Detekcia anomálií a ich lokalizácia v obraze

Bc. Andrej Ovečka

Diplomová práce
2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Andrej Ovečka**
Osobní číslo:	**A20145**
Studijní program:	**N0613A140022 Informační technologie**
Specializace:	**Softwarové inženýrství**
Forma studia:	**Kombinovaná**
Téma práce:	**Detekce anomálií a jejich lokalizace v obraze**
Téma práce anglicky:	**Anomaly Detection and Localization in Images**

Zásady pro vypracování

1. Vypracujte literární rešerši zabývající se modely pro detekci a lokalizaci anomálií v obraze.
2. Připravte vhodný dataset pro detekci a lokalizaci anomálií.
3. Natrénujte vybrané modely konvolučních sítí z literární rešerše na připravených datech.
4. Proveďte kvantitativní a kvalitativní srovnání výsledků realizovaných modelů.
5. Zhodnoťte dosažené výsledky.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. DEFARD, Thomas, et al. Padim: a patch distribution modeling framework for anomaly detection and localization. In: International Conference on Pattern Recognition. Springer, Cham, 2021. p. 475-489.
2. GOODFELLOW, Ian, BENGIO, Yoshua a Aaron COURVILLE. Machine learning basics. Deep learning. MIT Press. 2016. ISBN 9780262035613.
3. RAMSUNDAR, Bharath a Reza Bosagh ZADEH. TensorFlow for deep learning: from linear regression to reinforcement learning. Beijing: O'Reilly Media. 2018. ISBN 9781491980422.
4. ROSEBROCK, Adrian. Starter Bundle. In: Deep Learning for Computer Vision with Python. PyimageSearch.com. 2017.
5. ROSEBROCK, Adrian. Practitioner Bundle. In: Deep Learning for Computer Vision with Python. PyimageSearch.com. 2017.
6. RASCHKA, Sebastian a Vahid MIRJALILI. Python machine learning: machine learning and deep learning with Python, scikit-learn, and TensorFlow . Second edition. Birmingham: Packt, 2017, xviii, 595 s. ISBN 978-1-78712-593-3.
7. CHOLLET, François. Deep learning v jazyku Python: knihovny Keras, Tensorflow . Praha: Grada Publishing, 2019, 328 s. Knihovna programátora. ISBN 978-80-247-3100-1.

Vedoucí diplomové práce: **doc. Ing. Zuzana Komínková Oplatková, Ph.D.**
Ústav informatiky a umělé inteligence

Konzultant diplomové práce: **MSc. Sina Mirshahi**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **2. prosince 2022**
Termín odevzdání diplomové práce: **26. května 2023**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 7. prosince 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 24.5.2023

Andrej Ovečka, v.r.
podpis studenta

ABSTRAKT

Táto diplomová práca sa zaoberá detekciou anomálií a ich lokalizáciou v obraze. Cieľom je popísať modely, ktoré sa touto problematikou zaoberajú a následne tieto modely natrénovať a zhodnotiť. Práca je rozdelená na dve časti, teoretickú a praktickú. V teoretickej časti sú popísané jednotlivé modely detekcie a lokalizácie anomálií. Praktická časť práce sa venuje trénovaniu modelov na pripravenom datasete. Záver práce je zameraný na zhodnotenie dosiahnutých poznatkov.

Kľúčové slová: spracovanie obrazu, detekcia anomálií v obraze, lokalizácia anomálií v obraze, strojové učenie, konvolučné neurónové siete, PaDiM, SimpleNet, RDOCE, PatchCore, RIAD, DRÆM, SPADE

ABSTRACT

This Master's thesis on anomaly detection and localization in images. The aim is to describe models that deal with this issue and subsequently train and evaluate these models. The thesis is divided into two parts, theoretical and practical. The theoretical part describes the individual models of anomaly detection and localization. The practical part of the thesis is devoted to training models on a prepared dataset. The conclusion of the thesis focuses on the evaluation of the acquired knowledge.

Keywords: image processing, image anomaly detection, image anomaly localization, machine learning, convolutional neural networks, PaDiM, SimpleNet, RDOCE, PatchCore, RIAD, DRÆM, SPADE

Ďakujem vedúcej diplomovej práce váženej pani doc. Ing. Zuzane Komínkovej Oplatkovej Ph.D. za jej cenné poznatky, rady a pripomienky, ktorými mi bola nápomocná pri tvorbe tejto práce. Taktiež by som chcel poďakovať svojej rodine a manželke, za ich neustálu podporu a motiváciu počas celého štúdia.

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

V dnešnej dobe sa z obrazovej analýzy a spracovania stáva stále dôležitejší nástroj v mnohých oblastiach. Jedným z nástrojov obrazovej analýzy je aj detekcia a lokalizácia anomálií v obraze.

Anomálie sú nezvyčajné, neobvyklé alebo výnimočné javy, ktoré sa vyskytujú v porovnaní s normálnym, očakávaným správaním alebo vzorom. Môžu zahŕňať nezvyčajné tvary, farby, textúry alebo vzory v objektoch alebo oblastiach. Tieto anomálie môžu byť príznakom poškodenia, choroby alebo iných zmien v objektoch, ale môžu sa tiež vyskytovať v dôsledku chýb v obrazovom zázname.

Táto práca je zameraná na problematiku detekcie anomálií a ich lokalizácie v obraze. Detekcia anomálií v obraze sa zvyčajne definuje ako identifikácia objektov alebo oblastí v obraze, ktoré sa výrazne líšia od normálneho stavu alebo od priemeru v tréningových dátach. Lokalizácia anomálií v obraze sa zase zaoberá identifikáciou miesta alebo oblasti v obraze, kde sa anomália nachádza.

Cieľom detekcie a lokalizácie anomálií v obraze je zlepšiť kvalitu spracovania a zabezpečiť presnejšie výsledky vo viacerých aplikáciách, ako napríklad v oblastiach bezpečnosti, medicíny, priemyselnej výroby a mnohých ďalších.

Existuje niekoľko prístupov k detekcii a lokalizácii anomálií v obraze, vrátane metód založených na rekonštrukcii, metód založených na embeddingu alebo metód založených na zarovnaní. Každý prístup má svoje silné a slabé stránky a voľba metódy závisí od konkrétnej aplikácie a požiadaviek.

Táto diplomová práca sa zameriava na popis a porovnanie metód pre detekciu anomálií v obraze a ich následnú lokalizáciu. Práca je rozdelená na dve časti - teoretickú a praktickú. Teoretická časť sa zaoberá rešeršou existujúcich metód pre detekciu a lokalizáciu anomálií v obraze. Konkrétne popisujem metódy založené na rekonštrukcii, zarovnaní a embeddingu. V rámci metód založených na embeddingu sú popísané modely PaDiM, SimpleNet, RDOCE a PatchCore. V kapitole o metódach založených na rekonštrukcii sú zase popísané modely RIAD a DRÆM. Posledná kapitola teoretickej časti je zameraná na metódy založené na zarovnaní, v rámci ktorých je opísaný model SPADE.

Tieto modely boli vybrané na základe ich výkonnosti a presnosti v detekcii anomálií. Modely boli podrobené testovaniu a porovnaniu na datasete MVTec AD v rámci viacerých

literárnych prác, ktoré sú použité v teoretickej časti. MVTec AD je dataset navrhnutý na benchmarking metód detekcie a lokalizácie anomálií. Obsahuje viac ako 5000 obrázkov, ktoré sú rozdelené do pätnástich rôznych kategórií.

Prvá kapitola praktickej časti sa venuje príprave datasetu, ktorý je použitý pri trénovaní modelov. V ďalšej kapitole sú natrénované vybrané modely neurónových sietí na pripravenom datasete. Posledná kapitola praktickej časti je venovaná zhrnutiu dosiahnutých výsledkov.

I. TEORETICKÁ ČASŤ

1 METÓDY ZALOŽENÉ NA EMBEDDINGU

Metódy založené na embeddingu sú metódy strojového učenia, ktoré využívajú vektory na reprezentáciu vstupných dát. Tieto vektory sa nazývajú embeddingy a sú to relatívne nízkorozmerné priestory, do ktorých sa dajú preložiť vysokorozmerné vektory. Embeddingy uľahčujú strojové učenie na veľkých vstupných dátach, ako sú napríklad riedke vektory reprezentujúce slová [1].

Pri detekcii a lokalizácii anomálií v obraze sa embedding-based metódy vkladajú normálne príznaky do komprimovaného priestoru. Anomálne príznaky sú v embedding priestore vzdialené od normálnych clusterov. Typické metódy používajú na extrakciu príznakov vopred natrénované konvolučné neurónové siete (CNN) na datasete ImageNet [4]. Príkladom takejto siete môže byť reziduálna neurónová sieť (ResNet). ResNet je umelá sieť, ktorá zaviedla takzvané „skrátené spojenie identity“, ktoré umožňuje modelu preskočiť jednu alebo viacero vrstiev. Tento prístup umožňuje trénovanie modelu na tisíckach vrstiev bez ovplyvnenia výkonu [2].

Nasledujúce podkapitoly sa zaoberajú aktuálne používanými modelmi, ako sú PaDiM, PatchCore, RDOCE a SimpleNet.

1.1 PaDiM

Patch Distribution Modeling Framework for Anomaly Detection and Localization (PaDiM) je model detekcie a lokalizácie anomálií, ktorý využíva vopred natrénovanú konvolučnú neurónovú sieť na extrakciu embeddingu a má nasledujúce vlastnosti:

- každá pozícia patchu je popísaná multivariačnou Gaussovou distribúciou,
- zohľadňuje korelácie medzi rôznymi sémantickými úrovňami vopred natrénovanej konvolučnej neorónovej siete.

S týmto efektívnym prístupom má PaDiM počas testovania nízku časovú a priestorovú zložitosť, nezávislú od veľkosti tréningového datasetu, čo je výhodou pre priemyselné aplikácie [4].

1.1.1 Extrakcia embeddingu

Vopred natrénované CNN sú schopné výstupu relevantných vlastností pre detekciu anomálií. Preto sa táto metóda vyhýba zbytočnému optimalizovaniu neurónových sietí tým, že používa iba vopred natrénovanú CNN na generovanie vektora vloženia patchov. Počas trénovania je

každý patch normálnych obrázkov spojený so svojimi priestorovo zodpovedajúcimi aktivačnými vektormi na vopred natrénovaných aktivačných mapách CNN. Aktivačné vektory z rôznych vrstiev sa potom spájajú do embedding vektorov, ktoré nesú informácie z rôznych sémantických úrovní a rozlíšení, aby sa zakódovali jemné a globálne kontexty. Keďže aktivačné mapy majú nižšie rozlíšenie ako vstupný obrázok, mnoho pixelov má rovnaké embeddingy, a potom vytvárajú patch embeddingy bez prekrytia v pôvodnom rozlíšení obrázka. Preto sa vstupný obrázok môže rozdeliť na mriežku $(i, j) \in [1, W] \times [1, H]$, kde $W \times H$ je rozlíšenie najväčšej aktivačnej mapy použitej na generovanie embeddingov. Nakoniec každá pozícia patchu (i, j) v tejto mriežke je spojená s embedding vektorom x_{ij} vypočítaným tak, ako je opísané vyššie [4].

Generované vektory vloženia patchov môžu niesť redundantné informácie, preto sa zmenšuje ich veľkosť. Náhodný výber niekoľkých rozmerov je účinnejší, ako klasický algoritmus Principal Component Analysis (PCA) [3]. Táto jednoduchá redukcia dimenzionality výrazne znižuje zložitosť modelu pre trénovanie aj testovanie pri zachovaní výkonu na úrovni štandardu. Nakoniec sa patch embedding vektory z testovacích obrázkov s pomocou naučenej parametrickej reprezentácie normálnej triedy použité na vytvorenie mapy anomálií [4].

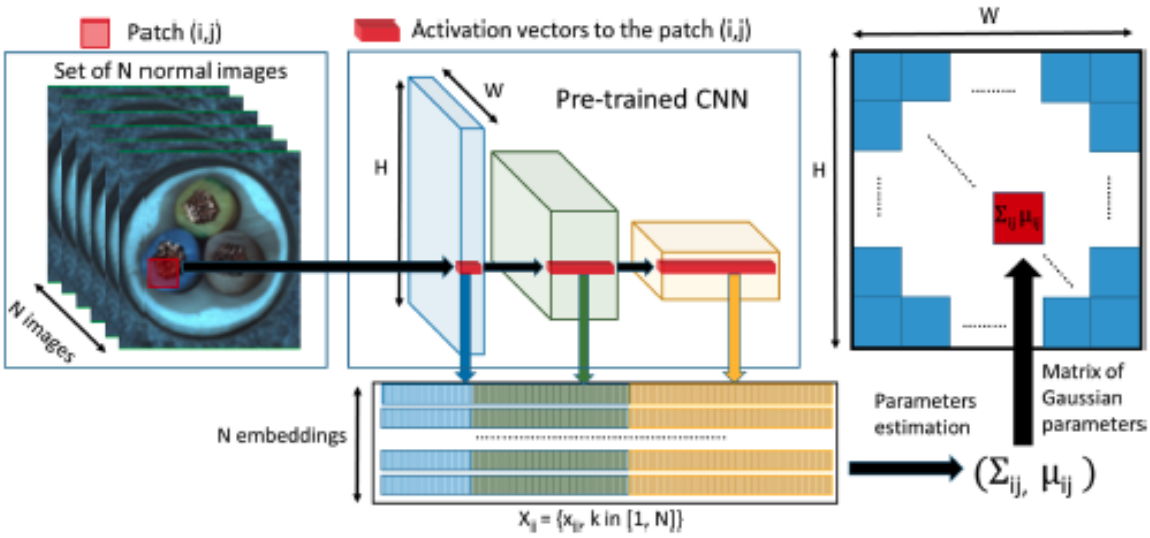
1.1.2 Učenie sa normalite

Na získanie charakteristík normálneho obrazu v pozícii (i,j) je najskôr nutné vypočítať množinu patch embedding vektorov na pozícii (i,j), $X_{ij} = \{x_{ij}^k, k \in [1, N]\}$ z N normálnych tréningových obrázkov. Pre zhrnutie informácií obsiahnutých v tejto množine sa predpokladá, že X_{ij} je generované viacrozmerným Gaussovým rozdelením $N(\mu_{ij}, \sum_{ij})$, kde μ_{ij} je vzorkový priemer X_{ij} a vzorková kovariancia $\sum_{ij}$ sa odhaduje nasledovne (1):

$$\sum_{ij} = \frac{1}{N-1} \sum_{k=1}^{N} (x_{ij}^k - \mu_{ij})(x_{ij}^k - \mu_{ij})^T + \epsilon I \tag{1}$$

kde regularizačný člen ϵI spôsobuje, že vzorkovacia kovariačná matica $\sum_{ij}$ je plnohodnotná a invertovateľná. Nakoniec je každá možná pozícia patchu spojená s viacrozmerným Gaussovým rozdelením, ako je znázornené na obrázku nižšie, maticou parametrov Gaussovho rozdelenia [4].

Patch embedding vektory nesú informácie z rôznych sémantických úrovní. Preto každé odhadnuté viacrozmerné Gaussovo rozdelenie $N(\mu_{ij},\ \sum_{ij})$ zachytáva informácie z rôznych úrovní a $\sum_{ij}$ obsahuje medziúrovňové korelácie [4].



Obrázok 1. Patch embedding proces [4]

Na obrázku 1 je zobrazený patch embedding proces. Pre každý patch obrázka zodpovedajúci pozícii (i, j) v najväčšej mape príznakov CNN sa PaDiM učí Gaussove parametre $(\mu_{ij},\ \sum_{ij})$ z množiny N trénovaných embedding vektorov $X_{ij} = \{x_{ij}^k, k \in [1, N]\}$, vypočítaných z N rôznych tréningových obrázkov a troch rôznych vopred natrénovaných vrstiev CNN.

1.1.3 Inferencia

Výpočet mapy anomálií používa Mahalanobisovu vzdialenosť $M(x_{ij})$ na priradenie skóre anomálie pre patch na pozícii (i,j) testovacieho obrázka. $M(x_{ij})$ môže byť interpretovaná ako vzdialenosť medzi testovacím patch embeddingom x_{ij} a naučenou distribúciou $N(\mu_{ij},\ \sum_{ij})$, kde $M(x_{ij})$ je vypočítaná nasledovne (2):

$$M(x_{ij}) = \sqrt{(x_{ij} - \mu_{ij})^T \sum_{ij}^{-1} (x_{ij} - \mu_{ij})} \tag{2}$$

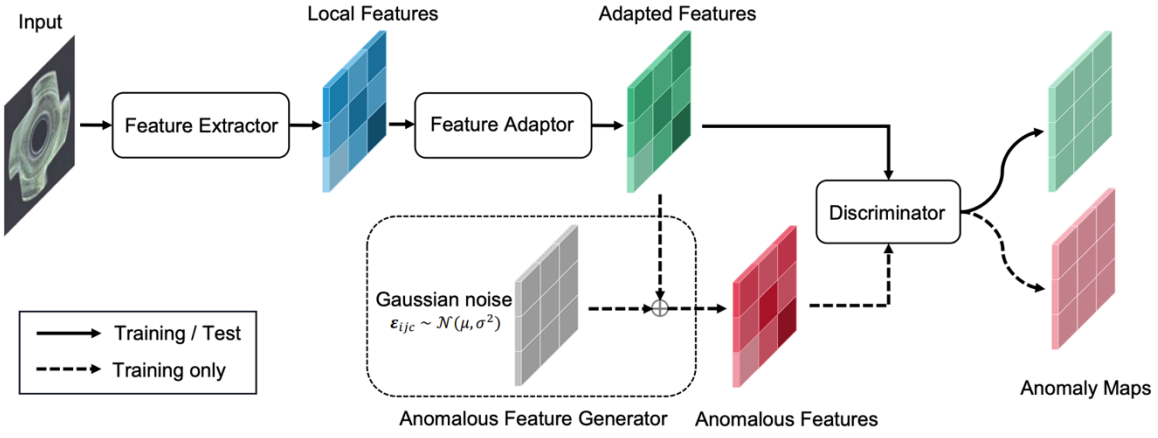
Preto môže byť vypočítaná matica Mahalanobisových vzdialeností $M = (M(x_{ij}))_{1<i<W,1<j<H}$, ktorá tvorí mapu anomálií. Vysoké skóre na tejto mape indikuje oblasti anomálie. Konečné skóre anomálie celého obrázka je maximum z mapy anomálií M. Pri testovaní metóda nemala problém so škálovateľnosťou metód založených na k-

najbližších susedoch (K-NN), preto nebolo potrebné vypočítavať a triediť veľké množstvo hodnôt vzdialeností na získanie skóre anomálie patchu [4].

1.2 SimpleNet

SimpleNet využíva spôsoby založené na embeddingu a syntéze a prináša niekoľko vylepšení. Namiesto priameho použitia vopred natrénovaných príznakov sa používa adaptér príznakov na vytvorenie cieľovo orientovaných príznakov, ktoré znižujú doménové skreslenie. Namiesto priamej syntézy anomálií na obrázkoch sa generujú anomálne príznaky tým, že sa na normálne príznaky v priestore príznakov aplikuje šum. S riadne kalibrovanou mierkou šumu možno získať úzko ohraničený normálny priestor príznakov. Postup detekcie anomálií sa zjednodušuje trénovaním jednoduchého diskriminátora, ktorý je výpočtovo oveľa efektívnejší, ako zložité štatistické algoritmy používané embedding metódami. SimplneNet využíva vopred natrénovanú základnú kostru pre extrakciu normálnych príznakov, nasledovanú adaptérom príznakov na prenos príznaku do cieľovej domény. Potom sa anomálne príznaky jednoducho generujú pridaním Gaussovského šumu k adaptovaným normálnym príznakom. Na týchto príznakoch sa trénuje jednoduchý diskriminátor pozostávajúci z niekoľkých vrstiev viacvrstvového perceptrónu na diskrimináciu anomálií [5].

SimpleNet je ľahko trénovateľný a aplikovateľný, s vynikajúcim výkonom a rýchlosťou vyhodnocovania. So široko používanou základnou kostrou WideResnet50 [24] dosahuje na datasete MVTec AD [19], 99,6 percentnú úspešnosť v metrike AUROC pri rýchlosti 77 snímok za sekundu, čím prekonáva predchádzajúce najlepšie publikované metódy detekcie anomálií z hľadiska presnosti a účinnosti [5].



Obrázok 2. Prehľad siete SimpleNet [5]

Ako je znázornené na obrázku 2, SimpleNet pozostáva z extraktora príznakov, adaptéra príznakov, generátora anomálnych príznakov a diskriminátora. Generátor anomálnych príznakov sa používa iba počas tréningu, takže SimpleNet sa pri inferencii riadi jednosmerným postupom [5].

1.2.1 Extraktor príznakov

Extraktor príznakov získava lokálne príznaky. Tento proces môže byť preformulovaný nasledovne. Tréningová množina a testovacia množina sú označené ako $\mathcal{X}_{train}$ a $\mathcal{X}_{test}$. Pre akýkoľvek obrázok $x_i \in \mathbb{R}^{H \times W \times 3}$ v $\mathcal{X}_{train} \cup \mathcal{X}_{test}$ extrahuje vopred natrénovaná sieť ϕ príznaky z rôznych hierarchií, ako sa to zvyčajne robí so základnou kostrou ResNet. Keďže vopred natrénovaná sieť je zaujatá voči datasetu, na ktorom bola trénovaná, je rozumné vybrať iba podmnožinu úrovní pre cieľový dataset. Formálne sa L definuje ako podmnožina obsahujúca indexy hierarchií na použitie. Mapa príznakov z úrovne $l \in L$ sa označuje ako $\phi^{l,i} \sim \phi^{l}(x_i) \in \mathbb{R}^{H_l \times W_l \times C_l}$, kde H_l, W_l a C_l sú výska, šírka a veľkosť kanála mapy príznakov. Pre vstup $\phi_{h,w}^{l,i} \in \mathbb{R}^{C_l}$ na pozícii (*h, w*) sa jej susedstvo s veľkosťou patchu *p* definuje ako (3):

$$N_p^{(h,w)} = \{(h', y') | h' \in [h - \lfloor p/2 \rfloor, \dots, h + \lfloor p/2 \rfloor], y' \in [w - \lfloor p/2 \rfloor, \dots, w + \lfloor p/2 \rfloor] \} \tag{3}$$

Agregáciou príznakov v susedstve $N_p^{h,w}$ s agregačnou funkciou f_{agg} sa získava lokálny príznak $z_{h,w}^{l,i}$ ako (4):

$$z_{h,w}^{l,i} = f_{agg}(\{\phi_{h',y'}^{l,i} | (h', y') \in \mathcal{N}_p^{h,w}\}) \tag{4}$$

Pre kombinovanie príznakov $z_{h,w}^{l,i}$ z rôznych hierarchií sa všetky mapy príznakov lineárne zmenšia na rovnakú veľkosť (H_0, W_0), teda veľkosť najväčšej z nich. Jednoduchým spojením máp príznakov podľa kanálov vznikne mapa príznakov $o^i \in \mathbb{R}^{H_0 \times W_0 \times C}$. Tento proces je definovaný ako (5):

$$o^i = f_{cat}(resize(z^{l',i}, (H_0 \times W_0)) | l' \in L) \tag{5}$$

Ako vstup o^i na pozícii (*h, w*) je definované $o_{h,w}^i \in \mathbb{R}^C$. Vyššie uvedené výrazy môžu byť zjednodušené nasledovne (6):

$$o^i = F_\phi(x^i) \tag{6}$$

kde F_ϕ je extraktor príznakov [5].

1.2.2 Adaptér príznakov

Na prenos tréningových príznakov do cieľovej domény sa využíva adaptér príznakov G_θ, pretože priemyselné obrázky majú obvykle iné rozdelenie, ako dataset použitý pri predtréningu základnej kostry. Adaptér príznakov G_θ prevádza lokálny príznak $o_{h,w}$ na adaptovaný príznak $q_{h,w}$ nasledovne (7):

$$q_{h,w}^{i} = G_\theta(o_{h,w}^{i}) \tag{7}$$

Adaptér príznakov môže byť zložený z jednoduchých neurónových blokov, ako napríklad plne prepojená vrstva alebo viacvrstvový perceptrón. Experimentálne bolo zistené, že jedna plne prepojená vrstva poskytuje dostatočný výkon [5].

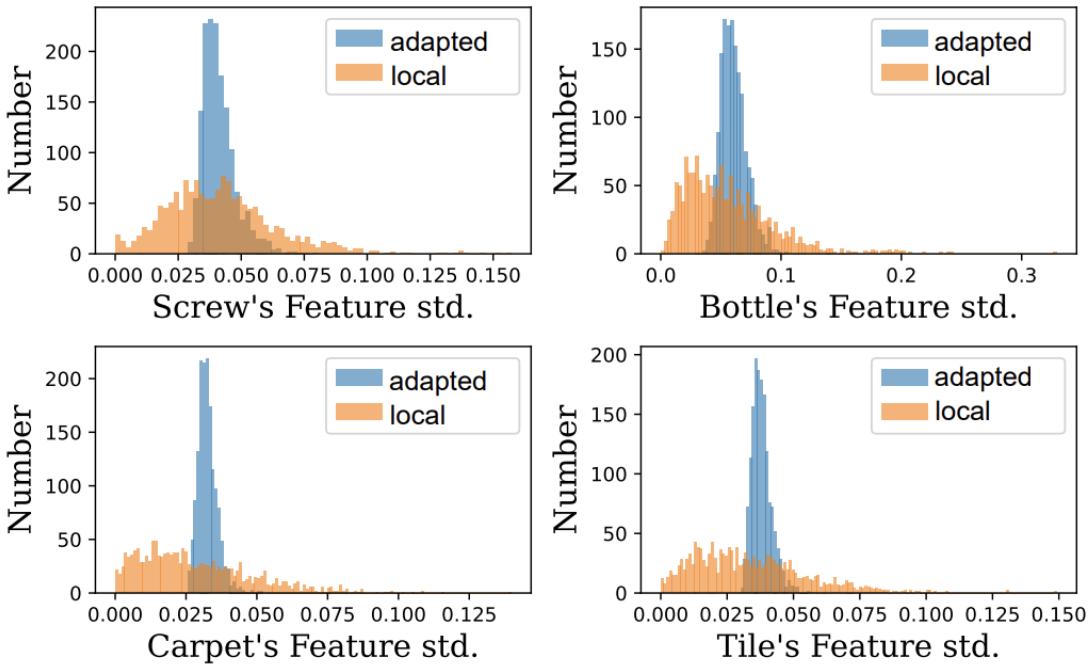
1.2.3 Generátor anomálnych príznakov

Pre trénovanie diskriminátora na odhad pravdepodobnosti, že vzorky sú normálne, je najjednoduchšie vzorkovať negatívne vzorky, teda chybových príznakov, a optimalizácia spolu s normálnymi vzorkami. Nedostatok chýb robí odhad distribúcie výberu neriešiteľným. Namiesto závislosti na dodatočných dátach na syntetizovanie obrázkov s chybami sa na normálne vzorky v priestore príznakov pridáva jednoduchý šum, čo prekonáva manipulované metódy [5].

Anomálne príznaky sú generované pridaním Gaussovského šumu na normálne príznaky $q_{h,w}^{i-} \in \mathbb{R}^C$. Formálne je vektor šumu $\epsilon \in \mathbb{R}^C$ vzorkovaný, pričom každá položka sa riadi nezávislým a rovnomerne distribuovaným Gaussovým rozdelením $N(\mu, \sigma^2)$. Anomálny príznak $q_{h,w}^{i-}$ je fúzovaný ako (8):

$$q_{h,w}^{i-} = q_{h,w}^{i} + \epsilon \tag{8}$$

Obrázok 3 ilustruje vplyv anomálnych príznakov na štyri triedy MVTec AD. Štandardná odchýlka pozdĺž každej dimenzie adaptovaných príznakov má tendenciu byť konzistentná. Priestor adaptovaných príznakov sa stáva kompaktnejším pri tréningu s anomálnymi príznakmi, než priestor lokálnych príznakov [5].



