak probíhalo pořizování datasetu a dále je zde stručně popsán MATLAB script primárně se zaměřením na nastavení kamery.

3.4.1 Proces focení

Bylo pořizeno 1000 kusů připínáčků, 100 kusů bylo pomocí náradí znehodnoceno pro vytvoření NOK kusů. Proces focení probíhal následovně připínáček byl umístěn na podstavu, následně byl spuštěn MATLAB script, který pořídil zpracoval a uložil fotku do adresáře a vyfocený připínáček byl vyměněn za nový. Celkový počet fotek NOK kusů je v datasetu vyšší než jejich fyzický počet, jelikož díky jejich nepravidelnému tvaru bylo možné některé kusy vyfotit při jiné orientaci a vytvořit tak více dat, dataset následně obsahuje i fotky cizích objektů jako např imbusový klíč vrut nebo šroubek.

3.4.2 MATLAB script

MATLAB script je rozdělen do dvou částí, první část se zabývá nastavením ukládacího adresáře a nastavením parametrů kamery, druhá část se věnuje samotnému pořizování fotek ve while smyčce. Celý script DATASET.m se nachází v elektronické příloze.

```
kam = videoinput('winvideo',1);
kam.ReturnedColorSpace = 'grayscale';
triggerconfig(kam,'manual');
kam.FramesPerTrigger = 1;
param = getselectedsource(kam);
param.ExposureMode = 'manual';
param.Exposure = -6;
param.Gain = 260;

kam_side = videoinput('winvideo',2);
kam_side.ReturnedColorSpace = 'grayscale';
triggerconfig(kam_side,'manual');
param_side = getselectedsource(kam_side);
param_side.ExposureMode = 'auto';
param_side.Exposure = -7;
param_side.Gain = 260;
```

Expozici, zesílení a intenzitu osvětlení scény je nutno vybalancovat, obecně chceme mít obraz dobře zaostřený, jelikož se jedná o statickou scénu nejsme nijak ovlivněni maximální délkou expozice. Zesílení tedy můžeme nastavit relativně nízko, aby nedocházelo ke zvýrazňování šumu v obraze, následně můžeme upravovat intenzitu osvětlení tak aby nedocházelo k odrazům, ale zároveň aby snímáný objekt dostatečně vystupoval z pozadí.

```

while 1
    disp('ready')
    pause % cekani na stisk klavesy

    img = getsnapshot(kam); % porizeni snimku
    img = img(90:90+319,120:120+319); % oriznuti pripinacku z obrazu
    img = img(1:2:end,1:2:end);
    img = flipud(img);
    img = flip(img);
    img = imrotate(img, -270);

    img_side = getsnapshot(kam_side); % porizeni snimku
    img_side = img_side(315:315+319,600:600+319); % oriznuti
pripinacku z obrazu
    img_side = img(1:2:end,1:2:end);
    img_side = flipud(img_side);
    img_side = flip(img_side);
    img_side = imrotate(img_side, -270);

    imshowpair(img, img_side, 'montage'); % zobrazeni fotek pro
operatora

    filename_side = "side_"+string(i)+".png"; % vytvoreni nazvu
fotky
    filename_up = "up_"+string(i)+".png"; % vytvoreni nazvu fotky
    file_side = fullfile(dir_side , filename_side); % vytvoreni cesty
pro ulozeni fotky
    file_up = fullfile(dir_up , filename_up); % vytvoreni cesty pro
ulozeni fotky

    imwrite(img, file_side); % ulozeni fotky do cesty
    imwrite(img_side, file_up); % ulozeni fotky do cesty

    i = i+1
    pause(0.1);
end

```

Na prvním řádku smyčky program čeká na stisknutí klávesy operátorem, následně dojde k vyfocení fotky horní kamerou, oříznutí připínáčku z obrazu, podvzorkování a opravy orientace. Následně je obdobně vyfocena, ořezána, podvzorkována a otočena fotka z boční kamery, oba snímky jsou ihned zobrazeny na monitoru pro potvrzení operátorem. Následně jsou fotky automaticky očíslovány a uloženy do předem zadaného adresáře.

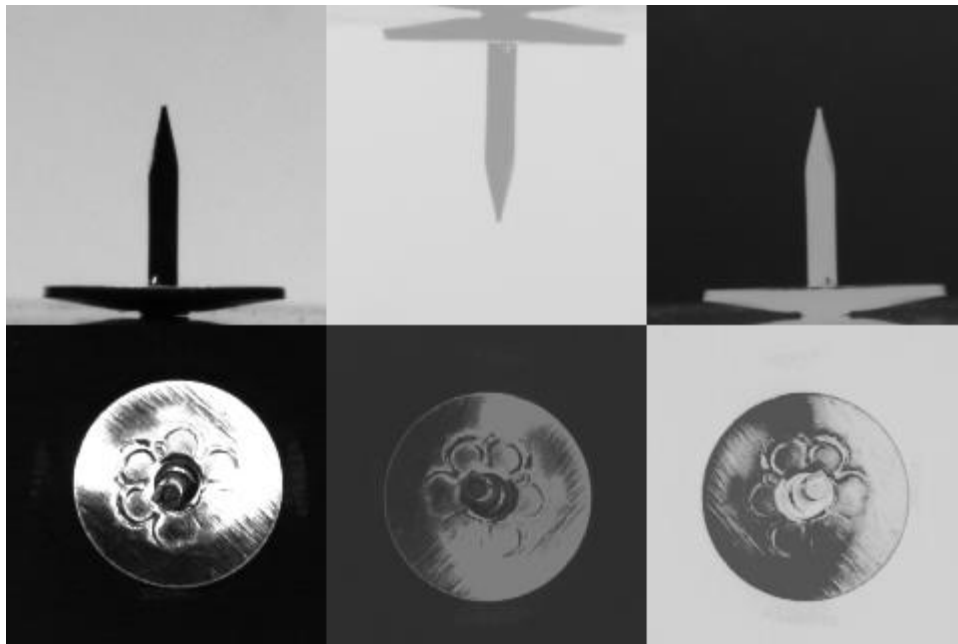
Jako nedostatek tohoto skriptu by se mohl jevit fakt, že připínáček musí být vždy na relativně stejném místě, program má rámeček výřezu z obrazu pevně daný. Lepší možností by bylo, kdyby algoritmus ořezal připínáček z obrazu sám. Pro tuhle aplikaci ale tato funkce nebyla nutná, jelikož je jako přísvit horní kamery použito kruhové světlo. Z toho důvodu je nejlepší, když se jednotlivé připínáčky budou nacházet přibližně na stejném místě. Skutečným nedostatkem je, že se nedá rozhodnout, zda bude vytvořená fotka uložena nebo ne. Po zmáčknutí klávesy a rozběhnutí programu vždy dojde k uložení fotky, takže jsou uloženy i fotky které nejsou validní nebo byly pořízeny omylem, tyto snímky se potom musí manuálně odstranit z adresáře a dojde k nesouladu počtu fotek a indexu.

3.5 Augmentace dat

V následující podkapitole je nastíněn proces augmentace dat pomocí programu v Pythonu.

Každá fotka projde řadou úprav: zrcadlení obrázku po vertikální ose, nastavení náhodného kontrastu, převedení obrázku do negativu, nastavení náhodné saturace, nastavení náhodného jasu, nastavení náhodného zabarvení a náhodné rotování s krokem 90°. Podrobnější popis se nachází v kapitole [4.2.1](#).

Na obrázku 3.19 jsou uvedeny příklady augmentovaných fotek.



Obrázek 3.19 Příklady augmentovaných fotek (první sloupek jsou výchozí data)

Pro účely učení algoritmu bylo z každého obrázku vygenerováno maximálně 1-5 augmentovaných dat.

4. IMPLEMENTACE ALGORITMU

V úvodu této kapitoly je odůvodněn výběr algoritmu a následně popsána reimplementace vybraného algoritmu.

4.1 Výběr algoritmu

Po konzultaci s vedoucím byl jako algoritmus pro tuto práci vybrán konvoluční autonekodér (CAE), implementace tohoto algoritmu byla převzata z práce [24][25].

CAE byl vybrán z několika důvodů. Za prvé, konvoluční vrstvy jsou schopny efektivně extrahovat prostorové vzory z obrázkových dat, což je pro tuto úlohu klíčové. Dále, jsou CAE schopny zachytit hierarchické a složité vzory ve vstupních datech, což je výhodné pro rozpoznávání anomálií. Jejich schopnost redukovat dimenzi dat při zachování důležitých informací je také užitečná pro snížení příznaků na podstatné prvky pro identifikaci anomálií. V praktickém ohledu je výhodou že existuje mnoho volně dostupných implementací, tato konkrétní byla vybrána z důvodu, že na ní stále probíhá výzkum a experimenty provedené v této práci mohou pomoci pracovníkům ze skupiny počítačového vidění.

4.2 Popis reimplementovaného algoritmu

Za účelem ověření funkce klasifikátoru byl v jazyce Python 3.10 reimplementován kód z [24][25] a byly provedeny úpravy, hlavně zjednodušení kódu. Vstupní a výstupní vrstvy byly zachovány ale konvoluční jádro bylo omezeno pouze na dva modely BAE1 a BAE2 [25]. Kompletní kód je spustitelný souborem `run_app.bak` v adresáři projektu, vše se nachází v elektronické příloze.

