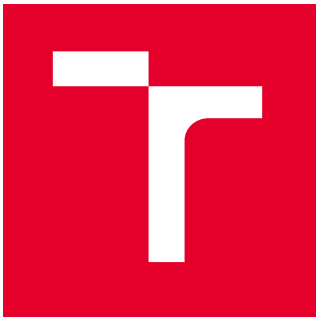


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

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 — **Bc. Jakub Lukaszczyk**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Karel Horák, Ph.D.**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

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

Ústav automatizace a měřicí techniky

Student: Bc. Jakub Lukaszczyk ***ID:*** 211157

Ročník: 2 ***Akademický rok:*** 2022/23

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 výsledky detekce anomálií a navrhnout a implementovat úlohu experimentálního ověření jednotřídní klasifikace, a to na multi-kamerovém systému. Jde tedy o úlohu třízení 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.
3. Rozšiřte/modifikujte stávající řídicí aplikaci pro použití s HW platformou detekce anomálií (pořízení obrazu z více kamer současně a řízení HW platformy).
4. Podle domluvy s vedoucím pořiďte vlastní anotovaný dataset čítající 1000+ realizací průmyslového výrobku.
5. Společně s vedoucím zvolte vhodný klasifikační algoritmus a proveďte z hlediska řídicí aplikace nutné změny, popř. algoritmus modifikujte podle potřeb vámi navržených experimentů.
6. Pomocí řídicí aplikace proveďte na pořízeném datasetu sadu experimentů se zaměřením na zjištění vlivu definice anomálií na výkonnost klasifikátoru.
7. Výsledky experimentů vyhodnoťte formou matice záměn a 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. Tax D.: One-class classification - Concept-learning in the absence of counter-examples.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 17.5.2023

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

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

ABSTRAKT

Práce se zabývá detekcí anomálií v obrazových datech pořízených na průmyslovém výrobku. V první části je nastíněna problematika a přístupy k jejímu řešení pomocí hlubokého učení. Dále jsou rozebrány některé z používaných architektur, které lze pro tuto úlohu použít. V praktické části je poté popsána platforma pro průmyslovou inspekci, použitý software a tvorba vlastního anotovaného datasetu. Poskytnutý software je rozšířen o funkcionalitu pro ovládání platformy a práci s více kamerami. Poslední část se zabývá experimenty, jejichž cílem je zjistit vliv datasetu na výsledný model a odhad jeho výkonnosti. Experimenty vyhodnocují vliv jak v trénovací, tak i v testovací fázi.

KLÍČOVÁ SLOVA

Detekce anomálií, Obraz, DeepSVDD, DDR-ID, AnoGAN, Autoenkodér, Více pohledů, Tvorba datasetu, Vliv datasetu

ABSTRACT

This work deals with the detection of anomalies in image data taken on an industrial product. The first part outlines the problem and approaches to its solution using deep learning. Then, some of the architectures that can be used for this task are discussed. The practical part then describes the platform for industrial inspection, the software used and the creation of the annotated dataset. The software is extended with features for controlling the platform and working with multiple cameras. The last section deals with experiments designed to investigate the effect of the dataset on the resulting model and the estimation of its performance. The experiments evaluate the influence in both training and testing phases.

KEYWORDS

Anomaly detection, Image, DeepSVDD, DDR-ID, AnoGAN, Autoencoder, Multiview, Dataset Creation, Influence of the Dataset,

LUKASZCZYK, Jakub. *Vizuální detekce anomálií v průmyslové výrobě*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2023, 108 s. Diplomová práce. Vedoucí práce: Ing. Karel Horák, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Jakub Lukaszczyk
VUT ID autora:	211157
Typ práce:	Diplomová práce
Akademický rok:	2022/23
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/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Karlovi Horákovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

V dnešní době můžeme napříč průmyslovými odvětvími pozorovat velký tlak na kvalitu výroby. Zatímco v době první průmyslové revoluce byl tlak kladen na dělení a zrychlení výroby, tak dnes, o 200 let později, se dostáváme do stavu, kdy se výrobní takt stává natolik rychlým, že klasické systémy pro kontrolu kvality již nestíhají s výrobou držet krok.

Navíc jsou v dnešní době požadavky na kvalitní výrobu na velice vysoké úrovni. Výrobce musí totiž splnit nejen specifikaci, kterou sám definoval, ale také musí splnit množství norem, nařízení, zákonů a dalších předpisů, které definují požadavky na výrobky uváděné na trh. Vypuštění výrobku, který neodpovídá specifikaci do prodeje tak představuje obrovský marketingový problém a s ním spojené finanční ztráty. Právě proto jsou systémy pro detekci anomálií stále více žádané.

Anomálie mohou mít mnoho podob v různých doménách, od elektronických odchylek, přes mechanickou odolnost až po samotný vzhled výrobku. Právě vzhledu výrobku se věnuje tato práce. Zpracování obrazové informace je obecně složitá úloha, jelikož se potýká s množstvím dat, a i pouhá změna polohy objektu ve snímku naprosto mění jeho vlastnosti.

V případě, že jsou anomálie známé a jasně definovatelné, nabízí se přístup k detekci pomocí na míru navržených algoritmů. Bohužel, v technické praxi je obvykle nedostatek vzorů, které představují anomální výrobky a výčet možných anomálií bývá omezený a nekompletní.

Proto je vhodnější využití prostředků hlubokého učení. To znamená, že pro návrh detekčního systému nepotřebujeme znát analytický popis kontrolovaného produktu. Stačí pouze dostatečné množství jeho realizací. Jedná se o elegantní systém, který vytváří model pouze z vyhovujících výrobků a při následné detekci anomálií zkoumá, zda předložená data tomuto modelu odpovídají či nikoliv.

Cílem této práce bude prozkoumat existující metody pro vizuální detekci anomálií a zvolit metodu vhodnou pro detekci vad v průmyslovém výrobku. Dále bude zkoumán vliv změny definice anomálií na výsledný model. Je totiž velice důležité zjistit, jaký vliv má například použití jen omezeného typů anomálií při testování datasetu. Pokud nemá typ anomálií vliv na odhad výkonnosti, pak není nutné vytvářet rozsáhlý a nákladný testovací dataset. Naopak pokud vliv typu testovacích anomálií na výsledné odhady je významný, pak je nutné již ve fázi návrhu systému zajistit dostupnost dostatečného množství anomálních výrobků.

Celá problematika je v práci rozdělena do tří částí. V první části bude provedena rešerše existujících metod. Druhá část se potom zabývá použitým inspekčním systémem, jeho úpravami a tvorbou datasetu. Náplní třetí, poslední, části bude provedení experimentů za účelem zjištění vlivu definice anomálií na výsledný model.

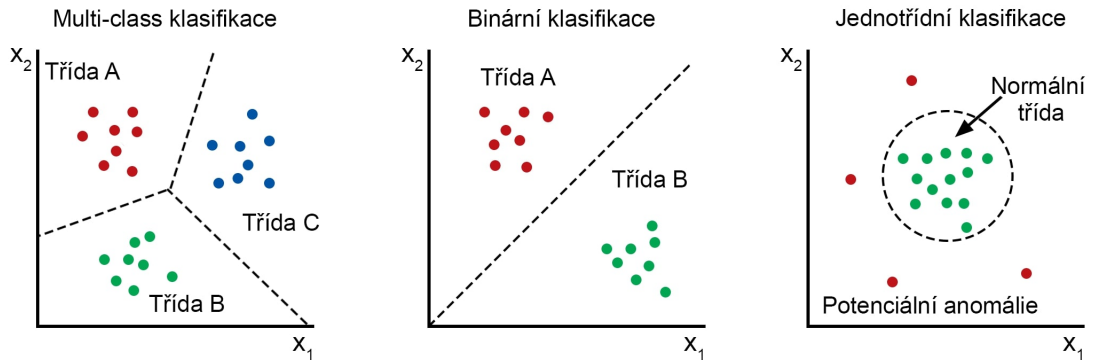
1 Teorie klasifikačních metod

Kapitola se zabývá typy klasifikačních úloh, které jsou krátce popsány a je diskutována jejich vhodnost pro použití pro detekci anomálií. Druhá část této kapitoly se potom zabývá konkrétními architekturami pro unární klasifikaci obrazových dat. Poslední část potom krátce představuje technické prostředky pro evaluaci klasifikátorů.

1.1 Klasifikační metody

V rámci strojového učení definujeme tři základní typy klasifikačních úloh: Vícetřídní klasifikace, binární klasifikace a jednotřídní klasifikace [1]. Někdy jsou tyto skupiny dále děleny na vyvážené a nevyvážené [2] podle počtu trénovacích vzorů v jednotlivých třídách.

Při řešení libovolné úlohy strojového učení je nutné v prvé řadě správně určit typ klasifikace, který bude pro daný problém vhodný. Špatná volba může totiž vést k vytvoření neobjektivního klasifikátoru a to proto, že se použije mechanismus, který klasifikuje na základě předpokladů, které neodpovídají skutečné povaze úlohy. Cílem této kapitoly je tudíž krátce popsat a definovat různé přístupy k řešení klasifikačních problémů vyskytujících se v oblasti strojového učení a vydělit úlohy, pro které je jejich volba vhodná. Vzhledem k povaze této práce je na vhodnost metod pohlíženo převážně z perspektivy detekce anomálií.



Obr. 1.1: Srovnání základních typů klasifikačních úloh

1.1.1 Multi-class klasifikace

Také označována jako diskriminační přístup. Cílem je přiřadit vzor do jedné z předdefinovaných tříd (minimálně dvou). Pro použití tohoto přístupu ke klasifikaci je

zásadní, aby všechny třídy byly jasně definovány a reprezentativně zastoupeny ve formě vzorů [3].

Použití vícetřídní klasifikace pro detekci anomálií přichází v úvahu pouze v případě, že jsou předem známy všechny možné anomálie, které mohou nastat a je k dispozici dostatek vzorů, pro reprezentativní naučení modelu.

Další možností využití vícetřídní klasifikace pro detekci anomálií, je kombinace s unární klasifikací, kde využijeme unární klasifikátor pro detekci anomálních vzorů a ty následně pomocí vícetřídního klasifikátoru třídíme do předem definovaných tříd definujících různé vady.

Obecně je vícetřídní klasifikace vhodnější pro dělení různých objektů do tříd, než použití pro přímou detekci vad v rámci jednoho objektu [4].

Mezi metody pro vícetřídní klasifikaci se řadí například:

- Naivní Bayesův klasifikátor
- K-nejbližších sousedů
- Rozhodovací stromy
- Neuronové sítě

S rostoucí dimenzí vstupních dat (obraz) roste potřeba extrakce příznaků. Pro tyto účely lze použít například konvoluční sítě [5].

1.1.2 Binární klasifikace

Binární klasifikace je specifickým typem vícetřídní klasifikace. V rámci binární klasifikace rozlišujeme pouze mezi dvěma třídami [6].

Častou aplikací tohoto přístupu jsou úlohy, kde předem víme, že vzory budou vždy připadat do jedné ze dvou tříd. Příkladem může být klasifikátor vyhodnocující, zda se v obraze nachází pes nebo kočka. Důležitou vlastností, kterou je třeba mít při použití na paměti, je fakt, že v případě předložení vzoru, který nepatří ani do jedné z tříd, bude i tak výstupem klasifikace do jedné z těchto tříd.

Další častou aplikací pro binární klasifikaci je kontrola shody se specifikací. Zde jedna třída reprezentuje shodu se specifikací a do druhé třídy patří vzory, které specifikaci neodpovídají. V takovémto případě je odpovídající třída dobře definovatelná a vzory lze obvykle dobře realizovat.

Naproti tomu třída, která specifikaci neodpovídá je obvykle reprezentovaná příliš malým množstvím vzorů a není dostatečně konkrétně definována (apriorně nejsou známy všechny možnosti, jak se mohou vzory vymykat specifikaci). Tato nevyváženost vede k neobjektivním klasifikačním pravidlům a dále potom k neobjektivním klasifikacím pro nové vzory [3].

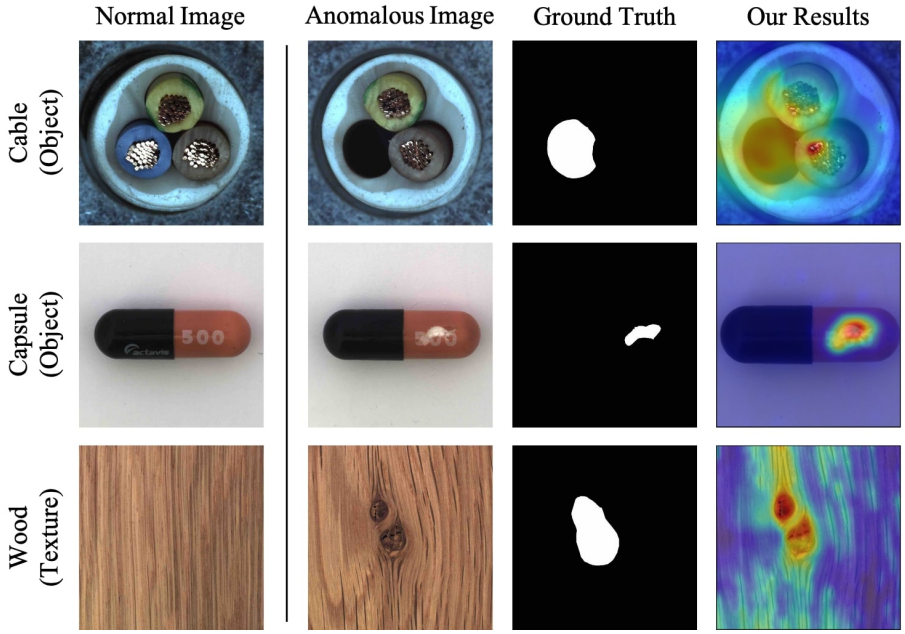
Je tedy zřejmé, že ačkoliv bývá binární klasifikátor používán pro detekci anomálií, tak tento přístup může vést k neobjektivním a zaujatým závěrům [3].

Pro binární klasifikaci se obecně používají podobné metody jako pro klasifikaci obecnou. Tyto metody jsou často upraveny pro binární povahu klasifikačního problému.

1.1.3 Unární klasifikace

Unární nebo také jednotřídní (one-class) klasifikace je typ úlohy, kde cílem není roztřídit vstupní data do několika tříd, ale pouze identifikace zájmových objektů (vzorů) jedné třídy mezi všemi ostatními objekty. Typické pro tento typ klasifikace je učení pouze se vzory zájmové třídy [7].

Příkladem může být opět úloha shody se specifikací. Oproti binární klasifikaci, zde požadujeme pouze sestavení modelu zájmové třídy, tedy jasnou definici realizací, které vyhovují specifikaci a jejich dostatečné zastoupení ve formě vzorů. Tento úkol je již v praxi mnohem realističtější, jelikož specifikace bývají jasně definovány společně s možnými odchylkami. Realizace množství vzorů, které tuto specifikaci reprezentativně pokrývají také není problematická.



Obr. 1.2: Ukázka obrazových anomálií a jejich detekce (převzato z [8])

Na základě těchto údajů lze potom sestavit model, který definuje hranici kolem zájmové třídy a nové vzory jsou pak klasifikovány objektivně, jelikož rozhodování není zatíženo špatně reprezentovanou *třídou neodpovídající specifikaci*.

Tento typ úlohy je nejtypičtější pro detekci anomálií [3], která je náplní této práce. Obvykle jsou totiž známy realizace, které jsou v pořádku, a anomální realizace nejsou explicitně zadány.

Unární klasifikace může být použita pro jednotřídní i vícetřídní problémy. Pro použití s vícetřídním problémem je nutné pro každou definovanou třídu sestavit vlastní model nezávislý na ostatních třídách. Je však nutné mít na paměti, že vytvořením nezávislých modelů mohou vzniknout nejednoznačnosti, kdy je vzor zároveň klasifikován do více tříd [3]. Tyto nejednoznačnosti je nutno řešit buďto dalšími mechanismy, nebo zvážit, zda překrývající se oblasti tříd ve skutečnosti nedefinují samostatnou třídu.

Metody používané pro unární klasifikaci jsou obvykle specificky navržené pro tuto úlohu. Mezi takové metody se řadí například UNEQ [9] nebo SVM [10]. V praxi je také populární přístup, kdy jsou pro detekci anomálií použity modifikace metod, sloužících původně k jinému účelu. Zde se řadí například metody založené na autoenkodérech nebo GAN sítích. Jednotlivé metody pro unární klasifikaci v kontextu detekce anomálií budou probrány v kapitole 1.2.

1.1.4 Volba vhodného klasifikátoru

Jak již bylo uvedeno v předchozích kapitolách, volba vhodného klasifikátoru záleží v prvé řadě na povaze úlohy.

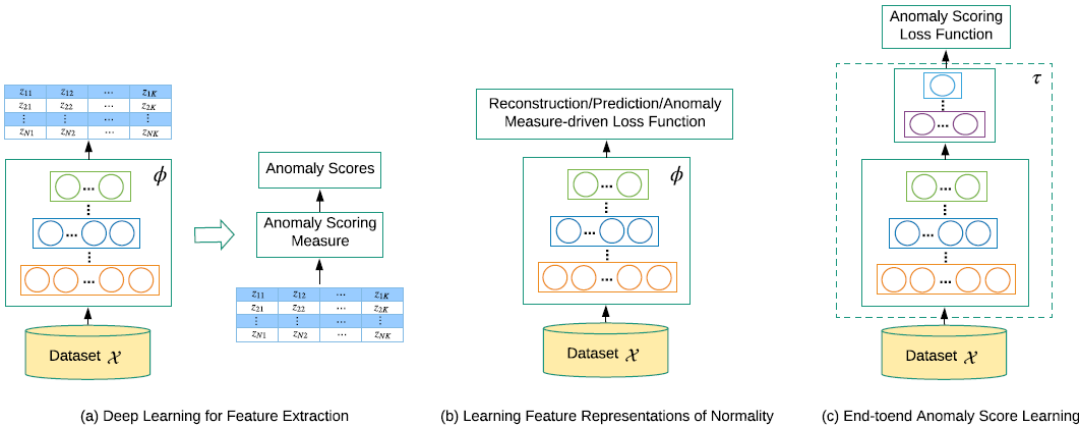
Binární, nebo obecně vícetřídní, klasifikátor připadá v úvahu v případech, kdy jsou jednotlivé třídy jasně definované a cílem je z příznaků odvodit hranici mezi těmito třídami. V ideálním případě jsou trénovací množiny pro všechny třídy přibližně stejně velké, aby se předešlo zaujatosti výsledného modelu vůči některým třídám čistě z důvodu jejich početnosti.

Naopak jednotřídní klasifikace je vhodná v případech, kdy je cílem určit, zda vzor odpovídá naučené reprezentaci nebo nikoliv. V těchto úlohách máme definovánu pouze jednu třídu a vyhodnocujeme, zda předložený vzor této třídě odpovídá nebo ne. Množina vzorů, které specifikaci neodpovídají, je heterogenní a nelze ji tedy apriorně popsat nebo pro ni vytvořit reprezentativní sadu vzorů.

Pro detekci anomálií tedy v naprosté většině případů budou voleny architektury klasifikátorů určené pro jednotřídní klasifikaci, která umožní oddělení homogenní množiny definovaných vzorů od heterogenní množiny anomálií. Pokud by bylo požadováno třídění anomálií do kategorií, pak je vhodné použít vícetřídní klasifikátor až na výstup této unární úlohy a třídit pouze vzory, které v předchozím kroku byly označeny za anomální.

1.2 Architektury pro unární klasifikaci

Architektury sítí hlubokého učení pro detekci anomálií lze v zásadě rozdělit do tří základních kategorií [11]. První dvě kategorie jsou metody, které pro detekci vyžívají převzatý princip a anomálie detekují spíše jako vedlejší produkt. Třetí skupinou jsou potom metody, které byly navrženy přímo pro úlohu detekce anomálií, a tudíž v průběhu učení probíhá přímo optimalizace kritéria udávajícího anomalitu.



Obr. 1.3: Koncepce metod pro detekci anomálií (převzato z [11])

Pro detekci anomálií za použití hlubokého učení tedy existují tři základní paradigmata, která budou dále rozebrána:

- Využití hlubokého učení pro extrakci příznaků
- Učení příznaků, které definují normalitu
- End-to-end učení s definicí anomálního skóre

1.2.1 Využití hlubokého učení pro extrakci příznaků

Cílem je využití hlubokého učení pro extrakci příznakových vektorů o nízké dimenzi z vysoko-dimenzionálních příznakových vektorů (obraz). Hluboké učení je zde využito čistě jako mechanismus pro redukci dimenzionality, který lze symbolicky zapsat následovně [11]:

$$\boldsymbol{z} = \phi(\boldsymbol{x}; \Theta) \tag{1.1}$$

Kde ϕ představuje funkci mapování příznaků, jejíž parametry jsou získány pomocí hlubokého učení. $\boldsymbol{x}$ představuje vstupní data a Θ potom představuje váhy dané sítě.

Oproti metodám jako je PCA [12] nebo náhodná projekce [13], techniky hlubokého učení opakovaně vykazují podstatně lepší schopnost extrakce sémanticky relevantních a nelineárních vztahů mezi příznaky [11].

Při návrhu těchto metod se vychází z předpokladu, že nízko-dimenzionální příznaky identifikované pomocí hlubokého učení si zachovávají diskriminativní informaci, která je důležitá pro separaci anomálií a normálních instancí. Pro samotnou detekci anomálií se poté používají klasické metody, vhodné pro nízko-dimenzionální detekci.

Výhodou tohoto přístupu je možnost využít předtrénované modely a již existující metody pro detekci anomálií v nízko-dimenzionálních datech. Oproti klasickým metodám je využití hlubokého učení pro detekci anomálií mnohem efektivnější. Dále umožňují jednoduchou implementaci a nasazení vzhledem k použití dnes už standardních nástrojů [14].

Mezi nevýhody se potom řadí oddělení mechanismu extrakce příznaků a detekce anomálií, které vede k suboptimálním výsledkům.

Nízko-dimenzionální reprezentace obvykle pomůže s detekcí skrytých anomálií a snižuje množství falešně pozitivních detekcí, zároveň však může dojít ke stavu, kdy jsou vstupní data natolik zkomprimovaná, že výstup extraktoru není schopen z důvodu kapacity pojmout některé typy komplexnějších anomálií. Z tohoto důvodu je nutné vhodně zvolit množství extrahovaných příznaků, aby byly dostatečně konkrétní, ale zároveň aby jich pro klasifikátor nebylo příliš mnoho z důvodu výpočetní náročnosti [11].

1.2.2 Učení příznaků, které definují normalitu

Oproti předchozímu přístupu, tyto metody do jisté míry spojují extrakci příznaků s ohodnocováním anomálií. Mechanismy extrakce a klasifikace nejsou tedy plně oddělené [11].

Definice této kategorie je poměrně obecná a zahrnuje mnoho metod s mírně rozdílným přístupem k řešení problému. Z toho důvodu bude pouze nastíněn přístup k řešení a případný bližší popis bude uveden až u konkrétních metod v další kapitole.

Reprezentace dat je naučena optimalizací obecného cíle pro učení příznaků. Tento přístup využívá podobností mezi úlohami extrakce příznaků a detekcí anomálií a nedefinuje tedy přímo úlohu detekce anomálií.

Jedním z přístupů je využití teoretického základu z klasických metod mezipříznakové analýzy nebo modelů příznaků [15]. V praxi fungují metody tak, že sestavíme soubor modelů, kdy každý model se snaží předpovědět jeden příznak na základě ostatních příznaků. Konzistence předpovědi napříč modely potom udává

anomální skóre. Konkrétní výpočet skóre záleží na zvolené metodě (průměr, většina, logaritmické metody...).

Dalším přístupem může být snaha o provedení jednotřídní klasifikace normální třídy. Jedná se o nejvhodnější přístup pro detekci anomálií [3]. Často se využívá metod odvozených od SVM - Support Vector Machines [10] a SVDD [16].

Třetí přístup vychází ze shlukové analýzy, kde se předpokládá, že anomálie jsou jednoznačně oddělitelné od hustých shluků normálních instancí. Shluková analýza a detekce anomálií spolu úzce souvisí a není tedy překvapivé, že existuje velké úsilí o převedení úlohy analýzy shluků na úlohu detekce anomálií.

Mezi tyto metody se také řadí metody využívající rekonstrukci dat. Jedná se převážně o metody založené na GAN sítích [17][18] a na sítích typu autoenkodér [19][20][21][22][23]. Základní myšlenka vychází ze snahy porovnat předložený vzor a vzor, který je sítí rekonstruován. Pokud je vzor rekonstruován dobře (síť byla na obdobné vzory naučena a umí je dobře rekonstruovat), pak je vzorek označen za normální. V případě, že je rekonstrukční chyba velká, je vzorek označen za anomálii.

Metody náležející do skupiny extrakce příznaků definujících normalitu mají tu výhodu, že staví na silných základech metod pro extrakci příznaků. Tyto metody jsou v dnešní době dobře prozkoumané a chování takových architektur lze relativně dobře předpovídat, ať už z pohledu přesnosti, tak i z pohledu výpočetní náročnosti.

Na druhou stranu tyto metody obvykle trpí na problémy spojené se špatně anotovanými nebo zašuměnými daty, které jsou předloženy ve fázi učení [24]. Dále, jelikož učíme sítě extrahovat příznaky a teprve na jejich základě definujeme ohodnocení anomálie, tak se vystavujeme riziku, že výsledná reprezentace anomálií nebude optimální. V průběhu učení mohou být extrahovány příznaky, které jsou důležité pro rekonstrukci nebo klasifikaci normálních vzorků, ale není garantováno, že tyto příznaky budou relevantní také pro anomálie.

1.2.3 End-to-End učení s definicí anomálního skóre

Oproti ostatním metodám, tyto metody optimalizují váhy uvnitř sítí na základě ztrátové funkce, která optimalizuje úlohu detekce anomálií přímo [11]. Následující přeresolutions říká, že se snažíme najít takové nastavení vah a parametrů Θ sítě τ tak, aby výsledná ztrátová funkce ℓ byla minimální pro všechny vstupní vzory x [11].

$$\Theta^* = \underset{\Theta}{arg\,min} \sum_{x \epsilon \chi} \ell(\tau(x; \Theta)) \tag{1.2}$$

$$s_x = \tau(x; \Theta^*) \tag{1.3}$$

Kde:

- τ - neuronová síť
- x - vstupní data
- Θ - parametry sítě τ
- ℓ - ztrátová funkce
- s_x - výsledné anomální skóre

Dále, na rozdíl od ostatních metod, reprezentace příznaků a anomální skóre jsou vypočítávány současně již v průběhu učení. Tato vlastnost dává metodám větší potenciál pro kvalitní detekci anomálií.

Jednou z takových metod může být metoda, která svou architekturou vychází ze sítí GAN [25], ale definuje inovativní přístup k učení. Oproti dříve uvedeným metodám, se nesnažíme naučit generátor generovat věrohodnou reprezentaci, ale snažíme se naučit diskriminátor rozeznat rozdíl mezi normální instancí a anomálií [17]. Jelikož anomálie nebývají dostupné, může být síť sestavena tak, aby byla schopna si generovat vlastní umělé anomálie. Tato filozofie předpokládá, že anomálie lze efektivně syntetizovat. Normální instance lze potom třídit pomocí jednotřídního diskriminátoru.

Výhodou je, že detekce anomálií probíhá end-to-end. Síť je optimalizována přímo na úlohu detekce anomálií. Nevýhodou však je, že syntéza anomálií negarantuje, že tyto anomálie budou odpovídat skutečným anomáliím, které se v datech mohou objevit. Dále obecná nestabilita při učení GAN sítí může vést ke generování suboptimálních syntetických snímků a degradovat tak schopnost sítě detekovat anomálie.

Jedná se o metody s největším potenciálem, avšak tato oblast je v současné době mnohem méně probádaná, než oblast metod přejatých [11].

1.3 Rozbor architektur

V rámci práce byly prozkoumány různé používané metody. V následujících kapitolách jsou některé z metod popsány. Metody byly vybírány podle jejich zastoupení v literatuře, ale také podle různorodých přístupů k řešení problémů. Nutno podotknout, že většina těchto metod jsou spíše nadskupinou, v rámci které existuje mnoho modifikací, z nichž některé jsou zmíněny, ale vzhledem k počtu těchto modifikací se ani zdaleka nejedná o kompletní přehled.

Metody, které budou rozebrány dále jsou následující:

- Autoenkodér
- GAN
- DDR-ID
- DeepSVDD

1.3.1 Autoenkodér

Autoenkodér je metoda, která staví na redukci dimenzionality vstupních dat. Základní myšlenka je taková, že vstupní data jsou komprimována do prostoru o nízké dimenzi a následně jsou opačnou sítí zpětně rekonstruována [23]. Optimalizace metody je řízena snahou o minimalizaci rekonstrukční chyby (rozdílu mezi původním a rekonstruovaným vzorkem). Jelikož výstup kodéru a vstup enkodéru představuje prostor s nižší kapacitou, než je kapacita vstupních dat, je síť v průběhu učení nucena extrahovat příznaky, které jsou kritické pro rekonstrukci [20].

Jelikož při učení dostává síť pouze normální vzorky, tak se naučí kvalitně rekonstruovat pouze tuto normální třídu. Když je potom v klasifikační fázi předložen anomální vzorek, tak se očekává, že síť nebude schopna jej kvalitně rekonstruovat a tím pádem dojde ke zvýšení rekonstrukční chyby, která je použita jako anomální ukazatel.

Celá myšlenka vychází z předpokladu, že normální třída má proměnné, které jsou spolu vysoce korelovány a lze je tedy promítnout do prostoru o nižší dimenzi, zatímco vztahy a korelace mezi těmito proměnnými jsou u anomálií velmi rozdílné, a tudíž lze anomalitu v nižším prostoru lépe odhalit [11].

Autoenkodéry jsou pro detekci anomálií často používány pro jejich jednoduchost a přímočarý přístup k úloze detekce anomálií [11][20][26]. Další výhodou je, že autoenkodéry s konvolučními neuronovými sítěmi jsou schopny kvalitně detekovat i anomálie ve obzvláště vysoko-dimenzionálních datech.

Učení sítě cílí na minimalizaci rekonstrukční chyby. Rekonstrukční chyba je definována následovně [20]:

$$Err(i) = \sqrt{\sum_{j=1}^{D}(x_j(i) - \hat{x}_j(i))^2} \tag{1.4}$$

Kde:

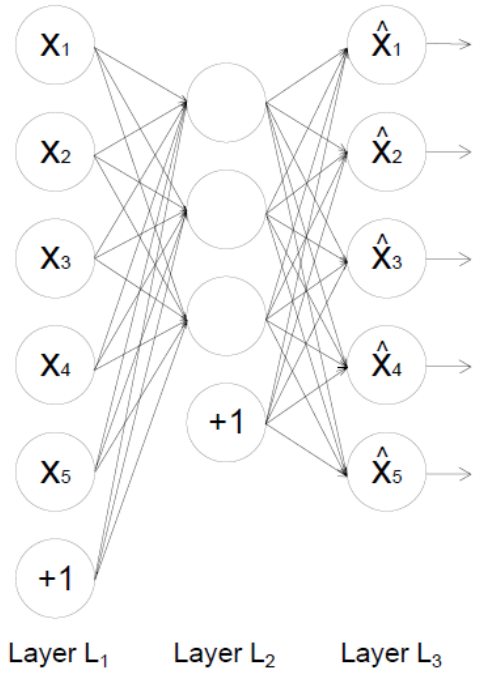
- $x_j(i)$ - j-tý příznak v i-tém příznakovém vektoru
- $\hat{x}_j(i)$ - ekvivalentní příznak, který je rekonstruován po plném průchodu sítí

Výsledky tří různých architektur založených na autoenkodérech byly porovnány v článku [27]. Výsledky lze vidět v tabulce 1.1.

Modifikace - převzorkování latentního prostoru

Toto modifikace řeší problém, že autoenkodér má často dostatečnou kapacitu a je schopen přesvědčivě rekonstruovat také anomální prvky [22] a tím efektivně sabotuje celou koncepci jeho využití pro detekci anomálií.