Obrázok 3. Histogram smerodajnej odchýlky pozdĺž každého rozmeru lokálneho príznaku a adaptovaného príznaku [5]

1.2.4 Diskriminátor

Diskriminátor D_ψ funguje ako hodnotiteľ známky normálnosti a priamo odhaduje normálnosť na každej pozícii (h,w). Keďže negatívne vzorky sú generované spolu s normálnymi príznakmi $\{q^i | x^i \in \mathcal{X}_{train}\}$, obe sa počas trénovania privádzajú do diskriminátora. Diskriminátor očakáva pozitívny výstup pre normálne príznaky a negatívny pre anomálne príznaky. Využíva sa štruktúra 2-vrstvého viacvrstvového perceptróna ako u bežných klasifikátorov a odhaduje sa normálnosť ako $D_\psi\ (q_{h,w}) \in \mathbb{R}$ [5].

1.2.5 Stratová funkcia a tréning

Jednoduchá skrátená $l1$ stratová funkcia je odvodená ako (9):

$$l^i_{h,w} = \max\left(0, th^+ - D_\psi\left(q^i_{h,w}\right)\right) + \max\left(0, -th^- + D_\psi\left(q^{i-}_{h,w}\right)\right) \tag{9}$$

kde th^+ a th^- sú ohraničujúce podmienky, ktoré zabraňujú preučeniu. Zvyčajne sú nastavené na 0,5 a -0,5. Cieľom tréningu je (10):

$$L = \min_{\theta,\psi} \sum_{x^i \in \mathcal{X}_{train}} \sum_{h,w} \frac{l^i_h, w}{H_0 * W_0} \tag{10}$$

[5]

1.2.6 Inferencia a skórovacia funkcia

Generátor anomálnych príznakov je pri inferencii vyradený. Zvyšné moduly môžu byť zoskupené do jednej end-to-end siete. Každé $x_i \in \mathcal{X}_{test}$ je postupne vložené do extraktora príznakov F_ϕ a adaptéra príznakov G_θ, aby boli získané adaptované príznaky $q^i_{h,w}$. Skóre anomálie poskytuje diskriminátor D_ψ ako (11):

$$s^i_{h,w} = -D_\psi(q^i_{h,w}) \tag{11}$$

Mapa anomálií, pre lokalizáciu anomálií počas inferencie, je definovaná ako (12):

$$S_{AL}(x_i) := \max_{(h,w)\in W_0 \times H_0} s^i_{h,w} \tag{12}$$

Potom je $S_{AL}(x_i)$ interpolované tak, aby malo priestorové rozlíšenie vstupného vzorku a Gaussovsky filtrované s $\sigma = 4$ pre hladké hranice. Keďže najcitlivejší bod existuje pre akúkoľvek veľkosť anomálnej oblasti, maximálne skóre mapy anomálií sa berie ako skóre detekcie anomálie pre každý obrázok [5].

1.3 RDOCE

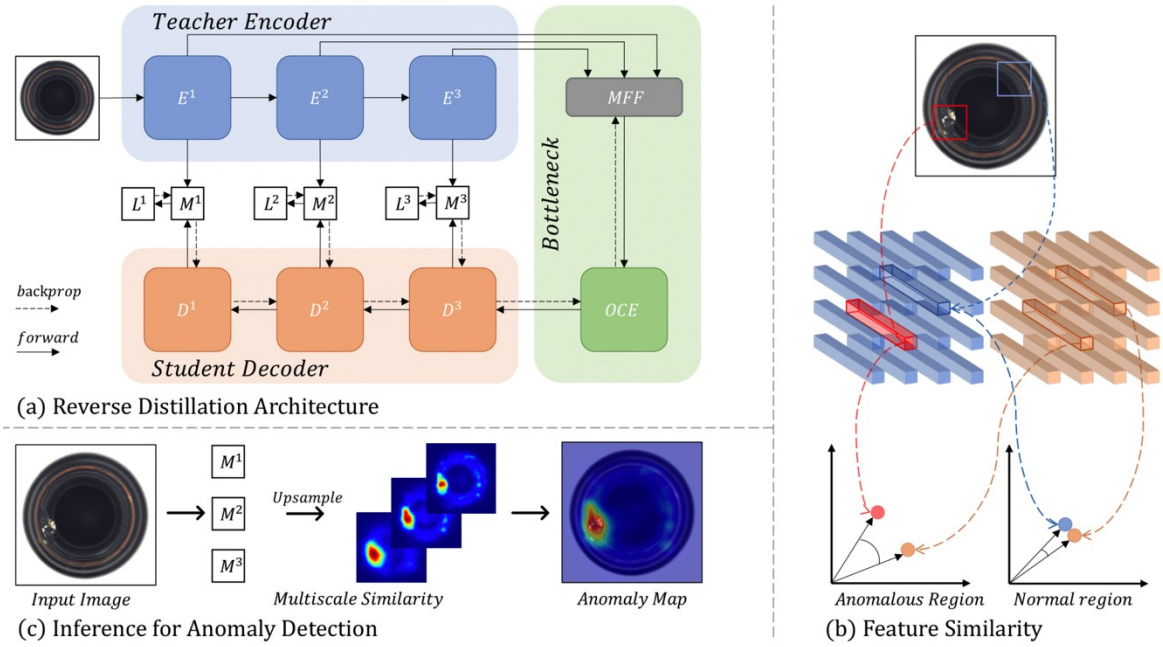
Reverse Distillation from One-Class Embedding (RDOCE) je metóda na detekciu anomálií pomocou tzv. „reverse distillation“ paradigma. Na rozdiel od konvenčného frameworku knowledge distillation (KD), kde učiteľ aj študent používajú štruktúru enkódera, v prípade reverse distillation má model učiteľa štruktúru enkódera, zatiaľ čo model študenta má štruktúru dekódera. Dekóder berie ako vstup nízkodimenzionálny embedding, ktorý sa používa na napodobnenie správania učiteľa tým, že obnovuje jeho reprezentácie v rôznych mierkach [6].

Tento prístup má dve významné výhody: nesimilárna štruktúra a kompaktný embedding. Prvá sa vyhýba zmäteniu spôsobenému nediskriminačnými filtrami, druhá pomáha zabrániť šíreniu nezvyčajných perturbácií k modelu študenta a zlepšuje diskrepanciu reprezentácií modelu na anomáliách. Tento prístup má dve významné výhody. Prvá výhoda je nesimilárna štruktúra, ktorá sa vyhýba zmäteniu spôsobenému nediskriminačnými filtrami. Druhá výhoda je kompaktný embedding, ktorý pomáha zabrániť šíreniu nezvyčajných perturbácií k modelu žiaka a zlepšuje nezrovnalosť reprezentácií modelu na anomáliách [6].

Okrem toho je zavedený jednotriedny bottleneck embedding modul (OCBE) na ďalšie zhustenie kódov príznakov. Ten pozostáva z bloku pre fúziu viacrozmerných vlastností

(MFF) a bloku pre jednorozmerný embedding (OCE), ktoré sú spoločne optimalizované s dekóderom [6].

Obrázok 4 zobrazuje reverse distillation framework navrhnutý autormi [6]. Tento framework pozostáva z trochu modulov: fixný vopred natrénovaný enkóder, trénovateľný jednotriedny bottleneck embedding modul a dekóder *D*. Pri danej vstupnej vzorke $I \in I^t$, učiteľ *E* extrahuje viacrozmerné reprezentácie. Študent *D* je trénovaný na obnovu príznakov bottleneck embeddingu. Počas testovania/inferencie môže reprezentácia extrahovaná enkóderom *E* v anomálnych vzorkách zachytiť abnormálne príznaky mimo distribúcie. Avšak dekóder *D* nedokáže rekonštruovať tieto anomálne príznaky z príslušného embeddingu. Nízka podobnosť anomálnych reprezentácií v navrhovanom teacher-student (T-S) modeli naznačuje vysoké hodnotenie abnormality [6]. Model T-S pozostáva z dvoch sieťových štruktúr, ktoré sa nazývajú teacher a student, teda učiteľ a študent. Model učiteľa je model so silnými schopnosťami, zatiaľ čo študent môže byť jednoduchý model. Model učiteľa sa používa na učenie modelu študenta prostredníctvom prenosu významných znalostí [7]. Heterogénne štruktúry enkódera a dekódera a opačné poradie knowledge distillation prispievajú k odlišným reprezentáciám anomálií. Okrem toho, trénovateľný OCBE modul ďalej kondenzuje viacrozmerné vzory do etrémne nízkorozmerného priestoru pre následnú rekonštrukciu normálnej reprezentácie. To zlepšuje odlišnosť príznakov na anomáliách v tomto T-S modeli, keďže abnormálne reprezentácie generované modelom učiteľa budú pravdepodobne opustené modelom OCBE [6]



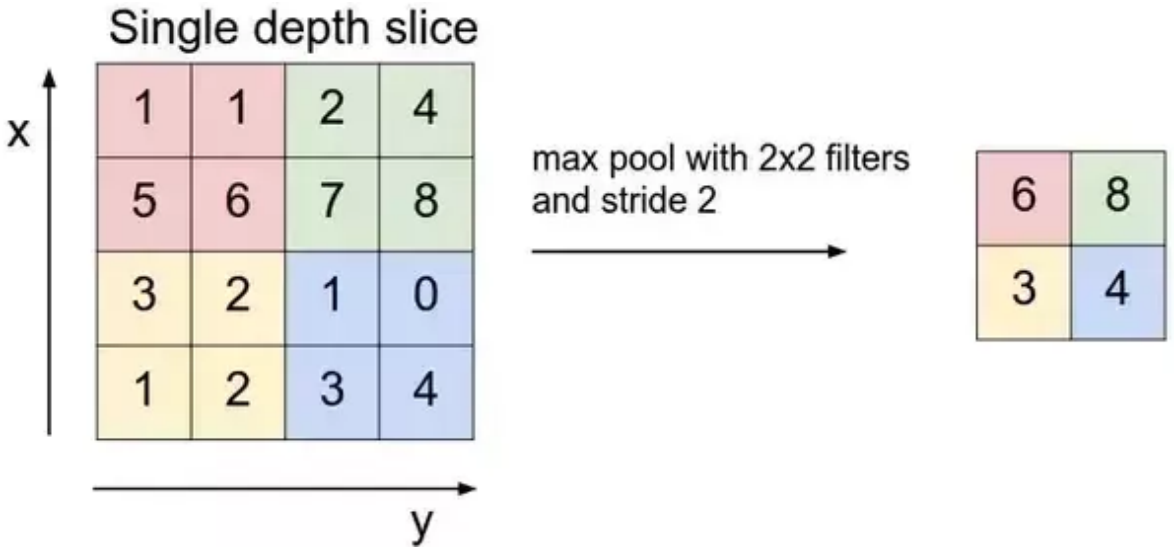
Obrázok 4. Reverse distillation framework [6]

1.3.1 Reverse Distillation

V konvenčnej knowledge distillation používa sieť študenta podobnú alebo identickú neurónovú sieť ako model učiteľa, prijíma surové dáta ako vstup a jej cieľom je zosúladiť svoje aktivačné príznaky s príznakmi učiteľa. V kontexte one-class distillation pre nekontrolovanú detekciu anomálií sa očakáva, že pri anomálnych vzorkách vygeneruje model študenta veľmi odlišné reprezentácie ako model učiteľa. Avšak disperzia aktivácií pri anomáliách sa niekedy stráca, čo vedie k zlyhaniu detekcie. Toto zlyhanie je pripisované podobným architektúram siete učiteľa a študenta a rovnakému toku dát počas T-S prenosu znalostí. Pre zlepšenie diverzity reprezentácií T-S modelu na neznámych vzorkách mimo distribúcie je navrhnuté paradigma Reverse Distillation, kde sa T-S model skladá z architektúry enkóder-dekóder a znalosti sú destilované z hlbokých vrstiev učiteľa na jeho predchádzajúce vrstvy, teda najprv sa na študenta prenášajú sémantické, vysokoúrovňové znalosti [6].

V paradigme Reverse Distillation sa enkóder zameriava na extrakciu komplexných reprezentácií. Ako základná kostra E je použitý enkóder vopred natrénovaný na datasete ImageNet. Aby sa zabránilo konvergencii T-S modelu k triviálnym riešeniam, sú počas prenosu znalostí všetky parametre učiteľa E zmrazené [6].

Architektúra dekódera *D* je symetrická, ale reverzná v porovnaní s *E*, aby sa zhodovala s priebežnými reprezentáciami *E*. Reverzný dizajn uľahčuje elimináciu reakcie siete študenta na abnormality, zatiaľ čo symetria umožňuje, aby mala rovnaký rozmer reprezentácie ako sieť učiteľa. Napríklad, ak je ako model učiteľa použitý ResNet, dekóder *D* pozostáva z niekoľkých reziduálnych dekódovacích blokov pre zrkadlovú symetriu. Konkrétne, ak je downsampling v ResNet realizovaný konvolučnou vrstvou s jadrom veľkosti 1 a krokom 2, príslušný dekódovací blok *D* používa dekonvolučné vrstvy s veľkosťou jadra 2 a krokom 2 [6] Downsampling je technika, ktorá umožňuje znížiť počet pixelov obrázka. Počet pixelov na danom obrázku sa znižuje v závislosti od vzorkovacej frekvencie, vďaka čomu sa znižuje rozlíšenie a veľkosť obrázka [20]. Konvolučné neurónové siete zvyčajne obsahujú niekoľko downsampling operátorov, ako napríklad strided konvolúcie alebo poolingové vrstvy [21]. Stride, alebo krok, je parameter filtra neurónovej siete, ktorý modifikuje množstvo pohybu nad obrázkom. Napríklad, ak je krok neurónovej siete nastavený na 1, filter sa posunie o jeden pixel. Veľkosť filtra ovplyvňuje výstupný kódovaný objem, takže sa krok často nastavuje na celé číslo [22]. Hlavnou úlohou poolingu je downsampling so zámerom redukcie komlexity pre ďalšie vrstvy. Pooling neovplyvňuje počet filtrov. Max-pooling je jedným z najbežnejších typov poolingu. Rozdeľuje obraz na obdĺžnikové podoblasti a vracia iba maximálnu hodnotu z tejto podoblasti. Ako je možné vidieť na obrázku nižšie, ak sa pooling vykonáva v ľavých horných 2x2 blokoch, ktoré sú zvýraznené ružovou farbou, posunie sa o dva bloky a zameriava sa na pravú hornú časť. To znamená, že je pri poolingu použitý stride s hodnotou 2.



Obrázok 5. Výsledok max-poolingu [27]

Downsampling sa používa iba vtedy, ak je skôr dôležitá prítomnosť informácií, než priestorové informácie. Pre zlepšenie účinnosti je možné využiť pooling s nepravidelnými filtrami a krokmi na zlepšenie efektivity. Napríklad 3 x 3 max-pooling so stride 2 udržiava určité prekrytie medzi oblasťami [23].

Počas trénovania sa dekóder *D* v reverse distillation snaží napodobniť správanie enkódera *E*. Pre detekciu anomálií bola v tejto metóde skúmaná viacrozmerná destilácia založená na príznakoch. Motiváciu za touto technikou je to, že plytké vrstvy neurónovej siete extrahujú lokálne deskriptory pre informácie na nízkej úrovni (napríklad farbu, hrany, textúru atď.), zatiaľ čo hlboké vrstvy majú širšie prijímacie polia, schopné charakterizovať regionálne/globálne sémantické a štrukturálne informácie. To znamená, že nízka podobnosť príznakov nízkej a vysokej úrovne v modeli T-S naznačuje lokálne abnormality a regionálne/globálne štrukturálne odchýlky [6].

Matematicky povedané, nech ϕ označuje projekciu z pôvodných dát *I* do jednotriedneho bottleneck embedding priestoru, spárovaná aktivačná korešpondencia v T-S modeli je $\{f_E^k = E^k(I), f_D^k = D^k(\phi)\}$, kde E^k a D^k reprezentujú k-tý enkódovací a dekódovací blok v modeli učiteľa a modeli študenta. $f_E^k, f_D^k \in \mathbb{R}^{C_k \times H_k \times W_k}$, kde C_k, H_k a W_k označujú počet kanálov, výšku a šírku k-tej vrstvy aktivačného tenzora. Pre prenos znalostí v T-S modeli sa za knowledge distillation stratu považuje kosínusová podobnosť, pretože zachytáva presnejšie vzťah v informáciách s vysokou aj nízkou dimenziou. Konkrétne, pre tenzory príznakov f_E^k a f_D^k je vypočítaná ich vektorová kosínusová strata podobnosti pozdĺž osi kanála a je získaná 2D mapa anomálií $M^k \in \mathbb{R}^{C_k \times H_k \times W_k}$ (13):

$$M^k(h,w) = 1 - \frac{(f_E^k(h,w))^T \cdot f_D^k(h,w)}{\|f_E^k(h,w)\| \|f_D^k(h,w)\|} \tag{13}$$

Vysoká hodnota M^k naznačuje vysokú anomáliu v danej oblasti. Pri zohľadnení viacrozmerného prenosu znalosti sa skalárna strata pre optimalizáciu študenta získa sčítaním viacrozmerných máp anomálií (14):

$$L_{KD} = \sum_{k=1}^{K} \left\{ \frac{1}{H_k W_k} \sum_{h=1}^{H_k} \sum_{w=1}^{W_k} M^k(h,w) \right\} \tag{14}$$

kde *K* indikuje počet príznakov použitých v experimente [6].

1.3.2 Jednotriedny Bottleneck Embedding

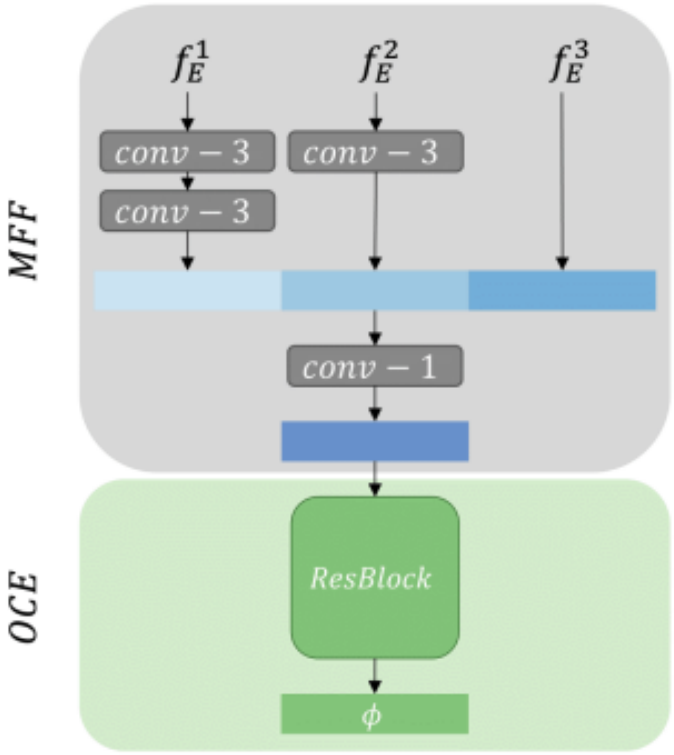
Keďže sa model študenta *D* snaží obnoviť reprezentácie modelu učiteľa *E*, môže sa priamo napojiť na výstup aktivačnej vrstvy posledného kódovacieho bloku *D*. Toto napojenie má však dva nedostatky. Prvým nedostatkom je, že má model učiteľa v KD zvyčajne vysokú kapacitu. Hoci vysokokapacitný model pomáha extrahovať bohaté príznaky, získané vysokorozmerné deskriptory majú pravdepodobne značnú redundanciu. Vysoká voľnosť a redundancia reprezentácií škodí modelu študenta pri dekódovaní podstatných príznakov bez anomálií. Druhým nedostatkom je to, že aktivácia posledného dekódovacieho bloku zvyčajne charakterizuje sémantické a štrukturálne informácie o vstupných dátach. V dôsledku reverzného poradia knowledge distillation predstavuje priame napojenie tejto vysokoúrovňovej reprezentácie na dekóder výzvu pre rekonštrukciu nízkoúrovňových príznakov [6].

Aby bol vyriešený prvý vyššie uvedený nedostatok pri one-class destilácii, bol zavedený trénovateľný once-class embedding blok, ktorý projektuje vysokorozmerné reprezentácie modelu učiteľa *E* do nízkorozmerného priestoru. Anomálne príznaky môžu byť formulované ako perturbácie na normálnych vzoroch. Tým pádom kompaktný embedding funguje ako informačný bottleneck a pomáha zabrániť šíreniu neobvyklých perturbácií do modelu študenta, čím sa zvyšuje rozdiel v reprezentácii T-S modelu na anomáliách [6].

Na riešenie problému obnovy nízkoúrovňových príznakov v dekóderi *D* zreťazuje MFF blok viacrozmerné reprezentácie pred one-class embeddingom. Na dosiahnutie zarovnania reprezentácie pri zreťazení príznakov sú plytké príznaky zmenšené prostredníctvom jednej alebo viacerých 3×3 konvolučných vrstiev s krokom 2, po ktorých nasleduje batch normalizácia a aktivačná funkcia rectified linear activation unit (ReLU) [6]. ReLU je typ aktivačnej funkcie, ktorá je matematicky definovaná ako $y = max(0, x)$. Je lineárna pre všetky kladné hodnoty a nulová pre všetky záporné hodnoty [28]. Potom sa využije 1×1 konvolučná vrstva s krokom 1 a batch normalizácia s ReLU pre získanie bohatých, ale kompaktných príznakov [6].

Modul OCBE je znázornený na obrázku číslo 6, kde MFF agreguje nízkoúrovňové a vysokoúrovňové príznaky na vytvorenie bohatého embeddingu pre rekonštrukciu normálnych vzorov a OCE sa zameriava na zachovanie podstatných informácií, ktoré umožňujú dekódovanie odpovede učiteľa. Konvolučné vrstvy, ktoré sú znázornené sivou

farbou a ResBlock, znázornený zelenou farbou, sú trénovateľné a optimalizované spoločne s modelom dekódera *D* počas knowledge distillation na normálnych vzorkách [6].



Obrázok 6. Jednotriedny bottleneck embedding modul [6]

1.3.3 Skórovanie anomálií

Vo fáze inferencie sa najprv zvažuje meranie skóre anomálie na úrovni pixelov pre lokalizáciu anomálií (AL). Keď je vzorka anomálna, model učiteľa je schopný reflektovať abnormality vo svojich príznakoch. Model študenta však pravdepodobne zlyhá pri obnove abnormálnych príznakov, pretože dekóder sa učí obnovovať iba reprezentácie bez anomálií z kompaktného one-class embedding pri knowledge distillation. Inými slovami, keď je testovacia vzorka anomálna, dekóder generuje nezhodné reprezentácie od učiteľa. Súbor máp anomálií je získaný z dvojíc reprezentácií T-S, kde hodnota v mape M^k vyjadruje bodovú anomáliu k-tého tenzora príznakov. Na lokalizáciu anomálií v obrázku je mapa M^k zväčšená na veľkosť obrázka. Nech Ψ označuje bilineárnu operáciu zväčšenia, potom je presná mapa skóre S_{Iq} formulovaná ako pixelová akumulácia všetkých máp anomálií (15):

$$S_{AL} = \sum_{i=1}^{L} \Psi\left(M^i\right) \tag{15}$$

Pre odstránenie šumov vo výslednej mape, je S_{AL} vyhladená pomocou Gaussovho filtra [6].

Pre detekciu anomálií je priemerovanie všetkých hodnôt v mape S_{AL} nespravodlivé pre vzorky s malými anomálnymi oblasťami. Najcitlivejší bod existuje pre akúkoľvek veľkosť anomálnej oblasti. Preto je maximálna hodnota v S_{AL} definovaná ako skóre anomálie na úrovni vzorky S_{AD}. Dôvodom je, že pre normálne vzorky neexistuje významná odozva v ich mape skóre anomálie [6].

1.4 PatchCore

Algoritmus PatchCore je založený na myšlienke, že obrázok môže byť klasifikovaný ako anomálny hneď, ako je anomálny aspoň jeden patch. Vstupná obrázok je rozdelený na dlaždice (tiles), ktoré slúžia ako patche a sú vstupom pre neurónovú sieť. Skladá sa z jednej vopred natrénovanej siete, ktorá sa používa na extrakciu patchov s príznakmi „strednej“ úrovne. „Stredná“ úroveň sa vzťahuje na vrstvu extrakcie príznakov modelu neurónovej siete. Príznaky nižšej úrovne sú vo všeobecnosti príliš rozsiahle a príznaky vyššej úrovne sú špecifické pre súbor údajov, na ktorom je model trénovaný. Príznaky extrahované počas trénovania sa ukladajú do pamäťovej banky príznakov patchov citlivej na okolie [9].

Počas inferencie sa táto pamäťová banka downsampluje pomocou algoritmu coreset subsampling. Coreset subsampling generuje podmnožinu, ktorá najlepšie aproximuje štruktúru dostupnej množiny a umožňuje nájdenie približného riešenia. Táto podmnožina pomáha znížiť náklady na vyhľadávanie spojené s hľadaním najbližšieho suseda. Skóre anomálie sa berie ako maximálna vzdialenosť medzi testovacím patchom v súbore testovacích patchov a každým príslušným najbližším susedom [9].

1.4.1 Lokálne citlivé príznaky patchu

Na označenie množiny všetkých nominálnych obrázkov ($\forall x \in X_N : y_x = 0$) dostupných počas trénovania sa používa X_N, pričom $y_x \in \{0, 1\}$ označuje, či je obrázok *x* nominálny (0), alebo anomálny (1). V súlade s tým je definované X_T, ako množina vzoriek poskytnutých počas testovania s $\forall x \in X_T : y_x \in \{0, 1\}$. PatchCore používa sieť ϕ, ktorá je vopred natrénovaná na datasete ImageNet. Keďže príznaky na špecifických hierarchiách siete zohrávajú dôležitú úlohu, je na označenie príznakov pre obrázok $x_i \in X$ (s datasetom *X*)

a hierarchickú úroveň j vopred natrénovanej siete ϕ použitá $\phi_{i,j} = \phi_j(x_i)$. Ak nie je uvedené inak, j indexuje mapy príznakov z architektúr podobných ResNet, ako napríklad ResNet-50 [25] alebo WideResnet-50 [24], pričom $j \in \{1, 2, 3, 4\}$ označuje končený výstup príslušných blokov s priestorovým rozlíšením [8].

Jednou z možností pre reprezentáciu príznakov by bola posledná úroveň v hierarchii príznakov siete. To však spôsobuje dva problémy. Po prvé, stráca sa viac lokalizovaná nominálna informácia. Typy anomálií, ktoré sa vyskytujú počas testovania nie sú známe apriori, čo sa stáva škodlivým pre nadväzujúcu výkonnosť detekcie anomálií. Po druhé, veľmi hlboké a abstraktné príznaky v sieťach vopred natrénovaných na datasete ImageNet sú zaujaté pre úlohu klasifikácie prirodzených obrázkov, ktorá sa len málo prekrýva s úlohou detekcie priemyselných anomálií [8].