Následující podkapitoly popisují funkce souborů projektu. HardNet byl převzat z [24] prakticky bez úprav a slouží jako deskriptor příznaků [26].

4.2.1 Popis funkce augmentace dat

V souboru `Aug.py` je implementován program na augmentaci dat. Program je rozdělen do dvou částí, v první části uživatel povolí ukládání a zobrazování, nastaví velikost, ve které se mají obrázky uložit, kolik nových obrázků chce z každé vstupní fotky vytvořit a nastaví cesty pro načtení a uložení obrázků.

V druhé části se nejprve načtou fotky ze zadané cesty a poté už následuje samotná augmentace. Každá fotka projde celou řadou úprav, nejprve se vypočítá náhodná hodnota, která bude sloužit jako seed pro následovné operace. Pořadí operací je následovné: zrcadlení obrázku po vertikální ose, nastavení náhodného kontrastu, převedení obrázku do negativu, nastavení náhodné saturace, nastavení náhodného jasu, nastavení náhodného zabarvení a náhodné rotování fotek s krokem 90°. Operace zrcadlení, negativ a rotace nemusí proběhnout vždy. Celý řetězec proběhne na každé

fotce několikrát, před každým průběhem je vygenerován nový seed, aby se zamezilo vzniku stejných fotek. Následně je fotka uložena nebo zobrazena podle nastavení uživatelem. Po vytvoření požadovaného počtu fotek program začne augmentovat další fotky z adresáře. Po vygenerování dat ze všech fotek dojde k ukončení programu. Příklady augmentovaných dat jsou v kapitole [3.5](#). Celý program se nachází v elektronické příloze.

4.2.2 Popis funkce uživatelského rozhraní

Pro přenos dat mezi uživatelským rozhraním a hlavním programem byla v souboru `Data_Container.py` implementována třída `DataContainer`, třída obsahuje všechny proměnné, které uživatel nastavuje v rozhraní. Program si pak tyto proměnné přebere a pracuje s nimi dál.

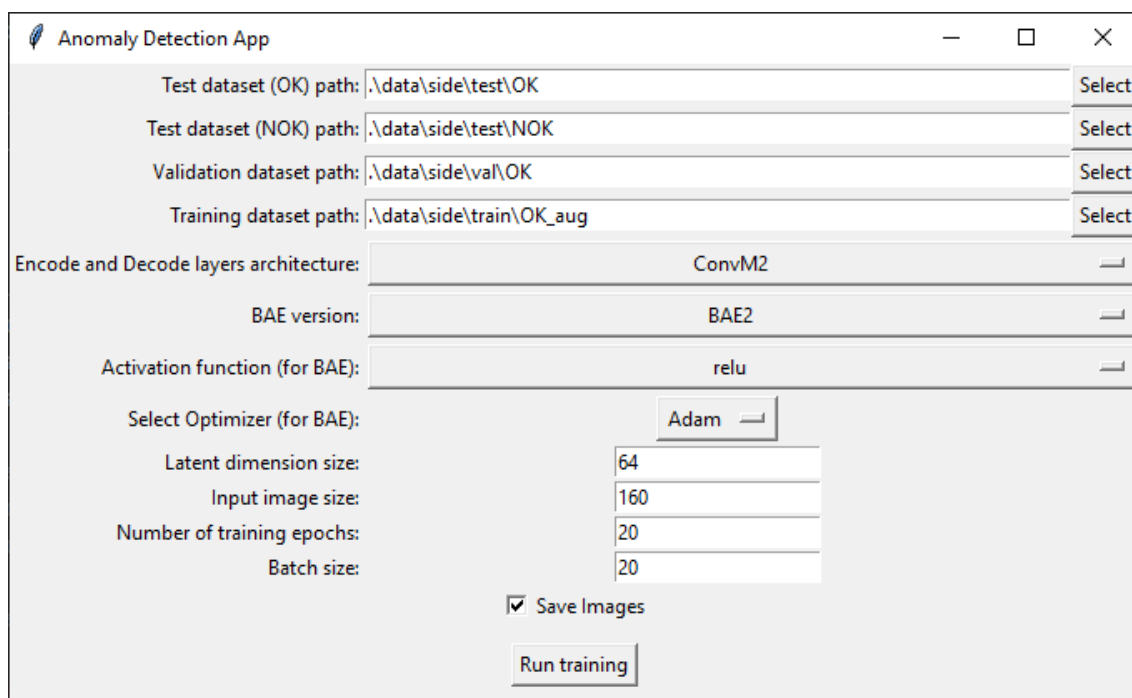
V souboru `AE_GUI.py` je implementováno grafické rozhraní pro práci s programem. Program je rozdělen do 4 částí:

V první části se inicializuje datový kontejner.

V druhé části jsou vytvořeny zadávací pole pro testovací OK a NOK, validační a trénovací datasey s přednastavenými cestami. Dále je zde možnost výběru architektury sítě kodéru a dekodéru `ConvM1...ConvM6` dle [24], výběr BAE verze dle [24], výběr aktivační funkce pro konvoluční jádro BAE, výběr optimizéru pro učení BAE, nastavení velikosti latentní dimenze CAE, nastavení rozměru, na který budou vstupní data převedena, dále počet tréninkových epoch, batch size (velikost tréninkové dávky), možnost uložení vstupních testovacích dat podle rozřazení v matici záměn a tlačítko pro spuštění učení.

Ve třetí části jsou po stisku tlačítka pro zahájení učení ze všech proměnných přeneseny hodnoty do datového kontejneru a grafické rozhraní je ukončeno.

Čtvrtá část se věnuje testování samotného rozhraní a není při běžném používání aktivní.



Obrázek 4.1 Uživatelské rozhraní s nastavenými parametry

4.2.3 Popis funkce hlavního programu

V souboru `main.py` je implementován hlavní program pro tvorbu, učení a použití CAE ke klasifikaci vstupních obrázků, celý kód je rozdělen na dvě hlavní části:

V první části jsou naimportovány veškeré knihovny a následuje popis funkcí.

`load_imgs(path)`: funkce načte obrázek ze zadané cesty a přeškáluje jasovou hodnotu na 0-1, vrací normalizovaná data.

`normalize2DData(data)`: normalizuje data pro tensor flow

`computeMetrics(org_data, org_labels, dec_data, a_batch_size)`: z `org_data` a `dec_data` vypočítá rozdíl a po dávkách je posílá do klasifikátoru HardNet. Vrací metriky a labels (značky).

`getEvaluationMetrics(scores, name, ax, labels)`: pomocí značek a skór vypočítá precision a recall, vykreslí ROC křivky s hodnotami AUC, a vrátí optimální hraniční hodnotu pro klasifikaci, pojmy jsou vysvětleny v kapitole [1.2.1](#)

`getConfusionMatrix(labels, labelsPred, name, ax)`: pomocí labels a predikované hodnoty `labelsPred` vypočítá precision, recall a F1 skóre, tyto hodnoty následně vypíše do konzole a do okna grafu vykreslí matici záměn, je volána ve funkci `dataClassify`.

`dataClassify(metrics, labels)`: určí optimální thresholdy a klasifikuje obrázky čtyřmi různými klasifikátory, pro každý vykreslí ROC křivku s AUC hodnotou a matici záměn pomocí funkce `getConfusionMatrix`. Funkce vrací predikované značky dat dle všech klasifikátorů, optimální thresholdy pro každý klasifikátor a hodnotu AUC všech klasifikátorů.

`visualiseTrainResults(trainHistory, typeAE, typeLayer)`: funkce vykreslí grafy průběhů tréninkové a validační ztráty CAE.