Řešením je podle autorů využití hlubokého autoregresivního modelu. Tím je omezen latentní prostor schopný generovat anomální vzory a je místo toho rekonstruován

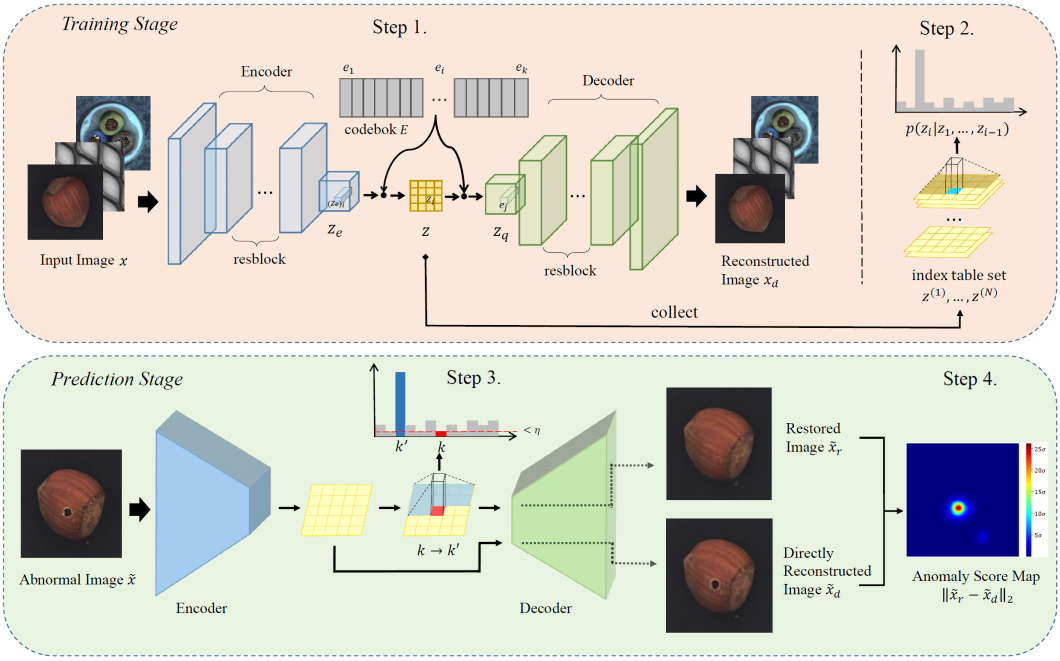


Obr. 1.4: Architektura autoenkodéru [20]

Tab. 1.1: Výsledky autoenkodérů na datasetu MNIST (přeloženo a zkráceno z [27])

	Třída	**Metoda**		
		Konvoluční autoenkodér	**Variační autoenkodér**	**Soupeřící autoenkodér**
$\mathbf{AUC_{ROC}}$	**0**	**0,9999**	**0,9999**	**0,9999**
	2	0,9933	0,9989	0,9974
	3	0,9964	0,9986	0,9966
	4	0,9980	0,9984	0,9974
	5	0,9979	0,9990	0,9982
	6	0,9942	0,9900	0,9959

normální vzorek, který je této anomálii nejpodobnější. Dále také pro detekci anomálií nespoléhají pouze na jednoduché prahování rekonstrukční chyby, ale hlouběji porovnávají rozdíl mezi rekonstrukcí s omezením latentního prostoru a bez něj [22].



Obr. 1.5: Autoenkodér s převzorkováním latentního prostoru - Architektura [22]

Výkonnost takto modifikované sítě byla otestována na datasetu MVTech. Oproti testování na MNIST nebo CIFAR-10 je výhodou, že tento dataset byl vytvořen přímo pro úlohu detekce anomálií, díky tomu lze těmto výsledkům přikládat mnohem větší váhu než v případě zbylých dvou zmíněných datasetů.

Z výsledků (viz tabulka 1.2) je patrné, že ve srovnání s ostatními metodami, které autoři testovali, navržený algoritmus podává velmi dobré výsledky s průměrnou úspěšností 85 %. Za zmínku stojí převážně porovnání s AnoGAN, který bude rozebrán později, ten měl úspěšnost o 30 % nižší [22].

Tab. 1.2: Autoenkodér s převzorkováním latentního prostoru - Výsledky (přeloženo a zkráceno z [22])

	Kategorie	Navržený klasifikátor	GAVGA-$\mathbf{D_u}$	$\mathbf{AE_{SSIM}}$	$\mathbf{AE_{L2}}$	AnoGAN
Textury	**Koberec**	0,71	**0,73**	0,67,	0,50	0,49
	Mřížka	**0,91**	0,75	0,69	0,78	0,51
	Dřevo	**0,96**	0,85	0,83	0,74	0,68
	Kůže	**0,96**	0,71	0,46	0,44	0,52
Předměty	**Láhev**	**0,99**	0,89	0,88	0,80	0,69
	Kabel	**0,72**	0,63	0,61	0,56	0,53
	Kapsle	0,68	**0,83**	0,61	0,62	0,58
	Tranzistor	**0,73**	**0,73**	0,52	0,71	0,67
	Průměr	**0,85**	0,78	0,63	0,71	0,55

1.3.2 GAN - Generative Adversial Network

GAN využívá dvě sítě, generátor a diskriminátor. V průběhu učení se generátor na základě šumu snaží vygenerovat obraz, o kterém se následně diskriminátor snaží prohlásit, zda jde o originál nebo synteticky vygenerovaný obraz. Cílem učení diskriminátoru je maximalizovat množství odhalených falešných dat ($\max_D$). Pro generátor se snažíme minimalizovat množství vygenerovaných falzifikátů, které diskriminátor korektně odhalí ($\min_G$). Tyto dvě sítě spolu tedy hrají minimax hru tak dlouho, dokud nedosáhnou Nashovy rovnováhy a ani jedna síť už není schopná získat výhodu nad sítí druhou [28].

Těchto vlastností můžeme využít také pro detekci anomálií. Přístupů, jak využít vlastností GAN sítí je více a některé z nich budou dále probrány.

Základní rovnice, popisující učení těchto sítí [25] je následující:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[logD(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[log(1 - D(G(\boldsymbol{z})))] \tag{1.5}$$

Kde:

- D - Síť diskriminátoru
- G - Síť generátoru
- **z** - Šumový vstupní vektor
- **x** - Skutečná vstupní data
- $\mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}$ - Entropie skutečných dat, která procházejí diskriminátorem
- $\mathbb{E}_{\boldsymbol{z}\sim p_z(z)}$ - Entropie šumových dat, která procházejí generátorem a následně diskriminátorem

Učení GAN sítí je v praxi problematickou záležitostí, jelikož se ukázalo, že mizející gradienty ve vysoko-dimenzionálních prostorech zamezují dosažení uspokojivých výsledků [18]. Tyto nestability se snaží vyřešit mnoho modifikací GAN sítí. Na základě toho vznikly metody jako WGAN [24] nebo DCGAN [29].

Architektura ADGAN

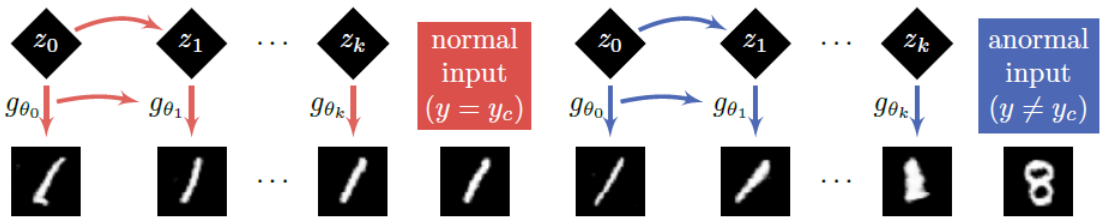
Tato architektura pracuje s předpokladem, že pokud byl generátor naučen na správné vzorky, pak se dá v latentním prostoru generátoru nalézt bod, který když projde generátorem, tak výstupem bude prvek velmi podobný vstupnímu OK vzorku. Tato podobnost je poté použita pro rozhodování o tom, zda se jedná o OK vzorek nebo anomálii. Jelikož v procesu učení neměla síť k dispozici anomální vzorky, tak generovaný NOK vzorek z latentního prostoru bude mít velkou odchylku od skutečného vzorku.

Jak je z popisu metody zřejmé, Diskriminátor se používá pouze v procesu učení. V procesu klasifikace už dochází pouze k porovnání generovaného a skutečného vzorku [18].

Algoritmus samotný nijak neupravuje učení GAN sítě. Vyhodnocování anomálií probíhá až po tom, co GAN síť na základě učících vzorků konvergovala.

Nyní je předložen vzorek, který má být klasifikován. Pomocí generátoru je vygenerována prvotní rekonstrukce. Z této rekonstrukce a vzorku se vypočte rekonstrukční chyba podle rovnice 1.4.

nyní se v k krocích rekonstrukční chyba zpětně propaguje. V tomto kroku je generátoru umožněna drobná parametrizace, aby bylo vyhodnocení flexibilnější, dle článku [18], tato úprava podává lepší výsledky než plné zmrazení generátoru po ukončení učení. V dalším kroku je pak vygenerovaná nová rekonstrukce, která by se měla více blížit předloženému vzorku (tuto zpětnou rekonstrukci blíže popisuje [30]). Jakmile je dosaženo k-té iterace, vzorek je označen za anomálii, pokud je rekonstrukční chyba větší než nastavený práh. úpravy provedené v generátoru v průběhu vyhodnocování jsou poté vráceny do stavu před vyhodnocováním.

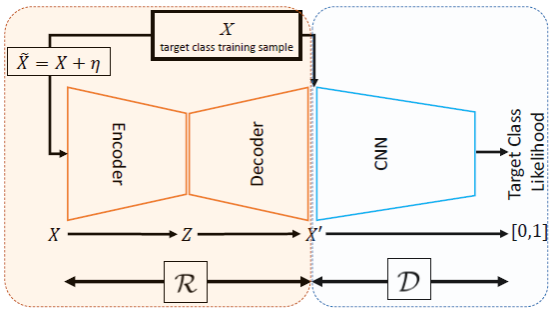


Obr. 1.6: ADGAN - Průběh rekonstrukce [18]

Výsledky autoři porovnali s dalšími metodami na datasetech MNIST a CIFAR-10, a to pro každou ze tříd, zatímco ostatní třídy sloužily jako anomálie. Toto porovnání může být trochu zavádějící, jelikož mezi třídami jsou často rozdíly mnohem větší, než jaké odpovídají reálným anomáliím. Z tohoto srovnání vyšel ADGAN v průměru jako nejlepší metoda [18].

Architektura - End-to-End GAN

Oproti předchozímu přístupu, tato metoda po naučení diskriminátor D nevyřazuje, ale používá jej jako detektor anomálií nebo také one-class klasifikátor. Generátor, který je v této architektuře značen jako R, potom podporuje diskriminátor tím, že zlepšuje kvalitu (rafinuje, rekonstruuje) OK vzorků, a naopak ještě více znehodnocuje anomální vzorky. Tím by mělo být dosaženo lepší separability dat pro diskriminátor [17].



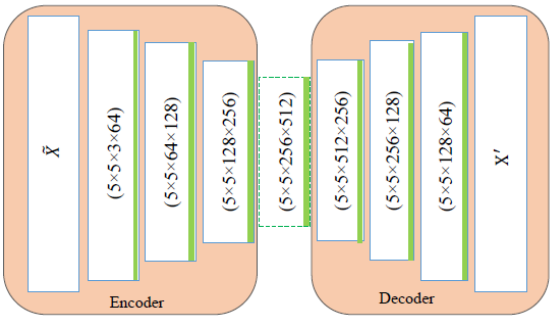
Obr. 1.7: End-to-End GAN - architektura [17]

Cílem učení je potom snaha o to, aby síť R byla schopna generovat přesvědčivé vzorky, které síť D nerozezná od skutečných, podobně jako v klasické GAN architektuře. Díky tomu, že jsou sítě učeny pouze za použití OK vzorů, se dá očekávat, že se síť D naučí tyto vzory dobře rozpoznat. Po předložení anomálního vzoru, by síť D měla s velkou jistotou rozpoznat netypický vzor.

Na druhé straně je síť R, která se naučí kvalitně rekonstruovat OK vzory, zatímco anomálie nebude schopna generovat přesvědčivě. Tím pádem bude síť OK vzorky dále rafinovat (například odstraní šum), zatímco NOK vzorky budou decimovány a jejich užitná hodnota se dále sníží [17]. Nutno podotknout, že vstupem sítě R není šum, jako v případě GAN sítí, ale testovaný vzorek, na který je uměle aplikován Gaussovský šum pro zvýšení robustnosti. Tuto skutečnost lze vidět na obrázku 1.7 zobrazujícím architekturu této sítě.

Architektura sítě R odpovídá architektuře autoenkodéru. Fungování autoenkodéru bylo popsáno výše, v kapitole 1.3.1. V rámci této architektury však není rekonstrukční chyba použita jako anomální ukazatel, ale síť slouží pouze pro úpravu vstupních dat takovým způsobem, aby data, která budou dále procházet sítí D, měla

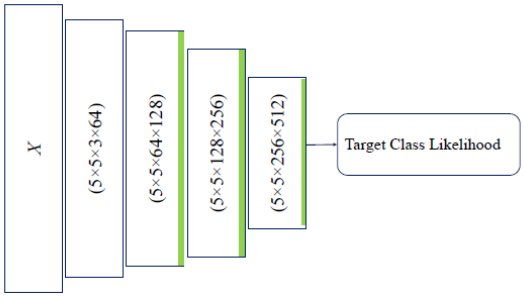
co největší odstup mezi normální a anomální třídou. Konkrétní použitá architektura je na následujícím obrázku 1.8.



Obr. 1.8: End-to-End GAN - Síť R [17]

Jelikož síť neprovádí klasifikaci, není velká úroveň generalizace žádoucí. Z toho důvodu neobsahuje síť R pooling vrstvy. Další zvýšení stability bylo dosaženo skrze normalizaci dávek [17].

Architektura sítě D je potom konvoluční sítí na jejímž výstupu by měla být informace o tom, zda předložený vzorek je nebo není anomálií.



Obr. 1.9: End-to-End GAN - Síť D [17]

Proces klasifikace probíhá jako jednoduché prahování výstupní hodnoty podle přeresolutionsu [17]:

$$OCC_2(X) = \begin{cases} Cílová\ Třída & \mathcal{D}(\mathcal{R}(X)) > \tau \\ Anomálie\ (Outlier) & jinak \end{cases} \tag{1.6}$$

Kde τ je ručně přednastavený práh, který je porovnáván s výstupem po průchodu vstupního vzorku sítí $\mathcal{R}$ a $\mathcal{D}$.

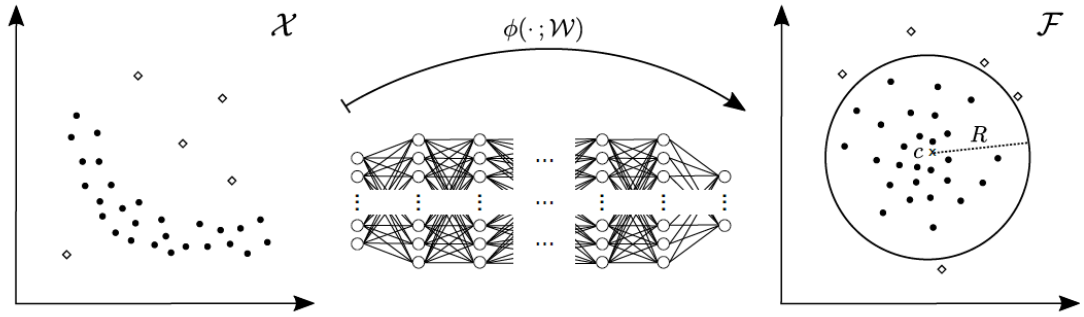
Experimenty autoři provedli na obrazových anomáliích na datasetech MNIST a Caltech256. Pro dataset Caltech256 dosahovala metoda vždy velmi dobrých výsledků, ve většině případů si vedla o 1-2 % lépe než ostatní metody, a to i v případech, kdy jako normální třída bylo zvoleno více různorodých tříd, se kterými bylo dále pracováno jako s jedinou [17]. Výsledky lze vidět v následující tabulce 1.3.

Tab. 1.3: End-to-End GAN - Experimenty Caltech256 - řádky představují modely, pro které byla z celkových 256 kategorií za normální třídu považována 1, 3 nebo 5 kategorií (přeloženo a zkráceno z [17])

Metoda	**CoP**	**Ourlier Pursuit**	**R-graph**	**Navržené D(R(x))**
AUC - 1 kategorie	0,905	0,837	**0,948**	0,942
AUC - 3 kategorie	0,676	0,788	0,929	**0,938**
AUC - 5 kategorií	0,487	0,629	0,913	**0,923**

1.3.3 DeepSVDD

Obvyklý přístup k detekci anomálií v obraze často využívá principů, které jsou pouze přejaty z jiných podobných úloh a využívají některé z jejich vlastností [11]. DeepSVDD se snaží tento problém řešit definicí cíle, který je přímo uzpůsoben úloze detekce anomálií a má tak potenciál dosahovat mnohem lepších výsledků [31].



Obr. 1.10: DeepSVDD - Základní princip [31]

DeepSVDD pracuje s myšlenkou, že je určena neuronová síť a optimalizační pravidlo. Optimalizační pravidlo se snaží minimalizovat hyperkouli, která obepíná

reprezentaci dat. Tato reprezentace je výstupem neuronové sítě. Minimalizace objemu nutí síť k tomu, aby ze vstupních dat extrahovala pouze společné příznaky což zapříčiní jejich mapování do blízkosti středu koule [31].

Učení probíhá současně jak pro extrakci příznaků, tak i pro jejich mapování do hyperkoule o minimálním objemu. Tento cíl lze popsat následujícím kritériem [31]:

$$\min_{R,\mathcal{W}} R^2 + \frac{1}{\nu n}\sum_{i=1}^{n} max\{0, ||\phi(x_i;\mathcal{W}) - c||^2 - R^2\} + \frac{\lambda}{2}\sum_{\ell=1}^{L}||\boldsymbol{W}^{\ell}||_F^2 \tag{1.7}$$

Kde cílem je nastavení vah $\mathcal{W}$ tak, aby byly všechny vzorky mapovány dovnitř hyperkoule o poloměru R, jejíž poloměr se současně s nastavováním vah snažíme minimalizovat. Ostatní parametry ve výše uvedeném přeresolutionsu mají následující význam:

- c - střed hyperkoule (jeho poloha může být také hyperparametrem)
- n - počet trénovacích vzorků
- $\nu \in (0; 1\rangle$ - jedná se o hyperparametr, který nastavuje váhu penalizace za prvky, které jsou mapovány vně hyperkoule na úkor jejího minimálního poloměru
- ϕ - označuje síť samotnou, která za pomocí vah $\mathcal{W}$ mapuje vstupní vzorky x ze vstupního prostoru $\boldsymbol{X}$ o rozměru $\mathbb{R}^d$ do výstupního prostoru $\mathcal{F}$ o rozměru $\mathbb{R}^p$
- λ - hyperparametr řídící rychlost rozpadu vah
- L - počet vrstev sítě

Jednotlivé členy kritéria mají potom následující význam:

- První člen - označuje poloměr hyperkoule, která obklopuje transformovaná data
- Druhý člen - Penalizace za vzorek, který se po průchodu sítí mapuje vně hyperkoule
- Třetí člen - řídí rychlost rozpadu vah [32]

Výsledkem této optimalizace je to, že normální vzorky jsou mapovány blízko ke středu hyperkoule, zatímco anomálie se mapují dál od středu nebo úplně mimo hyperkouli. Pokud předpokládáme, že všechna nebo naprostá většina učících dat jsou normální vzory, tak lze optimalizační kritérium zjednodušit následovně [31]:

$$\min_{\mathcal{W}} \frac{1}{n}\sum_{i=1}^{n} ||\phi(x_i;\mathcal{W}) - c||^2 + \frac{\lambda}{2}\sum_{\ell=1}^{L}||\boldsymbol{W}^{\ell}||_F^2 \tag{1.8}$$

Po této úpravě se již neminimalizuje poloměr hyperkoule, ale je pouze snaha o minimalizaci vzdálenosti mezi středem hyperkoule a polohou vzorku po jeho průchodu mapovací funkcí (sítí).

Mapovací funkce je obecně libovolná síť pro extrakci příznaků. Pokud jsou jako data použity například jednorozměrné signály, může být pro extrakci použita například umělá neuronová síť. V případě nárůstu dimenze vstupních dat (jako je například obraz, nebo vysoko-dimenzionální senzorická data) se nabízí využití konvolučních neuronových sítí s plně propojenou vrstvou na konci. Tím lze zaručit efektivní extrakci příznaků a mapování vysoko-dimenzionálních příznaků do prostoru o mnohem nižší dimenzi. Autoři sítě jako extraktor příznaků využívají architekturu LeNet [33], která se skládá ze tří konvolučních vrstev a plně propojené vrstvy.

Jak lze vidět z přeresolutionsů 1.7 a 1.8, při optimalizaci se výstup sítě neporovnává s očekávanou hodnotou (label), jako tomu je u standardní binární nebo vícetřídní klasifikace, ale je vypočítávaná vzdálenost od nastaveného středu.

Tento zdánlivě drobný rozdíl umožňuje síti efektivní učení za použití pouze jedné třídy (normální třída) a není tedy nutné v procesu učení znát možné anomálie, které navíc netvoří homogenní třídu, jak již bylo uvedeno v kapitole 1.1 a jak bude experimentálně ověřeno v kapitole 3.

Poloha středu c hyperkoule není pevně daná. Teoreticky může jít o libovolný bod v prostoru extrahovaných příznaků. Jelikož cílem je síť naučit tak, aby mapovala trénovací vzorky co nejblíže právě tomuto středu. I přesto dokáže vhodné prvotní nastavení středu zkrátit čas konvergence. Obvyklým přístupem je použití předtrénování, kdy se část sítě pro extrakci příznaků doplní o část pro zpětnou rekonstrukci dat a provede se vlastně standardní učení autoenkodéru. Za střed se potom zvolí výstup nejužšího místa autoenkodéru a prvotní váhy sítě se nastaví na hodnotu nejlepší epochy v procesu předtrénování. Tím je zajištěno, že hned v prvních epochách učícího procesu se budou prvky mapovat relativně blízko středu hyperkoule a metoda tak bude mít možnost rychleji konvergovat.

Polohu středu lze také brát jako volný parametr, který bude v průběhu učení také optimalizován. Zde je však nutné dát pozor na polohu v počátku stavového prostoru, jelikož pokud bude střed nastaven do tohoto bodu, může metoda nalézt triviální řešení, kdy nastaví všechny váhy sítě na 0 a bude potom na trénovacím datasetu vykazovat 100 % úspěšnost, ačkoliv nebylo dosaženo žádné relevantní extrakce příznaků.

Metoda má tu vlastnost, že pokud je špatně zvolena architektura sítě pro extrakci příznaků nebo střed hyperkoule, pak hrozí, že se DeepSVDD naučí triviální reprezentace, které nenesou informace kritické pro detekci anomálií [31]. Autoři sítě ve svém článku popisují 4 příklady, kdy k takovému stavu dojde, což by mělo při následném praktickém nasazení sítě zjednodušit její prvotní nastavení a pomoci vyhnout se těmto problémům.

Výsledky sítě a její výkonost byly otestovány na obrazových datasetech MNIST a CIFAR-10. Dále je použit dataset GTSRB obsahující dopravní značky Stop. Metoda

je porovnána s dalšími často používanými metodami. Výsledky porovnání lze vidět v následující tabulce 1.4.

Tab. 1.4: DeepSVDD - Experimenty (přeloženo a zkráceno z [31])

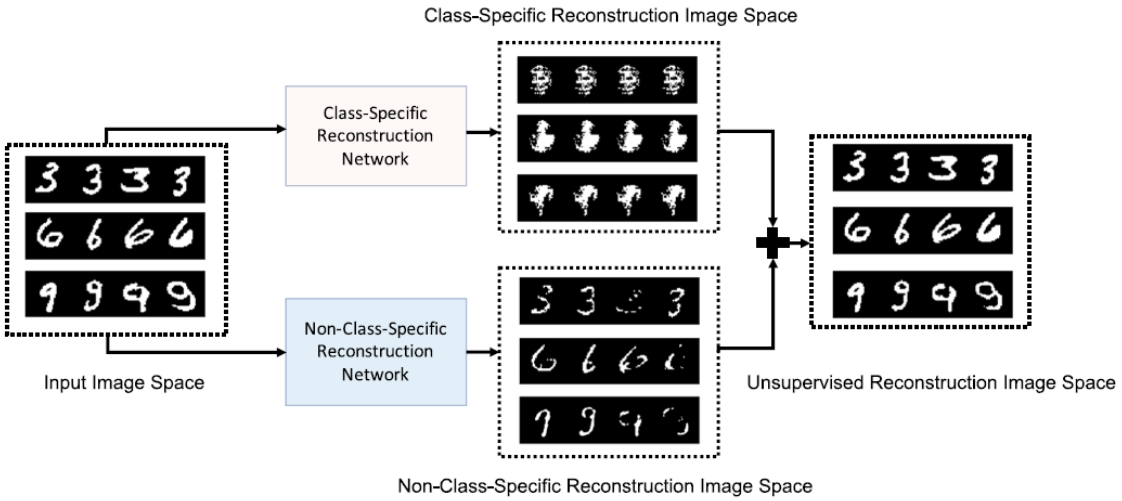
Normální třída	**OC-SVM**	**KDE**	**IF**	**DCAE**	**AnoGAN**	**One-class DeepSVDD**
0	**98,6**	97,1	98,0	97,6	96,6	98,0
1	99,5	98,9	97,3	98,3	99,2	**99,7**
2	82,5	79,0	88,6	85,4	85,0	**91,7**
3	88,1	86,2	89,9	86,7	88,7	**91,9**
4	**94,9**	87,9	92,7	86,5	89,4	**94,9**
Letadlo	61,6	61,2	60,1	59,1	**67,1**	61,7
Auto	63,8	64,0	50,8	57,4	54,7	**65,9**
Pták	50,0	50,1	49,2	48,9	**52,9**	50,8
Kočka	55,9	56,4	55,1	58,4	54,5	**59,1**
Srna	66,0	**66,2**	49,8	54,0	65,1	60,9

1.3.4 DDR-ID

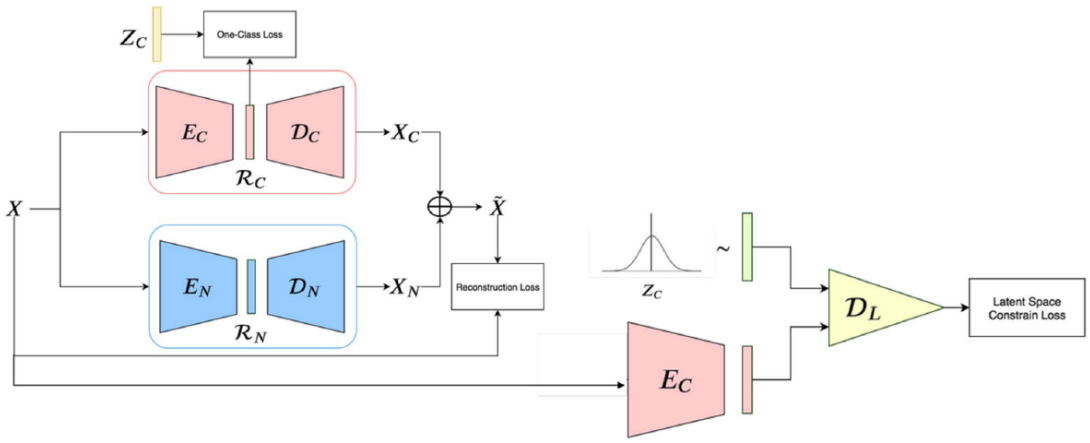
Tato architektura oproti ostatním staví na optimalizaci tří ztrátových funkcí v průběhu učení [34]: jednotřídní ztráta, ztráta v omezení latentního prostoru, a rekonstrukční chyba. V průběhu klasifikace potom síť předložený vzor dekomponuje do části, která reprezentuje normální vzor, a reziduum, které představuje anomalitu daného vzoru. Konečné ohodnocení anomality je potom vypočteno jak z normální části, tak i z rezidua. Toto skóre lze pak prahovat pro oddělení anomálních a normálních vzorů.

Síť je učena end-to-end s přímou optimalizací úlohy detekce anomálií. Uvnitř sítě najdeme dva kodéry-dekodéry. Jedna z těchto sítí, označována jako $\mathcal{R}_C$, má za úkol rekonstruovat co nejlépe normální vzory a představuje tedy class-specific část. V zásadě jde o extrakci podobností, mezi normálními vzorky. Druhá síť, která je označena $\mathcal{R}_N$, potom extrahuje reziduum mezi rekonstruovaným vzorkem a vstupním vzorem a představuje tak non-class-specific komponentu. Po průchodu těmito sítěmi získáme dvě komponenty vstupního obrazu - normální a anomální část. Tyto části jsou poté spojeny a je vytvořena výsledná rekonstrukce [34].

Enkodér se skládá z několika konvolučních vrstev, které jsou nakonec napojeny na vrstvu plně propojenou - tím je docíleno vtištění informací obsažených v obraze do



Obr. 1.11: DDR-ID - koncepce (převzato z [34])



Obr. 1.12: DDR-ID - Architektura [34]

jednorozměrného latentního vektoru. Dekodér je potom reciprokou verzí enkodéru. Autoři empiricky vysledovali, že použití architektury odvozené z GAN sítí podávalo lepší výsledky než jednoduchý autoenkodér [34]. Pro účely vyhodnocení se proto dále do architektury přidává pomocná síť diskriminátoru označovaná jako $\mathcal{D}$, ta je naučena na rozpoznání normálního vzorku z jeho latentní reprezentace. Samotné sítě jsou propojeny ve velmi podobném stylu jako GAN sítě, které byly popsány dříve v kapitole 1.3.2.

Učení probíhá ve dvou fázích. První fází učení této sítě je předtrénování. V této fázi jsou sítě $\mathcal{R}_C$ a $\mathcal{R}_N$ učeny jako standardní autoenkodéry pro rekonstrukci obrazu, symbolický zápis [34] je uveden v rovnicích 1.3.4 a 1.3.4. Tato počáteční fáze slouží ke dvěma účelům. Prvním je snaha o naučení sémanticky důležitých reprezentací,

které jsou důležité pro kvalitní rekonstrukci a druhým je potom selekce latentní reprezentace, která co nejlépe vystihuje normální vzor.

$$\min_{\theta_{\mathcal{R}_C}} \frac{1}{I} \sum_i ||X_i - \mathcal{R}_C(X_i; \theta_{\mathcal{R}_C})||^2 \tag{1.9}$$

$$\min_{\theta_{\mathcal{R}_N}} \frac{1}{I} \sum_i ||X_i - \mathcal{R}_N(X_i; \theta_{\mathcal{R}_N})||^2 \tag{1.10}$$

Kde:

- X_i - i-tý trénovací vzorek
- $\mathcal{R}_C/\mathcal{R}_N$ - Sítě autoenkodéru
- $\theta_{\mathcal{R}_C}/\theta_{\mathcal{R}_N}$ - Parametry sítí
- I - množství trénovacích vzorů (normální třída)

Jakmile je předtrénování dokončeno, je vypočten vektor Z_C, což je průměrná latentní reprezentace normálního vzoru. Tím je vytvořena reprezentace normální třídy v latentním prostoru autoenkodéru $\mathcal{R}_C$.

Nyní následuje optimalizační fáze učení. Zde byly navrženy tři ztrátové funkce, které jsou dále optimalizovány. Jedná se o jednotřídní ztrátovou funkci, ztrátovou funkci omezení latentního prostoru a rekonstrukční chybu.