Preto sa používa pamäťová banka *M* patchových príznakov obsahujúcich stredné príznakové reprezentácie na využitie poskytnutého tréningového kontextu a vyhnutie sa príliš všeobecným, alebo príliš silne zaujatým príznakom voči ImageNet klasifikácii. V konkrétnom prípade architektúr podobných ResNetu by to zahŕňalo napríklad $j \in \{2, 3\}$. Na formalizovanie reprezentácie patchu je rozšírená predtým zavedená notácia. Za predpokladu, že mapa príznakov $\phi_{i,j} \in \mathbb{R}^{c^* \times h^* \times w^*}$ je trojrozmerný tenzor s hĺbkou c^*, výškou h^* a šírkou w^*, je použitá $\phi_{i,j}(h, w) = \phi_j(x_i, h, w) \in \mathbb{R}^{c^*}$ na označenie c^*-dimenzionálneho fragmentu príznaku na pozícii $h \in \{1, \dots, h^*\}$ a $w \in \{1, \dots, w^*\}$. Za predpokladu, že veľkosť recepčného poľa každej $\phi_{i,j}$ je väčšia ako 1, ide efektívne o reprezentácie príznakov obrazových patchov. Ideálne by mala každá reprezentácia patchu fungovať na dostatočne veľkom recepčnom poli, aby zohľadňovala významné anomálne kontexty odolné voči lokálnym priestorovým variáciám. Aj keď by sa to dalo dosiahnuť pomocou strided poolingu a ďalším zostupom hierarchou siete, takto vytvorené príznaky sa stanú špecifickými pre ImageNet, a teda menej relevantnými pre úlohu detekcie anomálií, zatiaľ čo sa zvyšujú náklady na tréning a efektívne rozlíšenie mapy príznakov klesá [8].

To motivuje použitie lokálnej agregácie okolitých hodnôt pri komponovaní každej reprezentácie príznaku na úrovni patchu na zvýšenie veľkosti recepčného poľa a robustnosti voči malým priestorovým odchýlkam, bez strát priestorového rozlíšenia alebo použiteľnosti máp príznakov. Preto je vyššie uvedená notácia pre $\phi_{i,j}(h, w)$ rozšírená tak, aby zohľadňovala nerovnomerné veľkosti patchov p, pričom sú zahrnuté vektory príznakov z okolia ako (16):

$$N_p^{(h,w)} = \{(a,b)|a \in [h - \lfloor p/2 \rfloor, \dots, h + \lfloor p/2 \rfloor], b \in [w - \lfloor p/2 \rfloor, \dots, w + \lfloor p/2 \rfloor]\} \tag{16}$$

a lokálne citlivé príznaky na pozícii (*h, w*) ako (17):

$$\phi_{i,j}\left(N_p^{(h,w)}\right) = f_{agg}(\left\{\phi_{i,j}(a,b)|(a,b) \in N_p^{(h,w)}\right\}) \tag{17}$$

s funkciou f_{agg} pre agregáciu vektorových príznakov v okolí $N_p^{(h,w)}$. Pre PatchCore je použitý adaptívny priemerný pooling. Je to podobné lokálnemu vyhladzovaniu každej individuálnej funkčnej mapy a vedie k jedinej reprezentácii na pozícii (*h, w*) s preddefinovanou dimezionalitou *d*, ktorá sa vykonáva pre všetky dvojice (*h, w*), kde $h \in \{1, \dots, h^*\}$ a $w \in \{1, \dots, w^*\}$, čím sa zachováva rozlíšenie máp príznakov. Pre tenzor mapy príznakov $\phi_{i,j}$ je jeho kolekcia lokálne citlivých príznakov na základe patchu $P_{s,p}\left(\phi_{i,j}\right)$ (18):

$$P_{s,p}(\phi_{i,j}) = \left\{\phi_{i,j}\left(N_p^{(h,w)}\right)|h,w \bmod s = 0, h < h^*, w < w^*, h,w \in \mathbb{N}\right\} \tag{18}$$

s voliteľným použitím parametra stridingu *s*, ktorý je nastavený na 1. Empiricky bolo zistené, že agregácia viacerých hierarchií príznakov ponuka určité výhody. Avšak, aby sa zachovala všeobecnosť použitých príznakov a priestorové rozlíšenie, používa PatchCore iba dve stredne pokročilé hierarchie príznakov j a $j+1$. To je dosiahnuté jednoduchým výpočtom $P_{s,p}(\phi_{i,j+1})$ a agregáciou každého prvku s jeho odpovedajúcim príznakom patchu na najnižšej použitej úrovni hierarchie, teda pri najvyššom rozlíšení, čo je dosiahnuteľné bilineárnym preškálovaním $P_{s,p}(\phi_{i,j+1})$, aby sa $P_{s,p}(\phi_{i,j+1})$ a $P_{s,p}(\phi_{i,j})$ zhodovali. Nakoniec sa pamäťová banka PatchCore *M* pre všetky nominálne tréningové vzory $x_i \in X_n$ definuje jednoducho ako (19):

$$M = \bigcup_{x_i \in X_N} \mathcal{P}_{s,p}(\phi_j(x_i)) \tag{19}$$

[8]

1.4.2 Coreset-redukovaná pamäťová banka príznakov patchu

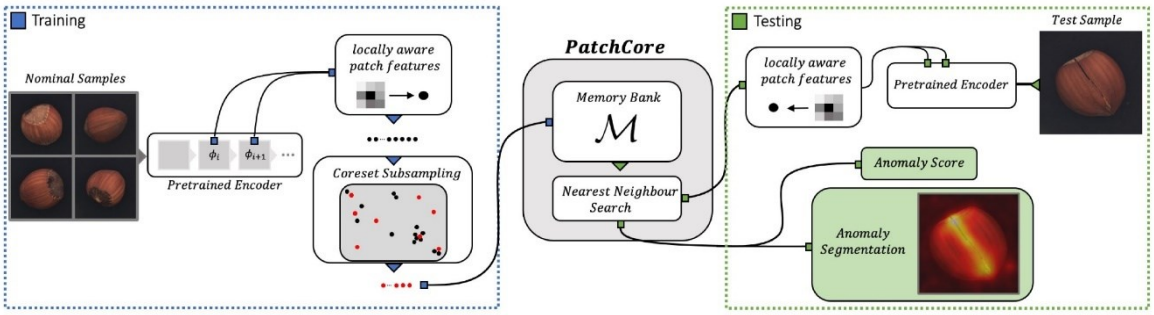
Pri zvyšovaní veľkosti X_N sa *M* stáva nadmerne veľkou a s tým sa zvyšuje aj čas inferencie potrebný na vyhodnotenie nových testovacích údajov a potrebné úložisko. Tieto problémy je možné riešiť tým, že bude *M* zmysluplne vyhľadateľná pre väčšie veľkosti a počty obrázkov, čo umožní porovnávanie na základe patchov, ktoré je prospešné pre detekciu anomálií a segmentáciu. To vyžaduje, aby pokrytie nominálnych príznakov enkódovaných

v *M* zostalo zachované. Bohužiaľ, náhodný subsampling, najmä v niekoľkých magnitúdach, stratí významné informácie, ktoré sú k dispozícii v *M* enkódovaní v pokrytí nominálnych príznakov. V tomto modeli je na redukciu *M* použitý coreset subsampling mechanizmus, ktorý znižuje čas inferencie pri zachovaní výkonu [8].

V konceptuálnom zmysle má výber coresetu za úlohu nájsť podmnožinu $S \subset A$ tak, aby riešenia problému nad *A* boli čo najpresnejšie a hlavne rýchlejšie aproximované riešeniami, ktoré boli vypočítané nad *S*. V závislosti od konkrétneho problému sa líši coreset, ktorý je predmetom záujmu. Keďže PatchCore používa výpočty najbližšieho suseda, je použitý minimax facility location coreset výber, na zaistenie približne podobného pokrytia *M*-coresetu M_c v priestore príznakov na úrovni patchu ako v pôvodnej pamäťovej banke *M* (20):

$$M_C^* = \underset{M_C \subset M}{\operatorname{argmin}} \max_{m \in M} \min_{n \in M_C} \|m - n\|_2 \tag{20}$$

[8]



Obrázok 7. Prehľad modelu PatchCore [8]

1.4.3 Detekcia anomálie s PatchCore

S nominálnou pamäťovou bankou príznakov patchov *M* je skóre anomálie na úrovni obrázka $s \in \mathbb{R}$ pre testovací obrázok x^{test} odhadované ako maximálne skóre vzdialeností s^* medzi testovacími príznakmi patchov v jeho kolekcii patchov $P(x^{test}) = P_{s,p}(\phi_j(x^{test}))$ ku každému príslušnému najbližšiemu susedovi m^* v *M* (21):

$$\begin{aligned} m^{test,*}, m^* &= \underset{m^{test} \in P(x^{test})}{\operatorname{argmax}} \underset{m \in M}{\operatorname{argmin}} \|m^{test} - m\|_2 \\ s^* &= \|m^{test,*} - m^*\|_2 \end{aligned} \tag{21}$$

Pre získanie hodnoty *s* je použité škálovanie *w* na hodnotu s^* na zohľadnenie správania susedných patchov. Hodnota anomálie sa zvyšuje, ak sú príznaky pamäťovej banky najbližšie k anomálnemu kandidátovi $m^{test,*}$, m^* sami ďaleko od susedných vzoriek (22):

$$s = \left(1 - \frac{\exp \|m^{test,*} - m^*\|_2}{\sum_{m \in N_b(m^*)} \exp \|m^{test,*} - m\|_2}\right) \cdot s^* \tag{22}$$

kde *b* v $N_b(m^*)$ sú najbližšie príznaky patchov v *M* pre testovací príznak patchu m^*. Táto hodnota je robustnejšia ako iba maximálna vzdialenosť patchov. Priamo po získaní *s* nasleduje proces identifikácie a ohraničenia oblastí, v ktorých sa vyskytujú anomálie. Tento proces sa nazýva segmentácia. Skóre anomálií na úrovni obrázkov vyžaduje výpočet skóre anomálií pre každú časť patchu pomocou operácie arg max. V rovnakom kroku môže byť vypočítaná segmentačná mapa, pre usporiadanie vypočítaných skóre anomálií častí patchov, na základe ich príslušnej priestorovej polohy. Výsledok je zväčšený bilineárnou interpoláciu, aby sa dosiahlo pôvodné vstupné rozlíšenie. Na záver je výsledok vyhladený Gaussovým filtrom s jadrom šírky σ = 4 [8].

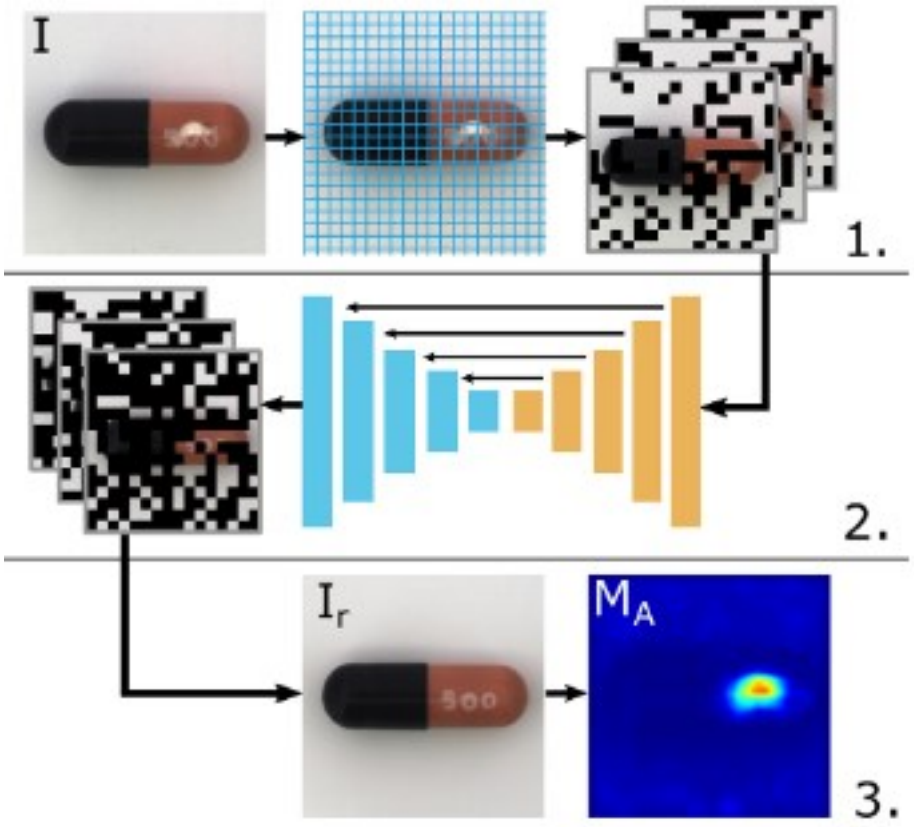
2 METÓDY ZALOŽENÉ NA REKONŠTRUKCII

Tieto metódy vychádzajú z predpokladu, že anomálne oblasti obrázkov by nemali byť správne zrekonštruované, pretože neexistujú v tréningových vzorkách. Niektoré metódy využívajú generatívne modely na kódovanie a rekonštrukciu normálnych dát, ako sú napríklad autoenkódery a generatívne adversariálne siete. Iné metódy formulujú detekciu anomálií ako problém inpaintingu, kde sa náhodne maskujú fragmenty z obrázkov. Potom sú využité neurónové siete na predpovedanie chýbajúcich informácií. Pri trénovaní sa často používa structured similarity index strata (SSIM) [10]. Mapa anomálií sa generuje ako pixelová odchýlka medzi vstupným obrázkom a jeho rekonštrukciou. Avšak, ak anomálie zdieľajú spoločné vzory, napríklad lokálne hrany, s normálnymi tréningovými dátami, alebo ak je dekóder „príliš silný“ na to, aby dekódoval niektoré abnormálne kódovania, je pravdepodobné, že anomálie na obrázkoch budú dobre zrekonštruované [5].

2.1 RIAD

Reconstruction by inpainting for visual anomaly detection (RIAD) je model detekcie anomálií, ktorý využíva metódu opravy obrazu (inpainting) a rieši detekciu anomálií ako problém rekonštrukcie obrázka. Na rozdiel od auto-enkóderov, RIAD rekonštruuje lokálne oblasti iba na základe ich okolitého prostredia a nezahŕňa pixely v oblasti, ktorá sa rekonštruuje. Tým pádom je pravdepodobnosť presnej rekonštrukcie anomálie zovšeobecnením vzhľadu okolia veľmi nízka, pretože ich okolité prostredie môže byť odlišné. Na druhej strane, rekonštrukcia neanomálnych oblastí nie je ovplyvnená, pretože je sieť trénovaná na obrázkoch bez anomálií, a preto sú takéto oblasti veľmi dobre modelované [11].

Obrázok I je rozdelený na mriežku obdĺžnikových oblastí veľkosti $k \times k$ pixelov. Sada obdĺžnikových oblastí je náhodne rozdelená na n disjunktných podmnožín. Pre každú podmnožinu sú oblasti patriace do tejto podmnožiny odstránené z pôvodného obrázka, teda nastavené na 0, čím sa získa n vstupných obrázkov. Vstupné obrázky sú rekonštruované inpainting sieťou, ktorá generuje n vstupných obrázkov, pričom každý rekonštruuje oblasti odstránené vo svojom príslušnom vstupnom obrázku. Jednotlivé rekonštruované oblasti z n čiastočných rekonštrukcií sú znovu zoskupené do jedného rekonštruovaného obrázka I_r. Kvalita rekonštrukcie sa potom vyhodnocuje s cieľom vytvoriť mapu anomálií M_A [11].



Obrázok 8. Hlavné kroky modelu RIAD [11]

2.1.1 Formulácia rekonštrukcie pomocou inpaintingu

V tomto prístupe sú náhodne vybrané oblasti vstupného obrázka nastavené na hodnotu nula a doplnené pomocou naučenej siete. Konkrétne, niekoľko prepojených oblastí sa vzorkuje a následne odstráni z obrázka. Vstupný obrázok sa upraví odstránením množiny pixelov, rozdelením vstupného obrázka na štvorcové oblasti veľkosti k. Každý obrázok sa rozdelí na mriežku s rozmermi $\frac{H}{k} \times \frac{W}{k}$, kde W a H sú šírka a výška obrázka. Každý prvok mriežky je štvorec o veľkosti $k \times k$ pixelov. Obrázky sa zmenšia tak, aby ich rozmery boli deliteľné k. Mriežka je náhodne rozdelená na n disjunktných množín S_i, pričom každá obsahuje $\frac{N}{n}$ prvkov mriežky, kde $N = \frac{H}{k} \times \frac{W}{k}$ je počet všetkých prvkov mriežky. Daná množina S_i obsahuje približne $\frac{1}{n}$ všetkých pixelov. Veľkosť a podiel odstránených oblastí môžu byť kontrolované nastavením hyperparametrov k a n. Pre každú množinu oblastí S_i sa generuje maska M_{S_i} rovnakej šírky a výšky ako vstupný obrázok i. M_{S_i} je binárna maska, ktorá obsahuje nuly v oblastiach patriacich do S_i. Počas inferencie sa M_{S_i} používa na nastavenie oblastí patriacich do S_i na nulu v obrázku i. Oblasti nastavené na nulu sa potom rekonštruujú

pomocou natrénovanej siete. Celý obrázok sa rekonštruuje čiastočnou rekonštrukciou každej množiny S_i, kde $i \in \{1, 2, \dots, n\}$, ako zjednotenie oblastí patriacich do množín S_i pre všetky $i \in \{1, 2, \dots, n\}$, pokrývajú celý obrázok (23):

$$\sum_{i=1}^{n} \overline{M_{S_i}} = 1_{H \times X} \tag{23}$$

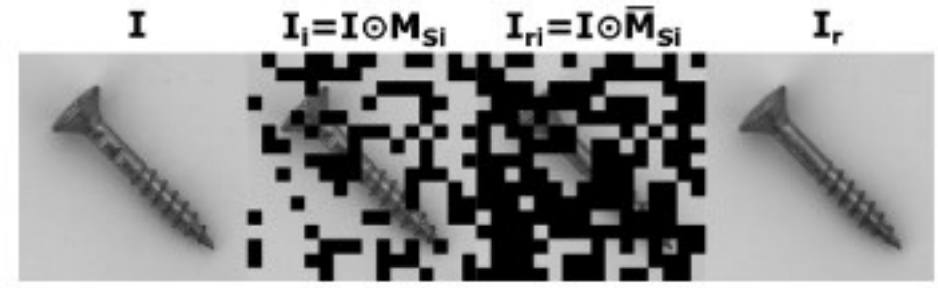
kde $\overline{M_{S_i}}$ je binárna inverzia M_{S_i} a $1_{H \times X}$ je matica jednotiek veľkosti $H \times W$ [11].

Sieť prijíma ako vstup maskované obrázky $l_i = M_{S_i} \odot l$. Obrázky l_i sú do siete vkladané postupne a sieť vykonáva inpainting pre každý obrázok individuálne. Rekonštruované obrázky l_{ri} sú maskované a sčítané do konečnej rekonštrukcie l_r (24):

$$l_r = \sum_{i=1}^{n} \overline{M_{S_i}} \odot l_{ri} \tag{24}$$

[11]

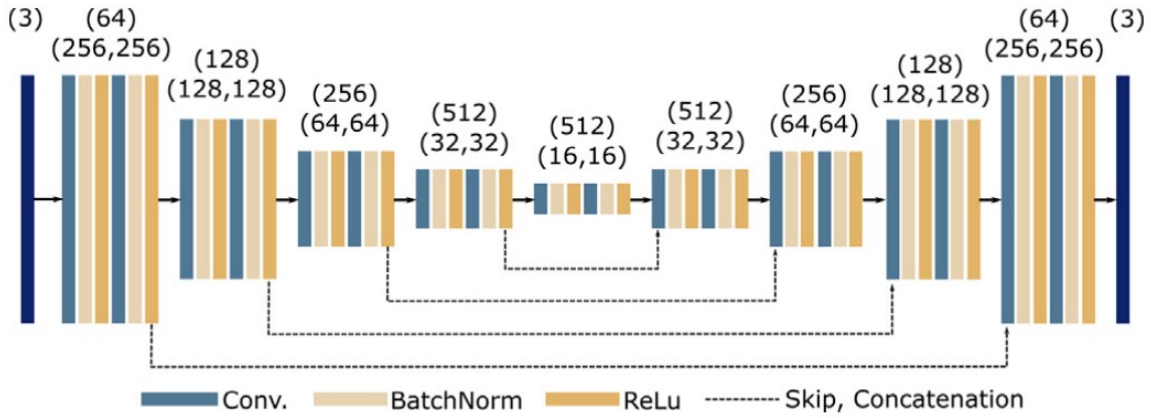
Konečný rekonštruovaný obrázok l_i je zostavený z čiastočne rekonštruovaných obrázkov l_{ri}. V každom čiastočne rekonštruovanom obrázku l_{ri} sú oblasti, ktoré nepatria do S_i, nastavené na nulu násobením l_{ri} s M_{S_i}. Každý l_{ri} tak prispieva len oblastiam patriacom do S_i, ktoré boli odstránené na vstupe. Príklad postupu maskovania je zobrazený na obrázku 9 [11].



Obrázok 9. Proces maskovania obrázka [11]

2.1.2 Architektúra rekonštrukčnej siete

Na rekonštrukciu odstránených oblastí sa používa encoder-decored sieť založená na U-Net architektúre [18]. Architektúra použitej siete je zobrazená na obrázku 10. Preskočenie spojení sa používa na prenos príznakov medzi rôznymi vrstvami siete, čo vedie k presnej rekonštrukcii detailov, ktoré sú inak ťažko rekonštruovateľné [11].



Obrázok 10. Architektúra konvolučnej neurónovej siete [11]

Pri trénovaní autoenkóderov sa zvyčajne používa per-pixel strata L_2. Tá však predpokladá nezávislosť medzi susednými pixelmi, čo je často nesprávne. Z toho dôvodu sa používajú straty, ktoré penalizujú štrukturálne rozdiely medzi rekonštruovanými oblasťami a oblasťami patriacimi k pôvodnému obrázku. Konkrétne sa navrhuje použiť multi-scale gradient magnitude similarity (MSGMS) strata a structured similarity index (SSIM) strata. Obe sa využívajú ako metriky podobnosti patchu, ktoré sa zameriavajú na rôzne vlastnosti obrázka [11].

Gradient magnitude similary (GMS) medzi pôvodným a rekonštruovaným obrázkom sa vypočíta najprv vypočítaním máp gradientovej magnitúdy pôvodného a rekonštruovaného obrázka (25):

$$g(l) = \sqrt{(l * h_x)^2 + (l * h_y)^2} \tag{25}$$

kde $g(l)$ je mapa magnitúdy gradientu pre obraz I, h_x a h_y sú Prewitt filtre s rozmermi 3×3 pozdĺž x a y rozmerov, a $*$ predstavuje operáciu konvolúcie. Mapa podobnosti gradientovej magnitúdy medzi pôvodným obrázkom I a rekonštruovaným obrázkom I_r je potom definovaná ako (26):

$$GMS(I, I_r) = \frac{2g(I)g(I_r) + c}{g(I)^2 + g(I_r)^2 + c} \tag{26}$$

kde c je konštanta zabezpečujúca numerickú stabilitu a pomáha s reguláciou odpovede v oblastiach s vysokým šumom. GMS je rozšírená na viacrozmernú variantu MSGMS, výpočtom na niekoľkých škálach obrázka. Strata MSGMS sa vypočítava na pyramíde obrázkov so štyrmi rôznymi rozmermi. Vyhladené podvzorkované obrázky sa generujú na

niekoľkých škálach pomocou priemerného poolingu každého obrázka, niekoľkokrát, s posuvným oknom veľkosti 2×2 a krokom 2. Okrem pôvodného obrázka sa výsledná pyramída skladá z obrázkov, ktoré majú veľkosť $\frac{1}{2}, \frac{1}{4}$ a $\frac{1}{8}$ pôvodnej veľkosti obrázka. MSGMS sa definuje ako priemerná hodnota mapy vzdialenosti GMS na niekoľkých škálach (27):

$$L_G(I, I_r) = \frac{1}{4}\sum_{i=1}^{4}\frac{1}{N_l}\sum_{i=1}^{H_1}\sum_{j=1}^{W_1} 1 - GMS(I_l, I_{rl})_{(i,j)} \tag{27}$$

kde H_l a W_l sú výška a šírka obrázka a N_l predstavuje počet pixelov na škále l. I_l a I_{rl} sú pôvodný a rekonštruovaný obrázok na škále l. $GMS(I_l, I_{rl})_{(i,j)}$ je hodnota mapy GSM pre I_l a I_{rl} na pixeli (i, j) [11].

Strata SSIM je definovaná ako (28):

$$L_s(I, I_r) = \frac{1}{N_p}\sum_{i=1}^{H}\sum_{j=1}^{W} 1 - SSIM(I, I_r)_{(i,j)} \tag{28}$$

kde I a I_r sú pôvodný a rekonštruovaný obrázok, N_p je počet pixelov v obrázku I a $SSIM(I, I_r)_{(i,j)}$ je hodnota SSIM medzi dvomu patchmi I a I_r so stredom v bode (i, j). Celková strata zohľadňuje straty MSGMS a SSIM, ako aj pixelovú stratu L_2 pre reguláciu (29):

$$L = \lambda_G L_G + \lambda_S L_S + L_2 \tag{29}$$

kde λ_G a λ_S sú váhy individuálnych strát [11].

2.1.3 Viacrozmerné trénovanie

Presnosť rekonštrukcie oblasti závisí od veľkosti oblasti, ktorá bola počas inferencie odstránená. Keďže detekcia anomálií sa spolieha na to, že rekonštrukcia bude v neanomálnych oblastiach čo najvernejšia, výkonnosť detekcie anomálií môže závisieť aj od použitej veľkosti oblasti k a od veľkosti anomálie, ktorá sa rekonštruuje. Ak je k oveľa väčšie ako anomália, presná rekonštrukcia pomocou inpaintingu nie je možná. Naopak, ak je k príliš malé, sieť by mohla inferovať časti anomálie zo svojho okolia. Keďže sa anomálie vyskytujú v rôznych veľkostiach, ich detekcia musí zohľadňovať niekoľko škál. Spoľahlivejšia mapa chýb rekonštrukcie môže byť generovaná zohľadnením viacerých rekonštrukcií jednotlivých obrázkov, vygenerovaných pomocou niekoľkých k hodnôt [11].

Pre zvýšenie robustnosti detekcie anomálií sa počas trénovania zohľadňuje niekoľko veľkostí maskovaných lokálnych oblastí. Konkrétne veľkosť *k* sa vyberá z množiny hodnôt $K = \{k_i\}_{i=1:N_K}$, kde N_K je kardinalita množiny [11].

2.1.4 Detekcia anomálie

Vstupný obrázok *I* je najprv rozdelený na mriežku, ktorá je rozdelená na *n* disjunktných množín S_i. Proces generovania rekonštruovaného obrázka I_r je rovnaký ako počas trénovania. Skóre anomálií na úrovni pixelov, ako aj na úrovni obrázka, sa potom vypočítajú porovnaním I_r a *I* [11].