`visualiseEncDecResults(actStr, orgData, encData, decData, imIndxList, typeAE, typeLayer)`: funkce vykreslí příklady výchozích, zakódovaných, dekodovaných a rozdílových dat pro typ enkoderu zadaný uživatelem

`fsVisualise(metrics, labels, typeAE, typeLayer)`: funkce vypočítá příslušné metriky a vykreslí příznakový prostor pomocí t-SNE a PCA. [24]

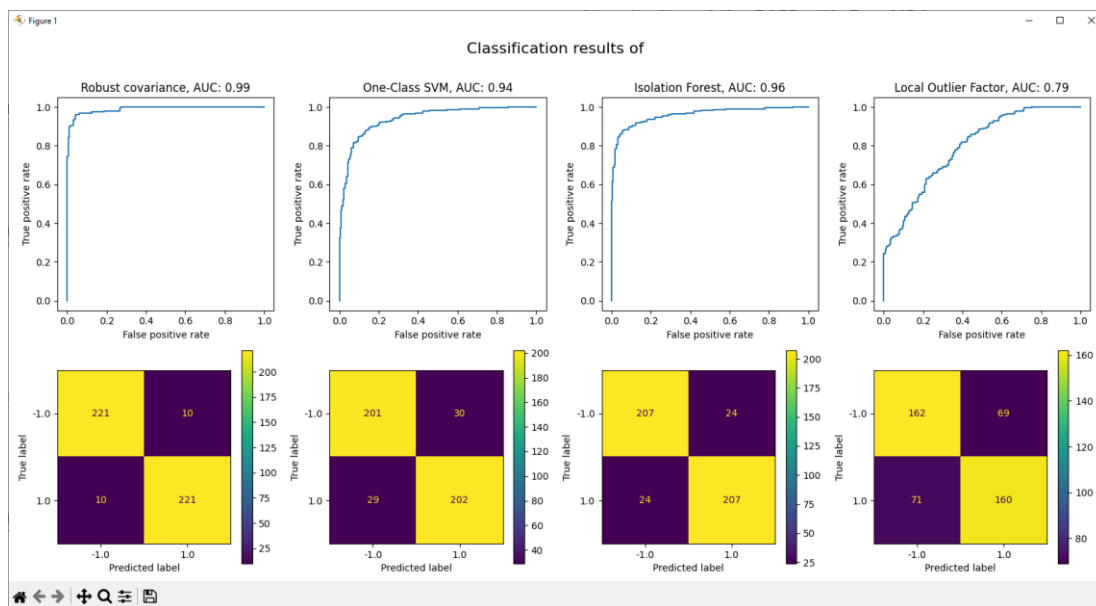
`encode(enc_input_img, name='ConvM2')`: funkce zakóduje vstupní obrázky `enc_input_img` pomocí jedné z přednastavených architektur enkoderů. Funkce vrací zakódované obrázky, počet filtrů jednotlivých architektur a výšku a šířku zakódovaných obrázků.

`decode(dec_input_img, filterCount, name='ConvM2', imChannel=1)`: funkce rozkóduje vstupní obrázky `dec_input_img` pomocí jedné z přednastavených architektur dekodéru a počtu filtrů. Funkce vrací rozkódované obrázky.

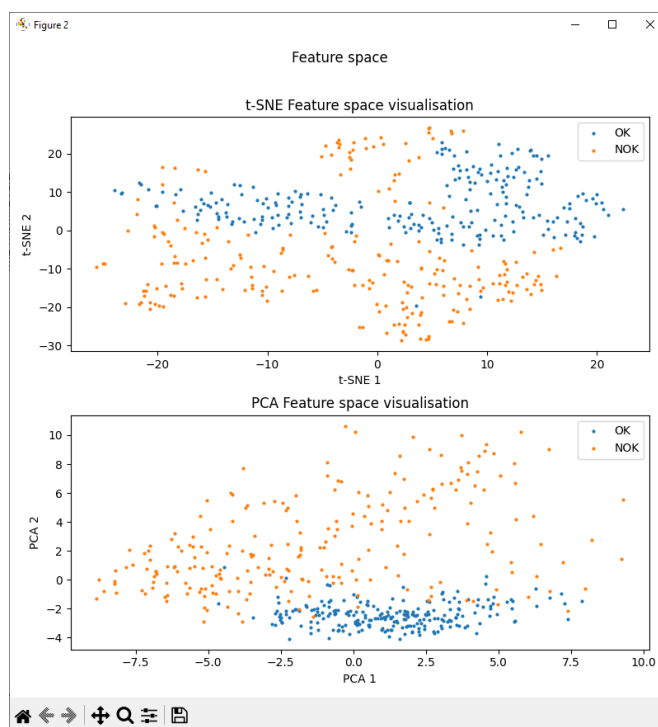
`build_bae_model(version, input_img, latentDim, name, activ, opt)`: funkce nastaví model CAE podle uživatelem vybrané verze (BAE1, BAE2), velikostí latentní dimenze, architektury kodéru a dekodéru, aktivační funkce konvolučního jádra a optimizéru učení [1.4.1](#). Funkce vrací model CAE.

`saveClassifiedImages(save_images_in, test_images_in, test_images_labels_in, y_pred_in, scores_roc_auc_in)`: funkce pro ukládání klasifikovaných obrázků, podle RUC křivky se vybere nejlepší klasifikátor a obrázky jsou podle jeho predikce OK a NOK kusů rozřazeny do složek, představující indexy matice záměn (TP, TN...), složky s rozřazenými snímky lze najít v adresáři projektu/data/classification_results. Flag `save_images` nastavený uživatelem, povoluje nebo zakazuje ukládání souborů.

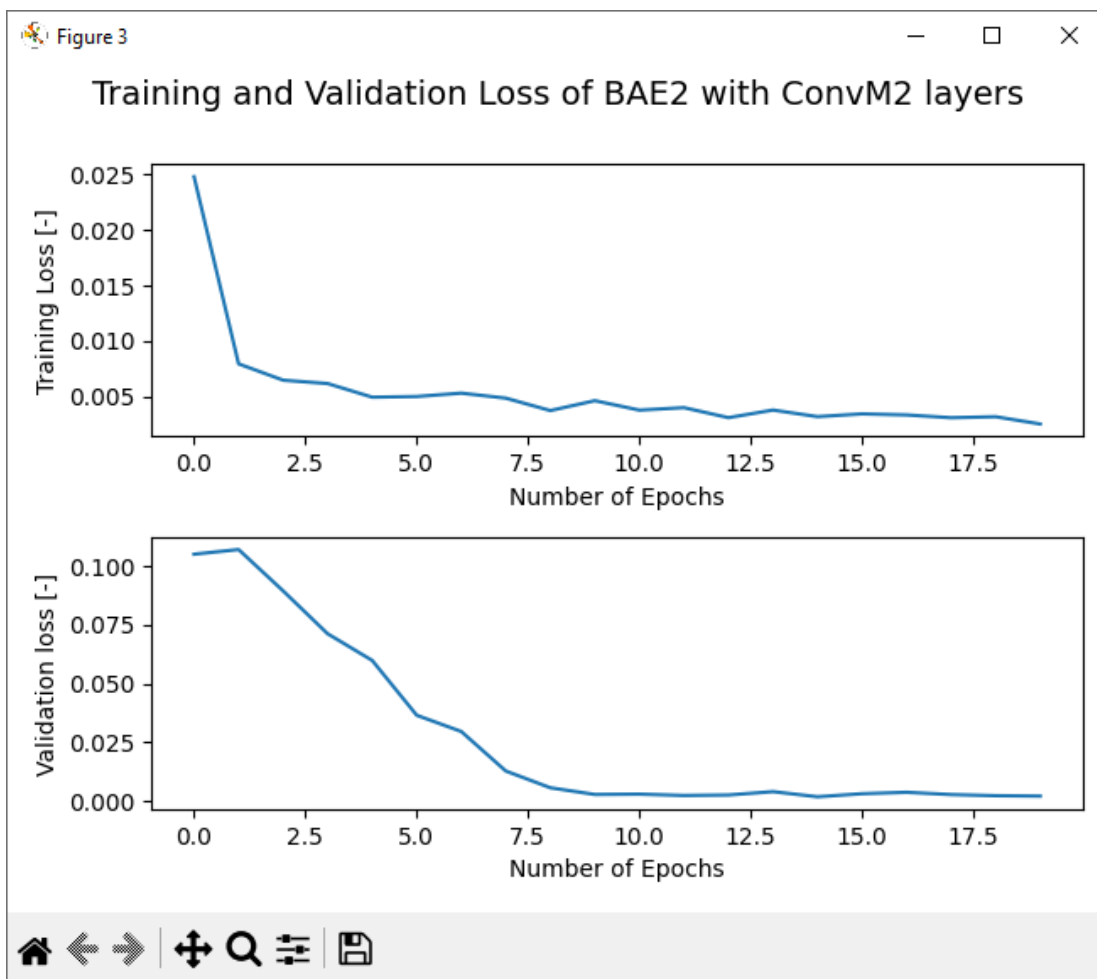
V druhé části už následuje samotný program, kde se na začátku inicializuje kontejner pro přenos dat z grafického rozhraní, následně je vyvoláno grafické rozhraní, kam uživatel zadá parametry, následuje předání parametrů do hlavního programu. Po předání parametrů jsou sestaveny cesty k datasetům a fotky jsou nahrány programu, dále dojde k anotaci datasetů a jejich náhodnému zamíchání, převedení velikosti vstupních dat na uživatelem zadané rozlišení. Potom následuje inicializace CAE s uživatelem nastavenými parametry, inicializace enkodéru dle nastavení CAE, kódování dat pro pozdější vykreslení, samotné učení CAE s parametry nastavenými uživatelem, generace výstupních dat, výpočet metrik a značek pro označení datasetu, klasifikace dat, vizualizaci příznakového prostoru, vizualizace průběhu učení, zobrazení testovacích, kódovaných, dekodovaných dat a rozdílu mezi testovacími a dekodovanými daty, uložení klasifikovaných obrázků do složek podle matice záměn, výpis uživatelem nastavených parametrů a vykreslení příkladů testovacích a jim odpovídajících dekodovaných dat s anotací. K ukončení programu dojde po zavření všech oken s výsledky učení. Na 4.2, 4.3, 4.4 a 4.5 jsou uvedeny příklady vizualizace průběhu a výsledků učení.



