FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Optimalizace systému pro automatizovanou detekci vad ve spojení s metodami nedestruktivního testování
Student:	Martin Vojtíšek
Vedoucí:	Ing. Jakub Novák
Studijní program:	Informatika
Obor / specializace:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	do konce letního semestru 2023/2024

Pokyny pro vypracování

Cílem práce je seznámit se s dříve vytvořeným systémem pro detekci vad používaných na výrobky zkoumané metodami nedestruktivního testování, analyzovat jednotlivé dílčí kroky zpracování a vyhodnocení dat a navrhnout jejich vylepšení.

Práce navazuje na dříve obhájenou práci https://projects.fit.cvut.cz/theses/3728. Součástí práce byl návrh vyhodnocovacího algoritmu včetně výběru a implementace architektury detekční a klasifikační neuronové sítě. Tato práce se nebude věnovat snímací soustavě, do hloubky však analyzuje všechny kroky zpracování dat a optimalizuje práci vyhodnocovacího algoritmu.

Úkoly:
1) Seznamte se s hotovým systémem pro detekci vad a všemi dílčími kroky jeho vyhodnocovacího algoritmu.
2) Zaměřte se na části práce s datasetem (balancování a generování dat), předzpracování dat i vyhodnocení.
3) Proveďte rešerši vhodných metod a možností aplikovatelných na výše zmíněné části.
4) Navrhněte úpravy a vylepšení algoritmů výše zmíněných částí, optimalizujte celý systém.
5) Otestujte výsledky optimalizace a zhodnoťte úspěšnost.
6) Vizualizujte výsledky algoritmu na obrazových datech.

Elektronicky schválil/a Ing. Magda Friedjungová, Ph.D. dne 4. února 2023 v Praze.

Bakalářská práce

OPTIMALIZACE SYSTÉMU PRO AUTOMATIZOVANOU DETEKCI VAD VE SPOJENÍ S METODAMI NEDESTRUKTIVNÍHO TESTOVÁNÍ

Martin Vojtíšek

Fakulta informačních technologií
Katedra aplikované matematiky
Vedoucí: Ing. Jakub Novák
10. května 2023

Obsah

Seznam obrázků

Seznam tabulek

Tímto bych chtěl poděkovat především vedoucímu práce Ing. Jakubu Novákovi za velmi časté konzultace a přátelský přístup. Rovněž děkuji Bc. Adamovi Malečkovi za pomoc při navazování na jeho kód. V neposlední řadě děkuji celému ImproLabu za poskytnuté výpočetních prostředků nutných k trénování modelů.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel licenční smlouvu o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 10. května 2023 .....

Abstrakt

Práce se zabývá optimalizací detekce vad v metodách NDT (nedestruktivního testování). Problém je řešen navržením nového detekčního algoritmu, založeného na modelu U-NET, který řeší nedostatky dřívějšího modelu. Zároveň je rozebíráno, jak algoritmicky rozšířit bázi dat pro trénování. Výsledkem práce jsou optimalizace procesu, které podstatně zvýšily přesnost celého systému detekce. Tyto optimalizace jsou univerzálně aplikovatelné na problém detekce malých vad a mohou tak zlepšit výkonost systému i mimo doménu NDT.

Klíčová slova detekce vad, nedestruktivní testování, U-NET, generování umělých dat, aproximace distribuce vad, segmentace velmi malých objektů, předzpracovaní snímků

Abstract

The thesis deals with the optimization of defect detection in NDT (non-destructive testing) methods. The problem is solved by designing a new detection algorithm, based on the U-NET model, which solves the shortcomings of the earlier model. Also, it is discussed, how to algorithmically expand data for training. The result of this work is an optimization of the process, which significantly increased the accuracy of the entire detection system. These optimizations are universally applicable to the problem of detecting small defects and can thus improve system performance even outside the NDT domain.

Keywords defect detection, non-destructive testing, U-NET, generation of artificial data, approximation of defect distribution, segmentation of very small objects, image preprocessing

Seznam zkratek

ANN a NN	Umělá neuronová Síť
lr	Learning rate
NDT	Nedestruktivní testování
GAN	Generative adversarial network
KNN	K nejbližších sousedů
CNN	Konvoluční neuronová síť
FFT	Dopředná rychlá Fourierova transformace
IFFT	Zpětná rychlá Fourierova transformace
MAX	Maximum / Maximální hodnota
AVG	Průměr / Průměrná hodnota
SVM	Support vector machine
RPN	Region proposal network
px	Pixel / Pixelů
TP	True positive
FP	False positive
TN	True negative
FN	False negative
TPR	True positive rate
FPR	False positive rate

Kapitola 1
Úvod

Důležitou součástí výroby dílů v průmyslu je testování kvality. Díly jsou vyráběny ve velkém množství, a proto se jejich manuální kontrola člověkem stává velmi obtížnou. V průmyslu je dlouhodobě snaha tyto procesy zautomatizovat, a tedy zvýšit objektivnost detekce. Práce navazuje na závěrečnou práci Adama Malečka [1], ve které se věnoval automatizaci detekce a vyhodnocení indikací souvisejících s metodami nedestruktivního testování (NDT). Metody se používají pro detekci okem neviditelných vad v kovových výrobcích. Implementace dosahovala dobrých výsledků na původním datasetu, kde byly chyby i na první pohled dobře rozeznatelné.

Od té doby se spolupráce s firmou ATG rozšířila a přibylo mnoho nových typů dílů, a tím pádem vznikl nový dataset, který je nafocený přímo v dané firmě. V novém datasetu (se složitějšími díly) jsou však chyby již obtížněji rozeznatelné od šumu vlivem složitějšího snímání, horšího osvětlení a z důvodu malé velikosti pravých indikací. Dosavadní metody detekce tak selhávají. Práce má za cíl zmapovat celé řešení problematiky včetně dílčích přístupů a najít algoritmy schopné tyto obtížnější vady detekovat. Dále se zabývá problematikou množství vstupních dat a případným rozšířením báze trénovacích dat tak, aby došlo ke zlepšení detekce.

Kapitola 2
Literární výzkum

Předpokládané řešení vyžaduje zorientovat se v problematice neuronových sítí, generování dat a předzpracování vstupu.

2.1 Systém pro automatizovanou detekci vad ve spojení s metodami nedestruktivního testování

V práci [1] se řešil problém detekce indikací v NDT, pouze na jiných datech. Finálním výstupem práce byl natrénovaný model typu Faster R-CNN [2] pro detekci vad. Na práci [1] navazuje současná práce, která je de facto pokračováním výzkumu na stejném projektu pod záštitou Laboratoře zpracování obrazu na FIT ČVUT.

Práce [1] využívá primárně Faster R-CNN [2] vycházející z Fast R-CNN [3] a klasického R-CNN [4]. Také se v práci [1] zmiňuje detekce pomocí klasických metod strojového vidění [5].

2.2 U-Net: Convolutional Networks for Biomedical Image Segmentation

V článku [6] byl publikován model architektury U-Net. Tento model slouží k segmentaci objektů pomocí výstupní masky. Zde byl použit se značným úspěchem na problém v oblasti biomedicíny.

Také je v článku [6] zmíněn nástroj elasticdeform [7] pro diverzifikaci vstupních dat pomocí grid-based deformací.

2.3 End-to-end training of a two-stage neural network for defect detection

V článku [8] je zavedena two-stage architektura. Kde první část slouží k segmentaci a učí se pouze na masce indikace a na ní je napojená část klasifikační učící se na třídě klasifikace. Také je zde zaveden způsob, jak obě části trénovat najednou pomocí kombinace klasifikační a segmentační loss funkce.

2.4 Generování dat pomocí GAN

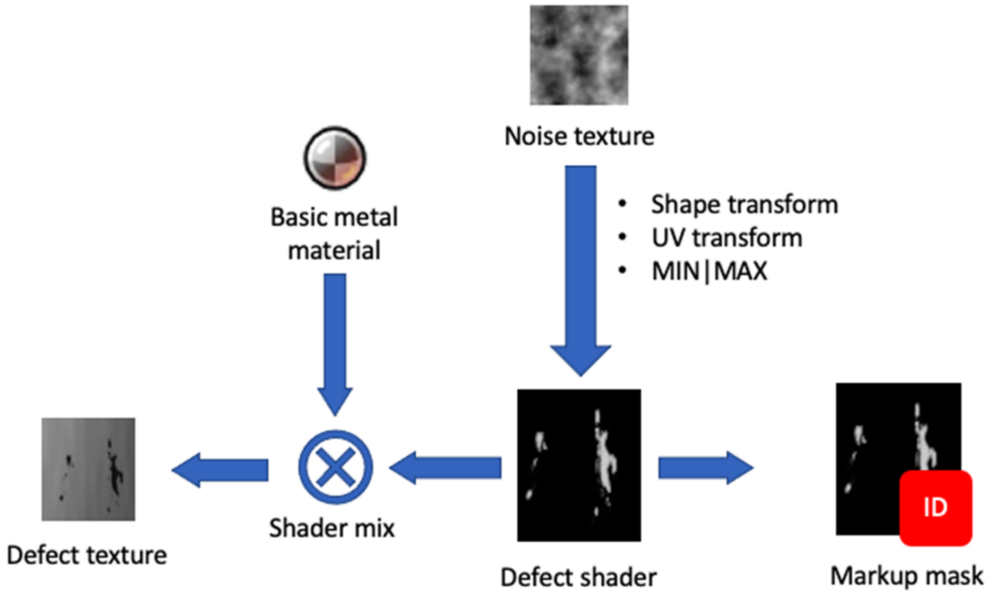
V článku [9] byl použit model GAN pro generování nových augmentovaných dat obrázků lidského mozku. Pro použití tohoto postupu je však potřeba značné množství dat. V článku je ale demonstrováno, že syntetická data mohou zlepšit přesnost modelu (Reálná + Syntetická).

Tabulka 2.1 Výsledky článku na Dice skóre evaluace

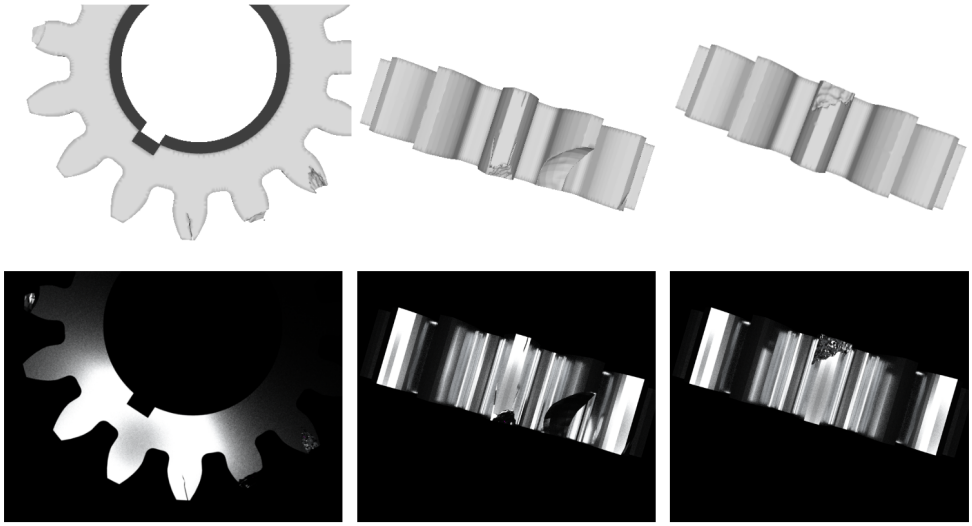
Metody	Reálná	Reálná + Syntetická	Syntetická	Syntetická + fine-tuning 10 % reálných
GAN-based (no aug)	0,64/0,14	0,80/0,07	0,25/0,14	0,80/0,18
GAN-based (with aug)	0,81/0,13	0,82/0,08	0,44/0,16	0,81/0,09
Wang *et al*,	0,85/0,15	0,86/0,09	0,66/0,13	0,84/0,15