Mapa skóre anomálií sa získa najskôr výpočtom mapy GMS, ktorá zohľadňuje vstupný a rekonštruovaný obrázok v rôznych mierkach. Pre každú škálu *I* sa vypočíta škálovaná mapa GMS zo vstupného obrázka I_l a rekonštruovaného obrázka I_{rl}, ktorý je zmenšený na škálu *I*, použitím rovnakého postupu zmenšovania, ako pri výpočte straty MSGMS počas trénovania. $GMS(I_l, I_{rl})$ je potom zväčšené na pôvodné rozlíšenie. Viacrozmerná GMS mapa $MSGMS(I,I_r)$ je potom vypočítaná ako priemerná hodnota jednotlivých škálovaných GMS máp pre každý pixel. Anomálie majú tendenciu zaujímať väčšie, priestorovo prepojené oblasti, preto sa môže chyba rekonštrukcie agregovať nad väčšou oblasťou pre presnejšiu detekciu anomálií. MSGMS mapa je preto spracovaná pomocou konvolúcie s priemerovacím filtrom a odpočítaná od matice jednotiek na vygenerovanie mapy anomálií $G(I, I_r) \in [0,1]^{H \times W}$ (30):

$$G(I, I_r) = 1_{H \times W} - (MSGMS(I, I_r) * f_{s_f \times s_f}) \tag{30}$$

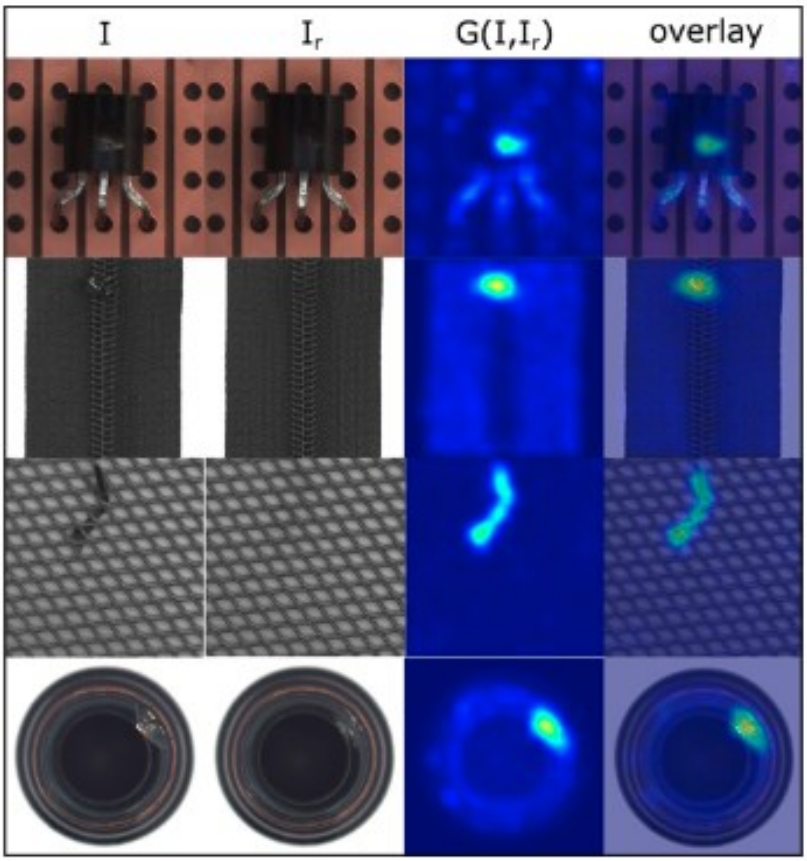
kde $f_{s_f \times s_f}$ je priemerovací filter veľkosti $s_f \times s_f$ použitý na vyhladenie, $*$ predstavuje operácia konvolúcie a $1_{H \times W}$ je matica jednotiek veľkosti $H \times W$. $MSGMS(I, I_r)$ je MSGMS mapa vygenerovaná z *I* a I_r. Vyhladenie zlepšuje robustnosť detekcie v prítomnosti vysokých hodnôt MSGMS v malých oblastiach, ktoré sú výsledkom nedokonalého rekonštrukčného procesu neanomálnych objektov alebo šumu pozadia namiesto skutočných anomálií. Počas inferencie sa jednotlivý obrázok maskuje a inpaintuje niekoľkokrát pre každé $k \in K$. Výstup metódy sa potom konštruuje ako priemer máp anomálií *G* generovaných pre každú rekonštrukciu obrázka pri každom *k* (31):

$$G_A(I, I_r) = \frac{1}{N_K} \sum_{k \in K} G(I, I_r)_k \tag{31}$$

kde $G(I, I_r)_k$ predstavuje mapu anomálií vygenerovanú s použitím hodnoty maskovania veľkosti anomálie k. Nakoniec sa skóre anomálie na úrovni obrázka $\epsilon(I, I_r)$ vypočíta z maximálnej hodnoty $G_A(I, I_r)$ ako (32):

$$\epsilon(I, I_r) = \max(G_A(I, I_r)) \tag{32}$$

kde I je vstupný obrázok a I_r je konečný rekonštruovaný obrázok, ktorý indikuje, či obrázok obsahuje anomáliu, alebo nie. Príklady anomálnych obrázkov I, ich rekonštrukcie I_r a lokálne vyhladené mapy anomálií $G(I, I_r)$ sú zobrazené na obrázku 11 [11].



Obrázok 11. Príklady rekonštrukcie a odhadu skóre anomálie [11]

2.2 DRÆM

Discriminatively trained reconstruction embedding model (DRÆM) sa skladá z rekonštrukčnej a diskriminačnej podsiete. Rekonštrukčná podsieť je trénovaná na implicitné detekovanie a rekonštruovanie anomálií sémanticky pravdepodobným obsahom bez anomálií, pričom neanomálne oblasti vstupného obrázka zostávajú nezmenené. Súčasne sa diskriminačná podsieť učí spoločný rekonštrukčný anomálny embedding a z konkatenovaného rekonštruovaného a pôvodného vzhľadu vytvára presné mapy

segmentácie anomálií. Anomálne tréningové príklady sú vytvorené konceptuálne jednoduchým procesom, ktorý simuluje anomálie na obrázkoch bez anomálií. Táto metóda generovania anomálií poskytuje ľubovoľné množstvo anomálnych vzoriek, ako aj pixelovo dokonalé mapy segmentácie anomálií, ktoré môžu byť použité na trénovanie navrhovanej metódy bez skutočných anomálnych vzoriek [12].

2.2.1 Rekonštrukčná podsieť

Rekonštrukčná podsieť je formulovaná ako architektúra encoder-decoder, ktorá prevádza lokálne vzory vstupného obrázka na vzory bližšie k distribúcii normálnych vzoriek. Sieť je trénovaná na rekonštrukciu pôvodného obrázka I z umelo poškodeného obrázka I_a získaného pomocou simulátora [12].

V metódach detekcie anomálií založených na rekonštrukcii sa často používa strata l_2, avšak tá predpokladá nezávislosť medzi susednými pixelmi, preto je navyše použitá stratová funkcia SSIM založená na patchoch, vypočítaná ako (33):

$$L_{SSIM}(I, I_r) = \frac{1}{N_p} \sum_{i=1}^{H} \sum_{j=1}^{W} 1 - SSIM(I, I_r)_{(i,j)} \tag{33}$$

kde H a W sú výška a šírka obrázka, I, N_p sa rovná počtu pixelov v obrázku a I_r je rekonštruovaný obrázok vytvorený sieťou. $SSIM(I, I_r)_{(i,j)}$ je hodnota SSIM pre patche obrázkov I a I_r, zameraná na súradnice (i, j). Strata rekonštrukcie je preto (34):

$$L_{rec}(I, I_r) = \lambda L_{SSIM}(I, I_r) + l_2(I, I_r) \tag{34}$$

kde λ je hyperparameter, ktorý slúži na vyváženie straty [12].

2.2.2 Diskriminačná podsieť

Diskriminačná podsieť používa architektúru podobnú U-net. Vstupom do podsiete je I_c, definované ako kanálová konkatenácia výstupu rekonštrukčnej podsiete I_r a vstupného obrázka I. Vďaka vlastnosti obnovovania normálnosti rekonštrukčnej podsiete sa spoločný vzhľad I a I_r v anomálnych obrázkoch výrazne líši, čo poskytuje informácie potrebné pre segmentáciu anomálií. V metódach detekcie anomálií, založených na rekonštrukcii, sa mapy anomálií získavajú pomocou funkcií podobnosti, ako napríklad SSIM, na porovnanie pôvodného obrázka s jeho rekonštrukciou, avšak je ťažké vytvoriť mieru podobnosti špecifickú pre detekciu povrchových anomálií. Naopak, diskriminačná podsieť sa učí vhodné meranie vzdialenosti automaticky. Výstupom siete je mapa skóre anomálií M_o

rovnakej veľkosti ako *I*. Na výstup diskriminačnej podsiete sa aplikuje Fokálna strata, pre zvýšenie robustnosti voči presnej segmentácii ťažkých príkladov [12].

Zohľadňujúc oba ciele, segmentáciu a rekonštrukciu, oboch podsietí sa celková strata použitá pri trénovaní DRÆM definuje ako kombinácia straty rekonštrukcie a straty segmentácie (35):

$$L(I, I_r, M_a, M) = L_{rec}(, I_r) + L_{seg}(M_a, M) \tag{35}$$

kde M_a a M sú referenčné riešenie a výstupné masky segmentácie anomálií [12].

2.2.3 Simulované generovanie anomálií

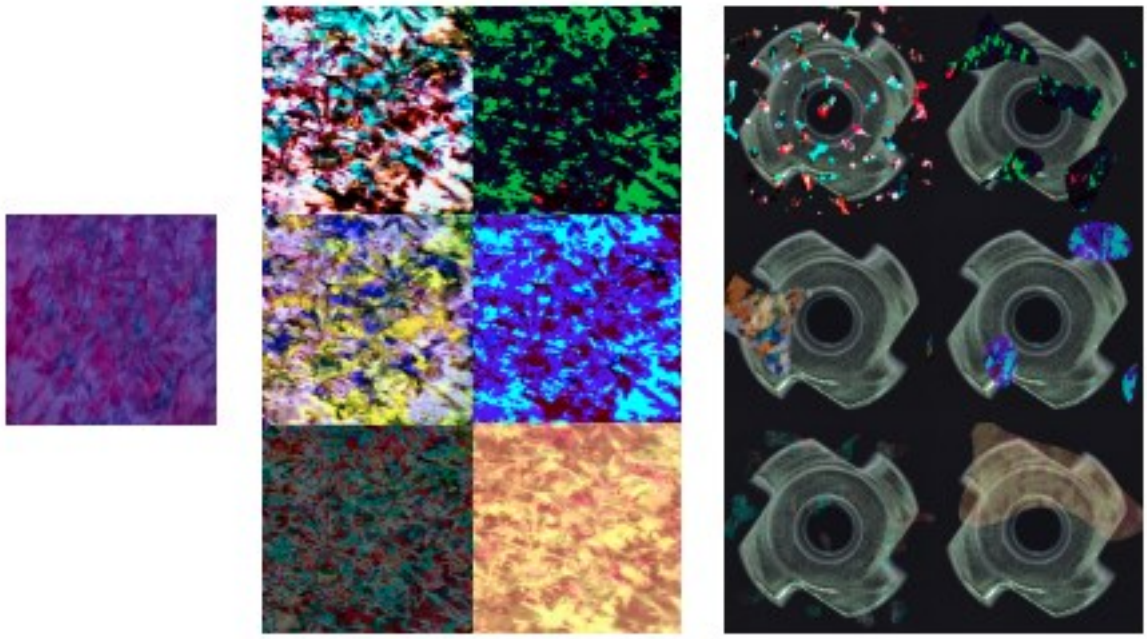
DRÆM nevyžaduje aby simulácie realisticky reflektovali skutočný výskyt anomálie v cieľovej oblasti, ale skôr generovali just-out-of-distribution výskyty, ktoré umožňujú naučiť sa vhodnú funkciu vzdialenosti na rozpoznanie anomálie podľa jej odchýlky od normy. Toto paradigma nasleduje navrhovaný generátor anomálií [12].

Obraz šumu je generovaný pomocou generátora Perlinovho šumu, ktorý zachytáva rôzne tvary anomálií a binarizuje sa pomocou náhodne vzorkovaného prahu na mapu anomálií M_a. Zdrojový obraz anomálnej textúry A sa vzorkuje z datasetu zdrojových obrázkov s anomáliami, ktorý nesúvisí s distribúciou vstupného obrázka. Následne sa aplikuje náhodné augmentačné vzorkovanie pomocou sady troch náhodných augmentačných funkcií, ktoré sa vyberú z tejto množiny: (posterize, sharpness, solarize, equalize, brightness change, color change, auto-contrast). Augmentácia je proces umelého rozšírenia tréningovej sady vytvorením upravených kópií existujúcich dát. Tento proces môže zahŕňať geometrické transformácie, náhodné mazanie niektorých častí obrázka alebo transformácie farebného priestoru. [17]. Augmentovaný obrázok textúry A je maskovaný pomocou mapy anomálií M_a a zlúčený s I, aby sa vytvorili anomálie, ktoré sú just-out-of-distribution a pomáhajú tak zúžiť rozhodovaciu hranicu v naučenej sieti. Augmentovaný tréningový obrázok I_a je preto definovaný ako (36):

$$I_a = \bar{M}_a \odot I + (1 - \beta)(M_a \odot I) + \beta(M_a \odot A) \tag{36}$$

kde $\bar{M}_a$ je inverzná hodnota M_a, $\odot$ je operácia elementárneho násobenia a β je parameter nepriehľadnosti pri zlúčení. Tento parameter sa vzorkuje rovnomerne z intervalu $\beta \in [0.1, 1.0]$. Náhodné zjednotenie a augmentácia umožňujú generovanie rôznorodých

obrázkov s anomáliami z jednej textúry, ako je možné vidieť na obrázku 12 [12].



Obrázok 12. Príklad augmentácie [12]

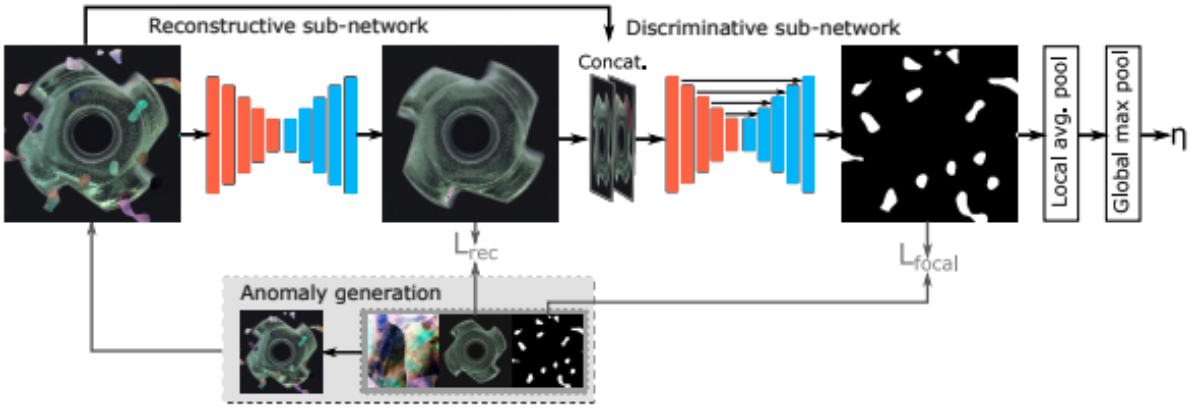
2.2.4 Lokalizácia a detekcia povrchových anomálií

Výstupom diskriminačnej podsiete je maska detekcie anomálií na úrovni pixelov M_o, ktorá môže byť interpretovaná priamo pre odhad skóre anomálie na úrovni obrázka, teda či je v obrázku prítomná anomália, alebo nie [12].

Najprv sa M_o vyhladí konvolučnou vrstvou priemerovacieho filtra na agregáciu informácií o lokálnej odozve anomálie. Konečné skóre anomálie η sa na úrovni obrázka vypočíta z maximálnej hodnoty vyhladenej mapy skóre anomálie ako (37):

$$\eta = \max\left(M_o \times f_{s_f} \times f_{s_f}\right) \tag{37}$$

kde $f_{s_f} \times f_{s_f}$ je priemerovací filter veľkosti $s_f \times s_f$ a $*$ je operácia konvolúcie [12].



Obrázok 13. Proces detekcie anomálií použitý modely DRÆM [12]

3 METÓDY ZALOŽENÉ NA ZAROVNANÍ

Tieto metódy pre detekciu a lokalizáciu anomálií sú založené na zarovnaní medzi anomálnym obrázkom a konštantným počtom podobných normálnych obrázkov. Sú založené na viacrozmerných príznakoch na identifikáciu anomálnych oblastí v obrázku [13]. Príkladom metódy založenej na zarovnaní je Semantic Pyramid Anomaly Detection (SPADE), ktorý nevyžaduje prakticky žiadny čas na trénovanie.

3.1 SPADE

Sub-Image Anomaly Detection with Deep Pyramid Correspondences model nevyžaduje dlhú tréningovú fázu a je rýchla a robustná. Metóda pozostáva z niekoľkých fáz:

- extrakcia obrazových príznakov pomocou vopred natrénovanej hlbokej neurónovej siete,
- získania najbližších susedou z najbližších K normálových obrázkov k cieľu,
- zarovnania pixelov s hlbokými pyramídovými korešpondenciami [13].

3.1.1 Extrakcia príznakov

Prvou fázou metódy je extrakcia silných príznakov na úrovni obrazu. Rovnaké príznaky sa neskôr používajú na zarovnanie obrazu na úrovni pixelov. Najčastejšie používanou možnosťou extrakcie príznakov je self-supervised učenie príznakov, čiže učenie príznakov od začiatku priamo na vstupných normálnych obrázkoch. Hoci je to atraktívna možnosť, nie je zrejmé, že príznaky získané na malých tréningových datasetoch budú dostatočné na to, aby slúžili ako vysokokvalitné miery podobnosti. Analýza ukazuje, že príznaky získané self-supervised učením zaostávajú za ResNet príznakmi natrénovanými na súbore ImageNet pre účely detekcie anomálií. Preto je v tejto metóde použitý extraktor príznakov ResNet vopred natrénovaný na ImageNet datasete. Ako príznaky na úrovni obrazu bol použitý vektor príznakov, získaný po globánom poolingu poslednej konvolučnej vrstvy. Ak je globálny extraktor príznakov označený ako *F*, extrahované príznaky pre daný obrázok x_i sú označené ako f_i (38):

$$f_i = F(x_i) \tag{38}$$

Príznaky pre všetky tréningové obrázky sú pri inicializácii vypočítané a uložené. Pri inferencii sú extrahované iba príznaky cieľového obrázka [13].

3.1.2 Získanie obrázkov metódou K najbližších susedov

Prvým krokom v tejto metóde je určenie, ktoré obrázky obsahujú anomálie použitím hlbokej detekcie anomálií, pomocou metódy najbližších susedov. Pre daný testovací obrázok *y* sa získa jeho *K* najbližších normálnych obrázkov z tréningovej množiny $N_k(F_y)$. Vzdialenosť sa meria pomocou euklidovskej metriky medzi reprezentáciami príznakov na úrovni obrazu ako (39):

$$d(y) = \frac{1}{K} \sum_{f \in N_K(f_y)} \| \mathrm{f} - f_y \|^2 \tag{39}$$

V tejto fáze sú obrázky označené ako normálne alebo anomálne. Pozitívna klasifikácia sa určuje overením, či je vzdialenosť k-NN väčšia ako prahová hodnota τ. Očakáva sa, že väčšina obrázkov je normálnych a iba niekoľko obrázkov je označených ako anomálne [13].

3.1.3 Detekcia anomálií v podobrazoch prostredníctvom zarovnania

Po tom, čo sú obrázky označené ako anomálne na úrovni obrazu, je cieľom lokalizovať a segmentovať pixely jednej alebo viacerých anomálií. V prípade, že bol obrázok označený ako anomálny chybne, je cieľom neoznačiť žiadne pixely [13].

Zarovnaním testovacieho obrázka k získanému normálnemu obrázku a následným nájdením rozdielov, by sa dali odhaliť anomálne pixely. Táto metóda má však niekoľko nedostatkov. Ak existuje viacero normálnych častí z ktorých sa môže prípadne objekt skladať, zarovnanie s normálnymi obrázkami môže zlyhať. Ďalej, v prípade malých datasetov alebo objektov s komplexnou variabilitou sa môže stať, že normálny tréningový obrázok, ktorý by bol podobný testovaciemu obrázku vo všetkých ohľadoch, nebude nikdy nájdený a vyvolal by falošné pozitívne detekcie. Tretím nedostatkom je, že výpočet rozdielu medzi obrázkami by bol veľmi citlivý na použitú stratovú funkciu [13].

Aby sa predišlo vyššie uvedeným problémom, je použitá metóda viac-obrázkových korešpondencií. Táto metóda extrahuje hlboké príznaky na každom mieste pixelu $p \in P$ z relevantných testovacích a normálnych tréningových obrázkov pomocou extraktoru príznakov $F(x_{i,}, p)$. Na všetkých pixelových miestach k najbližších susedov sa konštruuje galéria príznakov $G = \{F(x_1, p)|p \in P\}\}.. \cup \{F(x_K, p)|p \in P\}\}$. Hodnota anomálie na pixeli *p* v cieľovom obrázku *y* je určená priemernou vzdialenosťou medzi prvkami $F(y, p)$ a jeho *K* najbližšími prvkami z galérie *G*. Takto získaná hodnota anomálie pixela *p* môže

slúžiť na detekciu anomálií v cieľovom obraze. Hodnota anomálie pixela p v cieľovom obraze y je daná vzťahom (40):

$$d(y,p) = \frac{1}{K} \sum_{f \in N_K(F(y,p))} \| f - F(y,p) \|^2 \tag{40}$$

Pre daný prah θ je pixel označený ako anomálny, ak $d(y,p) > q$, teda ak nemôže nájsť blízko korešpondujúci pixel v K najbližších normálnych obrázkoch [13].

3.1.4 Párovanie príznakov pyramídy

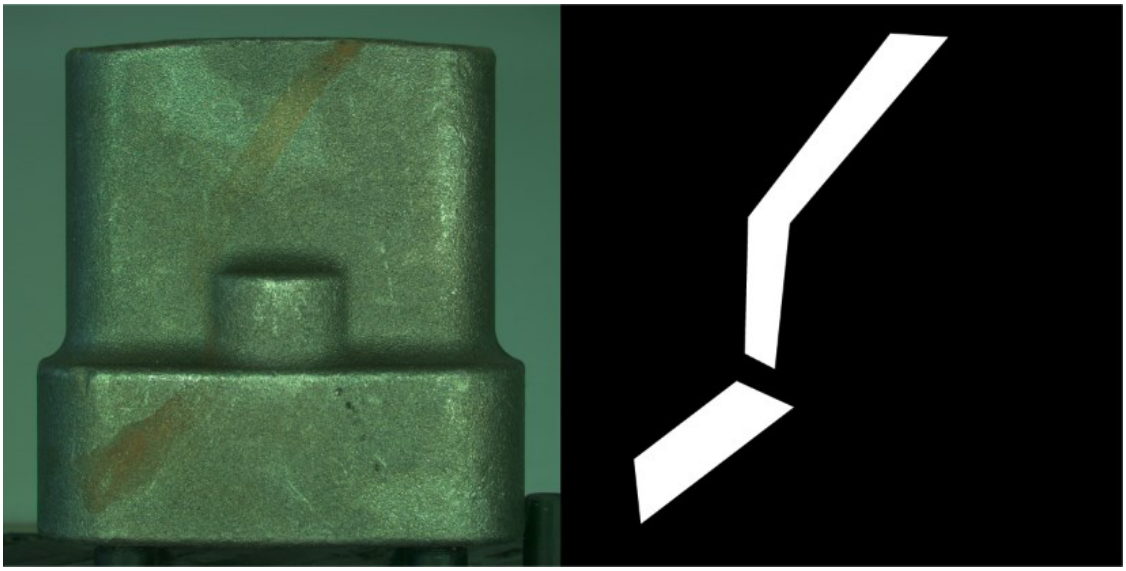
Zarovnanie pomocou hustých korešpondencií je efektívny spôsob určenia častí obrazu ktoré sú normálne, oproti tým, ktoré sú anomálne. Pre efektívne zarovnanie je nevyhnutné určiť príznaky pre párovanie. Podobne ako v predchádzajúcej fáze, metóda používa príznaky z vopred natrénovanej hlbokej konvolučnej neurónovej siete ResNet. Výsledkom siete ResNet je pyramída príznakov. Podobne ako v prípade obrázkových pyramíd, skoršie vrstvy (úrovne) vedú k príznakom s vyšším rozlíšením, ktoré kódujú menej kontextu. Neskoršie vrstvy kódujú príznaky s nižším rozlíšením, ktoré kódujú viac kontextu, ale s nižším priestorovým rozlíšením. Na vykonanie účinného zarovnania je každé miesto opísané pomocou príznakov z rôznych úrovní pyramídy príznakov. Konkrétne spojením príznakov z výstupu posledných M blokov. Príznaky kódujú jemné lokálne príznaky a globálny kontext. To modelu umožňuje nájsť korešpondencie medzi cieľovým obrázkom a $K \geq 1$ normálnymi obrázkami bez nutnosti explicitného zarovnania obrázkov, čo je technicky náročnejšie a menej spoľahlivé. Táto metóda je ľahko škálovateľná a jednoducho sa dá nasadiť v praxi [13].

II. PRAKTICKÁ ČASŤ

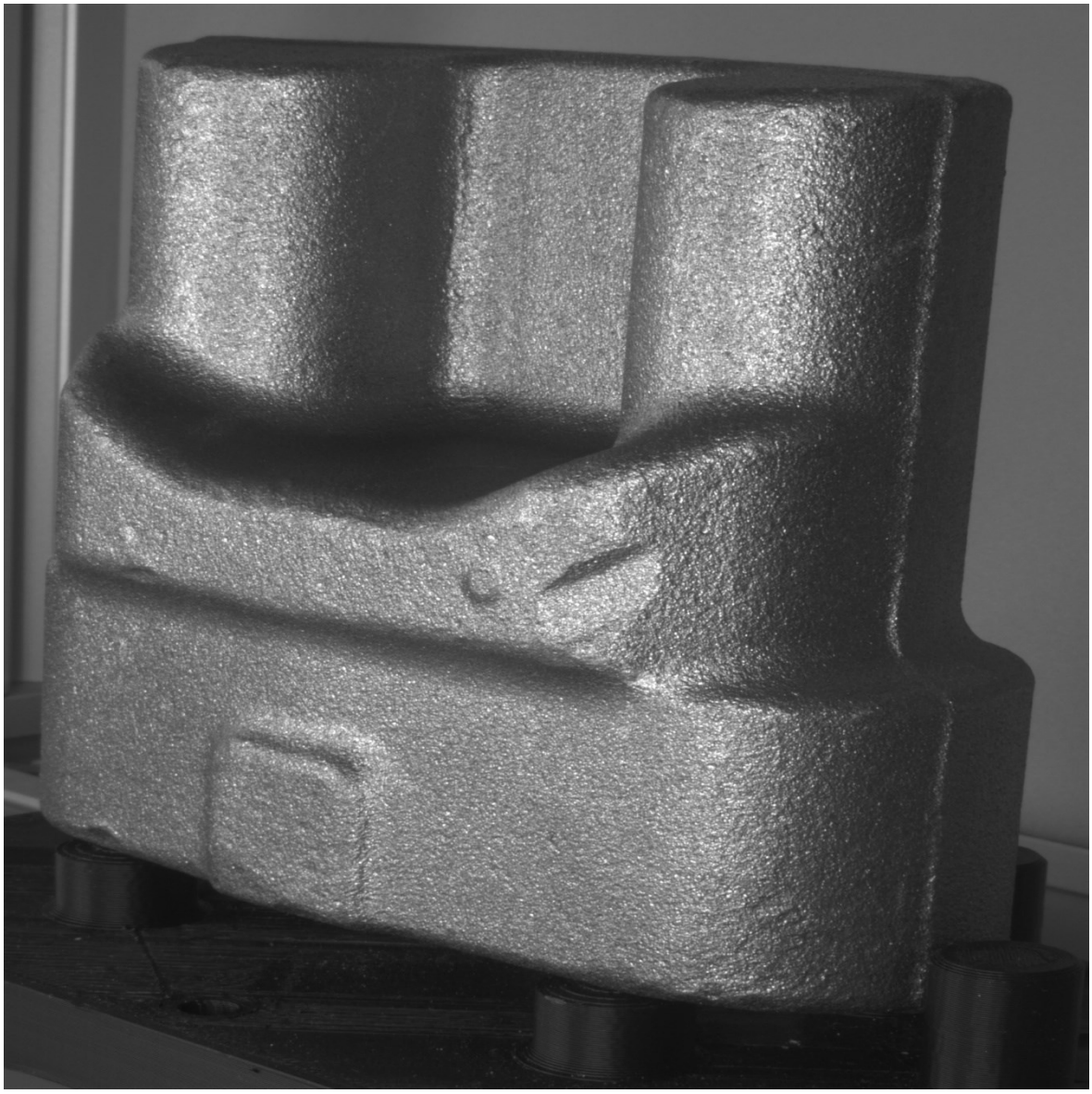
4 PRÍPRAVA DATASETU

Príprava datasetu je kľúčovým krokom v strojovom učení a zahŕňa niekoľko postupov, ktoré pomáhajú vytvoriť súbor údajov vhodný pre strojové učenie [14]. Medzi tieto postupy patrí zhromažďovanie dát, kategorizácia dát, predspracovanie, transformácia, vývoj a výber funkcií a rozdelenie údajov na tréningové a testovacie sady [15].