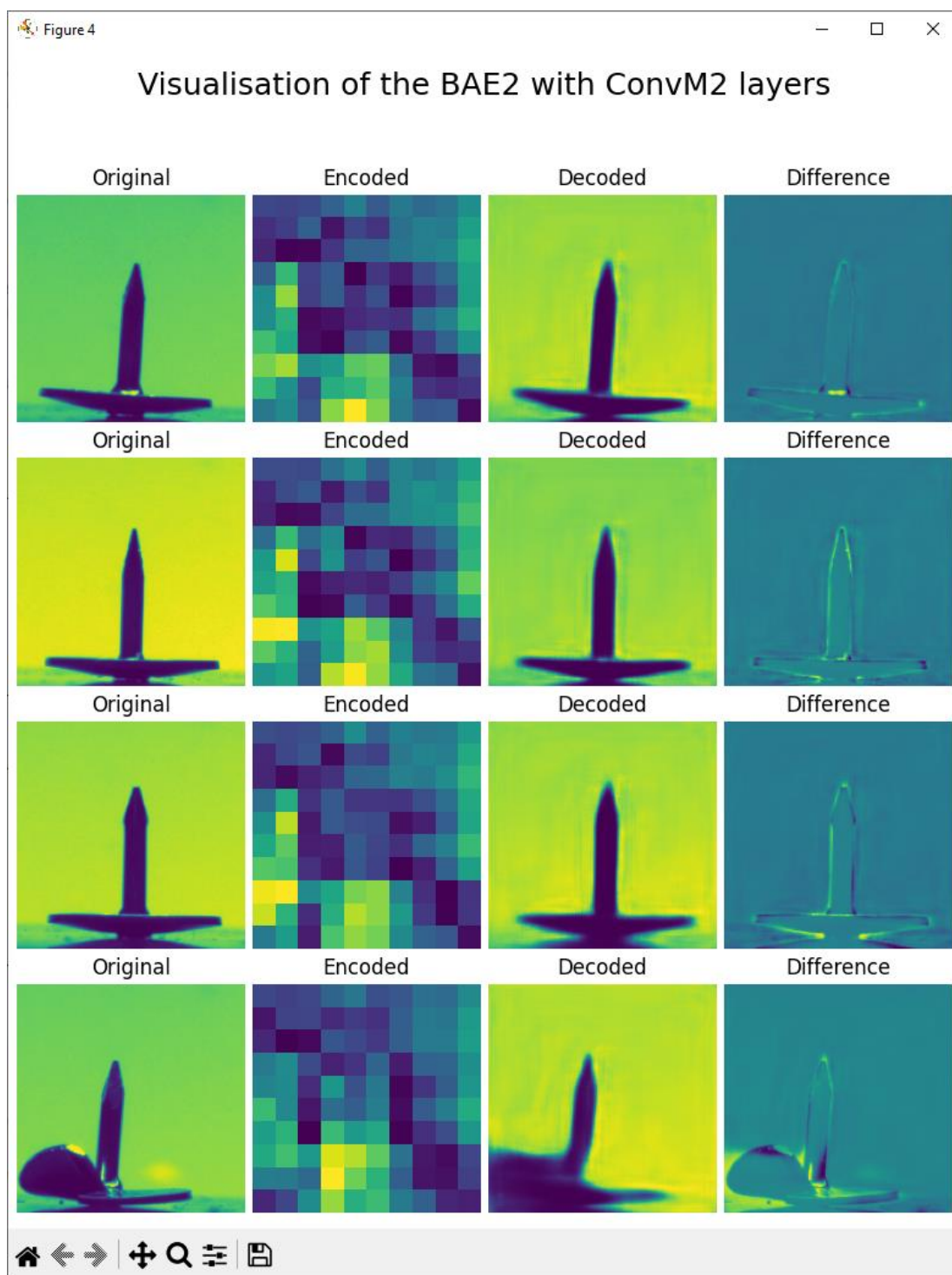
Obrázek 4.2 Příklad vizualizace výsledků klasifikace pro různé klasifikátory



Obrázek 4.3 Příklad vizualizace příznakového prostoru pomocí t-SNE a PCA.



Obrázek 4.4 Příklad vizualizace výsledků klasifikace pro různé klasifikátory



Obrázek 4.5 Příklad vizualizace originálních, zakódovaných, dekódovaných a rozdílových dat

5. SADA EXPERIMENTŮ

Tato kapitola se věnuje popisu experimentů provedených na AE, v kapitole [5.1](#) se experimenty věnují různému nastavení hyperparametrů, na AE s nejlepšími nalezenými parametry byly následně provedeny další experimenty věnující se cílenému zavedení chyb do datasetu v kapitole [5.2](#), experimenty věnující se výpočtení náročnosti v kapitole [5.3](#) a redukci datasetu v kapitole [5.4](#). Protože se práce nevěnuje fúzi dat z obou kamer a jelikož se data z horního pohledu ukázaly jako málo deskriptivní zabývají se následující experimenty primárně fotkami z bočního pohledu.

5.1 Různé nastavení hyperparametrů

V úvodu podkapitoly jsou uvedeny a vysvětleny hyperparametry, které je možné pro proces učení měnit, následně jsou popsány experimenty prováděné s různým nastavením hyperparametrů.

5.1.1 Nastavitelné hyperparametry

Architektura enkoderu a dekoderu: výběr z přednastavených architektur ConvM1- struktura z příkladu z knihovny keras, ConvM2-struktura MVTec podle [27], ConvM3- základní struktura, ConvM4-druhá verze základní struktury, ConvM5-základní struktura z příkladu z knihovny keras, ConvM6-asymetrický encoder a decoder, kombinace ConvM5 a convM4. Konkrétní struktury všech variant jsou v programu v elektronické příloze nebo [24].

Tabulka 5.1 Vlastností architektur dekoderu

Struktura	Velikost výstupního obrázku	Vsrtvy	Kernel	Stride	Padding
ConvM1	input_size/32	5	3x3	2	1
ConvM2	input_size/16	9	3x3, 4x4, 8x8	1, 2	1
ConvM3	input_size/4	2	3x3	2	1
ConvM4	input_size/4	2	3x3, 5x5	0	1
ConvM5	input_size/8	3	3x3	0	1

BAE verze: verze konvolučního jádra BAE1 – základní model autoenkoderu, BAE2 – základní modem autoenkodéru s plně propojenými vrstvami před enkodováním [24].

Aktivační funkce: výběr z možností aktivačních funkcí, ReLU, Sigmoid, Tanh a Leaky ReLU.

Velikost latentní dimenze: rozměr prostoru, ve kterém jsou reprezentovány zakódované proměnné, které model vytváří ze vstupních dat. Větší velikost latentního prostoru může poskytnout modelu větší možnost flexibility a kapacity pro reprezentaci

dat, ale může vést k přeučení na daný trénovací dataset. Naopak malý rozměr latentního prostoru povede na jednodušší reprezentaci vstupního obrazu, která může být méně flexibilní, ale může pomoci zamezení přeučení a lepší generalizaci modelu.

Velikost vstupního obrazu: udává na jakou velikost mají být vstupní snímky převedeny programem. Redukcí rozměrů vstupních snímků dochází ke ztrátě informace, ale naopak vede k rychlejšímu učení, jelikož algoritmus nemusí zpracovávat tolik dat.

Počet trénovacích epoch: určuje kolikrát bude opakováno učení na celém tréninkovém datasetu.

Velikost dávky (Batch size): určuje po jakém počtu budou tréninková data vkládána do CAE v průběhu učení, menší velikost dávky by teoreticky měla vést k rychlejší adaptaci, jelikož po každé dávce dojde k adaptaci vah, ale může vznikat problém kdy se síť přeučí na tento malý dataset.

Optimizér: výběr z možných optimizérů, algoritmů, které upravují vlastnosti sítí jako jsou váhy a koeficienty učení s cílem snížit chybovost sítě. Optimizéry jsou blíže popsány v kapitole [1.4](#).

5.1.2 Experimenty s hyperparametry

Jako výchozí nastavení celé architektury CAE byla zvolena kombinace enkoderu a dekoderu ConvM2, která ukázala nejlepší výsledky [25], konvolučního jádra verze BAE2, jelikož jde o strukturu implementovanou pracovníky skupiny počítačového vidění, velikost latentní dimenze byla nastavena na 32, velikost vstupních obrázků byla ponechána na velikosti fotek v datasetu (160x160 pixelů), počet trénovacích epoch byl nastaven na 10, velikost dávky byla nastavena na 32 a ostatní parametry na výchozí nastavení dle [24]. Trénovací dataset obsahoval celkově 2238 OK fotek, 373 pořízených a 1865 augmentovaných, tedy v poměru 1:5, validační dataset obsahoval 400 OK kusů a testovací dataset obsahoval 231 OK a 231 NOK kusů

5.2 Dělení datasetů

Nastavení hyperparametrů bylo zvoleno jako nejlepší nalezené z kapitoly [6.1](#). Následně byly provedeny experimenty s učením a klasifikací CAE pro různé poměry objemu dat v trénovacím, validačním a testovacím datasetu. Primárně šlo o změny poměrů mezi trénovacím a validačním datasetem, počet dat v testovacím datasetu zůstal stejný (231 OK a 231 NOK kusů).