Jednotřídní ztrátová funkce vychází z funkce navržené v [31], která penalizuje vzdálenost latentní reprezentace vzorku X_i od průměrné reprezentace Z_C, která byla vypočtena na konci fáze předtrénování. Tuto funkci popisuje následující rovnice [34]:

$$\mathcal{L}_{OC}(E_C) = \frac{1}{I} \sum_i ||E_C(X_i; \theta_{E_C}) - Z_C||^2 \tag{1.11}$$

Kde:

- X_i - i-tý trénovací vzorek
- $E_C(X_i; \theta_{E_C})$ - latentní reprezentace vzorku
- Z_C - průměrná reprezentace normálního vzorku
- I - množství trénovacích vzorů (normální třída)

Ztrátová funkce omezení latentního prostoru vychází z definice ztrátové funkce GAN sítí (viz kapitola 1.3.2). Oproti GAN sítím však nejde o snahu vytvořit věrohodnou reprezentaci ze šumových dat, ale je snaha síť naučit tak, aby latentní reprezentace vstupních dat (vektorový výstup enkodérové části v síti $\mathcal{R}_C$) co nejlépe odpovídala průměrné latentní reprezentaci Z_C získané na konci fáze předtrénování. Tuto část sítě lze z pohledu architektury vidět na obrázku 1.12 vpravo. Ztrátová

funkce je potom definována následovně [34]:

$$\begin{aligned}\mathcal{L}_{LSC}(E_C, \mathcal{D}_L) = \mathbb{E}_{\boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{Z_C},\sigma^2\boldsymbol{I})}[log\mathcal{D}_L(\boldsymbol{Z};\theta_{\mathcal{D}_L})]+ \\ +\mathbb{E}_{\boldsymbol{X}\sim p_{data}}[log(1-\mathcal{D}_L(E_C(\boldsymbol{X};\theta_{E_C});\theta_{\mathcal{D}_L}))]\end{aligned} \tag{1.12}$$

Kde:

- $\mathcal{D}_L$ - Síť diskriminátoru
- $\mathbf{Z}$ - Vektor normálního rozložení odvozený z průměrného latentního vektoru Z_C
- $\theta_{\mathcal{D}_L}$ - Parametry diskriminátoru
- $\mathcal{N}(\boldsymbol{Z_C}, \sigma^2\boldsymbol{I})$ - Normální rozložení se středem odpovídajícím latentnímu vektoru a malou hodnotou [34] rozptylu
- $\mathbf{X}$ - Vstupní data (obraz)
- E_C - Enkodérová část sítě $\mathcal{R}_C$
- θ_{E_C} - Parametry enkodéru

Třetí ztrátovou funkcí je potom rekonstrukční chyba. Tato chyba je svou definicí podobná chybě použité u autoenkodérů (kapitola 1.3.1). Jedná se o výpočet rozdílu mezi vstupním vzorkem a jeho rekonstruovaným protějškem. Symbolický zápis je poté následující [34]:

$$\mathcal{L}_R(\mathcal{R}_C, \mathcal{R}_N) = \frac{1}{I}\sum_i ||\tilde{x}_i - x_i||^2 \tag{1.13}$$

Kde:

- $\tilde{x}_i$ - i-tý rekonstruovaný vzorek vzniklý jako $\mathcal{R}_C(X_i;\theta_{\mathcal{R}_C}) + \mathcal{R}_N(X_i;\theta_{\mathcal{R}_N})$
- x_i - i-tý vstupní vzorek
- I - počet trénovacích vzorků

Optimalizace potom probíhá pro všechny tyto tři ztrátové funkce současně podle přeresolutionsu [34]:

$$\min_{\theta_{\mathcal{R}_C},\theta_{\mathcal{R}_N}} \max_{\theta_{\mathcal{D}_L}} \mathcal{L}_{OC} + \mathcal{L}_{LSC} + \mathcal{L}_R \tag{1.14}$$

Klasifikace již nevyužívá síť $\mathcal{R}_N$, ta je zásadní pouze ve fázi učení, jelikož pomáhá s optimalizací cíle anomální detekce [34]. Je využito pouze sítí $\mathcal{R}_C$ a $\mathcal{D}_L$. Autoři sítě definují dva druhy ohodnocení anomálie, které jsou popsány dále.

Latentní ohodnocení anomálie

Z latentního prostoru daného vzoru X je spočítána euklidovská vzdálenost oproti průměrnému latentnímu vektoru Z_C, který představuje naučenou normální reprezentaci. Latentní ohodnocení anomálie tedy nerekonstruuje předložený vzor zpět do obrazové podoby. Toto srovnání lze zapsat následovně [34]:

$$AS_l(X^t) = ||E_C(X^t; \theta_{E_C}) - Z_C||^2 \tag{1.15}$$

Rekonstrukční anomální skóre

Oproti DeepSVDD, zde není po předtrénování síť dekodéru zahozena a rekonstrukční chyba je vypočtena jako vzdálenost mezi class-specific rekonstruovaným obrazem (testovaný vzor projde celou sítí $\mathcal{R}_N$) a vzorem, který byl sítí naučen v průběhu učení (latentní reprezentace Z_C projde pouze diskriminátorem $\mathcal{D}_L$). Výpočet skóre je zapsán v následujícím vztahu [34]:

$$AS_r(X^t) = ||\mathcal{R}_C(X^t; \theta_{\mathcal{R}_C}) - \mathcal{D}_C(Z_C; \theta_{\mathcal{D}_C})||^2 \tag{1.16}$$

Výsledné vyhodnocení, zda je vzorek anomální, poté probíhá jako pouhé prahování anomálního skóre 1.3.4[34]. Je možné využít jednu z navrhovaných metrik, ale lepších výsledků lze dosáhnout kombinací těchto ukazatelů [34].

$$\begin{cases} Cílová\ Třída & AS(X^t) < \tau, \\ Anomálie\ (Outlier) & jinak \end{cases} \tag{1.17}$$

Výsledky srovnání několika architektur jsou uvedena v následující tabulce 1.5.

Tab. 1.5: DDR-ID - Výsledky MNIST a CIFAR-10 (přeloženo a zkráceno z [34])

Metoda	**Dataset**	
	MNIST	**CIFAR-10**
OC-SVM/SVDD	91,3	64,8
KDE	86,8	64,9
AnoGAN	91,3	61,8
One-class DeepSVDD	94,8	64,8
Navržená DDR-ID	**96,2**	**65,4**

1.3.5 Srovnání architektur

Následující porovnání srovnává výsledky, kterých jednotlivé metody dosáhly na datasetech MNIST a CIFAR-10. Nutno podotknout, že tyto výsledky jsou pouze kompilací z různých článků, a tudíž jich mohlo být dosaženo za mírně odlišných podmínek. V případě, že byla daná architektura použita ve více článcích je uveden průměr z těchto výsledků.

Tab. 1.6: Srovnání metod - MNIST a CIFAR-10

	MNIST	**CIFAR-10**
ANOGAN	91,89	61,62
ADGAN	96,8	63,4
DCAE	93,19	61,35
DDR-ID	96,2	**65,4**
CAE	**99,25**	55,6
D(R(X))	96,8	-
DeepSVDD	93,84	63,62

Z výsledků (tabulka 1.6) lze vidět, že síť CAE (Konvoluční autoenkodér) podávala nejlepší výsledky na datasetu MNIST. Jedná se o síť, která patří do skupiny sítí, které extrahují příznaky definující normalitu.

Síť ADGAN, náležející do stejné skupiny, dosahovala také dobrých výsledků. Tato síť vnitřně také používá architekturu autoenkodéru, v průběhu učení však využívá soutěžní princip typický pro GAN sítě.

Co se týče sítí ze třetí skupiny, neboli sítí, které v průběhu učení přímo optimalizují úlohu detekce anomálií, tak zde si nejlépe vedla síť DDR-ID následovaná sítí DeepSVDD. Výsledky dosažené těmito sítěmi jsou horší, než v případě druhé skupiny. Tato skutečnost je způsobena mimo jiné faktem, že teoretický základ pro tento typ sítí je mnohem slabší a tyto metody tak jsou spíše průkopníky v tomto směru detekce anomálií. I přesto mají tyto sítě velký potenciál pro optimalizaci už jen v rámci modifikace hyperparametrů.

1.4 Evaluace klasifikátorů

Pro účely srovnání jednotlivých modelů a výstupů z experimentů budou v práci využity standardní hodnotící prostředky. Mezi tyto prostředky patří předně matice záměn a z ní vycházející metriky, dále ROC křivka a z ní vycházející AUC metrika.

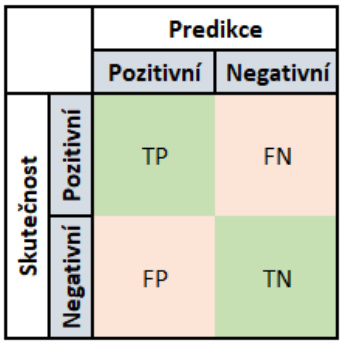
V následujících podkapitolách budou tyto prostředky blíže představeny a bude krátce popsáno, jak se významnost jednotlivých metrik může při hodnocení jednotřídních klasifikátorů lišit oproti binárním nebo vícetřídním klasifikační úlohám.

1.4.1 Matice záměn

Jedná se o metodu navrženou primárně pro vyhodnocení binárních klasifikátorů [35]. Tuto metodu lze použít také pro hodnocení detektorů anomálií. Je však nutné mít

na paměti, že nevyvážená povaha anomálií vůči normální třídě, co se počtu vzorů týče, může zkreslovat představu o skutečných vlastnostech modelu.

Následující obrázek 1.13 ukazuje matici záměn. Ve sloupcích se nachází hodnota vystupující z modelu, zatímco řádky obsahují skutečnou anotaci dat. Jednotlivá pole potom popisují počet daných kombinací predikce x skutečnost pro daný model a testovací dataset.



Obr. 1.13: Matice záměn - obecný příklad

Z těchto hodnot lze potom vypočítat řadu metrik. Mezi nejpoužívanější patří následující:

- **ACC - Accuracy** - Správnost udává počet správně zařazených vzorů vzhledem k celkovému počtu vzorů.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.18}$$

- **TPR - True Positive Rate** - V češtině také označována jako senzitivita. Udává schopnost správně detekovat pozitivní prvky. je definována následovně:

$$TPR = \frac{TP}{TP + FN} \tag{1.19}$$

- **TNR - True Negative Rate** - Neboli specificita. Označuje, jak robustně dokáže model označit negativní třídu. Definice je následovná:

$$TNR = \frac{TN}{FP + TN} \tag{1.20}$$

- **FPR - False Positive Rate** - Jedná se o metriku popisující chybu 1. druhu. Neboli prvky negativní třídy, které byly modelem označeny za pozitivní třídu. Jde o doplněk ke specificitě.

$$FPR = \frac{FP}{FP + TN} \tag{1.21}$$

- Existuje řada dalších metrik, které však nejsou v kontextu této práce relevantní a nebudou tedy dále rozebírány

Pro účely detekce anomálií jsou nejvýznamnější metriky senzitivity (schopnost skutečně detekovat anomálie) a FPR (množství normálních vzorů, které byly neprávem označeny za anomálie). Jak již bylo nastíněno výše, metriky vypočtené z matice mohou být v případě nevyváženého datasetu zavádějící, což lze demonstrovat na následujícím extrémním případu, definovaném maticí záměn na obrázku 1.14.

		Predikce	
		Pozitivní	Negativní
Skutečnost	Pozitivní	5	5
	Negativní	0	100

Obr. 1.14: Matice záměn, z níž vypočtené metriky mají zavádějící charakter

Modely bývají často hodnoceny pouze metrikou správnosti. V tomto případě by šlo o hodnotu $ACC = 95,45$ % což při pohledu na počty vzorů v jednotlivých buňkách matice nejsou reprezentativní výsledky, neboť model je schopen detekovat pouze $TPR = 50$ % anomálií, což je pro detektor anomálií nepřijatelná hodnota.

Pro relevantní popis modelu je tedy nutné uvést správné metriky, kterými jsou v tomto případě přednostně *senzitivita* a FPR. Stále se však jedná jen o jedno nastavení modelu, které například v případě DeepSVDD odpovídá jednomu nastavení hranice vzdálenosti od středu hyperkoule, po jejímž překročení budou prvky označeny za anomálií.

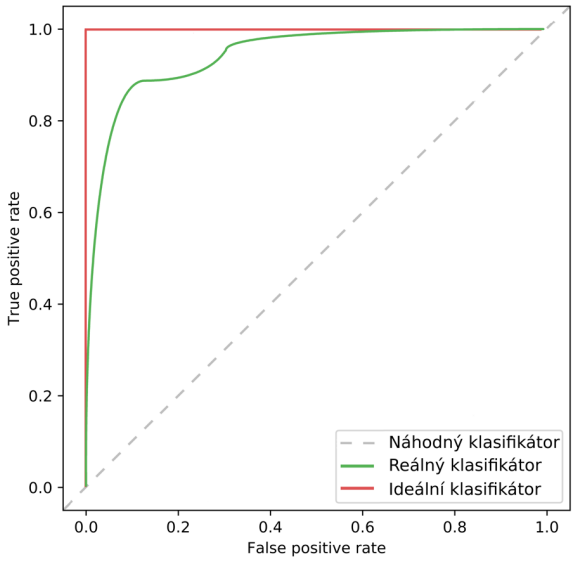
Je tedy zřejmé, že výběr reprezentativní matice záměn pro srovnání více modelů mezi sebou není jednoznačná záležitost. V rámci této práce bylo nastavení hranice prováděno tak, že po vyhodnocení všech vzorů v testovacím datasetu byl zvolen bod, kde bylo správně detekováno 90 % anomálií a pak bylo pozorováno, jak moc narostl počet falešně detekovaných anomálií (jak velký kompromis musel být přijat pro detekování 90 % anomálií).

1.4.2 ROC křivka

Mnohem lepším přístupem ke srovnání modelů je ROC křivka a AUC metrika, která z ní vychází. ROC křivka podává grafický pohled na postupný vývoj kompromisu mezi počtem správně detekovaných anomálií a počtem falešně detekovaných anomálií. Měnícím se parametrem pro vykreslení křivky je volba hraniční vzdálenosti od

středu hyperkoule. ROC křivka tak dává ucelený pohled na vývoj dvou nejdůležitějších metrik v rámci detekce anomálií v závislosti na změně prahování výstupních hodnot z modelu.

S ROC křivkou se dále pojí metrika AUC (Area Under Curve - plocha pod křivkou), která říká kolik % ideálního ROC grafu pokrývá vyhodnocovaný model. Ideální ROC křivku lze vidět na obrázku 1.15. V zásadě jde o model, který správně klasifikuje všechny vzory pozitivní třídy pro všechna nastavení prahu výstupních hodnot.



Obr. 1.15: Srovnání ROC křivky ideálního a reálného klasifikátoru

2 Praktická část - inspekční systém

Cílem této kapitoly je seznámit čtenáře s technickými prostředky, jejich výchozím stavem a případnými úpravami, které bylo nutné provést, aby bylo možné provádět následné experimenty, které budou součástí následující kapitoly 3.

Na základě zkoumaných metod v kapitole 1.2 bylo nutno vybrat vhodnou metodu, se kterou se bude v této práci dále operovat. Jako metoda byla po domluvě s vedoucím zvolena DeepSVDD, která byla rozebrána v kapitole 1.3.3.

Tato metoda byla zvolena z několika důvodů. Jelikož se jedná o metodu spadající do skupiny metod s přímou definicí anomálního skóre, tak bude přínosné vyšetřit vliv definice anomálií (viz bod 6 zadání) na výkon metody spadající do této méně vyzkoumané skupiny [11]. Z praktického hlediska je potom výhodou, že k metodě existuje dobře zpracovaná a dokumentovaná implementace.

V neposlední řadě byla metoda zvolena také proto, že se využívá v dalších projektech v rámci skupiny počítačového vidění a výstupy a poznatky pramenící z této práce mohou být použity v další výzkumné činnosti.

Tato kapitola se dále dělí na podkapitolu definující samotnou úlohu, její specifika, omezení a předpoklady. Dále je krátce popsán hardware přípravku pro průmyslovou inspekci a detekci anomálií společně s úpravami, které na něm byly provedeny. Třetí část potom popisuje použitý software, kterým je implementace samotné metody DeepSVDD a také uživatelská aplikace Anubis sloužící pro ovládání kamer, platformy, sbírání datasetu a také provádění predikce na základě naučeného modelu. Poslední část této kapitoly se bude zabývat realizací vlastního datasetu, který bude představovat průmyslový výrobek pohybující se na dopravníkovém pásu.

2.1 Popis úlohy

Jak bylo uvedeno již v úvodu této práce, tlak na rychlou a kvalitní detekci anomálií se stále zvyšuje. Je proto nutné vytvořit systémy, které dokáží tyto požadavky splnit. Cílem praktické části této práce je tedy na poskytnuté platformě pro průmyslovou inspekci (viz kapitola 2.2) vytvořit scénu, která bude simulovat dopravníkový pás na výrobní lince. Tento pás je simulovaný kruhovým karuselem a umožňuje tak simulovat dopravníkový pás generováním nekonečné smyčky.

Do takto vytvořené scény budou potom vstupovat vzorky průmyslových výrobků. Pro tuto práci byl jako demonstrační objekt po domluvě s vedoucím zvolen produkt potravinářského průmyslu. Konkrétně se potom jedná o slepované sušenky značky Lotus (EAN: 5410126006377).

Výrobky pohybující se na tomto přípravku jsou snímány dvěma kamerami, přičemž každá z kamer představuje jiný úhel pohledu. Jako reprezentativní byly zvoleny

pohledy zboku a shora, ale při nasazení obdobného systému do provozu mohou být zvoleny jiné pohledy, nebo může být jejich počet zvýšen (pro detekci vad, které jsou na ostatních pohledech skryty). Je však nutné brát v potaz, že každý přidaný pohled vyžaduje samostatný trénovací dataset (jeho tvorba bude blíže popsána v kapitole 2.4).

Smysl použití vícepohledového systému tkví v představě, že některé vady na výrobku mohou při pouze jednom pohledu zůstat skryty a že informace z různých pohledů dokáží svou synergií předčit dva samostatné jednopohledové systémy. V rámci této práce však nebude zkoumán vliv fúze dat z více kamer, ale vliv různého dělení datasetu na výsledný odhad přesnosti sítě, vždy pro každý z pohledů samostatně.

S touto definicí úlohy lze nyní uvést cíle, kterých se tato práce snaží dosáhnout. Tyto cíle lze rozdělit do dvou skupin. První skupinou jsou cíle, které tuto úlohu realizují. K této práci byl poskytnut laboratorní přípravek a software, která má být využit. Pro realizaci úlohy bylo nutné tento software a hardware vhodně upravit, aby byly splněny požadavky úlohy. Tyto změny a úpravy jsou dokumentovány v kapitolách 2.2 a 2.3. Dále bylo nutné navrhnout a následně vytvořit samotný dataset. Jeho tvorba je popsána v kapitole 2.4.

Druhou skupinou jsou experimenty provedené na vytvořeném datasetu. Zde je cílem zjistit, jaký vliv má definice anomálií (ať už ve fázi učení nebo testování) na odhad výkonu klasifikátoru. Těmito experimenty se dopodrobna zabývá kapitola 3.

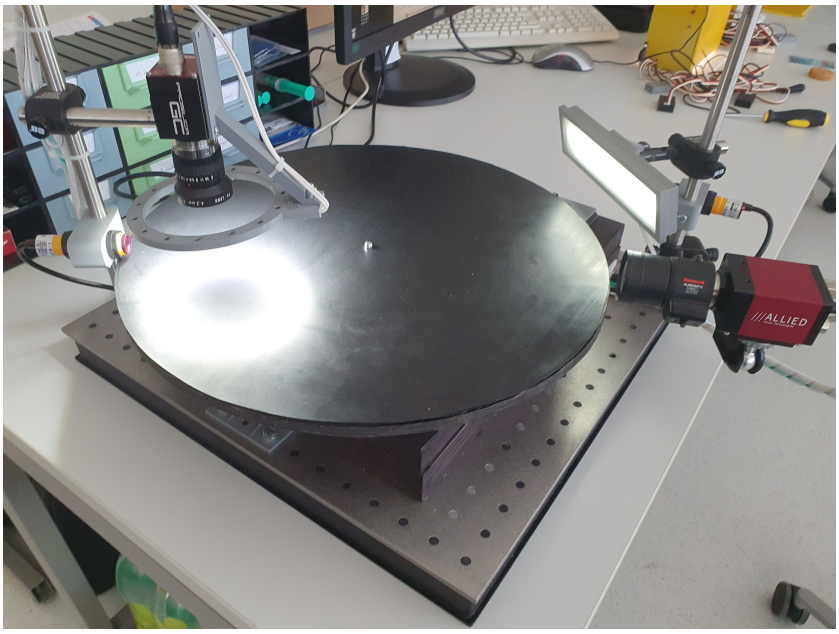
2.2 Hardware

Platforma, na které byla realizována úloha popsaná v předchozí kapitole je ve svém výchozím stavu zobrazena na obrázku 2.1. Platforma po provedení úprav, které budou popsány dále, je potom zobrazena na obrázku 2.2.

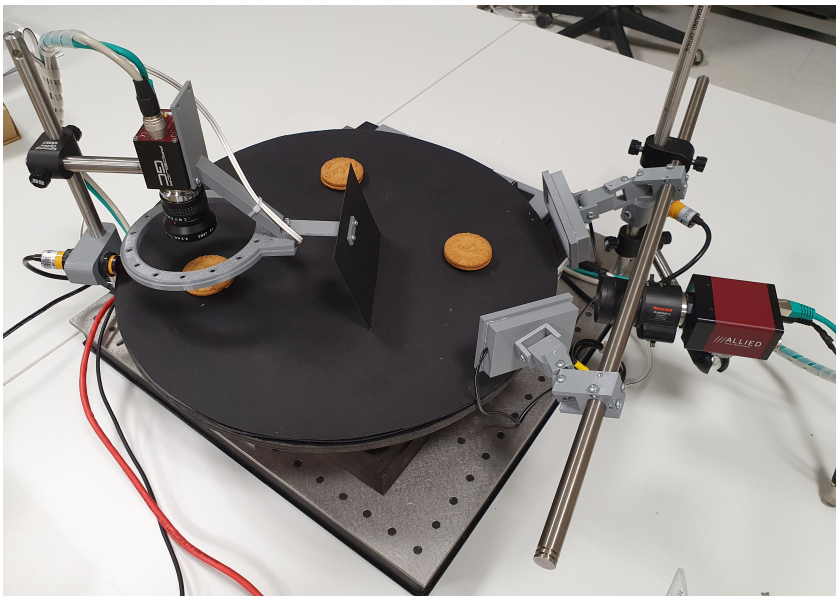
Platforma sestává ze dvou kamerových stanovišť, přičemž každé obsahuje kameru, nezávisle ovládané osvětlení a ultrazvukový snímač přítomnosti objektu. Výrobky se potom pohybují po kruhové dráze na otočném karuselu, který je poháněn krokovým motorem a jeho otáčky lze plynule regulovat. Tyto prvky a jejich případné modifikace budou blíže popsány v následujících kapitolách.

2.2.1 Otočná platforma

Otočný karusel je, jak už bylo zmíněno, poháněn krokovým motorem a realizuje nekonečnou smyčku. Původní realizace obsahovala povrch z lesklé gumy. Při pohledu na snímek 2.3a, který byl pořízen s tímto povrchem, je vidět, že takto vytvořená scéna výrazně odráží světlo od podložky do kamery, což výrazně degraduje jasové



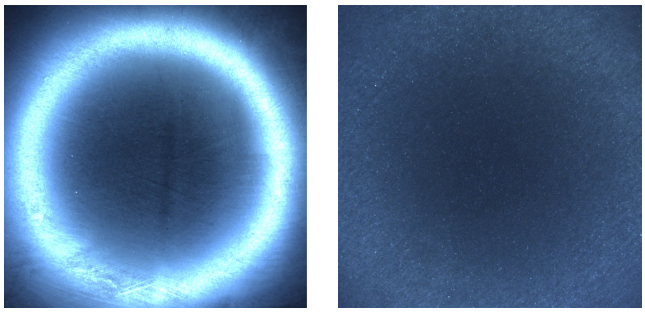
Obr. 2.1: Platforma pro průmyslovou inspekci - výchozí stav



Obr. 2.2: Upravená platforma pro průmyslovou inspekci

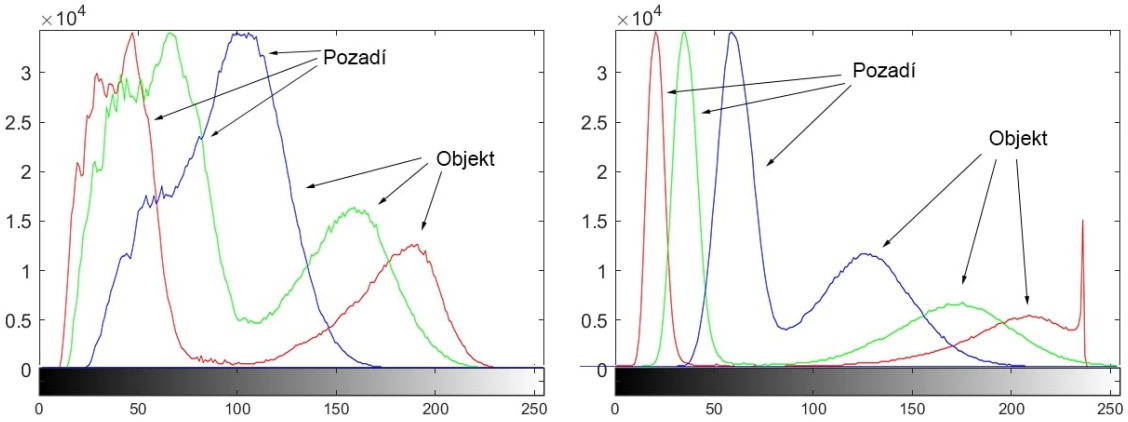
poměry ve scéně. Tato podložka byla tedy vyměněna za matný černý papír, který tyto odlesky potlačuje. Černá barva byla zvolena jako reprezentativní pro skutečný pás na existující výrobní lince, na kterou by systém mohl být instalován dodatečně. Snímek z modifikované scény je vidět na obrázku 2.3b. Na obrázku 2.4 lze vidět porovnání histogramů snímků se starou a novou podložkou. Je vidět, že použití

matné podložky zajistilo lepší odstup obrazových dat zájmového objektu od pozadí. Při použití původní podložky nebylo například z histogramu modrého kanálu vůbec možné rozpoznat, zda se ve scéně objekt nachází nebo ne.



a) b)

Obr. 2.3: Snímky povrchu platformy - a) před úpravou - lesklý povrch, b) po úpravě - matný povrch



a) b)

Obr. 2.4: Srovnání histogramů pro snímky s různým povrchem (snímek s objektem). a) lesklý povrch - jasové úrovně objektu a pozadí se překrývají, b) matný povrch - jasové úrovně objektu a pozadí jsou jasně rozlišitelné

2.2.2 Kamery

Platforma je připravena pro osazení GigE Vision kamerami (ačkoliv po vyvedení vhodného konektoru není omezena pouze na tento standard). Kamery mohou být napájeny vlastním zdrojem, nebo může být využito vyvedeného standardního konektoru typu Hirose HR10A [36]. Při použití tohoto konektoru je však nutné nejprve

zkontrolovat, že zapojení jednotlivých pinů odpovídá specifikaci výrobce kamery a případně tyto piny přepojit uvnitř platformy.

Pro upevnění kamer je využit stavebnicový systém [37] společně s tyčemi pro upevnění kamer, světel a dalších HW prvků snímací soustavy od Edmund Optics [38], který umožňuje posun a natočení kamer v rámci platformy. V rámci úlohy realizované v této práci jsou kamery umístěny tak, že realizují horní a boční pohled na simulovaný dopravníkový pás. Bližší informace o konkrétních kamerách a objektivech použitých pro realizaci datasetu a následnou detekci anomálií v reálném čase jsou uvedeny v kapitole 2.4.2.

2.2.3 Osvětlení

Platforma obsahuje dva nezávisle ovládané zdroje světla. Jeden je kruhový a slouží pro nasvícení scény při pohledu shora. Druhý zdroj je lineární a slouží pro nasvícení bočního pohledu.

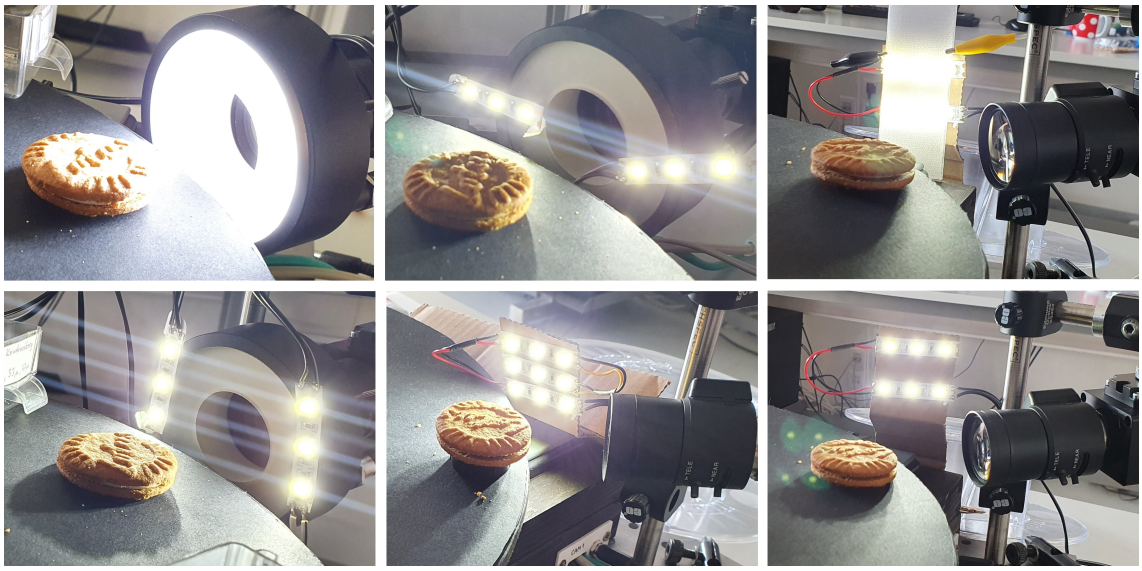
Zdroj pro horní pohled nebyl nijak modifikován. Tento světelný zdroj je realizován kruhovým LED osvětlením s průměrem 120 mm a typovým označením DPD-R120-R104-0417. Tento zdroj má definovanou barevnou teplotu v rozsahu 6000-6500 K s vyzařovacím úhlem 120°.

Zdroj bočního světla sestával původně z dlouhého lineárního segmentu (viz obrázek 2.1), který znemožňoval nasvítit objekt rovnoměrně z obou stran, navíc neumožňoval dostatečně volnou manipulaci s polohou vůči kameře.

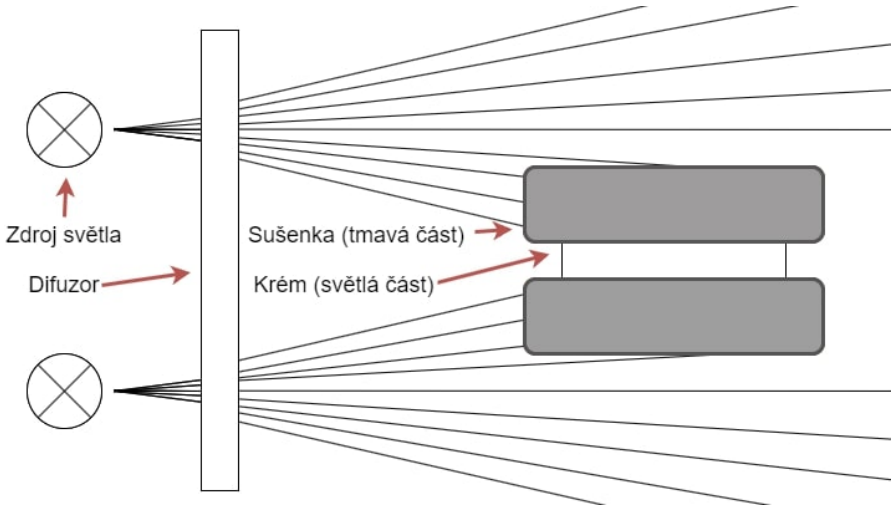
Z toho důvodu byl navržen nový světelný segment. V prvé řadě byly vytvořeny různé experimentální konfigurace osvětlení s cílem určit nejvhodnější polohu a distribuci světelných segmentů. Několik těchto experimentálních konfigurací lze vidět na obrázku 2.5.