2.5 Algoritmické generování syntetických dat z 3D modelu

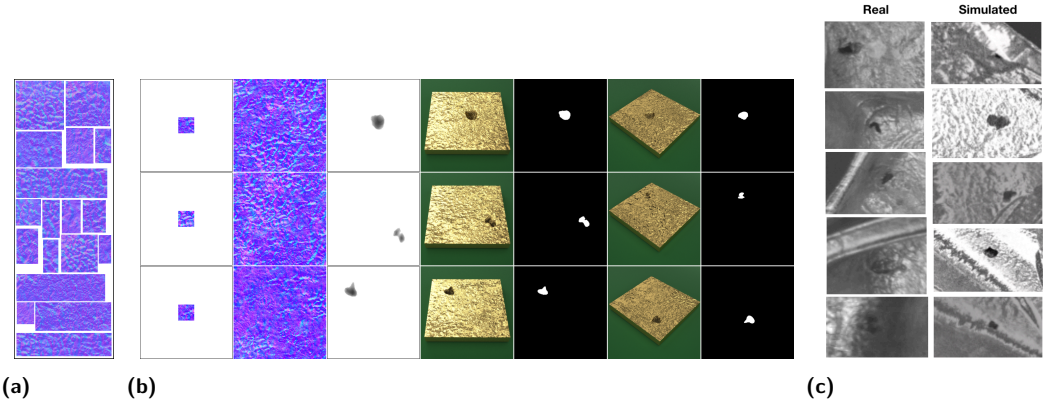
V článku [10] byla generována data pomocí vkládání chyb generovaných ze speciálně vytvořené textury na simulovaný díl železa. Podobný přístup byl zvolen v [11] a [12], kde jsou data generována pomocí 3D modelu dílu náhodným obtisknutím takzvaného defect toolu. Následně je modelu přiřazena textura a je z něj vyrendrován 2D obrázek. Generování pomocí 3D modelu je použito také v [13] s rozdílem, že defect tool je generován náhodně aplikováním různých typů šumu na kruh, ze kterého ve finále vznikne hloubková mapa – tedy vlastně defect tool. Také je zde zmíněn postup, jak generovat texturu ze seznamu již existujících textur pomocí iterativního KNN algoritmu. Postupy založené na 3D modelech však lze použít jen obtížně, pokud jsou chyby velmi rozmanité, a tedy nelze vytvořit výše zmíněné defect tools tak, aby pokryly významnou část distribuce indikací.



Obrázek 2.1 Schéma generování dat v [10].



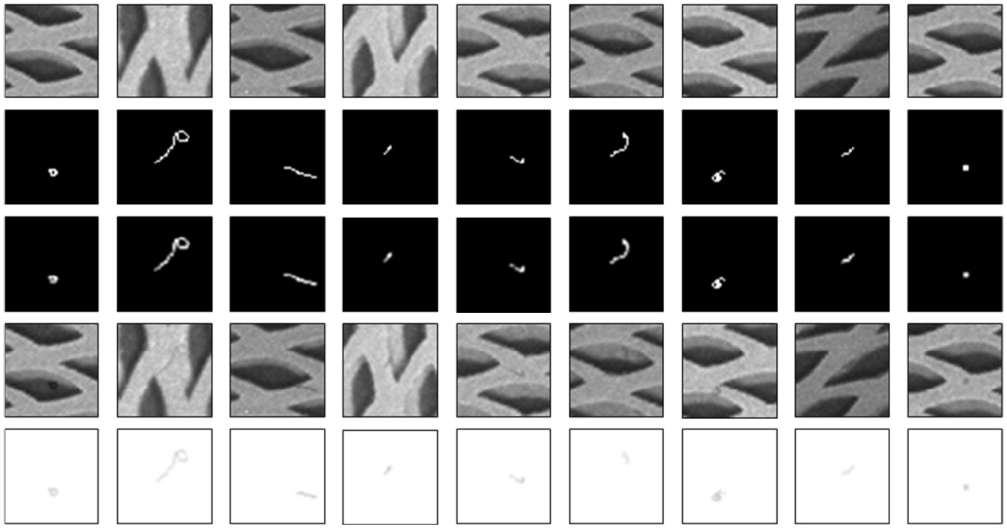
Obrázek 2.2 Chyby vygenerované v [11].



Obrázek 2.3 Generování vad v [13] (a) Seznam již existujících textur; (b) První sloupec: seed pro generování; Druhý sloupec: syntetické textury; Třetí sloupec: syntetické vady; Čtvrtý a Šestý sloupec: 3D rendery; Pátý a Sedmý sloupec: masky vad; (c) Reálná vs. syntetická data.

2.6 Generování indikací ve 2D

Článek [14] se zabývá generováním prasklin na základě náhodné procházky v polárním souřadnicovém systému, která je řízena velikostí kroku a úhlem definovaným pomocí dvou Bernuliho distribucí určujících pravděpodobnost změny v kroku I a dvou uniformních distribucí určujících rozsah změny, kde první pár Bernuliho a uniformní distribuce určují bias kroku tj. složku která má být konzistentnější a udržovat určitý směr procházky. Tato procházka pak vytvoří skeleton, kterému je přiřazena textura a následně je rozmazána, takto vznikne výsledná chyba, která je pak vložena do obrázku dílu bez defektu. Tato metoda je využívána také v článku [15] pro trénování segmentační CNN. Přístup neumí získat tyto distribuce z dat. Počítá tedy s manuálním vytvořením distribucí, což může být velmi nepraktické. Postup však lze použít jako stavební kámen pro další metody generování umělých dat.



Obrázek 2.4 Chyby vygenerované v [14].

2.7 Předzpracování dat před vstupem do NN

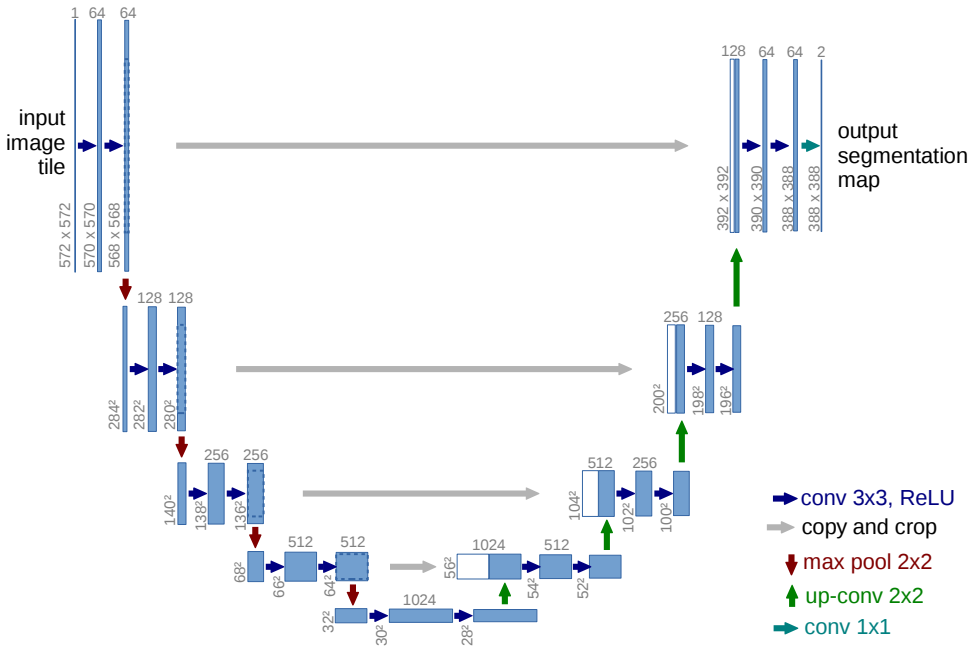
Článek [16] zavádí takzvaný spektrální pooling, ve kterém je vstup převeden pomocí FFT do spektrální domény a v ní jsou oříznuty vysoké frekvence. Zbytek frekvencí je převeden pomocí IFFT zpět. Tento článek je zajímavý tím, že dle autorů se při tomto postupu neztrácí tolik informace jako při max poolingu a tedy je především použitelný pro prvotní zmenšení obrázku, kde se ztrácí nejvíce informace bez jakékoli šance na naučení její vhodné extrakce (nachází se mimo NN).

Obrázek 2.5 Demonstrace spectral poolingu v [16]

Kapitola 3

Teoretické zázemí

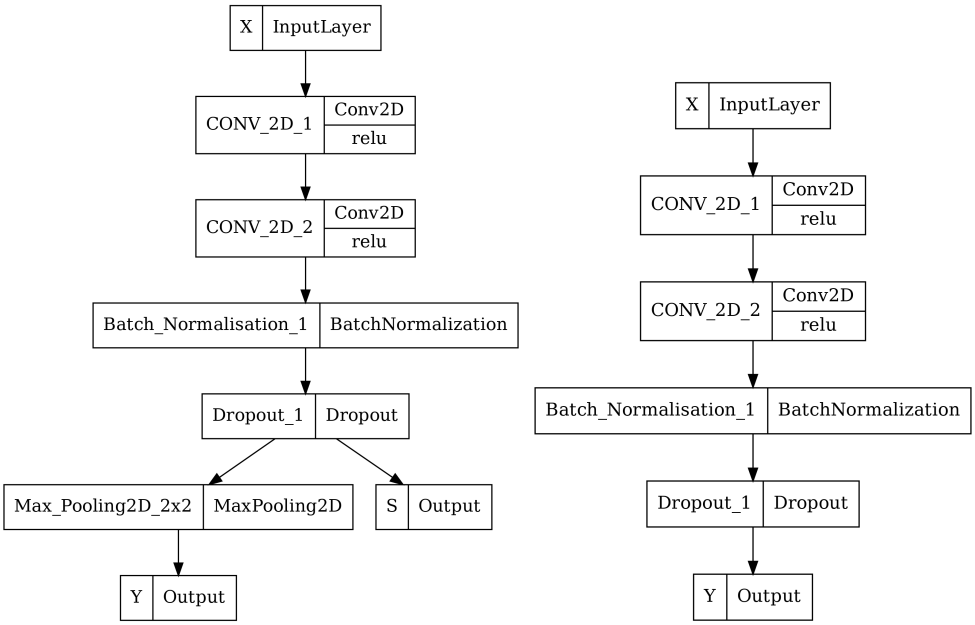
Řešení předpokládá základní teoretické znalosti z oblastí strojového vidění a hlubokého učení. Pro pochopení práce je třeba znát podrobně model U-NET, a proto je zde rozebrána jeho architektura. Model jako takový se skládá z enkodér a dekodér bloků.



Obrázek 3.1 U-NET model vizualizace v [6].

3.1 Enkodér blok

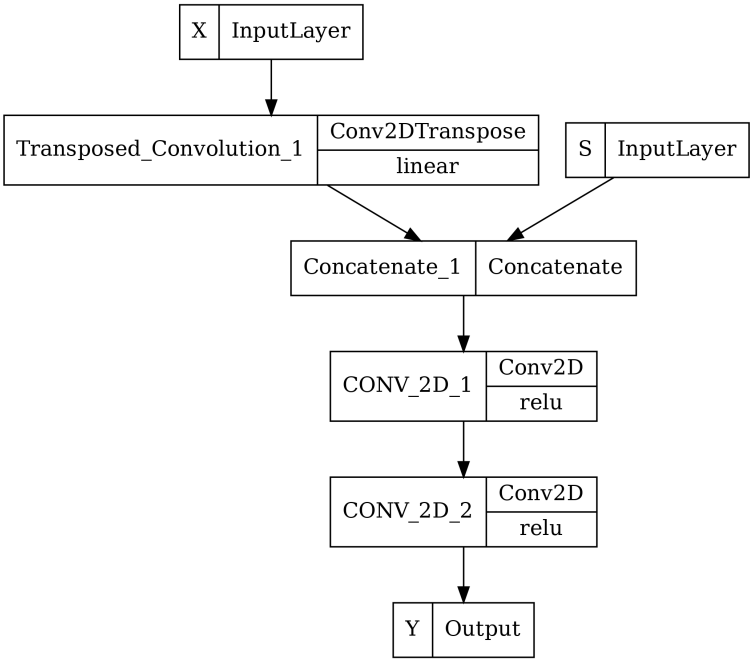
Každý enkodér i přijímá vstup X a vrací výstup Y, který je přijímán $i+1$ enkodérem nebo prvním dekodérem, druhý výstup S slouží jako skip connection do dekodéru $N-i$ (kde N je počet všech bloků i s prostředkem sítě). Skip connection posledního enkodéru se zahazuje a zároveň poslední enkodér už neobsahuje max pooling.