5.3 NOK kusy v tréninkovém datasetu

Za účelem ověření nereprezentativnosti datasetu byl do trénovacího datasetu přenesen vždy zvyšující se počet náhodných snímků z testovacího NOK datasetu, následně byla provedena augmentace a poté proběhl proces učení. Hyperparametry CAE zůstaly stejné

jako nejlepší nalezené v experimentu z kapitoly [6.1](#) a dataset byl rozdělen dle nejlepšího poměru nalezeného v experimentu z kapitoly [6.2](#).

5.4 Redukce datasetu

Předposledním z řady provedených experimentů byl test redukce datasetu, velikosti trénovacího a validačního datasetu byly metodou půlení intervalu postupně a se stejným nastavením hyperparametrů bylo provedeno učení algoritmu. Experiment ověřuje mohutnost datasetu nutnou pro dosažení dostatečné klasifikační úrovně.

5.5 Časová náročnost algoritmu

V posledním experimentu provedeno trénování s velmi krátkou a relativně velmi dlouhou dobou učení. Účelem tohoto experimentu bylo zjistit, jak krátká doba stačí pro kvalitní výsledky klasifikace a jak silný dopad bude mít přeučení algoritmu na tréninkový dataset.

6. ZHODNOCENÍ EXPERIMENTŮ

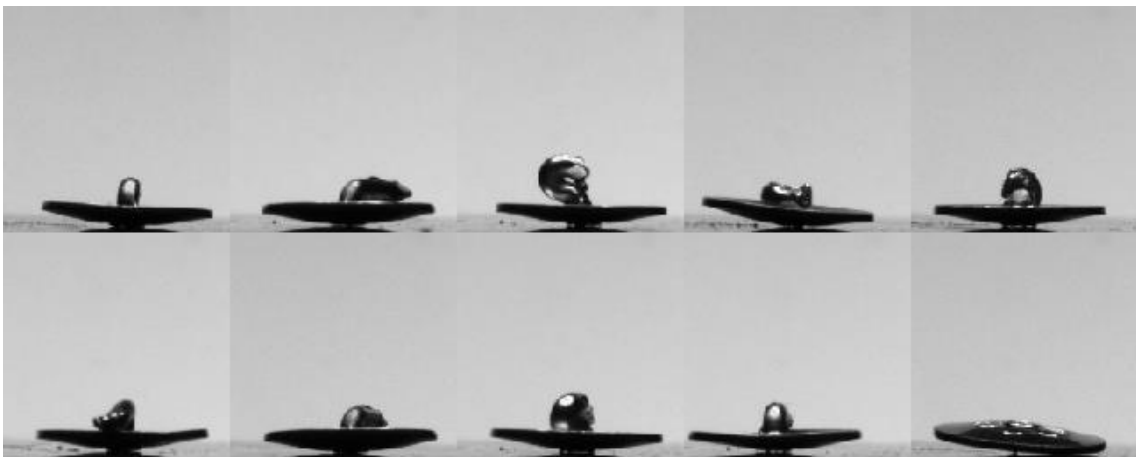
V následující kapitole je popsán průběh experimentů prováděných na CAE. CAE byl učen pouze na CPU a hardware počítače byl: procesor AMD Ryzen 7 1700, grafická karta NVIDIA GeForce GTX 1070 Ti, SSD disk Kingston SA400S37240G, základní deska Gigabyte AB350-Gaming 3.

6.1 Nalezení optimálních hyperparametrů

V následující kapitole je popsán průběh experimentu pro určení optimální kombinace hyperparametrů pro vytvoření nejlepšího klasifikátoru.

6.1.1 Volba rozměru obrázku

Pro vybranou architekturu BAE2+ConvM3 s velikostí latentní dimenze 32, počtem epoch 10 a velikosti dávky 32 byla experimentálně určena velikost vstupního obrazu. Rozměr obrázku byl vybrán v závislosti na základě precision, recall a F1 skóre. Klasifikátory s velmi podobným skóre byly srovnány podle typu FP anomálií. Pro většinu architektur totiž představoval problém jeden specifický typ anomálie, kdy připínáčku chyběl nebo byl poškozen hrot, jak ukazuje obrázek 6.1. Dělení datasetu bylo následovné: trénovací dataset obsahoval celkově 2238 OK fotek, 373 pořízených a 1865 augmentovaných, validační dataset obsahoval 400 OK kusů a testovací dataset obsahoval 231 OK a 231 NOK kusů.



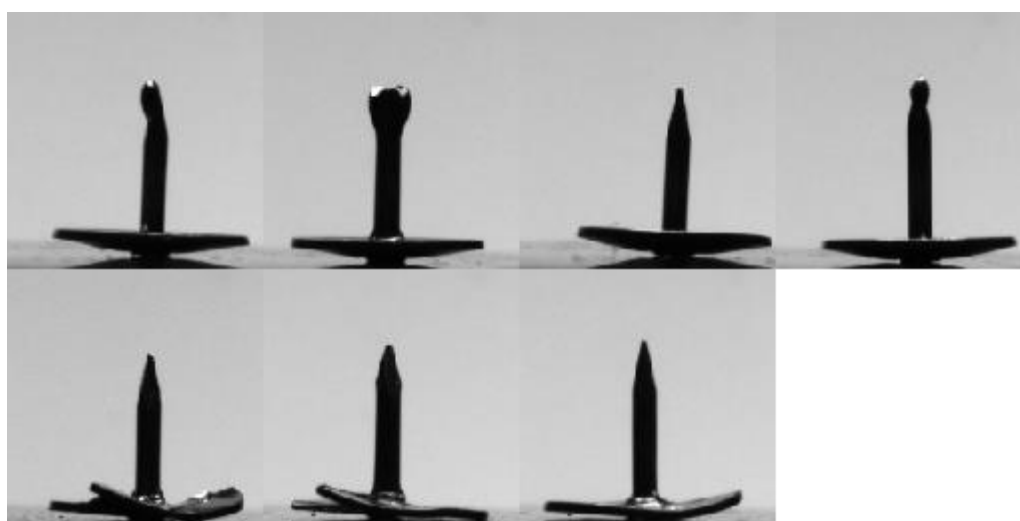
Obrázek 6.1 Ukázka specifického typu anomálie

Následující tabulka ukazuje výsledky různých architektur kodéru a dekodéru v závislosti na velikosti obrázků.

Tabulka 6.1 Výpis výsledků pro různé architektury cod/dec a velikosti vstupních obrázků (BAE2, LD32)

Verze jádra:	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2
Aktivační funkce	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Optimizér:	Adam	Adam	Adam	Adam	Adam	Adam
Velikost latentní dimenze:	32	32	32	32	32	32
Rozměr obrázků	8	16	32	64	128	160
Počet epoch	10	10	10	10	10	10
Velikost dávky	32	32	32	32	32	32
ConvM1	Klasifikátor: Robust covariance					
Precision	-	-	0.9391	0.9481	0.9394	0.9356
Recall	-	-	0.9351	0.9481	0.9394	0.9437
F1 skóre	-	-	0.9371	0.9481	0.9394	0.9397
ConvM2	Klasifikátor: Robust covariance					
Precision	-	0.9091	0.9610	0.9149	0.9394	0.9397
Recall	-	0.9091	0.9610	0.9307	0.9394	0.9437
F1 skóre	-	0.9091	0.9610	0.9227	0.9394	0.9417
ConvM3	Klasifikátor: Robust covariance					
Precision	0.7922	0.9221	0.9351	0.9520	0.9870	0.9569
Recall	0.7922	0.9221	0.9351	0.9437	0.9827	0.9610
F1 skóre	0.7922	0.9221	0.9351	0.9478	0.9848	0.9590
ConvM4	Klasifikátor: Robust covariance					
Precision	0.8874	0.9356	0.9435	0.9478	0.9610	0.9697
Recall	0.8874	0.9437	0.9394	0.9437	0.9610	0.9697
F1 skóre	0.8874	0.9397	0.9414	0.9458	0.9610	0.9697
ConvM5	Klasifikátor: Robust covariance					
Precision	0.8701	0.9432	0.9610	0.9610	0.9694	0.9437
Recall	0.8701	0.9351	0.9610	0.9610	0.9610	0.9437
F1 skóre	0.8701	0.9391	0.9610	0.9610	0.9652	0.9437
ConvM6	Klasifikátor: Robust covariance					
Precision	0.8783	0.9651	0.9697	0.9610	0.9351	0.9261
Recall	0.8745	0.9567	0.9697	0.9610	0.9351	0.9221
F1 skóre	0.8764	0.9609	0.9697	0.9610	0.9351	0.9241

Všechny zvýrazněné klasifikátory byly naučeny ještě znovu pro ověření správnosti výsledků. Jako nejlepší se ukázala architektura Conv3 spolu s rozměry obrázku 128x128 pixlů, která dosáhla pouhých 4 FN vzorků a 3 FP vzorků. Následující obrázek ukazuje FN a FP vzorky tohoto algoritmu.