Jako nejvhodnější byla zvolena konfigurace, kdy jsou světelné segmenty umístěny po obou stranách kamery, přičemž každý světelný segment je realizován dvěma řádky LED pásků typu KU-5050AD [39] tak, aby byla tmavá část výrobku (sušenka) osvícena co možná nejlépe a světlá část (krém) byl osvětlený méně. Tím je maximalizováno využití dynamického rozsahu kamery při současné minimalizaci saturovaných bodů na zájmovém objektu. Koncepci tohoto osvětlení znázorňuje obrázek 2.6. Srovnání snímků s původním a novým osvětlením lze vidět na obrázku 2.7.

Po nalezení nejvhodnější konfigurace byl navržen držák, který byl následně vytištěn na 3D tiskárně. Tento držák lze vidět na obrázku 2.8. Oproti původnímu držáku, je tvořen ramenem se čtyřmi klouby, které umožňují precizně nastavit polohu a natočení světelného zdroje ve scéně.



Obr. 2.5: Experimentální konfigurace pro volbu vhodné konfigurace osvětlení pro scénu bočního pohledu na zájmový objekt

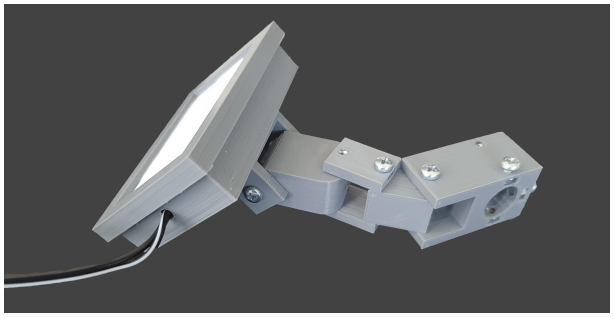


Obr. 2.6: Koncepce bočního osvětlení pro maximalizaci využití dynamického rozsahu kamery

Vzhledem k tomu, že dráha produktů je kruhová, je dále nutné pro pohled zboku vytvořit pozadí, aby kamera nesnímala sušenky na opačné straně karuselu. Pozadí je vyrobeno ze stejného materiálu jako podložka rotačního karuselu. Držák pozadí byl opět navržen na míru a vytištěn na 3D tiskárně. Držák byl navržen tak, že pozadí není umístěno kolmo k podložce, ale je mírně nahnuté. Díky tomu je dále omezeno množství světla, které je od pozadí odraženo zpět do kamery a je tím pádem dále zvýšen kontrast mezi pozadím a zájmovým objektem. Na snímku 2.9 je

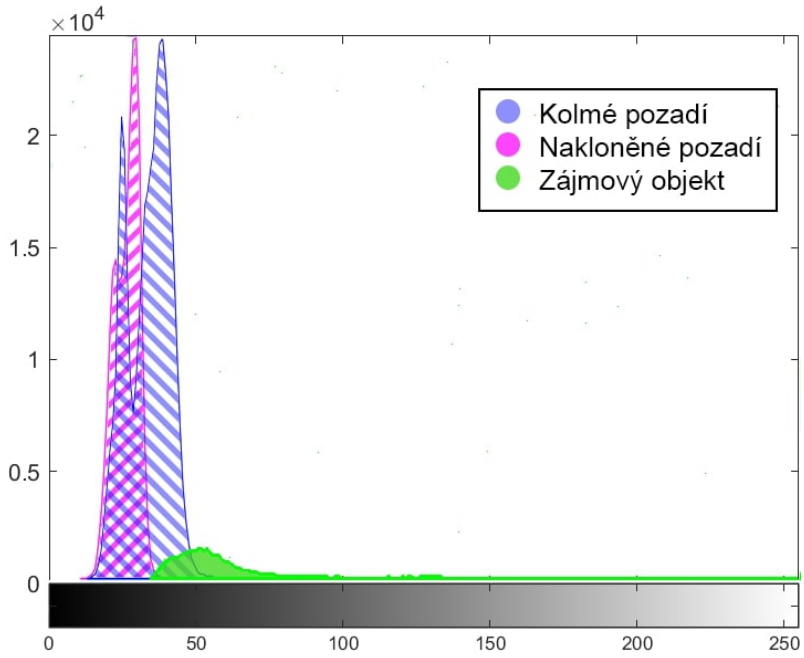
a) b)

Obr. 2.7: Srovnání snímků z bočního pohledu při použití původního osvětlení (a) a nového osvětlení (b)



Obr. 2.8: Navržený kloubový držák osvětlení pro boční scénu

vidět histogram srovnávající použití nakloněného a kolmého pozadí. Toto srovnání je provedeno pouze v kanále modré barvy pro lepší přehlednost obrázku, modrý kanál byl zvolen proto, že vliv nakloněného pozadí je zde nejvýraznější. Z obrázku je vidět, že při kolmém umístění pozadí již nelze rozlišit, kde přesně končí pozadí a začíná zájmový objekt, zatímco v případě nakloněného pozadí je kontrast mezi pozadím a zájmovým objektem jednoznačný.



Obr. 2.9: Srovnání vlivu zpětně odraženého světla při použití kolmého a nakloněného pozadí (srovnání pro modrý barevný kanál)

2.3 Software

Pro tuto práci byly poskytnuty dva programy. První z těchto programů je aplikace Anubis. Jedná se o aplikaci s grafickým uživatelským rozhraním sloužící pro ovládání kamer a provádění strojového učení.

Pro účely této práce bude provedeno refaktorování kódu aplikace a rozsáhlé rozšíření aplikace o možnost připojení více kamer a možnost ovládání platformy popsané v kapitole 2.2. Dále bude funkcionalita pro provádění strojového učení nahrazena funkcionalitou pro detekci anomálií za použití DeepSVDD modelu.

Druhým programem je implementační řešení detektoru anomálií založeném na metodě DeepSVDD a konzolové rozhraní pro provádění jejího učení a vyhodnocování.

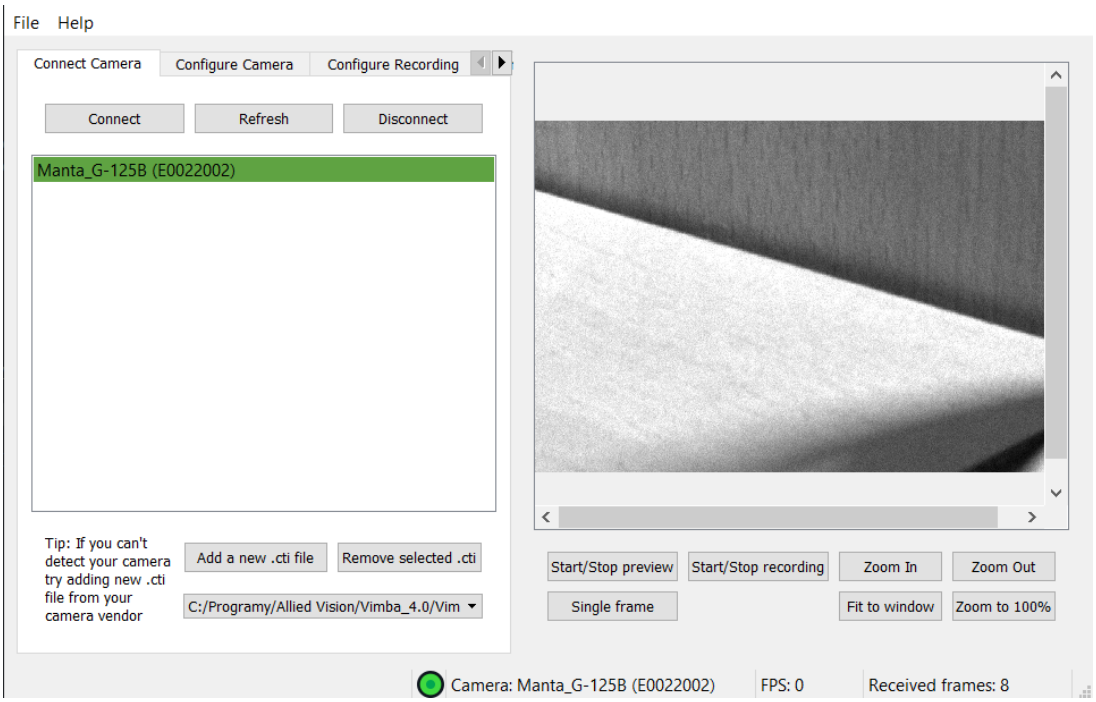
Pro oba programy budou dále do detailu popsány jejich možnosti a bude krátce popsán způsob jejich použití.

2.3.1 Anubis - uživatelská aplikace

Aplikace Anubis slouží pro komunikaci s kamerami pomocí grafického uživatelského rozhraní. Aplikace umožňuje připojení kamery, nastavení cesty pro ukládání snímků a plnou konfiguraci dostupných parametrů. Dále jsou k dispozici základní operace s náhledem kamery společně s možností spustit a zastavit nahrávání nebo náhled.

Aplikace ve svém výchozím stavu umožňuje připojit jednu kameru a plynulost jejího běhu není optimální, obzvláště potom v době aktualizace a čtení parametrů z kamery.

Aplikace ve výchozím stavu neobsahovala vazbu na detekci anomálií, ale obsahovala možnost učení modelů neuronových a konvolučních sítí společně s náhledem výsledků přímo v aplikaci. Pro účely této práce byla tato funkcionalita odstraněna a bude nahrazena detektorem anomálií založeném na metodě DeepSVDD.



Obr. 2.10: Anubis - Výchozí stav [40]

Pro potřeby této práce bylo nutné aplikaci doplnit o novou funkcionalitu a optimalizovat její chod. Byla provedena analýza kódu a chodu aplikace, na základě které byly vytyčeny následující cíle úprav:

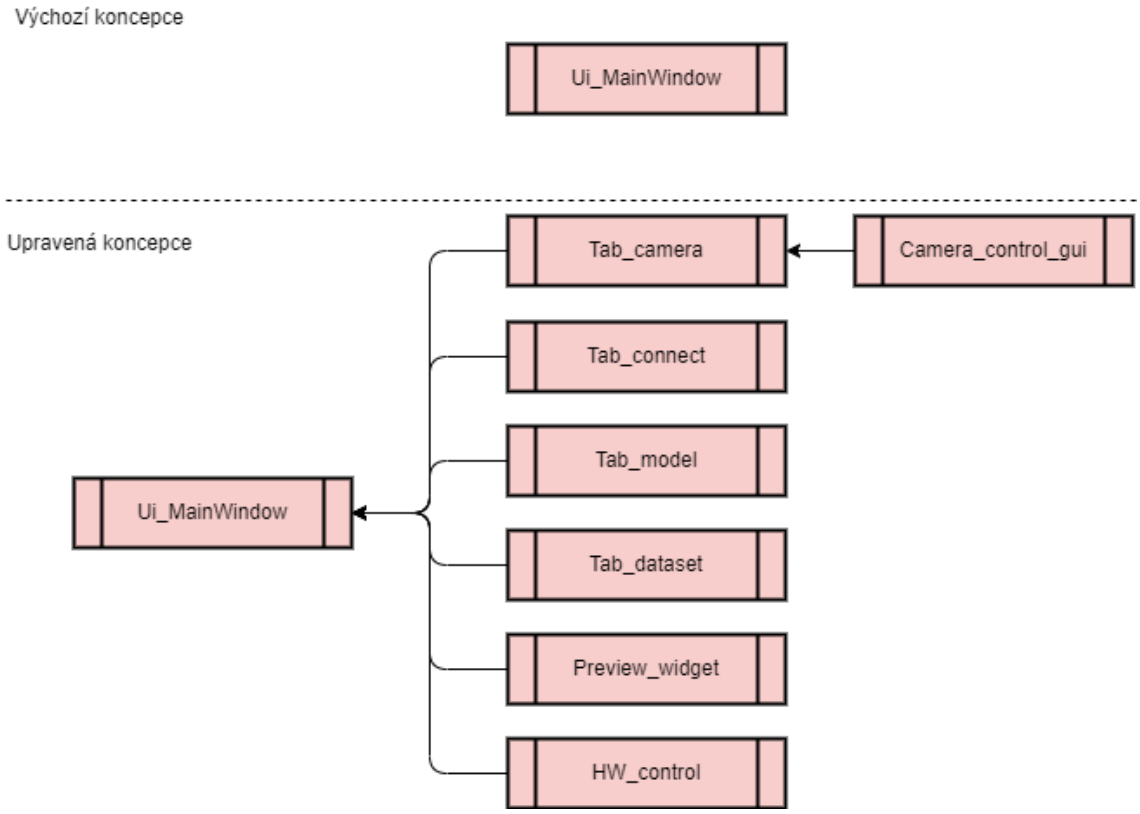
- Odstranit nalezené chyby
- Upravit kód uživatelského rozhraní tak, aby byl logicky lépe dělen
- Rozšířit uživatelské rozhraní o prvky pro:
  - Ovládání hardwarové platformy
  - Ovládání více kamer
  - Náhled z více kamer
  - Možnost automatického sbírání datasetu
  - Možnost vyhodnocování anomálií na základě naučeného DeepSVDD modelu
- Optimalizovat backend (kód běžící na pozadí) pro rychlejší chod

- Upravit kód backendu pro připojení více kamer

Postup implementace těchto úprav je blíže rozepsán dále.

Optimalizace uživatelského rozhraní

Ještě před úpravou vzhledu uživatelského rozhraní pro fungování s více kamerami bylo vhodné provést refaktorování stávajícího kódu. Jednotlivé logické části uživatelského rozhraní proto byly rozděleny do samostatných tříd a v rámci těchto tříd byly provedeny optimalizace, které dělají aplikaci uživatelsky přívětivější. Tímto bylo dosaženo přehlednější souborové struktury aplikace a vytvoření přehlednějších souborů pouze s kódem, který se týká dané části rozhraní. Jak je vidět na následujícím obrázku 2.11, tak kód z třídy *Ui_MainWindow* byl rozdělen do samostatných logických celků, které vždy představují určitou část uživatelského rozhraní a třída *Ui_MainWindow* nyní slouží pouze pro zajištění komunikace mezi těmito bloky a jejich umístění v rámci okna aplikace.



Obr. 2.11: Třídní hierarchie uživatelského rozhraní

Dále byla upravena správa parametrů kamery. Jedná se o způsob, jakým jsou čteny aktuální hodnoty z kamery. V původní verzi aplikace bylo čtení parametrů řešeno časovačem, který po uplynutí provedl nové čtení parametrů. Bohužel toto

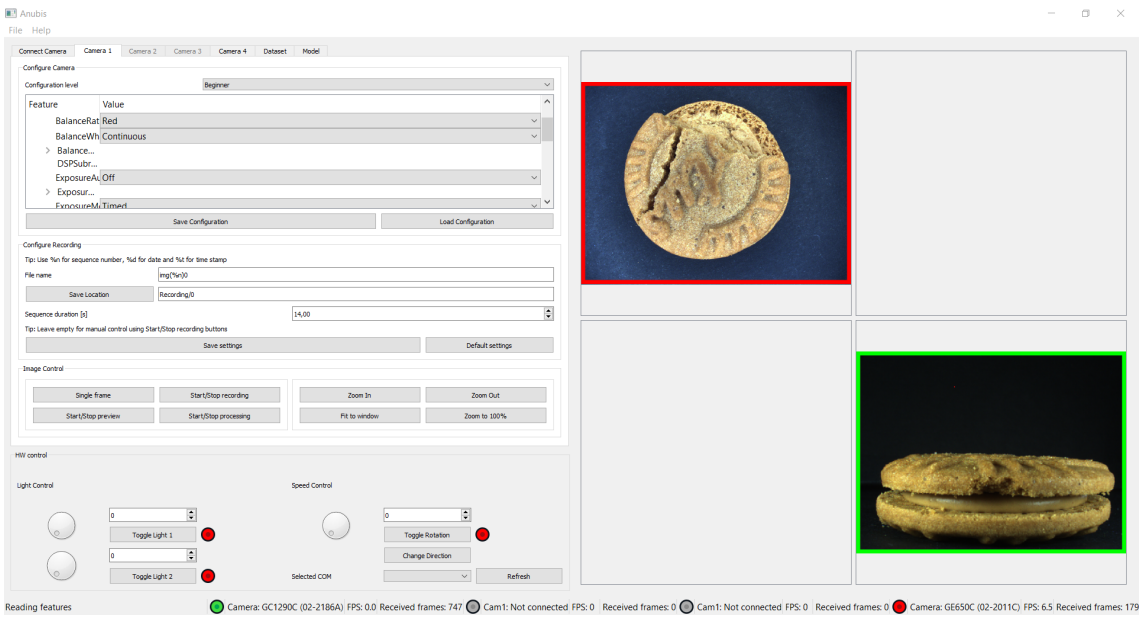
čtení je relativně dlouhé a ustavení veškerých parametrů do uživatelského rozhraní způsobovalo viditelné záseky a krátkodobé zamrzání aplikace. Proto byl vytvořen mechanismus, který nečte všechny informace o jednotlivých parametrech, ale pouze jejich hodnoty, ty jsou dále porovnány s hodnotou aktuálně zobrazenou v uživatelském rozhraní a aktualizace je provedena pouze pokud jsou tyto hodnoty rozdílné. Díky těmto úpravám aktualizace parametrů aktuálně nezpomaluje uživatelské rozhraní a uživatel nezaznamená žádné zpomalení.

Rozšíření uživatelského rozhraní

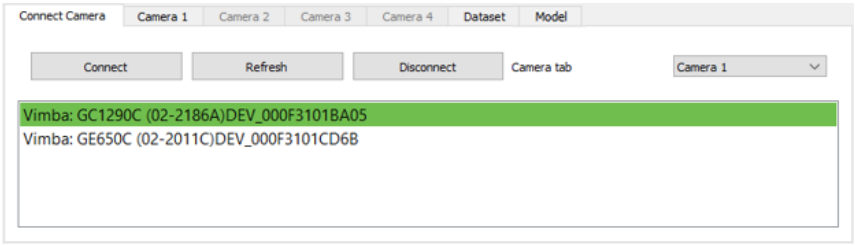
Rozšíření uživatelského rozhraní sestává z pěti částí. Těmi jsou:

- Rozšíření ovládacích prvků kamery tak, aby bylo možné ovládat až 4 různé kamery
- Přidání náhledu ve vlastním okně
- Přidání ovládacích prvků pro HW platformu
- Přidání záložky pro automatické sbírání datasetu
- Přidání záložky pro konfiguraci detektoru anomálií

Oproti původní verzi aplikace (Obrázek 2.10), byly přidány samostatné záložky pro každou kameru. Dále byly ovládací prvky přesunuty z oblasti pod náhledem do záložky každé z kamer. V neposlední řadě bylo okno náhledu rozděleno na 4 samostatná okna, která lze také otevřít samostatně pomocí dvojkliku. Upravené uživatelské rozhraní lze vidět na obrázku 2.12 a v prlné velikosti potom v příloze A.

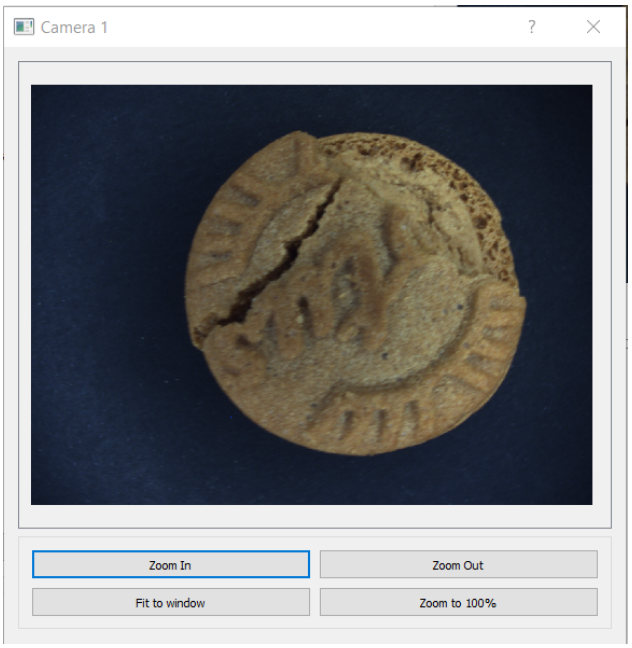


Obr. 2.12: Upravené uživatelské rozhraní aplikace



Obr. 2.13: Záložka pro připojení kamer

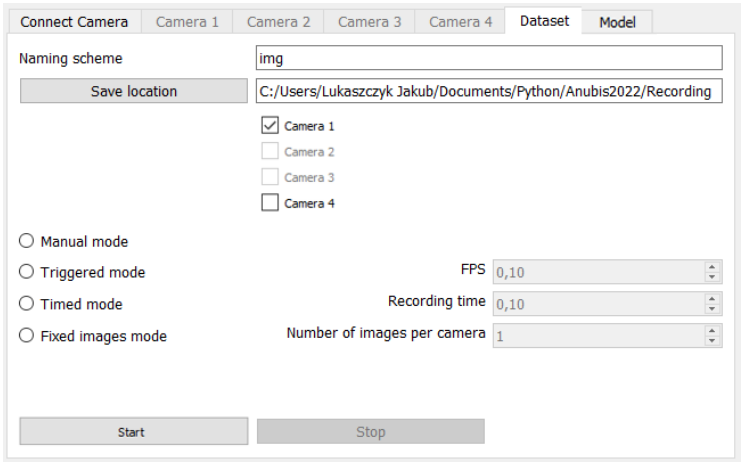
Po rozkliknutí náhledu kamery, se tento náhled otevře ve vlastním okně. Náhled ve vlastním okně umožňuje stejné operace s obrazem jako náhledy přímo v aplikaci. Přiblížení obrazu je pro každé okno samostatné. Dále je také možné se v okně pohybovat pomocí tažení myši v náhledové oblasti.



Obr. 2.14: Samostatné okno náhledu kamery

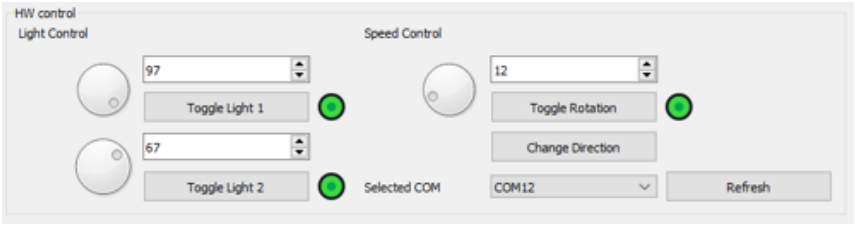
Pro automatizované sbírání datasetu byla navržena záložka (zobrazena na obrázku 2.14), která uživateli umožňuje ukládat snímky na základě některé ze zvolených podmínek (ukládání všech dostupných snímků, externí spoušť, definovaná snímkovací frekvence, definovaný počet snímků). Uživateli je dále umožněno vybrat kamery, ze kterých mají být snímky ukládány, formát názvu snímků a umístění pro ukládání snímků.

Pro účely ovládání laboratorního přípravku byly přidány ovládací prvky pro nezávislá světla a také pro ovládání motoru. Hodnoty lze zadávat buďto v číselné



Obr. 2.15: Záložka pro snímání datasetu

formě nebo pomocí rotačních prvků. Pro univerzální použití je přidána nabídka pro volbu sériového portu, ke kterému je platforma připojena. Tuto část uživatelského rozhraní lze vidět na obrázku 2.16.



Obr. 2.16: Ovládací prvky pro platformu

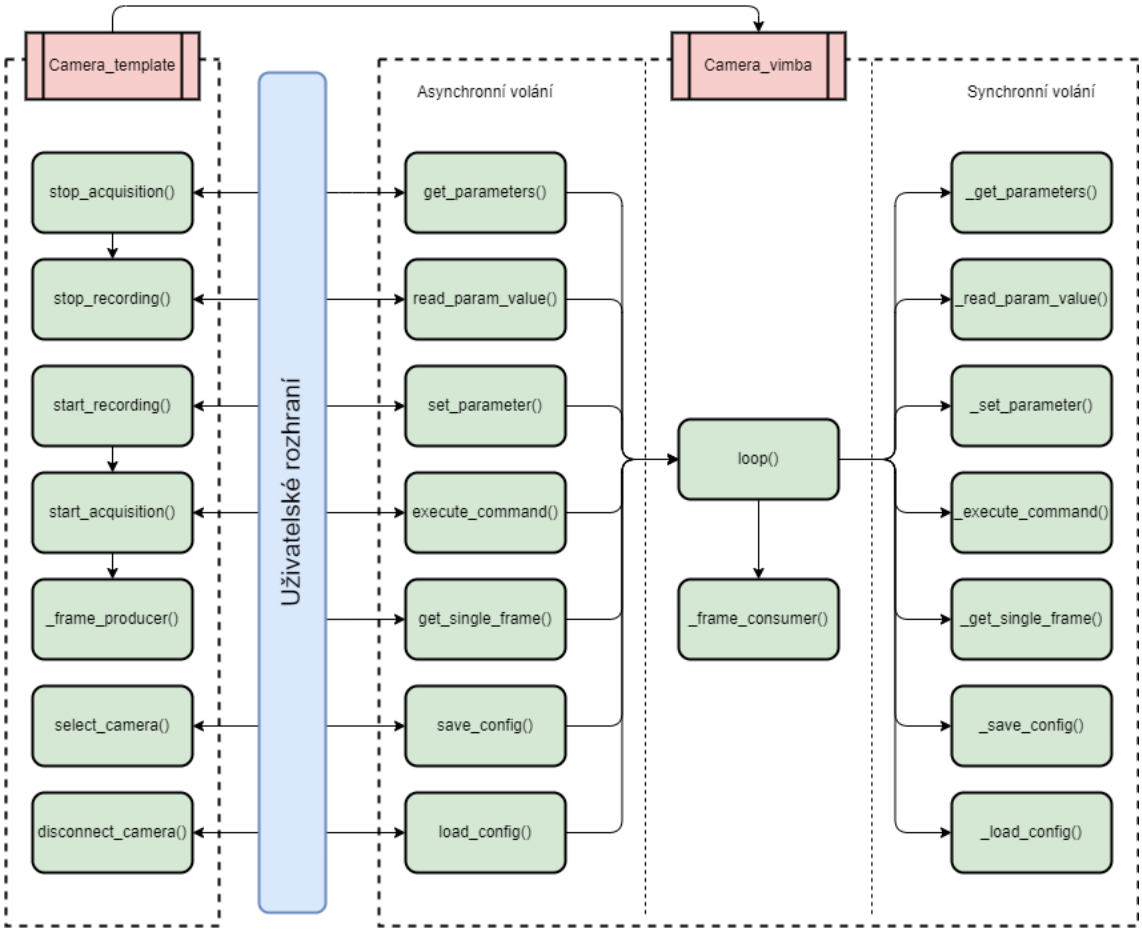
Optimalizace backendu

Ve výchozím stavu aplikace byl přístup k metodám backendu asynchronní a vznikalo tak riziko chyb z důvodu souběhu vláken a neřízeného přístupu ke sdíleným prostředkům. Docházelo tak k situacím, kdy aplikace náhodně zamrzla, nebo nebyla schopna zaručit přístup k požadovanému hardware.

Tento problém byl vyřešen změnou celé koncepce přístupu ke kameře. Byly vytvořeny metody pro rozhraní, které lze volat asynchronně z uživatelského rozhraní. Tyto metody nastavují příznaky, které backendu oznamují požadavek na různou obsluhu kamery. v hlavním vlákně kamery jsou poté tyto příznaky kontrolovány a je zajištěn přístup k hardwaru kamery v deterministickém pořadí a je zajištěno, že v jeden okamžik bude k hardwaru přistupovat vždy jen jedna metoda.

Dále tento přístup umožňuje otevření komunikačního kanálu s kamerou jednorázově a jeho držení po celou dobu připojení kamery. Toto je výhodné, protože ve staré koncepci bylo nutné v každé jednotlivé metodě otevřít komunikační kanál a na konci jej zavřít. V závislosti na kameře toto otvírání může trvat desítky milisekund až přibližně sekundu. Je tedy očekávatelné, že opakované otevíraní komunikace negativně ovlivňuje schopnost programu v reálném čase reagovat na podněty.

Koncepci tohoto přístupu skrze rozhraní lze vidět na následujícím obrázku 2.17.



Obr. 2.17: Princip fungování backendu

Dále byly identifikovány části kódu, které celkový běh zpomalovaly. Jedná se o neefektivní čtení parametrů, čtení jednotlivého snímku a také o neefektivní práci s komunikačním kanálem, jak bylo uvedeno výše.

Čtení parametrů úzce souvisí s architekturou kódu uživatelského rozhraní a bylo blíže popsáno v předchozí části zabývající se optimalizací této části aplikace.

Čtení jednotlivých snímků bylo v původní aplikaci nespolehlivé z důvodu paralelního souběhu vláken přistupujících ke kameře a bylo proto nutné kontrolovat, zda

byl snímek skutečně sejmut a pokud ne, bylo nutné akci opakovat. Jelikož tato konkrétní akce navíc neběžela ve vlastním vedlejším vlákně, tak v době snímání se stalo uživatelské rozhraní neresponzivní. Tato problematika byla vyřešena změnou koncepce, která byla popsána výše a díky řízenému přístupu ke kameře se stal kontrolní mechanismus irelevantním a bylo umožněno zrychlení chodu kódu, dále byla také přidána konverze atypických barevných modelů tak, aby bylo možné korektně zpracovat veškeré barevné modely popsané v [41] a korektně je zobrazit v uživatelském rozhraní.

Rozšíření backendu pro práci s více kamerami

Původní koncepce komunikace s kamerou využívala třídu *Camera*, jejíž instance udržovala veškeré sdílené zdroje. Ačkoliv není tento přístup špatný, tak neumožňuje komunikaci s více kamerami zároveň, jelikož vícenásobné instance této třídy mezi sebou nedokáží sdílet prostředky jako je přístup k médiu nebo přístup k API výrobce kamery (to se chová jako singleton a nelze z něj tedy vytvářet vícenásobné instance a přístup k němu je nutno řídit deterministicky v rámci programu). Pro tyto účely byla vytvořena třída *Camera_connector*. Cílem této třídy je zprostředkovat komunikační kanál mezi jednotlivými objekty typu *Camera*. Při požadavku na nalezení kamer budou vráceny nalezené kamery a v případě požadavku na připojení kamery bude zvolené kameře přiřazen unikátní identifikátor, fronta pro ukládání snímků a bude pro ni vytvořena nová instance upraveného objektu *Camera*.

Princip fungování lze vidět na následujícím obrázku 2.18.