■ **Obrázek 3.2** Vpravo klasický enkodér, vlevo poslední enkodér.

3.2 Dekodér blok

Dekodér i pak přijímá vstup X z předešlého dekodéru $i-1$ nebo posledního enkodéru a skip connection S z příslušného enkodéru $N-i$ a vrací výstup Y do dalšího dekodéru nebo do poslední vrstvy sítě.



■ **Obrázek 3.3** Dekodér vizualizace.

Kapitola 4
Analýza

Pro zlepšení systému [1] je třeba analyzovat jeho jednotlivé části a v těchto částech navrhnout úpravy, které by mohly vést k vylepšení jeho přesnosti na novém datasetu.

4.1 Dataset

Původní dataset v práci [1] obsahoval pouze 27 trénovacích a 10 testovacích snímků. Současný dataset je tvořen čtyřmi díly a obsahuje 955 trénovacích a 347 validačních snímků. Rozdělení dat bylo provedeno tak, aby byly ve validačním setu fyzicky jiné díly (nestačí jen jiná strana konkrétního dílu) než v trénovacím. Po zvolení finálního modelu se počítá s tím, že se vytvoří separátní testovací set, a to buďto novým nafocením jiných dílů nebo přegenerováním ze současných dat. Testovací set nesmí obsahovat žádné snímky použité v původním validačním setu.

Rozložení dílů je následující:

Tabulka 4.1 Velikosti jednotlivých datasetů pro konkrétní typy dílů.

Typ Dílu	počet TRAIN	počet VAL
m114	347	126
m083	258	97
m072	290	104
m028	60	20

Nový dataset se od toho použitého v [1] významně liší:

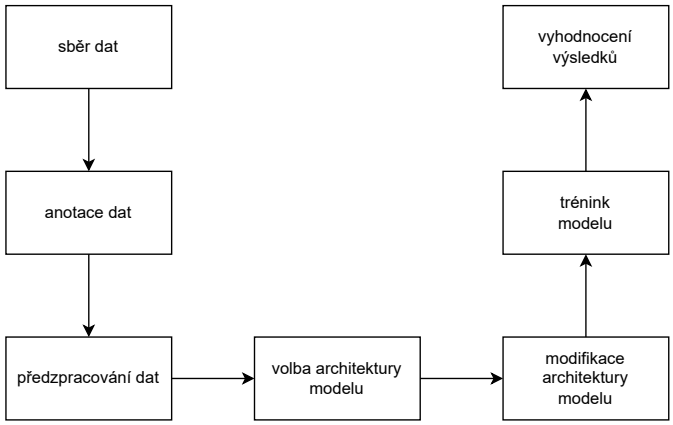
1. V množství šumu, který je podobný indikacím, kde na horním obrázku 4.1 je několik artefaktů, které se podobají indikaci, ale indikací nejsou. Naproti tomu v dolním obrázku z původních dat jsou artefakty lehce odlišitelné od pravých indikací.

2. Ve velikosti chyb, kde chyby v novém datasetu mohou být v extrémních případech i velikosti např. 3×20 px oproti 4096×3000 px obrázku (0,0005 % obsahu).

■ **Obrázek 4.1** Indikace v novém datasetu s upravenou expozicí pro lepší viditelnost (nahoře). Indikace v práci [1] (dole).

4.2 Optimalizace jednotlivých částí systému

Původní systém [1] je tvořen z těchto částí: sběr dat, anotovaní dat, předzpracování dat, volby architektury modelu, modifikace architektury modelu, trénování modelu a vyhodnocení výsledků. To je přehledně znázorněno na diagramu 4.2. Je třeba analyzovat a optimalizovat postupně všechny jednotlivé části systému.



Obrázek 4.2 Jednotlivé části systému.

4.2.1 Sběr dat

Původní systém pracuje pouze s nafocenými daty, nad kterými je prováděna náhodná rotace. Ke zlepšení modelu by přispělo větší množství reálných dat. Ta bohužel nejsou k dispozici, a proto dává smysl využít metody tvorby umělých dat. Práce nahlíží na rozšíření dat a diverzifikaci základních dílů (bez indikací) ze dvou různých pohledů.

1. Rozšířit data tak, aby byla co nejpodobnější těm reálným i za cenu toho, že lze takto vygenerovat jen určité omezené množství. To je prováděno pomocí zahlazování indikací v chybných dílech tak, aby byly nerozpoznatelné od těch pravých. Rotace ani změna škály se v tomto přístupu neuvažují, jelikož původní díly se vždy fotí ze stejného úhlu a vzdálenosti. Už zde by tedy vznikala určitá systematická odlišnost od reálných dat.

2. Rozšířit data tak, aby výsledné rozšíření pokrývalo co největší distribuci všech možných vstupů. Zároveň se počítá i s tím, že dataset bude obsahovat data, která jsou v realitě nemožná (jiná rotace, škála, ...). Přístup počítá s vygenerováním velkého množství dat. K tomu je využit nástroj elastic deform [7] společně s náhodnými rotacemi a změnou škály.

4.2.2 Anotace dat

V původním systému se pracuje pouze s manuálně anotovanými indikacemi. Augmentace rotací je v tomto smyslu schopná pouze velmi nepatrné diverzifikace, jelikož vada rotuje společně s dílem. Současná práce problematiku v 4.2.1 rozšiřuje a vylepšuje:

1. Pro rozšíření, které je nejpodobnější reálným datům se do bezchybných dat vkládají indikace (které lze ještě augmentovat rotací a přeškálováním) z dat s chybami tak, aby byly výsledné snímky nerozeznatelné od těch reálných (neobsahovaly systematický bias).

2. Pro rozšíření pokrývající co největší distribuci vstupních dat. Indikace (konkrétně praskliny) jsou generovány pomocí náhodné procházky (podobně jako v [14]). Avšak na rozdíl od článku

se cílí na to, jak odhadnout distribuci délky, tloušťky a průběhu těchto indikací tak, aby co nejlépe vystihovala realitu pomocí reálného datasetu.

4.2.3 Předzpracování dat

V původním systému se data rovnou vkládají do modelu bez předzpracování. Díky tomu, že současný dataset obsahuje snímky velmi velké velikosti (4096×3000 px), je potřeba je před vstupem zmenšit. Je třeba najít takové zmenšení, při kterém se ztrácí co nejméně vstupních informací. Pro tyto účely je porovnáváno klasické bilineární zmenšení oproti postupu odříznutí vysokých frekvencí navrženém v [16]. Zároveň je vyzkoušeno, jestli modelu pomůže aplikování frekvenčního filtru potlačujícího pozadí před vstupem. Jelikož CNN teoreticky fungují na základě prostorového filtrovaní a ne frekvenčního. Jiné úpravy však už nedávají příliš smysl, protože CNN jsou velmi silné modely, které mohou velkou většinu běžných úprav vstupu (založených na konvoluci) teoreticky provést samy.

4.2.4 Volba architektury modelu

Modely zmíněné v předchozí práci [1] nejsou vhodné pro současný dataset. Proto je navržen jiný model, z jehož principů vychází i další použité odvozené modely.

Detekce pomocí klasických metod strojového vidění

Práce [1] se odkazuje na článek [5], ve kterém se detekce provádí pomocí kombinací Gaussova rozostření, top-hat transformace, prahování a morfologických transformací. Následně se z výsledku vyextrahují lokální deskriptory (obsah regionu, poměr barev) a podle nich se rozhoduje o indikaci. Další variantou, která se uvádí, je tyto příznaky dát do klasifikátoru (náhodné lesy) klasického strojového učení.

Problémy se současným datasetem Metody nejsou schopné se jakýmkoli způsobem přizpůsobit datasetu. A jelikož rozdíly mezi šumem a indikacemi nejsou velké a samotné indikace nejsou příliš kontrastní, tak by tento postup nefungoval dobře.

Detekce pomocí R-CNN a Fast R-CNN

R-CNN používá [17] pro nalezení regionů a následně [18] sloučí do 2000 finálních návrhů, které dává na vstup přetrénované CNN. Následně na výstupu CNN pomocí SVM rozhoduje, zda se v daném návrhu nachází daná třída.

Fast R-CNN, na rozdíl od R-CNN, celý obrázek zpracuje CNN a z jejího výstupu pak vybírá reagiony na základě aplikace [17] a [18] na původním obrázku. SVM klasifikace je nahrazena ROI poolingem regionů, klasifikační vrstvou a regresní vrstvou sloužící pro zlepšení bounding boxu.

Problémy se současným datasetem Algoritmus [17], který je inspirován kruskalovým algorimem (tedy jistým způsobem bere obrázek jako graf a spojuje komponenty grafu, které jsou si podobné), není schopen se nikterak naučit cokoli z dat. To je ale problém nových dat, kde nejsou chyby moc rozdílné od šumu. Také R-CNN dostává jen 2000 výběrů, což působí při malé velikosti indikací problémy.

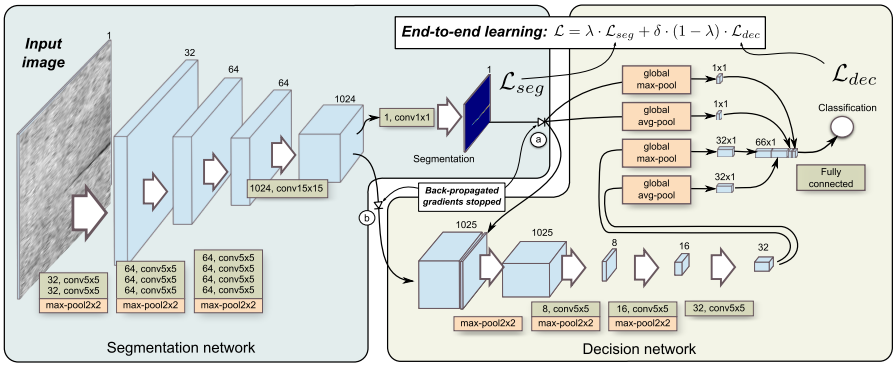
Detekce pomocí Faster R-CNN

Faster R-CNN oproti Fast R-CNN používá region proposal network (RPN) místo výše zmíněných algoritmů na vytváření regionů. RPN bere ze společné předtrénované sítě pro klasifikaci okénka určité velikosti $N \times N$ (většinou 3×3) a z každého okénka predikuje $k = 9$ anchor pointu

pro různé velikosti a poměry boxu vybíraného ze vstupu. Každý anchor point se skládá z výstupu klasifikace určujícího, zda-li je okénko pozadí nebo ne a z regrese předpovídající úpravu výstupního boxu oproti výchozí velikosti boxu anchor pointu.

Problémy se současným datasetem Tato detekční metoda je schopná se naučit výběr regionů z datasetu, avšak generuje hodně návrhů. Jelikož v datasetu existuje šum podobný pravým indikacím, šance falešné detekce se tímto dost zvyšuje. Také je zde problém netriviálního rozhodování při kolika a jak výrazných detekcích brát díl jako vadný.

End-to-end training of a two-stage neural network for defect detection



■ **Obrázek 4.3** Ilustrace původní architektury v článku [8].

Model [8] obsahuje segmentační část učící se pouze z masky chyby, tedy klasifikační loss funkce nijak neovlivňuje tuto část. Výstup předposlední vrstvy segmentační CNN společně s výstupní maskou je pak vstupem do klasifikační CNN. Do finální klasifikační vrstvy směřuje globální AVG a MAX z filtrů klasifikační CNN a z masky vystupující ze segmentační části sítě.