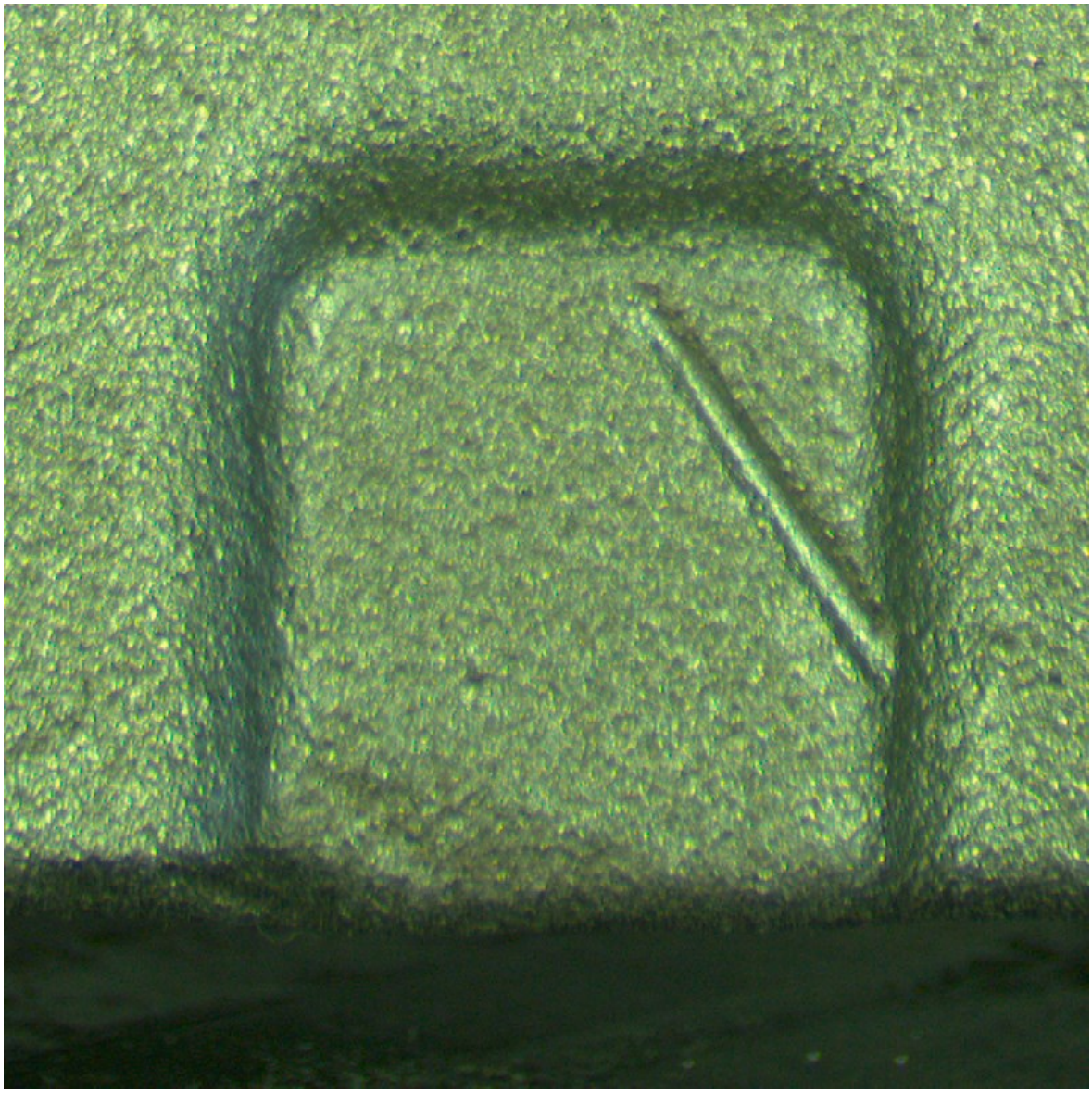
Pre túto prácu bol vybraný dataset, ktorý pozostáva z obrázkov výkovkov (výrobkov získaných kovaním) z obrobne CV Machining s.r.o. Tento dataset obsahuje 340 obrázkov a bol pripravený tak, aby mal podobnú štruktúru ako dataset MVTec, čo uľahčuje jeho implementáciu v modeloch. Každá kategória obsahuje zložky „ground_truth“, „test“ a „train“. Modely tak môžu efektívne získať a separovať dáta podľa typu anomálií. Tréningová sada datasetu „train“ obsahuje iba obrázky bez defektov. Testovacia sada „test“ obsahuje obrázky s rôznymi typmi chýb, ako napríklad praskliny, hrdza, alebo škrabance, ale aj obrázky bez chýb. Obrázky výkovkov sú vyhotovené z rôznych strán (side1 a side2) a uhlov (viewA, viewB, viewC), takže kategória side0_viewA obsahuje obrázky výkovkov, ktoré sú vyhotovené z prednej strany. Rozlíšenie obrázkov je v rozmedzí od 807x807 pixelov do 2905x2905 pixelov. Každý obrázok, ktorý obsahoval nejaký defekt, bol manuálne anotovaný označením anomálnych oblastí, čím vznikla ground truth pre každý anomálny obrázok. Ground truth je dôležitým prvkom pre trénovanie a overovanie výkonnosti algoritmov a slúži ako referenčné dáta. Na základe týchto referenčných dát sú modely trénované a potom sú použité na detekciu anomálií v testovacích dátach. Príklady dát z datasetu sú na obrázkoch 14-16.



Obrázok 14. Výkovok s anomáliou v podobe hrdze z kategórie side0_viewA (vľavo) a ground truth obrázka (vpravo)



Obrázok 15. Výkovok s anomáliou z kategórie side1_viewB



Obrázok 16. Výkovok s anomáliou z kategórie zoom_front

5 APLIKÁCIE PRE DETEKCIU A LOKALIZÁCIU ANOMÁLIÍ V OBRAZE

Na platforme GitHub[1] je dostupných niekoľko repozitárov obsahujúcich aplikácie pre detekciu a lokalizáciu anomálií v obraze. Avšak veľa z týchto repozitárov sú len nedokončené projekty. Okrem toho niektoré projekty využívajú staršie verzie knižníc. Tieto knižnice môžu obsahovať funkcie, ktoré nie sú dostupné v novších verziách.

Všetky aplikácie v nasledujúcich podkapitolách sú implementované v jazyku Python. V aplikáciách zameraných na neurónové siete sa využívajú epochy a batche. Pretože nie je možné spracovať celý dataset naraz, je rozdelený na menšie časti nazývané batche. Batch_size definuje počet tréningových vzoriek, ktoré sú prezentované v jenom batch-i. Jedna epocha predstavuje prechod celého datasetu cez neurónovú sieť oboma smermi práve raz.

Pre hodnotenie kvality segmentácie pixelov v obrázkoch je v tejto práci použitá metrika pixel AUROC. Táto metrika je založená na koncepte Area Under the Receiver Operating Characteristic (AUROC) a umožňuje kvantitatívne vyhodnotiť presnosť segmentácie pixelov. Čím vyššia je hodnota AUROC, tým lepšie sú segmentačné schopnosti modelu. Každému pixelu je priradená predikovaná hodnota a porovnáva sa s referenčnou hodnotou. Na základe týchto binárnych hodnôt sa vytvorí Receiver Operating Characteristic (ROC) krivka. Pixel AUROC sa vypočíta ako plocha pod touto krivkou. Okrem toho, pre hodnotenie kvality celých obrázkov alebo skupín pixelov, je použitá metrika image AUROC. Tá rozširuje AUROC na úroveň celého obrázka a poskytuje hodnotu pre celkovú segmentáciu. Tieto metriky umožňujú kvantitatívne vyhodnotiť presnosť segmentácie pixelov a celých obrázkov, a tým umožňujú objektívne zhodnotenie výkonu jednotlivých modelov.

Funkčné aplikácie boli testované a ich výsledky popísané v nasledujúcich podkapitolách.

[1] Webová služba podporujúca vývoj softwaru za pomoci verzovacieho nástroja Git

Aplikácie boli spustené na zariadení s:

- Procesor: Intel Core i7-7700HQ,
- RAM: 16GB,
- Grafická karta: NVIDIA GeForce GTX 150 Ti.

5.1 Anomalib

Anomalib [26] je knižnica, ktorej cieľom je zhromažďovanie najmodernejších algoritmov na detekciu anomálií. Poskytuje niekoľko pripravených implementácií algoritmov detekcie anomálií, ako aj niekoľko nástrojov, ktoré uľahčujú vývoj a implementáciu vlastných modelov. Táto knižnica sa výrazne zameriava na detekciu anomálií v obraze. Umožňuje použiť verejné, ale aj vlastné datasety [16].

5.1.1 Inštalácia

Požiadavky pre spustenie aplikácie:

- Git
- Python

Prvým krokom pri inštalácii je vytvorenie virtuálneho prostredia a následne aktivácia tohto prostredia pomocou nasledujúcich príkazov:

- `python -m venv anomalib_env`
- `anomalib_env\Scripts\activate`

Následne je potrebné naklonovať repozitár z GitHubu.

- `git clone ttps://github.com/openvinotoolkit/anomalib.git`
- `cd anomalib`

Posledným krokom prípravy aplikácie je inštalácia balíčkov. Tento krok nainštaluje OpenVIVO a ďalšie závislosti, ako napríklad Jupyter Lab, Kornia, Tensorboard, Pandas a TorchMetrics:

- `python -m pip install --upgrade pip wheel setuptools`
- `pip install -e .[full]`

5.1.2 Spustenie aplikácie

Pred spustením tréningu každého modelu, z knižnice Anomalib, musel byť upravený jeho konfiguračný súbor. Najdôležitejšími úpravami boli zmena názvu datasetu, cesta k datasetu a kategória. Následne bol konfiguračný súbor uložený v kmeňovom priečinku aplikácie. Obrázok 17 zobrazuje konfiguračný súbor *padim_config.yaml* pre model PaDiM. Spustenie modelu s pripraveným datasetom bolo vykonané príkazom:

- `python tools/train.py --config custom_padim.yaml`

Dataset bol uložený do zložky „datasets“ v kmeňovom priečinku. Výstupom aplikácie sú hodnoty pixel AUROC a image AUROC, ich grafy a vizualizácia segmentácie anomálie. Tieto výstupy sú uložené v zložke „results“, ak nebolo špecifikované inak pri úprave konfiguračného súboru.

```yaml
dataset:
  name: custom_dataset
  format: mvtec
  path: ./datasets/custom_dataset
  category: side0_viewA
  task: segmentation # classification or segmentation
  mask: null #optional
  normalization: imagenet # data distribution to which the images will be normalized: [none, imagenet]
  extensions: null
  split_ratio: 0.2 # ratio of the normal images that will be used to create a test split
  image_size: 256
  train_batch_size: 32
  eval_batch_size: 32
  num_workers: 8
  transform_config:
    train: null
    eval: null
  validation_split_mode: true
  test_split_mode: from_dir # options: [from_dir, synthetic]
  test_split_ratio: 0.2 # fraction of train images held out testing (usage depends on test_split_mode)
  val_split_mode: same_as_test # options: [same_as_test, from_test, synthetic]
  val_split_ratio: 0.5 # fraction of train/test images held out for validation (usage depends on
val_split_mode)
  tiling:
    apply: false
    tile_size: null
    stride: null
    remove_border_count: 0
    use_random_tiling: False
    random_tile_count: 16

model:
  name: padim
...
```

Obrázok 17. Konfiguračný súbor pre model PaDiM

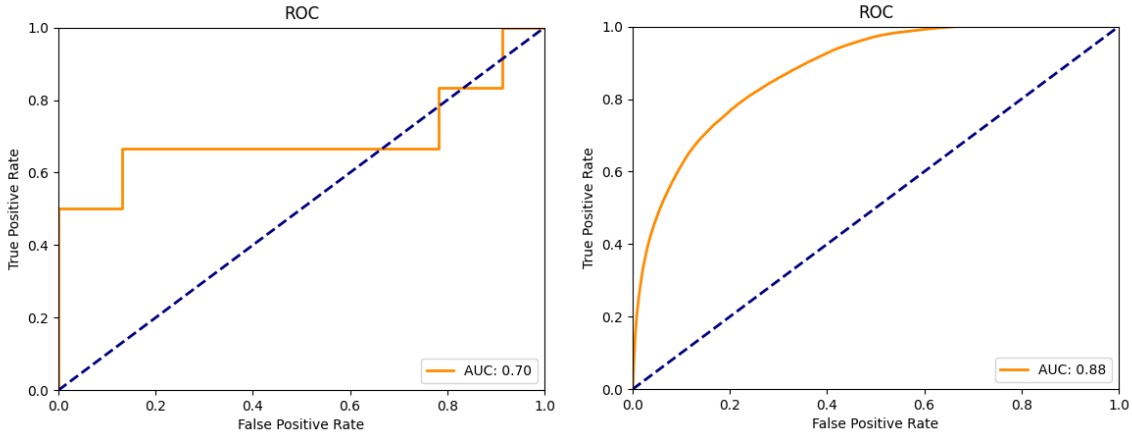
5.1.3 PaDiM

V tabuľke 1 je možné vidieť, že model dosiahol najlepší výsledok v segmentácii na úrovni obrázka v kategórii zoom_front, konkrétne 86,7%. Najvyšší výsledok segmentácie na úrovni

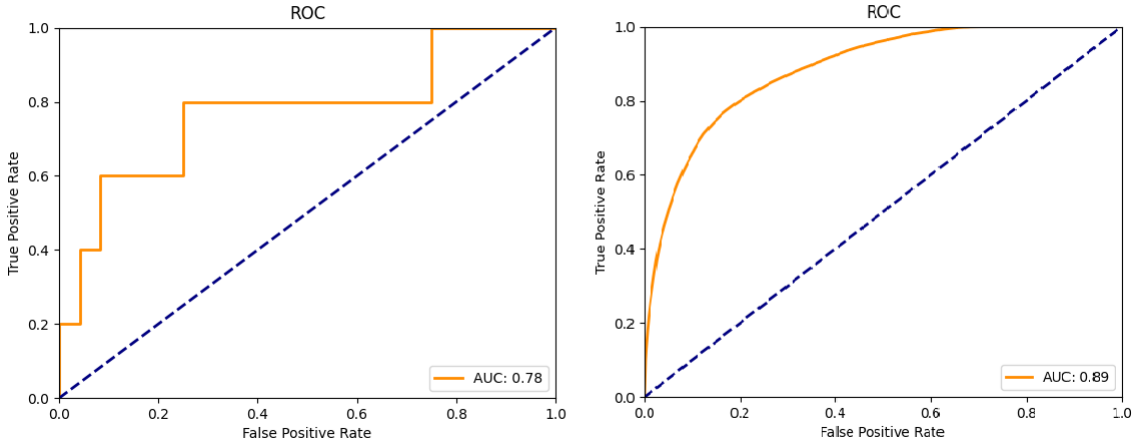
pixelov bol dosiahnutý v kategórii side1_viewA, a to 92,6%. Najnižšie skóre segmentácie na úrovni obrázka bolo v kategórii side0_viewA s hodnotou 61%. Pri segmentácii na úrovni pixelov bolo najnižšie skóre 87,2% v kategórii side1_viewB. Priemerný výsledok modelu pri segmentácii na úrovni obrazu bol 75,3% a na úrovni pixelov 89%. Obrázky 18 až 24 predstavujú image a pixel ROC grafy, ktoré zobrazujú výkonnosť modelu PaDiM.

Tabuľka 1. Výsledky modelu PaDiM

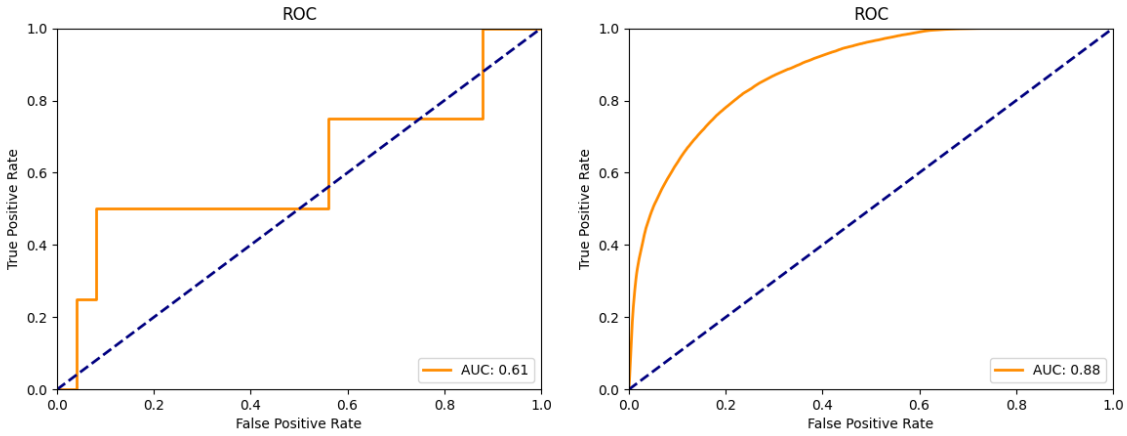
	side0_viewA	side0_viewB	side0_viewC	side1_viewA	side1_viewB	side1_viewC	zoom_front	priemer
Image AUROC	69,6%	77,5%	61%	85,3%	69,7%	77,9%	86,7%	75,3%
Pixel AUROC	87,8%	88,5%	88,2%	92,6%	87,2%	91%	88%	89%



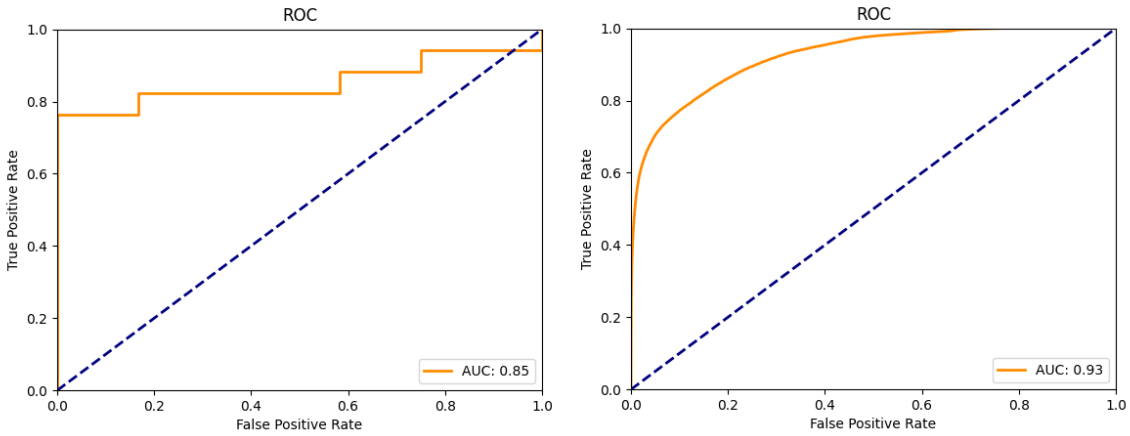
Obrázok 18. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewA



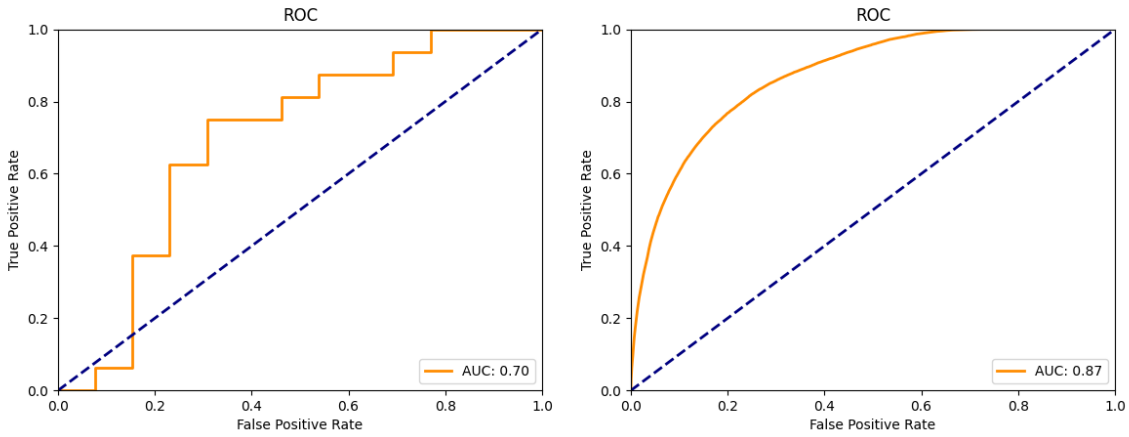
Obrázok 19. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewB



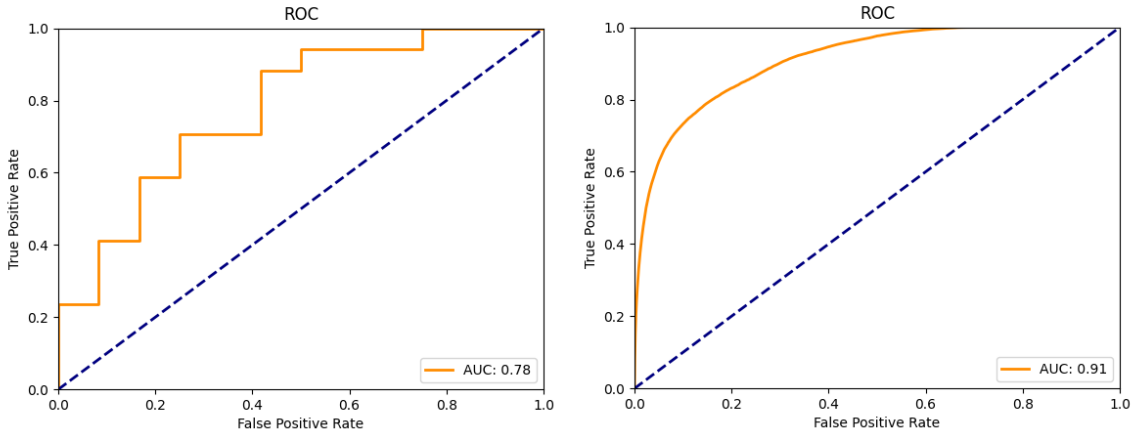
Obrázok 20. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewC



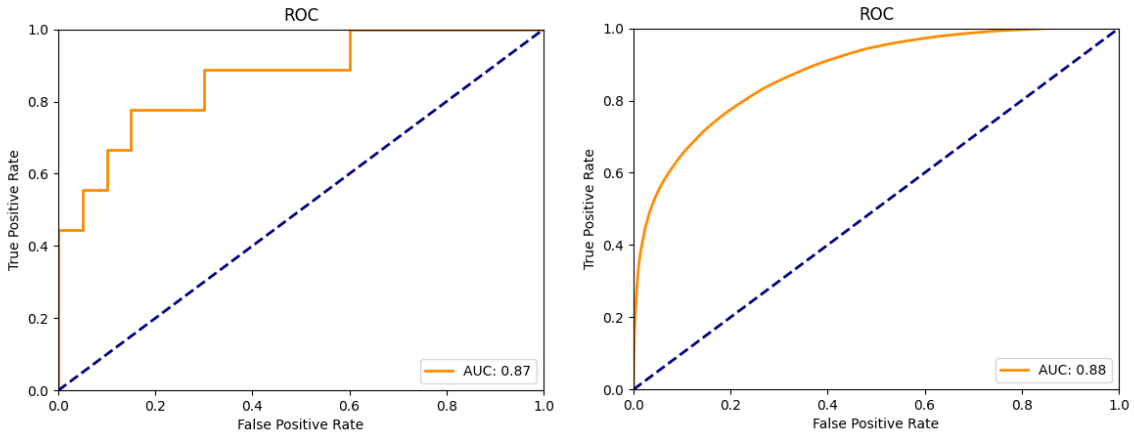
Obrázok 21. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewA



Obrázok 22. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewB

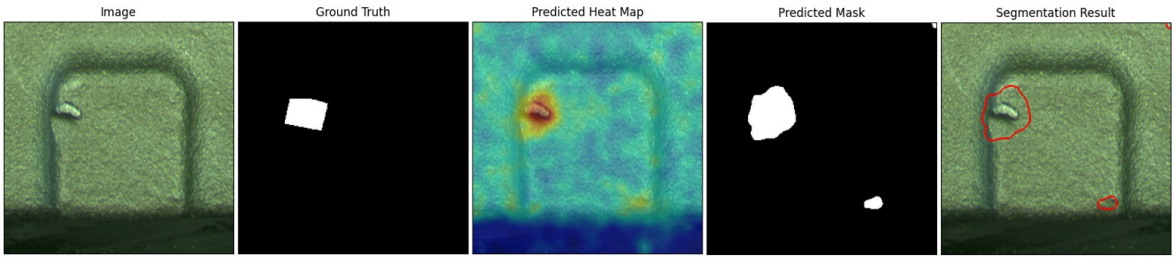


Obrázok 23. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewC



Obrázok 24. PaDiM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie zoom_front

Na obrázku 25 je príklad správnej detekcie anomálie a jej lokalizácie. Vľavo je obrázok z kategórie zoom_front, vedľa ktorého sa nachádza ground truth obrázok s anotáciou anomálnej oblasti. Uprostred je predikovaná heat mapa získaná z modelu. Vedľa heat mapy je zobrazená, model vytvorená, predikovaná maska, ktorá označuje oblasti obsahujúce anomálie. Vpravo sa nachádza výsledok segmentácie.



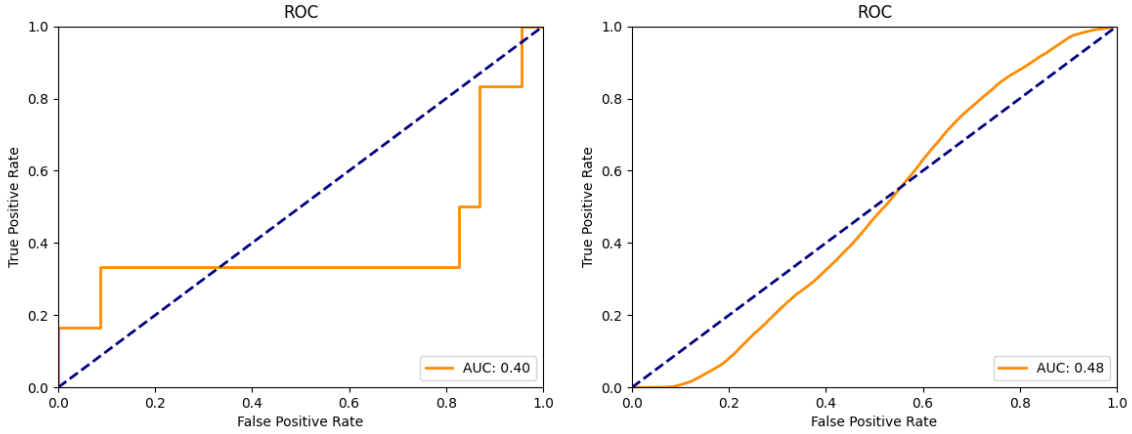
Obrázok 25. Príklad detekcie a lokalizácie anomálie modelom PaDiM

5.1.4 RDOCE

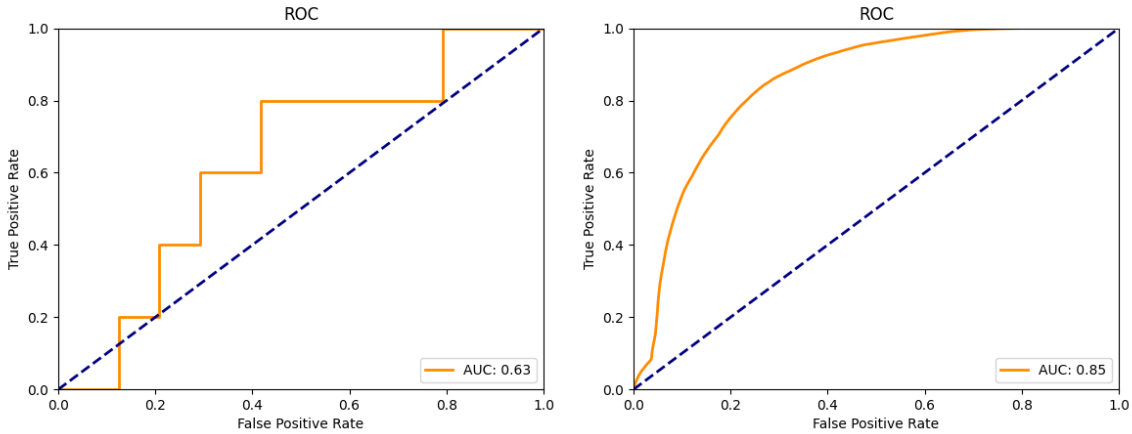
Z tabuľky 2 vyplýva, že priemerný výsledok pri segmentácii na úrovni obrázka modelu RDOCE bol 53,3% a na úrovni pixelov 64,9%. Najvyšší výsledok image AUROC bol dosiahnutý v kategórii side1_viewB. V kategórii zoom_front bol dosiahnutý najvyšší výsledok pixel AUROC 92,2%. Najnižší výsledok image AUROC dosiahol model v kategórii side0_viewA, konkrétne 39,9%. Pri segmentácii na úrovni pixelov bol najnižší výsledok 42,2%, a to v kategórii side1_viewB. Obrázky 26 až 32 predstavujú image a pixel ROC grafy, ktoré zobrazujú výkonnosť modelu RDOCE.