Návod k použití

Pro zprovoznění aplikace je nutné mít na systému nainstalován Python ve verzi 3.7.9 a vyšší. Další prerekvizitou je instalace Allied Vision Vimba SDK společně s Python knihovnou Vimba. Postup pro tuto instalaci lze nalézt na [42]. Po splnění těchto prerekvizit je nutné navigovat do složky s aplikací Anubis a pomocí aplikace pip nainstalovat potřebné Python balíčky zavoláním příkazu:

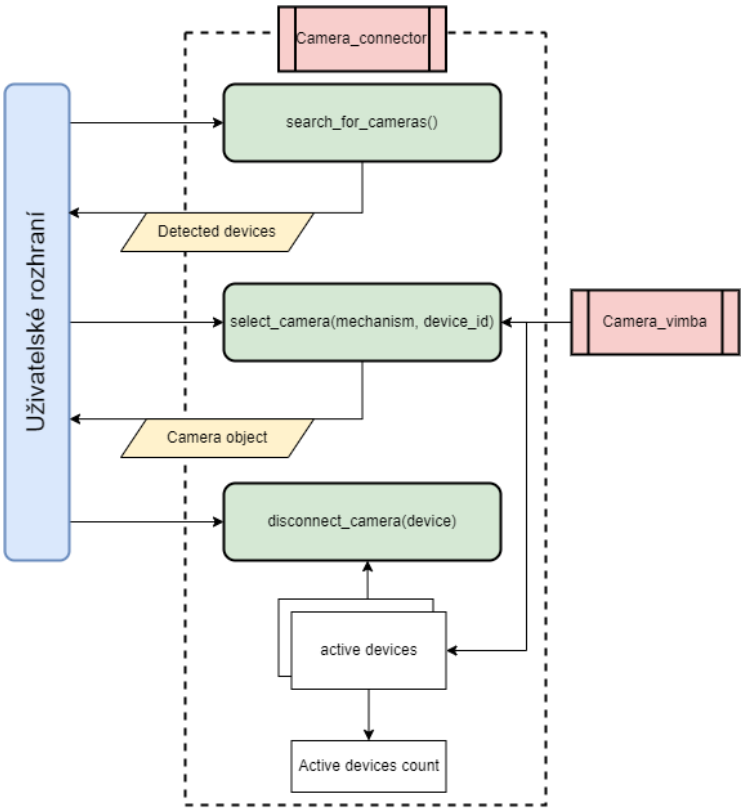
```
pip install -r requirements.txt
```

Nyní již je možné aplikaci spustit pomocí:

```
python anubis.py
```

Práce s kamerou

Pro vyhledání kamer se stiskne tlačítko *Refresh*. K nalezené kameře se potom lze připojit jejím výběrem a stisknutím tlačítka *Connect* nebo dvojklikem na název



Obr. 2.18: Diagram třídy pro správu připojených kamer

příslušné kamery. Výběr slotu pro kameru lze provést výběrem v pravé horní části karty. Tato část uživatelského rozhraní je zobrazena na obrázku 2.13.

Operace s danou kamerou potom lze provádět na záložce příslušné kamery (viz obrázek 2.12). V horní části záložky lze modifikovat parametry v jedné z dostupných uživatelských úrovní (začátečník, expert, guru). Pod oknem s konfigurací parametrů se nachází část pro konfiguraci záznamu. Vzhledem k technické povaze aplikace se záznam ukládá ve formě jednotlivých snímků pro účely jejich dalšího zpracování a pro minimalizaci ztráty detailu jednotlivých snímků vlivem komprese videa.

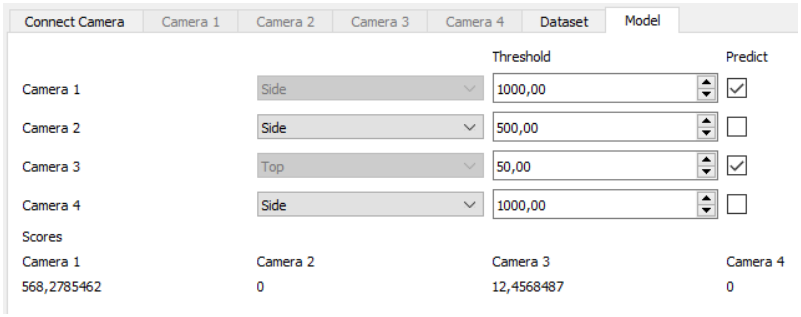
Poslední část umožňuje spustit náhled nebo nahrávání a provádět úpravu přiblížení obrazu náhledu. Pro zobrazení náhledu ve vlastním okně nezávislém na zbytku aplikace stačí provést dvojklik na příslušný náhled (obrázek 2.14).

Detekce anomálií

Pro detekci anomálií je součástí aplikace záložka *Model*. Na této záložce je možné pro každou kameru vybrat samostatně jeden z dostupných modelů (ty je nutno umístit do složky *Anubis/svdd_anubis/models*) a spustit predikci na živém náhledu z kamery. Jak bylo uvedeno již v kapitole 1.3.3, výstupem predikce při použití

DeepSVDD je vzdálenost od středu hyperkoule a výsledná klasifikace závisí na prahování této hodnoty.

Výstupní vzdálenost lze vidět na obrázku 2.19 ve spodní části. Nastavený práh potom v části nazvané *Threshold*. Vizuální výstup detekce je uživateli prezentován ve formě barevného rámečku kolem náhledového snímku (červená - anomálie, zelená - normální třída - viz obrázek 2.12).



Obr. 2.19: Záložka pro obsluhu detektoru anomálií

2.3.2 DeepSVDD - implementace

Pro účely této práce byla poskytnuta implementace metody DeepSVDD v jazyce Python. K této implementaci byla také poskytnuta konzolová aplikace, která umožňuje provádět učení a vyhodnocení modelů tak, jak bylo autory metody popsáno v původním článku, který byl rozebrán v kapitole 1.3.3.

Oproti těmto základním možnostem umožňuje implementace provést předtrénování pomocí autoenkodéru pro lepší výchozí nastavení vah sítě.

Vzhledem k tomu, že návrh a vývoj samotného algoritmu detekce anomálií není bodem zadání této práce, tak bude pouze krátce popsán způsob použití a možností tohoto implementačního řešení, dále budou popsány případné modifikace.

Učení a testování modelu

V této kapitole bude na jednoduchém příkladu vysvětlen postup naučení modelu pro detekci anomálií na sušenkách Lotus (viz 2.4). Pro učení modelu je nutné mít připraveno několik podkladů. Těmito podklady jsou:

- Zvolená architektura extraktoru příznaků, kterou bude model využívat
- Hodnoty povinných parametrů (a případně hodnoty volitelných parametrů)
- Zkompilovaný dataset
- Vytvořenou složku pro uložení výstupů

Přehled architektur extraktorů příznaků, které jsou pro učení k dispozici lze nalézt na přiloženém DVD č.1 v souboru *DeepSVDD/src/networks/main.py*. Je možné také implementovat vlastní extraktor (bude blíže popsáno v následující části).

Výčet povinných a nepovinných parametrů lze nalézt opět na DVD č.1 v souboru *DeepSVDD/src/main.py*, kde jsou povinné parametry označeny jako *argument* a nepovinné parametry jako *option*. Volitelné parametry lze programu předat jako součást spouštěcího příkazu nebo je uvést v konfiguračním souboru (viz výpis 2.1) a programu při spuštění pouze předat cestu k tomuto souboru. Pro spuštění učení s povinnými parametry a parametrem pro načtení konfiguračního souboru slouží následující příkaz:

```
python main.py  top lotus_LeNet_single ../log/output
    ../data/dataset_lotus_standard --load_config
    ../log/config.json --device "cuda"
```

Výpis 2.1: Konfigurační soubor s parametry pro učení modelu DeepSVDD

```
{"dataset_name": "top", "net_name": "lotus_4conv",
"net_res": 256, "net_rep_dim": 256,
"xp_path": "../log/", "normal_class": 2,
"data_path": "../data/dataset_lotus",
"load_config": true, "load_model": null,
"objective": "one-class", "nu": 0.1,
"device": "cuda", "seed": 211157,
"optimizer_name": "adam", "lr": 0.0001,
"n_epochs": 100, "lr_milestone": [25, 50, 75],
"batch_size": 10, "weight_decay": 0.0001,
"pretrain": true, "ae_optimizer_name": "adam",
"ae_lr": 0.001, "ae_n_epochs": 100,
"ae_lr_milestone": [], "ae_batch_size": 100,
"ae_weight_decay": 1e-05, "n_jobs_dataloader": 0}
```

Dataset, který bude použit pro učení musí být programu poskytnut v jasně definovaném formátu. Tento formát může být odlišný pro každý implementovaný dataset. V souboru *DeepSVDD/src/datasets/main.py* lze nalézt výčet dostupných datasetů. Konkrétní formát dat lze potom vyčíst z implementačního souboru daného datasetu (složka *DeepSVDD/src/datasets*). Pro účely použití datasetu, který byl vytvořen v této práci a jehož tvorba je náplní kapitoly 2.4 je nutné jej nejprve zkompilovat z formátu samotných snímků do tří samostatných balíčků (trénovací, validační, testovací).

Pro kompilaci slouží skript *DeepSVDD/src/utils/dataset_creating_lotus.py*, jehož použití vyžaduje cestu ke zdrojovému datasetu, cestu pro uložení výstupu a konfigurační soubor. Příklad příkazu pro sestavení datasetu je uveden níže:

```
python dataset_creating_lotus.py -i dataset
    -o ../../Data/dataset_lotus -c config.json
```

Konfigurační soubor potom v první části obsahuje výčet skupin, do kterých jsou původní data tříděna a počet vzorů v těchto skupinách.

Druhá část konfiguračního souboru potom představuje přeresolutions, jak tato vstupní data rozdělit mezi trénovací, validační a testovací data. Příklad konfiguračního souboru lze vidět ve výpisu 2.2.

Výpis 2.2: Konfigurační soubor s parametry pro kompilaci datasetu

```
{
 "baseline": {
    "folder":         "dataset",
    "subfolders":     ["Foreign_items", "Scene", "Mixed",
                       "Cracks", "Missing_parts", "OK"],
    "numbers":        [101,50,100,80,121,686],
    "start":          [1,1,1,1,1,1]
 },
 "new": {
    "name":           "dataset_lotus",
    "subsets":        ["train","valid","test"],
    "subsets_#":      [[0, 0, 0, 0, 0, 500],
                       [0, 0, 0, 0, 0, 46],
                       [101, 50, 100, 80, 121, 140]],
    "subsets_labels":[[1, 1, 1, 1, 1, 2],
                       [1, 1, 1, 1, 1, 2],
                       [1, 1, 1, 1, 1, 2]]
 }
}
```

Poslední prerekvizitou pak je pouhé vytvoření složky, do které bude uložen výsledný model společně s dalšími soubory jako jsou ROC křivka, chybně klasifikované prvky, matice záměn nebo třeba záznam průběhu učení.

Pro vygenerování těchto podpůrných souborů pro výsledný model slouží skript *viz.py*, který je po úspěšném naučení modelu spuštěn následujícím příkazem:

```
python viz.py --xp_path ../log/experiment
```

Implementace architektury extraktoru příznaků

Pro implementaci vlastního extraktoru příznaků je nutné pro něj vytvořit vlastí soubor ve složce *src/networks*. Ukázka implementace extraktoru je obsahem výpisu 2.3. Pro definici jednotlivých vrstev a dalších funkcí je využita knihovna PyTorch [43]. V inicializační funkci jsou nadefinovány jednotlivé vrstvy a jejich parametry.

Metoda *forward* potom definuje sekvenci dopředného průchodu vzorku extraktorem.

Výpis 2.3: Definice třídy pro extraktor příznaků

```
class LOTUS_LeNet_single(BaseNet):
    def __init__(self, width, height, rep_dim = 128):
        super().__init__()
        self.rep_dim = rep_dim
        self.width = width
        self.height = height
        self.pool = nn.MaxPool2d(2, 2)

        self.conv1 = nn.Conv2d(3, 32, 5, bias=False, padding=2)
        self.bn2d1 = nn.BatchNorm2d(32, eps=1e-04, affine=False)
        self.conv2 = nn.Conv2d(32, 64, 5, bias=False, padding=2)
        self.bn2d2 = nn.BatchNorm2d(64, eps=1e-04, affine=False)
        self.conv3 = nn.Conv2d(64, 128, 5, bias=False, padding=2)
        self.bn2d3 = nn.BatchNorm2d(128, eps=1e-04, affine=False)
        self.fc1 = nn.Linear(
                        128*int(self.width/8)*int(self.height/8),
                        self.rep_dim, bias=False)

    def forward(self, x):
        x = self.conv1(x)
        x = self.pool(F.leaky_relu(self.bn2d1(x)))
        x = self.conv2(x)
        x = self.pool(F.leaky_relu(self.bn2d2(x)))
        x = self.conv3(x)
        x = self.pool(F.leaky_relu(self.bn2d3(x)))
        x = x.view(x.size(0), -1)
        x = self.fc1(x)
        return x
```

Pokud má být v trénovací fázi použit autoenkodér pro předtrénování, je nutné jej do tohoto souboru obdobným způsobem nadefinovat. Při definici autoenkodéru je však nutné dbát na to, aby se architektura enkodérové části shodovala s architekturou extraktoru příznaků. V opačném případě by nebylo možné přenést předtrénované váhy.

Další příklady implementovaných architektur lze nalézt ve složce *src/networks*. Po implementaci sítě je nutné zanést informaci o její existenci do zbytku programu. Zanesení informace je v rámci výpisu 2.4 demonstrováno na síti *lotus_LeNet_single*.

Výpis 2.4: Zanesení informace o novém extraktoru příznaků

Soubor
DeepSVDD/src/networks/main.py

```
    Na začátek souboru přidat
        from .lotus_LeNet_single import LOTUS_LeNet_single
        from .lotus_LeNet_single import LOTUS_LeNet_ae_single

    build_network()
        implemented_networks = (... "lotus_LeNet_single", ...)

        if net_name == "lotus_LeNet_single":
            net = LOTUS_LeNet_single(
                                net_res, net_res, net_rep_dim)
    build_autoencoder()
        implemented_networks = (... "lotus_LeNet_single", ...)

        if net_name == "lotus_LeNet_single":
            net = LOTUS_LeNet_ae_single(
                                net_res, net_res, net_rep_dim)
```

Soubor
DeepSVDD/src/main.py

```
    @click.argument('net_name',
        type=click.Choice([... "lotus_LeNet_single", ...]))
```

2.4 Tvorba datasetu

Obsahem této kapitoly je definice terminologie a sémantiky, která bude kritická pro následující experimenty. Dále je popsána použitá soustava pro snímání datasetu včetně nastavení jednotlivých prvků jako jsou například kamery a osvětlení. Poslední část kapitoly se potom věnuje samotnému vytvořenému datasetu a definuje jednotlivé anomálie.

2.4.1 Terminologie a sémantika

Cílem jednotřídního klasifikátoru je vždy oddělit takzvané *normální vzory* od takzvaných *anomálií*. Výsledek klasifikace je potom samozřejmě dán tím, jaké vzorky vstupního datasetu byly ve fázi učení považovány za normální a jaké za anomální.

V této práci se ale množina normálních vzorů (vstupů ve fázi učení) vždy nutně zcela neshoduje s množinou takzvaných OK vzorů (tedy reálně bezchybných vzorů datasetu). Důvodem je záměrná záměna částí těchto množin z důvodu analýzy chování klasifikačního algoritmu - detailněji viz dále.

S ohledem na uvedené je vhodné doplnit, že pojmy OK vzor a normální třída jsou v technické praxi obvykle zaměnitelné a jejich význam je rozdílný pouze při výše uvedené analýze chování algoritmu.

OK vzory

Jde o objekty (v kontextu této práce sušenky), které byly v době snímání datasetu označeny jako v pořádku (tedy neobsahují žádné kosmetické vady).

NOK vzory

Objekty, které byly při snímání ručně anotovány, že obsahují kosmetickou vadu některého z definovaných typů (viz kapitola 2.4.4). V technické praxi je tento pojem zaměnitelný s pojmem anomálie. Pro potřeby experimentů v této práci jsou někdy NOK vzory definovány jako normální třída nebo jsou dokonce míseny s OK vzory.

Anomálie

Objekt, který se vymyká naučené definici objektu. Jedná se o hledané vzory. V kontextu metody DeepSVDD anomalita prvků roste se vzdáleností jejich mapování od středu hyperkoule. Jelikož jde o vzory, jejichž detekce je požadována, tak pro účely vyhodnocení pomocí matice záměn nebo ROC křivky jsou anomálie považovány za pozitivní třídu.

V průběhu učení nejsou prvky, které by byly explicitně označeny jako anomálie dostupné (ačkoliv se mohou v určité míře objevovat v rámci normální třídy vlivem nesprávné anotace).

Normální vzory

Jedná se o třídu vzorů, která by měla být dobře známá a která je explicitně označená a dostupná pro účely učení modelu. Cílem metody DeepSVDD je na základě společných znaků mezi normálními vzory sestavit mapovací funkci. Jakmile je model naučen, tak se z pohledu detekce anomálií jedná o již nezajímavou množinu a je tedy dále označovaná za negativní (tedy neobsahuje anomálie).

2.4.2 Sestava pro snímání

Pro realizaci datasetu byla použita dvojice kamer. Umístění kamer bylo popsáno již dříve v kapitole 2.2.2. Snímky shora byly snímány kamerou Allied Vision Technologies Prosilica GC1290c [44]. Snímky zboku potom byly snímány kamerou Allied Vision Technologies Prosilica GE650C [45]. Obě kamery byly s PC propojeny pomocí ethernetového kabelu a komunikace probíhala pomocí standardu GigE Vision,

Ke kameře pro pohled shora byl zvolen objektiv Pentax, typ FL-HC0416X-VG. Jedná se o objektiv s ohniskovou vzdáleností 4,2 mm a s plynule nastavitelnou clonou až do hodnoty f/1,6. Celá specifikace je uvedena v dokumentu [46].

Kamera bočního pohledu byla osazena objektivem Honeywell HLM5V60F13 s nastavitelnou ohniskovou vzdáleností v rozsahu 5-50 mm a nastavitelnou clonou až do hodnoty f/1,3. Kompletní specifikace tohoto objektivu jsou uvedeny v [47].

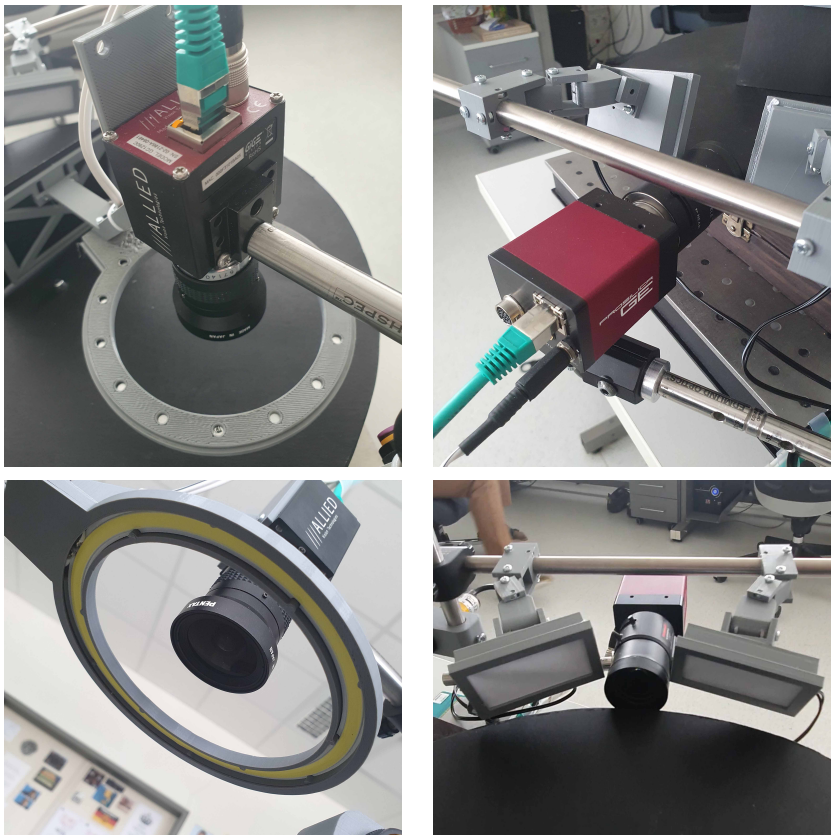
Pohled na fyzické umístění světel vůči kamerám a kamer v rámci inspekční platformy lze vidět na obrázku 2.20.

Po fyzickém nastavení scény bylo nutné nastavit parametry kamer tak, aby bylo dosaženo co nejlepších snímků. Cílem bylo nastavit kamery, osvětlení a objektivy tak, aby výsledné snímky byly co nejostřejší, co nejlépe využívaly dynamický rozsah kamery a aby zájmový objekt měl co nejlepší odstup od pozadí.

Prvotní nastavení scény probíhalo na zcela odcloněném objektivu, kde byla nastavena poloha objektu vůči kameře a snímek byl co nejlépe zaostřen.

Následně byl objektiv co nejvíce zacloněn, aby byla maximalizována hloubka ostrosti. V tomto kroku byla současně nastavovaná intenzita osvětlení (viz následující kapitola 2.4.3) a expoziční doba kamery.

V následující tabulce 2.1 lze vidět nastavené hodnoty základních parametrů kamer. Na přiloženém DVD č.1 pak jsou připravené konfigurační soubory (tyto soubory se nacházejí ve složce *Anubis/camera_config*, které nastaví kamery přesně do stavu, ve kterém byly v době snímání datasetu.



Obr. 2.20: Vzájemné umístění kamer a světel ve scéně

Tab. 2.1: Nastavení základních parametrů kamer

	GC1290c	**GE650C**
Expoziční doba [μs]	9 114	91 070
Zisk [-]	0	0
Barevný formát [-]	BayerRG8	RGB8Packed
Výška snímku [px]	960	494
Šířka snímku [px]	960	494
Vyvážení barev - modrá [-]	2,83	2,11
Vyvážení barev - červená [-]	0,95	1,33

2.4.3 Nasvícení vzorků

V kapitole 2.2.3 byla popsána fyzická konfigurace použitých světel. V rámci této kapitoly bude popsáno, jak byla nastavována vhodná intenzita světla a její konečné nastavení tak, aby mohly být podmínky pro snímání (a vyhodnocování) vzorků

korektně replikovány. Pro účely měření intenzity osvětlení byl použit luxmetr s typovým označením PU550 značky Metra Blansko.

Nastavení osvětlení pro pohled shora je relativně jednoduché, neboť je pozice světla vůči kameře pevně daná a měnit lze pouze výšku celé této sestavy od podložky. Předchozí kapitola 2.4.2 popisující sestavu pro snímání jasně definuje tuto pozici. Osvětlení by potom mělo být nastaveno na takovou intenzitu, aby v oblasti přímo pod kamerou byla intenzita osvětlení o 12 100 lx vyšší než při vypnutém osvětlení v případě horního pohledu. Pozici měřicího prvku lze vidět na obrázku 2.21. Přehledová tabulka 2.2 potom ukazuje jaké intenzity osvětlení je horní světlo schopné dosáhnout, to v zásadě omezuje maximální množství parazitního světla před tím, než nebude možné jej osvětlením na platformě kompenzovat. Na základě nastavené intenzity osvětlení je dále nutné adekvátně upravit zaclonění objektivu.

a) b)

Obr. 2.21: Pozice snímacího prvku pro nastavení osvětlení scény za pomocí luxmetru. a) nastavení scény shora, b) nastavení boční scény

Tab. 2.2: Měření osvětlení scény pomocí luxmetru

	Horní scéna		**Boční scéna**	
Nastavená intenzita osvětlení [%]	**Osvětlená místnost** [lx]	**Zatemněná místnost** [lx]	**Osvětlená místnost** [lx]	**Zatemněná místnost** [lx]
0 (referenční)	390	4,01	191,6	2,27
30	5 250	5 080	898	395
50	9 060	9 170	1 232	580
80	14 890	15 230	1 679	818
100	19 980	21 000	1 872	930

Při nastavování scény zboku je nutné provést obdobný postup pro nastavení dostatečné intenzity osvětlení. Pro boční pohled by intenzita osvětlení měla být o 1 200 lx vyšší než ve scéně s vypnutým osvětlením. Z přehledové tabulky 2.21 je vidět, že boční scéna je mnohem více náchylná na parazitické osvětlení a je tedy vhodné ji nastavit jako první a zajistit, že okolní podmínky vůbec umožňují její realizaci. V případě příliš silného parazitického osvětlení, které již nelze kompenzovat umělým osvětlením na platformě, je nutné toto osvětlení minimalizovat jiným způsobem (například zakrytím přípravku neprůhledným materiálem).

2.4.4 Hotový dataset

Po domluvě s vedoucím práce byl jakožto průmyslový výrobek zvolen výrobek potravinářského průmyslu, konkrétně potom slepované sušenky Lotus. V rámci tvorby datasetu byly snímány OK a NOK kusy. Tyto dvě třídy jsou vyvážené a každá obsahuje přibližně 650 realizací (konkrétní počty lze nalézt v přehledové tabulce 2.3). NOK kusy jsou poté dále děleny do 7 různých kategorií:

- **Praskliny** - definovány jako sušenky, které jsou kompletní, ale nachází se na nich alespoň jedna prasklina velikosti minimálně 2,5 cm.
- **Cizí předměty** - sušenky, které neobsahují přímo vadu ve své struktuře, ale obsahují nějaký typ cizího předmětu (bužírka, provázek, matička, kousek papíru).
- **Chybějící části** - jedná se o sušenky, jejichž libovolná část chybí. Vzhledem k fyzické realizaci tohoto typu vady, byla tato vada definována dále tak, že musí obsahovat alespoň jednu chybějící část, ale současně může obsahovat libovolný počet prasklin.
- **Nastavení scény** - tato vada je definována jako scéna, která byla vlivem vnějších vlivů modifikována tak, že se v záběru nenachází celá sušenka (nebo se na snímku nenachází vůbec). Případně je scéna přeexponovaná nebo podexponovaná nebo jde o rozmazanou scénu.
- **Kombinované anomálie** - jedná se o anomálie, které obsahují libovolnou kombinaci nejméně dvou typů vad popsaných výše.
- **Jednostranné zboku** - jedná se o speciální skupinu kombinovaných anomálií, kde jde daná vada vidět jen na snímku zboku.
- **Jednostranné shora** - jedná se o speciální skupinu kombinovaných anomálií. Vada je viditelná pouze na snímku shora.

Všechny realizace byly snímány ze dvou pohledů, a to konkrétně shora a zboku. Tyto dva pohledy byly zvoleny jako reprezentační pro demonstraci úlohy detekce anomálií ve více pohledovém zasazení.

Dataset byl koncipován tak, aby jednotlivé anomálie byly viditelné vždy z obou

snímaných směrů pro prvních 5 kategorií. Jednostranné anomálie potom představují stav, kdy lze anomálie vidět pouze z jednoho pohledu. V případě skutečného nasazení systému podobného ražení by bylo použito dalších pohledů (například také pohled zezadu, případně zezdola) tak, aby se nemohlo stát, že anomálie budou mimo zorné pole použitých kamer.

Tab. 2.3: Počty realizací ve vytvořeném datasetu

Typ		Počet realizací
Normální		686
Anomálie	Praskliny	100
	Cizí předměty	101
	Chybějící části	121
	Nastavení scény	50
	Kombinované	100
	Jednostranné zboku	80
	Jednostranné shora	83

Normální kusy Praskliny

Cizí předměty Chybějící části

Nastavení scény Kombinované anomálie

Jednostranné shora Jednostranné zboku

Obr. 2.22: Ukázka datasetu

3 Praktická část - experimenty

Tato kapitola se zabývá experimentální částí práce. Kapitola bude dělena na dvě části. První část se bude zabývat ověřením činnosti detektoru anomálií a nalezením vhodné kombinace hyperparametrů pro následné nastavení pracovního bodu a také pro použití jako výchozí bod v následujících experimentech.

Druhá část se potom bude zabývat vlivem definice anomálií v procesu učení (vliv nesprávně anotovaných dat a citlivostní analýza) na klasifikační schopnosti modelu. Dále bude také zkoumán vliv definice testovacích anomálií na odhad výkonnosti modelu a možnosti zobecnění výsledků dosažených na vybraném typu anomálií na celé anomální spektrum.

Poznámka k provedeným experimentům

V rámci práce byly experimenty, které budou rozebrány dále, provedeny vždy samostatně pro pohled shora a pohled zboku. Jelikož se jedná jen o jiný pohled na stejné vady, bylo očekáváno, že výsledky budou velice podobné. Z toho důvodu a také z důvodu lepší přehlednosti práce budou zde, v hlavním textu, prezentovány pouze výsledky dosažené na snímcích při pohledu shora. Výsledky na snímcích zboku potom budou obsahem přílohy B.

Na přiloženém DVD č.1 jsou potom přiloženy konfigurační soubory pro modely a datasety použité v jednotlivých experimentech. Z kapacitních důvodů je na DVD obsažen pouze výběr nejlepších modelů, ostatní modely musí být naučeny na základě přiložených konfiguračních souborů.

3.1 Experimenty - ověření vlastností sítě

Tato sada experimentů měla za cíl ověřit, zda implementace metody funguje správně, zda reaguje na použitá data. Dále bylo cílem ověřit platnost hypotézy, že nelze spolehlivě modelovat NOK vzory (anomálie) jako plnohodnotnou třídu. Ještě před provedením těchto experimentů však bylo nutné nalézt vhodné nastavení hyperparametrů pro nastavení optimálního pracovního bodu sítě pro následné experimenty.

3.1.1 Návrh metodiky

Cílem bylo navržení metodiky pro nalezení hyperparametrů, které umožní modelu se naučit rozeznávat anomálie s co nejvyšší přesností. Tyto hyperparametry měly následně sloužit jako neměnný výchozí bod pro další experimenty, které budou zkoumat už jen vliv měnícího se trénovacího, validačního a testovacího datasetu.

Jako dataset byla použita data s rozdělením, které lze vidět v následující tabulce 3.1. Toto rozdělení dat bylo nazváno jako *standardní dataset* a bude na něj takto odkazováno v dalším textu.

Primárním kritériem při návrhu tohoto rozdělení dat bylo zajištění dostatečného množství trénovacích vzorů. Sekundárním parametrem bylo zajištění dostatečně mohutné anomální testovací množiny. Vzhledem k povaze problematiky (viz kapitola 1.1) nebyly do trénovacího ani validačního datasetu zahrnuty anomální vzory. Vzhledem k celkovému počtu vzorů ve zdrojovém datasetu není možné rozdělit data tak, aby zároveň měl model dostatek trénovacích vzorů a současně byla testovací množina vyvážená a zároveň mohutná. Výsledné rozdělení je tedy částečně kompromisem a v následných experimentech je nutné při hodnocení brát v potaz určitou nevyváženost testovacích dat.

Aby byly výsledky na pohledu zboku a shora srovnatelné, nebyly do tohoto datasetu dále zahrnuty kategorie jednostranných anomálií.

Tab. 3.1: Dělení dat ve standardním datasetu

		OK vzory	**NOK vzory**
Trénovací	**Normální třída**	500	-
	Anomálie	-	-
Validační	**Normální třída**	46	-
	Anomálie	-	-
Testovací	**Normální třída**	140	-
	Anomálie	-	452

Konfiguraci hyperparametrů lze hledat mnoha způsoby. Mezi nejčastěji používané patří grid search, evoluční přístup a náhodný výběr [48][35][49].

Grid search funguje na principu hrubé síly. Neboli model je učen na všech možných kombinacích hyperparametrů [48]. V případě, že byť jen jeden parametr není omezen shora nebo zdola nebo jde o reálné číslo, tak teoretické množství konfigurací jde k nekonečnu. Proto se obvykle definuje spodní mez, horní mez a krok pro každý parametry. I v případě, že by byl každý parametr omezen jen na 4 hodnoty, tak v případě použité implementace a jejích možností jde o 18^4 konfigurací, což je časově nerealizovatelné.

Další možností je náhodný výběr [49]. Zde jsou jednotlivé konfigurace vybírány náhodně. Metoda nezaručuje nalezení optimální konfigurace, ale při dostatečně vysokém počtu iterací by se optimu měla statisticky blížit. Opět však pro dosažení relevantních výsledků je nutné provést nejméně o tisíc experimentů.

Třetí metodou je využití evolučních algoritmů [35]. Jedná se o metody, kdy po každé iteraci jsou vybrány modely s nejvyšší úspěšností a jejich nastavení je potom kříženo a je z nich vygenerována nová generace. Oproti grid search sice nezaručují optimální řešení, ale vyžadují podstatně méně iterací.

Vzhledem k časové náročnosti volby hyperparametrů a velkému množství experimentů, které je pro nalezení ideálního řešení nutné, byly hyperparametry voleny z části na základě zkušeností a za použití výchozího nastavení hyperparametrů poskytnutého autory sítě. Hyperparametry byly tedy voleny postupně od obecně nejvlivnějších po ty s nejmenším vlivem na celkové schopnosti sítě.