Vhodnost pro současný dataset Postup zmíněný v článku [8] však žádný z výše zmíněných problémů nemá. Detekce je prováděna pomocí segmentační části sítě, která se přímo učí z dat a klasifikace se provádí pomocí klasifikační sítě napojené na tu segmentační. Tedy vrací pouze pravděpodobnost pro celý snímek, a tak nehrozí, že model bude příliš náchylný na falešně pozitivní detekce při segmentaci. Zároveň je v článku popsán postup, jak obě části trénovat najednou, což značně šetří čas nutný k potenciálnímu přetrénování modelu na novějších datech.

4.3 Modifikace architektury modelu

Postup zmíněný v [8] však počítá s nemalým zmenšováním masky (viz schema 4.3), což je pro současná data s velmi malými indikacemi problém. Zároveň jeho klasifikační část není až tak silným modelem, aby dokázala segmentovat některé složitější chyby. Proto je segmentační část nahrazena modelem U-NET [6] a klasifikační část bude patřičně upravena, aby byla s U-NETem kompatibilní.

4.4 Trénink modelu

Trénování zůstává nezměněné oproti [8] tedy Loss funkce $\mathcal{L}_{total}$ se řídí vzorcem:

$$\lambda = 1 - \frac{n}{total\_epoch} \tag{4.1}$$

$$\mathcal{L}_{total} = \lambda \cdot \mathcal{L}_{seg} + \delta \cdot (1 - \lambda) \cdot \mathcal{L}_{cls}, \tag{4.2}$$

Kde n je aktuální epocha, $\mathcal{L}_{seg}$ loss funkce segmentační části sítě (binary cross entropy) a $\mathcal{L}_{cls}$ je loss funkce klasifikační sítě (binary cross entropy). δ je hyperparamer vyrovnávající velikosti $\mathcal{L}_{seg}$ a $\mathcal{L}_{cls}$, tak aby byly zhruba stejné (klasifikační chyba bývá v tom případě vždy větší). Zde bylo empiricky zvoleno $\delta = 0,5$. Jako optimizer je použit pro obě části sítě Adam [19] s výchozími parametry.

Trénovací proces se provádí ve dvou, případně třech krocích. Nejprve může být v některých případech model přetrénován na 50 epochách a $lr = 0,0002$ na čistě syntetických datech zmíněných v bodě 2 sekce 4.2.2 .

Následně je trénovací proces tvořen trénováním o 50 epochách a $lr = 0,001$ na veškerých datech. Z procesu jsou vybrány váhy epochy s největší validační přesností. Na vahách je proveden finetuning na samostatných třídách dílů o 15 epochách a $lr = 0,0001$. Nejlepší váhy z finetuningu dle validačních dat jsou prohlášeny jako výsledek trénování.

4.5 Vyhodnocení výsledků

Po nalezení vah je výkon modelu změřen na validačních datech. Výkonem je zde konkrétně myšleno TPR a FPR modelu. Dle TPR a FPR je následně provedeno porovnání jednotlivých modelů. Finální testování je provedeno po finálním výběru modelu způsobem zmíněným v sekci 4.1.

Kapitola 5

Implementace

Pro ucelené pochopení řešení je třeba dopodrobna rozebrat implementaci návrhů zmíněných v analytické části týkajících se především generování umělých dat, detekčního algoritmu a předzpracování vstupu.

5.1 Sběr dat

Pro umělé generovaní je třeba získat rozmanité obrázky základních dílů bez indikace. Toho lze dosáhnou dvěma různými způsoby.

5.1.1 Generování dat co nejpodobnější reálným datům

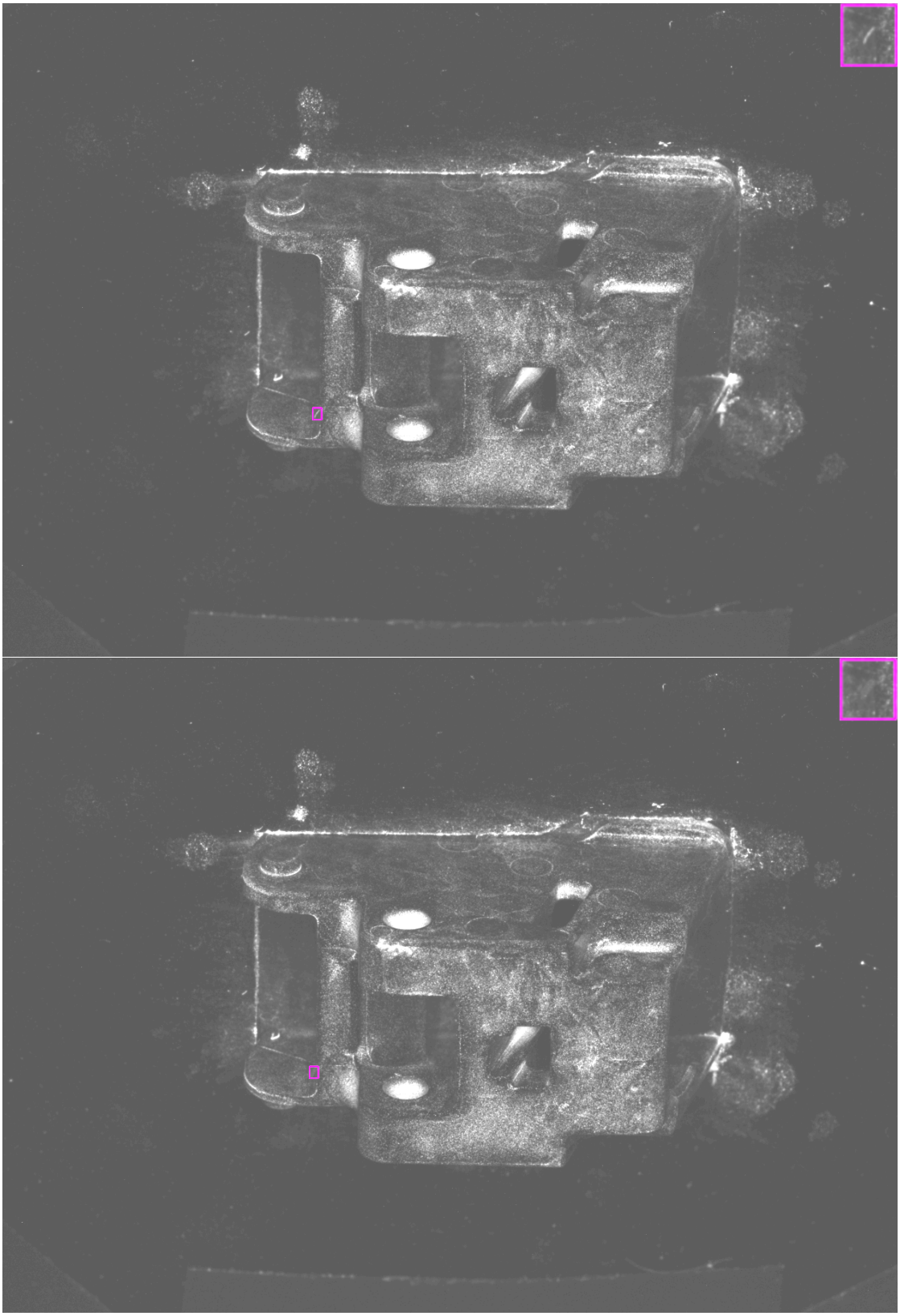
Typ generování zmíněný v bodě 1 sekce 4.2.1 zahlazuje chyby náhodným výběrem z okolí chyby, které není pozadím. Z toho vyplývá, že postup lze použít pouze tehdy, pokud jsou chyby relativně malé a díl má v okolí poměrně jednolitou texturu.

Myšlenka algoritmu je následující:

Algorithm 1 Algoritmus zahlazování

1: $\epsilon_s \leftarrow Maska_{Dilu} \setminus Maska_{Chyby}$
2: **for each** $Bod \in \mathcal{M}aska_{Chyby}$ **do**
3: $\quad Bod \leftarrow$ náhodně vyber z ϵ_s kde vzdálenost od Bod je $\leq \varepsilon_{max}$

Kde $Maska_{Dilu}$ je získána vhodným prahováním a $Maska_{Chyby}$ je už známa z ručních anotací. Vzdálenost omezená ε_{max} je obdélník přípustných bodů o 100 px se středem v daném bodě.



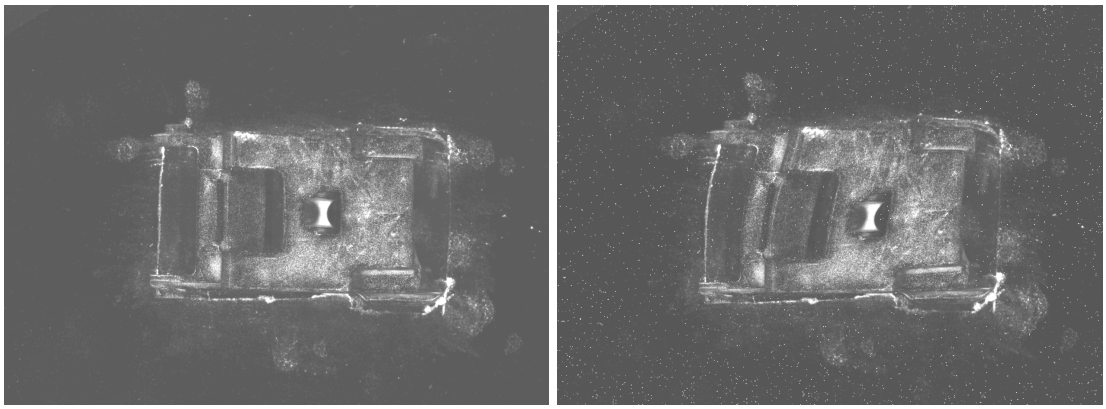
Obrázek 5.1 Nahoře obrázek s vadou, dole obrázek se zahlazenou vadou

5.1.2 Generování pokrývající co největší distribuci

Pro typ generování zmíněný v bodě 2 sekce 4.2.1 se používá nástroj elasticdeform [7] s náhodným rotováním a přeškálováním. Parametry jsou zvoleny následovně.

Tabulka 5.1 Parametry generování

Typ parametru	Hodnota
elasticdeform sigma	náhodně uniformní od 5 do 50
elasticdeform points	náhodně uniformní z hodnot 3, 5, 9
pravděpodobnost rotace	50 %
pravděpodobnost přeškálování	25 %
pravděpodobnost kladného škálování	50 %
maximální rozsah kladného a záporného škálování	0,4 a 0,2



Obrázek 5.2 Vlevo původní obrázek, vpravo obrázek s použitím elasticdeform.

5.2 Anotace dat

Byla uměle vygenerována rozmanitá data snímků bez indikací. Nyní lze do snímků vkládat umělé vady, které lze získat dvěma způsoby.

5.2.1 Generování dat co nejpodobnější reálným datům

V případě bodu 1 sekce 4.2.2 je generování nových dílů s indikacemi prováděno pomocí vkládání indikací z jiných obrázků do obrázků, které žádné neobsahují.

Myšlenka algoritmu je následující.