Tabuľka 2. Výsledky modelu RDOCE

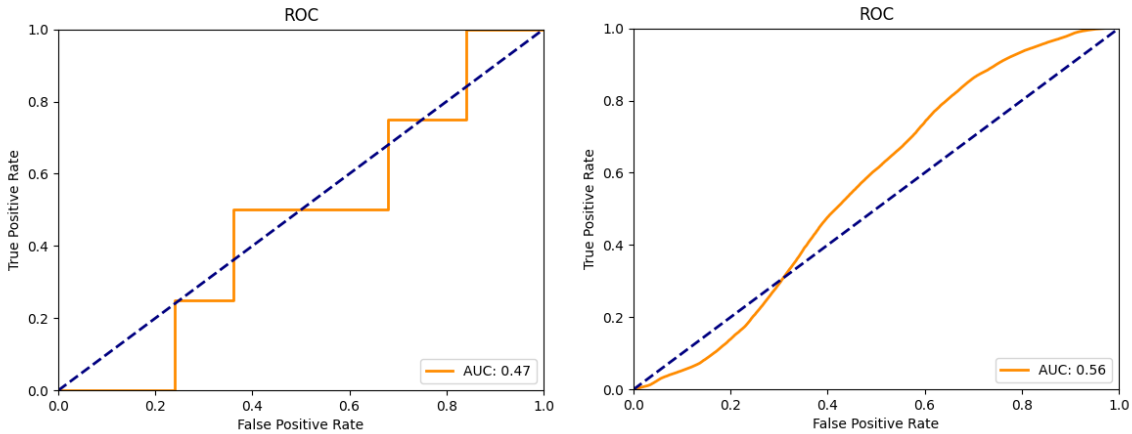
	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front	priemer
Image AUROC	39,9%	63,3%	46%	63,7%	68,8%	43,1%	48,3%	53,3%
Pixel AUROC	48,4%	85,2%	55,9%	42,2%	40,4%	90,3%	92,2%	64,9%



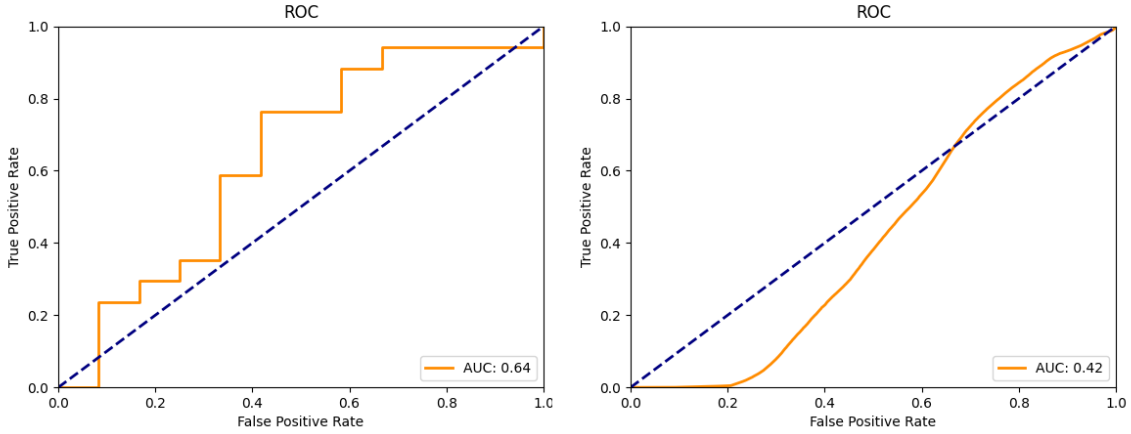
Obrázok 26. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewA



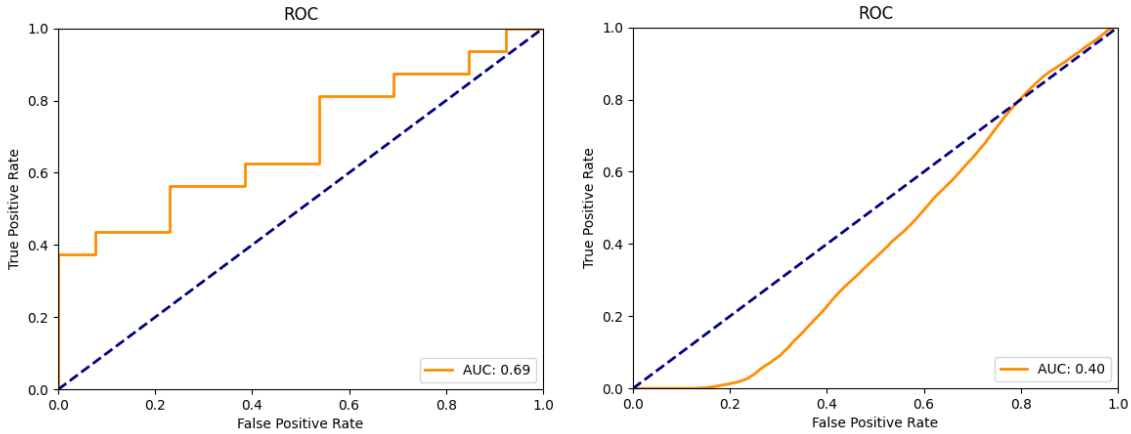
Obrázok 27. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewB



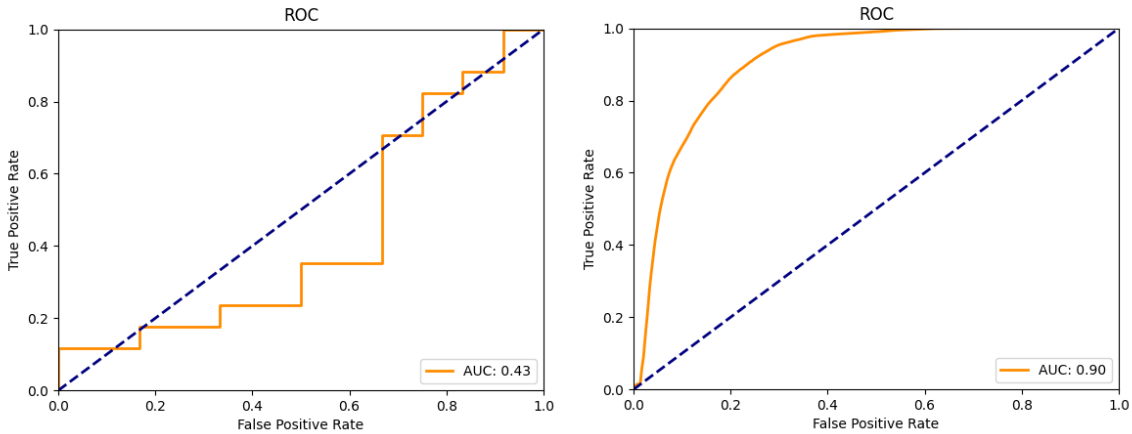
Obrázok 28. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewC



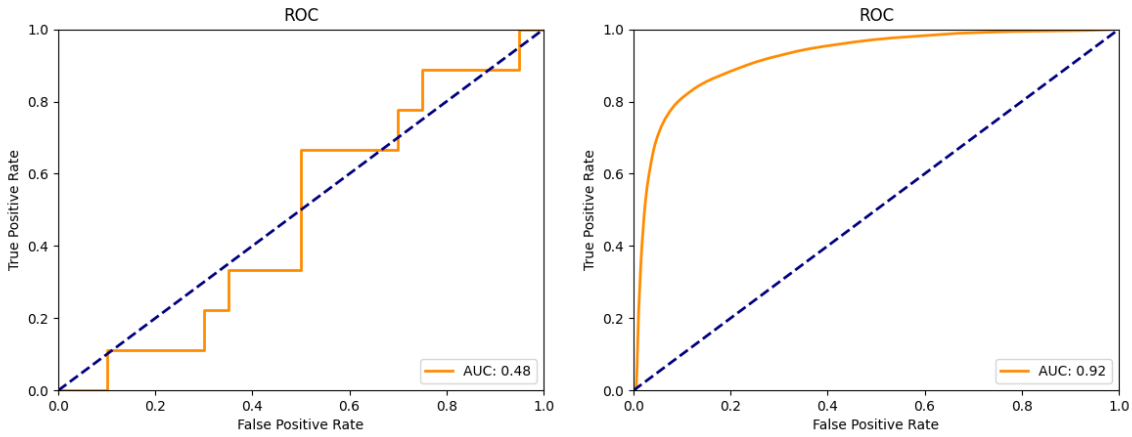
Obrázok 29. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewA



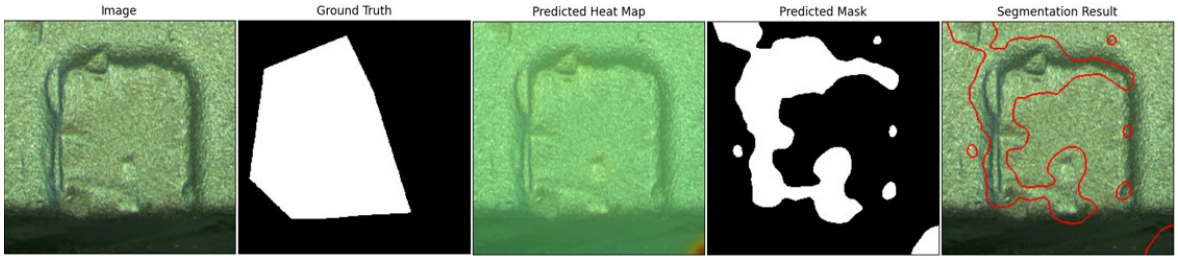
Obrázok 30. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewB



Obrázok 31. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewC



Obrázok 32. RDOCE image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie zoom_front



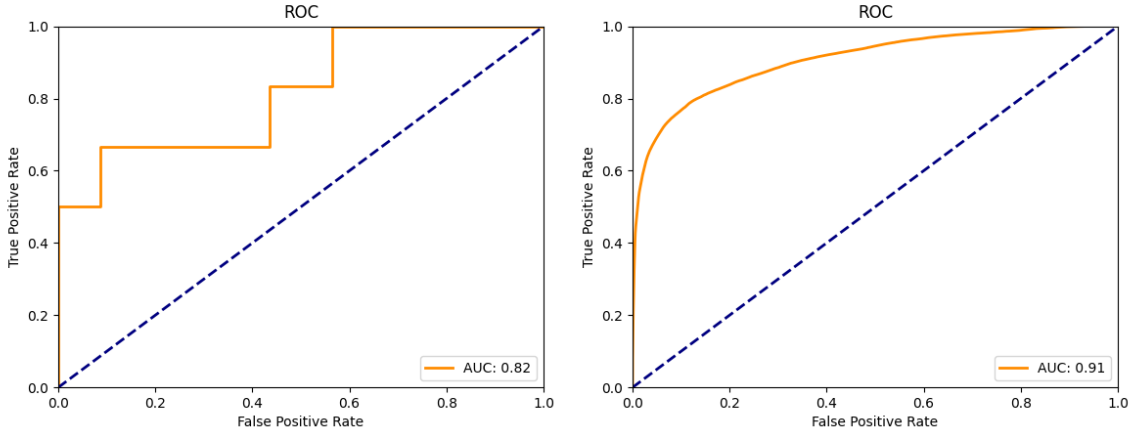
Obrázok 33. Príklad detekcie a lokalizácie anomálie modelom RDOCE

5.1.5 PatchCore

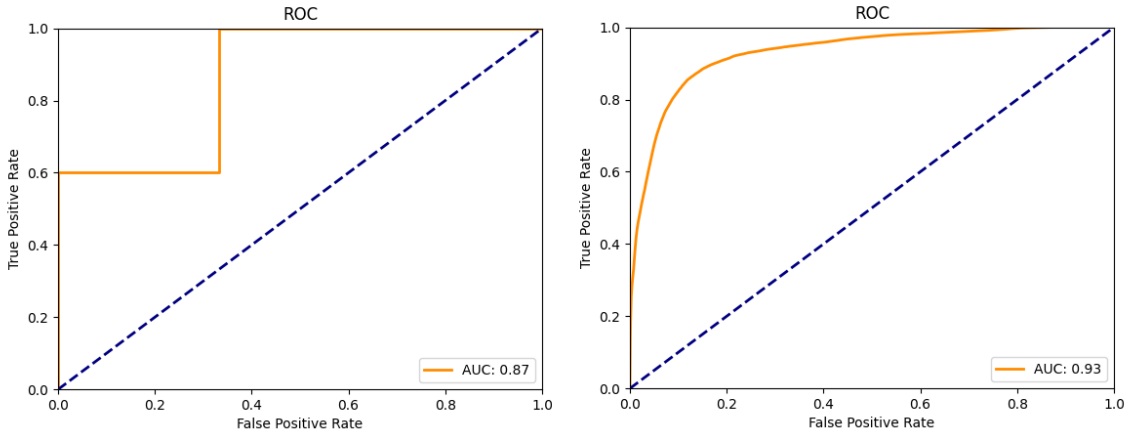
Ako je možné vidieť v tabuľke 3, priemerný výsledok segmentácie na úrovni obrázka bol 87,9% a na úrovni pixelov 93,3%. Najlepšie výsledky segmentácií boli dosiahnuté v kategórii zoom_front. Pre segmentáciu na úrovni obrázka to bolo 98,9% a pre segmentáciu na úrovni pixelov 96%. Naopak, najnižšie skóre 79,4% dosiahol model pri segmentácii na úrovni obrázka v kategórii side1_viewA a pri segmentácii na úrovni pixelov 90,7% v kategórii side0_viewA. Obrázky 34 až 40 ukazujú image a pixel ROC grafy, ktoré zobrazujú výkonnosť modelu PatchCore.

Tabuľka 3. Výsledky modelu PatchCore

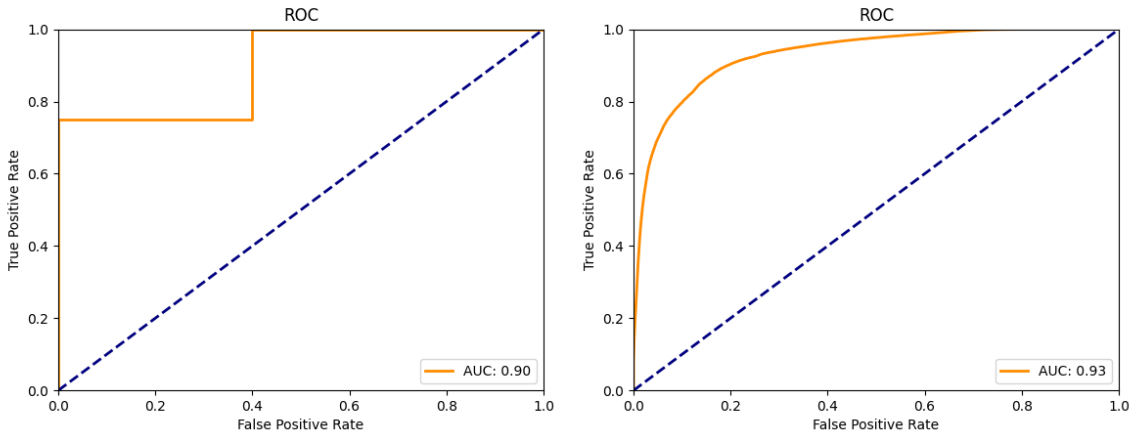
	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front	priemer
Image AUROC	81,8%	86,7%	90%	95,1%	83,7%	79,4%	98,9%	87,9%
Pixel AUROC	90,7%	93,1%	93,1%	95,2%	92,9%	91,9%	96%	93,3%



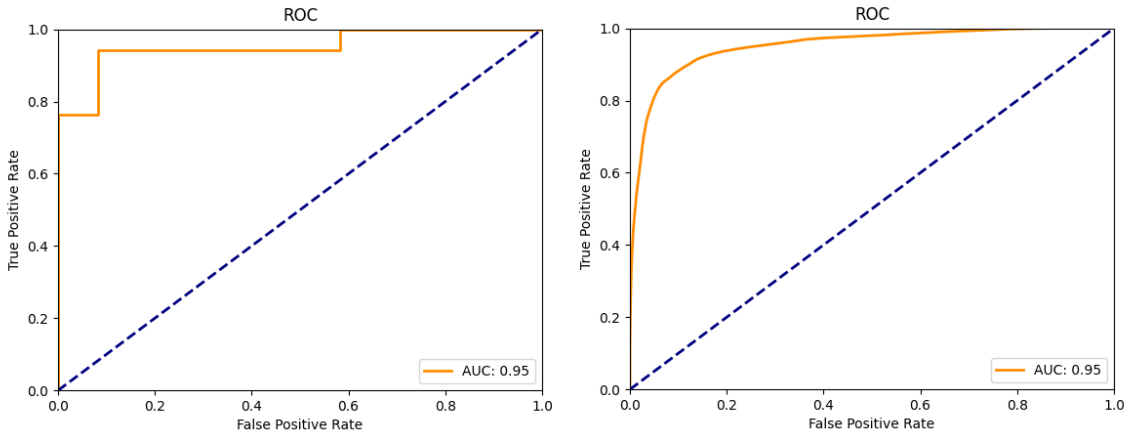
Obrázok 34. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewA



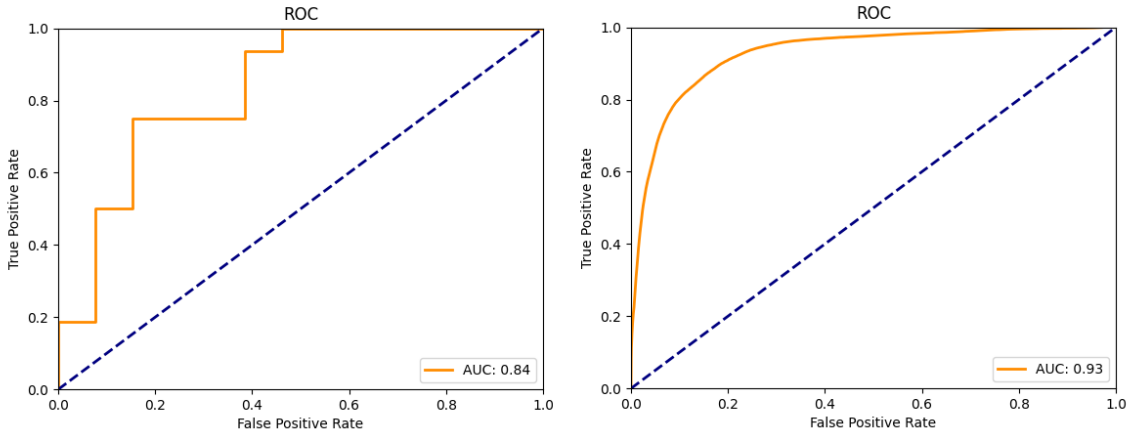
Obrázok 35. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewB



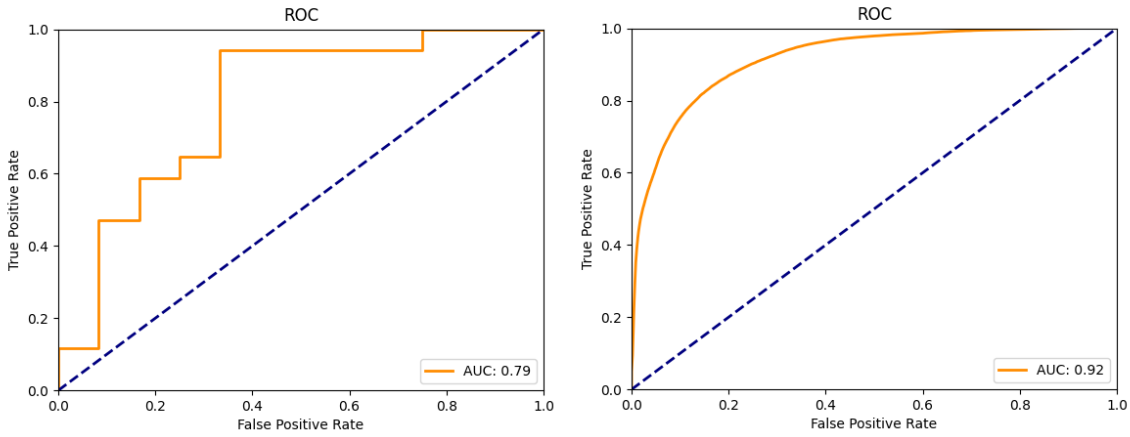
Obrázok 36. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewC



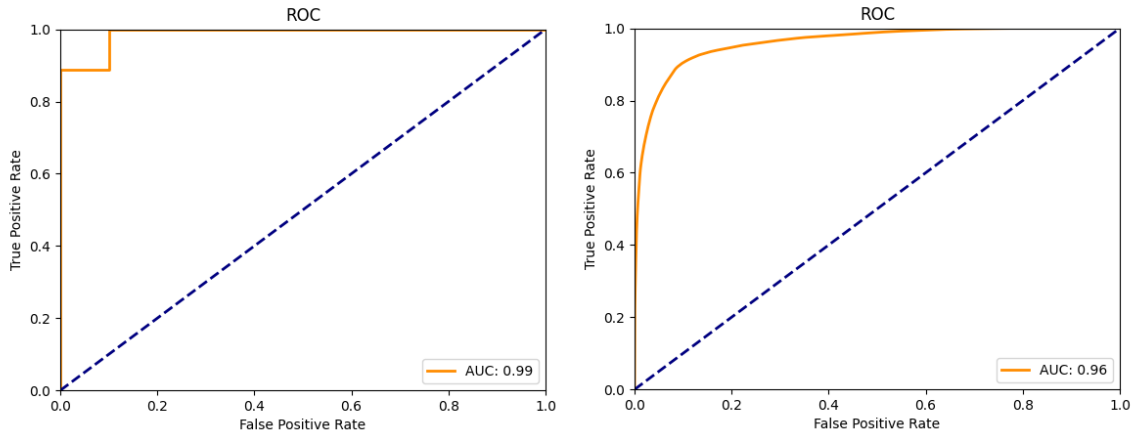
Obrázok 37. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewA



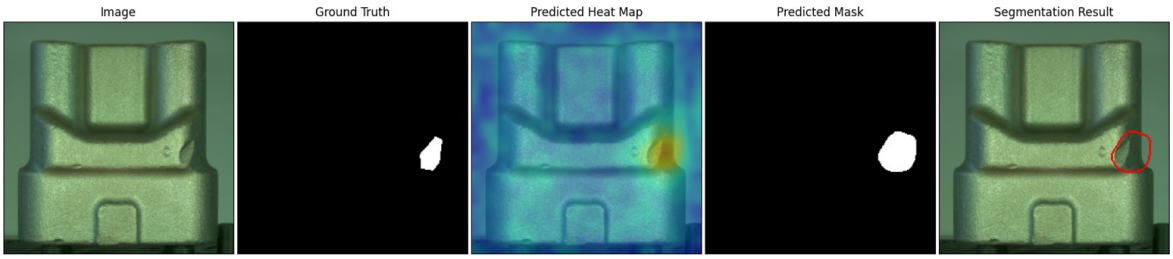
Obrázok 38. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewB



Obrázok 39. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewC



Obrázok 40. PatchCore image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie zoom_front



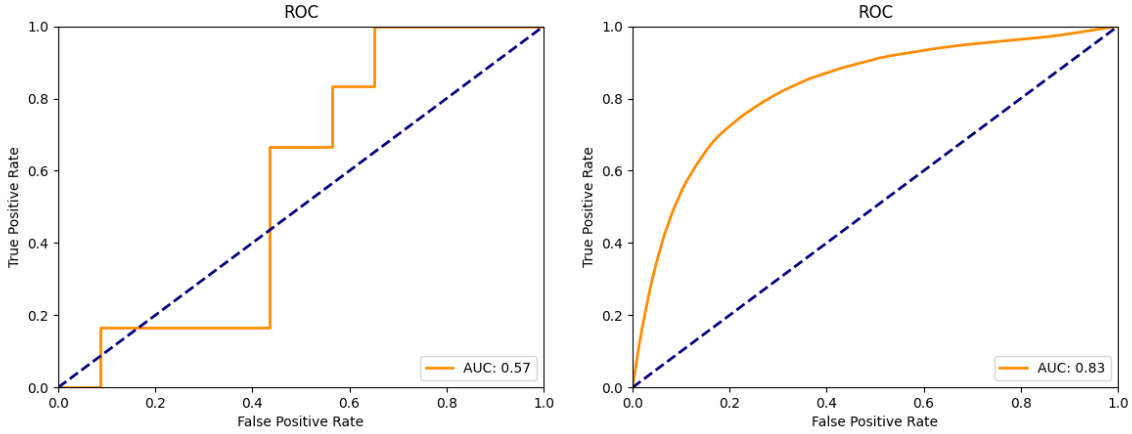
Obrázok 41. Príklad správnej detekcie anomálie a jej lokalizácie modelom PatchCore

5.1.6 DRÆM

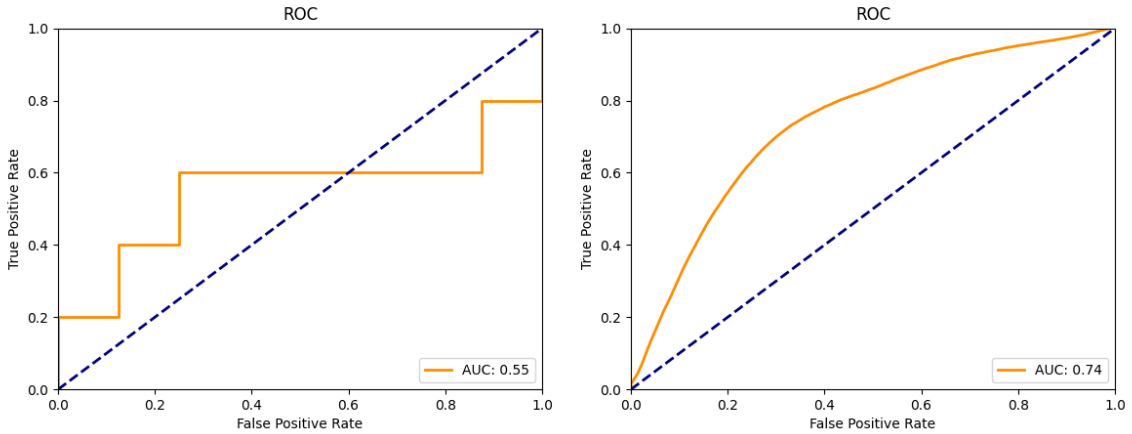
Tabuľka 4 zobrazuje výsledky modelu DRÆM. Najlepší výsledok pri segmentácii na úrovni obrazu bol 70,6% a bol dosiahnutý v kategórii zoom_front. Naopak, najhorší výsledok segmentácie na úrovni obrazu dosiahol model v kategórii side1_viewB, konkrétne 39,4%. Segmentácia na pixelovej úrovni bola najúspešnejšia v kategórii side0_viewA so skóre 82,6%. Najnižšie skóre segmentácie na úrovni pixelov 66% bolo dosiahnuté v kategórii zoom_front. Priemerné skóre výsledku je 57,1% pri segmentácii na úrovni obrázka a 74,1% pri segmentácii na úrovni pixelov.