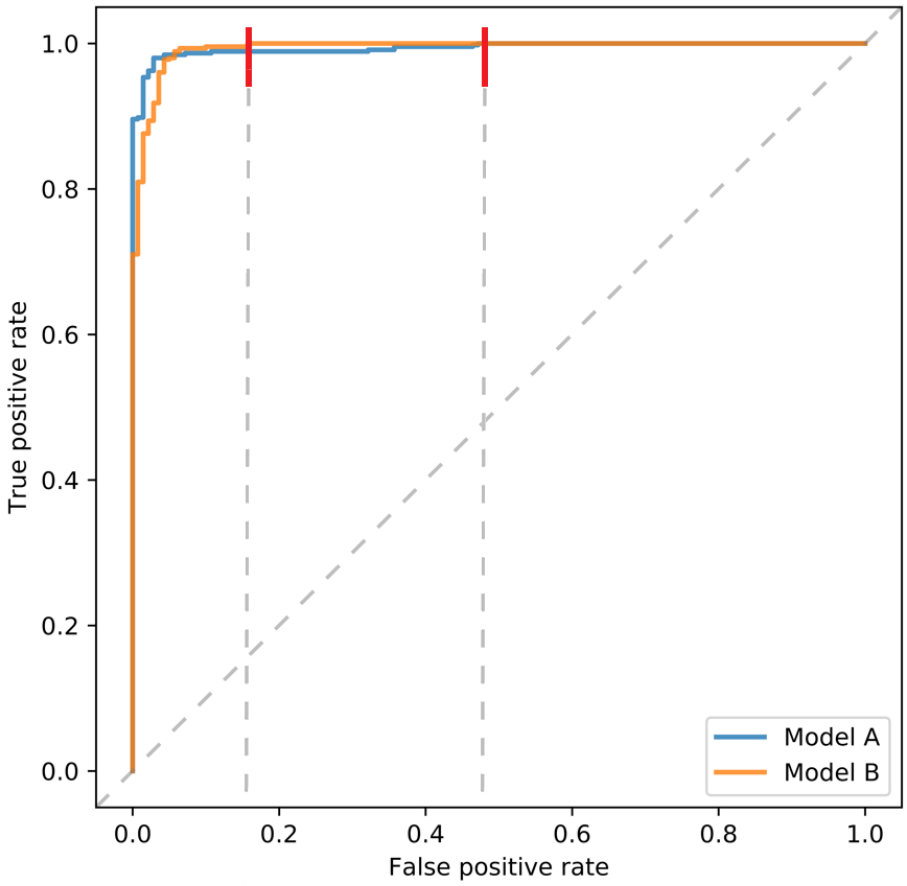
Pro vyhodnocování a srovnání modelů byly využity matice záměn pro model s nastavenou prahovou vzdáleností, tak aby bylo odhaleno 90 % testovacích anomálií. Jako primární metrika se použila celková přesnost.

Jak bylo popsáno dříve v kapitole 1.4.1, tento přístup může být zavádějící. To se ukázalo při zpětné kontrole výsledků a také při následném vykreslení ROC křivek. Na následujících obrázcích 3.1 a 3.2 je uveden příklad dvou takových modelů. Při pohledu na matici záměn a z ní vypočtených metrik může model A (ACC 96,12 %, TPR 93,35 %, FPR 1,43 %) působit jako lepší než model B (ACC 93,07 %, TPR 91,81 %, FPR 2,86 %). Avšak při pohledu na ROC křivku (obrázek 3.2) je vidět, že v případě modelu A dochází při snaze o zvýšení TPR k rychlému zvyšování falešně pozitivních detekcí. Je tedy nutné pro konkrétní úlohu zvážit jak velký vliv bude přikládán falešným detekcím při zvyšování počtu správně odhalených anomálií. Model B v tomto případě například při nastavení hraniční vzdálenosti pro detekci 100 % testovacích anomálií bude generovat podstatně méně falešně pozitivních detekcí (vyznačeno na obrázku červeně) než model A. Model B by se tedy dal považovat za lepší i přesto, že má pro danou hraniční vzdálenost horší celkovou přesnost.

		Predikce	
		Pozitivní	Negativní
Skutečnost	Pozitivní	431	21
	Negativní	2	138

		Predikce	
		Pozitivní	Negativní
Skutečnost	Pozitivní	415	37
	Negativní	4	136

Obr. 3.1: Matice záměn použité pro nevhodnou metodiku vyhodnocení (vlevo model A, vpravo model B)



Obr. 3.2: Srovnání ROC křivek modelů prezentovaných v obrázku 3.1

3.1.2 Volba hyperparametrů

Na základě zkušeností získaných v předchozí sadě experimentů byla modifikována metodika pro srovnávání modelů. Postup volby hyperparametrů byl obdobný, akorát byl rozšířen také o změnu samotné architektury extraktoru příznaků. Jednotlivé modely byly dále porovnávány primárně pomocí ROC křivky a metriky AUC. Matice záměn dále slouží pouze pro srovnání modelů se stejným nastavením prahové vzdálenosti.

V rámci architektury byly měněny počty konvolučních vrstev, vstupní rozlišení sítě, počty filtrů v jednotlivých vrstvách a v neposlední řadě také počet samotných extrahovaných příznaků. Jako výchozí bod byla zvolena architektura LeNet [33], která byla také použita autory sítě. Postup volby hyperparametrů byl popsán v předchozí kapitole 3.1.1 a seznam finálních zvolených konfigurací lze vidět v přehledové tabulce 3.2. Zvolené hyperparametry pro pohled zboku jsou uvedeny v příloze B.

Při výběru modelů a pro učení byl použit shodný *standardní dataset* jako v

Tab. 3.2: Nastavení hyperparametrů tří vybraných reprezentativních modelů pro detekci anomálií při pohledu shora (*základní počet filtrů v první konvoluční vrstvě je 16. Koeficient znamená, jakým číslem byla tato hodnota vynásobena. V dalších vrstvách je pak vždy počet filtrů dvojnásobný oproti předchozí vrstvě)

Podmínka výběru	Nejrychlejší	Kompromis	Nejpřesnější
Parametry sítě			
Název sítě	lotus_LeNet_single	lotus_2conv	lotus_4conv
Počet konvolučních vrstev	3	2	4
Rozlišení na vstupu [px]	16x16	32x32	256x256
Velikost filtru	5	5	5
Koeficient filtrů*	2	2	2
Parametry učení			
Extrahované příznaky	128	256	256
Optimizér	Adam	Adam	Adam
Learning rate	0,0001	0,001	0,0001
Počet epoch	100	100	100
Epochy pro dělení LR	25; 50; 75	25; 50; 75	25; 50; 75
Velikost dávky	10	10	10
Rozpad vah	1^{-6}	1^{-6}	1^{-5}
Parametry předtrénování			
Optimizér	Adam	Adam	Adam
Learning rate	0,001	0,001	0,001
Počet epoch	100	100	100
Epochy pro dělení LR	-	-	-
Velikost dávky	100	100	100
Rozpad vah	1^{-5}	1^{-5}	1^{-5}

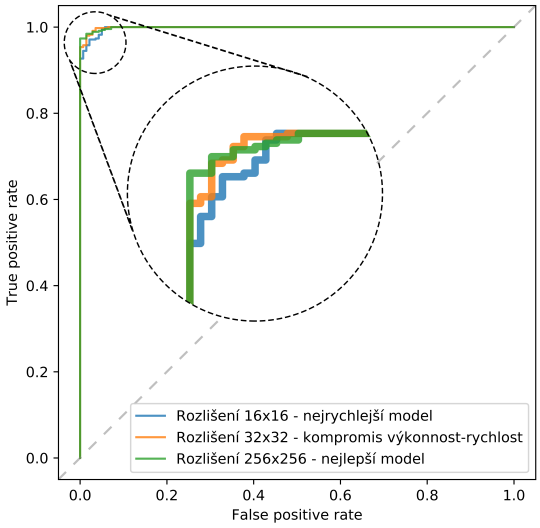
předchozím případě. Z těchto experimentů byly vybrány reprezentativní modely.

Pro pohled shora byl vybrán model se vstupním rozlišením 256x256 px a 4 konvolučními vrstvami. Tento model dosahoval nejlepších výsledků, avšak jedná se zároveň o nejsložitější model. Z toho důvodu byl pro rychlostní porovnání vybrán také jeden z nejrychlejších modelů s rozlišením na vstupu 16x16 px a 2 konvolučními vrstvami. Dále byl ještě vybrán model, který představoval kompromis mezi rychlostí a přesností; jedná se o model se vstupním rozlišením 32x32 px a 3 konvolučními vrstvami.

Při výběru vhodného nastavení hyperparametrů pro pohled zboku dosahoval nejlepších výsledků model s rozlišením 32x32 px a pro porovnání byl vybrán také model

s rozlišením 16x16 px. Modely s vyšším rozlišením nebyly vybrány, protože jsou výpočetně náročnější a v případě bočního pohledu nedosahovaly lepších výsledků.

Na následujícím obrázku 3.3 a v tabulce 3.3 lze vidět srovnání ROC křivek a výkonnostních metrik pro zvolené modely.



Obr. 3.3: Srovnání ROC křivek zvolených konfigurací hyperparametrů (pohled shora)

Tab. 3.3: Numerické srovnání zvolených konfigurací hyperparametrů - pohled shora (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Model	**Nejrychlejší**	**Kompromisní**	**Nejpřesnější**
AUC [%]	99,802	99,897	99,907
TPR [%]	92,699	95,354	97,345
FPR [%]	0,000	0,000	0,000
ACC [%]	94,426	96,453	97,973

Výpočetní náročnost jednotlivých modelů je obecně silně ovlivněna dostupným HW, použitou knihovnou a také možnostmi paralelizace, které jsou silně vázány na architekturu samotné sítě. V čase učení má také velký vliv mohutnost učícího datasetu. Je-li tedy možné provést experimentální ověření výpočetní náročnosti jednotlivých modelů, pak by tento postup měl být preferován před pouhým teoretickým odhadem výpočetní náročnosti.

V následující tabulce 3.4 jsou uvedeny časové náročnosti učení na PC Neuron (CPU: AMD Ryzen 5 3600, RAM: 32 GB, GPU: nVidia RTX 2080 Ti (12 GB VRAM)) a to na GPU, CPU a na 1 CPU jádře s vypnutou optimalizací. Časová náročnost vyhodnocování je uvedena v tabulce 3.5.

Uvedena je pouze tabulka hodnot zjištěných na modelech učících se na pohled shora (Tabulka 3.4). Rozdíly mezi modely učenými na pohled shora a zboku byly zanedbatelné (pro příklad, rozdíl mezi shodnou architekturou byl při 1000 trénovacích vzorech maximálně 4 sekundy a někdy byl rychlejší model pohledu shora, jindy zase pohledu zboku. Nebylo tedy možné vysledovat jednoznačnou závislost).

Při vyhodnocování byly průměrné časy pro vyhodnocení 1 vzoru shodné, nehledě na počet vzorů celkově přítomných v testovacích datech. V přehledové tabulce 3.5 jsou tudíž uvedeny pouze průměrné časy nutné pro vyhodnocení 1 vzoru.

Tab. 3.4: Srovnání časové náročnosti učení modelu v závislosti na množství trénovacích dat a použité výpočetní jednotce

			Doba potřebná k naučení [hod:min]		
		Počet vzorů	**GPU**	**CPU**	**CPU 1 jádro**
Model	**Nejsložitější**	100	0:09	1:14	6:01
		500	0:32	6:07	19:41
		1000	1:04	12:11	39:06
	Kompromisní	100	0:03	0:03	0:04
		500	0:13	0:16	0:21
		1000	0:25	0:32	0:42
	Nejrychlejší	100	0:02	0:03	0:03
		500	0:12	0:13	0:16
		1000	0:24	0:26	0:31

Z uvedených tabulek je vidět, že při použití složitějšího modelu se dramaticky zvyšují časové nároky jak na provádění učení, tak i vyhodnocování. V případě nejrychlejšího (a nejjednoduššího) modelu byl čas učení na 1 jádře CPU oproti GPU pouze 1,29krát vyšší, zatímco v případě nejsložitějšího modelu byla časová náročnost vyšší 36,65krát.

Při vyhodnocování již nebyly tyto rozdíly natolik patrné, avšak i zde byl nejrychlejší model na 1 jádře CPU pouze o 4,7 % pomalejší, zatímco nejpřesnější model zaznamenal zpomalení o 90,1 %.

Důvod těchto rozdílů je patrně ten, že v případě složitého modelu je nutno optimalizovat mnohem více parametrů a aplikace má mnohem více možností efektivně

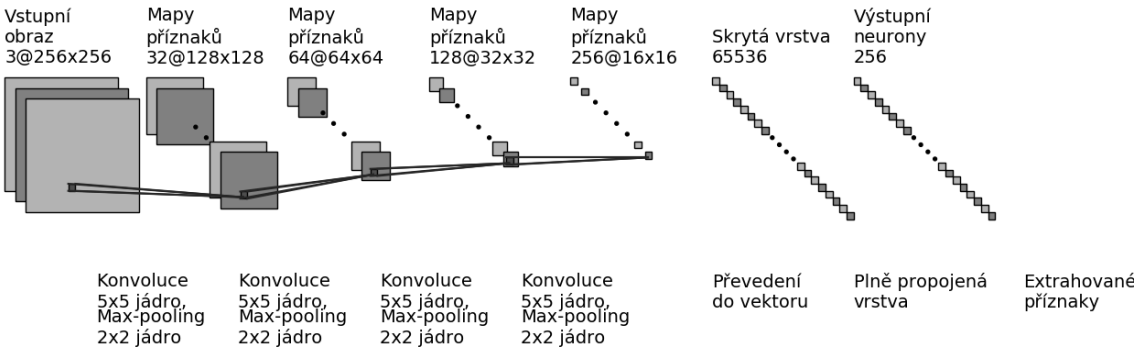
Tab. 3.5: Srovnání časové náročnosti vyhodnocení vzoru v závislosti na typu modelu a výpočetní jednotce

		Doba potřebná k vyhodnocení jednoho vzoru [ms]		
		GPU	**CPU**	**CPU 1 jádro**
Model	**Nejpřesnější**	7,130	21,781	71,721
	Kompromisní	3,721	3,974	4,243
	Nejrychlejší	3,445	3,545	3,615

paralelizovat výpočet. Naproti tomu jednoduché modely mají podstatně méně parametrů a nejsou tedy schopny grafickou akceleraci využít natolik, aby byl rozdíl oproti sekvenčnímu zpracování pomocí pouze procesorového jádra dostatečně patrný.

Na obrázku 3.4 je vidět vizualizace architektury nejúspěšnějšího z modelů. Pro ostatní modely je architektura obdobná, jen se mění počet konvolučních vrstev, velikost vstupního obrazu (a v návaznosti na ni také velikosti jednotlivých příznakových map) a počet extrahovaných příznaků v poslední vrstvě. Výše uvedená tabulka 3.2 uvádí tyto parametry v části *parametry sítě*.

V případě nejsložitějšího modelu je nutno optimalizovat celkem 18 938 816 parametrů, kompromisní model potom obsahuje 1 162 176 parametrů a nejrychlejší model potom pouze 323 936 parametrů. Je tedy zřejmé, že při použití grafické akcelerace je u nejsložitějšího modelů mnohem větší potenciál pro paralelizaci úlohy, než v případě modelu jednoduchého (navíc, ne všechny kroky učení lze efektivně paralelizovat).



Obr. 3.4: Ukázka architektury sítě (256x256 vstupní obraz, 4 konvoluční vrstvy, 256 extrahovaných příznaků)

Po domluvě s vedoucím bylo rozhodnuto, že následující experimenty budou již

prováděny pouze na nejpřesnějším modelu pro každý z pohledů. Toto rozhodnutí bylo provedeno proto, aby mohly dále v práci být prezentovány pouze nejrelevantnější výsledky a nebyly zatíženy hodnotami dosaženými na suboptimálních klasifikátorech.

3.1.3 Nulová hypotéza

Jak již bylo uvedeno v kapitole 1.1, v technické praxi se očekává, že normální třída je tvořena OK vzory a je homogenní a jasně definovaná. Naproti tomu anomálie by měly být tvořeny NOK vzory, a tudíž nesplňovat podmínku homogenity a úzké definice.

Pro ověření tohoto tvrzení byl proveden experiment nulové hypotézy, tedy experiment, který se snaží prokázat hypotézu opačnou.

Nulová hypotéza tedy zní: Pokud bude normální třída tvořena pouze NOK vzory a OK vzory budou považovány za anomálie, pak bude výkonnost modelu srovnatelná s modelem, kde je normální třída tvořena OK vzory a NOK vzory jsou považovány za anomálie.

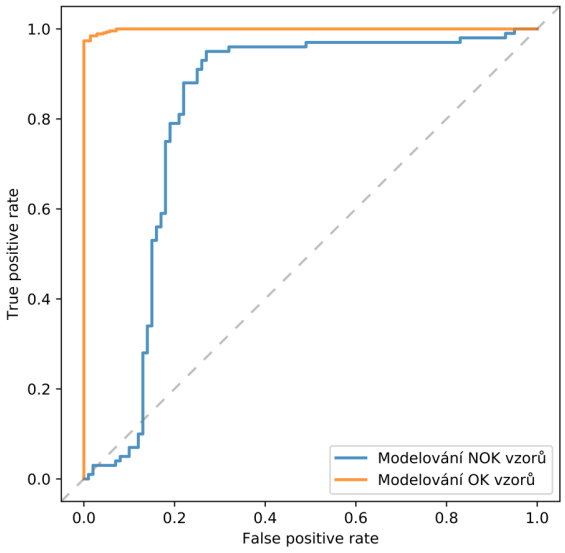
Pro učení modelu typického pro technickou praxi byla použita data ze *standardního datasetu*. Pro učení modelu nulové hypotézy bylo použito stejného množství NOK vzorů, které byly prohlášeny za normální třídu.

Pro účely vyhodnocení byl použit dataset čítající pouze 100 OK a 100 NOK prvků, důvodem je zamezení použití trénovacích dat pro testování a zároveň použití stejných testovacích dat pro oba modely (avšak s navzájem opačnou anotací).

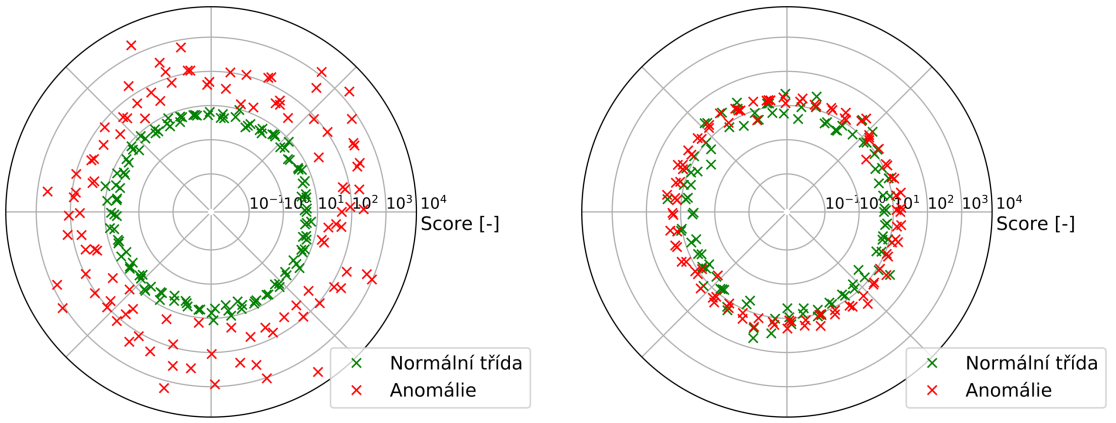
Na obrázku 3.5 lze vidět porovnání ROC křivek těchto dvou modelů. Je zřejmé, že při modelování NOK vzorů má síť problém tyto vzory odlišit, a i při velké hraniční vzdálenosti jich velké množství klasifikuje nesprávně a zároveň není schopna klasifikovat OK vzory. Naproti tomu model modelující OK třídu již při malé vzdálenosti od středu je schopen správně klasifikovat většinu NOK vzorů i OK vzorů.

Pokud se testovací dataset zobrazí v kontextu jednotlivých prvků a jejich vzdáleností od středu po průchodu sítí (obrázek 3.6), pak lze relativně jednoduše vysvětlit toto chování. Ačkoliv jsou obě sítě schopny mapovat vzory své trénovací třídy blízko středu (zelené prvky), tak síť modelující OK vzory mapuje NOK vzory daleko od této množiny (často o několik řádů dále). Naproti tomu model modelující NOK vzory mapuje OK vzory (anomálie) velmi blízko své normální množině a nelze tedy jednoduše najít hranici mezi těmito množinami.

Toto chování tedy vyvrací nulovou hypotézu a potvrzuje teoretické tvrzení, že nelze modelovat NOK prvky jako plnohodnotnou třídu. Dalším závěrem, který lze z tohoto experimentu vyvodit je fakt, že metoda DeepSVDD pracuje správně a



Obr. 3.5: Modelování NOK vzorů X Modelování OK vzorů (Pohled shora, rozlišení 256x256)



a) b)

Obr. 3.6: Porovnání vzdálenosti mapování prvků normální a anomální třídy od středu hyperkoule. a) Model modelující OK vzory, b) Model modelující NOK vzory (výstupem modelu je pouze vzdálenost od středu, poloha na kružnici je pouze pro lepší vizualizaci výsledků)

skutečně z trénovacích dat extrahuje společné příznaky (viz kapitola 1.3.3). Při pohledu na NOK sušenky je totiž vidět, že jednotlivé anomálie jsou velmi rozmanité, ale přesto jsou pro všechny tyto sušenky společné jejich neporušené části (tento koncept lépe popisuje obrázek 3.7). Síť tedy bude mít tendenci i z NOK dat extrahovat znaky společné s OK třídou, což lze vidět na obrázku 3.6.

Obr. 3.7: Ukázka společných rysů OK vzoru a NOK vzorů dvou různých kategorií

3.1.4 Resubstituční chyba

Experimenty provedené v této kapitole si kladly za cíl zjistit, jak velký vliv na vyhodnocovací metriky bude mít resubstituce (znovupoužití trénovacích dat pro testování).

V kontextu binárních úloh se očekává, že výsledný klasifikátor bude nadhodnocovat přesnost, jelikož bude testován na datech, jejichž anotaci měl k dispozici již v průběhu učení [35].

Vzhledem k definici přesnosti (viz 1.4.1) se dá určitý nárůst také očekávat při použití jednotřídního klasifikátoru. Naopak metrika senzitivity dává do kontextu detekované anomálie vůči všem testovacím anomáliím. Jelikož při resubstituci je testovací dataset obohacen pouze o normální vzory, tak se dá očekávat, že tyto metriky zůstanou nezměněny, nebo budou dokonce negativně ovlivněny.

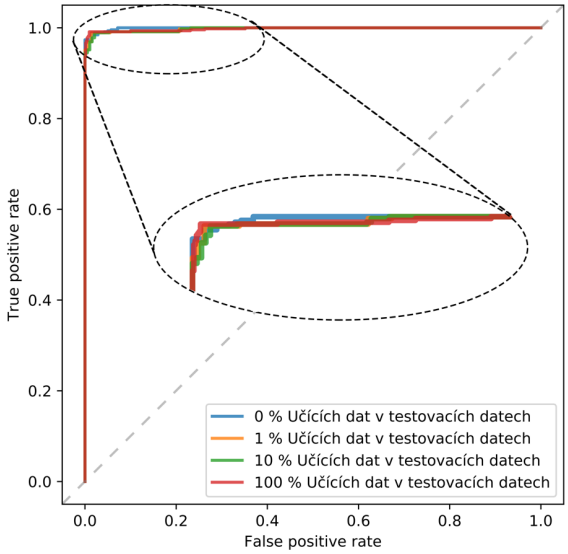
Pro účely určení míry vlivu resubstituce byl model naučený na *standardním datasetu* otestován na 4 různých testovacích množinách.

První množina odpovídala *standardnímu datasetu* a neobsahovala tedy žádná učící data. Druhá množina odpovídala *standardnímu datasetu*, jehož testovací data byla obohacena o 1 % trénovacích vzorů (5 vzorů). Podobně třetí množina byla rozšířena o 10 % trénovacích vzorů (45 vzorů) a čtvrtá množina byla potom pro testování obohacena o celou trénovací množinu.

Srovnání ROC křivek těchto experimentů je vidět na obrázku 3.8. Matice záměn pro shodnou hraniční vzdálenost lze vidět na obrázcích 3.9. Je vidět, že na počet detekovaných anomálií nemá použití trénovacích dat vliv (což bylo samozřejmě očekávané, neboť nebyl testovací dataset o žádné anomálie rozšířen)

Při pohledu na ostatní metriky (tabulka 3.6) je vidět, že pro definovaný poměr správně určených anomálií jsou ovlivněny metriky AUC a ACC, které ovlivňuje mohutnost nové testovací množiny. Metrika celkové přesnosti ale není pro ohodnocení detektoru anomálií tak důležitá jako například senzitivita nebo FPR, jelikož pro detektor anomálií je v prvé řadě klíčový počet správně detekovaných anomálií

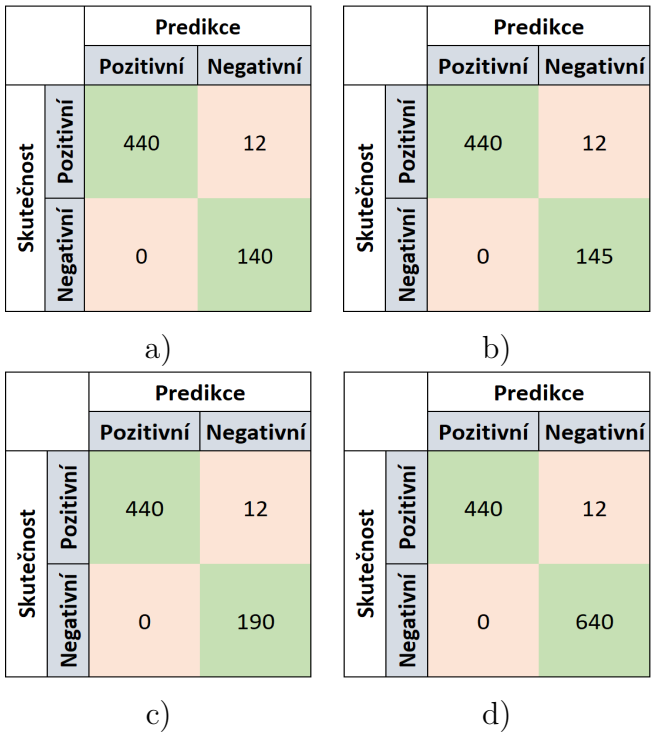
(senzitivita) a na druhém místě je počet falešně detekovaných anomálií (FPR). Jak jsou senzitivita a FPR ovlivněny při modifikaci hraniční vzdálenosti lze vysledovat z obrázku porovnání ROC křivek (obrázek 3.8).



Obr. 3.8: Srovnání vlivu resubstituce (Pohled shora, rozlišení 256x256)

Tab. 3.6: Numerické srovnání modelů s různou úrovní resubstituce (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Úroveň resubstituce	**0 %**	**1 %**	**10 %**	**100 %**
AUC [%]	99,907	99,980	99,985	99,965
TPR [%]	97,345	97,345	97,345	97,345
FPR [%]	0,000	0,000	0,000	0,000
ACC [%]	97,980	97,980	98,131	98,901



Obr. 3.9: Matice záměn pro modely s různou úrovní resubstituce. Úroveň resubstituce: a) 0 %, b) 1 %, c) 10 %, d) 100 %

3.2 Experimenty - vliv změny definice anomálií na síť

V předchozí sekci bylo určeno vhodné nastavení hyperparametrů, byla ověřena funkčnost sítě a byl analyzován vliv resubstituce. Tato kapitola využije výstupy předchozích experimentů a bude zjišťovat, jaký vliv mají použitá data a jejich anotace (normální třída x anomálie) na schopnosti výsledného modelu.

Zatímco do teď provedené experimenty pracovaly s OK a NOK vzory jako s neměnnými množinami, nyní bude cílem zkoumat vliv mísení těchto množin (kapitola 3.2.1) nebo jejich dalšího dělení (kapitola 3.2.2 a 3.2.3)

3.2.1 Vliv nesprávné anotace

Při sbírání velkých datasetů se může stát (a často se také stává), že některá data budou nasnímána chybně nebo budou nesprávně anotována. Dojde tak k situaci, kdy normální třída, která tvoří trénovací množinu, neobsahuje pouze OK vzory, ale také určitý podíl NOK vzorů.

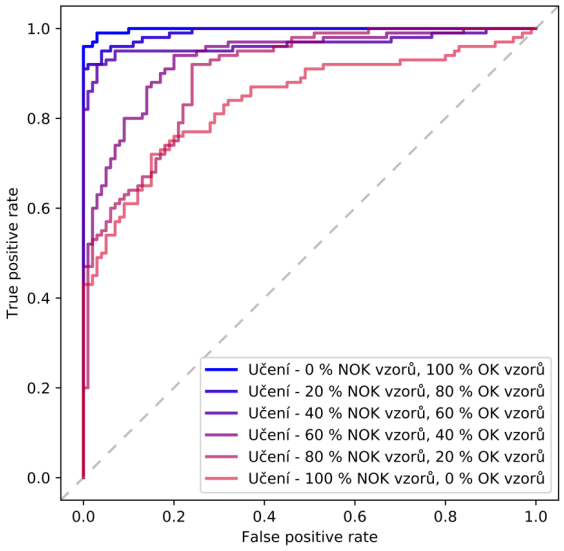
Jak již bylo zjištěno experimentem nulové hypotézy v kapitole 3.1.3, při učení na NOK prvcích má síť tendenci extrahovat znaky, které jsou společné také pro OK

třídu. Z toho důvodu se dá očekávat, že i při učení se nesprávně anotovanými daty bude síť dosahovat relativně dobrých výsledků.

Oproti testu nulové hypotézy, kde byla normální třída tvořena NOK vzory a anomálie byly tvořeny OK vzory, v tomto experimentu jsou anomálie tvořeny NOK vzory, normální třída je potom v průběhu učení tvořena mixem OK a NOK vzorů a v průběhu validace a testování již obsahuje jen OK vzory.

Pro účely experimentů bylo vytvořeno 6 datasetů s různou úrovní nesprávně anotovaných trénovacích dat. Postupně byl po 20 % měněn poměr OK prvků a NOK prvků, které tvoří trénovací dataset. Validační a testovací data byla beze změny a sestávala za stejného počtu prvků normální třídy (OK vzory) a anomálií (NOK vzory).

Srovnání ROC křivek těchto 6 různých modelů lze vidět na obrázku 3.10 a numerické výsledky potom v tabulce 3.7. Z výsledků je patrné, že se snižujícím se poměrem NOK vzorů se přesnost modelu rychle zvyšuje avšak i model naučený na čistě nesprávně anotovaných datech dosahuje AUC 83,95 %. Je tedy vidět, že metoda DeepSVDD je velmi robustní a malé množství nesprávně anotovaných dat se na výsledném modelu neprojeví nijak výrazně.



Obr. 3.10: Srovnání ROC křivek pro modely naučené na různý poměr nesprávně anotovaných dat (Pohled shora, rozlišení 256x256)

Tab. 3.7: Srovnání výkonnostních metrik pro modely naučené na různý poměr nesprávně anotovaných dat - Pohled shora, rozlišení 256x256 (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Učících NOK vzorů [%]	**0**	**20**	**40**	**60**	**80**	**100**
Učících OK vzorů [%]	**100**	**80**	**60**	**40**	**20**	**0**
AUC [%]	99,82	99,14	96,49	93,44	89,9	83,95
TPR [%]	96,00	91,00	92,00	90,00	92,0	91,00
FPR [%]	0,00	0,00	3,00	17,00	24,0	49,00
ACC [%]	98,00	95,50	94,50	86,50	84,0	71,00

3.2.2 Citlivostní analýza

V rámci následujících experimentů byla provedena analýza citlivosti sítě. Tato analýza spočívá ve snaze rozdělit třídu OK sušenek na normální třídu a anomálie, přičemž za normální třídu budou považovány naprosto perfektní vzory, zatímco za anomálie budou prohlášeny vzory, které jsou sice v pořádku ale nedají se nazývat bezchybnými (například sušenka není perfektně kulatá).

Cílem je zjistit, zda je síť schopná detekovat i velmi malé anomálie. Dalším cílem je následně zjistit, zda je pro učení sítě lepší použít menší množství perfektních vzorů, nebo zda větší roli hraje počet tréninkových vzorů.

Rozdělení na perfektní (normální třída) a ostatní OK (anomálie) sušenky bylo provedeno dvěma technicky rozdílnými způsoby. Jeden dataset byl vybírán ručně podle subjektivního hodnocení, zda se jedná o perfektní sušenku či nikoliv. Druhý dataset byl vybrán automaticky tak, že byl naučen model za pomocí všech OK vzorů a na těchto vzorech byl následně také otestován. Jako perfektní sušenky byly vybrány ty, které se mapovaly nejblíže středu hyperkoule.

Takto byly vytvořeny 2 skupiny datasetů, kdy každá obsahovala 3 datasety s různým počtem perfektních sušenek (250, 300, 450). Z těchto počtů bylo vždy 50 vyhrazeno pro testování, 30 pro validaci a zbylé vzory byly použity pro učení.

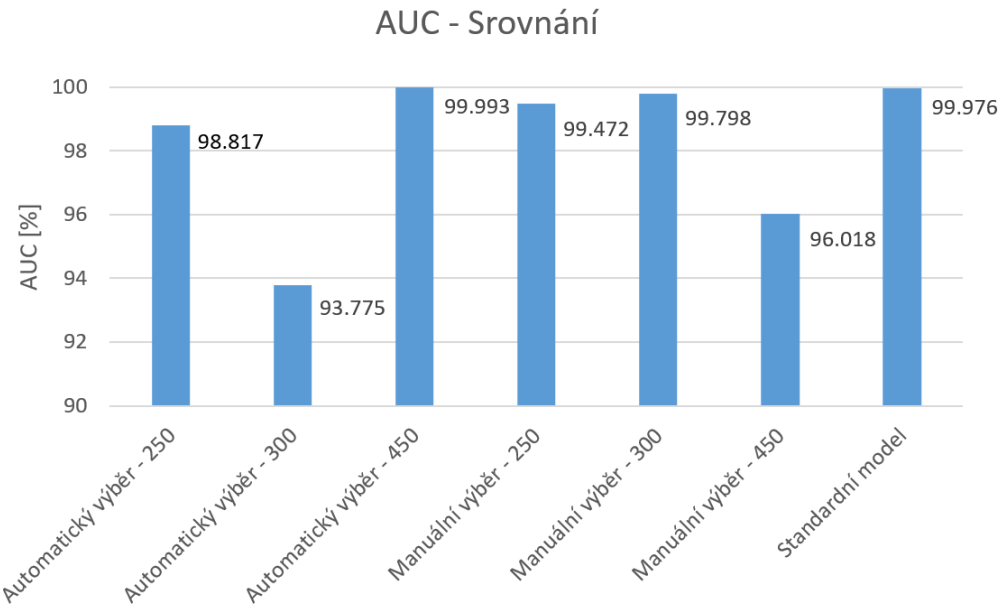
Tabulka 3.8 obsahuje výkonnostní metriky pro těchto 6 modelů. Ani v případě automatického ani manuálního výběru nebyla síť schopna spolehlivě odlišit perfektní vzory od ostatních OK vzorů. Tyto výsledky naznačují, že OK vzory datasetu vytvořeného v této práci tvoří natolik rigidní množinu, že je nelze dále dělit. Další možností interpretace je, že rozdíly mezi perfektními a ostatními OK prvky jsou natolik malé, že jsou pod rozlišovací schopností sítě. Tyto hypotézy podporuje skutečnost, že obdobných výsledků bylo dosaženo při obou pohledech i obou metodách

výběru.

Obrázek B.7 potom ukazuje AUC pro jednotlivé modely při jejich testování na *standardním datasetu*. Zde se ukazuje, že všechny modely dosáhly alespoň uspokojivého AUC. Z výsledků však není zřejmá jednoznačná závislost a nelze tedy s jistotou tvrdit, zda je důležitější čistota trénovacích dat nebo jejich počet.

Tab. 3.8: Srovnání výkonnostních metrik pro modely naučené na perfektní představitele OK vzorů (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

	Automatický výběr			**Manuální výběr**		
Počet perfektních reprezentantů	**250**	**300**	**450**	**250**	**300**	**450**
AUC [%]	47,783	51,781	51,592	55,197	59,376	63,495
TPR [%]	90,887	90,730	91,262	95,567	90,730	90,291
FPR [%]	90,000	90,000	92,000	88,000	82,000	78,000
ACC [%]	82,018	80,788	75,000	86,404	81,773	76,953



Obr. 3.11: Srovnání jednotlivých modelů naučených na perfektních představitelích OK vzorů při jejich testování na *Standardním datasetu* (pro zvýšení přehlednosti je osa y omezená na rozsah 90-100 %)

3.2.3 Vliv typu testovacích anomálií

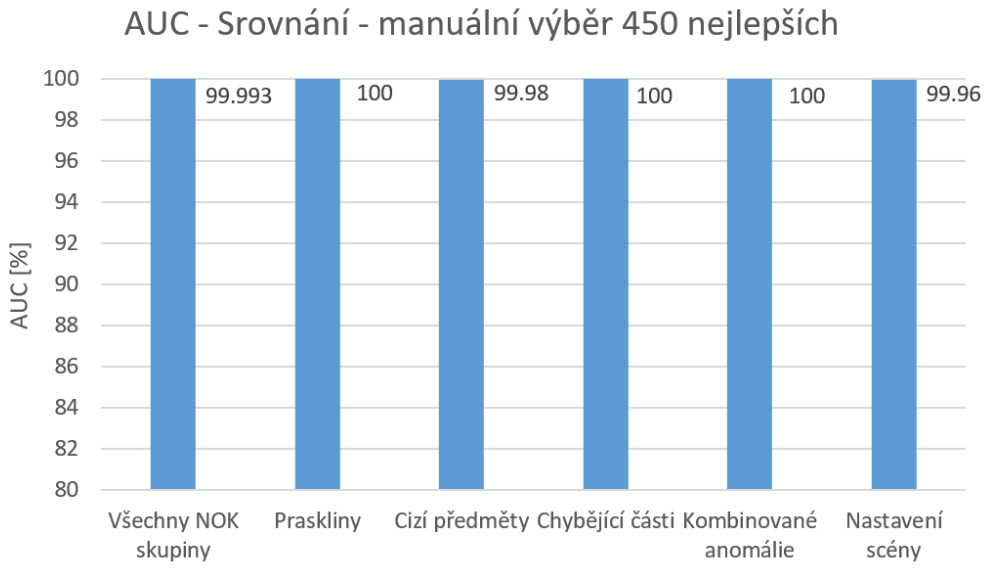
Posledním provedeným experimentem v rámci této práce je zjištění vlivu definice anomálií v testovacích datech na celkový odhad výkonnosti modelu.

Jelikož jsou anomální prvky obvykle špatně dostupné nebo jsou jen omezeného typu, tak je nutné vyšetřit, zda je možné na základě omezeného typu anomálií provádět kvalifikované odhady výkonnosti celého modelu.

Pro účely experimentů byly vytvořeny testovací sady, které vždy obsahovaly jen jeden typ NOK vzorů. Tyto datasety pak byly porovnány s datasetem, který obsahoval mix všech typů NOK vzorů.

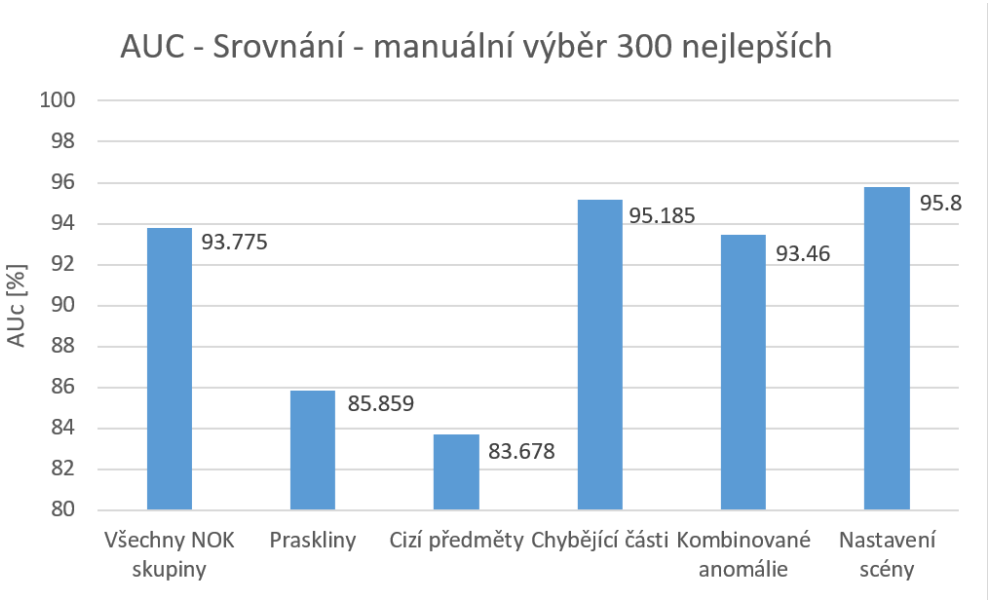
Testování probíhalo na 6 modelech vytvořených v předchozích experimentech, aby bylo vyšetřeno, jestli má výběr trénovacích OK vzorů nějaký zásadní vliv na detekci určitého typu anomálií. Dále bylo testováno také na modelu naučeném na standardním datasetu (tedy bez cíleného výběru perfektních představitelů normální třídy).

Plochy pod ROC křivkou pro jednotlivé modely jsou obsahem přiloženého DVD č.1 ve složce *Podklady k experimentům/Kapitola 3/Kapitola 3.2/3.2.3*, zde v textu jsou pouze uvedeny 2 modely pro demonstraci výsledků (obrázky 3.12 a 3.13).



Obr. 3.12: Srovnání odezev modelu naučeném na manuálním výběru 450 perfektních reprezentantů OK třídy. Ukazuje odezvu na všech NOK vzorech bez rozlišení jejich kategorie ve srovnání s odezvami na samostatné kategorie (pro zvýšení přehlednosti je osa y omezená na rozsah 80-100 %)

Výsledky naznačují, že pro modely s velmi dobrou úspěšností (nad 99 %) lze zobecnit výsledky získané na jednom typu anomálie na celé spektrum. Na druhou



Obr. 3.13: Srovnání odezev modelu naučeném na manuálním výběru 300 perfektních reprezentantů OK třídy. Ukazuje odezvu na všech NOK vzorech bez rozlišení jejich kategorie ve srovnání s odezvami na samostatné kategorie (pro zvýšení přehlednosti je osa y omezená na rozsah 80-100 %)

stranu horší modely tuto vlastnost nesdílí a reagují výrazně lépe na některé typy anomálií.

Pro praktické využití tak vyvstává otázka, zda je v případě modelu s vysokou přesností na jednom typu anomálie zobecnění opodstatněné, protože se jedná o dobře naučený model, nebo zda by bylo zobecnění zavádějící, protože model reaguje dobře pouze na tento konkrétní typ anomálií. Odpověď na tuto otázku nelze zjistit bez znalosti všech možných anomálií a v technické praxi je tak toto zjištění jen těžko aplikovatelné.

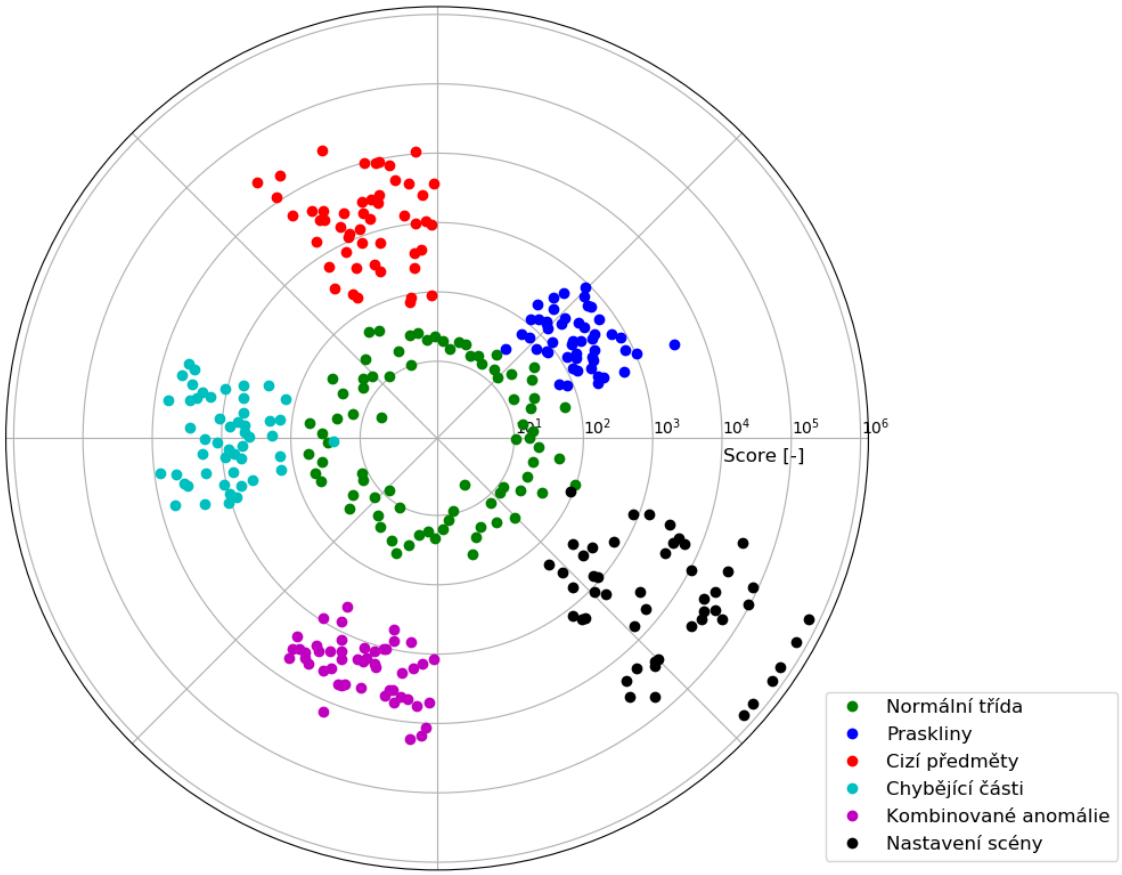
Dále bylo zkoumáno, zda lze čistě ze vzdálenosti mapování prvku od středu usuzovat na typ anomálie. Pro tyto účely bylo vytvořeno zobrazení vzdálenosti mapovaných prvků od středu s rozlišením jejich typu. Toto zobrazení lze vidět na obrázku 3.14. Nutno podotknout, že výstupem modelu je pouze vzdálenost prvku od středu, úhel vykreslení jednotlivých bodů je měněn pouze z důvodu lepší vizualizace výsledků.

Je vidět, že jednotlivé prvky anomálií stejného druhu jsou často mapovány na hodnoty, které přesahují mnoho řádů. Ze vzdálenosti jednoho prvku tedy nelze usuzovat typ jeho anomálie.

Avšak při vyhodnocení množství vzorů jako celku se již dá na některé typy anomálií usuzovat. V případě zde prezentovaných výsledků lze rozlišit shluk *prasklin*,

který je mapován na hodnoty v rozsahu $10^2 - 10^3$, shluk *kombinovaných anomálií* (mapování $10^3 - 10^4$) a dále shluk *nastavení scény* jehož mapování zasahuje do rozsahu hodnot $10^2 - 10^6$. Mezi shluky *cizích předmětů* a *chybějících částí* nelze jednoznačně rozlišit.

Je nutné znovu upozornit, že tato klasifikace typu anomálií je možná pouze v případě, že je vyhodnocení provedeno na skupině prvků, u které není znám typ jejich anomálie, ale je známo, že se jedná pouze o jeden konkrétní typ.



Obr. 3.14: Srovnání mapovací vzdálenosti jednotlivých typů anomálií (výstupem modelu je pouze vzdálenost od středu. Poloha na kružnici je modifikována pouze z důvodu lepší vizualizace výsledků)

3.3 Zhodnocení experimentů

Tato kapitola se zabývala experimentální částí práce. V první části bylo cílem zjistit základní vlastnosti sítě a nalézt vhodný pracovní bod pro následující experimenty.

Byla popsána metodika volby hyperparametrů, která měla vést k nalezení optimálního pracovního bodu. Obrázky 3.1 a 3.2 poukazují na limity srovnání modelů pro detekci anomálií pouze na základě matice záměn.

Bylo zvoleno nastavení hyperparametrů, které pro daný dataset generuje nejlepší model. Dále bylo také zvoleno nastavení, které upřednostňuje jednoduchost a nízkou výpočetní náročnost výsledného modelu. Srovnání hyperparametrů těchto modelů je uvedeno v tabulce 3.2 a pro pohled zboku potom v příloze B.

Zvolené modely byly také porovnány na základě jejich výpočetní náročnosti. Byla zkoumána časová náročnost učení s ohledem na počet vzorů v trénovací množině a také s ohledem na použité výpočetní jednotce. Výpočetní náročnosti pro pohled shora a zboku byly srovnatelné a jsou uvedeny v tabulkách 3.4 a 3.5.

Jelikož měly všechny pokusy stejné nastavení velikosti dávky, tak nárůst výpočetního času se změnou počtu trénovacích vzorů byl lineární a lze tedy pro dané nastavení a výpočetní jednotku s velkou přesností odhadnout celkovou dobu učení velkého datasetu (například 1 000 000 vzorů) extrapolací doby potřebné k naučení datasetu menšího (například 100 vzorů).

Při změně výpočetní jednotky se ukázalo, že modely s podstatně více parametry velmi dobře využívají možnosti paralelizace pramenící z využití grafické akcelerace. Naproti tomu jednodušší modely nemají dostatek parametrů, jejichž výpočet lze dobře paralelizovat na to, aby bylo jejich trénování výrazně urychleno při použití grafické karty. V tabulce 3.4 lze vidět, že zatímco při přechodu z GPU na 1 CPU jádro se doba pro trénování nejsložitějšího ze zvolených modelů zvýšila z 1:04 hodiny na 39:06 hodin, tak v případě nejjednoduššího z modelů narostl potřebný čas o pouhých 7 minut a to z 24 minut na 31 minut. Toto srovnání vyplývá ze srovnání při 1000 trénovacích vzorech. Srovnání pro ostatní časy lze vyčíst z tabulky 3.4.

Při srovnání času potřebného pro vyhodnocení modelu (viz tabulka 3.5) byl opět zaznamenán větší nárůst při vyhodnocování složitějších modelů oproti jednodušším, ale zde nebyl rozdíl natolik markantní. I v případě nejsložitějšího modelu bylo možné jej na 1 CPU jádře vyhodnotit za 71 ms, což odpovídá 14 FPS. V nejlepším případě bylo možno na GPU RTX 2080 Ti dosáhnout vyhodnocení nejjednoduššího modelu za 3,45 ms, což odpovídá možnosti detekovat a vyhodnocovat snímek v reálné aplikaci rychlostí 290 FPS.

Na základě dohody s vedoucím byly následující experimenty již prováděny na architektuře s nastavením hyperparametrů generujícím nejlepší celkové výsledky.

Cílem dalšího experimentu bylo vyhodnotit nulovou hypotézu a zjistit, zda je

možné modelovat NOK vzory se stejným úspěchem jako OK vzory. Obrázek 3.5 ukazuje srovnání ROC křivek modelující OK vzory, respektive NOK vzory. Z výsledků je patrné, že NOK vzory nelze spolehlivě modelovat. Obrázky 3.6 a 3.7 potom blíže poukazují na to, že i v případě NOK trénovacích vzorů má DeepSVDD tendenci kvalitně extrahovat společné znaky mezi těmito vzory.

Dále byl zkoumán vliv znovupoužití trénovacích dat pro testování. Při pohledu na tabulku 3.6 je vidět, že s vyšším podílem opakovaně použitých dat mírně roste celkové AUC, což vede na hypotézu, že model má tendenci nadhodnocovat chybu. Avšak při pohledu tvar ROC křivek na obrázku 3.8, je vidět, že model se 100 % trénovacích dat v testovací množině generuje při nastavení větší prahové vzdálenosti více falešně pozitivních detekcí. To znamená, že některé vzory trénovací množiny se mapují dále od středu hyperkoule než vzory původní testovací množiny (data nezatížena resubstituční chybou). Lze tedy říci, že ačkoliv při opakovaném použití trénovacích dat pro testování obecně dochází k nadhodnocení základních metrik (AUC, ACC), tak počet skutečně odhalených anomálií při stejné prahové vzdálenosti zůstane stejný a počet falešně detekovaných anomálií zůstane buďto stejný (metrika FPR nebyla resubstituční chybou ovlivněna) nebo se dokonce zvýší.

Druhá část experimentů se zaměřovala na vliv změny definice anomálií na model a jeho odhad výkonnosti. Zkoumal se vliv jak změny definice anomálií v trénovacích, tak i v testovacích datech.

Prvním z experimentů bylo zjištění vlivu nesprávné anotace. Bylo zkoumáno, jak velký poměr nesprávně anotovaných dat (NOK vzory, které byly prohlášeny za součást normální třídy ve fázi učení) může trénovací množina ještě obsahovat, aby byl model použitelný.

Bylo zjištěno, že i při trénování na pouze nesprávně anotovaná data dosahuje model AUC 83,95 % což je vzhledem k trénovací množině dobrý výsledek, ale rozhodně se nejedná o použitelný model. V případě přítomnosti 20 % nesprávně anotovaných dat již model dosahoval AUC 99,14 %. Ačkoliv se stále nejedná o výsledky použitelné v průmyslovém prostředí, tak lze z těchto výsledků, které jsou detailněji uvedeny v tabulce 3.7, usuzovat, že malé množství nesprávně anotovaných trénovacích dat bude mít na schopnosti klasifikátoru jen zanedbatelný vliv.

Dále byla provedena citlivostní analýza, která měla za cíl zjistit, zda je DeepSVDD (konkrétně zvolená architektura extraktoru a nastavení hyperparametrů) schopna detekovat i velmi malé odchylky.

Z experimentů vyšlo najevo, že OK vzory tvoří velice rigidní třídu, kterou nelze dále dělit (viz tabulka 3.8) a rozlišení velice malých vad by vyžadovalo model s větší kapacitou (například více vrstev, více filtrů ve vrstvách nebo větší vstupní rozlišení). Naučit však komplexnější model by také vyžadovalo podstatně více trénovacích vzorů a tuto hypotézu tedy nelze ověřit jen na základě dat vytvořených

pro účely této práce.

V neposlední řadě byl testován vliv anomálií použitých v testovací fázi. Cílem bylo zjistit, zda je možné na základě omezeného typu anomálií provést robustní odhad výkonnosti modelu a také zda lze pouze na základě mapovací vzdálenosti jednotlivých prvků určit typ anomálie.

Z tohoto experimentu vyšlo najevo, že pro dobře naučené modely je možné s velkou přesností zobecnit výsledky pro jeden typ anomálií na celé anomální spektrum (viz graf 3.12). Na druhou stranu hůře naučené modely tuto vlastnost nesdílí a nelze tedy zobecnění provést (viz graf na obrázku 3.13). Využití tohoto poznatku v technické praxi je potom problematické, jelikož obecně nelze zjistit, zda výsledky na daném typu anomálií jsou dobré, jelikož byl model naučen kvalitně (zobecnění je opodstatněné), nebo zda se jedná o jeden z mála typů anomálií, které model dokáže detekovat (zobecnění není opodstatněné).

Při velice úzkém datasetu z pohledu rozmanitosti tedy není vhodné dobré výsledky zobecňovat a vždy je lepší zajistit co nejrůznorodější povahu testovacích dat.

Dále se zkoumalo, zda lze z mapovací vzdálenosti usuzovat na typ anomálie. Obrázek 3.14 ukazuje, že i anomálie stejného typu se obecně mapují do velice různorodých vzdáleností (někdy i přes několik řádů) a nelze tedy čistě z metriky vzdálenosti od středu hyperkoule rozpoznat typ anomálie.

Dále však bylo zjištěno, že skupiny anomálií stejného typu se jako celek mapují do určitých rozlišitelných rozsahů vzdáleností. Tento poznatek lze aplikovat v případech, kdy je například z fyzické podstaty procesu jisté, že vzniklé anomálie budou stejného typu (například dlouhodobě kapající lepidlo). Potom lze tyto anomální prvky jako celek (například skupina 200 anomálií se stejným typem vady) rozpoznat a přiřadit k určitému typu anomálie. Grafické zobrazení tohoto fenoménu lze vidět na obrázku 3.14.

Závěr

V úvodu práce byly krátce rozebrány typy klasifikačních úloh. Dále byla provedena rešerše existujících metod pro jednotřídní klasifikaci. Jednotlivé metody byly dopodrobna popsány, byly diskutovány jejich výhody a nevýhody společně s případnými omezeními. Z těchto metod byla pro další práci zvolena metoda DeepSVDD.

Praktická část potom byla rozdělena na dvě části. První část se zabývala inspekčním systémem. Byla popsána poskytnutá platforma pro průmyslovou inspekci a její úpravy. Dále byl popsán poskytnutý software DeepSVDD a Anubis. V případě uživatelské aplikace Anubis byly provedeny rozsáhlé změny a optimalizace.

Pro účely následných experimentů byl vytvořen vlastní dataset. Tento dataset byl koncipován jako dvoupohledový a každá realizace produktu tedy obsahuje dva snímky (jeden shora a druhý zboku). Bylo navrženo a následně realizováno 7 anomálních tříd. Celkově obsahuje dataset 1 321 realizací.

Druhá část praktické části práce se zabývala experimenty s využitím DeepSVDD a datasetu, který byl pro tyto účely vytvořen. Byla otestována nulová hypotéza a bylo ověřeno, že anomálie skutečně tvoří heterogenní množinu a že je není možné spolehlivě modelovat.

Experimenty ověřující vliv definice anomálií na celkovou výkonnost modelu byly rozděleny na experimenty, které zjišťovaly vliv definice anomálií v trénovací fázi modelu a vliv definice anomálií v testovací fázi. V trénovací fázi bylo zkoumáno, jaký vliv má poměr špatně anotovaných dat v trénovací množině. Bylo zjištěno, že malé množství špatně anotovaných dat v trénovací množině má na schopnosti modelu zanedbatelný vliv. Dále byly provedeny experimenty citlivostní analýzy. Bylo zkoumáno, jak malé vady dokáže síť ještě detekovat. Z těchto experimentů vyšlo najevo, že není možné s modelem použitým v této práci dosáhnout kvalitní separace OK třídy na bezchybné realizace a realizace nižší jakosti.

Poslední část experimentů se zabývala změnou definice anomálií v testovací množině. Zde bylo cílem zjistit, zda je možné zobecnit metriky vypočtené na omezeném typu testovacích anomálií na celé anomální spektrum. Bylo zjištěno, že ačkoliv je v případě dobře naučeného modelu zobecnění teoreticky možné, tak nelze toto zobecnění uplatnit v praxi. V rámci těchto experimentů bylo také sledováno, zda je možné rozlišit mezi jednotlivými typy anomálií jen na základě jejich výsledné mapovací vzdálenosti. Bylo zjištěno, že tato klasifikace není pro jednotlivé vzory možná. Avšak při analýze skupiny anomálií, o které je a priori známo, že obsahuje jeden typ anomálií, lze z mapovací vzdálenosti na některé anomální typy usuzovat.

Pro porovnání výstupů jednotlivých experimentů bylo využito převážně ROC křivek a AUC metriky. Pro srovnání některých experimentů byla použita také matice záměn. Pro některé experimenty by samotná matice a z ní vycházející metriky mohly

podávat zavádějící výsledky, často proto byly upřednostňovány jiné metriky. Byla srovnána také časová náročnost zvolených modelů a jaký vliv na ni má použitá výpočetní jednotka a množství trénovacích dat.

Literatura

[1] HALVANI, Oren, Christian WINTER a Lukas GRANER. Unary and Binary Classification Approaches and their Implications for Authorship Verification. *CoRR*. **2019**(abs/1901.00399). Dostupné také z: http://arxiv.org/abs/1901.00399

[2] CHEN, Huang, Li YINING, Loy CHEN CHANGE a Tang XIAOOU. Learning Deep Representation for Imbalanced Classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* [online]. **2016**, 5375-5384 [cit. 2022-12-29]. Dostupné z: https://openaccess.thecvf.com/content_cvpr_2016/html/Huang_Learning_Deep_Representation_CVPR_2016_paper.html

[3] OLIVERI, Paolo. Class-modelling in food analytical chemistry: Development, sampling, optimisation and validation issues — A tutorial. *Analytica Chimica Acta*. 2017, **982**, 9-19. ISSN 00032670. Dostupné z: doi:10.1016/j.aca.2017.05.013

[4] SHILTON, A., S. RAJASEGARAR a M. PALANISWAMI. Combined multiclass classification and anomaly detection for large-scale Wireless Sensor Networks. *2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing* [online]. IEEE, 2013, 2013, 491-496 [cit. 2022-12-29]. ISBN 978-1-4673-5501-8. Dostupné z: doi:10.1109/ISSNIP.2013.6529839

[5] CHEN, Yushi, Hanlu JIANG, Chunyang LI, Xiuping JIA a Pedram GHAMISI. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing* [online]. 2016, **54**(10), 6232-6251 [cit. 2022-12-29]. ISSN 0196-2892. Dostupné z: doi:10.1109/TGRS.2016.2584107

[6] KUMAR, Ajitesh. *Difference: Binary, Multiclass & Multi-label Classification* [online]. In: . May 16, 2022 [cit. 2022-12-29]. Dostupné z: https://vitalflux.com/difference-binary-multi-class-multi-label-classification

[7] KHAN, Shehroz S. a Michael G. MADDEN. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review* [online]. 2014, **29**(3), 345-374 [cit. 2022-12-29]. ISSN 0269-8889. Dostupné z: doi:10.1017/S026988891300043X

[8] ZHANG, Kaitai, Bin WANG, Wei WANG, Fahad SOHRAB, Moncef GABBOUJ a C. C. Jay KUO. AnomalyHop: An SSL-based Image Anomaly Localization Method. *ArXiv*. 2021, **2021**(2105.03797). Dostupné z: doi:https://doi.org/10.48550/arXiv.2105.03797

[9] DERDE, M. P. a D. L. MASSART. UNEQ: A class modelling supervised pattern recognition technique. *Mikrochimica Acta*. 1986, **89**(1-6), 139-152. ISSN 0026-3672. Dostupné z: doi:10.1007/BF01207313

[10] HEARST, M.A., S.T. DUMAIS, E. OSUNA, J. PLATT a B. SCHOLKOPF. Support vector machines. *IEEE Intelligent Systems and their Applications*. 1998, **13**(4), 18-28. ISSN 1094-7167. Dostupné z: doi:10.1109/5254.708428

[11] PANG, Guansong, Chunhua SHEN, Longbing CAO a Anton Van Den HENGEL. Deep Learning for Anomaly Detection. *ACM Computing Surveys* [online]. 2022, **54**(2), 1-38 [cit. 2022-11-14]. ISSN 0360-0300. Dostupné z: doi:10.1145/3439950

[12] MARTINEZ, A.M. a A.C. KAK. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **23**(2), 228-233. ISSN 01628828. Dostupné z: doi:10.1109/34.908974

[13] VEMPALA, Santosh S. *The random projection method*. Providence, R.I.: American Mathematical Society, c2004. ISBN 08-218-2018-4.

[14] VERCRUYSSEN, Vincent, Wannes MEERT a Jesse DAVIS. Transfer learning for time series anomaly detection. *IAL ECML PKDD*. 2017, **2017**, 27. Dostupné také z: https://lirias.kuleuven.be/1464239?limo=0

[15] NOTO, Keith, Carla BRODLEY a Donna SLONIM. FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection. *Data Mining and Knowledge Discovery*. 2012, **25**(1), 109-133. ISSN 1384-5810. Dostupné z: doi:10.1007/s10618-011-0234-x

[16] TAX, David M.J. a Robert P.W. DUIN. Support Vector Data Description. *Machine Learning* [online]. 2004, **54**(1), 45-66 [cit. 2022-11-14]. ISSN 0885-6125. Dostupné z: doi:10.1023/B:MACH.0000008084.60811.49

[17] SABOKROU, Mohammad, Mohammad KHALOOEI, Mahmood FATHY a Ehsan ADELI. *Adversarially Learned One-Class Classifier for Novelty Detection* [online]. 24 May 2018 [cit. 2022-11-14]. Dostupné z: https://arxiv.org/abs/1802.09088

[18] DEECKE, Lucas, Robert VANDERMEULEN, Lukas RUFF, Stephan MANDT a Marius KLOFT. Image Anomaly Detection with Generative Adversarial Networks. *Machine Learning and Knowledge Discovery in Databases* [online]. Cham: Springer International Publishing, 2019, 2019-01-18, 3-17 [cit. 2022-11-14]. Lecture Notes in Computer Science. ISBN 978-3-030-10924-0. Dostupné z: doi:10.1007/978-3-030-10925-7_1

[19] AMARBAYASGALAN, Tsatsral, Bilguun JARGALSAIKHAN a Keun Ho RYU. Unsupervised Novelty Detection Using Deep Autoencoders with Density Based Clustering. *Applied Sciences* [online]. 27 August 2018, **2018**(8) [cit. 2022-11-14]. Dostupné z: https://www.mdpi.com/2076-3417/8/9/1468

[20] SAKURADA, Mayu a Takehisa YAIRI. *Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction* [online]. 02 December 2014, 4-11 [cit. 2022-11-14]. Dostupné z: https://dl.acm.org/doi/10.1145/2689746.2689747

[21] JINWON, An a Sungzoon CHO. *Variational Autoencoder based Anomaly Detection using Reconstruction Probability* [online]. 2015 [cit. 2022-11-14]. Dostupné z: https://www.semanticscholar.org/paper/Variational-Autoencoder-based-Anomaly-Detection-An-Cho/061146b1d7938d7a8dae70e3531a00fceb3c78e8

[22] WANG, Lu, Dongkai ZHANG, Jiahao GUO a Yuexing HAN. Image Anomaly Detection Using Normal Data Only by Latent Space Resampling. *Applied Sciences* [online]. 2020, **10**(23) [cit. 2022-11-14]. ISSN 2076-3417. Dostupné z: doi:10.3390/app10238660

[23] MAKHZANI, Alireza a Brendan FREY. *Winner-Take-All Autoencoders* [online]. 7 Jun 2015 [cit. 2022-11-14]. Dostupné z: https://arxiv.org/abs/1409.2752

[24] GULRAJANI, Ishaan, Ahmed FARUK, Martin ARJOVSKY, Vincent DUMOULIN a Aaron COURVILLE. *Improved Training of Wasserstein GANs* [online]. 25 Dec 2017 [cit. 2022-11-14]. Dostupné z: https://arxiv.org/abs/1704.00028

[25] GOODFELLOW, Ian J., Jean POUGET-ABADIE, Mehdi MIRZA, Bing XU, David WARDE-FARLEY, Sherjil OZAIR, Aaron COURVILLE a Yoshua BENGIO. *Generative Adversarial Networks* [online]. 10 Jun 2014 [cit. 2022-11-14]. Dostupné z: https://arxiv.org/abs/1406.2661

[26] XIA, Yan, Xudong CAO, Fang WEN, Gang HUA a Jian SUN. Learning Discriminative Reconstructions for Unsupervised Outlier Removal. *2015*

IEEE International Conference on Computer Vision (ICCV) [online]. IEEE, 2015, 2015, 1511-1519 [cit. 2022-11-14]. ISBN 978-1-4673-8391-2. Dostupné z: doi:10.1109/ICCV.2015.177

[27] SCHNEIDER, Sarah, Doris ANTENSTEINER, Daniel SOUKUP a Matthias SCHEUTZ. Autoencoders - A Comparative Analysis in the Realm of Anomaly Detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2022, s. 1986-1992.

[28] HORÁK, Karel. *Generative Adversarial Networks for Image Data Augmentation* [online]. In: . TU Wien, 2019 [cit. 2022-11-15]. Dostupné z: http://vision.uamt.feec.vutbr.cz/STU/lectures/12_MachineLearning_GANs.pdf

[29] RADFORD, Alec, Luke METZ a Soumith CHINTALA. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *ArXiv* [online]. 7 Jan 2016, **2015** [cit. 2022-11-14]. Dostupné z: https://arxiv.org/abs/1511.06434

[30] LIPTON, Zachary C. a Subarna TRIPATHI. Precise Recovery of Latent Vectors from Generative Adversarial Networks. *CoRR*. **2017**(abs/1702.04782). Dostupné také z: http://arxiv.org/abs/1702.04782

[31] RUFF, Lukas, Robert VANDERMEULEN, Nico GOERNITZ, Lucas DEECKE, Shoaib Ahmed SIDDIQUI, Alexander BINDER, Emmanuel MÜLLER a Marius KLOFT. Deep One-Class Classification. *Proceedings of the 35th International Conference on Machine Learning* [online]. PMLR, 15 Jul 2018, **2018**, 4393-4402 [cit. 2022-11-14].

[32] VASANI, Dipam. *This thing called Weight Decay* [online]. In: . Apr 29, 2019 [cit. 2023-01-02]. Dostupné z: https://towardsdatascience.com/this-thing-called-weight-decay-a7cd4bcfccab

[33] LECUN, Y., L. BOTTOU, Y. BENGIO a P. HAFFNER. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. **86**(11), 2278-2324. ISSN 00189219. Dostupné z: doi:10.1109/5.726791

[34] LIN, Dongyun, Yiqun LI, Shudong XIE, Tin Lay NWE a Sheng DONG. DDR-ID: dual deep reconstruction networks based image decomposition for anomaly detection. *Journal of Ambient Intelligence and Humanized Computing* [online]. [cit. 2022-11-14]. ISSN 1868-5137. Dostupné z: doi:10.1007/s12652-021-03425-0

[35] HONZÍK, Petr. *Strojové učení*. Brno, 2006. Skripta. Vysoké učení technické v Brně.

[36] *Push-Pull Connectors: HR10 Series* [online]. In: . HRS [cit. 2023-05-09]. Dostupné z: https://www.tme.eu/Document/5660e782a17ce361c8c2c240284dedea/HR10-hirose.pdf

[37] *Standa Opto-mechanics: 1HB - Honeycomb Optical Breadboards* [online]. [cit. 2023-05-09]. Dostupné z: https://www.standa.lt/products/catalog/optical_tables?item=139

[38] Edmund Optics: English Stainless Steel Mounting Posts. *Edmund Optics* [online]. 2020 [cit. 2023-05-09]. Dostupné z: https://www.edmundoptics.com/f/english-stainless-steel-mounting-posts/13408/#

[39] KU-5050AD-60D-xW: Datasheet. In: *GM Electronic* [online]. Česká republika [cit. 2023-05-10]. Dostupné z: https://img.gme.cz/files/eshop_data/eshop_data/10/960-509/dsh.960-509.1.pdf

[40] LUKASZCZYK, Jakub. *Návrh programu pro obsluhu kamer a provádění strojového učení*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2021, 65 s. Bakalářská práce. Vedoucí práce: Ing. Šimon Bilík

[41] CAREY, Eric, Stephane MAURICE, Vincent ROWLEY, Thies MÖLLER a Karsten I. CHRISTENSEN. GenICam Standard Features Naming Convention. In: *EMVA: European Machine Vision Association* [online]. 2019-05-27 [ver. 2.5] [cit. 2022-11-28]. Dostupné z: https://www.emva.org/wp-content/uploads/GenICam_SFNC_v2_6.pdf

[42] KRÖGER, Niklas, fklostermann a Carola SCHOENROCK. Vimba Python. *GitHub* [online]. [cit. 2023-05-07]. Dostupné z: https://github.com/alliedvision/VimbaPython

[43] LF PROJECTS, LLC. *PyTorch* [online]. [cit. 2023-05-07]. Dostupné z: https://pytorch.org/

[44] ALLIED VISION TECHNOLOGIES CANADA INC. *Technical Manual: GC1290 / GC1290C*. August 27, 2010. Dostupné také z:

http://www.altavision.com.br/Arquivos/AVT/Manuals/GC1290_User_Manual.pdf

[45] PROSILICA INC. *USER MANUAL: GE650 GE650C*. February 21, 2007. Dostupné také z: https://pyramidimaging.com/specs/Prosilica/700014A%20-%20GE650%20User%20Manual.pdf

[46] *FL-HC0416X-VG Product specifications* [online]. [cit. 2023-01-03]. Dostupné z: https://cdn.alliedvision.com/fileadmin/content/documents/products/accessories/lenses/Ricoh_Pentax/Data_sheet/FL-HC0416X-VG.pdf

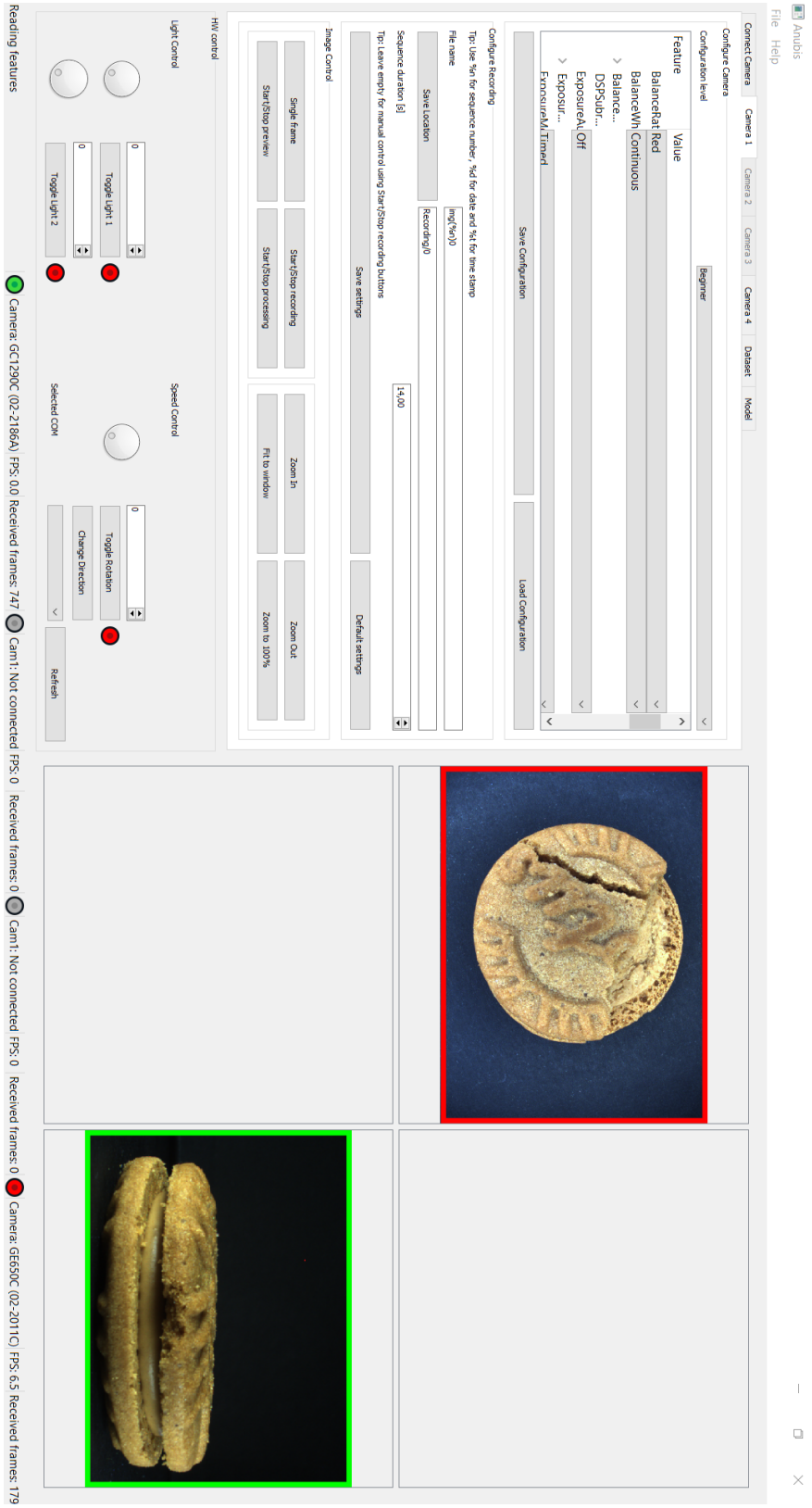
[47] *HLM5V50F13 1/3" Format Manual Iris Lens, 5 mm — 50 mm, F1.3* [online]. [cit. 2023-01-03]. Dostupné z: https://www.security.honeywell.com/-/media/SecurityME/Resources/ProductDocumentsME/9000363rev101web-pdf.pdf

[48] G, Siji George C a B.Sumathi -. Grid Search Tuning of Hyperparameters in Random Forest Classifier for Customer Feedback Sentiment Prediction. *International Journal of Advanced Computer Science and Applications*. 2020, **11**(9). ISSN 21565570. Dostupné z: doi:10.14569/IJACSA.2020.0110920

[49] BERGSTRA, James a Yoshua BENGIO. Random Search For Hyper-parameter Optimization. *Journal of Machine Learning Research*. 2012, **2012**(13), 281-305.

Seznam příloh

A Anubis - upravené uživatelské rozhraní

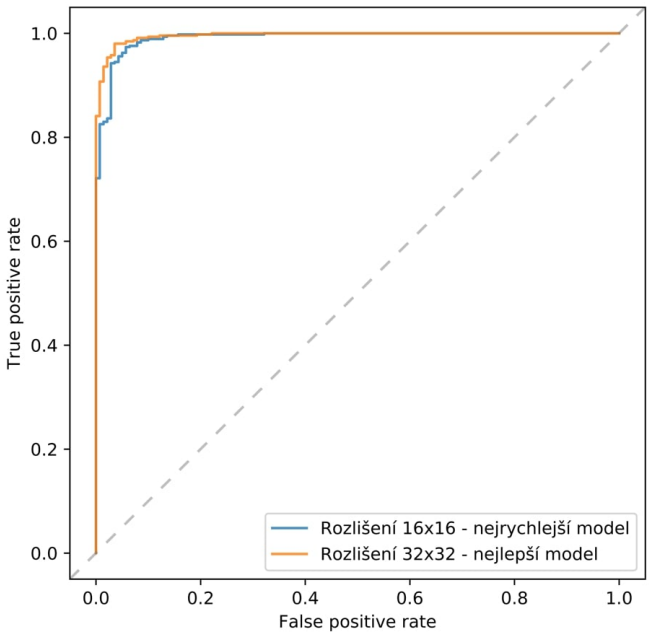


B Experimentální část práce - pohled zboku

B.1 Volba hyperparametrů

Tab. B.1: Nastavení hyperparametrů dvou vybraných reprezentativních modelů pro detekci anomálií při pohledu zboku (*základní počet filtrů v první konvoluční vrstvě je 16. Koeficient znamená, jakým číslem byla tato hodnota vynásobena. V dalších vrstvách je pak vždy počet filtrů dvojnásobný oproti předchozí vrstvě)

Podmínka výběru	Nejrychlejší	Nejpřesnější
Parametry sítě		
Název sítě	lotus_LeNet_single	lotus_2conv
Počet konvolučních vrstev	3	2
Rozlišení na vstupu [px]	16x16	32x32
Velikost filtru	5	5
Koeficient filtrů*	2	2
Parametry učení		
Extrahované příznaky	128	256
Optimizér	Adam	Adam
Learning rate	0,0001	0,0001
Počet epoch	100	100
Epochy pro dělení LR	25; 50; 75	25; 50; 75
Velikost dávky	10	10
Rozpad vah	1^{-6}	1^{-5}
Parametry předtrénování		
Optimizér	Adam	Adam
Learning rate	0,001	0,001
Počet epoch	100	100
Epochy pro dělení LR	-	-
Velikost dávky	100	100
Rozpad vah	1^{-5}	1^{-5}

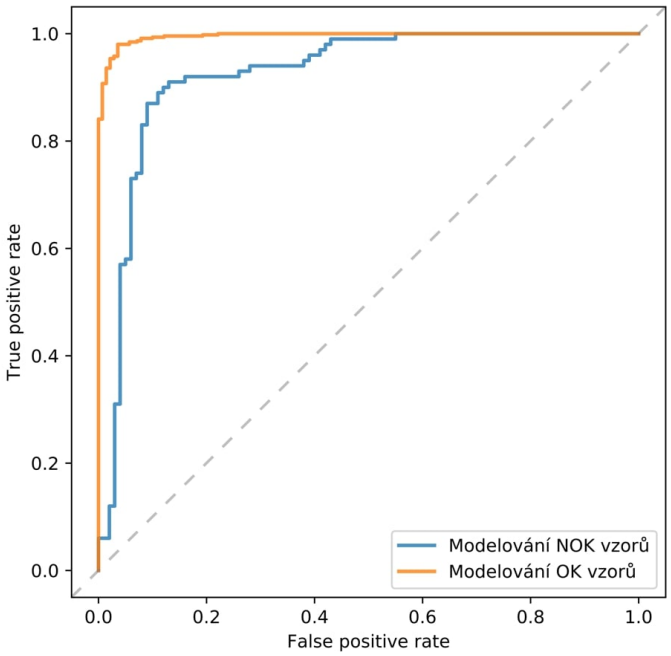


Obr. B.1: Srovnání ROC křivek zvolených konfigurací hyperparametrů (pohled zboku)

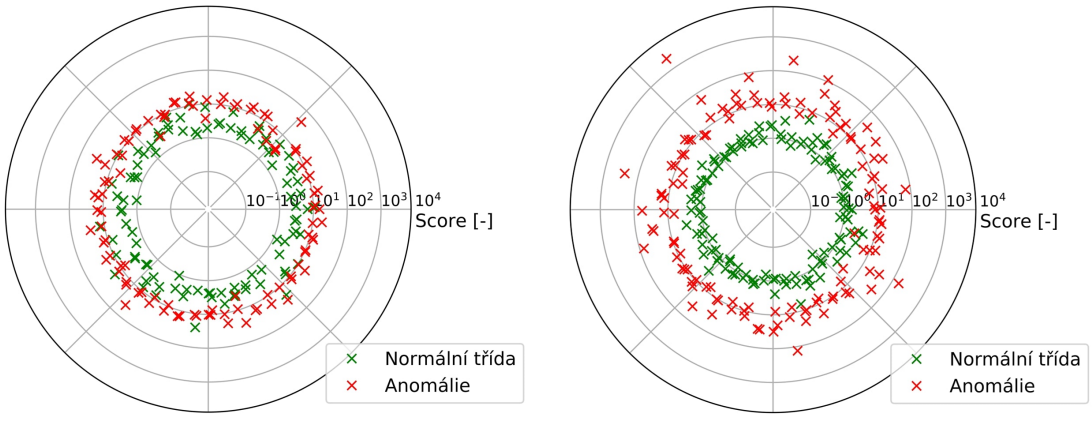
Tab. B.2: Numerické srovnání zvolených konfigurací hyperparametrů - pohled shora (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Model	**Nejrychlejší**	**Nejpřesnější**
AUC [%]	96,977	99,565
TPR [%]	90,266	90,708
FPR [%]	3,5714	0,7143
ACC [%]	91,723	92,737

B.2 Nulová hypotéza



Obr. B.2: Modelování NOK vzorů X modelování OK vzorů (pohled zboku, rozlišení 32x32)



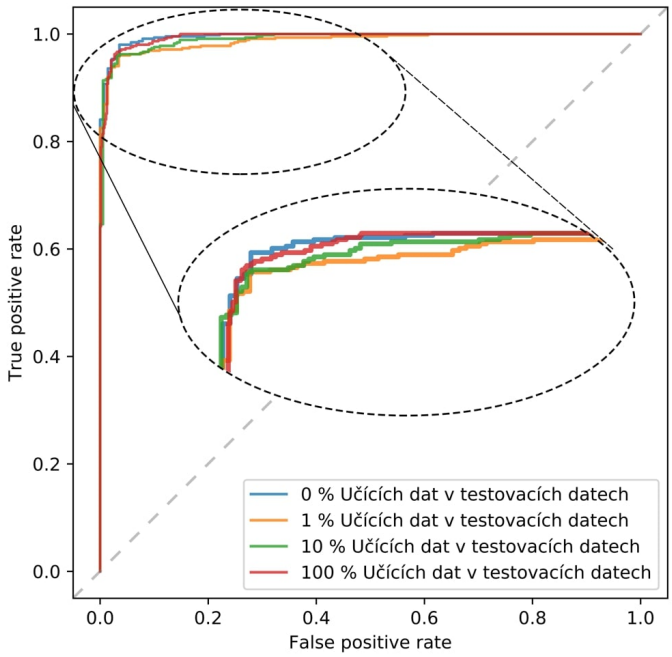
a) b)

Obr. B.3: Porovnání vzdálenosti mapování prvků normální třídy a anomálií od středu hyperkoule - pohled zboku. a) model modelující OK vzory, b) model modelující NOK vzory (výstupem modelu je pouze vzdálenost od středu, poloha na kružnici je pouze pro lepší vizualizaci výsledků)

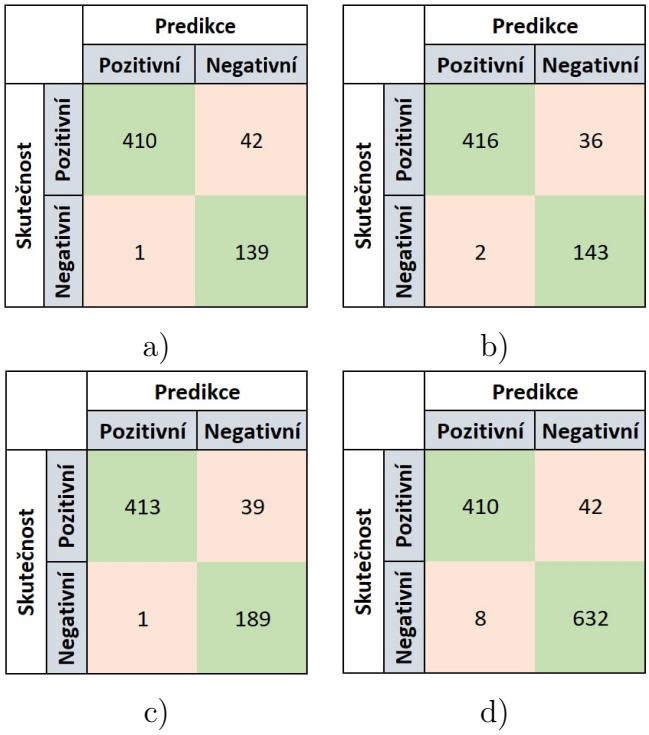
B.3 Resubstituční chyba

Tab. B.3: Numerické srovnání modelů s různou úrovní resubstituce - pohled zboku (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Úroveň resubstituce	**0 %**	**1 %**	**10 %**	**100 %**
AUC [%]	99,565	90,708	0,714	92,736
TPR [%]	98,828	92,035	1,379	93,635
FPR [%]	99,139	91,372	0,526	93,769
ACC [%]	99,442	90,708	1,250	95,421



Obr. B.4: Srovnání vlivu resubstituce (pohled zboku, rozlišení 32x32)

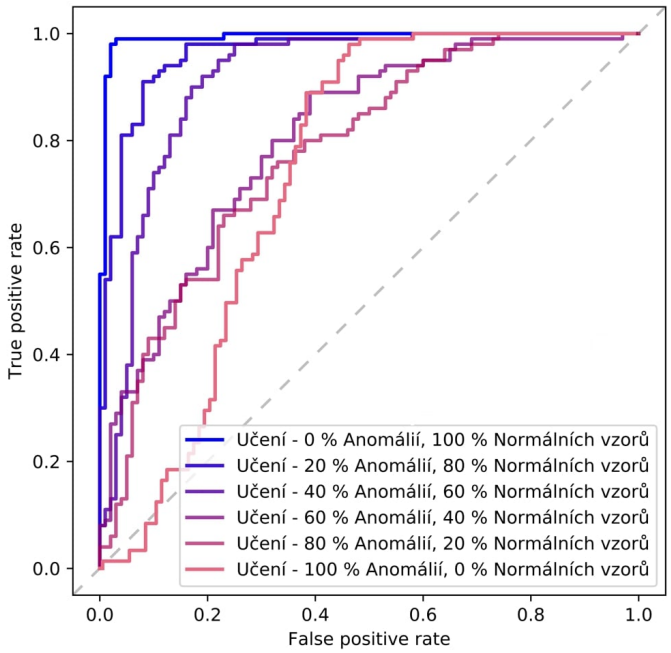


Obr. B.5: Matice záměn pro modely s různou úrovní resubstituce. Úroveň resubstituce: a) 0%, b) 1%, c) 10%, d) 100%

B.4 Vliv nesprávné anotace

Tab. B.4: Srovnání výkonnostních metrik pro modely naučené na různý poměr nesprávně anotovaných dat - Pohled zboku, rozlišení 32x32 (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

Učících NOK vzorů [%]	**0**	**20**	**40**	**60**	**80**	**100**
Učících OK vzorů [%]	**100**	**80**	**60**	**40**	**20**	**0**
AUC [%]	99,25	96,25	91,45	80,68	77,82	74,48
TPR [%]	92	91	90	92	91	91
FPR [%]	1	8	17	48	55	58
ACC [%]	95,5	91,5	86,5	72,0	68,0	66,5

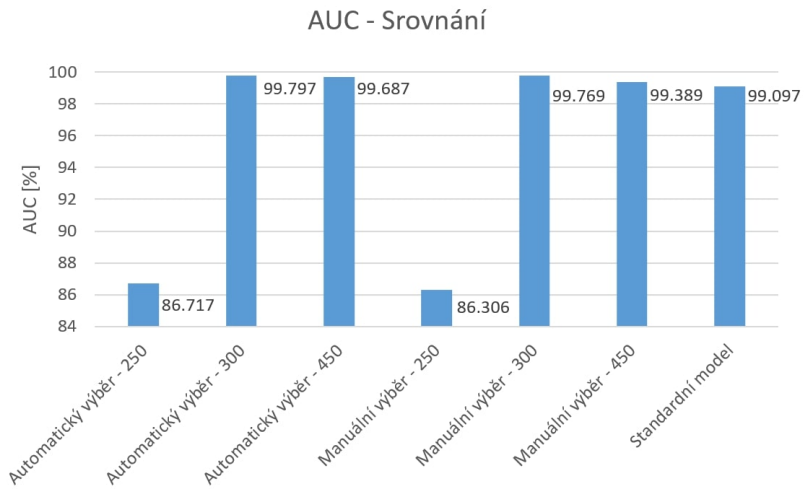


Obr. B.6: Srovnání ROC křivek pro modely naučené na různý poměr nesprávně anotovaných dat (pohled zboku, rozlišení 32x32)

B.5 Citlivostní analýza

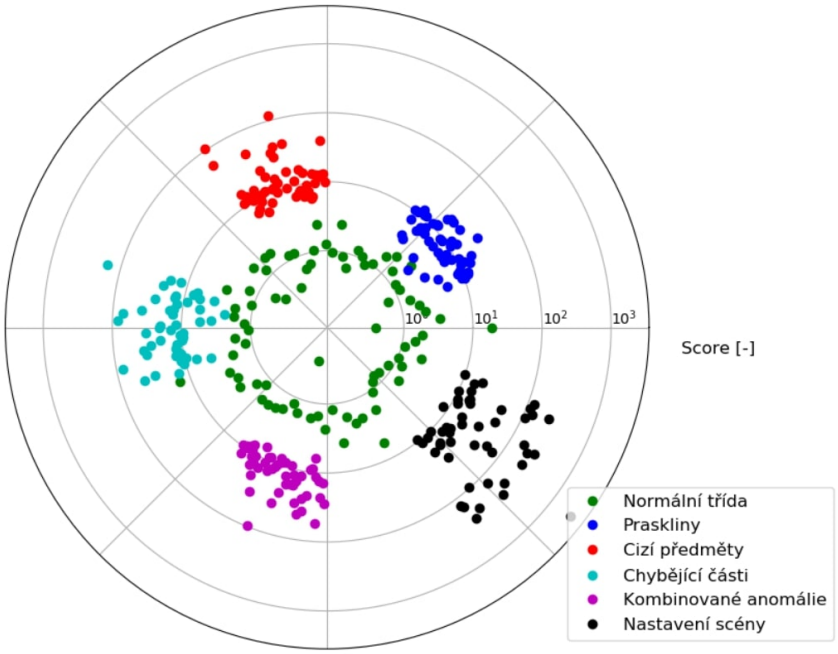
Tab. B.5: Srovnání výkonnostních metrik pro modely naučené na perfektní představitele OK vzorů - pohled zboku (vyjma hodnoty AUC jsou metriky vypočteny pro nastavení prahové vzdálenosti modelu ve vzdálenosti nejbližší stavu, kdy byl schopen správně detekovat 90 % testovacích anomálií)

	Automatický výběr			**Manuální výběr**		
Počet perfektních reprezentantů	**250**	**300**	**450**	**250**	**300**	**450**
AUC [%]	51,49	52,326	48,214	51,099	67,388	75,214
TPR [%]	92,118	92,697	91,262	90,148	90,169	92,718
FPR [%]	16	8	86	86	74	58
ACC [%]	91,228	92,611	76,172	81,798	82,266	82,813



Obr. B.7: Srovnání jednotlivých modelů naučených na perfektních představitelích OK vzorů při jejich testování na *Standardním datasetu* - boční pohled (pro zvýšení přehlednosti je osa y omezená na rozsah 90-100 %)

B.6 Vliv typu testovacích anomálií



Obr. B.8: Srovnání mapovací vzdálenosti jednotlivých typů anomálií (výstupem modelu je pouze vzdálenost od středu. Poloha na kružnici je modifikována pouze z důvodu lepší vizualizace výsledků)

C Obsah elektronické přílohy

- kořenový adresář přiloženého DVD č.1
  - Anubis ..... Kořenový adresář aplikace Anubis
    - Camera_config ..... Adresář obsahující konfigurační soubory kamer
    - icons
    - src ..... Obsahuje zdrojové soubory aplikace
    - svdd_anubis ..... Obsahuje minimální implementaci metody pro vyhodnocování
      - datasets ..... Obsahuje implementace datasetů
      - models ..... Obsahuje předučené modely pro použití v aplikaci
      - networks ..... Obsahuje implementace sítí pro extrakci příznaků
      - optim ..... Obsahuje implementaci optimalizačního pravidla
      - utils ..... Obsahuje pomocné skripty
    - anubis.py ..... Soubor pro spuštění aplikace
    - requirements.txt ..... Soubor pro instalaci požadovaných balíčků
  - DeepSVDD ..... Kořenový adresář implementace metody
    - data ..... Obsahuje zkompilované datasety
      - dataset_lotus_standard ..... Ukázkový zkompilovaný dataset
    - log ..... Složka pro ukládání výstupů učení
    - src ..... Obsahuje implementaci metody pro vyhodnocování a učení
      - base ..... Obsahuje prototypové implementace tříd
      - datasets ..... Obsahuje implementace datasetů
      - networks ..... Obsahuje implementace sítí pro extrakci příznaků
      - optim ..... Obsahuje implementaci optimalizačního pravidla
      - utils ..... Obsahuje pomocné skripty
  - Podklady k experimentům
    - Konfigurace modelů a jejich výstupy pro každý z experimentů (číslováno jako
    - ..... v textu práce - matice záměn, ROC křivka, vizualizace špatných klasifikací...)
- Kořenový adresář přiloženého DVD č.2
  - Dataset
    - OK ..... Realizace OK vzorů
      - Side, Top ..... Adresáře s pohledem zboku a pohledem shora
    - NOK
      - Cracks ..... Vady typu praskliny
        - Side, Top
      - Foreign_items ..... Vady typu cizí předměty
        - Side, Top
      - Missing_parts ..... Vady typu chybějící části
        - Side, Top
      - Mixed ..... Vady typu kombinované
        - Side, Top
      - Scene ..... Vady typu nastavení scény
        - Side, Top
      - Side_only_anomalies ..... Vady typu jednostranné zboku
        - Side, Top
      - Top_only_anomalies ..... Vady typu jednostranné shora