Algorithm 2 Algoritmus vkládání chyb do obrázku

$ERR_{DB} \leftarrow \{\}$
$IMAGE_{ok} \leftarrow$ obrázek bez indikace
$Exclusion \leftarrow$ prázdný obrázek
for each $Indikace \in Obrazky_{vadne}$ **do**
    $ERR_{DB} \leftarrow ERR_{DB} \cup \{Indikace\}$
$ERR_{used} \leftarrow \{\begin{bmatrix} x & 1 \end{bmatrix} \mid x \in ERR_{DB}\}$
while počet vložených chyb je $\leq ERR_{vloz}$ **do**
    $ERR \leftarrow$náhodně x_0 vyber dle pravděpodobností $P_x = \frac{1/x_1}{\sum_{y \in ERR_{used}} 1/y_1}$ z ERR_{used}
    zvyš u vybraného ERR ERR_1 o 1
    $ERR \leftarrow$ proveď náhodnou augmentaci ERR
    $MASK_{Insert} \leftarrow$ maska bodů kde je distance transform od pozadí v $Maska_{Dilu} \geq$ než největší poloměr v ERR od středu
    $MASK_{Excl} \leftarrow$ maska bodů kde je distance transform od již vložených indikací v $Exclusion \geq$ než největší poloměr v ERR od středu
    $MASK_{Toins} \leftarrow MASK_{Excl} \cap MASK_{Insert}$
    $IMAGE_{ok} \leftarrow$ vyber vhodný bod z $MASK_{Toins}$ pro střed a chybu ERR pomocí něj vlož do obrázku
    $Exclusion \leftarrow$ přidej kruh v daném bodě s poloměrem, který je rovný maximálnímu poloměru v chybě od středu
    $IMAGE_{ok} \leftarrow$ vhodně rozostři hranici mezi vloženou chybou a obrázkem
    vytvoř anotace pro vloženou ERR

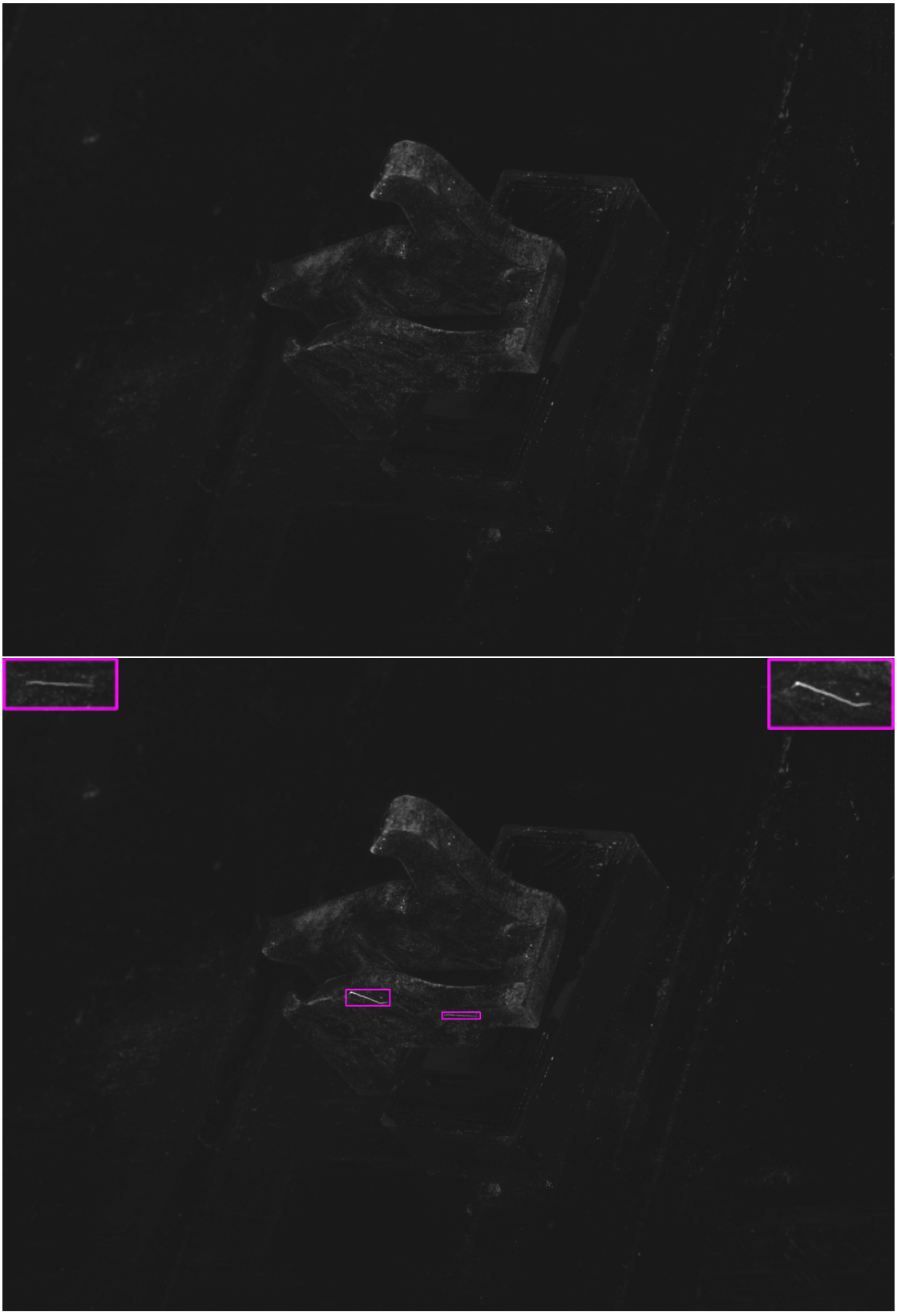
Kde $Maska_{Dilu}$ lze opět získat vhodným prahováním. $Obrazky_{vadne}$ je množina všech obrázků s anotovanými indikacemi. A ERR_{vloz} se volí uniformně od 2 do 6.

$MASK_{Insert}$ slouží k tomu, aby se žádná vada nevložila mimo díl. $MASK_{Excl}$ a $Exclusion$ zaručují, že se vkládané chyby nemohou překrývat. ERR_{used} zde slouží k tomu, aby se zachovala různorodost v jednom obrázku, i když chyb je málo a je jich vkládáno velké množství. To je však spíše ojedinělé pro současný dataset. Vlastnost by však mohla být žádoucí pro další experimenty s tímto typem tvoření nových dat.

Náhodná augmentace ERR se provádí pomocí náhodné rotace a přeškálování. Pravděpodobnosti a rozsahy jsou následující.

Tabulka 5.2 Parametry augmentace

typ parametru	hodnota
pravděpodobnost rotace	50 %
pravděpodobnost přeškálování	25 %
pravděpodobnost kladného škálování	50 %
maximální rozsah kladného a záporného škálování	0,4 a 0,1



Obrázek 5.3 Nahoře obrázek bez vad, dole obrázek s vloženými vadami.

Tvoření datasetu pomocí zahlazování a vkládání indikací

Těmito dvěma způsoby lze rozšířit dataset tak, že:

1. Nejprve jsou ke všem datům s indikacemi vytvořeny obrázky bez indikací způsobem zmíněným v algoritmu 1. Data jsou přidána k současným datům bez indikací.

2. Pak pomocí jsou pomocí algoritmu 2 dorovnána data tak, aby byla vybalancovaná. Snímky, do kterých jsou vkládány indikace, jsou vybírány dle následujícího postupu.

Algorithm 3 Algoritmus výběru předloh pro vkládání

1: **for each** $obr \in \mathcal{O}brazky_{BezIndikaci}$ **do**
2: $\quad OK_{DB} \leftarrow OK_{DB} \cup \{obr\}$
3: $OK_{used} \leftarrow \{\begin{bmatrix} x & 1 \end{bmatrix} \mid x \in OK_{DB}\}$
4: $OK \leftarrow$náhodně x_0 vyber dle pravděpodobností $P_x = \frac{1/x_1}{\sum_{y \in OK_{used}} 1/y_1}$ z OK_{used}
5: zvyš u vybraného OK OK_1 o 1

Kde OK je vybraný obrázek pro vkládání indikací v daném kroku. Postup více vynutí používání různorodých obrázků.

Postup při současném datasetu zvýší počet dat z 955 snímků na 1910 snímků.

5.2.2 Generování pokrývající co největší distribuci

Indikace v datasetu vznikající při metodách NDT mají několik dobrých vlastností:

1. Indikace svým tvarem připomínají vždy čáru, tedy jdou dobře aproximovat cestou.

2. Cesta aproximující indikaci se nikdy nekroutí nebo se nevrací zpět.

3. Cesta aproximující indikaci se téměř nikdy nerozdvojuje.

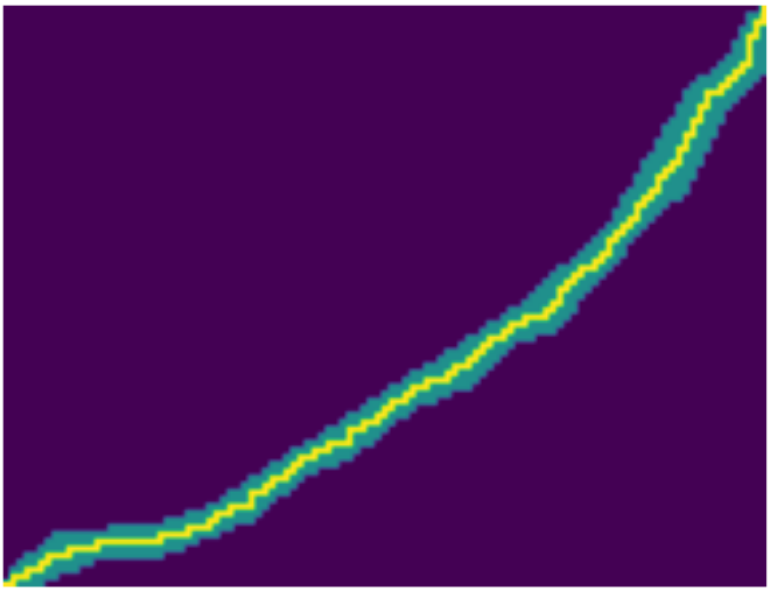
Je třeba chyby aproximovat a následně generovat takové aproximace, které mohou posloužit například k předtrénování modelu.

Aproximace distribuce indikací

První krokem je najít cestu aproximující vadu. Cesta musí být souvislá, jít co nejvíce středem indikace a začínat a končit v nejzazších bodech indikace. Protože běžné algoritmy skeletonizace, jako například [20] a [21], tyto požadavky nemusí vždy nutně splnit, byl navržen následující algoritmus.

Algorithm 4 Algoritmus hledání aroximace vady

1: $MASK \leftarrow$ maska indikace
2: $X, Y \leftarrow \max_{X,Y \in MASK}$ EuclideanDistance(X,Y)
3: $MASK_{inverseDIST} \leftarrow \frac{1}{DistanceTransform(MASK)}$
4: $PATH \leftarrow$ Najdi nejkratší cestu pomocí Dijkstrova algoritmu z X do Y s váhami bodů $MASK_{inverseDIST}$



Obrázek 5.4 Aproximace vady

Výstupem algoritmu je vždy spojitá cesta, rovnou lze tedy určit distribuci délek cest aproximující vady. Pro průběh cest jako takových je třeba udělat určitou aproximaci s předpokladem, že každý krok cesty je nezávislý na předchozích. To nemusí nutně odpovídat realitě, protože při závislých krocích by nebylo možné takovou distribuci reprezentovat s rozumnými nároky na paměť. Jednotlivé kroky jsou reprezentovány jako přírůstky oproti současnému umístění v cestě v polárních souřadnicích. Aproximace distribucí je prováděna následovně.

Algorithm 5 Algoritmus mapování distribuce průběhu cesty

$Xprev, Yprev \leftarrow 0, 0$
$I \leftarrow 0$
for each $X, Y \in \mathcal{C}esta$ **do**
    $rho, phi \leftarrow$ Převeď do polárních souřadnic ($X - Xprev$, $Y - Yprev$)
    $Distibuce_I \leftarrow$ Přidej do distribuce rho, phi
    $Xprev, Yprev \leftarrow X, Y$
    $I \leftarrow I + 1$

Po aproximaci průběhu cesty je třeba aproximovat její šířku. Pro aproximaci distribuce šířky se cesta rozdělí na $N = 10$ dílů. Aproximace se pak provádí následovně.

Algorithm 6 Algoritmus mapování distribuce šířky cesty

$prev \leftarrow 0$
$I \leftarrow 0$
$Vzdalenost_{OdKraje} \leftarrow$ Distance transform masky indikace
for each $SEG \in \mathcal{S}egmenty_{cesty}$ **do**
    $Size \leftarrow$ Pro segment SEG spočítej průměrnou hodnotu bodů v $Vzdalenost_{OdKraje}$ z prostřední 1/2 cesty
    $Distibuce_I \leftarrow$ Přidej do distribuce $(2 * Size) - (2 * Prev)$
    $prev \leftarrow Size$
    $I \leftarrow I + 1$

Nyní lze tyto distribuce odhadnout z dat pomocí Kernel density estimation [22]. S odhadnutými distribucemi lze generovat masky nových aproximací vad následujícím způsobem.