Obrázek 6.2 Mylně zaklasifikované vzorky (CAE BAE2, Conv3, LD 64, size 128, FN vzorky nahoře FP vzorky dole)

6.1.2 Volba velikosti latentní dimenze

Pro vybranou architekturu BAE2+ConvM3 a velikost vstupního obrazu 128x128 pixelů, počtu epoch 10 a velikosti dávky 32 byla experimentálně určena optimální velikosti latentní dimenze. Následující tabulka ukazuje výsledky pro násobky 2 velikosti latentní dimenze. Dělení datasetu bylo následovné: trénovací dataset obsahoval celkově 2238 OK fotek, 373 pořízených a 1865 augmentovaných, validační dataset obsahoval 400 OK kusů a testovací dataset obsahoval 231 OK a 231 NOK kusů.

Tabulka 6.2 Výpis výsledků pro různé nastavení velikosti latentní dimenze (ConvM3, BAE2, 128px)

Struktura ENC/DEC	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3	Conv M3
BAE verze	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2
Aktivační funkce	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Optimizér	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
Velikost latentní dimenze	2	4	8	16	32	64	80	128	256	384
Rozměr obrázků	128	128	128	128	128	128	128	128	128	128
Počet epoch	10	10	10	10	10	10	10	10	10	10
Velikost dávky	32	32	32	32	32	32	32	32	32	32
Robust covariance										
Precision	0.948	0.969	0.961	0.974	0.965	0.969	0.974	0.969	0.965	0.960
Recall	0.956	0.969	0.965	0.974	0.969	0.969	0.978	0.974	0.965	0.952
F1 skóre	0.952	0.969	0.963	0.974	0.967	0.969	0.976	0.971	0.965	0.956

Počet FP specifického typu	4	1	2	3	0	0	0	0	2	3
----------------------------	---	---	---	---	---	---	---	---	---	---

Na intervalu velikosti latentní dimenze 32-128 se rozdíly ukázaly velmi malé.

Experimentálně byl ověřen rozměr 80, který se ukázal jako nejlepší, algoritmy byly hodnoceny podle metrik precision, recall a F1 skóre nebo počet specifických anomálií ve FP výsledcích.

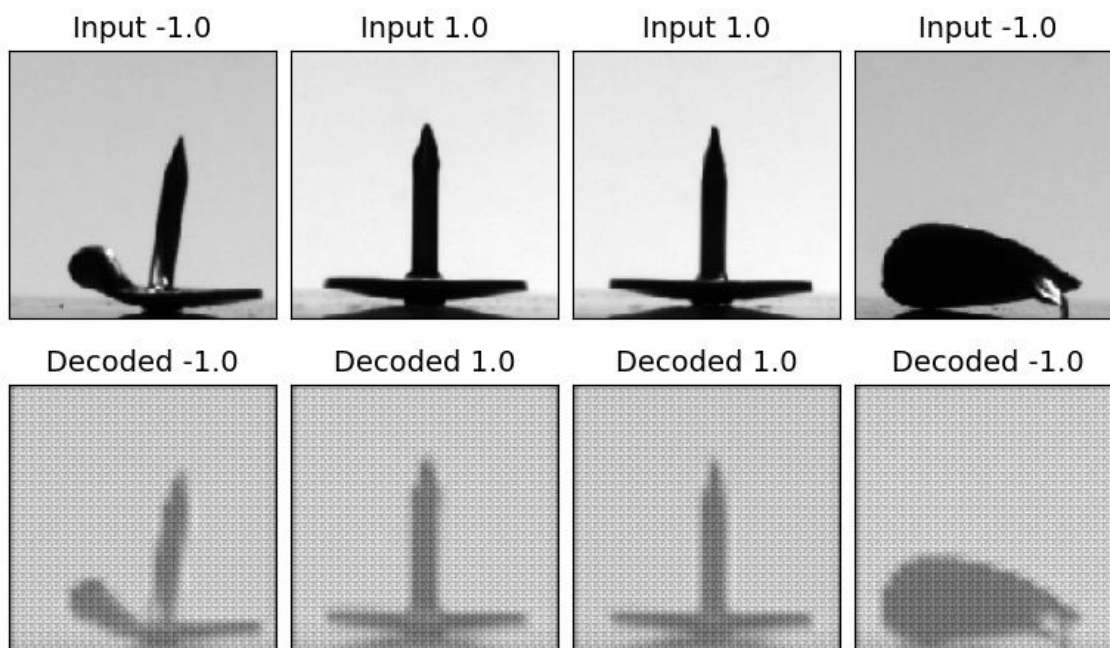
6.1.3 Volba velikosti dávky

Pro vybranou architekturu BAE2+ConvM3 s velikostí latentní dimenze 80 a velikost vstupního obrazu 128x128 pixelů a počtu epoch 10 byla experimentálně určena optimální velikost dávky. Následující tabulka ukazuje výsledky pro různé hodnoty hyperparametru. Dělení datasetu bylo následovné: trénovací dataset obsahoval celkově 2238 OK fotek, 373 pořízených a 1865 augmentovaných, validační dataset obsahoval 400 OK kusů a testovací dataset obsahoval 231 OK a 231 NOK kusů.

Tabulka 6.3 Výpis výsledků pro různé nastavení velikosti dávky (ConvM3, BAE2, LD80, 128px)

Struktura ENC/DEC	ConvM3	ConvM3	ConvM3	ConvM3	ConvM3	ConvM3	ConvM3	ConvM3
BAE verze	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2	BAE2
Aktivační funkce	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Optimizér	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
Velikost latentní dimenze	80	80	80	80	80	80	80	80
Rozměr obrázků	128	128	128	128	128	128	128	128
Počet epoch	10	10	10	10	10	10	10	10
Velikost dávky	5	10	20	32	40	60	100	200
Robust covariance								
Precision	0.9567	0.8836	0.9528	0.9741	0.9828	0.9827	0.9836	0.9784
Recall	0.9567	0.8874	0.9610	0.9784	0.9870	0.9827	0.9784	0.9784
F1 skóre	0.9567	0.8855	0.9569	0.9762	0.9849	0.9827	0.9805	0.9784
Počet FP specifického typu	6	11	3	0	0	1	0	1

Spolu se zvyšující se velikostí dávky mírně vzrostl také čas jednotlivých epoch z ~30 s/epochu při velikosti dávky 32 až na ~42 s/epochu při velikosti 200. Pro velikost dávky větší 40 se v dekódovaném obraze začaly rozbízet vzory, jak ukazuje obrázek 6.3 a kvalita klasifikace zůstávala prakticky stejná. Z toho důvodu byla ponechána výchozí hodnota 32.



Obrázek 6.3 Vzory v dekódovaném obraze pro velkou dávku (CAE BAE2, Conv3, LD 80, size 128, vstupní obrázky nahoře, dekódované dole)

Jako nejlepší otestované nastavení architektury CAE byla kombinace konvolučního jádra BAE2, architektury enkoderu a dekoderu ConvM3, velikosti latentní dimenze 80, velikosti vstupních obrázků 128x128 pixelů a velikosti dávky byla na 40, ostatní parametry byly ponechány na výchozím nastavení dle [24].

6.2 Poměry dělení datasetů

V následujícím experimentu byl pro vybranou architekturu BAE2+ConvM3 s velikostí latentní dimenze 80 a velikost vstupního obrazu 128x128 pixelů, počtu epoch 10 a velikostí dávky 40 zkoumán vliv poměrů testovacího a validačního datasetu na kvalitu klasifikace. Data byla postupně převáděna z validačního do trénovacího datasetu a bylo měněno i množství augmentovaných dat.

Tabulka 6.4 Výpis výsledků pro různé poměry trénovacího a validačního datasetu (ConvM3, BAE2, LD80, 128px, dávka 40)

Struktura ENC/DEC	ConvM3			
BAE verze	BAE2			
Aktivační funkce	ReLU			
Optimizér	Adam			
Rozměr obrázků	128			
Velikost trénovacího datasetu	373	946	1719	2692

Poměr augmentovaných dat	1:0	1:1	1:2	1:3
Velikost validačního datasetu	400	300	200	100
Robust covariance				
Precision	0.9698	0.9740	0.9784	0.9828
Recall	0.9740	0.9740	0.9784	0.9870
F1 skóre	0.9719	0.9740	0.9784	0.9849
Počet FP specifického typu	1	0	0	0

Při zvyšování objemu datasetu docházelo ke velmi mírnému zlepšení kvality klasifikace.

6.3 Robustnost vůči špatné anotaci

V následujícím experimentu byl pro vybranou architekturu BAE2+ConvM3 s velikostí latentní dimenze 80 a velikost vstupního obrazu 128x128 pixelů, počtu epoch 10 a velikostí dávky 40 do trénovacího datasetu přesunut postupně větší počet NOK kusů za účelem vyhodnocení robustnosti algoritmu vůči špatné anotaci. Na zamíchaném datasetu byla vždy provedena augmentace tak.