Tabuľka 4. Výsledky modelu DRÆM

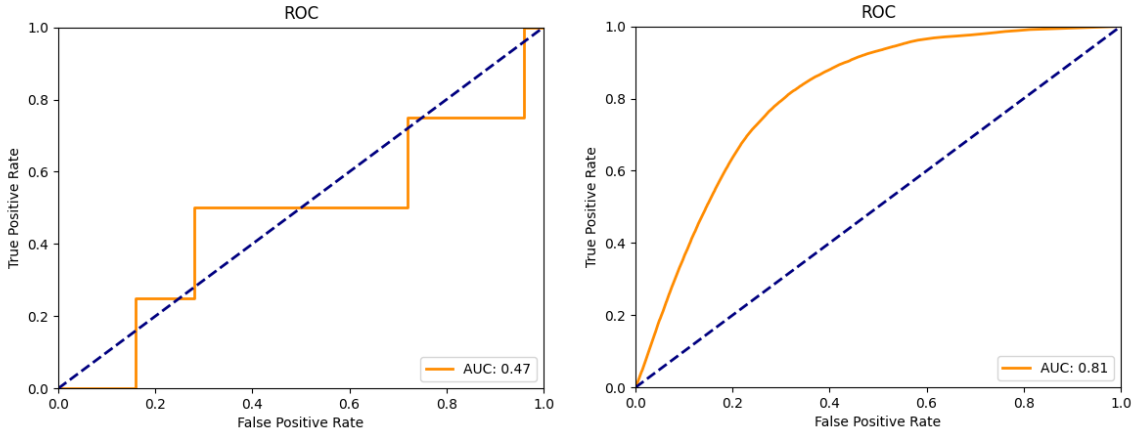
	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front	priemer
Image AUROC	56,5%	55%	47%	69,2%	39,4%	62,3%	70,6%	57,1%
Pixel AUROC	82,6%	74,3%	80,6%	74,8%	66,1%	74,4%	66%	74,1%



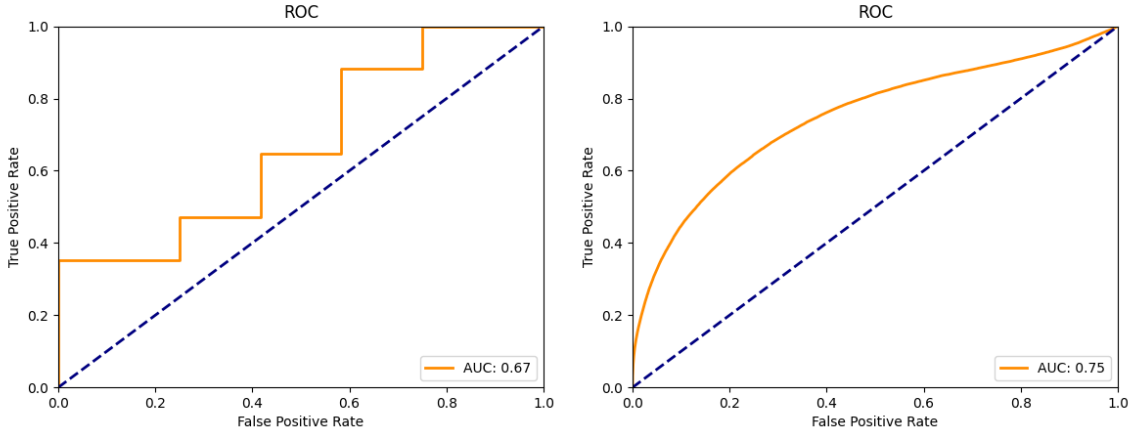
Obrázok 42. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewA



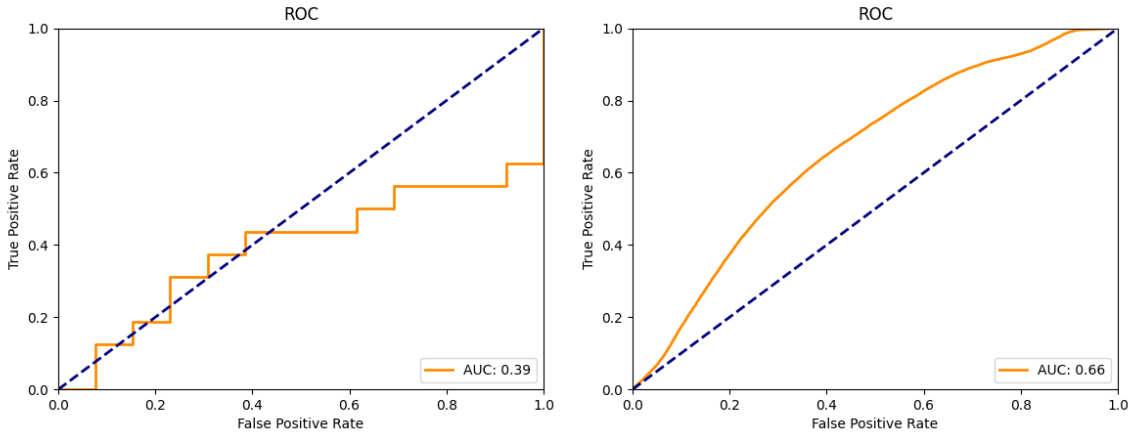
Obrázok 43. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewB



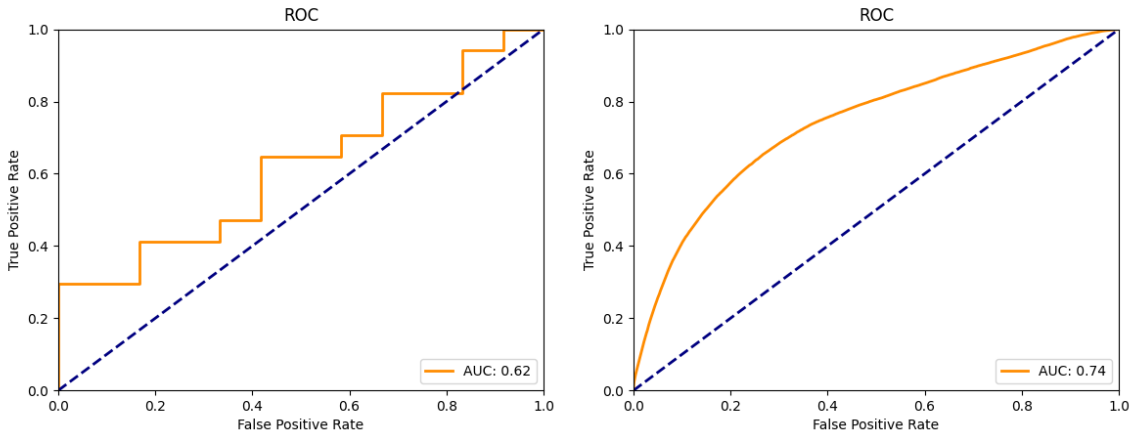
Obrázok 44. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side0_viewC



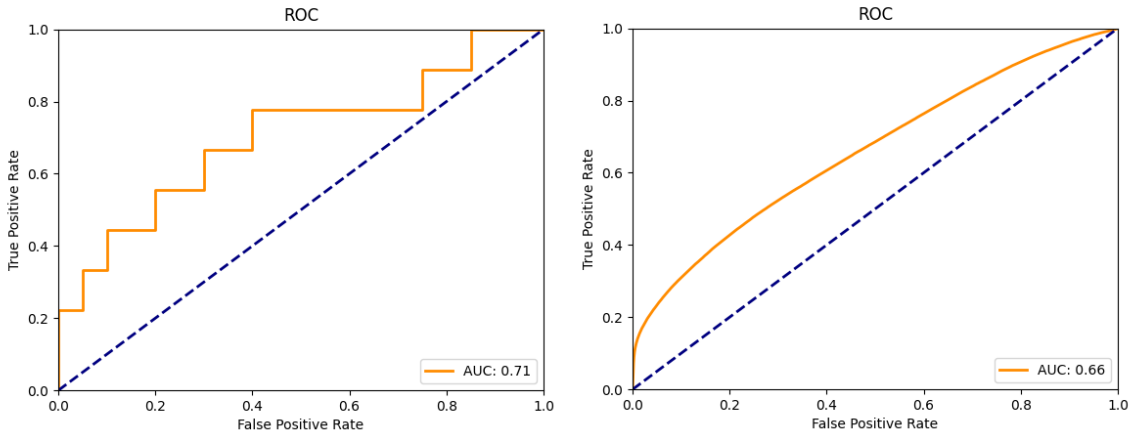
Obrázok 45. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewA



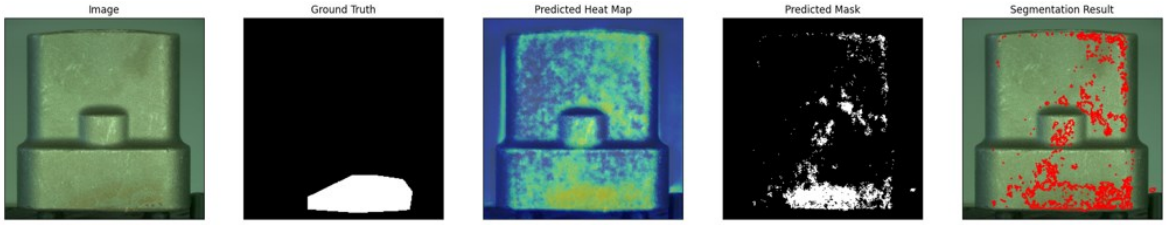
Obrázok 46. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewB



Obrázok 47. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie side1_viewC



Obrázok 48. DRÆM image ROC krivka (vľavo) a pixel ROC krivka (vpravo) kategórie zoom_front



Obrázok 49. Príklad detekcie a lokalizácie anomálie modelom DRÆM

5.2 RIAD

Reconstruction-by-inpainting-for-visual-anomaly-detection [29] je neoficiálnou implementáciou modelu RIAD. Požiadavky pre spustenie sú nasledovné:

- PyTorch 1.5
- Sklearn, matplotlib
- Kornia

Dataset bol uložený do zložky „datasets“ v kmeňovom priečinku. Následne bolo potrebné upraviť súbor *mvtec.py*, konkrétne názvy tried a cesta k datasetu, ako je možné vidieť na obrázku 50.

```
import os
# import tarfile
from PIL import Image
# import urllib.request

import torch
from torch.utils.data import Dataset
from torchvision import transforms

CLASS_NAMES = [
    'side0_viewA', 'side0_viewB', 'side0_viewC', 'side1_viewA', 'side1_viewB', 'side1_viewC', 'zoom_front'
]


class MVTecDataset(Dataset):
    def __init__(self,
                 dataset_path='../data/custom_dataset',
                 class_name='side0_viewA',
                 is_train=True,
                 resize=256,
                 ):
        assert class_name in CLASS_NAMES, 'class_name: {}, should be in {}'.format(class_name, CLASS_NAMES)
        self.dataset_path = dataset_path
        self.class_name = class_name
        self.is_train = is_train
        self.resize = resize
...
```

Obrázok 50. Upravený *mvtec.py* súbor modelu RIAD

Po úprave súboru *mvtec.py* nasledovalo trénovanie modelu. Napríklad pre kategóriu side0_viewA bolo trénovanie spustené príkazom:

- `python train.py --obj side0_viewA --data_path .\datasets\custom_dataset`

Po úspešnom natrénovaní modelu nasledovalo testovanie datasetu. Testovanie bolo spustené príkazom:

- `python test.py --obj side0_viewA --data_path .\datasets\custom_dataset --checpoint_dir .\mvtec\side0_viewA\seed_4450`

Parameter checkpoint_dir označuje zložku, kde boli uložené váhy modelu po fáze tréningu.

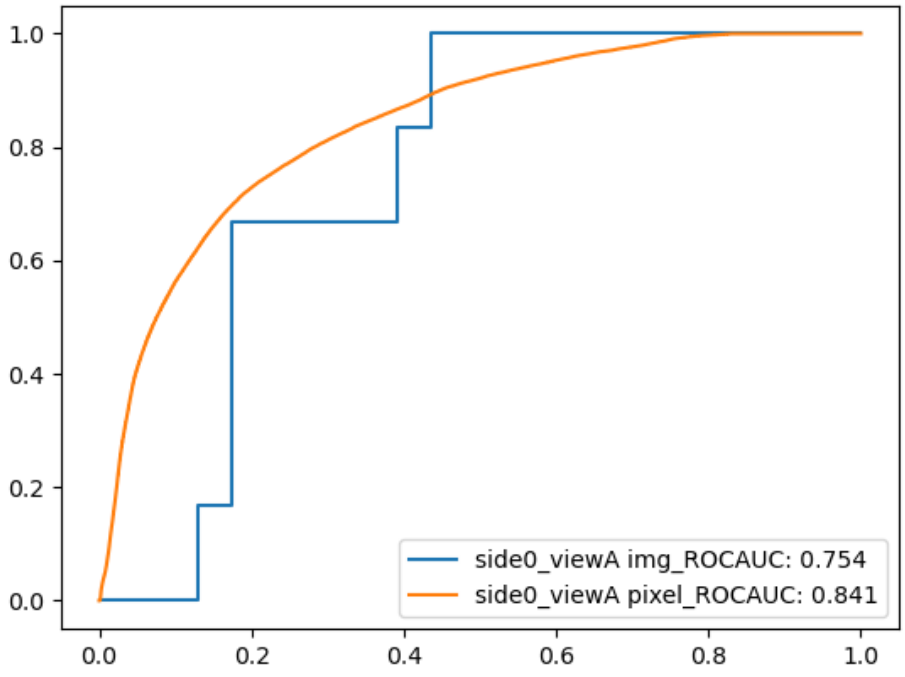
Výstupom tejto implementácie modelu RIAD sú rekonštruované obrázky, vizualizácia lokalizácie anomálie, graf obsahujúci image ROC a pixel ROC krivku a log súbor s informáciami o priebehu trénovania modelu.

V tabuľke 5 je možné vidieť, že model dosiahol najlepší výsledok v segmentácii na úrovni obrázka v kategórii sode0_viewC, konkrétne 85%. Najvyšší výsledok segmentácie na úrovni pixelov bol dosiahnutý v kategórii side1_viewA, a to 88,7%. Najnižšie skóre segmentácie na úrovni obrázka bolo v kategórii side1_viewA s hodnotou 62,3%. Pri segmentácii na úrovni pixelov bolo najnižšie skóre 66% v kategórii zoom_front. Priemerný výsledok

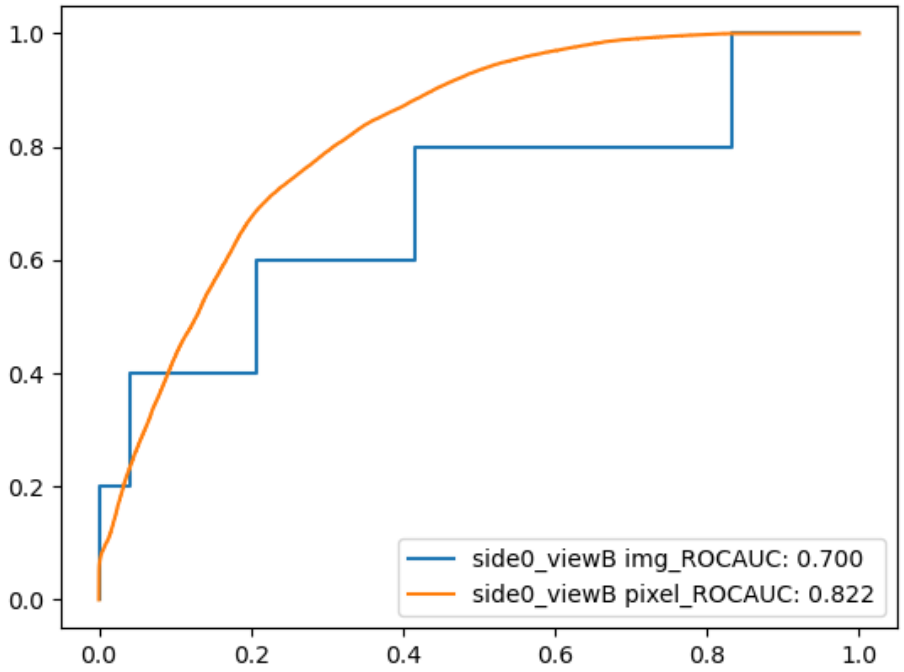
modelu pri segmentácii na úrovni obrazu bol 71,8% a na úrovni pixelov 81,9%. Obrázky 51 až 57 predstavujú image a pixel ROC grafy, ktoré zobrazujú výkonnosť modelu RIAD.

Tabuľka 5. Výsledky modelu RIAD

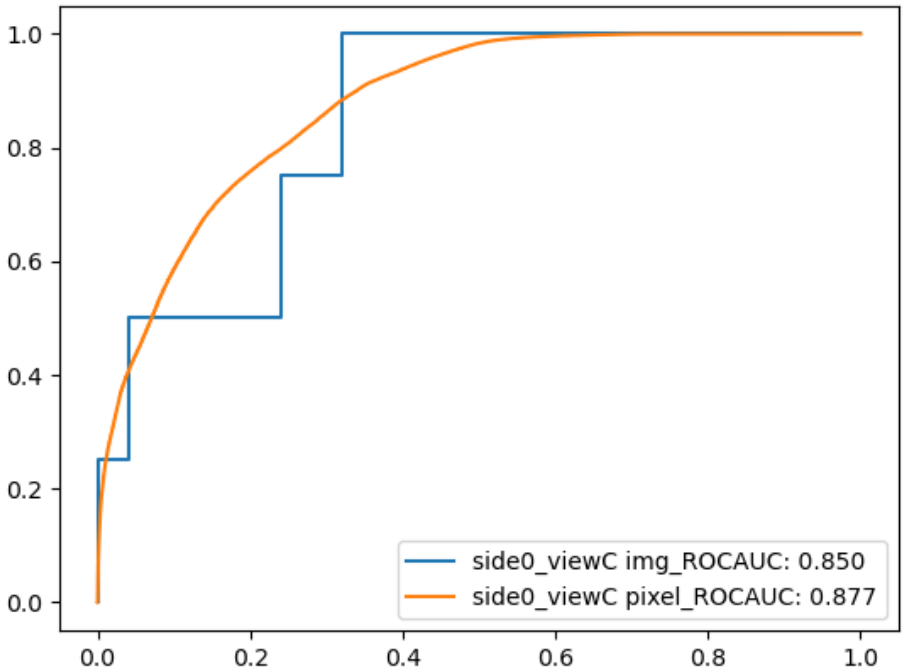
	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front	priemer
Image AUROC	75,4%	73,3%	85%	62,3%	68,3%	67,6%	70,6%	71,8%
Pixel AUROC	84,1%	82,2%	87,7%	88,7%	78.8%	86%	66%	81,9%



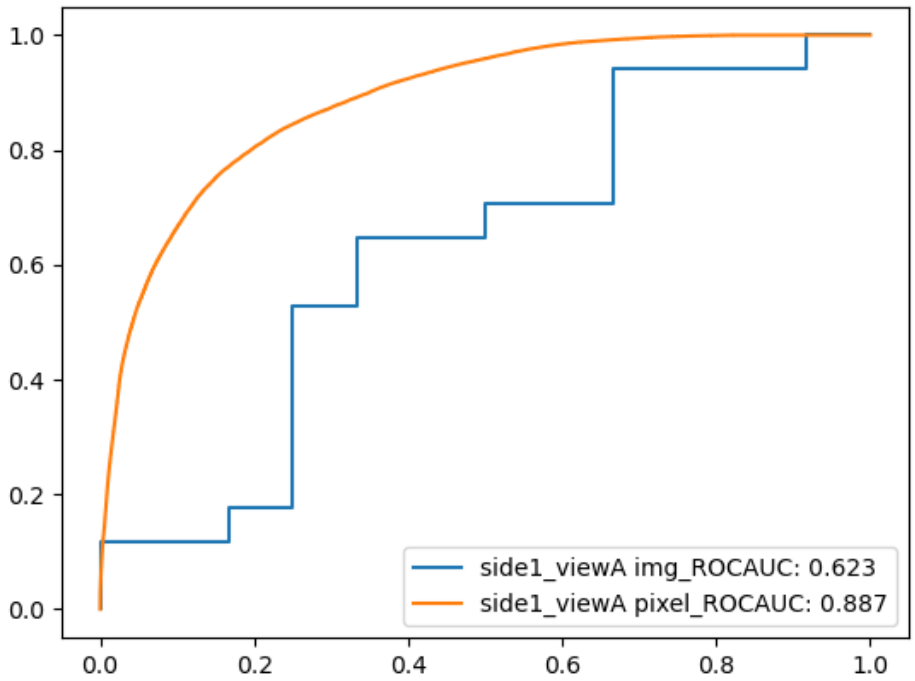
Obrázok 51. RIAD image ROC krivka a pixel ROC krivka kategórie side0_viewA



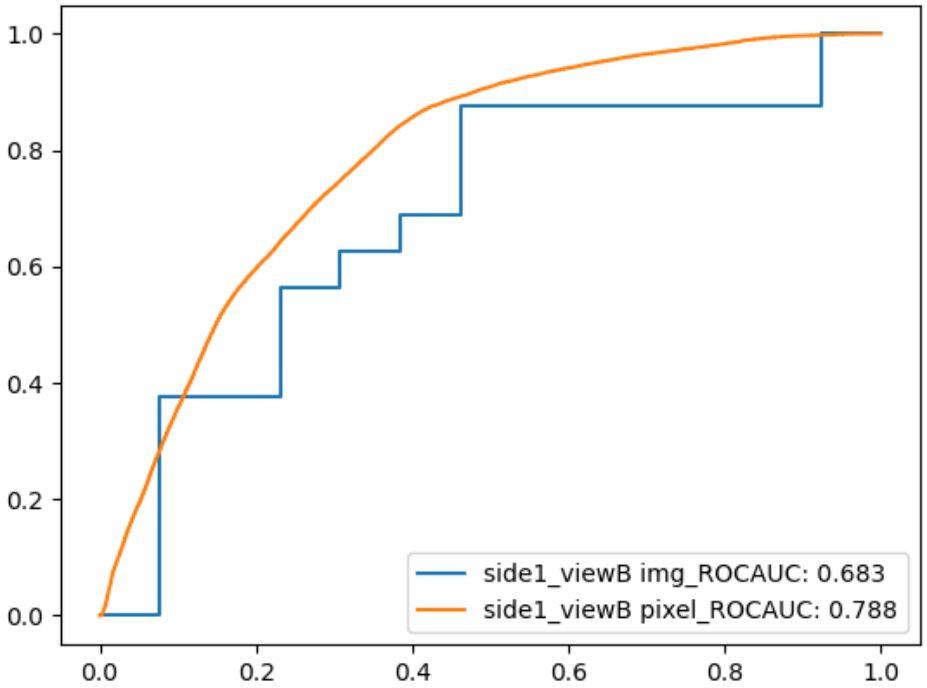
Obrázok 52. RIAD image ROC krivka a pixel ROC krivka kategórie side0_viewB



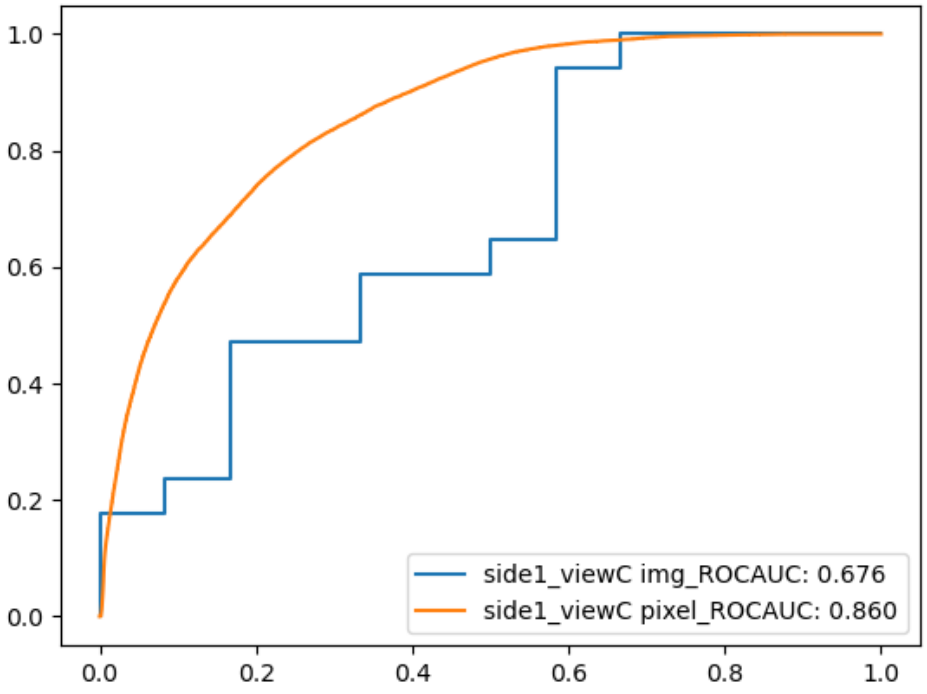
Obrázok 53. RIAD image ROC krivka a pixel ROC krivka kategórie side0_viewC



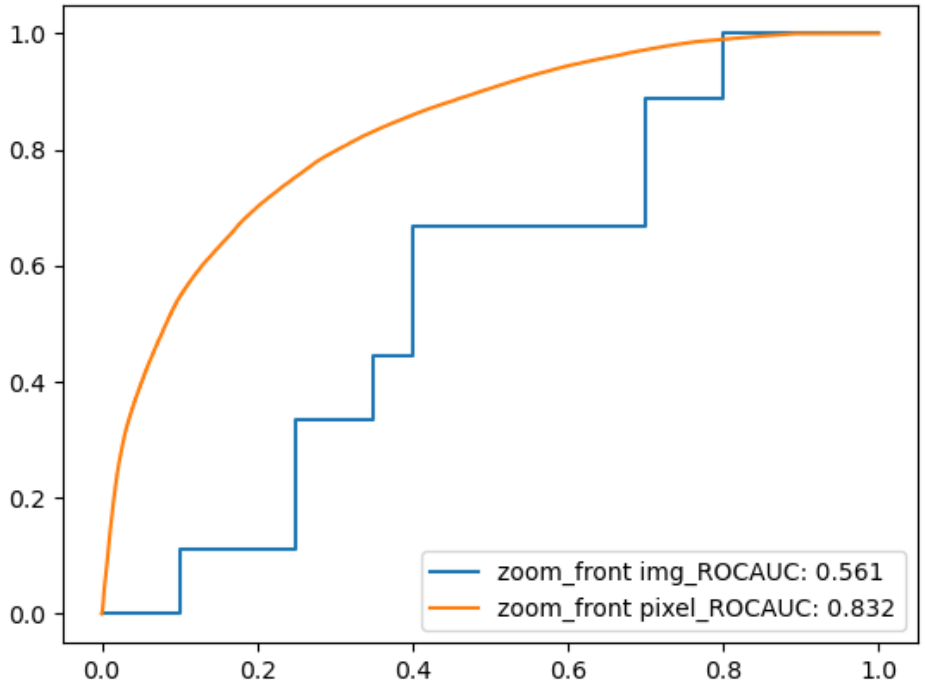
Obrázok 54. RIAD image ROC krivka a pixel ROC krivka kategórie side1_viewA



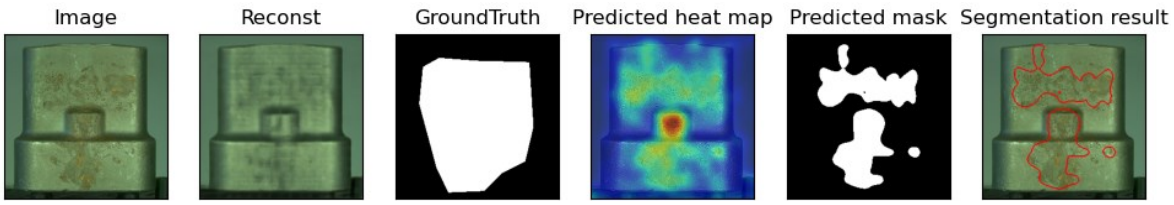
Obrázok 55. RIAD image ROC krivka a pixel ROC krivka kategórie side1_viewB



Obrázok 56. RIAD image ROC krivka a pixel ROC krivka kategórie side1_viewC



Obrázok 57. RIAD image ROC krivka a pixel ROC krivka kategórie zoom_front



Obrázok 58. Príklad správnej detekcie anomálie a jej lokalizácie modelom RIAD

5.3 SPADE

Táto implementácia [30] modelu SPADE má nasledovné požiadavky na spustenie:

- Python 3.6+
- PyTorch 1.5+
- Sklearn, matplotlib

Tieto požiadavky môžu byť nainštalované príkazom:

- `pip install -r requirements.txt`

Dataset bol uložený do zložky „datasets“ v kmeňovom priečinku. Následne boli upravené názvy tried a cesta k datasetu v súbore *mvtec.py*. Spustenie bolo vykonané zo zložky „src“ príkazom:

- `python main.py`

Výsledky aplikácie sú vizualizácie lokalizácií anomálií a image ROC krivky a pixel ROC krivky všetkých kategórií.