Algorithm 7 Algoritmus tvoření masky nových indikací dle distribuce

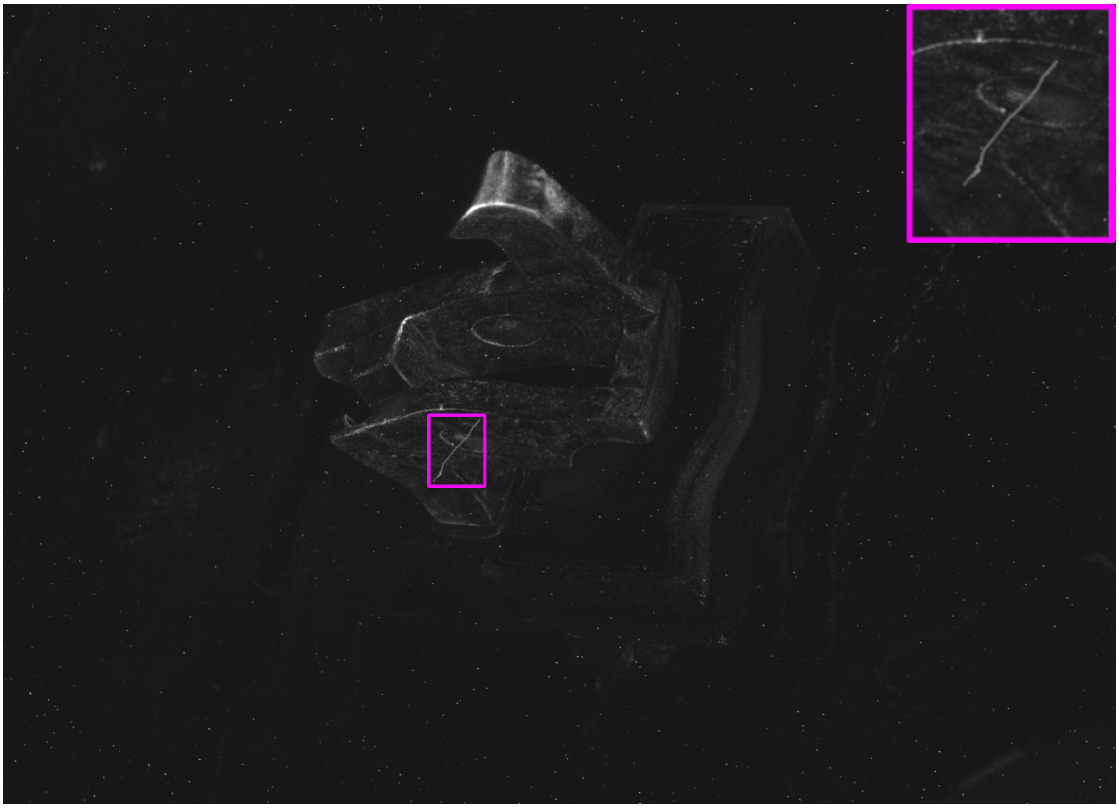
$DISTRdelka \leftarrow$ distribuce délek
$DISTRprubeh \leftarrow$ distribuce průběhu ve všech krocích
$DISTRsirka \leftarrow$ distribuce šířky ve všech krocích
$Delka \leftarrow$ vyber hodnotu z distribuce $DISTRdelka$
$I \leftarrow 0$
$X, Y \leftarrow 0, 0$
$MASKA \leftarrow$Vyplň masku nulami
while $I \leq Delka$ **do**
    $MASKA_{X,Y} \leftarrow 1$
    $rho, phi \leftarrow$ vyber hodnotu z distribuce $DISTRprubeh_I$
    $X, Y \leftarrow X, Y+$ převeď do kartézských souřadnic (rho, phi)
    $I \leftarrow I + 1$
Rozděl cestu v $MASKA$ na N=10 segmentů
$J \leftarrow 0$
for each $SEG \in \mathcal{S}egmenty_{cesty}$ **do**
    $Sirka \leftarrow$ vyber hodnotu z distribuce $DISTRsirka_J$
    Do každého bodu SEG obtiskni čtverec jedniček velikosti $Sirka \times Sirka$ tak, aby se tím nezvětšila délka celkové cesty
    $J \leftarrow J + 1$

Nyní je potřeba přiřadit masce texturu. Pixely z indikací datasetu jsou vybírány pomocí jejich vzdálenosti od okraje následujícím způsobem.

Algorithm 8 Algoritmus přiřazování textury masce

$Vysledek \leftarrow$ obrázek nul velikosti masky
$Indikace \leftarrow$ náhodně vyber indikaci z datasetu, kde pravděpodobnost výběru je přímo úměrná podobnosti velikosti plochy $Maska$ a dané indikace
$DistanceIndikace \leftarrow$ proveď distance trasform $Indikace$, následně jeho hodnoty znormalizuj MIN-MAX normalizací
$DistanceMaska \leftarrow$ proveď distance trasform $Maska$, následně jeho hodnoty znormalizuj MIN-MAX normalizací
for each $Bod \in Maska$ **do**
    $Vysledek_{Bod} \leftarrow$ vyber bod z $Indikace$, který je nepodobnější v $DistanceIndikace$ hodnotě $DistanceMaska_{Bod}$. Pokud je takových bodů více, rozhodni se dle podobnosti MIN-MAX normalizovaných souřadnic

Takto finálně vygenerovaná chyba je vložena do obrázku generovaného postupem v sekci 5.1.2 uvedeným algoritmem 2. Tímto způsobem je vygenerováno 5600 obrázků sloužících k předtrénování modelu.



Obrázek 5.5 Uměle vygenerovaná vada

5.3 Předzpracování dat

Před vstupem do hlavního vyhodnocovacího algoritmu je potřeba zmenšit obrazová data. Základní variantou zmenšování je klasické bilineární zmenšení.

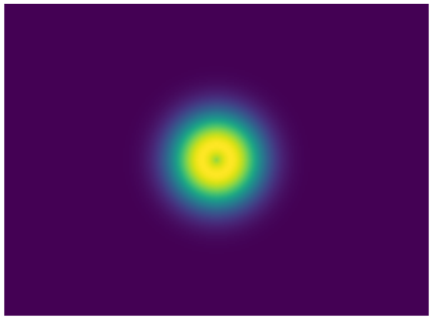
Jinou variantou je použití odříznutí vysokých frekvencí zmíněné v sekci 4.2.3. Jelikož obrázky obsahují poměrně ostré přechody, je toto zmenšení limitováno na 2048×2048 px z původních 4096×3000 px. Při větším snížení rozlišení vznikají touto metodou na obrázku určité artefakty, které by mohly mást vyhodnocovací algoritmus. Finální zmenšení do přijatelné velikosti 1024×1024 px je dosaženo 2×2 Max poolingem. Před oříznutím vysokých frekvencí je na frekvenční reprezentaci ještě aplikován filtr, který potlačuje pozadí oproti vadám. Předpis filtru je následující:

$distMAP_{x,y} \leftarrow$ euklidovská vzdálenost x, y od středu frekvenčního spektra

$MASK_{x,y} \leftarrow \exp \frac{-(distMAP_{x,y}-dist)^2}{\exp w}$

Parametry $dist$ a w jsou nalezeny pomocí grid search z vhodného rozsahu. Jako nejlepší jsou vybrány ty, které co nejvíce po aplikaci filtru na uměle vytvořený obrázek s velkým množstvím indikací maximalizují výraz:

$$\frac{\sum_{X \in pixely_{indikaci}} X}{|pixely_{indikaci}|} - \frac{\sum_{Y \in pixely_{ostatni}} Y}{|pixely_{ostatni}|}$$



Obrázek 5.6 Vizualizace vytvořené filtrační masky pro filtraci frekvenčního spektra.

5.4 Modifikace architektury modelu

Jak již bylo zmíněno v sekcích 4.2.4 a 4.3, model je založený na článku [8]. Segmentační část modelu však má dost značných nevýhod, a proto je v této práci navržen způsob, jak tuto část nahradit modelem U-NET [6].

Propojení U-NETu a klasifikační sítě je realizováno následovně.

1. Místo předposlední vrstvy segmentační CNN uvedené v článku, odebírá klasifikační CNN výstup z posledního enkodér bloku (viz obrázek 3.1) U-NETu.

2. Segmentační maska již nijak nevstupuje do klasifikační CNN. Jediné propojení segmentační masky s klasifikační částí sítě je přes globální AVG a MAX směřující do finální klasifikační vrstvy.

3. Kromě těchto dvou změn zůstává klasifikační část identická.

Propojení má dobrý smysl, jelikož počet kanálů směřující do klasifikační CNN se až na segmentační masku nemění. Navíc U-NET jako takový je vzdáleně podobný klasickému autoenkodéru, který má obvykle právě ve svém středu velmi efektivní reprezentaci vstupu, která se často používá k redukci dimensionality.

5.5 Trénink modelů

Pro zjištění optimálního modelu a generování dat bylo vyzkoušeno několik verzí postupů vytvoření vyhodnocovacího algoritmu. Jednotlivé verze se mezi sebou primárně liší tím:

1. Jak velký vstup přijímá segmentační část sítě.

2. Kolik enkodér bloků má U-NET, a jak jsou tyto bloky velké (kolik obsahují filtrů v konvolučních vrstvách).

3. Jaké zmenšení vstupu je používáno (bilineární nebo frekvenční zmíněné v sekci 5.3)

4. Je model přetrénován na umělém datasetu vygenerovaném postupem uvedeným v sekci 5.2.2

5. Je použito rozšíření trénovacích dat zmíněné v sekci 5.2.1

Tabulka 5.3 Parametry jednotlivých variant navržených modelů.

Název varianty	Velikost vstupu	Enkodér bloky	Zmenšení	Umělá data	Rozšíření dat
Baseline	1024×768	64,128,256,1024	Bilinear	NE	NE
GENMAX	1024×768	64,128,256,1024	Bilinear	NE	ANO
GENMAX_BIG	2048×1472	32,64,128,256,1024	Bilinear	NE	ANO
F_resize	1024×1024	64,128,256,1024	Frekvenční	NE	NE
Adata_pretr	1024×768	64,128,256,1024	Bilinear	ANO	NE

Kapitola 6
Výsledky

Jsou prezentovány jednotlivé výsledky dílčích optimalizací zmíněných v sekci 5.5. Výsledky detekce jsou vizualizovány pomocí matice záměn.

Je popsáno, jak moc je model vysvětlitelný.

Evaluace byla provedena na validačním setu. Jelikož je tento set nevybalancovaný, výsledky byly porovnány na základě TPR a FPR.

6.1 Tabulky úspěšností

Nejprve byla určena přesnost základní varianty Baseline, se kterou jsou porovnávány další dílčí vylepšení procesu.

Tabulka 6.1 Výsledky varianty modelu Baseline.

Typ dílu	TP	FP	TN	FN	TPR	FPR	rozdíl TPR a FPR
m083	23	0	66	8	0,74	0,00	0,74
m072	20	12	60	12	0,62	0,17	0,45
m114	22	5	84	15	0,59	0,06	0,53
m028	4	1	15	0	1,00	0,06	0,94

Varianta GENMAX zaznamenala výrazné zlepšení, a tedy postup sekci 5.2.1 se ukázal jako velmi prospěšný pro fungování modelu.

Tabulka 6.2 Výsledky varianty modelu GENMAX.

Typ dílu	TP	FP	TN	FN	TPR	FPR	Rozdíl TPR a FPR
m083	28	2	64	3	0,90 (+ 0,16)	0,03 (+ 0,3)	0,87 (+ 0,13)
m072	26	15	57	6	0,81 (+ 0,19)	0,21 (+ 0,04)	0,6 (+ 0,15)
m114	30	21	68	7	0,81 (+ 0,22)	0,24 (+ 0,18)	0,57 (+ 0,04)
m028	4	0	16	0	1,00 (+ 0,00)	0,00 (− 0,06)	1,00 (+ 0,06)

Varianta GENMAX_BIG s větší velikostí vstupu také zaznamenala zlepšení. Zvětšování vstupu je tedy prospěšné pro fungování modelu.