Tabulka 6.5 Výsledky zavedení chyby do datasetu (CAE BAE2, Conv3, LD 80, size 128,10 epoch, velikost dávky 32)

Struktura ENC/DEC	ConvM3			
BAE verze	BAE2			
Aktivační funkce	ReLU			
Velikost latentní dimenze	80			
Rozměr obrázků	128			
Počet epoch	10			
Velikost dávky	40			
Počet NOK kusů	100	231	231	231
Počet OK kusů	473	473	231	115
Poměr augmentovaných dat	1:3	1:2	1:3	1:3
poměr NOK a OK kusů	0,175	0,328	0,5	0,66763
Robust covariance				
Precision	0.9870	0.9696	0.9870	0.9827
Recall	0.9870	0.9654	0.9870	0.9827
F1 skóre	0.9870	0.9675	0.9870	0.9827

Výsledky tohoto experimentu nevyšly podle očekávání, chyba je pravděpodobně způsobena nedostatečným počtem anomálií, pravděpodobně dochází ke vzniku značné resubstitační chyby, jelikož byly animálie do trénovacího datasetu kopírovány.

6.4 Redukce datasetů

V následujícím experimentu byl pro vybranou architekturu BAE2+ConvM3 s velikostí latentní dimenze 80 a velikost vstupního obrazu 128x128 pixelů, počtu epoch 10 a velikostí dávky 40. Velikost trénovacího a validačního datasetu byla postupně snižována metodou půlení intervalů až na minimální velikost, při které, ponaučení CAE ještě došlo ke kvalitním výsledkům klasifikace.

Tabulka 6.6 Výpis výsledků pro různé nastavení velikosti vstupních obrázků (ConvM1, BAE2, LD32)

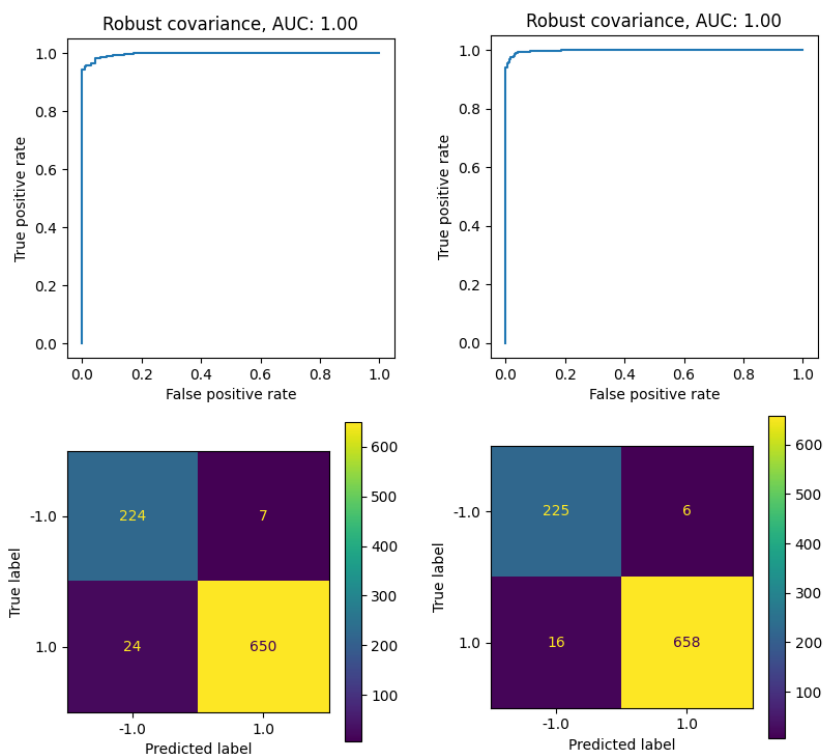
Struktura ENC/DEC:	ConvM3							
BAE verze:	BAE2							
Velikost latentní dimenze:	80							
Rozměr obrázků	128							
Počet epoch	10							
Velikost dávky	32							
Velikost trénovacího datasetu	373	373	373	280	140	-	70	35
Poměr augmentovaných dat	1:5	1:2	1:1	-	-	-	-	-
Velikost validačního datasetu	400	200	100	50	25	-	12	6
Robust covariance								
Precision	0.9785	0.9784	0.9740	0.9698	0.9698	FP	8	9
Recall	0.9870	0.9784	0.9740	0.9740	0.9740	FN	32	39
F1 skóre	0.9828	0.9784	0.9740	0.9719	0.9719	-	-	-

Průběh doby učení odpovídal očekávání, délka učení se pro každou iteraci zhruba půlila. Pro třetí iteraci byla část augmentovaných fotek smazána, aby bylo dosaženo požadovaného počtu. Pro velikosti testovacího datasetu 70 a 12 validačního datasetu byl zbytek OK fotek přesut do testovacího OK datasetu a byla změněna metrika hodnocení, jelikož poměry už nebyly přenositelné na předchozí měření. Experiment byl ukončen na velikosti testovacího datasetu 35 a trénovacího datasetu 6, dá se říci že mohutnost datasetu je velmi značná.

6.5 Náchylnost na přeučení

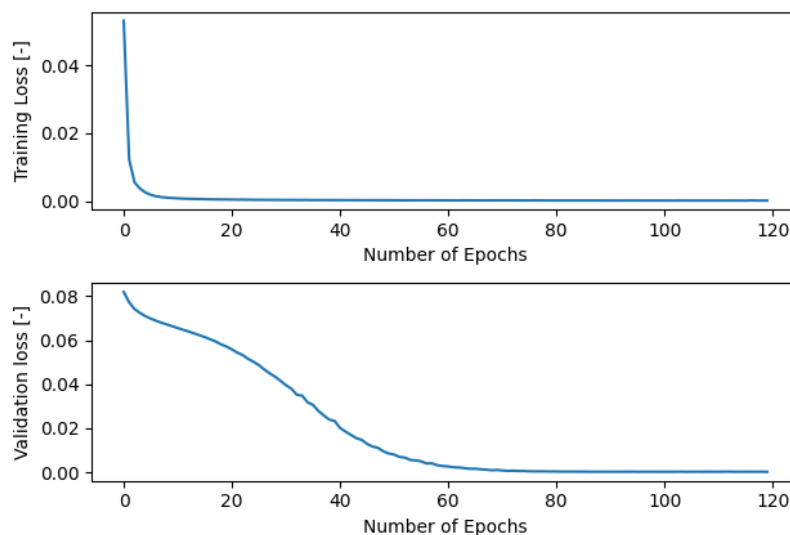
V následujícím experimentu byla použita architektura BAE2+ConvM3 s velikostí latentní dimenze 80 a velikost vstupního obrazu 128x128 pixelů a velikostí dávky 32. Z poznatků z předchozího experimentu lze dataset značně redukovat za účelem snížení výpočetní a časové náročnosti učení jednotlivých epoch. Dataset byl zredukován na 280 OK vzorků v trénovacím a 50 vzorků ve validačním datasetu, zbytek vzorků byl

ponechán v testovacím datasetu. Níže uvedené matice záměn ukazují výsledky experimentu



Obrázek 6.4 Srovnání výsledku algoritmů pro počet epoch 1 a 120 (CAE BAE2, Conv3, LD 80, size 128, velikost dávky 32)

Training and Validation Loss of BAE2 with ConvM3 structure



Obrázek 6.5 Průběhy trénovací a validační ztráty učení (CAE BAE2, Conv3, LD 80, size 128, počet epoch 120 velikost dávky 32)

ZÁVĚR

V úvodu práce byly rozebrány unární binární a obecné multi-class typy klasifikačních úloh a základní koncepce používané ve strojovém učení spolu s popsáním některých hyperparametrů a metod hodnocení klasifikačních algoritmů.

V druhé kapitole byla provedena rešerše počtu možných architektur klasifikátorů pro jednotřídní klasifikaci s jejich popisem, výhodami a nevýhodami a příklady použití. Pro praktickou část byl z těchto architektur vybrán konvoluční autoenkodér CAE.

Ve třetí kapitole je popsán proces tvorby datasetu. Konkrétněji definice OK, NOK kusů a tříd, postup tvorby vlastní scény pro tvorbu datasetu, samotný proces pořizování fotek, MATLAB script použitý pro ovládání kamer a proces augmentace dat. Celkově bylo pořizeno 2470 fotek, 1235 snímků pro boční a horní pohled, které se následně dělí na 231 NOK a 1004 OK snímků pro každý pohled zvlášť.

Ve čtvrté kapitole je popsán výběr implementace algoritmu pro provedení experimentů, její následný popis a popis převzatého/vytvořeného kódu. Byl vybrán CAE z práce [24][25], většina kódu byla reimplementována za účelem ověření vlastností dané architektury na pořízeném na datasetu.

V páté kapitole je popis koncepce provedených experimentů na implementovaném CAE, první experiment pojednává o různém nastavení hyperparametrů pro nalezení jejich optima pro detekci anomálií. Druhý experiment se věnuje různému nastavení poměrů fotek v trénovacím, validačním a testovacím datasetu. Třetí experiment ověřuje nereprezentativnost datasetu přidáváním postupně většího počtu NOK kusů do trénovacího datasetu, Čtvrtý experiment se věnuje výpočtení a časové náročnosti učení, zastavení procesu učení dříve nebo později a vyhodnocení kvality klasifikace Pátý experiment se zabývá redukcí samotných datasetů postupným snižováním počtu snímků v trénovacím a validačním datasetu, experiment ověřující mohutnost datasetu nutnou pro kvalitní klasifikaci.