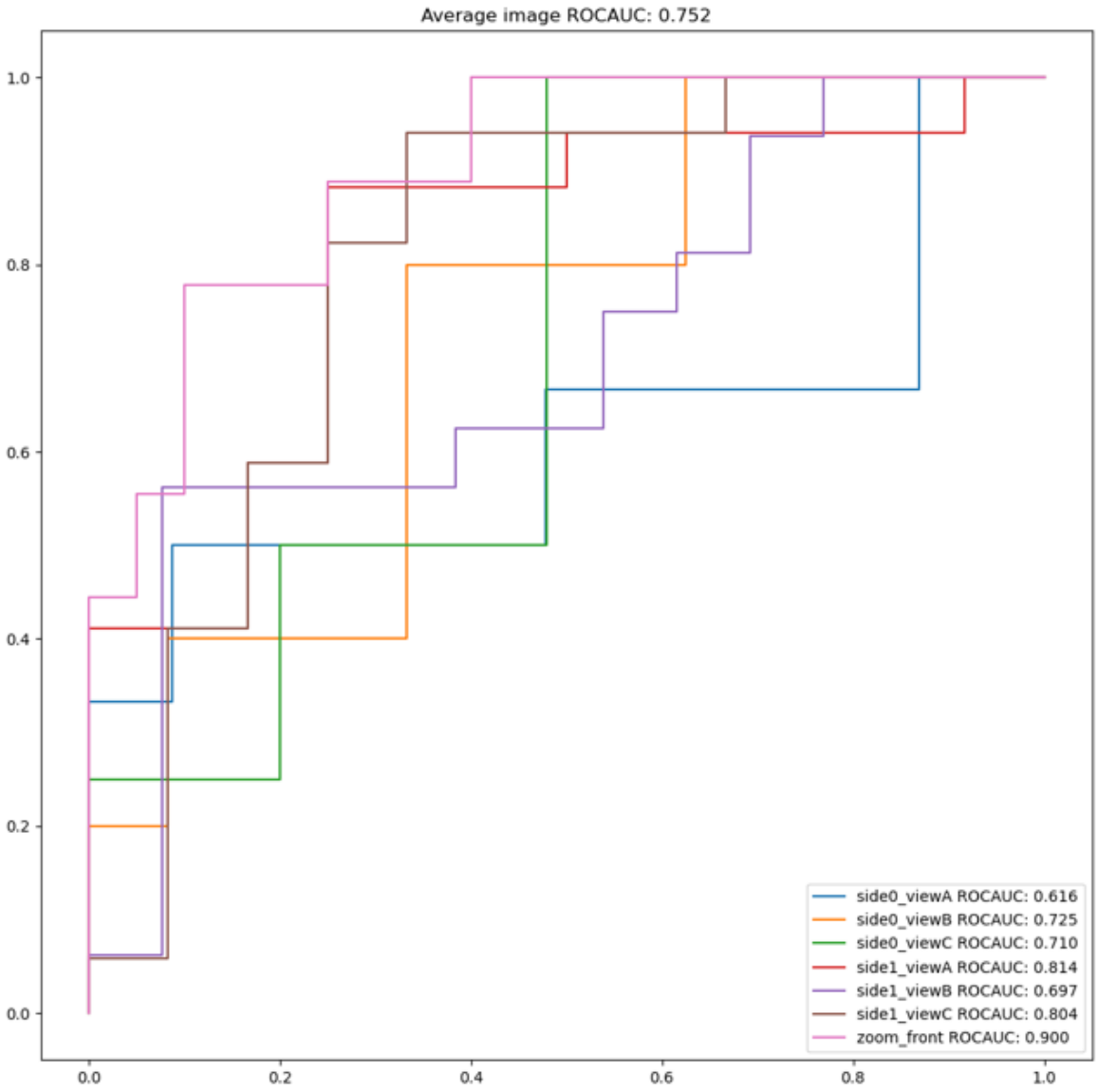
```
import os
import tarfile
from PIL import Image
from tqdm import tqdm
import urllib.request

import torch
from torch.utils.data import Dataset
from torchvision import transforms as T

CLASS_NAMES = ['side0_viewA', 'side0_viewB', 'side0_viewC', 'side1_viewA', 'side1_viewB',
               'side1_viewC', 'zoom_front']


class MVTecDataset(Dataset):
    def __init__(self, root_path='../data', class_name='side0_viewA', is_train=True,
                 resize=256, cropsize=224):
        assert class_name in CLASS_NAMES, 'class_name: {}, should be in {}'.format(class_name, CLASS_NAMES)
        self.root_path = root_path
        self.class_name = class_name
        self.is_train = is_train
        self.resize = resize
        self.cropsize = cropsize
        self.mvtec_folder_path = os.path.join(root_path, 'viva')
...
```

Obrázok 59. Upravený *mvtec.py* súbor modelu SPADE

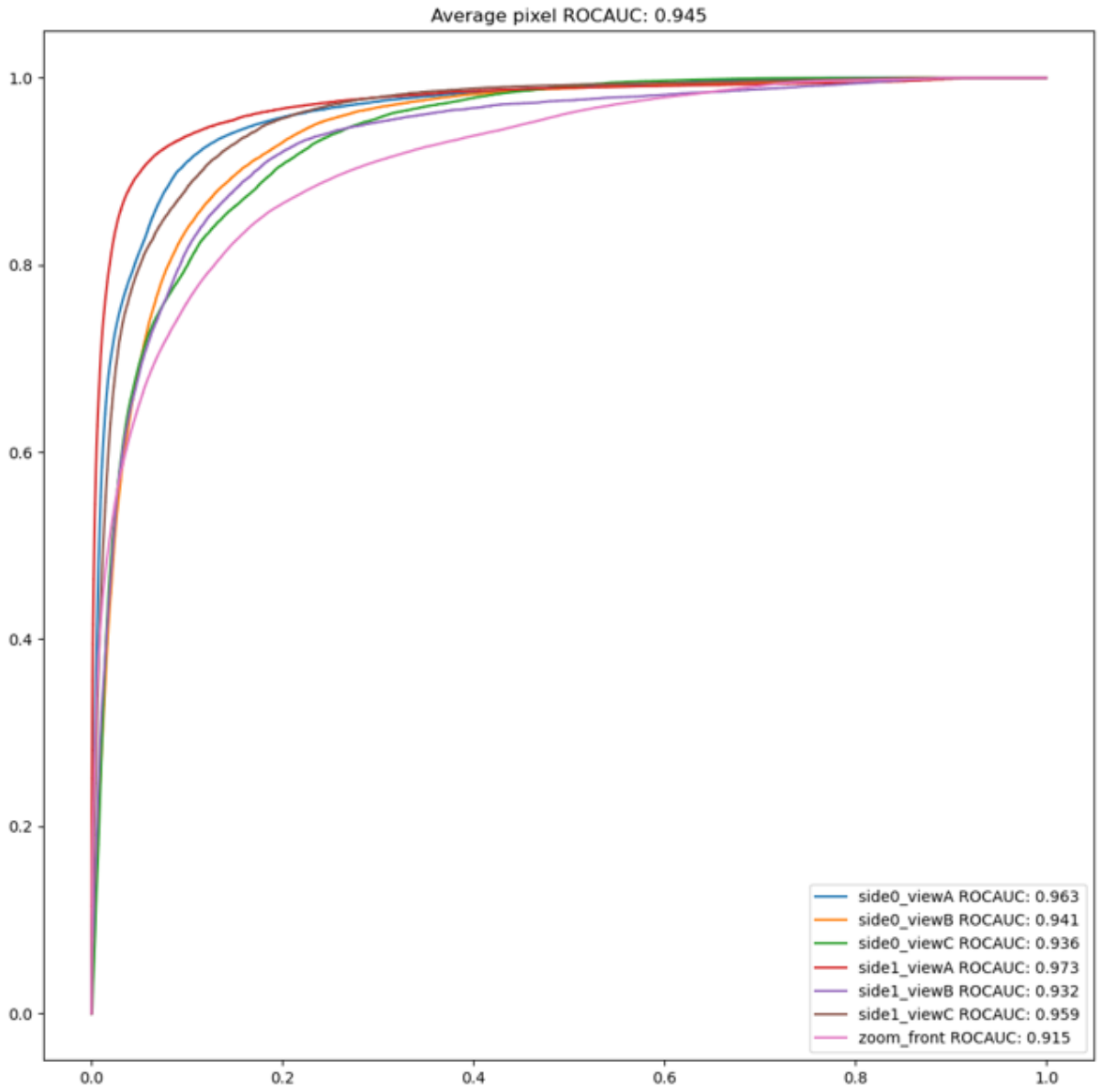
Z tabuľky 6 vyplýva, že priemerný výsledok pri segmentácii na úrovni obrázka modelu RDOCE bol 77,4% a na úrovni pixelov 90,9%. Najlepší výsledok pri segmentácii na úrovni obrazu bol 95,9% a bol dosiahnutý v kategórii side1_viewC. Naopak, najhorší výsledok segmentácie na úrovni obrazu, dosiahol model v kategórii side1_viewB, konkrétne 61,1%. Segmentácia na pixelovej úrovni bola najúspešnejšia v kategórii side1_viewA so skóre 97,3%. Najnižšie skóre segmentácie na úrovni pixelov bolo 74,1% a bolo dosiahnuté v kategórii side0_viewB.

Tabuľka 6. Výsledky modelu SPADE

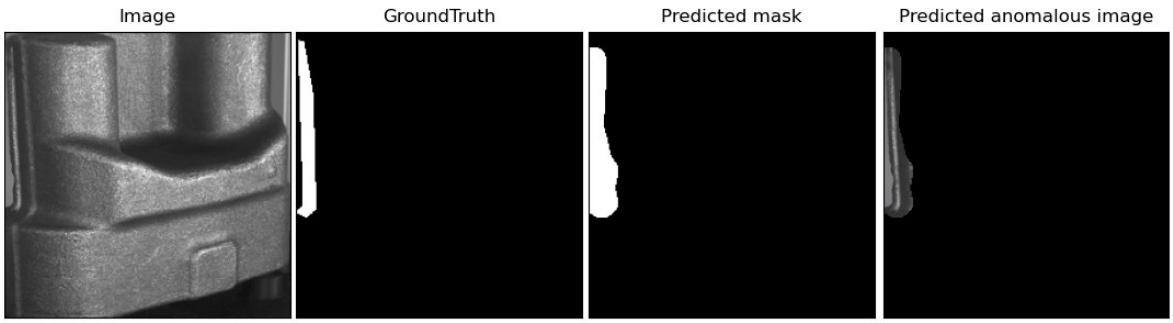
	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front	priemer
Image AUROC	61,1%	72,5%	71%	81,4%	69,7%	95,9%	90%	77,4%
Pixel AUROC	96,3%	74,1%	93,6%	97,3%	93,2%	90%	91,5%	90,9%



Obrázok 60. SPADE image ROC krivky všetkých kategórií



Obrázok 61. SPADE pixel ROC krivky všetkých kategórií



Obrázok 62. Príklad správnej detekcie anomálie a jej lokalizácie modelom SPADE

6 ZHODNOTENIE DOSIAHNUTÝCH VÝSLEDKOV

Každý model bol natrénovaný na pripravenom datasete, ktorý obsahuje 340 obrázkov. Rozlíšenie obrázkov je v rozmedzí od 807x807 pixelov do 2905x2905 pixelov. Pre použitie tohto datasetu bolo potrebné vykonať menšie úpravy voči každej implementácii modelu. Výstupy jednotlivých implementácií sa líšili v prezentácii aj v zobrazení. Modely zahrnuté v knižnici Anomalib zahŕňali individuálne image a pixel ROC grafy pre každú kategóriu anomálií. Naopak, implementácia modelu SPADE zobrazovala všetky kategórie v jednom grafe.

Z dôvodu nedostatočného výpočtového výkonu a podpory sa nepodarilo úspešne otestovať model SimpleNet, preto nie je zahrnutý v nasledujúcich výsledkoch.

Z tabuľky 7 vyplýva, že v oblasti detekcie anomálií na úrovni celých obrázkov je z testovaných modelov model PatchCore s hodnotou 87,9% najlepším modelom, nasledovaný modelmi SPADE (77,4%), PaDiM (75,3%), RIAD (71,8%) a DRÆM (57,1%). Model RDOCE dosiahol najnižšiu hodnotu s 53,3%.

Pri detekcii anomálií na úrovni pixelov dosiahol model PatchCore najvyššiu hodnotu AUROC s 93,3%. Nasledujú ho modely SPADE (90,9%), PaDiM (89%), RIAD (81,9%), DRÆM (74,1%) a RDOCE (64,9%).

Tabuľka 7. Priemerné výsledky modelov

	RDOCE	DRÆM	RIAD	PaDiM	SPADE	PatchCore
Image AUROC	53,3%	57,1%	71,8%	75,3%	77,4%	87,9%
Pixel AUROC	64,9%	74,1%	81,9%	89%	90,9%	93,3%

Tabuľka 8 zobrazuje image AUROC výsledky segmentácie na úrovni obrazu v jednotlivých kategóriách. Z tabuľky je zrejmé, že modely dosiahli rôzne úspešnosti pri rozpoznávaní významných detailov na snímkach zo všetkých uhlov, avšak najčastejšie dosiahli najlepšie výsledky na kategórii zoom_front.

Tabuľka 8. Image AUROC skóre modelov naprieč kategóriami

	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front
RDOCE	39,9%	63,3%	46%	63,7%	68,8%	43,1%	48,3%
DRÆM	56,5%	55%	47%	69,2%	39,4%	62,3%	70,6%
RIAD	75,4%	73,3%	85%	62,3%	68,3%	67,6%	70,6%
PaDiM	69,6%	77,5%	61%	85,3%	69,7%	77,9%	86,7%
SPADE	61,1%	72,5%	71%	81,4%	69,7%	95,9%	90%
PatchCore	81,8%	86,7%	90%	95,1%	83,7%	79,4%	98,9%

Tabuľka 9 zobrazuje pixel AUROC výsledky modelov v jednotlivých kategóriách. Najčastejšie dosiahli modely najlepšie výsledky segmentácie na úrovni pixelov v kategórii side1_viewA.

Tabuľka 9. Pixel AUROC skóre modelov naprieč kategóriami

	side0_ viewA	side0_ viewB	side0_ viewC	side1_ viewA	side1_ viewB	side1_ viewC	zoom_ front
RDOCE	48,4%	85,2%	55,9%	42,2%	40,4%	90,3%	92,2%
DRÆM	82,6%	74,3%	80,6%	74,8%	66,1%	74,4%	66%
RIAD	84,1%	82,2%	87,7%	88,7%	78.8%	86%	66%
PaDiM	87,8%	88,5%	88,2%	92,6%	87,2%	91%	88%
SPADE	96,3%	74,1%	93,6%	97,3%	93,2%	90%	91,5%
PatchCore	90,7%	93,1%	93,1%	95,2%	92,9%	91,9%	96%

Na základe zhodnotenia výsledkov je odporúčané v produkčnom prostredí zvážiť, pre detekciu a lokalizáciu anomálií v obraze, použitie modelu PatchCore. V porovnaní s

ostatnými testovanými modelmi dosiahol model PatchCore najvyššiu hodnotu AUROC na úrovni obrazu (87,9%) a najvyššiu hodnotu AUROC na úrovni pixelov (93,3%). Tieto výsledky naznačujú, že PatchCore má silnú schopnosť identifikovať anomálie a presne ich lokalizovať v obraze.

ZÁVER

Cieľom mojej diplomovej práce bolo vypracovanie literárnej rešerše modelov pre detekciu a lokalizáciu anomálií v obraze, príprava datasetu a natrénovanie vybraných modelov na pripravenom datasete. Zároveň táto práca poskytuje praktické odporúčanie pre výber modelu v produkčnom prostredí.

V teoretickej časti sa venujem literárnej rešerši, kde sú popísané jednotlivé modely, ktoré sa používajú na detekciu a lokalizáciu anomálií v obraze. Prvá kapitola sa venuje metódam založeným na embeddingu. Medzi tieto metódy patria PaDiM, SimpleNet, RDOCE a PatchCore. Druhá kapitola je zameraná na metódy založené na rekonštrukcii, medzi ktoré patria RIAD a DRÆM. V poslednej kapitole teoretickej časti je popísaný model SPADE, ktorý patrí medzi metódy založené na zarovnaní.

Praktická časť mojej práce sa venuje príprave datasetu a trénovaniu vybraných neurónových sietí z teoretickej časti. Pre trénovanie modelov bol pripravený dataset, ktorý pozostáva z 340 obrázkov výkovkov. Obrázky sú zhotovené z rôznych strán a uhlov. Dataset má podobnú štruktúru ako dataset MVTec, čo uľahčilo jeho implementáciu pri trénovaní modelov. Natrénované boli modely PaDiM, RDOCE, PatchCore a DRÆM z knižnice Anomalib. Ďalej boli natrénované modely SPADE a RIAD. Model SimpleNet z teoretickej časti nebolo možné natrénovať z dôvodu nedostatočného výpočtového výkonu a problémov so spustením, pričom autori projektu neodpovedajú na dotazy na GitHube. V závere praktickej časti sú vyhodnotené dosiahnuté výsledky a porovnané presnosti jednotlivých modelov. Model RDOCE dosiahol najnižšie hodnoty v porovnaní s ostatnými testovanými metódami. Jeho dosiahnutá presnosť detekcie anomálií v obraze sa vyčíslila na 64,9% pre pixel AUROC a 53,3% pre image AUROC. Naopak, model PatchCore dosiahol najvyššie hodnoty. Hodnota pixel AUROC modelu PatchCore bola 93,3% a hodnota image AUROC bola 87,9%.

Táto diplomová práca poskytuje prehľad rôznych metód a modelov v oblasti detekcie a lokalizácie anomálií v obraze. Odporúčanie použitia modelu PatchCore v produkčnom prostredí je výsledkom dôkladnej analýzy a potvrdzuje jeho vysokú presnosť a schopnosť lokalizovať anomálie v obraze.

ZOZNAM POUŽITEJ LITERATÚRY

[1] 2019. Embeddings. *Google for developers* [online]. [cit. 2023-05-23]. Dostupné z: https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture

[2] FENG, Zijian, 2022. An Overview of ResNet Architecture and Its Variants. *Built In* [online]. [cit. 2023-05-22]. Dostupné z: https://builtin.com/artificial-intelligence/resnet-architecture

[3] JAADI, Zakaria. A Step-by-Step Explanation of Principal Component Analysis (PCA). *Built In* [online]. [cit. 2023-05-22]. Dostupné z: https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[4] DEFARD, Thomas et al., 2020. *PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2011.08785

[5] LIU, Zhikang et al., 2023. *SimpleNet: A Simple Network for Image Anomaly Detection and Localization* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2303.15140

[6] DENG, Hanqiu a Xingyu LI, 2022. *Anomaly Detection via Reverse Distillation from One-Class Embedding* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2201.10703

[7] ABBASI, Sajjad et al., 2019. *Modeling Teacher-Student Techniques in Deep Neural Networks for Knowledge Distillation* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/1912.13179

[8] ROTH, Karsten et al., 2022. *Towards Total Recall in Industrial Anomaly Detection* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2106.08265

[9] *PatchCore* [online]. [cit. 2023-05-22]. Dostupné z: https://openvinotoolkit.github.io/anomalib/reference_guide/algorithms/patchcore.html

[10] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4):600–612, 2004. 2

[11] ZAVRTANIK, Vitjan, Matej KRISTAN a Danijel SKOČAJ, 2021. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition* [online]. **112** [cit. 2023-05-22]. ISSN 00313203. Dostupné z: doi:10.1016/j.patcog.2020.107706

[12] ZAVRTANIK, Vitjan, Matej KRISTAN a Danijel SKOČAJ, 2021 *DRÆM – A discriminatively trained reconstruction embedding for surface anomaly detection* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2108.07610v2

[13] COHEN, Niv a Yedid HOSHEN, 2020 *Sub-Image Anomaly Detection with Deep Pyramid Correspondences* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2005.02357

[14] 2021. Preparing Your Dataset for Machine Learning: 10 Basic Techniques That Make Your Data Better. *Altexsoft* [online]. [cit. 2023-05-22]. Dostupné z: https://www.altexsoft.com/blog/datascience/preparing-your-dataset-for-machine-learning-8-basic-techniques-that-make-your-data-better/

[15] MIKHAILOV, Andrew, 2022. How to Prepare a Machine Learning Dataset in 7 Steps?. *Zfort* [online]. [cit. 2023-05-22]. Dostupné z: https://www.zfort.com/blog/How-to-Prepare-a-Machine-Learning-Dataset-in-7-Steps

[16] AKCAY, Samet et al., 2022. Anomalib: A Deep Learning Library for Anomaly Detection. *GitHub: The world's leading software development platform* [online]. [cit. 2023-05-22]. Dostupné z: https://github.com/openvinotoolkit/anomalib

[17] AWAN, Abid, 2022. A Complete Guide to Data Augmentation. *Datacamp* [online]. [cit. 2023-05-22]. Dostupné z: https://www.datacamp.com/tutorial/complete-guide-data-augmentation

[18] RONNEBERGER, Olaf, Philipp FISCHER a Thomas BROX, 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* [online]. Cham: Springer International Publishing, 234-241 [cit. 2023-05-22]. Lecture Notes in Computer Science. ISBN 978-3-319-24573-7. Dostupné z: doi:10.1007/978-3-319-24574-4_28

[19] BERGMANN, Paul et al., 2021. The MVTec Anomaly Detection Dataset: A Comprehensive Real-World Dataset for Unsupervised Anomaly

Detection. *International Journal of Computer Vision* [online]. **129**(4), 1038-1059 [cit. 2023-05-22]. ISSN 0920-5691. Dostupné z: doi:10.1007/s11263-020-01400-4

[20] PAUL, Chris a Mandar GODAMBE, 2021. *Image Downsampling & Upsampling* [online]. [cit. 2023-05-22]. Dostupné z: https://www.researchgate.net/publication/359772964_Image_Downsampling_Upsamplinghttps://www.researchgate.net/publication/359772964_Image_Downsampling_Upsampling

[21] RIAD, Rachid et al., 2022. *Learning strides in convolutional neural networks* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/2202.01653

[22] Stride (Machine Learning). *Deepai* [online]. [cit. 2023-05-22]. Dostupné z: https://deepai.org/machine-learning-glossary-and-terms/stride?fbclid=IwAR0QAcw5DLv6g7RPS0pw_MgxPQgR3V4voeYov_4tdXO4c_7Vg2LZf4Tpl1s

[23] ALBAWI, Saad, Tareq Abed MOHAMMED a Saad AL-ZAWI, 2017. Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)* [online]. IEEE, 1-6 [cit. 2023-05-22]. ISBN 978-1-5386-1949-0. Dostupné z: doi:10.1109/ICEngTechnol.2017.8308186

[24] ZAGORUYKO, Sergey a Nikos KOMODAKIS, 2017. *Wide Residual Networks* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/1605.07146

[25] HE, Kaiming et al., 2015. *Deep Residual Learning for Image Recognition* [online]. [cit. 2023-05-22]. Dostupné z: https://arxiv.org/abs/1512.03385

[26] *Anomalib* [online]. [cit. 2023-05-22]. Dostupné z: https://openvinotoolkit.github.io/anomalib/

[27] JAIN, Shawn, 2020. SWAP: Softmax-Weighted Average Pooling: On improving gradient accuracy. *Towards data science* [online]. [cit. 2023-05-22]. Dostupné z: https://towardsdatascience.com/swap-softmax-weighted-average-pooling-70977a69791b

[28] LIU, Danqing, 2017. A Practical Guide to ReLU. *Medium* [online]. [cit. 2023-05-22]. Dostupné z: https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7

[29] XIA, Haifeng, 2020. Reconstruction-by-inpainting-for-visual-anomaly-detection. *GitHub: The world's leading software development platform* [online]. [cit.

2023-05-22]. Dostupné z: https://github.com/xiahaifeng1995/Reconstruction-by-inpainting-for-visual-anomaly-detection

[30] LEE, Byungjae, 2020 SPADE-pytorc. *GitHub: The world's leading software development platform* [online]. [cit. 2023-05-22]. Dostupné z: https://github.com/byungjae89/SPADE-pytorch

ZOZNAM POUŽITÝCH SYMBOLOV A SKRATIEK

RDOCE	Reverse Distillation from One-Class Embedding
PaDiM	Patch Distribution Modeling Framework for Anomaly Detection and Localization
DRÆM	Discriminatively trained reconstruction embedding model
RIAD	Reconstruction by inpainting for visual anomaly detection
SPADE	Sub-Image Anomaly Detection with Deep Pyramid Correspondences
CNN	Convolutional Neural Network
AUROC	Area Under the Receiver Operating Characteristic
ROC	Receiver Operating Characteristic
PCA	Principal Component Analysis
k-NN	k-Nearest Neighbors
AD	Anomaly Detection
KD	Knowledge Distillation
OCBE	One-class Bottleneck Embedding
OCE	Once-class Embedding
MFF	Multi-scale Feature Fusion
T-S	Teacher-Student
ReLU	Rectified Linear Activation Unit
AL	Anomaly Localization
SSIM	Structural Similarity
MSGMS	Multi-Scale Gradient Magnitude Similarity
GMS	Gradient Magnitude Similarity

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM PRÍLOH

Príloha P I: CD s diplomovou prácou vrátane zdrojových kódov testovaných aplikácií

PRÍLOHA P I: CD

Priložené CD obsahuje:

- Text
    - DP_AndrejOvecka_A20145.pdf
- Zdrojové kódy
    - prilohy.zip – zdrojové kódy testovaných aplikácií