Tabulka 6.3 Výsledky varianty modelu GENMAX_BIG.

Typ dílu	TP	FP	TN	FN	TPR	FPR	Rozdíl TPR a FPR
m083	26	2	64	5	0,84 (+ 0,10)	0,03 (+ 0,03)	0,81 (+ 0,07)
m072	26	4	68	6	0,81 (+ 0,19)	0,06 (− 0,11)	0,75 (+ 0,30)
m114	33	18	71	4	0,89 (+ 0,30)	0,20 (+ 0,14)	0,69 (+ 0,16)
m028	4	0	16	0	1,00 (+ 0,00)	0,00 (− 0,06)	1,00 (+ 0,06)

Postup v sekci 5.3 vedl u některých dílů ke zlepšení a u jiných ke zhoršení (viz tabulka 6.4). Je možné, že některé indikace mají významnou část informace zakódovanou ve vysokých frekvencích, a proto tento postup u některých dílů selhává.

Tabulka 6.4 Výsledky varianty modelu F_resize.

Typ dílu	TP	FP	TN	FN	TPR	FPR	Rozdíl TPR a FPR
m083	28	0	66	3	0,90 (+ 0,16)	0,00 (+ 0,00)	0,90 (+ 0,16)
m072	23	4	68	9	0,72 (+ 0,10)	0,06 (− 0,11)	0,66 (+ 0,21)
m114	20	6	83	17	0,54 (− 0,05)	0,07 (+ 0,01)	0,47 (− 0,06)
m028	3	0	16	1	0,75 (− 0,25)	0,00 (− 0,06)	0,75 (− 0,19)

Varianta Adata_pretr s předtrénováním na umělých datech zaznamenala v některých datasetech výrazné zlepšení. Pokud lze data vhodně aproximovat, tak postup v skeci 5.2.2 je prospěšný pro fungování modelu.

Tabulka 6.5 Výsledky verze modelu Adata_pretr.

Typ dílu	TP	FP	TN	FN	TPR	FPR	Rozdíl TPR a FPR
m083	27	1	65	4	0,87 (+ 0,13)	0,02 (+ 0,02)	0,85 (+ 0,11)
m072	30	6	66	2	0,94 (+ 0,32)	0,08 (− 0,09)	0,86 (+ 0,41)
m114	27	9	80	10	0,73 (+ 0,14)	0,10 (+ 0,04)	0,63 (+ 0,10)
m028	4	2	14	0	1,00 (+ 0,00)	0,12 (+ 0,06)	0,88 (− 0,06)

6.2 Vizualizace detekcí

Výsledky jsou prezentovány pro případy detekcí TP, FP, TN, FN na variantě modelu GENMAX_BIG a dílu typu m114. Následně jsou diskutována rozhodnutí modelu. Výstupní maska modelu je zde převedena na bounding boxy pro srozumitelnější vizualizaci.

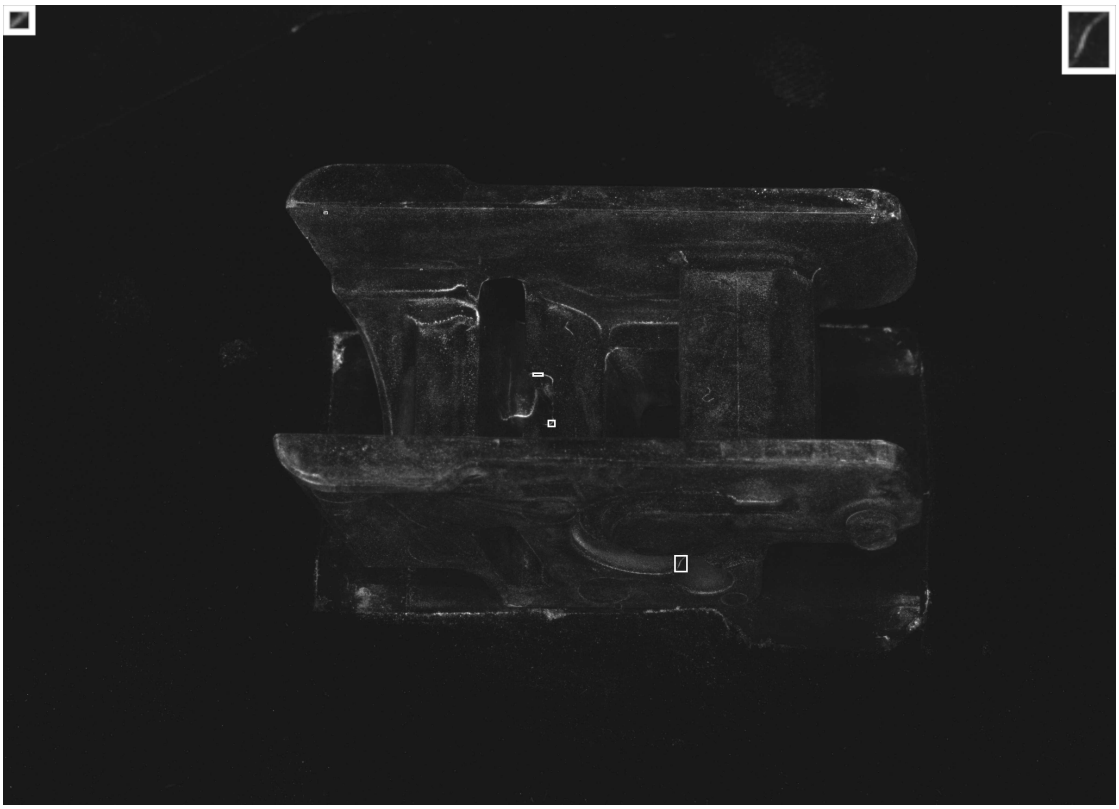
6.2.1 Detekce TP

Model vadu detekoval správně, ale maska zde plně neodpovídá anotaci (viz obrázek 6.1).

Obrázek 6.1 Anotace z dat zobrazeno fialově (nahoře), detekce modelu zobrazeno bíle (dole).

6.2.2 Detekce FP

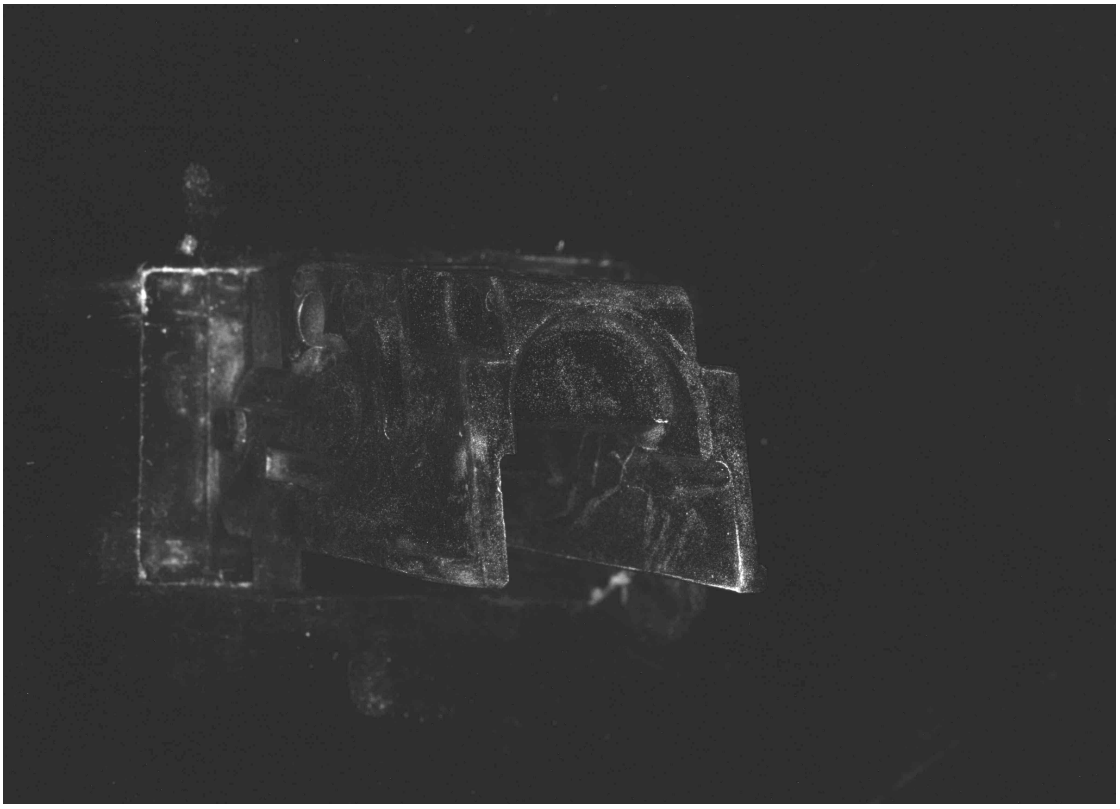
Model špatně detekoval hranu objektu, protože vypadá podobně jako vada (viz obrázek 6.2). Jedná se o nejčastější typ FP.



Obrázek 6.2 FP detekce modelu zobrazeno bíle.

6.2.3 Detekce TN

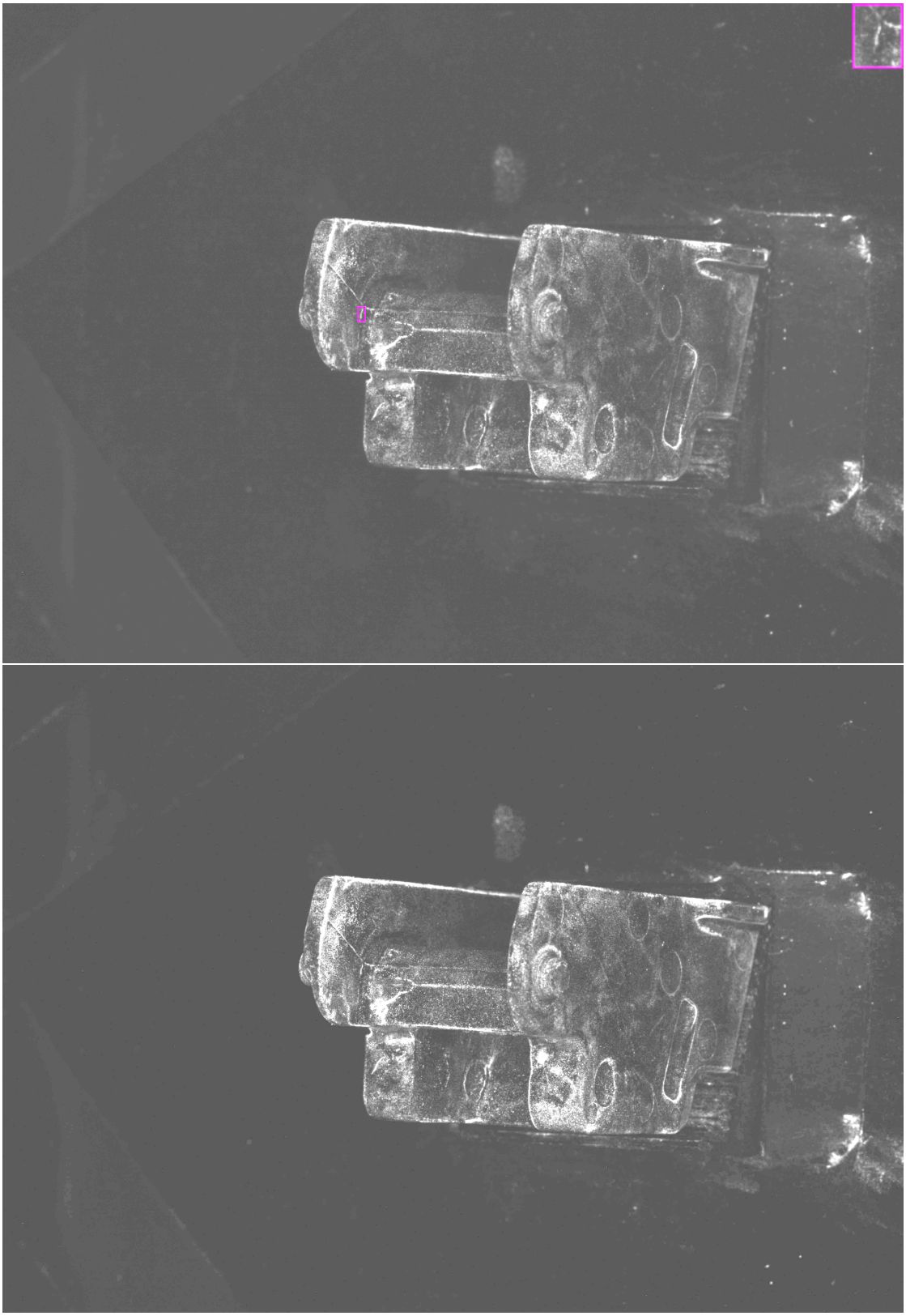
Obrázek obsahuje jak šum, tak výrazné hrany (viz obrázek 6.3). Model správně detekoval snímek jako bezvadný.