V šesté kapitole jsou popsány provedené experimenty. Jako nejlepší nastavení parametrů sítě se ukázala kombinace jádra BAE2 s architekturou kodéru a dekodéru ConvM3, s velikostí latentní dimenze 80, velikostí vstupních obrázků 128x128 a velikostí 40, detailnější hodnoty jsou v tabulce 6.1. Nejlepší poměr datasetu byl zvolen jako: trénovací dataset 473 OK+473 augmentovaných OK kusů a 300 OK kusů pro validační dataset. Při zvyšování počtu NOK kusů v trénovacím datasetu se ukázalo že experiment nebyl úspěšný pravděpodobně vlivem resubstituční chyby, jelikož z důvodů nedostatku NOK kusů byly fotky do trénovacího datasetu kopírovány, ale počet FP klasifikací, který by byl s resubstituční chybou v tomto případě očekáván nebyl vysoký. Předposledním experimentem byl vysledovány minimální objemy trénovacího a validačního datasetu, nutné ke kvalitní klasifikaci, výsledný objem trénovacího datasetu 35 OK kusů a výsledný objem validačního datasetu pouhých 6 OK kusů, experiment byl přerušen z důvodů ověření dostatečné mohutnosti datasetu.

Posledním experimentem byla ověřena náchylnost datasetu na přeučení kdy pro první iteraci byl algoritmus naučen pouze v jedné epoše a následně bylo provedeno 120 epoch, srovnání maticemi záměn je na obrázku 6.4. Výsledky byly příznivé, algoritmus není v podstatě vůbec náchylný na přeučení, tato vlastnost byla experimentálně ověřena na malém množství různých kombinací hyperparametrů se srovnatelnými výsledky.

Pro vyhodnocení kvality experimentů byla primárně použita ROC křivka a z ní vycházející AUC metrika nebo samotná matice záměn po případě počet FP a FN klasifikací.

LITERATURA

- [1] DOMINGOS, Pedro. A Few Useful Things to Know About Machine Learning. Communications of the ACM [online]. **2012** [cit. 2023-11-08]. Dostupné z: <https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf>
- [2] LUKASZCZYK, Jakub. *Vizuální detekce anomálií v průmyslové výrobě* [online] **2023**. [cit. 2023-11-17]. Dostupné z: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=253558
- [3] KHAN, Shehroz S. Michael G. MADDEN. *One-class classification: taxonomy of study and review of techniques*. The Knowledge Engineering Review [online]. **2014** [cit. 2023-11-20]. Dostupné z: doi:10.1017/S026988891300043X
- [4] PANG, Guansong, Chunhua SHEN, Longbing CAO a Anton Van Den HENGEL. Deep Learning for Anomaly Detection: A Review ACM Computing Surveys [online]. **2020**, [cit. 2023-12-10]. Dostupné z: doi:10.1145/3439950
- [5] KUMAR, Ajitesh. Difference: Binary, Multiclass & Multi-label Classification [online]. **2022** [cit. 2023-12-20]. Dostupné z: <https://vitalflux.com/difference-binary-multi-class-multi-label-classification>
- [6] CHEN, Sathe, Aggarwal, Turaga: Outlier Detection with Autoencoder Ensembles [online]. **2017** [cit. 2023-12-29]. Dostupné z: <https://saketsathe.net/downloads/autoencode.pdf>
- [7] PINTELAS, E.; Livieris, I.E.; Pintelas, P.E. A Convolutional Autoencoder Topology for Classification in High-Dimensional Noisy Image Datasets. Sensors [online]. **2021**, [cit. 2023-12-29]. Dostupné z: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8622369/>
- [8] ROCCA Joseph. Understanding variational autoencoders (vaes) – building, step by step, the reasoning that leads to vaes [online]. **2019**. [cit. 2023-12-30]. Dostupné z: <https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73>
- [9] SCHNEIDER Sarah, Doris ANTENSTEINER, Daniel SOUKUP a Matthias SCHEUTZ. Autoencoders - A Comparative Analysis in the Realm of Anomaly Detection [online]. **2022** [cit. 2023-12-29]. Dostupné z: [https://openaccess.thecvf.com/content/CVPR2022W/WiCV/papers/Schneider_Autoencoders -
_A Comparative Analysis in the Realm of Anomaly CVPRW 2022 paper.pdf](https://openaccess.thecvf.com/content/CVPR2022W/WiCV/papers/Schneider_Autoencoders_-_A_Comparative_Analysis_in_the_Realm_of_Anomaly_CVPRW_2022_paper.pdf)
- [10] SABOKROU, Khalooei, Fathy, Adeli: Adversarially Learned One-Class Classifier for Novelty Detection [online]. **2018** [cit. 2023-12-29]. Dostupné z: <https://arxiv.org/abs/1802.09088>
- [11] RAVANBAKSH, Nabi, Sangineto, Marcenaro, Regazzoni, Sebe: Abnormal Event Detection In Videos Using Generative Adversarial Nets [online]. **2017** [cit. 2023-12-29]. Dostupné z: <https://arxiv.org/abs/1708.09644>

- [12] MATHIEU, Couprie, LeCun: Deep Multi-Scale Video Prediction Beyond Mean Square Error [online]. **2016** [cit. 2023-12-29]. Dostupné z: <https://arxiv.org/abs/1511.05440>
- [13] NOTO, Brodley, Slonim: FRaC: A Feature-Modeling Approach For Semi-Supervised And Unsupervised Anomaly Detection [online]. **2012** [cit. 2023-12-30]. Dostupné z: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3359096/>
- [14] HONZÍK Petr: Učení založené na instancích [online]. **2014** [cit. 2023-12-30]. Dostupné z: http://vision.uamt.feec.vutbr.cz/STU/lectures/05_Uceni_zalozene_na_instancich.pdf
- [15] NGUYEN, Vien: Scalable And Interpretable One-Class Svms With Deep Learning And Random Fourier Features [online]. **2018** [cit. 2023-12-30]. Dostupné z: <https://arxiv.org/abs/1804.04888>
- [16] RUFF, Vandermaulen, Goernitz, Deecke, Siddiqui, Binder, Müller, Kloft: Deep One-Class Classification [online]. **2018** [cit. 2023-12-30]. Dostupné z: <https://proceedings.mlr.press/v80/ruff18a.html>
- [17] GRANDINI, Bagli, Visani: Metrics For Multi-Class Classification: An Overview [online]. **2020** [cit. 2023-12-30]. Dostupné z: <https://arxiv.org/abs/2008.05756>
- [18] OH, Iyengar: Sequential Anomaly Detection using Inverse Reinforcement Learning [online]. **2020** [cit. 2024-01-01]. Dostupné z: <https://arxiv.org/abs/2004.10398>
- [19] GUTMAN, Hyvärinen: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models [online]. **2010** [cit. 2024-01-01]. Dostupné z: <https://proceedings.mlr.press/v9/gutmann10a/gutmann10a.pdf>
- [20] Portál REC21 [online]. **2024** [cit. 2024-03-11]. Dostupné z: <https://eshop.rec21.cz/pripinacky-10mm-100ks-222-hodan-doprodej-105015cz557/>
- [21] Portál OEM CAMERAS [online]. **2024** [cit. 2024-03-16]. Dostupné z: <https://www.oemcameras.com/dfk-21au04.htm>
- [22] Portál AVSUPPLY [online]. **2024** [cit. 2024-03-16]. Dostupné z: <https://www.avsupply.com/ITM/6497/DFK%2041BU02.H.html>
- [23] Portál ADI [online]. **2024** [cit. 2024-03-20]. Dostupné z: <https://adiglobal.cz/cz/produkty900:3398817/objektiv-miv-1-3-f-550mm-cs>
- [24] BILÍK Šimon, Framework for the AE reconstruction and feature based AD [online]. [cit. 2024-04-05]. Dostupné z: <https://github.com/boortel/AE-Reconstruction-And-Feature-Based-AD>
- [25] Bilík Šimon, Feature space reduction as data preprocessing for the anomaly detection [online] **2021** [cit. 2024-04-05]. Dostupné z: https://www.fekt.vut.cz/conf/EEICT/archiv/sborniky/EEICT_2021_sbornik_1.pdf
- [26] Mishchuk, Mishkin, Radenovic, Matas: Working hard to know your neighbor's margins: Local descriptor learning loss [online]. **2017** [cit. 2024-04-11]. Dostupné z: <https://arxiv.org/pdf/1705.10872>

- [27] Bergmann, Fauser, Sattlegger, Steger: MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection [online]. **2019** [cit. 2024-04-18] Dostupné z:
https://www.mvtec.com/fileadmin/Redaktion/mvtec.com/company/research/datasets/mvtec_ad.pdf
- [28] Ligocki Adam: Convolutional NNs [online]. [cit. 2024-04-27]. Dostupné z:
http://vision.uamt.feec.vutbr.cz/STU/lectures/41_MUIN_nn_CNN_Ligocki_2019.pdf
- [29] Portál Keras [online]. [cit. 2024-04-28]. Dostupné z:
<https://keras.io/api/layers/activations/>