Obrázek 6.3 TN detekce modelu (žádná detekce na obrázku není).

6.2.4 Detekce FN

Na obrázku 6.4 je vada velmi málo kontrastní vůči svému okolí. Model ji nedetekoval.



■ **Obrázek 6.4** Anotace z dat zobrazena fialově (nahoře), detekce modelu zobrazena bíle (dole) – žádná není.

6.3 Vysvětlitelnost modelu

Současný model je relativně dobře vysvětlitelný oproti standardnímu klasifikátoru, jelikož jeho výstupem je kromě klasifikace alespoň segmentační maska. Bohužel oproti modelům typu R-CNN se vysvětlitelnost zhoršila, protože segmentační maska nevstupuje nijak výrazně do klasifikační sítě, která odebírá svůj vstup hlavně z prostředku segmentační sítě. To má za následek, byť ve vzácných případech (jelikož tyto sítě se stále trénují společně na stejných datech a klasifikační síť je poměrně malá), že se výstup klasifikace nemusí nutně shodovat se segmentační maskou. Případné řešení problému je zmíněno v sekci Diskuze.

Kapitola 7
Diskuze

Bylo navázáno na práci [1] s tím rozdílem, že se zde řeší problém nad složitějšími daty. Proto byl navržen nový algoritmus detekce založený na síti U-NET a klasifikační síti, který řeší nedostatky algoritmu v původní práci. Také bylo zmíněno, jak lze uměle rozšířit bázi dat, a jak data zpracovávat před vstupem do sítě.

Vůči metodám R-CNN použitých v původní práci se však zhoršila vysvětlitelnost modelu (viz sekce 6.3), jelikož klasifikační část sítě už není příliš závislá na výstupní masce. To má za následek, že se klasifikace nemusí nutně shodovat s tím, co je v segmentační masce, a tak se zhoršuje porozumění modelu člověkem. Problém by mohl být řešen zjištěním, které pixely vstupu nejvíce posouvají vstup sigmoid funkce nad nebo pod nulu, tj. mají velký vliv při klasifikaci. Kdyby se podařilo důležitost pixelů odhalit, mohla by spolu s kombinací segmentační masky poskytnout lepší vhled do rozhodování modelu.

Pro model jako takový by šlo ještě experimentovat s různými úpravami jeho architektury, jako například změnit propojení klasifikační a segmentační části nebo vyměnit klasifikační část za úplně jiný typ sítě.

Místo na vylepšení je také v generování čistě syntetických dat (viz sekce 5.2.2), kde by se distribuce průběhu cesty mohla odhadovat například po částech pomocí pravděpodobnostního stromu, a až v těchto částech předpokládat nezávislost. Také přiřazování textury by mohlo být vylepšeno určitými omezujícími podmínkami pro mapování pixelů, například podívat se do okolí pixelu, do kterého chceme přiřazovat a najít bod, který má okolí podobné.

Dalším zajímavým experimentem by mohlo být natrénovat některý z generativních modelů (např GAN, Variational autoencoder nebo Diffusion modely) na vygenerovaných datech (viz sekce 5.2.2) a z něj teprve generovat trénovací data. Tím by šlo zavést do procesu více náhody a odstranit některé nechtěné systematické biasy.

V poslední řadě by šel vstup modelu zvětšit na původní velikost 4096×3000 px. Jelikož bylo ukázáno, že u některých dílů to zvýšilo přesnost, lze se domnívat, že by tento krok mohl být prospěšný pro lepší klasifikaci, byť za cenu velmi dlouhého trénování modelu.

Kapitola 8
Závěr

V práci byla řešena problematika automatizované detekce indikací vad na dílech testovaných metodami nedestruktivního testování. Práce navazovala na práci [1] s cílem zlepšit detekci na složitějších dílech.

Jako první byla provedena diskuze, zda jsou algoritmy použité v původní práci vhodné pro nové složitější díly. Následně byl navržen nový detekční algoritmus řešící problémy původního algoritmu detekce. Dále byly zkoumány možné způsoby, jak lze rozšířit bázi trénovacích dat pro nový algoritmus.

Pro takové rozšíření byly zvoleny dva způsoby. První spočíval ve vygenerovaní dat tak, aby byla co nejpodobnější těm reálným. Druhý spočíval v tom, aby byla data generována tak, aby pokryla co nejvíce distribuce všech možných vstupů, a to i za cenu možných nepřesností a neshod s reálnými daty.

Dále se práce zabývala vhodným předzpracováním dat před vstupem do vyhodnocovacího algoritmu a vhodností použitého vyhodnocujícího modelu. Byly navrženy a ohodnoceny další varianty vyhodnocovacího modelu pro detekci vad. Detekované vady byly vizualizovány na obrazových datech.

Celý systém byl optimalizován. Cíle práce byly splněny.

Bibliografie

1. ADAM MALEČEK. *Systém pro automatizovanou detekci vad ve spojení s metodami nedestruktivního testování*. 2020. Dostupné také z: https://dspace.cvut.cz/handle/10467/90311.
2. REN, Shaoqing; HE, Kaiming; GIRSHICK, Ross; SUN, Jian. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. Dostupné z arXiv: 1506.01497 [cs.CV].
3. GIRSHICK, Ross. *Fast R-CNN*. 2015. Dostupné z arXiv: 1504.08083 [cs.CV].
4. GIRSHICK, Ross; DONAHUE, Jeff; DARRELL, Trevor; MALIK, Jitendra. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. Dostupné z arXiv: 1311.2524 [cs.CV].
5. MA, Tao; SUN, Zhenguo; ZHANG, Wenzeng; CHEN, Qiang. A machine vision assisted system for fluorescent magnetic particle inspection of railway wheelsets. *AIP Conference Proceedings*. 2016, roč. 1706, č. 1, s. 150003. Dostupné z DOI: 10.1063/1.4940615.
6. RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. Dostupné z arXiv: 1505.04597 [cs.CV].
7. TULDER, Gijs van. *elasticdeform: Elastic deformations for N-dimensional images*. Zenodo, 2022. Ver. v0.5.0. Dostupné také z: https://doi.org/10.5281/zenodo.7102577.
8. BOŽIČ, Jakob; TABERNIK, Domen; SKOČAJ, Danijel. End-to-end training of a two-stage neural network for defect detection. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021, s. 5619–5626. Dostupné z DOI: 10.1109/ICPR48806.2021.9412092.
9. SHIN, Hoo-Chang; TENENHOLTZ, Neil A; ROGERS, Jameson K; SCHWARZ, Christopher G; SENJEM, Matthew L; GUNTER, Jeffrey L; ANDRIOLE, Katherine; MICHALSKI, Mark. *Medical Image Synthesis for Data Augmentation and Anonymization using Generative Adversarial Networks*. 2018. Dostupné z arXiv: 1807.10225 [cs.CV].
10. BOIKOV, Aleksei; PAYOR, Vladimir; SAVELEV, Roman; KOLESNIKOV, Alexandr. Synthetic Data Generation for Steel Defect Detection and Classification Using Deep Learning. *Symmetry*. 2021, roč. 13, č. 7. ISSN 2073-8994. Dostupné z DOI: 10.3390/sym13071176.
11. BOSNAR, Lovro; SARIC, Doria; DUTTA, Siddhartha; WEIBEL, Thomas; RAUHUT, Markus; HAGEN, Hans; GOSPODNETIC, Petra. Image Synthesis Pipeline for Surface Inspection. 2020.

12. SCHMEDEMANN, Ole; BAASS, Melvin; SCHOEPFLIN, Daniel; SCHÜPPSTUHL, Thorsten. Procedural synthetic training data generation for AI-based defect detection in industrial surface inspection. *Procedia CIRP*. 2022, roč. 107, s. 1101–1106. ISSN 2212-8271. Dostupné z DOI: https://doi.org/10.1016/j.procir.2022.05.115. Leading manufacturing systems transformation – Proceedings of the 55th CIRP Conference on Manufacturing Systems 2022.

13. GUTIERREZ, Pierre; LUSCHKOVA, Maria; CORDIER, Antoine; SHUKOR, Mustafa; SCHAPPERT, Mona; DAHMEN, Tim. Synthetic training data generation for deep learning based quality inspection. In: KOMURO, Takashi; SHIMIZU, Tsuyoshi (ed.). *Fifteenth International Conference on Quality Control by Artificial Vision*. SPIE, 2021. Dostupné z DOI: 10.1117/12.2586824.

14. HASELMANN, Matthias; GRUBER, Dieter. Supervised Machine Learning Based Surface Inspection by Synthetizing Artificial Defects. In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2017, s. 390–395. Dostupné z DOI: 10.1109/ICMLA.2017.0-130.

15. HASELMANN, M.; GRUBER, D. P. Pixel-Wise Defect Detection by CNNs without Manually Labeled Training Data. *Applied Artificial Intelligence*. 2019, roč. 33, č. 6, s. 548–566. Dostupné z DOI: 10.1080/08839514.2019.1583862.

16. RIPPEL, Oren; SNOEK, Jasper; ADAMS, Ryan P. *Spectral Representations for Convolutional Neural Networks*. 2015. Dostupné z arXiv: 1506.03767 [stat.ML].

17. FELZENSZWALB, Pedro F.; HUTTENLOCHER, Daniel P. *Efficient graph-based image segmentation - International Journal of Computer Vision*. Kluwer Academic Publishers, [b.r.]. Dostupné také z: https://link.springer.com/article/10.1023/b:visi.0000022288.19776.77#citeas.

18. UIJLINGS, Jasper; SANDE, K.; GEVERS, T.; SMEULDERS, A.W.M. Selective Search for Object Recognition. *International Journal of Computer Vision*. 2013, roč. 104, s. 154–171. Dostupné z DOI: 10.1007/s11263-013-0620-5.

19. KINGMA, Diederik P.; BA, Jimmy. *Adam: A Method for Stochastic Optimization*. 2017. Dostupné z arXiv: 1412.6980 [cs.LG].

20. JAIN, Himanshu; KUMAR, Archana Praveen. *A Sequential Thinning Algorithm For Multi-Dimensional Binary Patterns*. 2017. Dostupné z arXiv: 1710.03025 [cs.CV].

21. LEE, D. Medial Axis Transformation of a Planar Shape. *IEEE transactions on pattern analysis and machine intelligence*. 1982, roč. 4, s. 363–9. Dostupné z DOI: 10.1109/TPAMI.1982.4767267.

22. WIKIPEDIA CONTRIBUTORS. *Kernel density estimation — Wikipedia, The Free Encyclopedia*. 2023. Dostupné také z: https://en.wikipedia.org/w/index.php?title=Kernel_density_estimation&oldid=1136297214. [Online; accessed 9-May-2023].

Obsah přiloženého média

- `readme.txt` ..... stručný popis zdrojových kódů implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF